Universal Lattice Decoding: Principle and Recent Advances

Wai Ho Mow[1][2]

Dept. of EEE,

Hong Kong Univ. of Science and Technology,

Clear Water Bay,

Hong Kong S.A.R., CHINA.

Fax: (852) 2358-1485

w.mow@ieee.org

KEY WORDS

basis reduction

nearest lattice point

sphere decoder

maximum likelihood detection

optimal performance

complexity

Summary

The idea of formulating the detection of a lattice-type modulation, such as M-PAM and M-QAM, transmitted over a linear channel as the so-called universal lattice decoding problem dates back to at least the early 1990s. The applications of such lattice decoders have proliferated in the last few years because of the growing importance of some linear channel models, such as multiple-antenna fading channels and multi-user CDMA channels. The principle of universal lattice decoding can trace its roots back to the theory and algorithms developed for solving the shortest/closest lattice vector problem for integer programming and cryptoanalysis applications. In this semi-tutorial paper, such a principle as well as some related recent advances will be reviewed and extended. It will be shown that the lattice basis reduction algorithm of Lenstra, Lenstra and Lovász can significantly improve the performance of suboptimal lattice decoders, such as the zero-forcing and VBLAST detectors. In addition, new implementation of the optimal lattice decoder that is particularly efficient at moderate to high signal-to-noise ratios will also be presented.

[1] Correspondence to: Wai Ho Mow, Dept. of EEE, Hong Kong Univ. of Science and Technology, Clear Water Bay, Hong Kong S.A.R., CHINA.

[2] Contract/grant sponsor: Hong Kong RGC; Contract/grant number: HKUST6246/02E

Contract/grant sponsor: NSFC/RGC; Contract/grant number: N_HKUST617/02 (No.60218001).

1 Introduction

The idea of formulating the detection of a lattice-type modulation transmitted over a linear channel as the so-called universal lattice decoding problem dates back to at least the early 1990s [1]. The resultant universal lattice decoder is particularly attractive for bandwidth efficent modulations, such as M-PAM and M-QAM, due to its various desirable properties [1], [2] including:

- its decoding complexity is independent of the modulation alphabet size M;
- its performance is nearly optimal, especially for large M;
- its average complexity is quadratic, as the signal-to-noise ratio tends to infinity.

The applications of such lattice decoders have proliferated in the past decade because of the growing importance of some linear channel models, such as intersymbol interference channels [1], [2], fading channels [3], [4], uncoded and space-time block coded multiple-antenna channels [5], multiuser CDMA channels [6], dispersive multiple-antenna channels [7] and their combinations.

The principle of universal lattice decoding can trace its roots back to the theory and algorithms developed for solving the shortest/closest lattice vector problem for integer programming and cryptoanalysis applications. The closest (lattice) vector problem (CVP) (also called the nearest lattice point problem) is a class of nearest neighbor searches or closest-point queries, in which the solution set to be searched consists of all the points in a lattice. Very efficient algorithms for solving the CVP [8, chap. 20], [9], [10] have been derived for a special class of lattices — the root lattices, which are generated by the root system of certain Lie algebras. These algorithms are important for implementing low-complexity lattice quantizers and coding schemes for Gaussian channels. Nonetheless, these algorithms cannot be generalized to solve the universal problem of decoding arbitrary lattices.

A general solution for the CVP was proposed by Kannan [11], [12] and was originally developed for solving some integer programming problems. Kannan was mainly interested in deriving theoretical results on the worst-case complexity. His algorithm does not lead to an efficient practical solution to the CVP. However the underlying ideas are simple and powerful, consisting of two steps:

Step 1: For the given lattice, find a "short" and fairly "orthogonal" basis, called the reduced basis.

Step 2: Enumerate all lattice points falling inside a certain sphere centered at the query point so as to identify the closest lattice point.

The procedure that transforms a lattice basis into a reduced one is known as the basis reduction algorithm, while the one achieving the second step is called the enumeration algorithm. We shall see that most efficient universal lattice decoders known to date are based on similar ideas.

Lattice algorithms related to the CVP have also been widely applied in cryptoanalysis applications, where the algorithmic complexity is of both practical and theoretical significance. In cryptoanalysis, it is well-known that choosing a good lattice basis is important to finding a good nearby lattice point, which can in turn speed up the closest lattice point search significantly.

In this paper, the principle of some important lattice algorithms related to the CVP and their recent advances will be reviewed and extended. After a brief introduction to the necessary notation and preliminary backgrounds on lattices in Section 2, we shall discuss the main theory and detailed implementation of the lattice basis reduction algorithm of Lenstra, Lenstra and Lovász (LLL) in Section 3. Two algorithms of Babai for finding a nearby lattice point based on the LLL reduction will be introduced in Section 4. Their relationships with the well-known zero-forcing and VBLAST detectors in communications will also be clarified in the same section. Next, the extension of the results and algorithms discussed in Sections 2 to 4 to the so-called complex lattices will be discussed in Section 5. In Section 6, the derivation and efficient implementation of the Pohst-Fincke enumeration algorithm and its variants will be presented in details. To demonstrate the efficiency of the introduced lattice algorithms as suboptimal or optimal lattice decoders, the results of some simulation experiments will be presented and discussed in Section 7. Finally, Section 8 contains the concluding summary.

2 Notation and Preliminaries on Lattices

Let m and n be two positive integers with $n \leq m$. A subset L of $\mathbf{R}^m$ is called a lattice of dimension n if there exist n linearly independent m-dimensional vectors $b_1, \cdots, b_n \in \mathbf{R}^m$ such that

$$L = L(B) = \{\eta_1 b_1 + \cdots + \eta_n b_n : \eta_i \in \mathbf{Z}\},$$

where $B = [b_1, \cdots, b_n]$ is a $m \times n$ matrix. The set of column vectors $b_1, \cdots, b_n$ and the matrix B are said to be the basis and the basis matrix of L, respectively. In words, a lattice is an integer linear combination of the basis vectors. Geometrically, the determinant $\det(L)$ of a lattice L is defined as the common content of the parallelepipeds spanned by any lattice bases. In general, a lattice has many possible bases but it has the same determinant. In the case of a full-rank square basis matrix, the determinant of the lattice $L(B)$ satisfies $\det(L) = |\det(B)|$. Hence the following inequality, called Hadamard's inequality, is natural from a geometric point of view:

$$\|b_1\| \cdots \|b_n\| \geq \det(L) \qquad (1)$$

with equality holds if and only if $b_1, \cdots, b_n$ are mutually orthogonal.

Let us consider the Gram-Schmidt orthogonalization of a given lattice basis. The m-dimensional orthogonalization vectors $b_1^*, \cdots, b_n^*$ and the real numbers μ_{ij}, for $1 \leq j < i \leq n$, are defined recursively by

$$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{ij} b_j^*, \qquad (2)$$

$$\mu_{ij} = \frac{(b_j^*)^T b_i}{\|b_j^*\|^2}, \qquad (3)$$

where $(\cdot)^T$ denotes the matrix transpose of $(\cdot)$. The orthogonal vectors $b_1^*, \cdots, b_n^*$ obtained in this way depend on the ordering of $b_1, \cdots, b_n$. Note that, for $1 \leq i \leq n$, $b_1^*, \cdots, b_i^*$ and $b_1, \cdots, b_i$ span the same vector subspace. By defining $\mu_{ii} = 1$, we have

$$b_i = \sum_{j=1}^{i} \mu_{ij} b_j^*, \qquad (4)$$

for $1 \leq i \leq n$. It is obvious that

$$\det(L(B)) = \prod_{i=1}^{n} \|b_i^*\|, \qquad (5)$$

whether the basis matrix B is square or not. Equation (4) can be expressed in matrix form as

$$B = B^*[\mu_{ij}]^T, \qquad (6)$$

where $B^* = [b_1^*, \cdots, b_n^*]$ and $[\mu_{ij}]$ is a $n \times n$ lower triangular matrix with all diagonal elements equal to 1.

By letting $u_i = b_i^*/\|b_i^*\|$ and $b_i(j) = \mu_{ij}\|b_j^*\|$, for $1 \leq j \leq i \leq n$, we also have

$$b_i = \sum_{j=1}^{i} b_i(j) u_j. \qquad (7)$$

Note that $b_i(i) = \|b_i^*\|$ for all i. The matrix form equivalent of (7)

$$B = U[b_i(j)]^T \qquad (8)$$

is in fact the QR factorization of B, where $U = [u_1, \cdots, u_n]$ gives an orthonormal basis for the column space of B. We note that the upper triangular matrix $[b_i(j)]^T$ is actually the Cholesky factor of $B^T B$ satisfying

$$B^T B = [b_i(j)][b_i(j)]^T. \qquad (9)$$

The latter also follows immediately from (8).

Conceptually, it is useful to interpret the effect of U in (8) as a change of the original m-dimensional Cartesian coordinate system, under which the basis vectors $b_1, \cdots, b_n$ are specified, into a new one with n unit axis vectors $u_1, \cdots, u_n$ respectively. Under the new coordinate system, the coordinates of the i-th basis vector are listed in the i-th row of the $n \times n$ lower triangular matrix $[b_i(j)]$:

$$\begin{pmatrix} b_1(1) & 0 & \cdots & 0 \\ b_2(1) & b_2(2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ b_n(1) & b_n(2) & \cdots & b_n(n) \end{pmatrix}$$

This lower triangular representation of the basis matrix was used by Kannan in [12].

Finally, we describe two important operations on vectors — projecting and lifting. Projecting a vector b onto the hyperplane through the origin with normal vector a yields $b - \frac{a^T b}{\|a\|^2} a$, which can also be interpreted as the projection of b perpendicular to a. Suppose that a is a non-zero vector in L, and L_a is the projection of L perpendicular to a. If b_a is a vector in L_a, there is a unique vector $\hat{b}$ in L such that $\hat{b}$ projects onto b_a and $-\frac{\|a\|}{2} < \frac{a^T \hat{b}}{\|a\|} \leq \frac{\|a\|}{2}$. The process is called lifting b_a to $\hat{b}$. In fact, $\hat{b} = b_a - \eta a$, where $\eta = \text{round}(\frac{a^T b}{\|a\|^2})$ is the integer nearest to $\frac{a^T b}{\|a\|^2}$. Figure 1 illustrates a two-dimensional example of projecting b perpendicular to a producing b_a and then lifting b_a to $\hat{b}$. Note that transforming a vector by projecting and then lifting guarantees that the resultant vector cannot be too long.

