MAXIMUM LIKELIHOOD SEQUENCE ESTIMATION FROM THE LATTICE VIEWPOINT

By

Mow Wai Ho

A thesis

submitted in partial fulfillment of the requirements

for the Degree of Master of Philosophy

Department of Information Engineering

The Chinese University of Hong Hong

June 1991

Dedicated to my mother

and the memory of my father

Acknowledgement

I would like to acknowledge my supervisor, Dr. C. P. Kwong, who introduced me the sequence estimation problem and encouraged me to look at it from new viewpoints. It is also him who taught me the right attitude, which I believe every true researcher ought to have, towards a difficult problem.

Abstract

It is well-known that the use of the Viterbi algorithm to implement a sequence estimator is an optimal way to remove the effect of intersymbol interference for digital transmission systems. However, such an implementation usually results in a very complicated receiver. In this thesis, we transform the problem of maximum likelihood sequence estimation into the problem of finding the closest lattice point. Some related lattice algorithms such as the basis reduction algorithms and the enumeration algorithms are analyzed and some improved versions are suggested. Then efficient algorithms finding the nearest lattice point are derived. Based on these lattice algorithms, simple but effective sequence estimators are proposed for the PAM systems and their complexities are analyzed. Under some mild assumptions, our algorithms have both polynomial space and time complexities, and are therefore much superior to the conventional Viterbi detectors. Simulation results on three different channels show that the performance of the new sequence estimators depend on the distance spectrum of the channel. But, general speaking, the performance approaches optimal as the size of the signal set and the signal-to-noise ratio increase. Finally, the extensions to other lattice-type modulation schemes and the impacts of the lattice viewpoint on the design of bandlimited transmission systems are discussed.

Contents

List of Figures

Chapter 1

Introduction

For high-rate digital transmission systems, a major difficulty with signalling is the increased amount of intersymbol interference (ISI). Performance of the symbol-by-symbol detector (SSD) becomes unsatisfactory since this form of interference cannot be tackled by simply raising the signal power. Early approach uses equalization technique. However, the linear equalizer cannot handle channels with spectral nulls. Decision feedback equalizer (DFE) is effective for the removal of ISI, but it is highly susceptible to the effect of error propagation and its error-rate performance is difficult to analyze [2].

In 1972, Forney [27] introduced the whitened matched filter so that an important class of channels can be described by a linear discrete-time model with additive white Gaussian noise (AWGN). In the same paper, he proposed to implement a maximum likelihood sequence estimator (MLSE) using the Viterbi algorithm (VA). Such estimator has been shown to be optimal in the sense of minimizing the probability of sequence error and is much superior to the conventional SSD. As an example, for partial-response systems, the symbol error rate of the MLSE can outperform that of the SSD by as much as 3dB [27]. Nevertheless, the complexity of VA is prohibitively large for many typical channels since it performs lm^v operations to detect a sequence

of l symbols with m possible transmit levels and v interfering components. A considerable amount of research has been undertaken to investigate suboptimal Viterbi-like estimators. The complexity of VA and its related suboptimal estimators will be discussed later.

Over the past two decades, almost all schemes proposed for sequence estimators were variants of the VA. Very few fundamentally new ideas had arisen. It should be emphasized that the VA is just one possible way to implement the optimal sequence estimator. Other alternatives should deserve more attentions.

Recently, Barbosa [1] viewed the channel as a linear map from the input space to the observation space, and the MLSE as a macro operation. This enables a unified approach to the performance analysis of various suboptimal receivers. He also derived a Viterbi-like algorithm using the mapping concept. Unfortunately, the proposed algorithm is not simplier than the original VA. This result led him to the following conclusion [1]:

> "The new approach not only gives insight into the macro operations involved but also shows that information preserving transformations do not simplify the complexity of the MLSE algorithm."

However, we can derive better MLSE algorithms by improved utilization of the information inherent in the source symbol sequences. This is made possible by the following observation.

Observation 1.1 *For an important class of modulation schemes, all the possible source symbol sequences form a finite integer lattice.*

Since a lattice after a linear transformation is still a lattice, the transmitted symbol sequences must be points in a finite lattice. The received sequence is the observation of the transmitted sequence in the presence of noise. Hence the problem of estimating the transmitted sequence with maximum likelihood corresponds to the problem of finding the lattice point closest to a given query point. Based on this lattice viewpoint, the

MLSE problem can be transformed into the closest lattice point problem. Making use of the regular structure of a lattice, we propose very efficient MLSE algorithms whose complexities are essentially independent of the number of transmit levels m. Obviously, these algorithms significantly simplify the implementation of an optimal receiver, especially for multi-level modulation schemes, as compared with the other algorithms known so far [27], [61], [1]. Nonetheless, the performance of our algorithms does depend on m. As an interesting result, unlike the VA whose complexity increases rapidly with m, the new algorithms favor m to be as large as possible.

The organization of this thesis is as follows. Chapter 1 states the channel model and its related assumptions, and some issues about the complexity measures. We then introduce the MLSE, and the VA associated with its complexity and performance analysis. A survey on the suboptimal Viterbi-like algorithms and a discussion on the trends of digital transmission follow. Chapter 2 explains a modified version of the VA, called the truncated VA, which enables the storage requirement of a sequence estimator to be independent of the length of the sequence. The performance of such estimator depends on the choice of the truncation depth and we give some suggestions on the choice so as to get optimal performance. The truncation approach of implementing a Viterbi detector is generalized to the concept of decomposition of a sequence estimator. After that, the MLSE problem is formulated as a closest lattice point problem, and a closely related problem, called the closest vector problem (CVP), is explained. Chapter 3 focuses on the CVP and some related lattice algorithms. First, an introduction to the lattice and some related facts is given. Second, a very important basis reduction algorithm, called the LLL-reduction algorihtm, is examined in details. A basis reduction algorithm converts a given representation of a lattice into another representation such that some nice properties of the lattice can be exploited easily. Our improved version of LLL-reduction algorithm is then presented. Third, enumeration algorithms, which enumerate lattice points inside certain regions, are explained together with their complexity analyses. Various enumeration

algorithms are classified according to the shapes of the regions to be enumerated. A unified treatment of these enumeration algorithms is developed based on the concept of isometric mapping. Besides, an enumeration algorithm, for the shortest lattice vector or the closest lattice vector, with improved complexity is derived. Forth, a straightforward CVP algorithm is obtained by combining the basis reduction algorithm and the enumeration algorithm. Such reduce-and-enumerate approach to the CVP is well-known and its worst-case complexity is discussed. Finally, we introduce the concept of norm approximation and propose a CVP algorithm with improved average-case complexity. Chapter 4 illustrate the new MLSE algorithms, which are derived easily as special cases of the CVP algorithms. A simplified MLSE algorithm is derived for the channels connected with the simple cubic lattice. The boundary effect, which is due to the finiteness of a lattice associated with the given modulation schemes, is reduced by a heuristic method. The simulation results for some selected channels are presented and explained. After that, extensions of the proposed MLSE algorithms to other lattice-type modulation schemes are discussed. We also suggest some potential applications of the algorithms and some possible future work related to the lattice interpretation.

1.1 Channel Model and Other Basic Assumptions

The lattice interpretation of MLSE holds for any modulation schemes with lattice-type signal constellation. These include very popular modulation schemes like Pulse Amplitude Modulation (PAM), Quadrature Amplitude Modulation (QAM). For the ease of illustration, from here on, we simply consider the uncoded multi-level PAM transmission systems as shown in figure 1.1.

In the data transmission system, the transmitting filter $f_T(t)$ converts source symbol sequence x into electrical signals suitable for transmission. The channel, described by $f_C(t)$, is merely the medium used to transmit the signal from the transmitting point

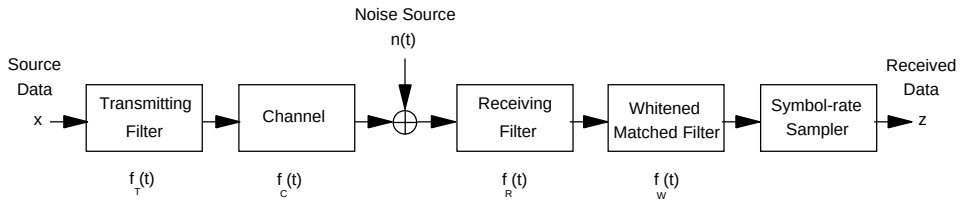

Figure 1.1: The data transmission system.

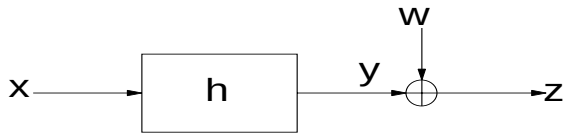

Figure 1.2: The discrete-time channel model.

to the receiving point. The noise source $n(t)$ is additive white Gaussian. The receiving filter $f_R(t)$ is used to suppress the noise outside the channel bandwidth. The structure that a whitened matched filter $f_W(t)$ followed by a symbol-rate sampler [27] enables us to focus on the received symbol sequence z, which provides sufficient statistics for the received signals.

The transmitting filter, the channel, the receiving filter, the whitened matched filter and the symbol-rate sampler can be modeled as a single discrete-time filter as illustrated in figure 1.2. In the figure, the channel impulse response h is defined by

$$h_i = [f_T(t) * f_C(t) * f_R(t) * f_W(t)]_{t=iT}$$

and the whitened noise sequence w is defined by

$$w_i = [n(t) * f_R(t) * f_W(t)]_{t=iT},$$

where T is the symbol interval and $*$ denotes the convolution operator. We remark that although the discrete-time channel model is derived from a baseband transmission system, it is easy to show that, in case of a passband transmission system, the same channel model is still valid. The only difference is that h becomes complex in the latter case.

From now on, we concentrate on the discrete-time channel model in figure 1.2, which satisfies the following assumptions.

Assumption 1.1 *The channel h is linear.*

Assumption 1.2 *The noise sequence w is additive white Gaussian with mean 0 and variance σ^2.*

Assumption 1.3 *Each source symbol x_i must take on one of the m integer values $0, 1, \cdots, m-1$ independently with equal probability.*

Now we define some important symbols which will be used throughout this thesis. In the following definitions, each symbol represents a sequence of some fixed length, though the same symbol may refer to a corresponding vector in later chapters. Note that some symbols have been defined previously but they are redefined below in a consistent manner with a specified length.

Definition 1.1 *Let h be the channel impulse response of length $v+1$, where v is called the channel memory length or equivalently the number of interfering symbols. Let x be the source symbol sequence of length l. Denote z and w as the received symbol sequence and noise sequence respectively, each of length $l+v-1$. Also, define y as the transmitted symbol sequence. Besides, define $\hat{x}$ as the detected symbol sequence and $\hat{y}$ as the corresponding transmitted symbol sequence.*

By definition, we have $z = y + w$. Due to assumption 1.1, $y = h * x$. Thus the channel can be viewed as a convolutional encoder over the real number field. Since the channel is not ideal, each transmitted symbol

$$y_i = h_0 x_i + (\sum_{j=1}^{v} h_j x_{i-j}),$$

where the second term $\sum_{j=1}^{v} h_j x_{i-j}$ represents the distortion due to the ISI.

The above equation also leads to a finite-state machine description of our channel as shown in figure 1.3. Regarding the contents of all storage elements as the state of

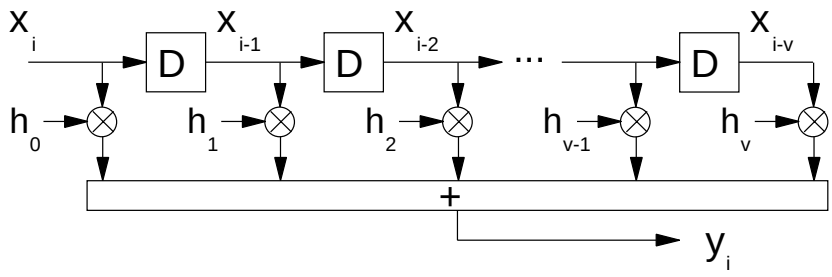

Figure 1.3: Finite-state machine model.

the machine, each transmitted symbol can be treated as the output due to a specific state transition.

Definition 1.2 *Define the state sequence* $s = \{s_i\}$*, where* $s_i = (x_{i-1}, x_{i-2}, \cdots, x_{i-v})$ *is the state at time* i*. Then the transmitted symbol* $y_i = y(s_i, s_{i+1})$ *is solely determined by the state transition* (s_i, s_{i+1})*.*

Note that s_i depends on the previous state s_{i-1} and the current source symbol x_i. Hence the channel can be described as a Markov process.

For all sequences except h, s'_i refers to the element of the sequence s' at time i. The elements of a sequence is ordered such that the current symbols appear first, i.e. $s' = (s'_i, s'_{i-1}, s'_{i-2}, \cdots)$.

1.2 Complexity Measure

Complexity of algorithm is usually described by their asymptotic growth rate or the order of functions. Common measures of order are $O(f)$, functions that grow no faster than f, and $\Omega(f)$, functions that grow at least as fast as f. As an example, consider the procedure of sequentially searching an unordered list having n elements for a match of the given item. If the item does not match any element in the list, at least n comparisons must be done in order to get the conclusion and hence the worst-case complexity is $\Omega(n)$. On the other hand, the procedure can take at most n

comparisons since all elements in the list have been examined. Therefore, the worst-case complexity is also $O(n)$. The precise definitions of these measures can easily be found in literatures on complexity analysis.

In our interested cases, these complexity measures are not accurate because the parameters governing the complexity are quite small. This is not surprising as the complexity of conventional MLSE algorithms increase exponentially with the channel memory length. In practice, few channels with length greater than 10 can be handled. Nevertheless, for the ease of comparison, we need to consider the asymptotic behaviors of different algorithms whenever the exact complexity description is too tedious.

Implementation of algorithms consumes both memory space and computational time. The space complexity of an algorithm is the number of storage elements that must be reserved for its use, while the time complexity counts the number of arithmetic operations that must be executed by a sequential machine. Here, an arithmetic operation may be an addition, a multiplication, a rounding operation or similar. Nowadays parallel processing is quite common. In such situation, it is possible to trade off space for time or vice versa. Thus a reasonable measure is the space-time complexity, which is the product of memory space and computational time assuming parallel computation.

1.3 Maximum Likelihood Sequence Estimator

In the receiver, only the received sequence z can be observed. Decision on which one of many permissible source sequences being transmitted is based on probabilistic argument. Denote $\mathcal{X}$ as the set of all possible source sequences. We want to maximize the a posteriori probability $P(x|z)$ for all x in $\mathcal{X}$. This maximum a posteriori (MAP) rule minimizes the error probability in detecting the whole sequence, and is thus optimum in this sense. A receiver detecting signals using the MAP rule is referred to as a MAP receiver.

Under the condition that all source sequences are equiprobable (i.e. the a priori probability $P(x)$ is the same for all x in $\mathcal{X}$), maximizing $P(x|z)$ is equivalent to maximizing $P(z|x)$. This is termed the maximum likelihood (ML) rule. A receiver detecting signals using the ML rule is referred to as a ML receiver or a MLSE. Note that the MLSE can be treated as a special case of the MAP receiver. Since the source sequence and the state sequence are one-to-one correspondent and the noise terms w_i are independent (by assumption 1.2), the log likelihood function

$$\ln P(z|x) = \ln P(z|s) = \sum_i \ln P(z_i|s_i, s_{i+1}) = \sum_i \ln P(z_i - y(s_i, s_{i+1})),$$

where $y(s_i, s_{i+1})$ is the transmitted symbol corresponding to the state transition (s_i, s_{i+1}). As the noise components are independent and Gaussian (by assumption 1.2), the joint probability density of the noise sequence w is

$$p(w) = \prod_i p(w_i) = \prod_i (\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{w_i^2}{2\sigma^2}}) = K e^{-\frac{\sum_i w_i^2}{2\sigma^2}},$$

where K is a constant. Obviously, we need only to minimize $\sum_i w_i^2 = \sum_i (z_i - y_i)^2$ instead. Thus the MLSE problem can be stated as follows:

Problem 1.1 (The MLSE Problem) *Given a received sequence z, determine the detected sequence $\hat{x}$, corresponding to a unique suspected transmitted sequence $\hat{y}$, among all permissible source sequences such that $\hat{y}$ is closest to z in Euclidean distance.*

1.4 The Viterbi Algorithm — An Implementation of MLSE

Obviously, a brute force approach to the problem of MLSE is to enumerate all permissible source sequences. This requires m^l calculations, each takes l squaring operations and $2l - 1$ additions (refer to section 1.1 for symbol definitions). This trivial method

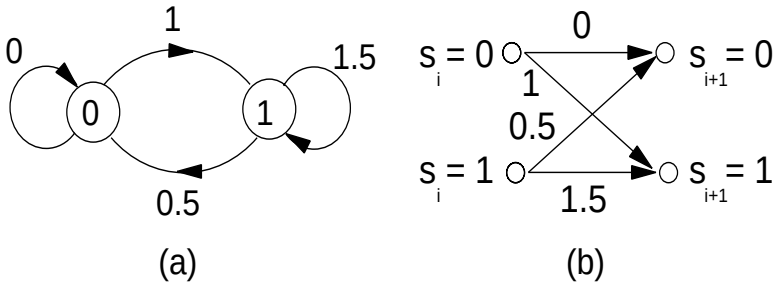

Figure 1.4: (a) State diagram of the channel $h = (1, 0.5)$ for binary transmission. (b) One stage of the corresponding two-state trellis. The weight associated with each transition (s_i, s_{i+1}) is $y(s_i, s_{i+1})$.

takes $O(lm^l)$ time and $O(l)$ space. The method is unacceptable as the computational time increases exponentially with the sequence length.

To derive a more efficient algorithm, we note that the channel is connected with a state diagram as it can be described by a finite-state machine (see figure 1.3). For example, a binary PAM system with channel impulse response $h = (1, 0.5)$ has the state diagram as shown in figure 1.4(a). Alternatively, it can be represented by a two-state trellis diagram which shows all possible transitions of states over time. The trellis diagram of our example is pictured in figure 1.4(b). Note that state s_i is defined as (x_{i-1}) and each transition (s_i, s_{i+1}) is associated with a weight $y(s_i, s_{i+1}) = h_0 x_i + h_1 x_{i-1}$.

Since the trellis can be regarded as a graph, from now on, a state is called a node, a transition is called a branch and a state sequence is called a path. The weight associated with a branch is termed branch metric and the accumulated weight associated with a path is termed path metric.