3 Lattice Basis Reduction

The concept of lattice basis reduction was proposed more than a century ago. The early work on the topic was formulated in terms of quadratic forms, instead of lattices. Reduced bases have some nice properties, which usually means that they consist of "short" and fairly "orthogonal" vectors. The definition of basis reduction is not unique. One of the most important definitions was given by Minkowski in 1890s. A basis is Minkowski-reduced if, for $i = 1, \cdots, n$, b_i is a shortest lattice vector that can be extended to a basis with $(b_1, \cdots, b_{i-1})$. In simple words, each basis vector in a Minkowski-reduced basis has to be as short as possible. The definition of a Minkowski-reduced basis is of fundamental importance in the geometry of numbers [13].

Basis reduction is naturally associated with the problem of finding the shortest non-zero lattice vector — the shortest vector problem (SVP). The SVP in the case of L_∞-norm is known to be NP-hard [14]. But it is still unclear whether the SVP in the case of Euclidean norm is NP-hard or not.

In 1982, Lenstra, Lenstra and Lovász [15] (LLL) achieved a breakthrough by constructing the celebrated LLL reduction algorithm, which can produce the so-called LLL-reduced basis from any given lattice basis in polynomial time and thereby approximating the shortest (non-zero) lattice vector up to a factor of $2^{(n-1)/2}$. Based on their algorithm, more efficient algorithms for solving the SVP and the CVP, achieving the Hermite normal form and the Minkowski reduction have been developed. In fact, their algorithm has given rise to efficient solutions to a variety of research problems, including integer programming [16], [12], finding irreducible factors of polynomials [17], minimal polynomials of algebraic numbers [18], simultaneous diophantine approximation [15], ellipsoid method in linear programming [19], attacks on knapsack-based crypto-systems [20], [21], disproof of Mertens' century-old conjecture in number theory. All these applications were made possible by the LLL-reduction algorithm.

References [15] and [22] have described the LLL-reduction algorithm with proof of its correctess and polynomial complexity. In the following, we shall rederive the reduction algorithm based on our own interpretation.

3.1 Size Reduction

As transforming vectors by projecting and then lifting can usually result in reasonably short vectors, such operations may be used to convert a lattice basis into a shorter one.

Consider the lower triangular representation $[b_i(j)]$ of a given basis $b_1, \cdots, b_n$. If, for $j < i$, the j-th coordinate of b_i is greater in magnitude than half of the length of b_j^* (i.e. $|b_i(j)| > \frac{b_j(j)}{2}$, or equivalently, $|\mu_{ij}| > \frac{1}{2}$), we can always reduce the length of b_i by projecting it onto the subspace spanned by $b_1^*, \cdots, b_j^*$ and then lifting it. The resultant vector $\bar{b}_i = b_i - \eta b_j$, for some integer η, must satisfy the condition that $|\bar{b}_i(j)| \leq \frac{b_j(j)}{2}$. Notice that the new basis $b_1, \cdots, b_{i-1}, \bar{b}_i, b_{i+1}, \cdots, b_n$ spans the same lattice as the original basis $b_1, \cdots, b_n$. The process can be repeated $\frac{n(n-1)}{2}$ times for all $1 \leq j < i \leq n$. The resultant basis consisting of shorter basis vectors is said to be size-reduced.

In summary, a basis $b_1, \cdots, b_n$ is size-reduced if $|\mu_{ij}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$. Recall that $\mu_{ii} = 1$ and $\mu_{ij} = 0$ for $j > i$. Any basis can be converted into a size-reduced basis by the following Procedure Size_Reduce, which requires $O(n^3)$ operations to execute.

```
Procedure Size_Reduce($B$)

1. For $i = 1$ to $n$ do
2.     For $j = i - 1$ downto 1 do
3.         If $|\mu_{ij}| > \frac{1}{2}$ do
4.             $\eta :=$ round($\mu_{ij}$); $b_i := b_i - \eta b_j$.
5.             For $k = 1$ to $j - 1$ do $\mu_{ik} := \mu_{ik} - \eta\mu_{jk}$.
6.             $\mu_{ij} := \mu_{ij} - \eta$.
7.         Endif.
8.     Endfor.
9. Endfor.
10. Return $b_i$'s and $\mu_{ij}$'s.
```

Note that making $|\mu_{ij}| \leq \frac{1}{2}$ will change the values of $\mu_{i1}, \cdots, \mu_{i,j-1}$. Therefore the elements of matrix $[\mu_{ij}]$ must be processed from right to left. Finally, we remark that $b_1^*, \cdots, b_n^*$ obtained from the orthogonalization process of the new basis is the same as before. This is obvious if we interpret the procedure as projecting and lifting operations of the basis vectors.

3.2 LLL Reduction

Given a basis and a specific orthogonalization $b_1^*, \cdots, b_n^*$, we can always apply Procedure Size_Reduce to transform it into a "better" basis. One may then ask how to find a "better" orthogonalization of a given basis.

For any orthogonalization of a given lattice, the product of the lengths of b_i^*'s must be a constant according to (5). Intuitively, it is desirable that the lengths of b_i^*'s are distributed as even as possible so that the basis, after size reduction, appears to be "shorter". Lovász [22] observed that typically the shorter vectors among $b_1^*, \cdots, b_n^*$ are at the end of the sequence. So it is desirable to make the orthogonalization sequence $b_1(1) = \|b_1^*\|, \cdots, b_n(n) = \|b_n^*\|$ lexicographically as small as possible.

For $1 \leq j \leq i \leq n$, define $b(i,j)$ as the projection of b_i onto the orthogonal complement of the subspace spanned by $u_1, \cdots, u_{j-1}$, or mathematically,

$$b(i,j) = \sum_{k=j}^{i} b_i(k) u_k.$$

For some $i < n$, consider the lengths of the projections of b_i and b_{i+1} onto $u_i, \cdots, u_n$, i.e., $b(i,i) = b_i(i)u_i$ and

$b(i+1,i) = b_{i+1}(i)u_i + b_{i+1}(i+1)u_{i+1}$. If $b(i,i)$ is longer than $b(i+1,i)$ (i.e., $b_i(i) > \|b(i+1,i)\|$), we can always swap b_i and b_{i+1} to get a lexicographically smaller orthogonalization sequence $b_1(1), \cdots, b_{i-1}(i-1), \|b(i+1,i)\|, \cdots, b_n(n)$. Hence a better orthogonalization is resulted.

After swapping some basis vectors, the basis may not be size-reduced any more. We can then apply Procedure Size_Reduce again to shorten the basis without changing the orthogonalization sequence. The two processes, namely, finding a better basis via size reduction for a given orthogonalization sequence and finding a better orthogonalization sequence via swapping basis vectors for a given basis, can be iterated until no further improvement is achievable. This is in essence the LLL-reduction algorithm in its most preliminary form.

```
Algorithm LLL_Reduction($b_1, \cdots, b_n$)
Step 1: Size-reduce the given basis.
Step 2: Check if there exists any $i$ such that $\delta \cdot \|b(i,i)\|^2 > \|b(i+1,i)\|^2$. If found, swap $b_i$ and $b_{i+1}$, update the orthogonalization sequence, and go to step 1. Otherwise, stop.
```

Note that a weaker test $\delta \cdot \|b(i,i)\|^2 > \|b(i+1,i)\|^2$ is used in step 2 instead of $\|b(i,i)\|^2 > \|b(i+1,i)\|^2$ to achieve faster convergence. (The convergence of the algorithm will be proved in the next subsection). The coefficient δ is chosen arbitrarily and may be replaced by any number less than 1. (The value of δ was originally chosen as $\frac{3}{4}$ in [15].) This suggests the following definition of LLL-reduced bases.

Definition 1 *A basis $b_1, \cdots, b_n$ of a lattice is LLL-reduced if it is size-reduced and, for $1 \leq i < n$,*

$$\delta \cdot \|b(i,i)\|^2 \leq \|b(i+1,i)\|^2. \tag{10}$$

3.2.1 Convergence Analysis

Like any iterative algorithm, we need to guarantee its termination. Let $\bar{b}_i^*$ and $\bar{b}_{i+1}^*$ be the updated versions of b_i^* and b_{i+1}^*, respectively, after swapping b_i and b_{i+1}. Note that all orthogonalization vectors except the i-th and $(i+1)$-th ones are unchanged since they lie in the orthogonal complement of the subspace spanned by $b(i,i)$ and $b(i+1,i)$. If we do swapping in step 2, there is a positive number $\alpha < \sqrt{\delta}$ such that $\|b(i+1,i)\| = \alpha\|b(i,i)\|$. Thus we have $\bar{b}_i^* = b(i+1,i)$, or $\|\bar{b}_i^*\| = \alpha\|b(i,i)\| = \alpha\|b_i^*\|$. Also, by (5), $\|\bar{b}_i^*\|\|\bar{b}_{i+1}^*\| = \|b_i^*\|\|b_{i+1}^*\|$. Then

$$\prod_{k=1}^{i} \|\bar{b}_k^*\| = \alpha \prod_{k=1}^{i} \|b_k^*\|,$$

and for all $1 \leq j < n$ and $j \neq i$,

$$\prod_{k=1}^{j} \|\bar{b}_k^*\| = \prod_{k=1}^{j} \|b_k^*\|.$$

This suggests the definition of the positive function

$$D(b_1, \cdots, b_n) = \prod_{j=1}^{n} \prod_{k=1}^{j} \|b_k^*\|^2 = \prod_{k=1}^{n} \|b_k^*\|^{2(n-k)}. \tag{11}$$

The function can be interpreted as a measure of achieved reducedness of the given basis because

$$\log(D(b_1, \cdots, b_n)) = \sum_{k=1}^{n} 2(n-k)\log(\|b_k^*\|),$$

which is a weighted sum of the log length of the orthogonalization vectors. (Note that a basis with a nice orthogonalization sequence need not be a short basis, but we can always get a nice basis by making it weakly reduced.) In consistent with our previous discussion, the smaller the function D is, the (lexicographically) smaller the orthogonalization sequence is. The value of D is decreased after each swapping in step 2 due to the multiplication of α. So the algorithm must terminate, otherwise D will tend to zero which is impossible. In fact, one can establish upper and lower bounds for the value of D, as in [15].

3.2.2 Properties of LLL-Reduced Bases

Some nice properties of the LLL-reduced bases are summarized in the following theorem, whose proof can be found in [22].

Theorem 1 (Lovász) *Let $b_1, \cdots, b_n$ be a LLL-reduced basis of a lattice L. Denote $\lambda(L)$ as the length of the shortest vector in L. Then*

1. $\|b_1\| \le 2^{(n-1)/2}\lambda(L)$;
2. $\|b_1\| \le 2^{(n-1)/4}\det(L)^{1/n}$;
3. $\|b_1\| \cdots \|b_n\| \le 2^{n(n-1)/4}\det(L)$.

Parts 1 and 2 of the theorem guarantee that the LLL-reduced basis includes a reasonably short vector while part 3 ensures a reasonably "orthogonal" basis. In addition to the nice properties of the reduced basis, it is the efficiency which enables its reduction algorithm to have important applications in various areas.

3.2.3 Implementation Details

We now focus on the detailed procedure to obtain a LLL-reduced basis. After swapping b_k and b_{k-1} in step 2, exactly two of the orthogonalization vectors b_k^* and b_{k-1}^* are changed. Therefore, only the μ_{ij}'s associated with b_k, b_{k-1}, b_k^* and b_{k-1}^* need to be updated. For the ease of understanding, we enclose in boxes these elements of the matrix

$$\begin{pmatrix} \mu_{11} & 0 & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \boxed{\mu_{k-1,1}} & \boxed{\cdots} & \boxed{\mu_{k-1,k-1}} & 0 & \ddots & \vdots \\ \boxed{\mu_{k,1}} & \boxed{\cdots} & \boxed{\mu_{k,k-1}} & \boxed{\mu_{k,k}} & \ddots & \vdots \\ \vdots & \ddots & \boxed{\vdots} & \boxed{\vdots} & \ddots & 0 \\ \mu_{n1} & \cdots & \boxed{\mu_{n,k-1}} & \boxed{\mu_{n,k}} & \cdots & \mu_{nn} \end{pmatrix}.$$

Denote $\bar{b}_i, \bar{b}_i^*$ and $\bar{\mu}_{ij}$ as the updated values of b_i, b_i^* and μ_{ij} respectively. Then the following update formulas can be derived in a straightforward manner [15].

$$\begin{array}{rcll} \bar{b}_{k-1} & = & b_k & \\ \bar{b}_k & = & b_{k-1} & \\ \bar{b}_{k-1}^* & = & b_k^* + \mu_{k,k-1} b_{k-1}^* & \\ \bar{b}_k^* & = & b_{k-1}^* - \bar{\mu}_{k,k-1}\bar{b}_{k-1}^* & \\ \bar{\mu}_{k,k-1} & = & \mu_{k,k-1}\frac{\|b_{k-1}^*\|^2}{\|\bar{b}_{k-1}^*\|^2} & \\ \bar{\mu}_{i,k-1} & = & \mu_{i,k-1}\bar{\mu}_{k,k-1} + \mu_{ik}\frac{\|b_k^*\|^2}{\|\bar{b}_{k-1}^*\|^2} & (k < i \le n) \\ \bar{\mu}_{ik} & = & \mu_{i,k-1} - \mu_{ik}\mu_{k,k-1} & (k < i \le n) \\ \bar{\mu}_{k-1,j} & = & \mu_{kj} & (1 \le j \le k-2) \\ \bar{\mu}_{kj} & = & \mu_{k-1,j} & (1 \le j \le k-2) \end{array}$$

Recall that $\mu_{ii} = 1$ for all i, so the diagonal elements need not be updated. Also, we simply need to swap μ_{kj}

and $\mu_{k-1,j}$ for $1 \leq j \leq k-2$, since $b_1^*, \cdots, b_{k-2}^*$ are unchanged. This also implies that only $b_{k-1}, \cdots, b_n$ need to be considered to obtain a size-reduced basis in step 1. Besides, in step 2, we have to find an i such that $\delta \cdot \|b(i,i)\|^2 > \|b(i+1,i)\|^2$. One straightforward way to implement this is to use a counter k (with initial value 2) to keep track of the dimension of the sublattice that has already been LLL-reduced. If the test is failed (i.e. $b_1, \cdots, b_{k-1}$ must be a LLL-reduced basis of the sublattice they span), the counter is incremented. Otherwise, we swap b_{k-1} and b_k and decrement the counter. In this way, the counter will count up and down during the run. However, the counter will reach $n+1$ sooner or later as the algorithm must terminate. The process can be thought of as extending the dimension of the sublattice which is LLL-reduced in the current basis representation. Clearly, during the pass when the counter value is k, we only need the first k basis vectors $b_1, \cdots, b_k$ to be size-reduced. Since $b_1, \cdots, b_{k-1}$ have been size-reduced before the counter value is incremented or decremented to k, only b_k need to be considered in step 1. With a closer look, further simplification for step 1 is possible. The test in step 2 requires the value of $\mu_{k,k-1}$ only. Thus we can make $|\mu_{k,k-1}| \leq \frac{1}{2}$ first, and process $\mu_{k,k-2}, \cdots, \mu_{k,1}$ only when the counter is incremented. With this observations, the LLL-reduction algorithm [15] can be constructed readily.

Procedure LLL_Reduce(B)

```
1.  Do Gram-Schmidt orthogonalization process to obtain $\mu_{ij}$'s and $\beta_i$'s.
2.  $k := 2$.
3.  Make, if necessary, $|\mu_{k,k-1}| \leq \frac{1}{2}$ and update $b_k$ and $\mu_{k1}, \cdots, \mu_{k,k-1}$.
4.  If $\beta_k < (\delta - |\mu_{k,k-1}|^2)\beta_{k-1}$ do
5.      Swap $b_{k-1}$ and $b_k$, swap $\mu_{k-1,1}, \cdots, \mu_{k-1,k-2}$ and $\mu_{k,1}, \cdots, \mu_{k,k-2}$, and update $\beta_{k-1}$, $\beta_k$, $\mu_{k+1,k-1}, \cdots, \mu_{n,k-1}$ and $\mu_{k+1,k}, \cdots, \mu_{n,k}$.
6.      If $k > 2$ do $k := k-1$.
7.  Else
8.      For $j = k-2$ downto 1 do make, if necessary, $|\mu_{kj}| \leq \frac{1}{2}$ and update $b_k$ and $\mu_{k1}, \cdots, \mu_{k,j}$.
9.      If $k \neq n$ do $k := k+1$; else goto line 12.
10. Endif.
11. Goto line 3.
12. Return $b_i$'s, $\beta_i$'s and $\mu_{ij}$'s.
```

(Here, β_i stores the value of $\|b_i^*\|^2$ for $1 \leq i \leq n$.)

It is noteworthy that Procedure LLL_Reduce does not require the orthogonal vectors b_i^*'s and only b_i's, μ_{ij}'s and β_i's need to be stored. To obtain μ_{ij}'s and β_i's without involving b_i^*'s, instead of applying (2) and (3), the Gram-Schmidt orthogonalization process in line 1 can be performed according to the following $O(n^2m)$-complexity procedure [23]:

Procedure Orthogonalize(B)

```
1. For $i = 1$ to $n$ do
2.     For $j = 1$ to $i - 1$ do
3.         $\mu_{ij} := (b_i^T b_j - \sum_{k=1}^{j-1} \mu_{jk}\mu_{ik}\beta_k)/\beta_j$.
4.     Endfor.
5.     $\beta_i := \|b_i\|^2 - \sum_{k=1}^{i-1} |\mu_{ik}|^2 \beta_k$.
6. Endfor.
7. Return $\beta_i$'s and $\mu_{ij}$'s.
```

Alternatively, one can obtain $[b_i(j)]^T$ as a Cholesky factor of $B^T B$ and then set $\beta_i = b_i^2(i)$ and $\mu_{ij} = b_i(j)/b_j(j)$. This approach was used by Fincke and Pohst [24] (and also by Viterbo and his co-workers [3], [4]).

3.2.4 Complexity analysis

We now compute the time complexity of Procedure LLL_Reduce. The Gram-Schmidt orthogonalization process in line 1 needs $\mathrm{O}(n^3)$ operations. Each execution of line 5 decreases the value of function D, defined in (11), by multiplying a factor of $\delta = \frac{3}{4}$, say. Let D_0 be the initial value of D. Following the analysis in [15], it can be proved that after j passes,

$$1 \leq \left(\frac{3}{4}\right)^j D_0 \leq \left(\frac{3}{4}\right)^j \beta^{n(n-1)/2},$$

where β is the maximum squared length of the given basis vectors. Hence, we have $j \leq \frac{3}{2} n(n-1)\log(\beta)$ and the number of times we pass through lines 5 and 6 is $\mathrm{O}(n^2)$. As the test in line 4 cannot be succeeded n times more than it is failed (otherwise $k = n$ and the procedure terminates), the number of times passing through lines 8 and 9 is also $\mathrm{O}(n^2)$. Thus each of the lines 3 to 11 is executed $\mathrm{O}(n^2)$ times. Each execution of line 3, line 5 and line 8 take $\mathrm{O}(n)$, $\mathrm{O}(n)$ and $\mathrm{O}(nm)$ operations respectively. Therefore, the overall complexity of Procedure LLL_Reduce is $\mathrm{O}(n^3 m)$ operations.

4 Finding a Nearby Lattice Point

Intuitively, it is expected that finding a lattice point nearby a given query point could be much simpler than finding the closest lattice point. Babai [25], [26] proved that given a LLL-reduced basis, a nearby lattice point that is closest, within a factor exponential in n, to the query point can be found by two simple polynomial-time algoirhtms, called Procedure Rounding Off and Procedure Nearest Plane, respectively.