Define the branch metric as $|z_i - y(s_i, s_{i+1})|^2$ for each branch (s_i, s_{i+1}) of the trellis. We can find the shortest path (the one with smallest path metric) by computing the branch metrics stage by stage. Note that each node has m incoming branches except a few stages in the beginning and in the end. Each incoming branch is due to the advent of a new source symbol. Of the m incoming branches, only the one connected with the minimum partial path metric is retained. That retained partial path is

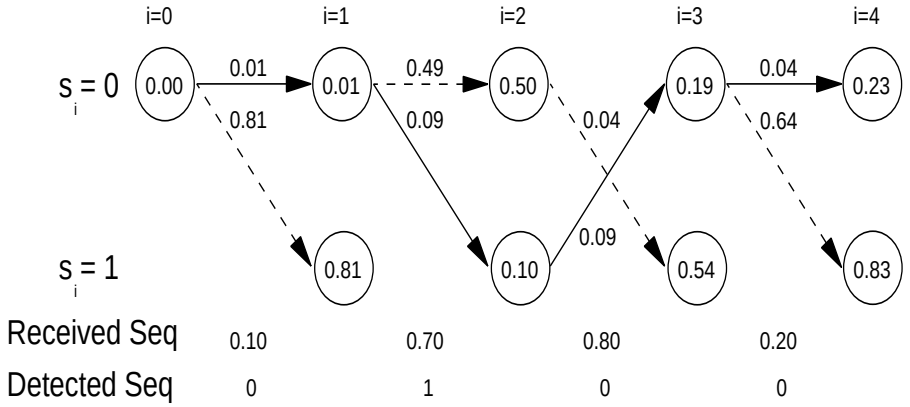

Figure 1.5: An example illustrates the use of the Viterbi algorithm to find the shortest path for the channel impulse response $h = (1, 0.5)$ and $m = 2$ with the received sequence $z = (0.2, 0.8, 0.7, 0.1)$. The initial state s_0 is assumed to be 0. The weight of each branch is the branch metric and the partial path metric is shown inside the node. The survivor paths at each stage are shown. The detected sequence is determined by the final survivor path which is represented as a chain of soild arrows.

referred to as survivor path. Partial path associated with the other $m - 1$ incoming branches are discarded since the shortest path must contain the survivor path if it goes through this particular node. Otherwise, a shorter path can be found by replacing the partial path up to this node by the survivor path. Therefore the number of survivor paths is exactly the same as the number of nodes in a stage. After all stages of the trellis have been gone through, the shortest path is the remaining survivor path which has the smallest path metric. Obviously, this shortest path corresponds to the ML sequence. The algorithm just described to find the shortest path in a trellis is the Viterbi algorithm (VA).

The details of how the VA calculates the shortest path is illustrated in figure 1.5. Again the example in figure 1.4 is considered. The received sequence z is (0.2, 0.8, 0.7, 0.1) and the detected sequence $\hat{x}$ found is (0, 0, 1, 0). Notice that knowledge of the initial state is assumed, i.e. $s_0 = 0$ or $x_{-1} = 0$. In general, the number of nodes in each stage is m^v and there are totally l stages. Therefore the algorithm requires m^v storage elements, each must be capable of storing a path metric (p bits) and survivor path ($l \log(m)$ bits). The space complexity of VA is $(l \log(m) + p)m^v$ bits. Assuming all m^v branch metrics being precomputed (this requires an extra storage of pm^v bits),

1 multiplication, 1 addition and $m-1$ binary comparisons are needed for each node. Thus the time complexity of VA is $O(m^{v+1})$ operations per detected symbol. The virtue of the VA is that the number of operations necessary for MLSE grows linearly with l rather than exponentially. Its main disadvantage is the huge storage necessary to store all survivor paths.

While VA was originally proposed for decoding convolutional codes, in its most general form, VA is the solution to the problem of MAP estimation of the state sequence of a finite-state discrete-time Markov process observed in memoryless noise [28]. It thus finds applications in many different areas like sequence estimation, digital magnetic recording and some pattern recognition problems in addition to decoding convolutional codes. An excellent introduction to VA can be found in references [28], [33].

1.5 Error Performance of the Viterbi Algorithm

Forney [27] presented upper and lower bounds on the symbol error probability when the MLSE is implemented using VA. The upper bound is tight for moderate-to-large signal-to-noise ratio (SNR). In order to understand the optimal performance of the VA and the important concept of error event, we repeat Forney's performance analysis on the VA below (see also [61]).

Recall that x represents the source sequence and $\hat{x}$ represents the detected sequence.

Definition 1.3 *Define the error sequence as $e = \hat{x} - x$ such that e_i must take on one of the $(2m-1)$ integer values from $-(m-1)$ to $m-1$.*

As consecutive symbol errors are not independent of each other, the concept of sequence error events is introduced to simplify error analysis. The beginning of an error events ε is arbitrarily assigned with time 0:

$$\varepsilon = (\cdots, 0, 0, e_0, e_1, \cdots, e_k, 0, 0, \cdots)$$

In most situations (i.e. when the SNR is moderate and no catastrophic behavior is present) , error events are short compared with the interval between them. The error events are effectively independent of each other.

For an error event ε to happen, a sufficient condition is the simultaneous occurrence of the following two subevents:

ε_1**:** $\hat{x}$ must be an allowable source sequence.

ε_2**:** the noise sequence w must be such that $\hat{x}$ has greater likelihood than x.

Then we have

$$P(\varepsilon) \leq P(\varepsilon_1, \varepsilon_2) = P(\varepsilon_1)P(\varepsilon_2|\varepsilon_1).$$

The subevent ε_1 is independent of ε_2, and depends only on the message ensemble. When $|e_i| = j$, only $m - j$ values of x_i are permissible, so that

$$P(\varepsilon_1) = \prod_{i=0}^{k} \frac{m - |e_i|}{m}.$$

Note we have used assumption 1.3 that source symbols are independent and equiprobable. In words, the subevent ε_2 requires suspected transmitted sequence $\hat{y}$ to be closer to the received sequence z than the true transmitted sequence y. Since the noise sequence is AWGN with equal variance σ^2 in all dimensions, it is spherically symmetric. Considering the two-dimensional subspace containing y, $\hat{y}$ and z, it is easy to see that the probability of ε_2 is simply the probability that a single Gaussian variable of variance σ^2 exceeds half the Euclidean distance between y and $\hat{y}$. Define $d(\varepsilon)$ as the Euclidean distance of the sequence $\hat{y} - y$ or $\|\hat{y} - y\|$. Then

$$P(\varepsilon_2) = \int_{d(\varepsilon)/2}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{n^2}{2\sigma^2}} dn = Q(\frac{d(\varepsilon)}{2\sigma})$$

where $Q(.)$ is the Gaussian distribution function.

Let $\mathcal{E}$ be the set of all possible error events ε, and $\mathcal{D}$ be the set of all possible $d(\varepsilon)$. For each $d \in \mathcal{D}$, let $\mathcal{E}_d$ be the set of $\varepsilon \in \mathcal{E}$ such that $d(\varepsilon) = d$. Then

$$P(\mathcal{E}) = \sum_{\varepsilon \in \mathcal{E}} P(\varepsilon) \leq \sum_{d \in \mathcal{D}} Q(\frac{d}{2\sigma}) \sum_{\varepsilon \in \mathcal{E}_d} (\prod_{i=0}^{k} \frac{m - |e_i|}{m}).$$

Due to the exponential decrease of the Gaussian distribution function, this expression will be dominated at large enough SNR by the term involving the minimum value d_{min} of $d(\varepsilon)$. Thus

$$P(\mathcal{E}) \simeq KQ(\frac{d_{min}}{2\sigma}), \tag{1.1}$$

where the constant

$$K = \sum_{\varepsilon \in \mathcal{E}_{d_{min}}} (\prod_{i=0}^{k} \frac{m - |e_i|}{m}).$$

This bound, derived as an approximation to upper bound, is actually an upper bound as shown by Hughes [35].

Note that the bound is valid under the condition that there are not too many minimum distance error events. For many channels, there are only a pair of such error events — the error event and its negated one. Nonetheless, some channels do have a lot of minimum distance error events.

Definition 1.4 *A catastrophic channel is one which have an infinite number of minimum distance error events.*

In the extreme case, for catastrophic channels, K in equation 1.1 may not be bounded. For example, the catastrophic channel $h = (1, -1)$ with $d_{min}^2 = 2$ has minimum distance error events in the forms of $(\cdots, 0, 1, \cdots, 1, 0, \cdots)$ and $(\cdots, 0, -1, \cdots, -1, 0, \cdots)$. This will cause an infinite number of detection errors. In practice, a precoding method is used to correct this problem [27].

The VA is optimal in the sense of minimizing the probability of sequence error. Namely all erroneous sequences with the same weight are equally bad. General speaking, optimum bit-by-bit detection and optimum sequence detection are not equivalent. However, for high SNR, erroneous detected sequences may not be too far from the true sequences and minimization on both criteria should give nearly the same performance.

1.6 Suboptimal Viterbi-like Algorithms

In spite of its optimal performance, the MLSE using VA is impractical due to two limitations.

Limitation 1.1 *The space complexity of the estimator, which grows exponentially with the channel memory length v, becomes prohibitively large for most practical channels. If parallel computation is not allowed, the time complexity is also an important constraint.*

Limitation 1.2 *For bandlimited channels, a practical way to increase the transmission rate is to use a large signal set. However, both space and time complexities of the estimator are very sensitive to the size of the signal set m in addition to v.*

A considerable amount of research has been undertaken to suggest effective suboptimum detectors which achieve good error performance but with manageable complexity.

Early work is mainly concerned with reducing the channel length by preprocessing techniques. Falconer and Magee [23] proposed a linear prefilter to force the channel memory to a desired length. Lee and Hill [48] suggested the use of DFE to shorten the channel length so as to reduce the noise enhancement in the linear equalizer. These schemes sacrificed a lot in performance because the noise after prefiltering is enhanced and is no more AWGN. This violates the basic assumption of VA that the noise is memoryless.

Later research concentrated on new definitions of states and branch metrics while leaving the framework of VA intact. The basic idea to reduce receiver complexity by neglecting paths unlikely to have the smallest path metric.

Clark and Clayden [9] proposed the pseudobinary VA which allows only two states in every stage of the trellis to survive. This always results in a two-state trellis. Penalty in performance is heavy since the optimal path will be discarded if in any

time there are two partial paths which have path metrics smaller than that of the optimal path.

Duel-Hallen and Heegrad [17] developed the method of the delayed decision-feedback sequence estimation (DDFSE) . Their main ideas are to define the states of trellis by considering the first few terms of ISI, and to include an estimate of the tail of ISI in the definition of branch metrics. This reduces the effect of error propagation. The algorithm can handle channel response of very great length or even infinite length, and allow the designer to tradeoff receiver complexity for performance by choosing the number of ISI terms involved in the definition of states.

More recent approach also considers the effect of large signal set. Wesolowski [63] considered the case of two-dimensional signal constellations and limited the survivor states to those associated with the signal points adjacent to the observed point. This reduces complexity due to modulation formats with large signal set.

Another more sophisticated algorithm is the reduced-state sequence estimation (RSSE), suggested by Eyuboglu and Qureshi [21]. Trellis states as defined in VA are merged using Ungerboeck-like set partitioning principles resulting in a simpler trellis. In this way, the minimum intraset distance is maximized such that in most cases only one signal point, called the representative point, in each set is close to the observed point. By defining the trellis state in terms of this set instead of signal points, a reduced-state trellis is obtained. The branch metric is calculated as distance between the observed point and the representative point. Besides, decision-feedback can be incorporated into the algorithm in a way similar to that of DDFSE to reduce complexity due to large number of ISI terms.

The problem of Wesolowski's algorithm and RSSE is in the definition of distance or closeness. Using Barbosa's terminology [1], the above algorithms approximate the metric of observation space by that of the input space. However, the metric of input space is not necessarily Euclidean even though that defined in observation space is so. As a result, two vectors close to each other in input space may lead to great separation

of their corresponding vectors in observation space. The algorithms designed under this inaccurate metric approximation, of course, do not have good error performance.

1.7 Trends of Digital Transmission and MLSE

Nyquist showed in 1928 that the maximum signalling rate achievable for a channel with the bandwidth B_0 Hz, for no ISI, is $2B_0$ bauds, and is known as the Nyquist rate [55]. In 1963, Lender [49] showed that this theoretical minimum bandwidth can be achieved without infinitely sharp filters. The technique employed is called duobinary signalling (or partial response signalling or correlative coding). The idea is to introduce some controlled amount of ISI into the data stream rather than trying to eliminate it completely. At the receiver, special procedure is used to reduce error rate due to severe ISI introduced. Hence, with MLSE, bandwidth compaction can be attained in this way. Other advantages of partial response signalling include higher data rates with fewer levels and error-detecting capabilities without introducing redundancy. Reference [57] gives a good tutorial on this subject.

In 1982, Ungerboeck [62] observed that for error-free transmission, in theory, a QAM system with m symbols allows at a SNR which is 5 dB above the Shannon limit for any modulation scheme; while doubling the channel symbols allows at a SNR which is only 1.2 dB above the Shannon limit. This observation not only interested a lot of information theoretists, but also influenced the trend of practical data transmission systems towards multilevel modulation systems. Recently, channel coding with an expanded signal set has been applied to 9600 bps, full duplex modems operating over the switched telephone network [7].

The complexity of VA increases rapidly with m and v, but hitherto most literatures focused on the effect of v. It seems that only a few researchers noticed that the complexity can be large for large m, even when v is very small [21]. In fact, the effect of m should deserve a lot of attention, as the effect of ISI in multilevel system is more

noticeable [60].

Chapter 2

New Formulation of MLSE

A main disadvantage of VA is the huge memory needed to store all the paths. In practice, the length of input sequence l can be very large or effectively infinite. Besides, in real systems, the detection delay of the whole sequence may be operationally undesirable. Therefore, some modifications must be done in order to make the VA pragmatic.

2.1 The Truncated Viterbi Algorithm

To reduce the complexity of the VA, the trellis is truncated to a manageable depth δ, called the truncation depth, and decision on input symbol $x_{i-\delta}$ is made at time i. After the decision, path history at or before time $i - v$ are discarded. Then the path metrics of the nodes at time $i + 1$ and the associated survivor paths are computed. In this way, only δm^v bits are needed to store the path history. Such modified algorithm is called the truncated Viterbi algorithm (TVA). The space complexity of the TVA becomes $(\delta \log(m) + p)m^v$ bits (refer to section 1.4). Here a p-bit path metric is assumed. Figure 2.1 shows the details of the TVA with $\delta = 2$. For the ease of comparison, the example in figure 1.5 is considered. Notice that the node associated with the shortest survivor is released and all path history at or before time

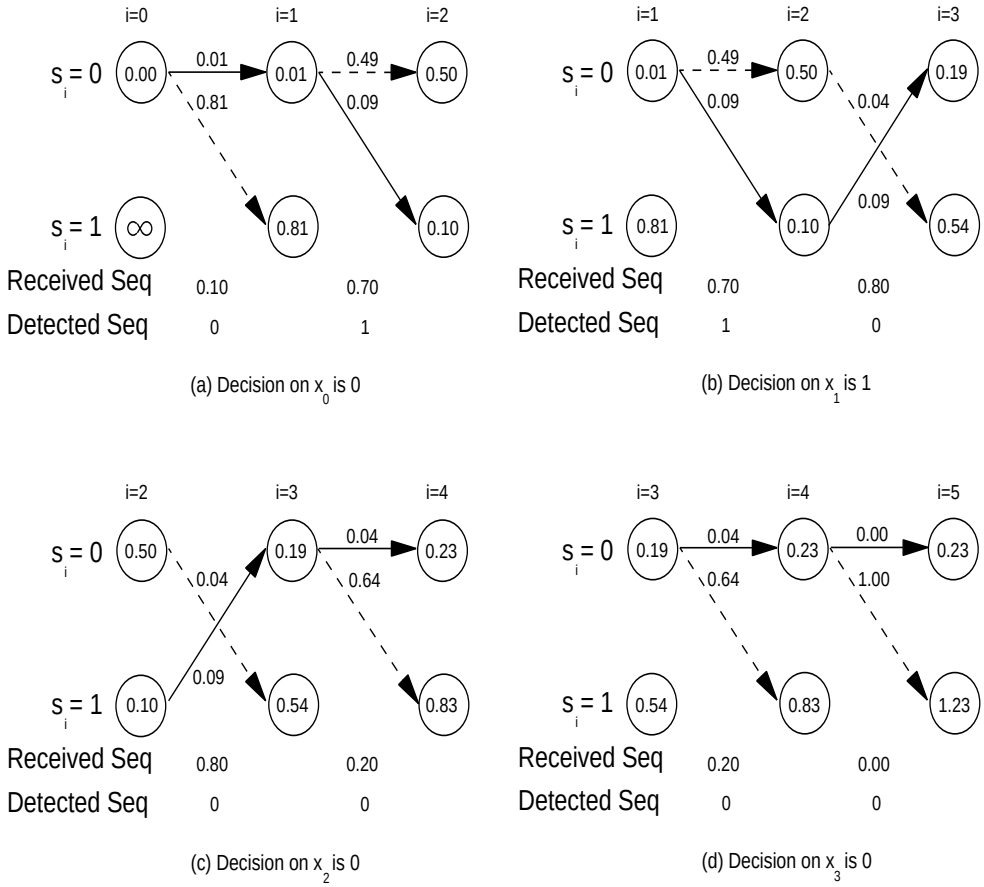

Figure 2.1: The truncated Viterbi Algorithm with a truncation depth of 2 is applied to the example in figure 1.5.

i is retained without re-calculation. In this way, only the path information involving the last stage need to be computed. The time complexity is $O(m^{v+1})$ operations per detected symbol (same as the non-truncated case).

2.2 Choice of Truncation Depth

Forney gave the following comments for the effect of truncation on error performance [28]:

> "In general, if the truncation depth δ is chosen large enough, there is a high probability that all time-k survivors will go through the same nodes up to time $k - \delta$, so that the initial segment of the maximum-likelihood

> path is known up to time $k - \delta$ and can be put out as the algorithm's firm decision; in this case truncation costs nothing. In rare cases when survivors disagree, any reasonable strategy for determining the algorithm's time-$(k - \delta)$ decision will work: choose an arbitrary time-$(k - \delta)$ node, or the node associated with the shortest survivor, or a node chosen by majority vote, etc. If δ is large enough, the effect on performance is negligible."

As the space complexity of the TVA depends on δ, we want to know what value of δ is "large enough" so that no remarkable change on performance is resulted. The exact performance degradation due to truncation is analytically intractable, and is normally found through experimentation and/or simulation [4].

Under the context of decoding convolutional codes, results of extensive simulation showed that the truncation depth of 4 to 5 times the constraint length (corresponding to channel memory length in our context) is large enough [34]. Recently, Leonard and Rodger [53] proved that a non-catastrophic convolutional code can be treated as a k-error-correcting code by using the TVA with $\delta \geq \text{sep}(2k + 1)$, where $\text{sep}(k)$ is one plus the length of the longest error sequence in the received word with a weight smaller than or equal to k. (The term "catastrophic" defined for the convolutional code is similar in meaning to that appeared in definition 1.4.)

In our context, argument similar to Leonard's is valid. Recall that $\mathbf{E}_{d_{min}}$ is the set of error events with minimum Euclidean weight d_{min}. Denote γ as one plus the length of the longest transmitted error sequence with minimum Euclidean weight d_{min}. For non-catastrophic channels, the truncation depth δ should be chosen as γ. This ensures that transmitted error sequences due to error events $\varepsilon \in \mathbf{E}_{d_{min}}$ will be trapped entirely in the path memory of the TVA. As a result, the truncation will not cause extra detection errors when the weight of the noise components is smaller than $\frac{d_{min}}{2}$. As error events in $\mathbf{E}_{d_{min}}$ occur most frequently and determine the error performance of the VA (refer to section 1.5), such choice of the truncation depth

should cause little degradation in the error performance of the estimator.

However, in a truncated trellis, it is possible for some truncated transmitted error sequences, consisting of the first δ elements, to have a weight even smaller than d_{min}. These truncated sequences may degrade the error performance of a TVA. For example, consider the PAM system with $h = (1, -1.5, 0.8)$ and $m = 4$. The minimum distance error event $(1, 2, 2, 1)$ causes a transmitted error sequence $(1, 0.5, -0.2, -0.4, 0.1, 0.8)$ with a weight 1.4491. Another error event $(1, 1, 1, 1, 1, 1)$ causes a transmitted error sequence (1, −0.5, 0.3, 0.3, 0.3, 0.3, 0.3, −0.7, −0.8) with a weight 1.6823. But, for a truncation depth of 7, the truncated transmitted error sequence $(1, -0.5, 0.3, 0.3, 0.3, 0.3, 0.3)$ has a weight 1.3038, which is even smaller than the minimum weight 1.4491.