Denote the query vector $q = B\vartheta = \vartheta_1 b_1 + \cdots + \vartheta_n b_n \in \mathbf{R}^m$, where $\vartheta \in \mathbf{R}^n$ is a n-dimensional column vector. Procedure Rounding Off simply finds a nearby lattice point by rounding ϑ_i's to their nearest integers. Namely, the nearby lattice point found is

$$B \cdot \mathrm{round}(\vartheta) = \mathrm{round}(\vartheta_1) b_1 + \cdots + \mathrm{round}(\vartheta_n) b_n,$$

which is clearly a valid lattice point in $L(B)$. We observe that the procedure is in fact equivalent to a classical detector in communications.

It is well-known that the zero-forcing (ZF) detector takes the received output vector q and detects the transmitted input vector as round$(B^\dagger q)$, where $B^\dagger = (B^T B)^{-1} B^T$ is the pseudo-inverse of B with $m \geq n$ such that $B^\dagger B = I_n$. Thus we have

$$\text{round}(B^\dagger q) = \text{round}(B^\dagger B \vartheta) = \text{round}(\vartheta),$$

which implies that Procedure Rounding Off and the ZF detector are algorithmically equivalent. Note that since the basis matrix B corresponds to a (noiseless) channel operator that linearly transforms the transmitted input vector into the received output vector, the goal of any detector to recover the channel input vector, rather than the channel output vector. Therefore, the output of the ZF detector is round(ϑ), instead of $B \cdot \text{round}(\vartheta)$.

Given q as the input, the ZF algorithm has a complexity of $O(nm)$ operations, which is dominated by the matrix vector product $B^\dagger q$. Here, we assume that the nearby lattice point problem has to be solved many times for a given lattice so that the pseudo-inverse matrix $B^\dagger$ can be pre-computed once only with a preprocessing complexity of $O(n^2 m)$ operations. The ZF procedure can be summarized as below:

```
**Procedure** Zero_Forcing($q$, $B^\dagger$)
1. Return round($B^\dagger q$).
```

Next, let us denote the query point $q = B^* \zeta = \zeta_1 b_1^* + \cdots + \zeta_n b_n^* \in \mathbf{R}^m$, where $\zeta \in \mathbf{R}^n$ is a n-dimensional column vector. Procedure Nearest Plane introduced by Babai [26] can be stated, for our purpose, as follows:

```
**Procedure** Nearest_Plane($q$, $(B^*)^\dagger$, $[\mu_{ij}]$)
1. $\zeta := (B^*)^\dagger q$.
2. $\eta_n :=$ round($\zeta_n$).
3. For $j = n-1$ downto 1 do
4.     $\eta_j :=$ round($\zeta_j - \sum_{i=j+1}^{n} \eta_i \mu_{ij}$).
5. Endfor.
6. Return $\eta$.
```

Note that the procedure gives the same output whether the basis is size-reduced (i.e., $|\mu_{ij}| \leq \frac{1}{2}$) or not.

As lines 2 to 5 involve $O(n^2)$ operations, Procedure Nearest Plane has a complexity of $O(nm)$, dominated by the matrix vector product $(B^*)^\dagger q$ in line 1. The input parameters μ_{ij}'s and $(B^*)^\dagger$ can be precomputed with a preprocessing complexity of $O(n^2 m)$.

We note that Procedure Nearest_Plane is equivalent to a specific type of decision feedback equalization (DFE) algorithm in communications, called the VBLAST nulling and cancellation detector [27], or simply VBLAST detector. A VBLAST detector distinguishes itself from other forms of DFE by adopting a detection order (i.e., $\eta_n, \cdots, \eta_1$, in our notation) in accordance with the descending order of signal-to-noise ratios (SNR) of different elements in a received vector.

Denote the $m \times n$ basis matrix of the lattice L' by $[b'_1, \cdots, b'_n] = (B^\dagger)^T$, where $L' = L((B^\dagger)^T)$ is called the dual lattice of $L(B)$. The VBLAST detection order is achieved by ordering (or re-indexing) the basis vectors such that $\|b'_n\| \leq \cdots \leq \|b'_1\|$. In other words, the detection order ensures that the transmitted vector element corresponding to the shortest dual basis vector is to be detected first, and so on. It is interesting to note that the VBLAST detection ordering as a preprocessing step for finding a nearby lattice point actually coincides with a preprocessing step proposed by Fincke and Pohst [24] for finding the closest lattice point. In fact, Fincke and Pohst also suggested that the dual lattice should be reduced first. Refer to Section 6.2 for further details.

It is noteworthy that although the two aforementioned algorithms for finding a nearby lattice point have been applied in communications in the form of ZF and VBLAST detectors, their full power as (suboptimal) universal lattice detectors have not been fully realized because the lattice basis was never reduced. As can be seen from Babai's proofs in [26], the assumption of a LLL-reduced basis is crucial to the goodness of the nearby lattice point found by the two algorithms. A first step in this direction has recently been taken by Yao and Wornell [28]. They extended the famous Gauss reduction algorithm for 2-dimensional real lattices to the so-called complex lattices (to be discussed in the next section) and showed that the symbol error rate performance of both the ZF and the nearest plane algorithms with the complex Gauss reduction preprocessing is within 3dB in SNR from the optimal maximum likelihood detection (MLD) performance. Note that the Gauss reduction is identical to the 2-dimensional LLL reduction with $\delta = 1$.

Finally, we should point out that VBLAST detectors are also applicable to some non-lattice decoding problems, such as those with MPSK modulation, in which lattice basis reduction is not well-defined. In Section 6, it will be clear that efficient algorithms for finding a nearby lattice point can also lead to a better algorithm for finding the closest lattice point as well.

5 Extension to Complex Lattices

Due to the use of the passband channel models, many communications detection problems are most naturally formulated in terms of complex lattices when lattice-type modulation schemes are used. If the modulation scheme involves both in-phase and quadrature-phase components (such as QPSK and QAM), the resultant (complex) lattice is a complex integer linear combination of some complex-valued basis vectors. Mathematically, a n-dimensional complex lattice $L \subset \mathbf{C}^m$ is defined as

$$L = \{\eta_1 b_1 + \cdots + \eta_n b_n : \eta_i \in \mathbf{Z} + \mathbf{Z}\sqrt{-1}\},$$

where $b_1, \cdots, b_n \in \mathbf{C}^m$ are n linearly independent m-dimensional complex basis vectors. It is noteworthy that we have simply replaced the real vector space $\mathbf{R}^m$ by the complex vector space $\mathbf{C}^m$ and the integer set $\mathbf{Z}$ by the complex integer set $\mathbf{Z} + \mathbf{Z}\sqrt{-1}$, respectively, in the conventional definition of lattices.

As previously mentioned, Yao and Wornell [28] have successfully extended the famous Gauss reduction algorithm for 2-dimensional complex lattices. Since the complex number field is well known to be an extension of the real number field, one may wonder how easy it is, if possible at all, to extend the general theory on lattices from the real to the complex case. In fact, the part of the theory reviewed up to here has already been extended in a careful manner in Sections 2, 3 and 4. In particular, the algorithms, the procedures and the complexity analysis presented therein are valid, as long as the operations involved are replaced by their complex arithmetics counterparts. When interpreting the aforementioned results for complex lattices, there are a few points deserve special attention:

- $(\cdot)^T$ becomes the matrix complex conjugate transpose operator.
- $|\cdot|$ refers to the magnitude of the possibly complex-valued argument. In particular, $|\mu_{ij}|^2$ should not be confused with μ_{ij}^2.
- The real and imaginary parts returned by the rounding operator round$(\cdot)$ are the integers nearest to the real and imaginary parts of the possibly complex-valued argument respectively.
- In (3), the role of b_j* and b_i are not exchangeable in the complex case, unlike the real case.

In fact, since a n-dimensional complex lattice is isomorphic to a $2n$-dimensional real lattice, every decoding problem that has a complex lattice formulation can also be re-formulated as a real lattice decoding problem. This re-formulation approach has been suggested, for example, in [1] and [5] and has now become a standard approach. However, working directly on the complex lattice can result in decoding algorithms with lower complexity, because the exploitation of the complex lattice structure allows the lattice dimension involved to be only half of that of the equivalent real lattice. The complexity advantage is particularly significant for high-dimensional complex lattices. However, conventional lattice algorithms and their complex counterparts are generally inequivalent, even when they are applied to equivalent lattices. In the Section 7, we shall compare their BER performances by computer simulation for a typical communications detection problem.

6 Finding the Closest Lattice Point

A straightforward method to find the closest lattice point is to enumerate all lattice points falling inside a sphere centered at the query point so as to identify the closest lattice point in the Euclidean metric. To avoid enumerating an unnecessarily large number of points, it is important to determine a reasonably small radius of the sphere. In the following subsections, the proper choice of the radius will be discussed first. Next, we shall consider an efficient enumeration algorithm originally developed by Pohst [29] and later elaborated by

Pohst and Fincke in [24]. After that, we shall present some important improvements and an efficient implementation of the improved Pohst-Fincke algorithm.

6.1 Choice of Initial Radius

The initial radius of the enumeration sphere has to be carefully chosen. If the radius is too small, there is no lattice point inside the sphere and the enumeration process has to be re-started again with a re-selected radius. If the radius is too large, there are too many lattice points inside the sphere resulting in an unnecessarily large enumeration complexity. To avoid enumerating an unnecessarily large number of points, it is important to determine a sufficiently small radius of the sphere which is sufficiently large to contain at least one lattice point.

One suggestion from [6] is to use the Rogers upper bound on covering radius [8], which requires the knowledge about the dimension and determinant of the lattice.

An alternative upper bound on the covering radius is

$$\frac{1}{2}\left(\sum_{k=1}^{n}\|b_k^*\|^2\right)^{1/2},$$

which follows from proposition 4.2 in [12]. Although the values of $\|b_k^*\|$'s are required for calculating the bound, they are typically required by other lattice algorithms and do not induce additional complexity. In fact, the nearby lattice point returned by Procedure Nearest_Plane, called the Babai point in [30], always satisfies this bound. If it is available, a better choice of the initial radius is $\|b-q\|$, where q and b denotes the query point and the Babai point respectively.