Proposition 2.1 *For non-catastrophic channels and fixed m, if the truncation depth δ is chosen such that*

1. *δ is greater than one plus the length of the longest transmitted error sequence with minimum Euclidean weight d_{min},*

2. *there does not exist a truncated transmitted error sequence of length δ with weight smaller than d_{min},*

then as SNR $\to \infty$ (or for large enough SNR), the probability of error event $P(\mathcal{E}) \to KQ(\frac{d_{min}}{2\sigma})$ as in the non-truncated case, where the constant

$$K = \sum_{\varepsilon \in \mathcal{E}_{d_{min}}} (\prod_{i=0}^{k} \frac{m - |e_i|}{m}).$$

Proof: Using the same argument in obtaining equation 1.1, for large enough SNR, the error performance of the algorithm is dominated by the minimum distance error events. Now conditions 1 and 2 in the above proposition imply that no error event having a weight smaller than d_{min} exists. Therefore, the performance is the same as that of the non-truncated case.

Q.E.D.

For catastrophic channels, truncation depth of 4 to 5 times the channel memory length should be chosen.

Finally, we stress that the choice of the truncation depth is important since both the space and time complexity of the algorithms to be proposed are quite sensitive to it.

2.3 Decomposition of MLSE

We have described how the VA can be truncated such that the memory requirement is significantly reduced with little loss in performance. Define a k-dimensional MLSE as one with truncation depth k. Then a TVA with truncation depth δ can be viewed as a way to implement a δ-dimensional MLSE, in which the path metrics are initialized according to the calculation of the previous MLSE (refer to figure 2.1). In general, if the performance is dominated by the minimum distance error events, the effect of truncation should be negligible independent of the way to implement the δ-dimensional MLSE. Based on this argument, we introduce a "natural" way to decompose a l-dimensional MLSE into l δ-dimensional MLSE's.

Let us consider a l-dimensional MLSE which finds the detected sequence $\hat{x} = (\hat{x}_l, \hat{x}_{l-1}, \cdots, \hat{x}_1)$ from the received sequence $z = (z_{l+v}, z_{l+v-1}, \cdots, z_1)$. If the truncation depth δ is chosen such that it satisfies proposition 2.1, most survivor paths with small path metric at stage δ will merge together at the first stage and all symbols associated with the merged path can be decided. Thus assuming that all survivor paths originated from a single known node associated with the previous decisions should cause little loss in performance. Let the feedback sequence $(\hat{x}_{i-1}, \hat{x}_{i-2}, \cdots, \hat{x}_{i-v})$ be the v detected symbols before x_i. To estimate x_i, we only need a δ-dimensional MLSE to find a detected sequence $(\tilde{x}_{i+\delta-1}, \tilde{x}_{i+\delta-2}, \cdots, \tilde{x}_i)$ from the received sequence $(z_{i+\delta-1}, z_{i+\delta-2}, \cdots, z_i)$ with decision feedback from $(\hat{x}_{i-1}, \hat{x}_{i-2}, \cdots, \hat{x}_{i-v})$. Then the

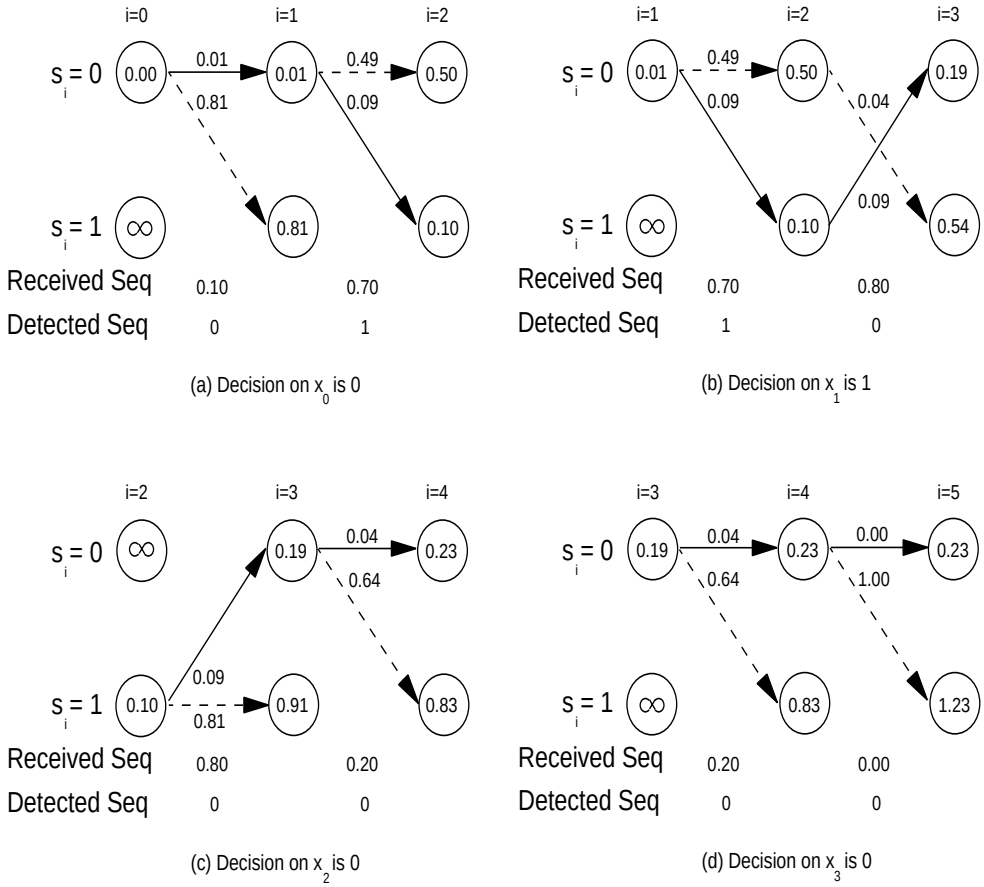

Figure 2.2: Decompostion of a 4-dimensional MLSE into 4 two-dimensional MLSE's is applied to the example in figure 1.5, where each two-dimensional MLSE is implemented using the VA.

symbol $\tilde{x}_i$ estimated is released as the detected symbol $\hat{x}_i$. To estimate the next symbol x_{i+1}, the estimator then find the detected sequence $(\tilde{x}_{i+\delta}, \tilde{x}_{i+\delta-1}, \cdots, \tilde{x}_{i+1})$ from the received sequence $(z_{i+\delta}, z_{i+\delta-1}, \cdots, z_{i+1})$ with the new feedback sequence $(\hat{x}_i, \hat{x}_{i-1}, \cdots, \hat{x}_{i-v+1})$. The symbol $\tilde{x}_{i+1}$ estimated is released as the detected symbol $\hat{x}_{i+1}$. In this way, the detected symbol sequence is found symbol by symbol. We say that the estimator is operating in an incremental mode. As a result, we can apply a single δ-dimensional MLSE to detect a sequence of l symbols in l passes.

As an example, the decompostion of a 4-dimensional MLSE into 4 two-dimensional MLSE's is shown in figure 2.2, where each two-dimensional MLSE is implemented using the VA. Again the example in figure 1.5 is considered. A comparsion of figure 2.2

and figure 2.1 clarifies the difference between the way to decompose a MLSE introduced in this section and the TVA as described in section 2.1. For the latter, history of all survivor paths are feedbacked while a single detected path is feedbacked in the former case. The decomposition of a MLSE should cause little performance degradation since the error performance is still dominated by the minimum distance error events, provided that the truncation depth is chosen according to proposition 2.1.

Proposition 2.2 *If the time and space complexities of a l-dimensional MLSE are $f_1(l)/l$ operations per symbol and $f_2(l)$ storage elements respectively, the decomposition approach reduces the time complexity to $f_1(\delta)$ operations per symbol, and the space complexity to $f_2(\delta)$ storage elements.*

The VA is well-known for enabling the complexity of MLSE to grow linearly with the sequence length instead of exponentially. But we arrive at the following surprising result.

Observation 2.1 *Even without the introduction of the VA, the decomposition approach enables the complexity of a MLSE to grow linearly with the sequence length.*

Finally, we emphasize that our way to decompose a MLSE into lower dimensional MLSE's is quite general and does not depend on how the decomposed MLSE's are implemented.

2.4 Lattice Interpretation of MLSE

Let us consider a δ-dimensional MLSE and its matrix formulation. Recall that h is the channel impulse response, z the received sequence, x the source symbol sequence, $\tilde{x}$ the suspected source symbol sequence. Let $\hat{x}_{i-1}, \hat{x}_{i-2}, \cdots, \hat{x}_{i-v}$ be the detected values of source symbols $x_{i-1}, x_{i-2}, \cdots, x_{i-v}$ respectively. Define the v-dimensional column vector

$$\hat{x} = \begin{pmatrix} \hat{x}_{i-1} \\ \vdots \\ \hat{x}_{i-v+1} \\ \hat{x}_{i-v} \end{pmatrix}$$

and the two δ-dimensional column vectors

$$\tilde{x} = \begin{pmatrix} \tilde{x}_{i+\delta-1} \\ \vdots \\ \tilde{x}_{i+1} \\ \tilde{x}_{i} \end{pmatrix}$$

and

$$z = \begin{pmatrix} z_{i+\delta-1} \\ \vdots \\ z_{i+1} \\ z_{i} \end{pmatrix}.$$

Note that the symbols $\hat{x}$, $\tilde{x}$ and z represent both sequences and the corresponding vectors. Also, define the $\delta \times \delta$ upper-triangular Toeplitz matrix

$$H = \begin{pmatrix} h_0 & h_1 & . & h_v & 0 & . & . & 0 \\ 0 & h_0 & . & h_{v-1} & h_v & . & . & 0 \\ . & . & . & . & . & . & . & . \\ . & . & 0 & h_0 & h_1 & . & h_v & 0 \\ . & . & . & 0 & h_0 & . & h_{v-1} & h_v \\ . & . & . & . & 0 & . & . & h_{v-1} \\ . & . & . & . & . & . & . & . \\ 0 & 0 & . & . & . & . & 0 & h_0 \end{pmatrix}$$

and the $\delta \times v$ lower-triangular Toeplitz matrix

$$G = \begin{pmatrix} 0 & . & . & . & 0 \\ . & . & . & . & . \\ 0 & . & . & . & . \\ h_v & 0 & . & . & . \\ h_{v-1} & h_v & 0 & . & . \\ . & . & . & . & . \\ . & . & . & . & 0 \\ h_1 & h_2 & . & . & h_v \end{pmatrix}.$$

The function of a MLSE is to find $\tilde{x}$ such that the Euclidean distance

$$\|h * (\tilde{x}\ \hat{x}) - z\|^2 = \left\| (H\ G) \begin{pmatrix} \tilde{x} \\ \hat{x} \end{pmatrix} - z \right\|^2 = \|H\tilde{x} - (z + G\hat{x})\|^2 \tag{2.1}$$

is minimized, subject to the constraint that each element of $\tilde{x}$ must be an integer in the range from 0 to $m-1$. The term $h*(\tilde{x}\ \hat{x})$ is in fact the suspected transmitted sequence, obtained by the convolution of the channel impulse response h and the suspected source sequence $(\tilde{x}\ \hat{x})$. Hence the weight measures the negative log likelihood or the distance between the received sequence z and the suspected transmitted sequence corresponding to $\tilde{x}$ and the feedback sequence $\hat{x}$.

It is observed that the set of all possible $\tilde{x}$ comprises a finite integral lattice $\{0, 1, \cdots, m-1\}^{\delta}$. (Intuitively, a lattice can be thought as the set of cross-points in a grid. Refer to section 3.1 for the formal definition of a lattice.) Since the linear transform of a finite lattice is again a finite lattice, $H\tilde{x}$ must be a point in a finite lattice. Define $q = z + G\hat{x}$. Then the sequence estimation problem can be interpreted as the following nearest (finite) lattice point problem.

Problem 2.1 *Given a δ-vector q, find a δ-vector $\tilde{x}$ in the lattice $H\{0, 1, \cdots, m-1\}^{\delta}$ such that the vector $\tilde{x}$ is closest to the vector q in Euclidean distance.*

As an example, consider the two-dimensional MLSE in figure 2.2(a), where $v = 1$, $m = 2$, $\delta = 2$, $h = (1, 0.5)$, $\hat{x} = (0)$ and $z = (0.7, 0.1)$. Then, according to equation 2.1, the distance to be minimized is

$$\left\| \begin{pmatrix} 1 & 0.5 & 0 \\ 0 & 1 & 0.5 \end{pmatrix} \begin{pmatrix} \tilde{x}_{i+1} \\ \tilde{x}_i \\ 0 \end{pmatrix} - \begin{pmatrix} 0.7 \\ 0.1 \end{pmatrix} \right\|^2 = \left\| \begin{pmatrix} 1 & 0.5 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \tilde{x}_{i+1} \\ \tilde{x}_i \end{pmatrix} - \begin{pmatrix} 0.7 \\ 0.1 \end{pmatrix} \right\|^2 .$$

The suspected source sequence $\tilde{x}$ must be one of the four ordered pairs (0,0), (0,1), (1,0) and (1,1). These four points comprise a two-dimensional finite lattice as shown in figure 2.3. The nearest lattice point of the query point (0.7,0.1) corresponds to $\tilde{x} = (1, 0)$. Hence the decision on x_i is 0. Notice that the estimation process just described is exactly what the two-dimensional MLSE in figure 2.2(a) has done.

Definition 2.1 *The Voronoi region of a lattice point b is the region in which all points has b as their nearest lattice point.*

If we know the Voronoi region of a lattice point, the nearest lattice point problem is equivalent to detecting which Voronoi region the given point belong to. Due to the regular structure of a lattice, most of the nearest neighbor regions have the same shape. Therefore it is unnecessary to search exhaustively all lattice points for the nearest one. This is the key to our efficient algorithms for the MLSE.

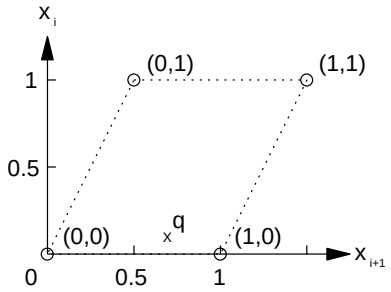

Figure 2.3: The lattice interpretation of the two-dimensional MLSE in figure 2.2(a), where o: lattice points corresponding to the permissible source sequences labelled by the ordered pairs , x: the query point q. The nearest lattice point corresponds to $\tilde{x} = (1, 0)$.

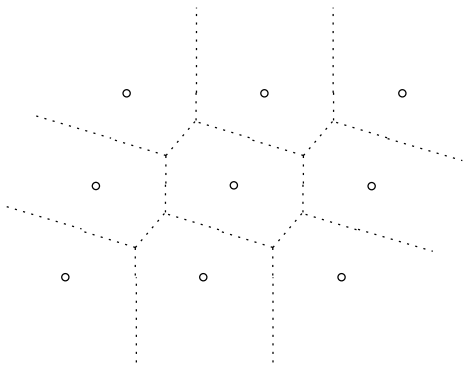

Figure 2.4: The Voronoi regions of a two-dimensional finite lattice.

Nevertheless, near the boundary of a finite lattice, the regularity breaks down. In other words, the Voronoi regions of lattice points near the boundary have irregular shapes. Even the accurate descriptions of such regions are very complicated. Figure 2.4 illustrates the Voronoi regions of a two-dimensional finite lattice.

To avoid such complication arising from the finite lattice, we simply assume the lattice is infinite and the Voronoi regions of all lattice points have the same shape. The error caused by the finiteness of lattice is referred to as the boundary error. As the deviation of vector q from a lattice point is caused by AWGN, with large enough SNR and m, the probability that the nearest lattice point of q is a point outside the finite lattice is small. Therefore, in many important cases, the assumption is reasonable and the boundary error is negligible.

With this assumption, all points in the lattice $H\mathbf{Z}^{\delta}$ is permissible, where $\mathbf{Z}$ is the

set of integers. From now on, we consider the following nearest lattice point problem (also called the closest vector problem) instead.

Problem 2.2 (The Closest Vector Problem) *Given a δ-vector q, find a δ-vector $\tilde{x}$ in the lattice $H\mathbf{Z}^{\delta}$ such that the vector $\tilde{x}$ is closest to the vector q in Euclidean distance.*

Chapter 3

Closest Vector Problem

The closest vector problem (CVP) (also called the nearest lattice point problem) is a special case of an important algorithmic problem studied in computational geometry — the post-office problem (POP) (also known as the nearest neighbor search or the closest-point queries).

Computational geometry is, in its broadest sense, the study of geometrical problems from a computational point of view. It finds a large number of applications areas such as pattern recognition, computer graphics, image processing, operations research, statistics, computer-aided design, robotics, etc. For a excellent survey on this subject, refer to reference [47] (see also [56]).

The POP is a fundamental problem of computational geometry, having many applications in statistics, operations research, interactive graphics, pattern recognition, coding theory and other areas.

Problem 3.1 (The Post Office Problem) *Given a set of k points, called sites, in Euclidean n-dimensional space, build a data structure so that for any arbitrary query point q, a closest point in the given set to q can be found quickly.*

The problem arose originally in two and three dimensional Euclidean spaces, in which real physical objects are arranged. Several asymptotically fast algorithms are

known for this problem in the planar ($n = 2$) case. They involve the construction of the Voronoi diagram (a diagram consisting of the Voronoi regions of all k points), and the use of fast methods for searching planar subdivisions resulting from that diagram [26], [59], [40], [18]. A different approach is to solve the problem by finding extreme points in three dimensions [19]. By these methods, a data structure requiring $O(k)$ space can be constructed in $O(k \log(k))$ time, so that a query can be answered in $O(\log(k))$ time. Chazelle [8] has given an algorithm for the case $n = 3$ that requires $O(k^2)$ preprocessing for $O(\log^2(k))$ query time.

The high-dimensional cases are much less examined and understood. Dobkin and Lipton [16] have described a data structure requiring $O(k^{2^{n+1}})$ time and space to construct, giving a query time of $O(\log(k))$. Assuming a certain probability distribution of the sites, several randomized algorithms were proposed to improve the average complexity. If the sites is uniformly distributed in a hypercube [3], or spatially Poisson-distributed [31], their Voronoi diagram has linear average complexity. Recently, Clarkson [11] gave an algorithm which requires $O(k^{\lceil n/2 \rceil (1+\epsilon)})$ preprocessing on the average, $O(k^{\lceil n/2 \rceil (1+\epsilon)})$ space in the worst case and $O(\log(k))$ query time, for any fixed $\epsilon > 0$. This is close to the optimal performance since in the worst case the Voronoi diagram may require $\Omega(k^{\lceil n/2 \rceil})$ storage [41].

The CVP is a POP with the sites forming a lattice. The complexities of the above algorithms are measured in terms of k, the number of given points. These algorithms do not exploit the regular structure of a lattice, and must be inefficient. Therefore, a totally different approach must be used although some of the techniques mentioned may be useful.

The Voronoi regions of all lattice points are of the same shape. However, even storing a single Voronoi region is inefficient, as the storage required to store such region already increases exponentially with the dimension. It is thus desirable to find a solution without involving the Voronoi region.

Very efficient CVP algorithms [14, chap. 20], [12], [13] 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 lattice quantizer and source coding for Gaussian channels. Nonetheless, they cannot be generalized to solve the problem for arbitrary lattices.

A general solution for the CVP was proposed by Kannan [37], [38]. It is related to other lattice algorithms and was originally developed for solving integer programming problems. Kannan was mainly interested in the theoretical bound on the worst-case complexity. Hence his CVP algorithm is very complicated and is not a practical solution to our problem. However the underlying idea is simple and is consisted of two steps:

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

Step 2: Enumerate all lattice points falling inside a sphere, which is centered at the query point, for the nearest lattice point.

The procedure transforming a basis into a reduced one is known as the basis reduction algorithm, while the one achieving the second step is called the enumeration algorithm.

The idea of our CVP algorithms are similar. After a brief introduction to the lattice, we will discuss the basis reduction algorithm and the enumeration algorithm. Then a CVP algorithm is developed based on the straightforward reduce-and-enumerate approach. Along with the discussions, we suggest improvements for almost all algorithms encountered. Finally, very efficient CVP algorithm is derived using the technique of norm approximation.

3.1 Basic Definitions and Facts About Lattices

Let n be a positive integer. A subset L of $\mathbf{R}^n$ is called a lattice if there exist linearly independent n-vectors $b_1, \cdots, b_n \in \mathbf{R}^n$ 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 $n \times n$ matrix. We say that $b_1, \cdots, b_n$ is a basis of L. The same lattice L may have many bases but they have the same determinant (up to sign). So we define the determinant of a lattice $L(B)$ as $\det(L) = |\det(B)|$. Geometrically, the determinant of a lattice is the common content of those parallelepipeds whose vertices are lattice points and which contain no other lattice point; equivalently, of those parallelepipeds spanned by bases. Hence the following inequality, called Hadamard's inequality, is natural from a geometric point of view:

$$\|b_1\| \cdots \|b_n\| \geq \det(L). \tag{3.1}$$

We recall the Gram-Schmidt orthogonalization process. The n-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

$$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{ij} b_j^*, \tag{3.2}$$

$$\mu_{ij} = \frac{b_i^T b_j^*}{\|b_j^*\|^2}. \tag{3.3}$$

The orthogonal vectors $b_1^*, \cdots, b_n^*$ obtained in this way depend on the order of $b_1, \cdots, b_n$. Also, for all $1 \leq i \leq n$, $b_1^*, \cdots, b_i^*$ and $b_1, \cdots, b_i$ span the same subspace. For all $1 \leq i \leq n$, by defining $\mu_{ii} = 1$, we have

$$b_i = \sum_{j=1}^{i} \mu_{ij} b_j^*. \tag{3.4}$$

or, in matrix form,

$$[b_1, \cdots, b_n] = [b_1^*, \cdots, b_n^*][\mu_{ij}]^T.$$

Note that $[\mu_{ij}]$ is an lower triangular matrix with each diagonal element equal to one. It is obvious that

$$\det(L(B)) = \prod_{i=1}^{n} \|b_i^*\|. \tag{3.5}$$

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

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

Note that $b_i(i) = \|b_i^*\|$ for all i. Besides, for $1 \leq j \leq i \leq n$, define $b(i,j)$ as the projection of b_i on the orthogonal complement of the subspace spanned by $u_1, \cdots, u_{j-1}$, or mathematically,

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

Conceptually, it is useful to think of $b_1, \cdots, b_n$ as being represented in a coordinate system with unit vectors $u_1, \cdots, u_n$. In this coordinate system, the $n \times n$ matrix with the basis vectors as its rows is lower triangular and has $b_i(j)$ as its (i,j)-th entry.

$$\begin{pmatrix}
b_1(1) & 0 & 0 & . & . & . & . & 0 \\
b_2(1) & b_2(2) & 0 & . & . & . & . & 0 \\
. & . & . & . & . & . & . & . \\
. & . & . & . & b_i(i) & 0 & . & . \\
. & . & . & . & b_{i+1}(i) & b_{i+1}(i+1) & . & . \\
. & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & 0 \\
b_n(1) & b_n(2) & . & . & . & . & . & b_n(n)
\end{pmatrix}$$

This is the lower triangular representation of the basis matrix introduced by Kannan [38].

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 $b - \frac{b^T a}{\|a\|^2}a$, which can also be interpreted as the projection of b perpendicular to a. Suppose a is a non-zero vector in L and L_a is the projection of L perpendicular to

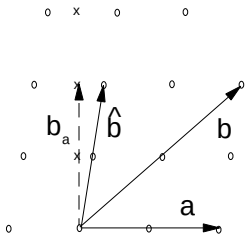

Figure 3.1: An two-dimensional example of projecting b perpendicular to a to get b_a and then lifting b_a to $\hat{b}$, where o: lattice points in L, x: lattice points in L_a.

a. If b_a is a vector in L_a, there is a unique vector $\hat{b}$ in L such that $\hat{b}$ projects into b_a and $-\frac{\|a\|}{2} < \frac{\hat{b}^T a}{\|a\|} \leq \frac{\|a\|}{2}$. The process is called lifting b_a to $\hat{b}$. In fact, $\hat{b} = b_a - \eta a$, where η is the integer nearest to $\frac{b_a^T a}{\|a\|^2}$. Figure 3.1 illustrates an two-dimensional example of projecting b perpendicular to a to get 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.

3.2 Lattice Basis Reduction

The concept of basis reduction has been proposed more than a century ago. The early work on the topic is 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 reducedness 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 element that can be extended to a basis with $(b_1, \cdots, b_{i-1})$. In simple words, Minkowski-reduced bases require that each basis vector is as short as possible. The definition of Minkowski-reduced bases is of fundamental importance in the geometry of numbers [6].

Basis reduction is naturally associated with the problem of finding the shortest lattice vector — the shortest vector problem (SVP). The SVP in the case of L_∞-norm is known to be NP-hard [20]. But it is not clear whether the SVP in the case

of Euclidean norm is NP-hard or not.

In 1982, Lenstra, Lenstra and Lovasz [51] achieved a breakthrough by constructing LLL-reduced bases and a polynomial reduction algorithm, 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 the SVP and the CVP, improved Hermite-reduction and Minkowski-reduction algorithms are developed. In fact, their algorithm has found applications in very wide areas, including integer programming [52], [38], finding irreducible factors of polynomials [50], minimal polynomials of algebraic numbers [39], simultaneous diophantine approximation [51], ellipsoid method in linear programming [32], attacks on knapsack-based crypto-systems [46], [58], disproof of Mertens' century-old conjecture in number theory. All these applications were made possible by the LLL-reduction algorithm.

References [51], [54] have described the LLL-reduction algorithm and proved its correctedness and polynomial complexity. In the following, we will derive the reduction algorithm based on our understanding and interpretations. Unlike most literatures on algorithms, we try to explain how the algorithm is discovered instead of simply listing the algorithm and giving relevant proofs. We believe that this approach will illuminate the spirit of the algorithm. After that we suggest improved versions of the algorithm based on new observations.

3.2.1 Weakly Reduced Bases

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

Think of the lower triangular representation 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 $\bar{b}_i = b_i - rb_j$, 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 $\frac{(n-1)(n-2)}{2}$ times for all $1 < j < i \leq n$. The resultant basis having shorter basis vectors is called weakly reduced.

In summary, a basis $b_1, \cdots, b_n$ is weakly reduced if $|\mu_{ij}| \leq \frac{1}{2}$ for $1 < j < i \leq n$. Remember that $\mu_{ii} = 1$ and $\mu_{ij} = 0$ for $j > i$. Any basis can be converted into a weakly reduced basis by the procedure Weakly_Reduce.

Procedure Weakly_Reduce($b_1, \cdots, b_n$)

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(μ_{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.

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 the procedure is interpreted as projecting and lifting operations of the basis vectors.

3.2.2 Derivation of the LLL-reduction Algorithm

Given a basis and a specific orthogonalization $b_1^*, \cdots, b_n^*$, we can always apply procedure Weakly_Reduce to transform it into a nice basis. One may then naturally ask how to find a nice orthogonalization of a given basis.

By equation 3.5, the length of b_i^*'s are constrainted. Intuitively, we want the length of b_i^*'s to be distributed as even as possible so that the basis appears to be more "short". Lovasz [54] observed that typically the short vectors among $b_1^*, \cdots, b_n^*$ are at the end of the sequence. So it is desirable to make the orthogonalization sequence $b_1(1), \cdots, b_n(n)$ lexicographically as small as possible.

For some $i < n$, consider the length of the projections of b_i and b_{i+1} on $u_i, \cdots, u_n$, i.e. $b(i,i)$ and $b(i+1,i)$. If $b(i,i)$ is longer than $b(i+1,i)$, 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 the swapping of basis vectors, the basis may not be weakly reduced any more. We can then apply procedure Weakly_Reduce again. It is now clear that the two processes, i.e. finding a better basis for a given orthogonalization and finding a better orthogonalization for a given basis, can be applied alternatively until we can do nothing better. This is the spirit of the LLL-reduction algorithm. In this way, the following reduction algorithm can be derived easily.

Algorithm LLL_Reduce0($b_1, \cdots, b_n$)

Step 1 Make the given basis weakly reduced.

Step 2 Check if there exists any i such that $\|b(i,i)\|^2 > \frac{4}{3}\|b(i+1,i)\|^2$. If found, swap b_i and b_{i+1}, update the orthogonalization, and go to step 1. Otherwise, stop.

Note that a weaker test $\|b(i,i)\|^2 > \frac{4}{3}\|b(i+1,i)\|^2$ is used in step 2 instead of $\|b(i,i)\| > \|b(i+1,i)\|$ to ensure faster convergence. (The convergence of the algorithm will be proved later). The coefficient $\frac{4}{3}$ is chosen arbitrarily and may be replaced by

any number slightly greater than 1. This suggests the following definition of reduced basis.

Definition 3.1 *A basis $b_1, \cdots, b_n$ of a lattice is LLL-reduced if it is weakly reduced and for $1 \leq i < n$,*

$$\|b(i,i)\|^2 \leq \frac{4}{3}\|b(i+1,i)\|^2. \tag{3.6}$$

Like any iterative algorithm, we need to guarantee its termination. Let $\bar{b}_i^*, \cdots, \bar{b}_{i+1}^*$ be the orthogonalization vectors after swapping. 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 $\alpha < \sqrt{\frac{3}{4}}$ 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 equation 3.5, $\|\bar{b}_i^*\|\|\bar{b}_{i+1}^*\| = \|b_i^*\|\|b_{i+1}^*\|$. 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 following function

$$D(b_1, \cdots, b_n) = \prod_{j=1}^{n} \prod_{k=1}^{j} \|b_k^*\|^2 = \prod_{k=1}^{n} \|b_k^*\|^{2(n-k)}. \tag{3.7}$$

The function can be interpreted as a negative measure of achievable 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, we can establish the upper and lower bounds for the value of D.

Lemma 3.1 ([51]) *Let $b_1, \cdots, b_n$ be a basis of an integer lattice L, then*

$$1 \leq D(b_1, \cdots, b_n) \leq (\max_i \|b_i\|^2)^{n(n-1)/2}.$$

Proof: Let $\beta = \max_i \|b_i\|^2$, and let $g_j = \prod_{k=1}^{j} \|b_k^*\|^2$. g_j is the squared determinant of the j-dimensional lattice spanned by $b_1, \cdots, b_j$. By Hadamard's inequality 3.1,

$$g_j \leq \prod_{k=1}^{j} \|b_k\|^2 \leq \beta^j.$$

Hence, the upper bound of D follows from its definition. It can be seen that

$$g_j = \det([b_1, ..., b_j]^T [b_1, ..., b_j]).$$

This is an integer greater than zero since all b_i's are integer vectors. Thus the lower bound follows easily. Q.E.D.

Though this lemma only applies to integer lattice, in practice, its result may be extended to real numbers in fix-point representation by proper scaling. Without loss of generality, from now on, integer lattice is assumed. We will prove a useful lemma, and then discuss the nice properties of a LLL-reduced basis.

Lemma 3.2 ([54]) *Let $b_1, \cdots, b_n$ be a basis of a lattice L and let $b_1^*, \cdots, b_n^*$ be its Gram-Schmidt orthogonalization. Denote $\lambda(L)$ as the length of the shortest vector in L. Then*

$$\lambda(L) \geq \min(\|b_1^*\|, \cdots, \|b_n^*\|).$$

Proof: Let b be the shortest non-zero vector in L. Then we can write $b = \sum_{i=1}^{k} \eta_i b_i$, where $1 \leq k \leq n$, all η_i's are integers and $\eta_k \neq 0$. Substituting from equation 3.4, we

have $b = \sum_{i=1}^{k} \alpha_i b_i^*$. As $\mu_{kk} = 1$, $\alpha_k = \eta_k$ is a non-zero integer. Thus

$$\|b\|^2 = \sum_{i=1}^{k} \alpha_i^2 \|b_i^*\|^2 \geq \alpha_k^2 \|b_k^*\|^2 \geq \|b_k^*\|^2.$$

Q.E.D.

Theorem 3.1 ([54]) *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. $\|b_1\| \leq 2^{(n-1)/2}\lambda(L)$;
2. $\|b_1\| \leq 2^{(n-1)/4} \det(L)^{1/n}$;
3. $\|b_1\| \cdots \|b_n\| \leq 2^{n(n-1)/4} \det(L)$.

Proof: Assuming our usual notation,

$$\begin{aligned} \|b_i^*\|^2 = \|b(i,i)\|^2 &\leq \frac{4}{3}\|b(i+1,i)\|^2 \\ &= \frac{4}{3}\|b_{i+1}^* + \mu_{i+1,i} b_i^*\|^2 \\ &= \frac{4}{3}\|b_{i+1}^*\|^2 + \frac{4}{3}\mu_{i+1,i}^2 \|b_i^*\|^2 \\ &\leq \frac{4}{3}\|b_{i+1}^*\|^2 + \frac{1}{3}\|b_i^*\|^2, \end{aligned}$$

and hence

$$2\|b_{i+1}^*\|^2 \geq \|b_i^*\|^2. \tag{3.8}$$

By induction,

$$2^{i-1}\|b_i^*\|^2 \geq \|b_1^*\|^2 = \|b_1\|^2, \tag{3.9}$$

and by lemma 3.2,

$$\begin{aligned} \|b_1\|^2 &\leq \min_i (2^{i-1}\|b_i^*\|^2) \\ &\leq 2^{n-1} \min_i \|b_i^*\|^2 \\ &\leq 2^{n-1}\lambda(L)^2. \end{aligned}$$

This proves part 1.Again by equation 3.9,

$$\begin{aligned} \|b_1\|^{2d} &\leq \prod_{i=1}^{n} 2^{i-1}\|b_i^*\|^2 \\ &= 2^{n(n-1)/2}\prod_{i=1}^{n}\|b_i^*\|^2 \\ &= 2^{n(n-1)/2}\det(L)^2. \end{aligned}$$

Hence part 2 follows. Using the property of weakly reducedness,

$$\|b_i\|^2 = \sum_{j=1}^{i}\mu_{ij}^2\|b_j^*\|^2 \leq \|b_i^*\|^2 + \frac{1}{4}(\|b_{i-1}^*\|^2 + \cdots + \|b_1^*\|^2).$$

By equation 3.8,

$$\|b_i\|^2 \leq \left(1 + \frac{1}{4}(2 + \cdots + 2^{i-1})\right)\|b_i^*\|^2 \leq 2^{i-1}\|b_i^*\|^2$$

and hence

$$\prod_{i=1}^{n}\|b_i\|^2 \leq 2^{n(n-1)/2}\prod_{i=1}^{n}\|b_i^*\|^2 = 2^{n(n-1)/2}\det(L)^2.$$

This proves part 3. Q.E.D.

Parts 1 and 2 of this theorem guarantee that the LLL-reduced basis includes a reasonably short vector while part 3 ensures a quite "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. We now focus on the detailed procedure to achieve 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

$$\begin{pmatrix}
\mu_{11} & 0 & 0 & . & . & . & . & 0 \\
\mu_{21} & \mu_{22} & 0 & . & . & . & . & 0 \\
. & . & . & . & . & . & . & . \\
\boxed{\mu_{k-1,1}} & \boxdot & \boxdot & \boxdot & \boxed{\mu_{k-1,k-1}} & 0 & . & . \\
\boxed{\mu_{k,1}} & \boxdot & \boxdot & \boxdot & \boxed{\mu_{k,k-1}} & \boxed{\mu_{k,k}} & . & . \\
. & . & . & . & \boxdot & \boxdot & . & . \\
. & . & . & . & \boxdot & \boxdot & . & 0 \\
\mu_{d1} & \mu_{d2} & . & . & \boxed{\mu_{n,k-1}} & \boxed{\mu_{n,k}} & . & \mu_{dd}
\end{pmatrix}.$$

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 [51].

$$\begin{aligned}
\bar{b}_{k-1} &= b_k \\
\bar{b}_k &= b_{k-1} \\
\bar{b}_{k-1}^* &= b_k^* + \mu_{k,k-1} b_{k-1}^* \\
\bar{b}_k^* &= b_{k-1}^* - \bar{\mu}_{k,k-1} \bar{b}_{k-1}^* \\
\bar{\mu}_{k,k-1} &= \mu_{k,k-1} \frac{\|b_{k-1}^*\|^2}{\|\bar{b}_{k-1}^*\|^2} \\
\bar{\mu}_{i,k-1} &= \mu_{i,k-1} \bar{\mu}_{k,k-1} - \mu_{ik} \frac{\|b_k^*\|^2}{\|\bar{b}_{k-1}^*\|^2} && \text{for } k < i \leq n \\
\bar{\mu}_{ik} &= \mu_{i,k-1} - \mu_{ik} \mu_{k,k-1} && \text{for } k < i \leq n \\
\bar{\mu}_{k-1,j} &= \mu_{kj} && \text{for } 1 \leq j < k-1 \\
\bar{\mu}_{kj} &= \mu_{k-1,j} && \text{for } 1 \leq j < k-1
\end{aligned}$$

Remember 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 $1 \leq j \leq k-1$, since $b_1^*, \cdots, b_{k-1}^*$ are unchanged. This also implies that only $b_k, \cdots, b_n$ need to be considered to achieve weakly reduced basis in step 1. Besides, in step 2, we have to find an i such that $\|b(i,i)\|^2 > \frac{4}{3}\|b(i+1,i)\|^2$. One straightforward way to implement this is to use a counter k with initial value 2. The counter keeps track of the dimension of sublattice which is LLL-reduced represented by the present basis vectors. 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 a run. However, the counter will reach $n+1$ sooner or later as the algorithm must terminate. The process can be thought as extending the dimension of the sublattice which is LLL-reduced by 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 weakly reduced instead of all basis vectors. Combining with the discussion in previous paragraph, 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 requests 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 [51] can be constructed readily.

Procedure LLL_Reduce($b_1, \cdots, b_n$)

1. Do Gram-Schmidt orthogonalization process to get μ_{ij}'s and β_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 $\beta_k < (\frac{3}{4} - \mu_{k,k-1}^2)\beta_{k-1}$ do
5. $\begin{cases} \text{Swap } b_{k-1} \text{ and } b_k, \text{ swap } \mu_{k-1,1}, \cdots, \mu_{k-1,k-2} \text{ and } \mu_{k,1}, \cdots, \mu_{k,k-2}, \\ \text{and update } \beta_{k-1} \text{ and } \beta_k \text{ , } \mu_{k+1,k-1}, \cdots, \mu_{n,k-1} \text{ and } \mu_{k+1,k}, \cdots, \mu_{n,k}. \end{cases}$
6. If $k > 2$ do $k := k - 1$.
7. Else
8. $\begin{cases} \text{For } j = k-2 \text{ downto } 1 \text{ do make, if necessary, } |\mu_{kj}| \leq \frac{1}{2} \\ \text{and update } b_k \text{ and } \mu_{k1}, \cdots, \mu_{k,j}. \end{cases}$
9. If $k \neq n$ do $k := k + 1$; else terminate.
10. Endif.