6.2 Enumerating Lattice Points in a Sphere

Let $q = \vartheta_1 b_1 + \cdots + \vartheta_n b_n \in \mathbf{R}^m$ be the query point. If a lattice point $a = \eta_1 b_1 + \cdots + \eta_n b_n \in L(B)$ is inside the sphere of radius r centered at q, it satisfies the sphere constraint $\|a - q\| \leq r$. The enumeration problem is to determine all valid combinations of $\eta_1, \cdots, \eta_n$ under the sphere constraint, which can be expressed in terms of $b_1^*, \cdots, b_n^*$ as:

$$\sum_{i=1}^{n}(\sum_{k=i}^{n}(\eta_k - \vartheta_k)\mu_{k,i})^2\|b_i^*\|^2 \leq r^2.$$

This suggests a recursive enumeration algorithm based on the following relationships:

$$r_n = r, \tag{12}$$

$$|\eta_n - \vartheta_n| \leq \frac{r_n}{\|b_n^*\|}, \tag{13}$$

and for $i = n-1, n-2, \cdots, 1$,

$$r_i = (r_{i+1}^2 - |\sum_{i+1}^{n}(\eta_k - \vartheta_k)\mu_{k,i+1}|^2\|b_i^*\|^2)^{1/2}, \tag{14}$$

$$|\eta_i + (\sum_{k=i+1}^{n}(\eta_k - \vartheta_i)\mu_{k,i})| \leq \frac{r_i}{\|b_i^*\|}. \tag{15}$$

The algorithm recursively divides an i-dimensional enumeration problem with radius r_i into $(\lfloor \frac{2r_i}{\|b_i^*\|} \rfloor + 1)$ $(i-1)$-dimensional similar problems with radii r_{i-1}'s. Eventually, the actual enumeration process occurs in many one-dimensional lattices. Geometrically speaking, the $(i-1)$-dimensional problems are similar to the original i-dimensional problem because, after fixing the values of $\eta_i, \cdots, \eta_n$ and projecting onto the subspace spanned by $b_1^*, \cdots, b_{i-1}^*$, the i-dimensional sphere becomes a $(i-1)$-dimensional sphere. The number of points to be enumerated is exactly those vectors shorter than r. Thus this method is efficient in the sense that it enumerates no invalid lattice points outside the sphere.

Fincke and Pohst [24] originally presented the algorithm for finding the shortest lattice vector with the sphere centered at the origin. The above enumeration algorithm was adapted for a sphere centered at the given query point. Unlike the derivation in [24] which is based on the Cholesky factorization of $B^T B$, the above derivation has the advantage that the geometric meaning of the variables involved is more obvious.

Inspired by the method of Dieter [31] and Knuth [32], Fincke and Pohst proposed a preprocessing procedure to speed up the enumeration algorithm. Recall that $[b_1', \cdots, b_n'] = (B^\dagger)^T$ is the $m \times n$ basis matrix of the dual lattice of $L(B)$. From [31], we have the inequality

$$|(a-q)^T b_i'| \leq \|a-q\| \cdot \|b_i'\|,$$

which implies

$$|\eta_i - \vartheta_i| \leq r\|b_i'\|. \tag{16}$$

To limit the search range of η_i, it is desirable to have a smaller value of $\|b_i'\|$, especially in the early stages of the enumeration (i.e., for large values of i). Based on these arguments, Fincke and Pohst [24] suggested two preprocessing steps, as follows:

Step 1: Use a reduction algorithm to obtain a "more orthogonal" basis for the dual lattice.

Step 2: Reorder the indices of the basis vectors $b_1, \cdots, b_n$ such that the lengths of the basis vectors of the dual lattice are in descending order, i.e., $\|b_1'\| \geq \cdots \geq \|b_n'\|$.

Since the enumeration algorithm recursively updates the values of η_i's, from $i = n$ down to 1 (like traversing a n-level multi-branch tree), steps 1 and 2 reduce the range of values of η_i's and hence the number of updating operations. The simulation results in [24] support that the above preprocessing steps can achieve a significant reduction in the overall complexity, especially when a LLL-reduction algorithm is used in step 1.

In the communications literature, the Pohst-Fincke enumeration algorithm without the LLL-reduction preprocessing is often called the sphere decoding algorithm, following the nomenclature in [4].

6.3 Some Improvements and Implementation Details

From (12) and (15), it can be observed that the range of η_i to be considered is inversely proportional to $\|b_i^*\|$. Thus a good orthogonalization of the lattice basis, for which the sequence $\|b_1^*\|, \cdots, \|b_n^*\|$ is lexicographically small, is desired. In other words, the basis of the given lattice, instead of its dual, should be reduced since (13) and (15) in general give a much tighter bound than (16).

We thus propose to reduce complexity by replacing the two preprocessing steps of Fincke and Pohst by the following one [1]:

Step 1': LLL-reduce the basis of the given lattice to obtain a lexicographically small orthogonalization sequence.

Unlike the original preprocessing steps which require matrix inversion, LLL reduction, sorting and permutation, the proposed preprocessing step involves LLL reduction alone and is both simpler and more effective.

Another improvement can be obtained by updating the values of $r_1, \cdots, r_n$ with r' replacing r whenever a lattice point b with $r' = \|b - q\| < r$ is enumerated. As suggested by Mow in [1], to avoid an unnecessarily large amount of updating operations, we can enumerate the values of η_i from its mid-value to its upper bound and then from its mid-value to its lower bound, instead of from the lower bound to the upper bound. In this way, the short vectors are likely to be enumerated first. Based on a similar observation, Schnorr and Euchner [33] suggested an even better enumeration order, which enumerates the value of η_i in the order of increasing distance from the mid-value. As pointed out by Agrell et al. [30], the first lattice point enumerated using the Schnorr-Euchner ordering is the Babai point. Thus the choice of the initial radius is naturally set according to the Babai point, as suggested in Section 6.1, but without an explicit execution of the nearest plane algorithm and the associated complexity. Recently, Chan and Lee [34] has rediscovered the Schnorr-Euchner ordering and their simulation results have demonstrated that impressive complexity reduction can be achieved when applied to 4-transmit and 4-receive antenna systems with 64-QAM. We shall present an efficient way to implement the Schnorr-Euchner ordering scheme. The key idea is to map the sequence $0, 1, 2, 3, 4, \cdots$ into $0, 1, -1, 2, -2, \cdots$ if the mid-value is rounded down, and into $0, -1, 1, -2, 2, \cdots$ otherwise. Our implementation is apparently simpler than those in [33], [30] and [34] since no sorting or multiplication is required.

In [1] and [2], Mow suggested that the average complexity can be reduced by adding a simple stopping test for detecting early if the closest lattice point has been found. Let $\gamma = \gamma(L)$ denote the packing radius of the lattice L (i.e., half the length of the shortest lattice vector). If an enumerated lattice point is found to be at a distance less than γ from the query point (i.e., the stopping test is satisfied), it is clearly a nearest lattice point and thus the enumeration process can be terminated right away. In lattice decoding applications, the query point is a noisy version of a lattice point. At a sufficiently large SNR, most of the query points are located very close to the original lattice point. Therefore, a nearby lattice point, such as the Babai point, is likely to be the nearest point as well. It was suggested in [2] that the Babai point is first found by applying the nearest plane algorithm and if it passes the stopping test, the whole enumeration process is skipped. As the computational cost of running the stopping test is very low, we apply the test whenever a new lattice point known to be currently the nearest is found.

Mow [2] argued that as the SNR tends to infinity, the average decoding complexity becomes quadratic for $m = n$ (or in general, $O(nm)$), namely, the complexity of the nearest plane algorithm with preprocessing. This result on the asymptotic average complexity of a universal lattice decoder is apparently consistent with the recent theoretical analysis of Hassibi and Vikalo [35] (see Figure 2 therein). The result remains valid for our implementation of the enumeration algorithm here, because the complexity of the Pohst-Fincke enumeration algorithm with the Schnorr-Euchner ordering up to the first enumerated lattice point (i.e., the Babai point) is also $O(nm)$, ignoring the preprocessing complexity.

The following procedure implements the improved Pohst-Fincke algorithm for finding the closest lattice point:

```
Procedure Closest_Point($q$, $B$)
1.  (Preprocessing) Find unimodular matrix $T$ such that $B := BT$ is LLL-reduced and compute $B^\dagger$, $\mu_{ij}$'s and $\beta_i$'s.
2.  $\vartheta := B^\dagger q$.
3.  For $i = 1$ to $n$ do $INC_i := 0$; $DEV_i := 0$.
4.  $RMIN := \infty$; $i := n$; $R_i := \infty$; $U_i := -\vartheta_i$; $Z := (R_i/\beta_i)^{1/2}$; $UB_i := \text{floor}(Z - U_i)$; $LB_i := \text{ceil}(-Z - U_i)$; $S_i := \text{sign}(-U_i - \eta_i)$.
5.  While $i \leq n$ do
6.      If $INC_i$ is even do
7.          $DEV_i := -DEV_i$.
8.      Else
9.          $DEV_i := S_i - DEV_i$.
10.     Endif.
11.     $\eta_i := \text{round}(-U_i) + DEV_i$; $INC_i := INC_i + 1$.
12.     If $\eta_i < LB_i$ or $\eta_i > UB_i$ do
13.         $i := i + 1$.
14.     Elseif $i \neq 1$ do
15.         $i := i - 1$; $R_i := R_{i+1} - \beta_{i+1}(\eta_{i+1} + U_{i+1})^2$; $U_i := -\vartheta_i + \sum_{k=i+1}^{n}(\eta_k - \vartheta_k)\mu_{ik}$; $Z := (R_i/\beta_i)^{1/2}$; $UB_i := \text{floor}(Z - U_i)$; $LB_i := \text{ceil}(-Z - U_i)$; $S_i := \text{sign}(-U_i - \eta_i)$; $INC_i := 0$; $DEV_i := 0$.
16.     Else
17.         $RX := R_n - R_1 + \beta_i(\eta_1 + U_1)^2$.
18.         If $RX < RMIN$ do
19.             $\eta := [\eta_1, \cdots, \eta_n]^T$; $RMIN := RX$.
20.             If $RMIN < \gamma$ do goto line 27.
21.             If $RMIN < 0.9R_n$ do
22.                 For $k = 1$ to $n$ do $R_k := R_k - R_n + RX$.
23.             Endif.
24.         Endif.
25.     Endif.
26. Endwhile.
27. Return $T\eta$.
```

In the above procedure, β_i stores $\|b_i^*\|^2$, and R_i stores r_i^2. $RMIN$ and RX hold the latest minimum squared distance and the currently found squared distance respectively. The function sign($\cdot$) returns the sign of the input argument, and it returns 1 even if the input argument is 0. ceil($\cdot$) and floor($\cdot$) are the rounding-up and rounding-down functions respectively. INC_i simply increments from 0 on until η_i exceeds one of the bounds UB_i or LB_i. DEV_i represents the actual deviation of η_i from the middle value round$(-U_i)$ in the Schnorr-Euchner strategy, which is implemented through lines 6 to 10 as well as the initialization of both INC_i and DEV_i to 0. Line 1 can be done once by preprocessing if the lattice remains the same. The procedure output $T\eta$ while the closest lattice point is $BT\eta$. If the LLL reduction is not performed, we may simply regard T as an identity matrix. Note that the factor 0.9 in line 21 is arbitrary, which ensures the R_i's is not updated too frequently, and may be replaced by any reasonable value slightly less than 1. Finally, the value ∞ should be implemented as a very large positive constant value.