11. Go to line 3.

Note that β_i stores the value of $\|b_i^*\|^2$ for $1 \leq i \leq n$.

We now compute the time complexity of procedure LLL_Reduce. The Gram-Schmidt orthogonalization process (see equations 3.2 and 3.3) in line 1 needs $O(n^3)$ operations. Each execution of line 5 decreases the value of function D (defined in equation 3.7) by multiplying a factor of $\frac{3}{4}$. Let D_0 be the initial value of D. By lemma 3.1, 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 $O(n^2)$. 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 $O(n^2)$. Thus each of the lines from 3 to 11 is executed $O(n^2)$ times. Each execution of line 3, line 5 and line 8 take $O(n)$, $O(n)$ and $O(n^2)$ operations respectively. Therefore, the overall time complexity of the algorithm is $O(n^4)$.

As the procedure handles integers, rational arithmetic operations are assumed. It can be proved that all numbers that appear in the course of the algorithm have binary length $O(n\log(\beta))$ [51]. Thus the algorithm is really polynomial. Finally, we remark that only b_i's, $\|b_i^*\|$'s and μ_{ij}'s need to be stored. It is not necessary to store b_i^*'s, as the orthogonalization process in line 1 can be done by the following procedure [36]:

Procedure Orthogonalize$(b_1, \cdots, b_n)$

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.

3.2.3 Improved Algorithm for LLL-reduced Bases

We observed that only the condition $|\mu_{k,k-1}| \leq \frac{1}{2}$ is important to enable further improvement of the orthogonalization, and other μ_{ij}'s are solely used for updating purposes. In fact, the nice property of a LLL-reduced basis summarized in lemma 3.1 also requires the condition $|\mu_{k,k-1}| \leq \frac{1}{2}$ only. This observation suggests that line 8 of procedure LLL_Reduce is totally unnecessary. According to the previous complexity analysis, this line is the bottleneck, which executes $O(n^2)$ operations. Removing it will decrease the computational complexity by an order of magnitude. Now the modified algorithm has time complexity $O(n^3)$. However, the resultant basis may not be weakly reduced and procedure Weakly_Reduce may be employed once to achieve this.

We notice that Lovasz [54] also has this observation but he insists the full strength of weak reduction so as to get the polynomial space complexity. Facing the same situation, we do not want to store the big numbers that may occur in the course of the modified algorithm.

It should be emphasized that we are interested in a practical algorithm, instead of its asymptotic behavior. The most important variables that require exact arithmetics is b_i's. Without loss of generality, integer lattice is assumed (i.e. all b_i's are integer vectors). Kaltofen [36] suggests the use of modulo-M arithmetics for b_i's so as to slightly improve the space complexity and the binary steps required. Let β be the maximum squared length of the given basis. With $|\mu_{ij}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$, then for all i's,

$$\|b_i\|^2 = \|b_i^*\|^2 + \sum_{j=1}^{i-1} \mu_{ij}^2 \|b_j^*\|^2 \leq \frac{i+3}{4}\beta \leq \frac{n+3}{4}\beta.$$

Hence, the updated entries of b_i will be absolutely bounded by $\frac{1}{2}\sqrt{(n+3)\beta}$, and M

can be chosen as any value greater than $\sqrt{(n+3)\beta}$. While Kaltofen considers this in a subroutine, we use the modulo-M arithmetics throughout the whole procedure and reconstruct the true values of b_i's just before termination. This will not cause any error as the operations on b_i's are solely for updating purpose, and the true values of b_i's are not used elsewhere. Besides, right before termination, the condition of weak reducedness guarantees that the values of b_i's are exact. We remark that the reconstruction operations can be omitted if two's complement representation of integer is used.

Though it can be shown that the binary length of β_i's (storing the values of $\|b_i^*\|^2$'s) still grows polynomially, that of μ_{ij}'s is not the case. However, the result of a reduction algorithm is the values of b_i's (the reduced basis), and we do not need the exact values of μ_{ij}'s for most practical purposes. As suggested by Odlyzko, the entries of μ_{ij}'s and β_i's could be made floating point numbers with extended precision. Whenever the loss of significant digits during roundoff becomes too great to decide $\beta_i < (\frac{3}{4} - \mu_{i,i-1}^2)\beta_{i-1}$, or to calculate the integer nearest to μ_{ij}, one can recompute this values from b_i's. We propose the following procedure for this recomputation task.

Procedure Recompute$(b_1, \cdots, b_n)$

1. For $i = 1$ to n do
2. For $j = i - 1$ to 1 do
3. $\mu_{ij} := (b_i^T b_j - \sum_{k=1}^{j-1} \mu_{jk}\mu_{ik}\beta_k)/\beta_j$.
4. If $|\mu_{ij}| > \frac{1}{2}$ do
5. $\eta := \text{round}(\mu_{ij})$.
6. $b_i := (b_i - \eta b_j) \bmod M$.
7. $\mu_{ij} := \mu_{ij} - \eta$.
8. Endif.

9. Endfor.

10. $\beta_i := \|b_i\|^2 - \sum_{k=1}^{i-1} \mu_{ik}^2 \beta_k$.

11. Endfor.

This procedure efficiently combines the orthogonalization process and weak reduction process. The latter process is important since all μ_{ij}'s recomputed is bounded by $\frac{1}{2}$ and thus their accuracy can be greatly increased. With procedure Recompute, we give the following modified LLL-reduction algorithm.

Procedure MLLL_Reduce($b_1, \cdots, b_n$)

1. Do Gram-Schmidt orthogonalization process to get μ_{ij}'s and β_i's.

2. $k := 2$.

3. Make, if necessary, $|\mu_{k,k-1}| \leq \frac{1}{2}$ and update b_k using modulo-M arithmetics and update $\mu_{k1}, \cdots, \mu_{k,k-1}$.

4. If $\beta_k < \frac{1}{2}\beta_{k-1}$ do

5. $\begin{cases} \text{Swap } b_{k-1} \text{ and } b_k, \text{ and swap } \mu_{k-1,1}, \cdots, \mu_{k-1,k-2} \text{ and } \mu_{k,1}, \cdots, \mu_{k,k-2}, \\ \text{and update } \beta_{k-1} \text{ and } \beta_k \text{ , } \mu_{k+1,k-1}, \cdots, \mu_{n,k-1} \text{ and } \mu_{k+1,k}, \cdots, \mu_{n,k}. \end{cases}$

6. If $k > 2$ do $k := k - 1$.

7. Else

8. If $k \neq n$ do

9. $k := k + 1$.

10. Else

11. Make $b_1, \cdots, b_n$ weakly reduced using modulo-M arithmetics.

12. Terminate.

13. Endif.

14. Endif.

15. Go to line 3.

For the ease of computation, in line 4, the slightly stronger test $\beta_k < \frac{1}{2}\beta_{k-1}$ is used to substitute the original test $\beta_k < (\frac{3}{4} - \mu_{k,k-1}^2)\beta_{k-1}$. The two's complement representation of b_i's is assumed here.

According to our interpretation of the LLL-reduction algorithm, it is an two-step iteration: one step improves the basis for fixed orthogonalization, and the other improves the orthogonalization for fixed basis. But these two steps are quite independent. In particular, we do not need the weak reducedness of the basis, or even the condition $|\mu_{k,k-1}| \leq \frac{1}{2}$ in order to improve the orthogonalization. Clearly, it seems that procedure LLL_Reduce makes $|\mu_{ij}| \leq \frac{1}{2}$ too frequently. We can improve the orthogonalization in a single step without making any $|\mu_{ij}| \leq \frac{1}{2}$ until all the b_k's satisfy inequality 3.6. Though this approach reduces the number of times we need to make $|\mu_{kj}| \leq \frac{1}{2}$ for $j = k-1, \cdots, 1$ (as in line 8), it seems to increase the number of swapping and thus the updating of μ_{ij}'s (as in line 5). Therefore the overall improvement is small. Nevertheless, this approach enables larger amount of parallelism and is more suitable for multiprocessing environment. Combining with the features of procedure MLLL_Reduce, we propose the following algorithm.

Procedure PMLLL_Reduce($b_1, \cdots, b_n$)

1. Do Gram-Schmidt orthogonalization process to get μ_{ij}'s and β_i's.

2. $k := 2$; swap_flag:=0.

3. While $k \leq n$ do

4. If $\beta_k < \frac{1}{2}\beta_{k-1}$ do

5. $\begin{cases} \text{Swap } b_{k-1} \text{ and } b_k, \text{ swap } \mu_{k-1,1}, \cdots, \mu_{k-1,k-2} \text{ and } \mu_{k,1}, \cdots, \mu_{k,k-2}, \\ \text{and update } \beta_{k-1},\ \beta_k,\ \mu_{k+1,k-1}, \cdots, \mu_{n,k-1} \text{ and } \mu_{k+1,k}, \cdots, \mu_{n,k}. \end{cases}$

6. swap_flag:=1.

7. If $k > 2$ do $k := k - 1$.

8. Else

9. $k := k + 1$.

10. Endif.

11. Endwhile.

12. If swap_flag $= 1$ do

13. $\begin{cases} \text{For } j = 2 \text{ to } n \text{ do make, if necessary, } |\mu_{j,j-1}| \leq \frac{1}{2} \text{ and update } b_j \\ \text{using modulo-}M \text{ arithmetics and update } \mu_{j1}, \cdots, \mu_{j,j-1}. \end{cases}$

14. Else

15. Make the basis $b_1, \cdots, b_n$ weakly reduced using modulo-M arithmetics.

16. Terminate.

17. Endif.

18. Go to line 2.

Finally, we remark that procedure MLLL_Reduce is equivalent to procedure LLL_Reduce, in the sense that they always return the same reduced basis for the same input basis, while procedure PMLLL_Reduce may find a reduced basis different from that found by procedure LLL_Reduce even for the same input.

3.3 Enumeration Algorithm

As mentioned in the beginning of this chapter, it is necessary to enumerate all vectors of a lattice falling inside a certain region in order to find the vectors of small length. Consequently, enumeration algorithm become an essential component of the SVP and CVP algorithms. To avoid enumerating unnecessarily large number of points for the shortest vector(s), the following two steps are usually done.

Step 1: Determine a reasonably small region (or the radius of a sphere in case of Euclidean distance) which must contain the shortest vector(s).

Step 2: Enumerate all vectors in that region and pick the shortest one(s).

We first discuss how to achieve step 2, assuming the radius of the sphere to be enumerated is given. There are three different ways to do enumeration. Each method has a different computational complexity and ease of implementation. Their computational complexities can be compared by the number of points to be enumerated. We will survey these enumeration algorithms and at the same time introduce a unified treatment of them. Each of them corresponds to enumerating points in a region of a specific shape. After that, we will describe how to achieve step 1 and suggest some improved algorithms based on our geometric interpretation.

3.3.1 Lattice and Isometric Mapping

A n-dimensional lattice $L = L(B)$ in a n-dimensional Euclidean space $M_L = (L, d)$ is the set of all integer linear combinations of the column vectors of the basis matrix $B = [b_1, \cdots, b_n]$. Any vector a in L has a length of $(a, a)^{1/2}$, where $(\cdot, \cdot)$ stands for the scalar product operator. As $L = B\mathbf{Z}^n$ is the linear transform of $\mathbf{Z}^n$ and B is invertible, B is an isometric mapping which maps $\mathbf{Z}^n$ into L and B^{-1} is the inverse mapping. Hence each point in $L(B)$ has a one-to-one correspondence in $\mathbf{Z}^n$. The latter lattice is associated with a metric space $M_Z = (\mathbf{Z}^n, d_Z)$, where $d_Z(a, a) = (Ba, Ba)^{1/2}$. For

example, a vector $a \in L$ with length $(a, a)^{1/2}$ corresponds to a vector $a_Z = B^{-1}a$ with length (Ba_Z, Ba_Z). Note that a and a_Z must have the same length since they are corresponding points in two isometric spaces. Besides, we only consider the case that the distance of two vectors in a metric space is defined as the length of their difference.

A region in space M_L corresponds to a region of different shape in space M_Z and vise versa. But, as long as the length of a vector is considered, enumerating points in the former region and points in the latter region are equivalent. We will look at the operations of an enumeration algorithm in both metric spaces. This usually gives a clear picture of the underlying operations and make the algorithms more understandable.

3.3.2 Enumerating Points in a Parallelepiped

Dieter [15] and Knuth [42] derived algebraically an enumeration algorithm. While Knuth considered the case of Euclidean norm, Dieter considered a more general definition of norm. We present Dieter's derivation for its generality, though we are most interested in the special case of Euclidean norm.

Consider the lattice $L = L(B) = \{\eta_1 b_1 + \cdots + \eta_n b_n : \eta_i \in \mathbf{Z}\}$ with norm $\|a\| = \min\{\alpha \in \mathbf{R} : a \in \alpha\Phi\}$, where Φ is a convex, compact set which has positive measure and is symmetric about the origin. Denote $(B^{-1})^T$ by B^{-T}. The dual lattice of L is $L' = L(B^{-T}) = \{\eta_1 b'_1 + \cdots + \eta_n b'_n : \eta_i \in \mathbf{Z}\}$. The polar of Φ is $\Phi' = \{a' \in \mathbf{R}^n : |a^T a'| \leq 1, \forall a \in \Phi\}$. The norm induced by Φ' is $\|a'\|' = \min\{\alpha' \in \mathbf{R} : a' \in \alpha'\Phi'\}$.

Now, we have the following inequality [15]:

$$|a^T a'| \leq \|a\| \|a'\|'.$$

This inequality implies that for any vector $a = \eta_1 b_1 + \cdots + \eta_n b_n \in L$,

$$|\eta_i| = |(\eta_1 b_1 + \cdots + \eta_n b_n)^T b'_i| = |a^T b'_i| \leq \|a\| \|b'_i\|'.$$

If we want to enumerate all vectors in L with length smaller than or equal to r, then η_i, for all i, is bounded by

$$|\eta_i| \leq r\|b_i'\|'. \tag{3.10}$$

Using the fact that η_i must be an integer, we get

$$|\eta_i| \leq \lfloor r\|b_i'\|' \rfloor,$$

where $\lfloor \alpha \rfloor$ indicates the greatest integer smaller than α. Employing a direct search through all combinations of η_i's and picking those vectors with lengths smaller than r will not miss any desired vectors. In this way, the number of points need to be considered is

$$P_1(r) = \prod_{i=1}^{n} \left(\lfloor 2r\|b_i'\|' \rfloor + 1 \right).$$

We will look at the lattice in space M_Z. A vector $\eta_1 b_1 + \cdots + \eta_n b_n$ in M_L corresponds to a point with integer coordinates $(\eta_1, \cdots, \eta_n)$ in M_Z. Thus the region obtained by bounding the ranges of η_i's is a rectangular box in n-dimensional space. The bounding box found by inequality 3.10 is the smallest one containing the set of points needed to be enumerated. This is because the equality sign can be achieved by putting $\eta_i = r\|b_i'\|'$ and $\eta_j = 0$ for all $j \neq i$.

For ease of illustration, consider the case of Euclidean norm. Our problem is to enumerate points in L inside a sphere $\{a \in \mathbf{R}^n : a^T a \leq r^2\}$. This is equivalent to enumerating points in $\mathbf{Z}^n$ inside the ellipsoid $\{a \in \mathbf{R}^n : (B^{-1}a)^T (B^{-1}a) \leq r^2\}$. Figure 3.2 illustrates a two-dimensional example.

Finally, a rectangular box in M_Z corresponds to a parallelepiped in M_L. Therefore, Dieter's method is to enumerate points in a parallelepiped in the space associated with the given lattice.

It is well-known that the application of a unimodular transform U (an integer matrix with determinant 1 or -1) always leaves a lattice unchanged. That is

$$L(B) = B\mathbf{Z}^n = (BU)\mathbf{Z}^n = L(BU).$$

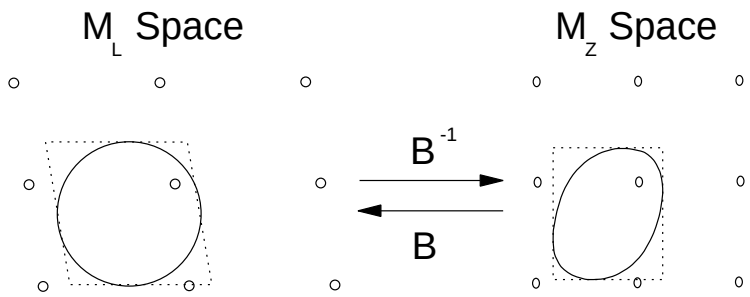

Figure 3.2: An two-dimensional example showing the correspondence between elements in M_L space and those in M_Z space.

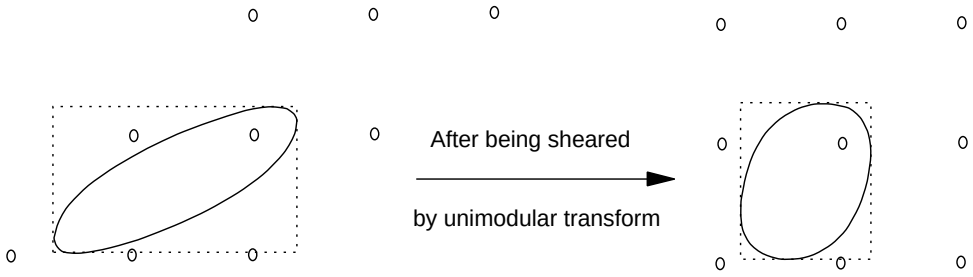

Figure 3.3: The effect of shearing an ellipsoid produced by a unimodular transform.

So we can consider any ellipsoid $\{a \in \mathbf{R}^n : ((BU)^{-1}a)^T((BU)^{-1}a) \leq r^2\}$ instead of the ellipsoid $\{a \in \mathbf{R}^n : (B^{-1}a)^T(B^{-1}a) \leq r^2\}$. Though all ellipsoids have the same content, their corresponding bounding boxes can be quite different in size. Hence it is desirable to apply unimodular transform to shape the ellipsoid such that the size of its bounding box is small enough or cannot be further reduced.

To shrink the bounding box, we "shear" the ellipsoid in a direction parallel to one face of the box (see figure 3.3). Such operation change all faces of the box except the pair of faces parallel to the shearing force. The operation is in fact achieved by the unimodular transform $U(i)$:

$$\bar{b}_j = \begin{cases} b_j + \sum_{k \neq i} \eta_k b_k & \text{for } j = i, \\ b_j & \text{for all } j \neq i, \end{cases}$$

or equivalently,

$$\bar{b}'_j = \begin{cases} b'_j & \text{for } j = i, \\ b'_j - \eta_j b'_i & \text{for all } j \neq i, \end{cases}$$

where $\bar{a}$ represents the updated content of a. By inequality 3.10, the distance between the j-th pair of parallel faces of the box is $2\|b'_j\|'$. To make all pairs of parallel faces (except the j-th pair) as close as possible, we choose

$$\eta_j = \text{round}\left(\frac{1}{2}\frac{(b'_i)^T(b'_j)}{(\|b'_i\|')^2}\right)$$

such that

$$\|b'_j - (\eta_j - 1)b'_i\|' \geq \|b'_j - \eta_j b'_i\|' \leq \|b'_j - (\eta_j + 1)b'_i\|'.$$

The transform $U(i)$ can be done repeatedly for all i until all η_j's found are zeroes for every i.

The method just described is in fact derived by Dieter (and Knuth independently) in an attempt to minimize $\|b'_i\|'$. He also suggested that when the transform U to the lattice $L(B)$ is struck, we can try similar transform U' to its dual lattice $L(B^{-T})$ and then apply U again, until both transforms are struck. We note that the application of U' is to relax the situation from the local minimum.

Dieter observed that his algorithm worked quite well in spite of occasional increases in the estimated number of points needed to be enumerated. We explain this fact by the following observation: though the content of the bounding box must decrease after each transformation, the estimate may increase slightly in some occasions since it counts the number of lattice points inside the bounding box instead of its content. We also remarked that Dieter's shrinking procedure using unimodular transforms is in fact a basis reduction algorithm.

3.3.3 Enumerating Points in a Cube

Kannan [38] derived an enumeration algorithm for the Euclidean norm based on the concept of projection. If a vector $a = \eta_1 b_1 + \cdots + \eta_n b_n \in L(B)$ is shorter than r, then

so is its projection on b_i^* for any i. Consider the case $i = n$, the projection of a on b_n^* is $\eta_n b_n^*$. Thus

$$|\eta_n| \leq \frac{r}{\|b_n^*\|}.$$

Now, consider the cases $i < n$ and assume that $\eta_{i+1}, \cdots, \eta_n$ are fixed integers. Then the projection of a on b_i^* is $\eta_i b_i^* + (\sum_{k=i+1}^{n} \eta_k \mu_{ik}) b_k^*$. Thus we have

$$|\eta_i + (\sum_{k=i+1}^{n} \eta_k \mu_{ik})| \leq \frac{r}{\|b_i^*\|}.$$

Remember that the second term on the left hand side is just a constant. In this way, the number of points needed to be enumerated is

$$P_2(r) \leq \prod_{i=1}^{n} \left(\left\lfloor \frac{2r}{\|b_i^*\|} \right\rfloor + 1 \right).$$

Kannan's method is to enumerate all vectors whose projections on $b_1^*, \cdots, b_n^*$ are all shorter than r. This is in fact enumerating points inside a n-dimensional cube with edges parallel to $b_1^*, \cdots, b_n^*$.

3.3.4 Enumerating Points in a Sphere

We observed that the length of vector a can be calculated exactly from its projections on $b_1^*, \cdots, b_n^*$. Therefore,

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

This suggests a recursive enumeration procedure according to the following relationship:

$$\left.\begin{array}{l} r_i = r \\ |\eta_i| \leq \frac{r}{\|b_i^*\|}, \end{array}\right\} \text{for } i = n,$$

and,

$$\left.\begin{array}{l} r_i = (r_{i+1}^2 - |\sum_{i+1}^{n} \eta_k \mu_{i+1,k}|^2 \|b_i^*\|^2)^{1/2} \\ |\eta_i + (\sum_{k=i+1}^{n} \eta_k \mu_{ik})| \leq \frac{r}{\|b_i^*\|}. \end{array}\right\} \text{for } i < n.$$

The procedure, in fact, recursively divides a 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.

Thus the actual enumeration process occurs in many one-dimensional lattices (in n-dimensional space).

The number of points to be enumerated is exactly those vectors shorter than r. Thus this method enumerates no undesired vectors and is optimal in this sense. Fincke and Pohst [25] derived an equivalent algorithm using Cholesky's Decomposition and the technique of quadratic completion. The equivalence between their algorithm and ours can be make obvious with the following relationship:

$$\begin{cases} q_{ii} = \|b_i^*\|^2 \\ q_{ij} = \mu_{ij} \end{cases}$$

where q_{ij}'s are defined by the equation

$$\|a\|^2 = \|\eta_1 b_1 + \cdots + \eta_n b_n\|^2 = \sum_{i=1}^{n} q_{ii}(\eta_i + \sum_{j=i+1}^{n} q_{ij}\eta_j)^2.$$

We believed that our algorithm is superior to theirs as it clearly reveals the meaning of the variables being dealt with. Fincke et. al. had also analyzed the complexity and were surprised by their result [25]:

> "But what happens, if we keep C [$= r^2$] fixed and just increase m [$= n$]? Then the enumeration method is still exponential whereas — somewhat surprisingly — (2.12) is polynomial time, if we additionally require that the lengths of the rows of R^{-1} for the matrix R of the Cholesky decomposition $A = R^T R$ [$= B^T B$] stay bounded."

According to our interpretation, this implies that the complexity increases polynomially with n, the number of dimension, provided the lengths of all edges of the bounding box associated with the ellipsoid to be enumerated are bounded. This is impossible since it is well-known that the number of lattice points inside a sphere is proportional to the volume of that sphere and thus increases with r^n. Note that the constraint on the bounding box is not significant since the radius r is fixed. This erroneous result may be due to a mistake in the derivation of equation (3.9) in reference [25].

However, the estimated number of points to be enumerated $P_3(r)$ must be upper bounded by $P_2(r)$ because Kannan's method considered a cube containing the given sphere.

Though the number of points to be enumerated is the smallest possible, learning from Dieter and Knuth, Fincke et. al. suggested two preprocessing steps to improve the algorithm.

Step 1: Use some reduction algorithm to get a quite orthogonal basis for the dual lattice.

Step 2: Reorder the indices of the basis $b_1, \cdots, b_n$ such that $\|b'_1\| \geq \cdots \geq \|b'_n\|$.

Since the algorithm recursively updates the values of η_i's, from $i = n$ down to 1 (just 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 times of updating operations.

Note that the reduction algorithm employed in step 1 has not been specified. From the simulation result [25], it seems that the LLL-reduction algorithm is more efficient than that of Dieter's.

3.3.5 Comparisons of Three Enumeration Algorithms

For large r, $P_1(r) \approx 2^n \prod_{i=1}^{n} r\|b'_i\|$; while for small r, $P_1(r) \approx 3^n \prod_{i=1}^{n} r\|b'_i\|$. So, in general, $P_1(r) \approx (\alpha_1 r)^n \prod_{i=1}^{n} \|b'_i\|$, for $2 \leq \alpha_1 \leq 3$. Similarly, $P_2(r) \approx \frac{(\alpha_2 r)^n}{\prod_{i=1}^{n} \|b^*_i\|} = (\alpha_2 r)^n \det(L(B^{-T}))$, for $2 \leq \alpha_2 \leq 3$. By Hadamard's inequality, $P_1(r) \geq P_2(r)$, for $\alpha_1 = \alpha_2$.

Therefore, it seems that Kannan's method is superior to Dieter's. In other words, enumerating lattice points in a cube is better than those in a parallelepiped. On the other hand, enumerating points in a sphere is obviously the best choice. However, it should be emphasized that Dieter's algorithm is applicable for very general definition of norms and can take full advantages of parallel computational power. The other two methods are more "sequential" in nature and developed for Euclidean norm only.

3.3.6 Improved Enumeration Algorithm for the CVP and the SVP

We observed that the last enumeration algorithm proceeds in a way similar to Kannan's method rather than Dieter's. Clearly, the range of η_i to be considered is inversely proportional to $\|b_i^*\|$. Thus a good orthogonalization of the lattice basis (instead of the dual lattice), 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. The preprocessing steps as suggested by Fincke et. al. should be replaced by:

Step 1': Use LLL reduction algorithm to get a good orthogonalization for the given lattice.

Unlike the original preprocessing steps which require to do matrix inversion, basis reduction, sorting and permutation, the above step does not need extra computation since basis reduction must be done so as to get a good value of r (as discussed later).

Another improvement may be obtained by updating the values of $r_1, \cdots, r_n$ by substituting r by r' whenever a vector of length r' smaller than r is encountered. However, to avoid unnecessarily large amount of updating operations, we can enumerate the value 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 encountered first.

By proposition 4.2 in reference [38], there must exist a lattice point b close to the query point q such that $\|b-q\| \leq \frac{1}{2}(\sum_{k=1}^{n} \|b_k^*\|^2)^{1/2}$. By putting $r = \frac{1}{2}(\sum_{k=1}^{n} \|b_k^*\|^2)^{1/2}$ after the lattice basis is reduced, we guarantee to find the shortest vector without enumerating an unnecessarily large number of points. (In practice, for SVP, a smaller value of r may usually be obtained from the length of the shortest basis vectors if the basis is properly reduced.) Similar argument suggests that during the breakdown of an i-dimensional problem into some $(i-1)$-dimensional problems as in the last

enumeration algorithm, it is sufficient to consider a sphere of radius $\frac{1}{2}(\sum_{k=1}^{i} \|b_k^*\|^2)^{1/2}$. In particular, it is required to enumerate at most two points in a one-dimensional enumeration problem. As a result, the number of points to be enumerated by the improved algorithm is bounded by

$$\min\left(2, \left\lfloor \frac{2r}{\|b_1^*\|} \right\rfloor + 1\right) \prod_{k=2}^{n} \left(\left\lfloor \frac{2r}{\|b_k^*\|} \right\rfloor + 1\right).$$

So far, the enumeration algorithm discussed is for the SVP. That is, we assumed the query point to be the origin and find a non-zero lattice point closest to it. However, it is easy to see that the suggested improvements apply equally well to enumeration algorithm for the CVP. For the CVP, we consider the inhomogeneous case that the query point can be any point in $\mathbf{R}^n$ and there is no restriction on the closest lattice points. To adapt the algorithms to this case, we use the technique of change of variables and replace η_i by $\bar{\eta}_i = \eta_i + \vartheta_i$, for all i, where $\vartheta_1 b_1 + \cdots + \vartheta_n b_n$ is the query point. After the closest lattice vector is found, we get back the integer vector by $\eta_i = \bar{\eta}_i - \vartheta_i$, for all i, and the closest vector is $B[\eta_1 \cdots \eta_n]^T$. It is easy to see that the complexity analysis for the homogeneous case still valids in this case.

The following enumeration procedure for the CVP is derived based on the above suggested improvements:

Procedure CVP$(\vartheta_1, \cdots, \vartheta_n, b_1, \cdots, b_n)$

1. Find unimodular matrix T such that BT is LLL-reduced, and get μ_{ij}'s and β_i's.

2. $RMAX_1 := \frac{1}{4}\beta_1$; for $i = 1$ to n do $RMAX_i := RMAX_{i-1} + \frac{1}{4}\beta_i$.

3. For $i = 1$ to n do $INCR_i := 1$.

4. $RMIN := \infty$; $i := n$; $R_i := RMAX_i$; $U_i := -\vartheta_i$; $Z := (\frac{R_i}{\beta_i})^{1/2}$; $UB_i := \lfloor Z - U_i \rfloor$; $UB_i := \lceil -Z - U_i \rceil$; $\eta_i := \lceil -U_i \rceil - 1$.

5. While $i \leq n$ do

6. $\quad \eta_i := \eta_i + INCR_i$.

7. If $\eta_i < LB_i$ do

8. $\quad i := i + 1$.

9. Elseif $\eta_i > UB_i$ do

10. $\quad \eta_i := \lceil -U_i \rceil$; $INCR_i := -1$.

11. Elseif $i \neq 1$ do

12. $$\begin{cases} i := i - 1;\ R_i := R_{i+1} - \beta_{i+1}(\eta_{i+1} + U_{i+1})^2; \\ U_i := -\vartheta_i + \sum_{k=i+1}^{n}(\eta_k - \vartheta_k)\mu_{ik};\ Z := (\frac{\min(R_i, RMAX_i)}{\beta_i})^{1/2}; \\ UB_i := \lfloor Z - U_i \rfloor;\ LB_i := \lceil -Z - U_i \rceil;\ \eta_i := \lceil -U_i \rceil - 1; \\ INCR_i := 1. \end{cases}$$

13. Else

14. $\quad RX := R_n - R_1 + \beta_i(\eta_1 + U_1)^2$.

15. $\quad$ If $RX < RMIN$ do

16. $\quad\quad \eta := [\eta_1, \cdots, \eta_n]^T$; $RMIN := RX$.

17. $\quad\quad$ If $RMIN < 0.9R_n$ do

18. $\quad\quad\quad$ For $k = 1$ to n do $R_k := R_k - R_n + RX$.

19. $\quad\quad$ Endif.

20. $\quad$ Endif.

21. Endif.

22. Endwhile.

23. $\eta := T\eta$; return η.

In the above procedure, β_i stores $\|b_i^*\|^2$, and R_i stores r_i^2. *RMIN* and *RX* hold the lengths of the currently shortest vector and the currently encountered vectors respectively. $INCR_i$ controls whether to increment or decrement η_i in the next pass. The input query point is $\vartheta_1 b_1 + \cdots + \vartheta_n b_n$ while the output closest lattice point is $B\eta$. Note that the factor 0.9 in line 16 is arbitrary, which ensures the R_i's is not updated too frequently.

Finally, we point out that an efficient enumeration algorithm for Euclidean norm can also lead to efficient enumeration algorithm for other norms. Since a compact, convex set Φ has positive measure, it is contained in a sphere $\bar{\Phi}$. We can find the vector a with smallest Euclidean length, and then enumerate points in a sphere $\alpha_0\bar{\Phi}$ which just contains $\alpha_0\Phi$, where $\alpha_0 = \min\{\alpha \in \mathbf{R} : a \in \alpha\Phi\}$. In this way, if the content of $\alpha_0\bar{\Phi}$ is not much larger than that of $\alpha_0\Phi$, a reasonably large number of lattice points is needed to be considered in the second invocation of the enumeration algorithm. In fact, this method uses the Euclidean norm to approximate another norm so that it can find a very short vector, and then enumerates a small number of points to solve the original problem.

3.4 CVP Algorithm Using the Reduce-and-Enumerate Approach

As mentioned in the beginning of this chapter, the CVP can be solved using a reduce-and-enumerate approach. With the LLL-reduction algorithms and enumeration algorithms discussed previously, the following CVP algorithm can be obtained.

Algorithm CVP1(q, B)

Step 1: Make the basis LLL-reduced.

Step 2: Enumerate lattice points in a proper sphere centered q for the closest lattice point.

Here, q is the query point and B is the basis matrix. The sphere to be enumerated must have a radius such that it contains at least one lattice point. A possible choice of the radius is $r = \frac{1}{2}(\sum_{k=1}^{i} \|b_k^*\|^2)^{1/2}$ (see section 3.3.6). In fact, procedure Enum_CVP discussed in the context of the enumeration algorithm is already an implementation of algorithm CVP1.

The time complexity of the LLL reduction algorithm is only polynomial. Thus the overall time complexity of algorithm CVP1 is dominated by the enumeration algorithm.

Let P be the worst-case time complexity of algorithm CVP1. In the enumeration algorithm, it takes at least kn operations per each encountered point, where k is a constant. Assuming the adverse case that all μ_{ij}'s equal to $\frac{1}{2}$, by theorem 3.1 (2),

$$P \geq kn\frac{2^n r^n}{\det(L)} = kn2^n 2^{n(n-1)/4}.$$

So $P = K^{\Omega(n^2)}$, for some constant K. The algorithm is obviously impractical for large n.

A better but more complicated CVP algorithm was developed by Kannan [38] which is $n^{O(n)} = K^{O(n\log(n))}$, for some constant K. The complexity is still very large. This is due the intrinsic property of the problem since the CVP is in fact NP-hard [20].

3.5 CVP Algorithm with Improved Average-Case Complexity

The bottleneck of algorithm CVP1 is the enumeration algorithm. The parameter determining the computational requirement in each invocation of the algorithm is r, the radius of sphere to be enumerated. Therefore, the average-case complexity can be improved by finding a value of $r < \frac{1}{2}(\sum_{k=1}^{i} \|b_k^*\|^2)^{1/2}$ before invoking the enumeration algorithm. This may be done by finding a close lattice point using some very efficient

algorithms. We will show that very efficient CVP algorithm for norms other than Euclidean does exist, and it can serve our purpose. This approach has in fact used the concept of norm approximation mentioned at the end of section 3.3.6.

3.5.1 CVP Algorithm for Norms Induced by Orthogonalization

For any point $a_0 \in \mathbf{R}^n$, define

$$\Gamma(a_0) = \{a \in \mathbf{R}^n : a_0 - a = \alpha_1 b_1^* + \cdots + \alpha_n b_n^*, \alpha_i \in (-\frac{1}{2}, \frac{1}{2})\}.$$

In words, Γ is a box centered at a_0 spanned by the orthogonalization vectors $b_1^*, \cdots, b_n^*$. Then the norm induced by Γ is

$$\|a\|^* = \sup\{\alpha \in \mathbf{R} : a \in \alpha\Gamma\}.$$

For any point a_0 of the lattice L, define

$$\Upsilon(a_0) = \{a \in \mathbf{R}^n : \|a_0 - a\|^* < \|b - a\|^*, \forall b \in L\}.$$

In other words, $\Upsilon(a_0)$ is the Voronoi region of the lattice point a_0 with respect to the norm $\|.\|^*$.

Lemma 3.3 $\forall a = \alpha_1 b_1^* + \cdots + \alpha_n b_n^* \in \mathbf{R}^n, \|a\|^* = 2\max_i\{|\alpha_i|\}$.

Proof: From definition,

$$\begin{aligned} a \in c\Gamma &\Leftrightarrow |\alpha_1|, \cdots, |\alpha_n| < \frac{c}{2} \\ &\Leftrightarrow \max_i\{|\alpha_i|\} < \frac{c}{2}. \end{aligned}$$

Hence, $\|a\|^* = \sup\{c \in \mathbf{R} : a \in c\Gamma\} = 2\max_i\{|\alpha_i|\}$.

Q. E. D.

Proposition 3.1 $\forall a \in L, \Upsilon(a) = \Gamma(a)$.

Proof: Without loss of generality, assume a is 0, the origin. For any non-zero vector $b \in L$, we can write $b = \eta_1 b_1 + \cdots + \eta_k b_k$ such that $1 \leq k \leq n$, all η_i's are integers and $\eta_k \neq 0$. Thus

$$b = \sum_{i=1}^{k} \eta_i b_i = \sum_{i=1}^{k} \eta_i \sum_{j=1}^{i} \mu_{ij} b_j^* = \sum_{j=1}^{k} \sum_{i=j}^{k} \eta_i \mu_{ij} b_i^*.$$

Let $\alpha_j = \sum_{i=j}^{k} \eta_i \mu_{ij}$. Then $b = \sum_{j=1}^{k} \alpha_j b_j^*$. We have $\alpha_k = \mu_{kk}\eta_k$, which is a non-zero integer as $\mu_{kk} = 1$.

For any vector $a' = \sum_{j=1}^{k} \zeta_j b_j^* \in \Gamma(0)$,

$$\max_i\{|\zeta_i|\} < \frac{1}{2} < |\zeta_k - \alpha_k| \leq \max_i\{|\zeta_k - \alpha_k|\}.$$

By lemma 3.3, $\|a'\|^* < \|a' - b\|^*$. Therefore, $\Gamma(0) \subset \Upsilon(0)$.

However, the fact that content of $\Gamma(0) = \det(L) =$ content of $\Upsilon(0)$, implies $\Upsilon(0) = \Gamma(0)$.

Q. E. D.

Taking into account those points on the boundary of $\Upsilon(a_0)$ and the uniqueness of the closest lattice point, we redefine the Voronoi region of a lattice point a_0 as

$$\bar{\Upsilon}(a_0) = \{a \in \mathbf{R}^n : a_0 - a = \alpha_1 b_1^* + \cdots + \alpha_n b_n^*, \alpha_i \in [-\frac{1}{2}, \frac{1}{2})\}.$$

Let $q = \sum_{i=1}^{n} \zeta_i b_i^*$ be the query point; and let a be the closest lattice point of q. Then we can write

$$a = \sum_{j=1}^{n} \eta_j b_j = \sum_{i=1}^{n} \alpha_i b_i^*,$$

where $\alpha_j = \sum_{i=j}^{n} \eta_i \mu_{ij}$. Since q must be inside the Voronoi region of a (i.e. $q \in \bar{\Upsilon}(a)$), we have, for $1 \leq j \leq n$,

$$-\frac{1}{2} \leq \alpha_j - \zeta_j = \sum_{i=j}^{n} \eta_i \mu_{ij} - \zeta_j < \frac{1}{2}.$$

Therefore,

$$\eta_j = \begin{cases} \text{round}(\zeta_j) & \text{for } j = n, \\ \text{round}(\zeta_j - \sum_{i=j+1}^{n} \eta_i \mu_{ij}) & \text{for } j < n. \end{cases} \tag{3.11}$$

From this equation, the following $O(n^2)$-time CVP algorithm for norm $\|.\|^*$ is easily deduced.