Since γ is equal to half the length of the shortest lattice vector, it can be found by a variant of the enumeration algorithm. The key idea is to set the query point as a lattice point and ensure that the algorithm excludes the input point in the enumeration process. If it is necessary to keep the preprocessing complexity small, we may replace γ in line 20 by an easy-to-compute lower bound on the packing radius. A useful lower bound for this purpose is

$$\gamma \geq \frac{1}{2} \min_k \|b_k^*\|.$$

To avoid a very loose bound, it is desirable that the lattice basis from which $\|b_k^*\|$'s are obtained is LLL-reduced.

Finally, we note that although Procedure Closet_Point can always find a closest lattice point, it may sometimes return an invalid transmitted vector. As all practical modulation schemes have a finite symbol alphabet, the set of valid transmitted vectors are located inside a certain finite region of the infinite lattice. If the closest lattice point is outside the finite region, the so-called boundary error is resulted. As pointed out by Viterbo and Boutros [4], for important cubic-shaped modulation schemes (such as PAM and QAM), it is easy to incorporate the alphabet constraint by restricting the range of every vector component (i.e., η_i in our notation). This simple modification totally eliminates the occurrence of boundary errors leading to an exact MLD algorithm. The modified enumeration region is in general not spherical, and might be empty even if the initial radius is chosen according to the Babai point. In the latter case, it is necessary to restart the enumeration process with a larger initial radius chosen according to some heuristics. In addition, to enable the cubic-shaped alphabet constraint to be easily incorporated into the enumeration algorithm, the original lattice basis must be used. It means that the MLD modification is incompatible with the complexity reduction technique of applying the LLL reduction in the preprocessing phase.

In [2], Mow demonstrated by simulation that the performance degradation due to boundary errors is insignificant and vanishes quickly as the alphabet size increases. If the performance degradation is unacceptable, we can lower the average complexity of the lattice MLD by taking advantages of the fact that the boundary errors rarely occur and are detectable. Namely, Procedure Closet_Point with LLL reduction can be used to enumerate lattice points in a spherical region as usual, and the modified enumeration region using the original lattice basis will be considered only if the closest lattice point found is invalid. In this way, the resultant algorithm, while achieving the optimal MLD performance, can also take advantages of the LLL reduction preprocessing for complexity reduction.

7 Simulation Experiments

In this section, the performance and complexity of various lattice decoding algorithms introduced in Sections 4 to 6 are evaluated and compared by computer simulation. The elements in the $m \times n$ basis matrix B are assumed to be independently and identically distributed complex Gaussian random variables with zero mean and unit variance. It corresponds to the channel gain matrix in a typical multiple antenna communication system with n-transmit and m-receive antennas. The i-th element η_i of the transmitted input vector η represents the information symbol taken from a fixed modulation alphabet to be transmitted by the i-th antenna. Without loss of generality, the modulation alphabet is defined as $\mathbf{Z}_M = \{0, 1, \cdots, M-1\}$ for M-PAM, and as $\mathbf{Z}_N + \mathbf{Z}_N\sqrt{-1}$ for M-QAM with squared integer $M = N^2$. The query point (corresponding to the received output vector) is the transmitted input vector transformed by the channel matrix and corrupted by an additive noise vector w, or in symbols,

$$q = B\eta + w.$$

The elements of the noise vector w are independently and identically distributed white complex Gaussian random variables with zero mean and unit variance, independent of the channel gain coefficients B_{ij}'s.

A 2-transmit 3-receive antenna system with 4-PAM is first considered. Although the basis matrix is complex-valued, η is a 2-dimensional integer vector. Thus the system gives rise to a 2-dimensional real lattice in a 6-dimensional vector space. Figure 2 shows the BER performance of the ZF detector, the Babai nearest plane algorithm and the VBLAST detector with and without applying LLL reduction of B as a preprocessing step. These algorithms and their detailed implementations have been discussed in Section 4. The MLD performance is also shown in the figure. Although an efficient implementation of the lattice MLD was discussed in Section 6, any correct implementation of the MLD should give the same optimal performance.

Let us take the performance of the Babai nearest plane algorithm at a BER of 3×10^{-4} as the reference point for the ease of comparison. The ZF detector performs worse by about 1dB. The VBLAST detector is in fact the nearest plane algorithm with the basis ordering preprocessing (c.f. Section 4). It can be observed from the figure that the basis ordering can provide almost 5dB gain. With the LLL reduction preprocessing, the three detectors (abbreviated as LLL-ZF, LLL-Babai and LLL-VBLAST, respectively) have similar performance that offers about 10dB gain over the reference point. It implies that the LLL basis reduction can provide an additional 5dB gain over the VBLAST basis ordering as a preprocessing step. Note that the LLL-VBLAST does not perform any better than the LLL-Babai, in spite of its higher complexity. It can be explained by the fact that the VBLAST basis ordering is a kind of swapping-only basis reduction, which is unlikely to further improve a basis that has already been LLL-reduced. Finally, it can be seen from Figure 2 that the LLL-Babai (as well as LLL-ZF and LLL-VBLAST) is only about 2.5dB from the MLD performance and has the same optimal diversity order. This observations are consistent with and extends those of Yao and Wornell in [28] for a 2-dimensional complex Gauss-reduced basis. Considering the low implementation complexity, the LLL-ZF and LLL-Babai are very attractive suboptimal lattice decoders.

Next, a 4-transmit 4-receive antenna system with 64-QAM is considered. It gives rise to a 4-dimensional complex lattice in a 4-dimensional complex vector space. As discussed in Section 5, we may perform the complex LLL reduction (abbreviated as CLLL) directly on the complex lattice or we may apply the conventional LLL reduction to an equivalent 8-dimensional lattice in a 8-dimensional real vector space. Figure 3 shows the BER performance of the 3 complex lattice based detectors (i.e., CLLL-ZF, CLLL-Babai and CLLL-VBLAST) as well as the 6 detectors previously considered.

For the ease of performance comparison, we shall take the performance of the Babai nearest plane algorithm at a BER of 10^{-3} as the reference point. Observations similar to those made for the previous system still hold. The ZF detector performs only a fraction of a dB worse. The VBLAST basis ordering offers nearly 5dB gain. The LLL-Babai provides an impressive gain of about 12dB. The LLL-VBLAST does not improve over the LLL-Babai. Somewhat unexpected is that the performance gap between the LLL-Babai and LLL-ZF widens to about 1dB, while their gap almost vanishes in the previous system. It is interesting to note that the complex LLL reduction can provide the same performance gain as the traditional real LLL reduction, in spite of its lower complexity (c.f. Section 5). Also, the CLLL-Babai (as well as LLL-Babai, CLLL-VBLAST and LLL-VBLAST) is only about 2.5dB from the MLD performance and achieves the same optimal diversity order. Therefore, the CLLL-Babai is the most attractive low-complexity suboptimal lattice decoder for the QAM system under consideration.

The time complexities of different implementations of the optimal lattice decoder applied to the previous 4-transmit 4-receive antenna system with 64-QAM are evaluated and compared in Figure 4. For simplicity, no LLL reduction preprocessing has been performed. Figure 4 shows the average CPU time per symbol of the Pohst-Fincke enumeration algorithm and its Schnorr-Euchner variant, with and without the stopping test discussed in Section 6. The initial radius is set according to the Babai point, explicitly for the Pohst-Fincke algorithm and implicitly for the Schnorr-Euchner variant.

Figure 4 shows that the Schnorr-Euchner ordering can speed up the Pohst-Fincke algorithm by about 3 times at 35dB. However, the speedup factor becomes smaller than 2 as the SNR increases to 50dB. This can be explained by the fact that as the SNR increases, the enumeration sphere contains only a few number of lattice points and hence the order of enumerating the lattice points become less significant. It can also be seen from the figure that the use of the stopping test can significantly speed up the Pohst-Fincke algorithm with both the original Pohst ordering and the Schnorr-Euchner ordering, but at different SNRs. In particular, the speedup factor for the former increases from 6% at 35dB to over 60% at 50dB, while that for the latter decreases from 36% at 35dB to only 2% at 50dB.

Following the discussion in Section 6.3, with the stopping test, the complexity of both the Pohst-Fincke algorithm and its Schnorr-Euchner variant should converge to that of the Babai nearest plane algorithm, as the SNR tends to infinity. It can be concluded from Figure 4 that the convergence has already occurred at 50dB. However, at practical values of SNR (i.e., at 40dB or smaller), the speedup factor due to the stopping test is significant only when combined with the Schnorr-Euchner ordering. Finally, we note that the Pohst-Fincke algorithm with both the Schnorr-Euchner ordering and the stopping test is the most efficient implementation of the optimal lattice decoder known, especially at moderate SNRs.

8 Concluding Summary

The general principle and implementation methods of several important lattice algorithms for performing the lattice basis reduction and for finding a nearby or the closest lattice point, have been reviewed. Specifically, the LLL reduction algorithm, the Babai nearest plane algorithm and the Pohst-Fincke enumeration algorithm with the Schnorr-Euchner ordering were discussed in details. Our general treatment allows the LLL reduction algorithm to be extended for complex lattices with unnecessarily square basis matrices. This generalization is important for designing low-complexity universal lattice decoders for typical communications applications, in which the passband quadrature phase modulation schemes (such as QPSK and QAM) are used. Our simulation result verified the the complex LLL reduction algorithm, if applicable, can reduce complexity without degrading performance.

The interpretation of the well-known VBLAST detector as the Babai nearest plane algorithm with basis ordering was introduced. Based on Babai's results on finding a nearby lattice point, the use of LLL reduction was shown to be a promising way to improve the performance of the ZF and VBLAST detectors with manageable additional complexity. Our simulation experiments have verified that the LLL reduction preprocessing improves the Babai nearest plane algorithm (and the ZF detector) by about 10dB to 12dB and achieves the optimal diversity order. To gain a further 2.5dB to 3dB and attain the MLD performance, we can employ the Pohst-Fincke enumeration algorithm with the Schnorr-Euchner ordering and the stopping test, which is the fastest universal lattice decoder known, especially at moderate SNRs.

Finally, we conclude by pointing out some promising new development of the Pohst-Fincke enumeration algorithm. It is not difficult to see that the universal lattice decoder based on the Pohst-Fincke enumeration algorithm or its variants is a natural list output decoder, from which soft output can be generated [36], [37], [38]. Hochwald and ten Brink [37] recently also realized the idea and extended the Pohst-Fincke enumeration algorithm to the so-called soft-output complex sphere decoder that is applicable to complex modulation schemes (such as M-PSK) without a lattice structure. Vikalo and Hassibi [39] further extended the algorithm to efficiently accept soft input. The soft-input soft-output sphere decoders resulted appear to be near-optimal as the channel capacity can be approached by applying the famous Turbo decoding principle.

Acknowledgments

The author would like to sincerely thank Prof. Pingzhi Fan for all the arrangements that make this paper possible. He also greatly appreciate Mr. Ying Hung Gan for implementing the lattice algorithms discussed here and generating all the simulation graphs in this paper.

References

[1] W. H. Mow. Maximum Likelihood Sequence Estimation from the Lattice Viewpoint. M.Phil. Thesis, Dept. of Information Engineering, the Chinese University of Hong Kong, June 1991; downloadable at http://www.ee.ust.hk/˜eewhmow.

[2] W. H. Mow. Maximum Likelihood Sequence Estimation from the Lattice Viewpoint. *IEEE Trans. Inform. Theory* 1994; **40**(5): 1591–1600.

[3] E. Viterbo, E. Biglieri. A universal decoding algorithm for lattice codes. *Proc. GRETSI*, Juan-les-Pins, France, Sept. 1993; pp. 611–614.

[4] E. Viterbo, J. Boutros. A universal lattice code decoder for fading channels. *IEEE Trans. Inform. Theory* 1999; **45**(5): 1639–1642.

[5] M. O. Damen, A. Chkeif, J.-C. Belfiore. Lattice code decoder for space-time codes. *IEEE Commun. Lett.* 2000; **4**(5): 161–163.

[6] L. Brunel, J. J. Boutros. Lattice Decoding for Joint Detection in Direct-Sequence CDMA Systems. *IEEE Trans. Inform. Theory* 2003; **49**(4): 1030–1037.

[7] H. Vikalo, B. Hassibi. Maximum-Likelihood Sequence Detection of Multiple Antenna Systems over Dispersive Channels via Sphere Decoding. *EURASIP Journal on Applied Signal Processing* 2002; **2002**(5), 525–531.

[8] J. H. Conway, N. J. A. Sloane. Sphere Packings, Lattices and Groups. Springer-Verlag, New York, 1988.

[9] J. H. Conway, N. J. A. Sloane. Fast Quantizing and Decoding Algorithms for Lattice Quantizers and Codes. *IEEE Trans. Inform. Theory* 1982; **28**: 227–232.

[10] J. H. Conway, N. J. A. Sloane. Soft Decoding Techniques for Codes and Lattices, including the Golay Code and the Leech Lattice. *IEEE Trans. Inform. Theory* 1986; **32**: 41–50.

[11] R. Kannan. Improved Algorithms for Integer Programming and Related Lattice Problems. In Proceedings of the 15th Annual ACM Sympos. Theory of Computing, 1983; pages 193–206, .

[12] R. Kannan. Minkowski's Convex Body Theorem and Integer Programming. *Math. Oper. Research* 1987; **12**: 415–440.

[13] J. W. S. Cassels. An Introduction to the Geometry of Numbers. Springer, Berlin/Heidelberg, 1959.

[14] P. van Emde Boas. Another NP-complete Partition Problem and the Complexity of Computing Short Vectors in a Lattice. Technical Report 81-04, Dept. of Mathematics, Univ. of Amsterdam, 1981.

[15] A. K. Lenstra, H. W. Lenstra, L. Lovász. Factoring Polynomials with Rational Coefficients. *Math. Ann.* 1982, **261**: 513–534.

[16] H. W. Lenstra. Integer Programming with a Fixed Number of Variables. *Math. Oper. Res.* 1983; **8**: 538–548.

[17] A. K. Lenstra. Lattices and Factorization of Polynomials. Technical Report IW 190/81, Mathematisch Centrum, Amsterdam, 1981.

[18] R. Kannan, A. K. Lenstra, L. Lovász. Polynomial Factorization and Nonrandomness of Bits of Algebraic and Some Transcendental Numbers. In *Proc. 16th Ann. ACM Symp. on Theory of Computing*, 1984; 191–200.

[19] M. Grotschel, L. Lovász, A. Schrijver. Geometric Methods in Combinatorial Optimization. In *Proc. Silver Jubilee Conf. on Combinatorics*, Univ. of Waterloo, 1982; **1**:167–183.

[20] J. C. Lagarias, A. M. Odlyzko. Solving Low Density Subset Sum Problems. In Proc. 24th IEEE Symp. on the Foundations of Computer Science 1983: 1–10.

[21] A. Shamir. A Polynomial Time Algorithm for Breaking the Merkle-Hellman Cryptosystem. In *Proc. 23rd IEEE Symp. on Foundations of Computer Science*, 1982; 145–152.

[22] L. Lovász. *An Algorithmic Theory of Numbers, Graphs and Convexity.* Capital City Press, Montpelier, Vermont, 1986.

[23] E. Kaltofen. On the Complexity of Finding Short Vectors in Integer Lattices. *Lecture Notes in Computer Science 162*, pages 236–244. Spinger-Verlag, Berlin/New York, 1983.

[24] U. Fincke and M. Pohst. Improved Methods for Calculating Vectors of Short Length in a Lattice, Including a Complexity Analysis. *Math. Comp.* 1985; **44**:463–471.

[25] L. Babai. On Lovász' lattice reduction and the nearest lattice point problem. In Proceedings of Symposium on Theoretical Aspects in Computer Science, *Lecture Notes in Computer Science*, Berlin: Springer-Verlag, 1985, **182**: 13–20.

[26] L. Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatoria* 1986, **6**(1): 1–13.

[27] G. J. Foschini. Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas. *Bell Labs Technical Journal* 1996; **1**(2): 41–59.

[28] H. Yao, G. W. Wornell. Lattice-Reduction-Aided Detectors for MIMO Communication Systems. In *Proc. IEEE Globecom*, Taipei, November 17-21, 2002.

[29] M. Pohst. On the computation of lattice vectors of minimal length, successive minima and reduced basis with applications. *ACM SIGSAM Bulletin* 1981; **15**: 37–44.

[30] E. Agrell, T. Eriksson. A. Vardy, K. Zeger. Closest Point Search in Lattices. *IEEE Trans. Inform. Theory* 2002, **48**: 2201–2213.

[31] U. Dieter. How to Calculate Shortest Vectors in a Lattice. *Math. Comp.* 1975; **29**:827–833.

[32] D. E. Knuth. *The Art of Computer Programming.* Addison-Wesley, Reading, Mass., second edition, 1981; **2**: 95–97.

[33] C. P. Schnorr, M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming* 1994; **66**: 181–191.

[34] A. M. Chan, I. Lee. A new reduced-complexity sphere decoder for multiple antenna systems. In *Proc. IEEE International Conference on Communications* 2002, **1**: 460–464.

[35] B. Hassibin, H. Vikalo. On the Expected Complexity of Sphere Decoding. In *Proc. Asilomar Conf. Signals, Systems and Computers* 2001; **2**: 1051–1055.

[36] C. Nill, C.-E. W. Sundberg. List and soft symbol Output Viterbi algorithms: extensions and comparisons. *IEEE Trans. Commun.* 1995; **43**(2/3/4):277–287.

[37] B. M. Hochwald, S. ten Brink. Achieving Near-Capacity on a Multiple-Antenna Channel. *IEEE Trans. Commun.* 2003; **51**(3): 389–399.

[38] W. H. Mow. Lattice sequence detectors for advanced communication systems with large bandwidth efficiency. Hong Kong RGC Competitive Earmarked Research Grant, Project Proposal no. HKUST6246/02E, Sept 2001.

[39] H. Vikalo, B. Hassibi. Towards Closing the Capacity Gap on Multiple Antenna Channels. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing* 2002; **3**: 2385–2388.