Procedure CVP_Ortho($\zeta_1, \cdots, \zeta_n$, $[\mu_{ij}]$)

1. $\eta_n := \text{round}(\zeta_n)$.
2. For $j = n - 1$ downto 1 do
3. $\quad \eta_j := \text{round}(\zeta_j - \sum_{i=j+1}^{n} \eta_i \mu_{ij})$.
4. Endfor.
5. Return η.

Note that the procedure works whether $|\mu_{ij}| \leq \frac{1}{2}$ or not. Here, it is assumed that μ_{ij}'s are given. If they are not available, calculating them from the given basis by procedure Orthogonalize takes $O(n^3)$.

Note that we have find a polynomial algorithm to solve the CVP with respect to the maximum norm if an cubic orthogonalization cell of the given lattice exists and can be found in polynomial time, although the problem for arbitrary lattice is NP-hard [20].

3.5.2 Improved CVP Algorithm using Norm Approximation

In addition to reducing the enumeration time by using procedure CVP_Ortho to find a close lattice point, the enumeration process may usually be eliminated by introducing a simple test.

Proposition 3.2 *Let $\Phi_1(a)$ be a convex, compact set which has positive measure and is symmetric about point a; and let its induced norm $\|a\|_1 = \min\{\alpha \in \mathbf{R} : a \in \alpha\Phi_1\}$. If $a_0 \in L$, $b \in \zeta\Phi_1(a_0) \subset \bar{\Upsilon}(a_0)$, for some constant ζ, then a_0 is the closest lattice point of b with respect to norm $\|.\|_1$.*

Proof: For all $a' \neq a_0$ in L,

$$\begin{aligned} & \zeta\Phi_1(a_0) \subset \bar{\Upsilon}(a_0) \text{ and } \zeta\Phi_1(a') \subset \bar{\Upsilon}(a') \text{ and } \bar{\Upsilon}(a') \cap \bar{\Upsilon}(a_0) = \emptyset \\ \Rightarrow \quad & \zeta\Phi_1(a_0) \cap \zeta\Phi_1(a') = \emptyset. \end{aligned}$$

Hence,

$$\begin{aligned} b \in \zeta\Phi_1(a_0) \quad & \Rightarrow \quad b \notin \zeta\Phi_1(a') \\ & \Rightarrow \quad \|b - a'\|_1 > \zeta \geq \|b\|_1. \end{aligned}$$

Q. E. D.

Let $\zeta_m = \sup\{\zeta \in \mathbf{R} : \zeta\Phi_1(a_0) \subset \bar{\Upsilon}(a_0)$. By proposition 3.2, if b passes the test

$$\|a_0 - b\|_1 < \zeta_m, \tag{3.12}$$

then a_0 must be the closest lattice point of b. Thus, only when the above test is failed, the enumeration process is done with $r = \|a_0 - b\|_1$. (Note that $b \in \bar{\Upsilon}(a_0) \Rightarrow \|a_0 - b\|_1 \leq \frac{1}{2}(\sum_{k=1}^{i} \|b_k^*\|^2)^{1/2})$. In this way, the expected time of finding the closest lattice point for any given point b is

$$T = P\{\|a_0 - b\|_1 < \zeta_m\}T_t + P\{\|a_0 - b\|_1 \geq \zeta_m\}T_e, \tag{3.13}$$

where T_t is the expected time to find a short vector a_0 and do the test 3.12, and T_e is the expected time to do the enumeration. If the probability distribution of the query point b is such that the first term in equation 3.13 dominate, then $T \approx T_t$. Consequently, if the the average-time behavior is a main concern, a very efficient CVP algorithm for norm $\|.\|_1$ can be obtained.

Denote $\Upsilon_1(a_0)$ as the Voronoi region of the lattice point a_0 with respect to the norm $\|.\|_1$. The actual performance of this method depends on how "close" to $\Upsilon_1(a_0)$ the orthogonalization cell $\Gamma(a_0)$ is. The definition of closeness in turn depends on the probability distribution of the query points. Intuitively, we want $\Phi_1(a_0)$ and $\Gamma(a_0)$ to intersect as much as possible. Return to our interested case that the norm

$\|.\|_1$ is the Euclidean norm $\|.\|$, the desired orthogonalization cell should be close to a cube. In other words, we want the lengths of the orthogonalization vectors show small variations. Such orthogonalization can be obtained by the LLL reduction algorithm.

With the query point $q = \sum_{i=1}^{n} \vartheta_i b_i$, the improved CVP algorithm for Euclidean norm is as follows:

Procedure CVP_Euclid$(\vartheta_1, \cdots, \vartheta_n, b_1, \cdots, b_n)$

1. Make $b_1, \cdots, b_n$ LLL-reduced and get μ_{ij}'s and β_i's.
2. $\zeta_m := \frac{1}{2}(\min_i\{\beta_i\})^{1/2}$.
3. Find $\theta = [\theta_1, \cdots, \theta_n]$ such that $q = \sum_{i=1}^{n} \theta_i b_i^*$.
4. Call procedure CVP_Ortho to get η, where $B\eta$ is a lattice point.
5. If $\|B(\theta - \eta)\| \geq \zeta_m$ do call an enumeration procedure similar to Enum_CVP to update η with $r = \|B(\theta - \eta)^T\|$.
6. Return η.

As shown in line 2, $\zeta_m = \min_i\{\|b_i^*\|\}$ for Euclidean norm. In line 3, θ can be found by the formula $\theta_j = \sum_{i=j}^{n} \vartheta_i \mu_{ij}$, for $1 \leq j \leq n$. We remark that, in procedure CVP_Euclid, T_t is $O(n^3)$ dominated by the LLL reduction algorithm in line 1 and the test in line 5.

In some situations, we are given a fixed lattice and want to solve the CVP for a large number of query points. Then basis reduction of the lattice needs to be done once (as in line 1), and so is the calculation of ζ_m (as in line 2). In case, the number of query points is really enormous, it is worthwhile to find a better value of ζ_m with more computational effort. If line 2 is replaced by

$$\zeta_m = \frac{1}{2} \min_{a \in L} \|a\|,$$

the above algorithm is still correct since spheres with the same radius ζ_m centered at different lattice points are all disjoint (refer to the proof of proposition 3.2). To calculate $\min_{a \in L} \|a\|$, a SVP algorithm similar to Enum_CVP can be employed.

Chapter 4

MLSE Algorithm

According to our lattice interpretation of the MLSE problem for the PAM system (see section 1.1), the problem can be formulated as a CVP with respect to Euclidean norm. Thus the CVP algorithms discussed previously are applicable for this problem. In the following discussion, we adopt the notations in section 2.4 and 3.1.

Then the correspondence becomes:

$$\begin{aligned} n &= \delta \\ B &= H \\ q &= z + G\tilde{x}. \end{aligned}$$

4.1 MLSE Algorithm for PAM Systems

Without loss of generality, assume the first term of the channel impulse response $h_0 = 1$. (If not, we can always normalize h in this way in $O(v)$ operations.) Since the basis matrix B, associated with the channel impulse response h, and the query point q, associated with the received sequence z and the detected sequence $\tilde{x}$, have certain nice properties, many simplifications of the CVP algorithm can be made. The following modifications refer to lines of procedure CVP_Euclid.

Since the basis matrix B is upper-triangular Toeplitz with all diagonal elements

equal to one, the orthogonalization matrix $[b_i^*] = I$, the identity matrix. Consequently, $\|b_1^*\| = \cdots = \|b_n^*\| = 1$ and the LLL-reduction algorithm invoked in line 1 simply weakly reduce the basis. In this case, it is easy to see that procedure Weakly_Reduce always returns an upper-triangular Toeplitz matrix $[\mu_{ij}]^T$ such that $UB = [\mu_{ij}]^T$, where U is an upper-triangular Toeplitz, unimodular matrix. For any upper-triangular Toeplitz matrix, it suffices to know its first row vector. Let $\breve{B}$, $\breve{U}$, $\breve{\mu}$ be the first row vectors of matrices B, U and $[\mu_{ij}]^T$ respectively. (Note that $\breve{U}_1 = \breve{\mu}_1 = 1$.) Then line 1 can be replaced by the following procedure:

Procedure Toeplitz_Reduce$(n, \breve{B})$

1. For $i = 2$ to n do
2. $\quad Z := \breve{U}_1 + \sum_{j=2}^{i-1} \breve{U}_j \breve{B}_{i+1-j}$; $\breve{U}_i := -\text{round}(Z)$; $\breve{\mu}_i := Z + \breve{U}_i$.
3. Endfor.
4. Return $\breve{\mu}$, $\breve{U}$.

Also, line 2 should be substituted by $\zeta_m := \frac{1}{2}$ as $\|b_1^*\| = \cdots = \|b_n^*\| = 1$. In the calculation of $q = z + G\hat{x}$, the query point is already expressed as $q = \sum_{i=1}^{n} \theta_i b_i^*$. Thus the computation of $\theta_1, \cdots, \theta_n$ in line 3 is unnecessary. In addition, procedure CVP_Ortho called in line 4 should be replaced by:

Procedure Toeplitz_Ortho$(n, q, \breve{\mu})$

1. $\eta_n := \text{round}(q_n)$; $d := (q_n - \eta_n)^2$.
2. For $j = n - 1$ downto 1 do
3. $\quad k := q_j - \sum_{i=j+1}^{n} \eta_i \breve{\mu}_{i-j+1}$; $\eta_j := \text{round}(k)$; $d := d + (k - \eta_j)^2$.
4. Endfor.
5. Return η, d.

Remark that procedure Toeplitz_Ortho corresponds to a kind of deconvolution operations and it works whether $|\breve{\mu}_i| \leq \frac{1}{2}$ or not. The second variable d returned is the distance between the query point and the returned lattice point, i.e. $d = \|q - H'\eta\|$. Summarizing these modifications and adapting the conventions of section 2.4, we have the procedure below:

Procedure MLSE_PAM$(v, \delta, h, \hat{x}, z)$

1. (Calculate $q = z + G\hat{x}$)$k := \delta - v$; For $i = 1$ to k do $q_i := z_i$; for $i = 1$ to v do $q_{k+i} := z_{k+i} + \sum_{j=0}^{i} h_{v-i+j}\hat{x}_{k+j}$.
2. (Find a close vector) Call procedure Toeplitz_Ortho(δ, q, h') to get η, d.
3. (If not closest, enumerate) If $d \geq \frac{1}{2}$ do
4. Call procedure Toeplitz_Reduce(δ, h') to get $\breve{U}$, $\breve{\mu}$ s.t. $H'U = [\mu_{ij}]^T$.
5. Enumerate lattice $[\mu_{ij}]^T$ with $r = d$ to get η; $\eta := U\eta$.
6. Endif.
7. Return η.

Here, H' is an upper-triangular Toeplitz matrix with the first row h'. $\hat{x}_i$ in line 1 refers to the i-th element of vector $\hat{x}$ and has nothing to do with the time order (in contrast to the usage in section 2.4). Note that vector h is not simply the first row of matrix H since its first element is h_0 instead of h_1 as defined in section 2.4. The reduction process in line 4 needs to be done only once for a given channel.

According to the discussion in section 2.2, δ is of the same order as v. If enumeration is not done, the complexity is dominated by lines 1 and 2, all require $O(\delta^2)$ time. Note that lines 1 and 5 involves multiplication of a Toeplitz matrix and a column vector which is $O(\delta^2)$ using direct method. If fast convolution algorithm is used, the operation can be done in $O(\delta \log(\delta))$ time [43]. Following the analysis in section 3.5.2, for sufficiently large SNR, the expected-time complexity of procedure MLSE_PAM is

$O(\delta^2)$. Because all matrices encountered in the procedure are Toeplitz, the space complexity is $O(\delta)$.

Note that procedure MLSE_PAM is suboptimal because of the boundary effect. Namely a detected sequence may not be a permissible sequence. But for sufficiently large number of transmit levels, its performance tends to be optimal.

4.2 MLSE Algorithm for Unimodular Channel

Definition 4.1 *We define a unimodular channel as one whose associated H, which is an upper-triangular and Toeplitz matrix with the channel impulse response h as its first row, is unimodular.*

Assuming $h_1 = 1$, this is equivalent to saying that all h_i's are integers. An important example is the partial-response system whose $h = (1, 0, \cdots, 0, -1)$. Since H is unimodular, so is H^{-1}. The lattice basis H can be reduced to I by unimodular transform H^{-1}. So the MLSE problem becomes a CVP for lattice $\mathbf{Z}^\delta$, this can easily be solved by the rounding operations. Our MLSE algorithm is simplified to:

Procedure MLSE_UNI$(v, \delta, h, \hat{x}, z)$

1. (Calculate $q = z + G\hat{x}$)$k := \delta - v$; For $i = 1$ to k do $q_i := z_i$; for $i = 1$ to v do $q_{k+i} := z_{k+i} + \sum_{j=0}^{i} h_{v-i+j}\hat{x}_{k+j}$.

2. (Find closest vector) For $i = 1$ to δ do $\eta_i := \text{round}(q_i)$; $\eta := H^{-1}\eta$.

3. Return η.

The space complexity of this procedure is $O(\delta)$. Using fast convolution algorithm for lines 1 and 2 (H^{-1} can be found by deconvolution), the time complexity of this procedure is exactly $O(\delta \log(\delta))$. It is easy to see that its space-time complexity is also $O(\delta \log(\delta))$.

4.3 Reducing the Boundary Effect for PAM Systems

The key to our efficient MLSE algorithms is the use of regular structure of a lattice. However, as mentioned in section 2.4, near the boundary of the finite lattice, this regularity breaks down. In a query, the CVP algorithm may return a point outside the given finite lattice as the nearest lattice point. In general, the boundary effect due to the finiteness of lattice is difficult to deal with.

Observation 4.1 *For a PAM system, the boundary of the associated lattice always forms a parallelepiped. There is a high probability for a query point outside the given finite lattice to have its nearest lattice point on the lattice boundary.*

Based on this observation, a heuristic method is derived to reduce the boundary effect. Assume the nearest lattice point falls on a face of the n-parallelepiped, which is itself a $(n-1)$-parallelepiped. Let q' be the vector obtained by projecting the query point q onto the hyperplane containing the face. Namely, $q' = q - (q^T a)a$, where a is the unit normal vector of the hyperplane. Obviously, the nearest lattice point of q' is exactly the same as that of q. If we replace the query point q by q', the CVP algorithm has a much smaller probability to return a point outside the finite lattice.

However, we do not know which face of the parallelepiped (if it exists) contains the nearest lattice point. The answer is simple. We can check for all n pairs of parallel faces and consider every hyperplane for which the query point q does not fall in between its corresponding pair of parallel faces. For every such hyperplane, a projected vector of q is found. Among these projected vectors, one which falls on the lattice boundary is picked to substitute the original query point.

Procedure Project(q,N,m)

1. $\bar{q} := H^{-1}q$.

2. For $i = 1$ to n do

3. $\bar{d}_i := \bar{q}_i - \frac{m-1}{2}$.

4. If $|\bar{d}_i| > \frac{m-1}{2}$ do $a_i := 1$; else $a_i := 0$.

5. Endfor.

6. $d := q - \frac{m-1}{2} H(\text{sign}(\bar{d}) + 1)$.

7. $r := \infty$; $q' = q$.

8. For $i := 1$ to n do

9. If $a_i = 1$ do

10. $\bar{q} := H^{-1}(q - (d^T N_i)N_i)$; $\bar{q} := \min(\max(\bar{q}, 0), m-1)$; $p := H\bar{q}$.

11. If $r > \|p - q\|$ do

12. $r := \|p - q\|$; $q' := p$.

13. Endif.

14. Endif.

15. Endfor.

16. Return q'.

In the above procedure, vector d is a difference vector pointing from a point on lattice boundary to q such that $\bar{q}$ in line 10 is a projected vector on the i-th boundary face. Lines 11 to 13 pick the projected vector with minimum distortion as the new query point q'. The matrix N has its i-th column vectors N_i as the unit normal vector to the i-th face of the parallelepiped. It can be seen that N_i is parallel to the i-th basis vector of the dual lattice. Consequently, N can be obtained by normalizing each column of $(H^{-1})^T$ and it is only necessary to allocate extra storage for the lengths of n dual basis vectors. Consequently the space complexity is $O(\delta)$.

But the time complexity, dominated by line 10, is $O(\delta^2 \log(\delta))$ assuming the use of fast convolutional algorithm. Note that for large SNR, the expected-time complexity of the MLSE algorithms discussed previously may be increased to $O(\delta^2 \log(\delta))$ if procedure Project is invoked.

4.4 Simulation Results and Performance Investigation for Example Channels

The error performance of the proposed MLSE algorithms is suboptimal mainly due to the boundary effect. However, as the SNR and the size of signal set m increase, the boundary effect diminishes and the error probability approaches optimal value. We investigate the performance degradation for various values of SNR and m by considering three examples. The symbol error probability for given SNR and m is obtained by simulation using a sequence of 100,000 symbols. Though the system designers may usually be interested in error performance for high SNRs, limited by computational power, only the performance for low SNRs is simulated. Nonetheless, we remark that the purpose of the simulations is to verify our theory and hence the reader should have confidence in our predictions on the performance for high SNRs.

Channel 1: $h = (1, 0.5)$. For this channel, there is a single error event $(1, 0, \cdots)$ with $d^2_{min} = 1.25$. The truncation depth δ is 4. Figure 4.1 show the simulated performance of procedure MLSE_PAM for $m = 2, 4, 8, 16$ respectively. The performance is optimal even for very low SNRs and the binary case.