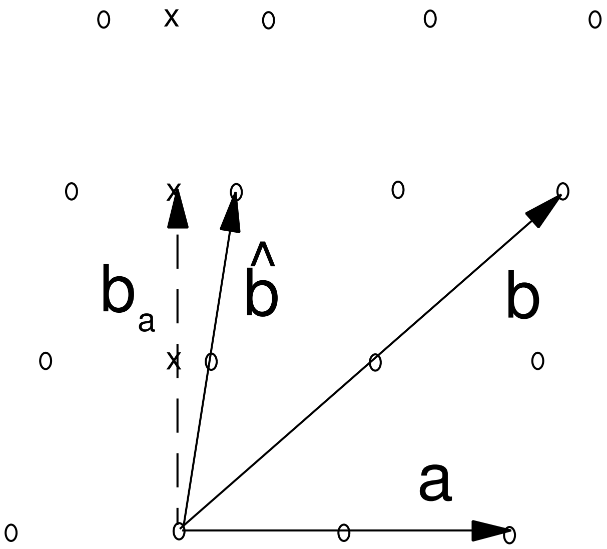

Figure 1: A two-dimensional example of projecting b perpendicular to a to produce b_a and then lifting b_a to $\hat{b}$, where “o” represent the lattice points in L, “x” represent the lattice points in the projected lattice L_a.

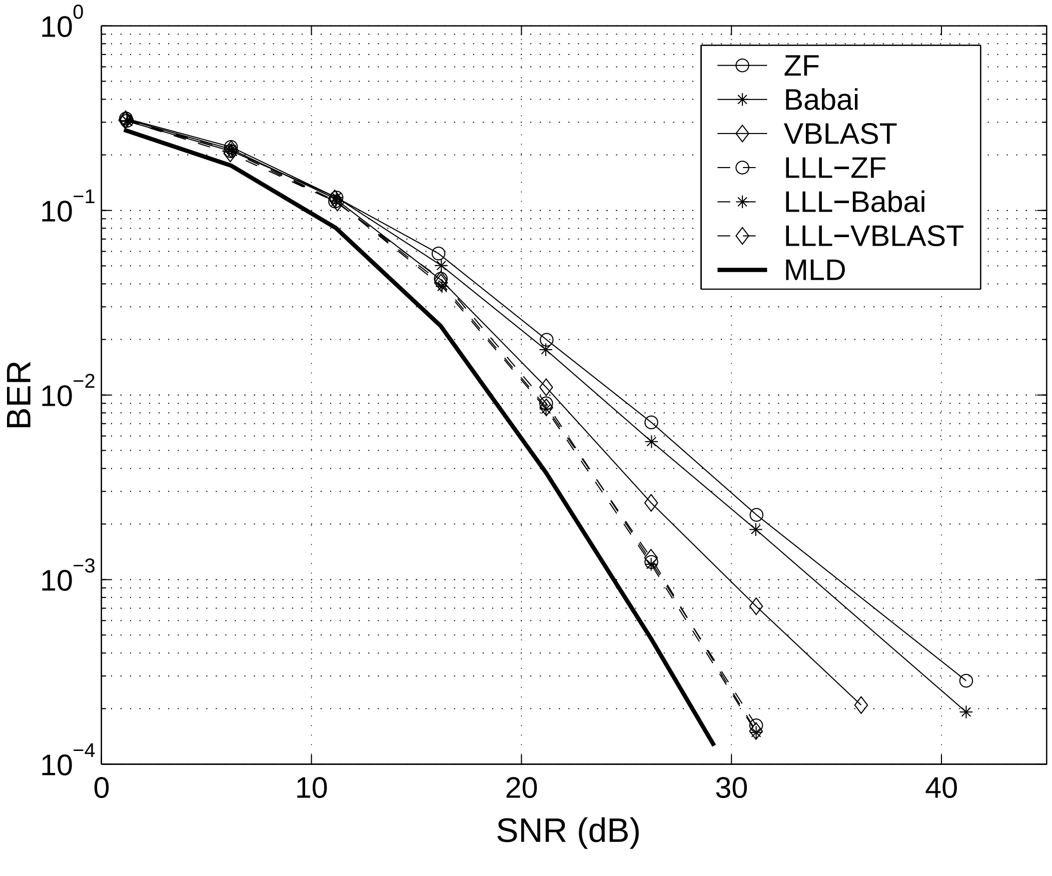

Figure 2: Performance of the zero-forcing detector, the Babai nearest plane algorithm, and the VBLAST detector, with and without the LLL reduction preprocessing, as well as the MLD in a 2-transmit 3-receive antenna system with 4-PAM.

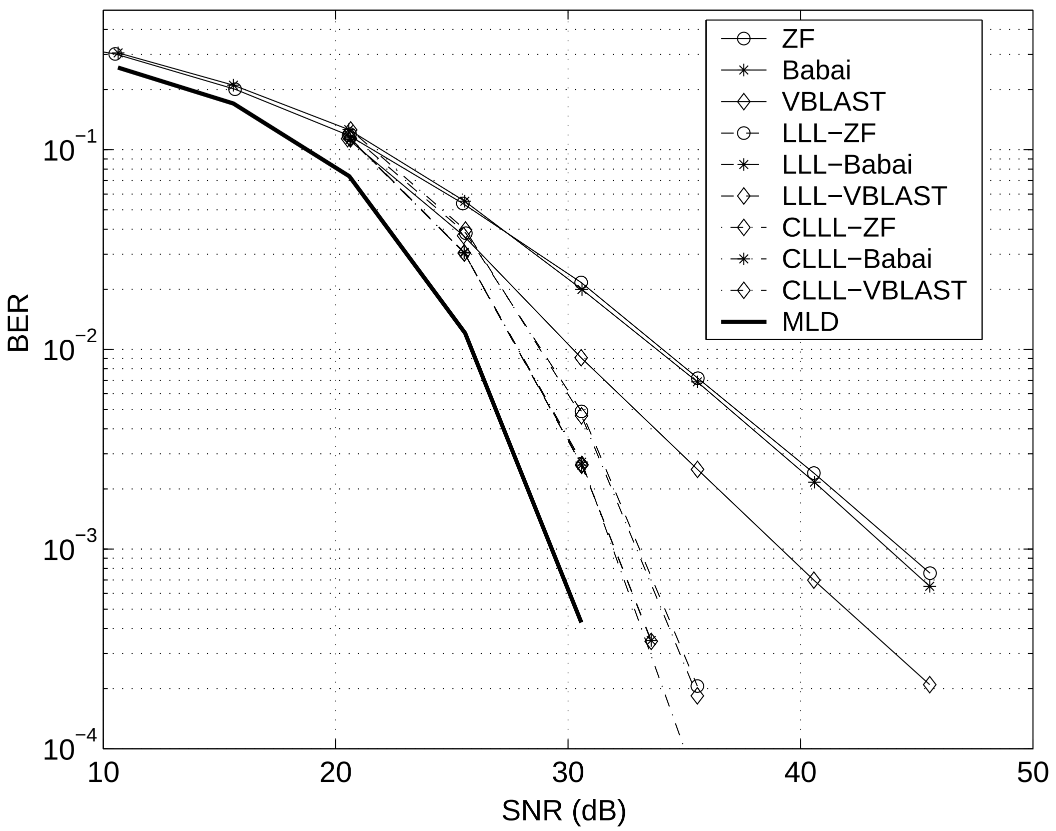

Figure 3: Performance of the zero-forcing detector, the Babai nearest plane algorithm, and the VBLAST detector, with and without the real or complex LLL reduction preprocessing, as well as the MLD in a 4-transmit 4-receive antenna system with 64-QAM.

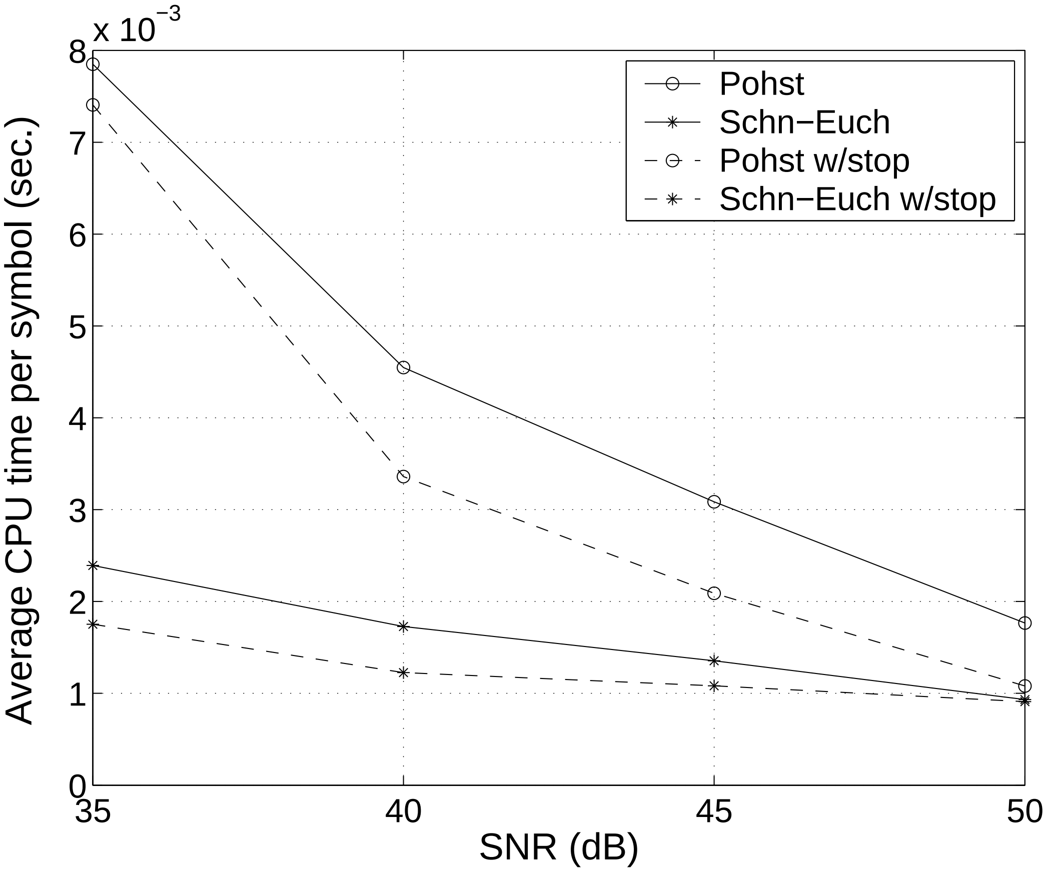

Figure 4: Time complexity of the Pohst-Fincke enumeration algorithm with the original Pohst ordering or the Schnorr-Euchner ordering, and with or without the stopping test, as a function of the SNR.