Channel 2: $h = (1, -1)$. This is a partial response channel which is well-known for its catastrophic behavior. The practical precoding technique is employed to prevent infinite error propagation (see [27]). The truncation depth is 7. As shown in figure 4.2, procedure MLSE_UNI gives a loss of 1dB in SNR for $m = 4$ and even larger loss for the binary case. However, for $m = 8, 16$, it approaches

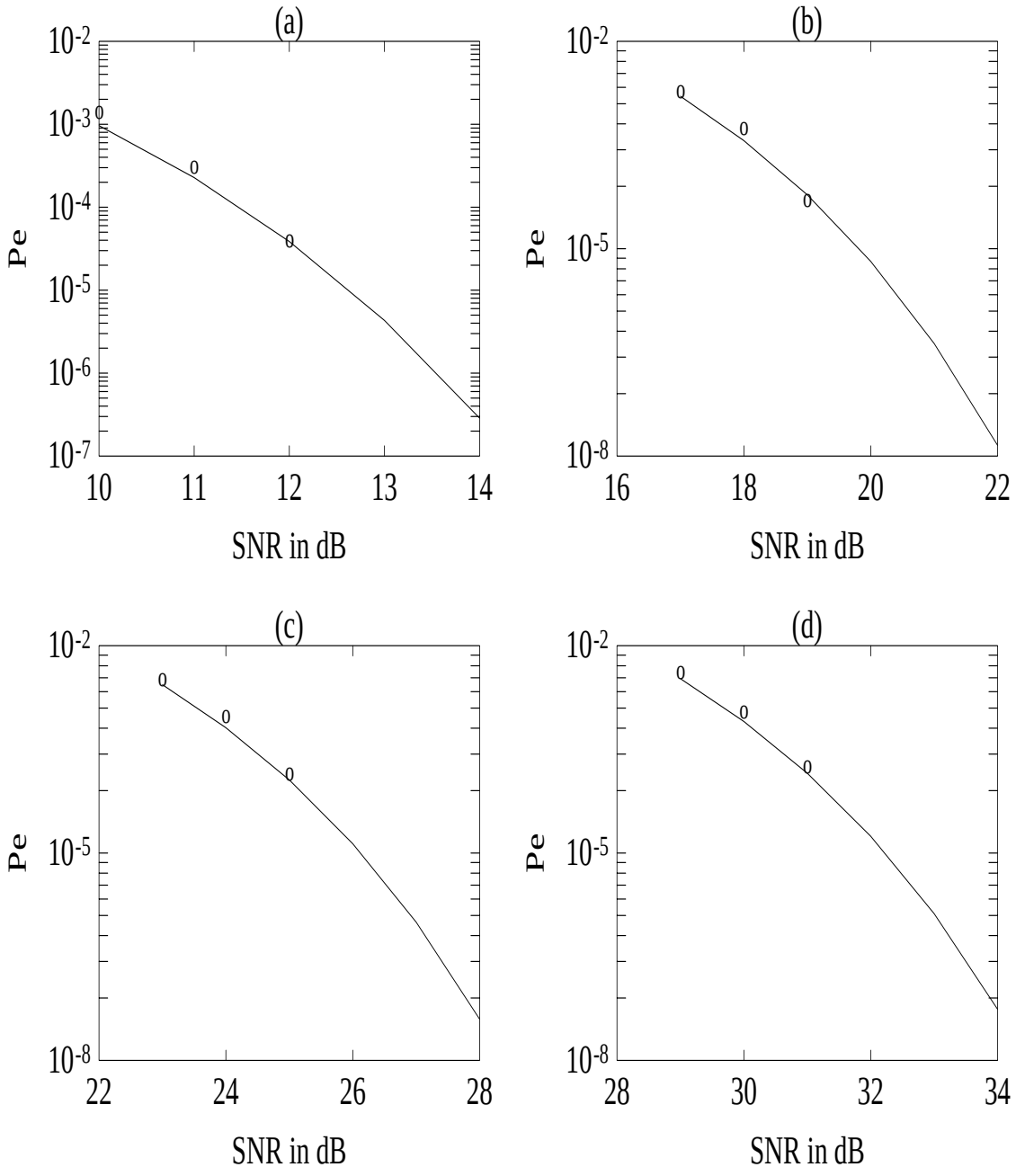

Figure 4.1: Simulated performance for channel 1: $h = (1, 0.5)$ and $\delta = 4$. (a) $m = 2$, (b) $m = 4$, (c) $m = 8$, (d) $m = 16$; where o: MLSE_PAM, —: union bound.

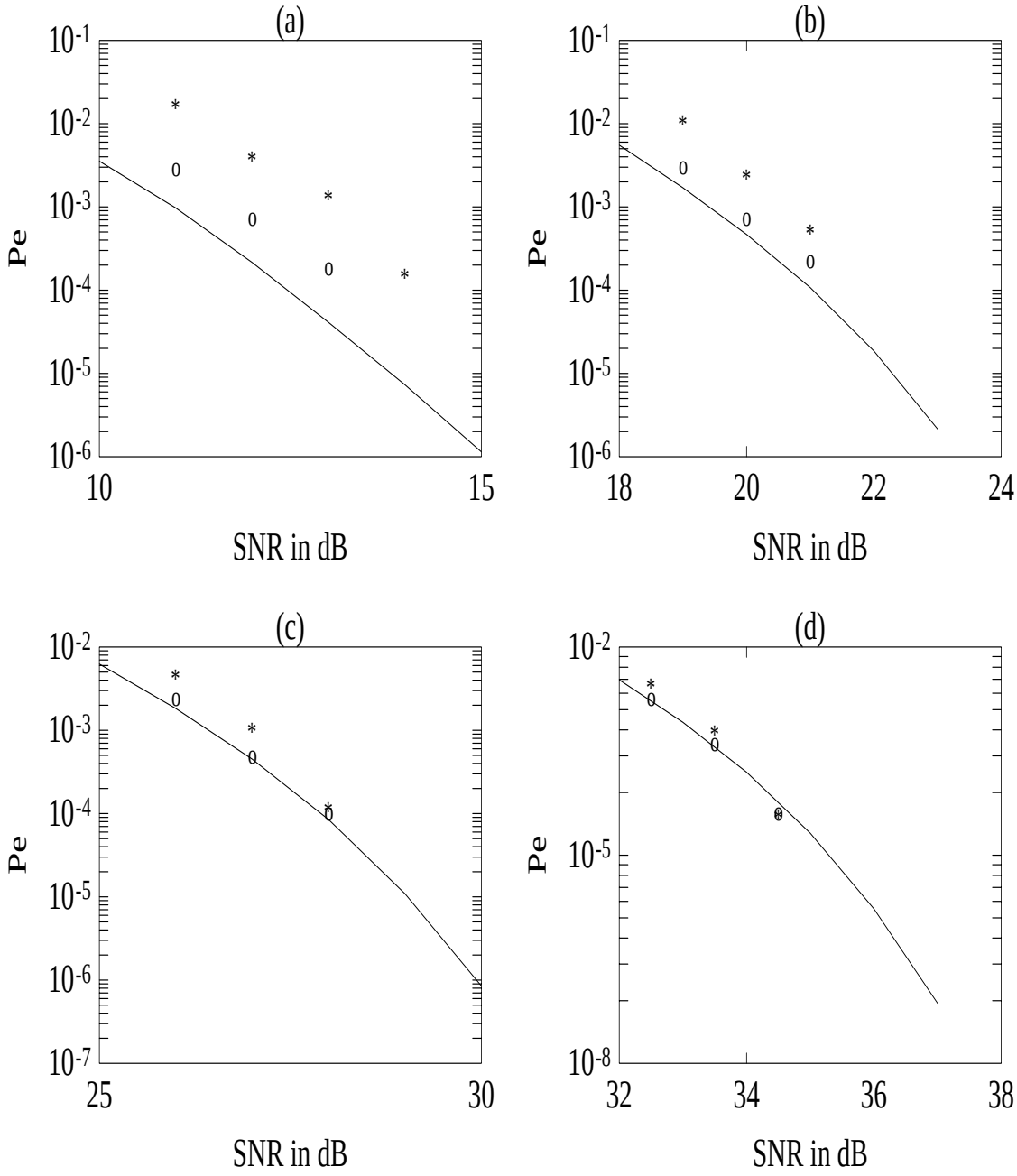

Figure 4.2: Simulated performance for channel 2: $h = (1, -1)$ and $\delta = 7$. (a) $m = 2$, (b) $m = 4$, (c) $m = 8$, (d) $m = 16$; where o: MLSE_UNI, *: MLSE_UNI with projection, —: union bound.

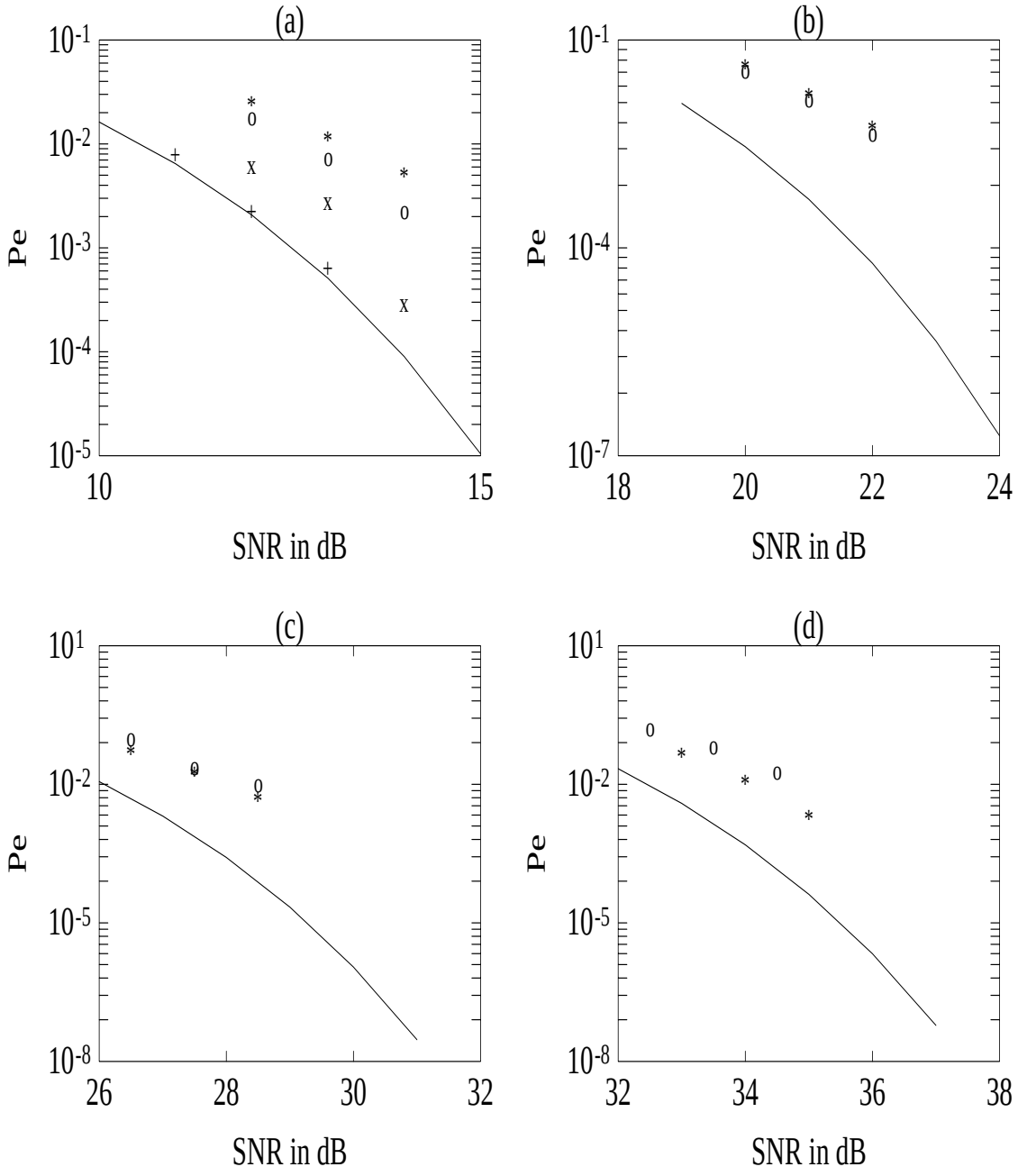

Figure 4.3: Simulated performance for channel 3: $h = (1, -1.5, 0.8)$. (a) $m = 2$, (b) $m = 4$, (c) $m = 8$, (d) $m = 16$; where x: VA with $\delta = 5$, +: VA with $\delta = 15$, o: MLSE_PAM with projection and $\delta = 5$, *: MLSE_PAM with projection and $\delta = 10$, —: union bound.

quickly the optimal values at medium SNR. If procedure Project is employed, for low SNRs, the loss is about 1dB in the binary case and only 0.5 dB in the case $m = 4$. It is expected that the degradation further diminishes as SNR increases.

Channel 3: $h = (1, 1.5, -0.8)$. The channel is chosen such that there are many error events with small weights. According to proposition 2.1, the truncation depth δ should be chosen as 15 for the binary case. Simulation results can be found in figure 4.3. For the binary case, the VA gives upto 1 dB loss in SNR for $\delta = 5$; but becomes optimal for $\delta = 16$. This verifies our proposition on the choice of the truncation depth. Then the performance of procedure MLSE_PAM with projected query points is considered. For all SNRs considered, $m = 2, 4, 8, 16$ and $\delta = 5, 10$, there is roughly 2 dB loss in SNR. The result is unsatisfactory. To account for this, let us consider two cases:

Case 1: When δ is small, the error performance is dominated by many truncated error events instead of the error event with minimum weight.

Case 2: When δ is large, i.e. the dimension of the associated lattice is large, the boundary effect is more remarkable since it is much likely for the CVP algorithm to return an unallowable point.

It should be noticed that for $m < 8$ the smaller δ is preferred due to boundary effect; but as m increases the larger δ becomes more favorable. This is consistent with our prediction since the boundary effect is less remarkable as the size of the lattice increases for a fixed dimension.

From the above simulations, it can be concluded that if the truncation depth as chosen according to proposition 2.1 is not large, the proposed MLSE algorithms gives nearly optimal performance for the multilevel PAM systems. For the binary case, the algorithm with projected query points is also an efficient suboptimal algorithm with instrumentable complexity.

Although this new algorithm has attractive space and time complexities especially for software implementation, its main limitations are:

Limitation 4.1 *It is unable to handle some channels which require large truncation depth in order to have optimal performance.*

Limitation 4.2 *For some applications, it is desirable to have really short detection delay and it affords to implement the sequence estimator on hardware. Unlike the VA, the general MLSE procedure involving enumeration cannot be readily implemented on VLSI circuit and the detection delay is erratic.*

4.5 MLSE Algorithm for Other Lattice-Type Modulation Systems

So far, the discussion has been limited to the PAM system. It is easy to see that our lattice interpretation of the MLSE can be applied equally well to all lattice-type modulation systems. Practically, many modulation systems, like the QAM system, use both in-phase and quadrature carriers. In such situations, the channel impulse response h and the source sequence x are complex. Under assumptions 1.1 and 1.2, our formulation in section 2.4 is still valid but the associated lattice basis matrix H and the query point q are both complex. Denote $H = H_R + iH_I$; $q = q_R + iq_I$; $\tilde{x} = \tilde{x}_R + i\tilde{x}_I$, where i denotes $\sqrt{-1}$ and $\tilde{x}_R, \tilde{x}_I \in \mathbf{Z}^\delta$. Then the MLSE estimates $\tilde{x}$ so as to minimize the Euclidean weight

$$\|H\tilde{x} - q\| = \left\| \begin{pmatrix} H_R & -H_I \\ H_I & H_R \end{pmatrix} \begin{pmatrix} \tilde{x}_R \\ \tilde{x}_I \end{pmatrix} - \begin{pmatrix} q_R \\ q_I \end{pmatrix} \right\|.$$

Thus the MLSE problem corresponds to the CVP with a 2δ-dimensional lattice. Note that the basis matrix is block-Toeplitz instead of Toeplitz and the CVP algorithms cannot be simplified as in section 4.1. Using procedure CVP_Euclid as a MLSE

algorithm, the space complexity is $O(\delta^2)$ and, for sufficiently large SNR, the expected-time complexity is $O(\delta^3)$.

In case of a QAM system, the boundary effect may be reduced by projection on the parallelepiped defined by the lattice boundary. A procedure similar to procedure Project can be derived in a straightforward way, but due to the lack of the Toeplitz structure, the space and time complexities are $O(\delta^2)$ and $O(\delta^3)$ respectively.

4.6 Some Potential Applications

For most practical data transmission systems, the channel is time-variant which requires an adaptive receiver. Because the proposed detector does not involve very complicated preprocessing steps, it is easily made adaptive following the conventional scheme developed for the Viterbi detector [22]. The basic idea is to include a channel estimator which identifies the channel by adjusting the tags of a transversal filter using the steepest descent algorithm so that the mean-square error between the actual received sequence and the received sequence estimate is minimized. With adaptive power, a very efficient receiver, for low-speed serial modem over telephone network [9] or HF radio link [10], can be built with inexpensive microprocessor and moderate storage.

Another important application is for partial response systems, though the new detector is suboptimal for $m \leq 4$. In addition to data transmission systems employing the signaling scheme for bandwidth compaction, digital magnetic recording systems are shown to be members of this class [44]. For the tape recorders, a channel rate as high as 120 Mbps is possible [64]. In this situation, procedure MLSE_UNI may be implemented directly on VLSI circuit since, unlike procedure MLSE_PAM, no enumeration process is involved.

In addition, the new MLSE algorithm like the VA can be applied to predictive waveform coders with delayed decision [45], under the mean squared error criterion

and uniform quantizer. This multipath search coding scheme is proved to outperform the conventional single path coder in applications like speech coding [24]. In this case, our algorithm is optimal since the number of quantization levels is usually much greater than 4 and the boundary effect should be negligible.

4.7 Further Research Directions

With the advent of our efficient MLSE algorithm, limitations 1.1 and 1.2 are much relaxed. Especially, in contrast to the VA, the new algorithms favor large number of transmit levels m. The feasibility of implementing practical multilevel systems with large m should be re-considered.

The proposed algorithm is imperfect for certain channels as described in limitation 4.1. Thus it is desired to develop better methods to combat the boundary effect for various lattice-type modulation schemes such that the lattice algorithm can be applied to a more general class of channels.

Another natural direction is to attack limitation 4.2. It is a common fact that algorithms unsuitable for VLSI implementation can find no place in real-time applications. How to modify the algorithm in a way to map directly into hardware circuits should deserve further investigations.

It should be emphasized that the importance of our lattice interpretation does not simply end with an efficient MLSE algorithm. Experience shows that new viewpoints can usually influence the existing scenery. The lattice viewpoint sheds light on the macro operations of a bandlimited channel. As observed by Burr [5], codes designed for ideal channel may be totally inefficient for bandlimited channels. The new viewpoint will undoubtably give hints on the design of such channel coding schemes.

Besides, the proposed lattice algorithms can be used in a seemingly less related area. It is widely known that the design of multi-dimensional signal constellation and the lattice vector quantizer are similar problems. It seems that most efficient signal

constellation schemes are of lattice-type [30], [29]. For both applications, efficient encoding and decoding algorithms are crucial factors detemining the applicability of these schemes. As a result, the choice of lattices is limited to those with known fast CVP algorithm [12]. Our CVP algorithms thus allow better choice of lattices and hence superior performance.

Chapter 5

Conclusion

We have developed a new interpretation of the MLSE for the lattice-type modulation systems. In this formulation, the MLSE problem is identified as the nearest lattice point problem. On unconstraining the lattice and utilizing its regular structure, very efficient sequence estimation algorithms are derived from the CVP algorithms. Comparing with the conventional Viterbi detector for the PAM systems, the space complexity is reduced from $O(\delta m^v \log(m))$ to $O(\delta)$ while, for sufficiently large SNR, the expected-time complexity is reduced from $O(m^{v+1})$ to $O(\delta^2 \log(\delta))$ operations per symbol, where v is the channel memory length, m is the number of transmit levels and δ is the truncation depth of the estimator. Remark that we have significantly simplified the sequence estimator. In particular, the dependence of the receiver complexity on m is removed. Considering the error performance, we favor m to be as large as possible. This is in consistence with the trend of multilevel transmission systems. Extension to other lattice-type modulation schemes are straightforward.

Unfortunately, the error performance is quite sensitive to the distance spectrum of the given channel though it approaches optimal for sufficiently large SNR and number of transmit levels. Another drawback is that for general channels the algorithm does not map easily into hardware circuits. This constrains it from many real-time applications.

Nonetheless, as demonstrated in our simulation results, the proposed algoirthm should find immediate applications in partial-response systems like some silent bandlimited channels and magnetic recording systems.

In addition to a new MLSE algorithm, our lattice interpretation sheds light on the macro operations of a bandlimited channel. So it gives hints on the answers to many related problems such as channel coding and signal constellation.

We have also contributed to the design of some lattice algorithms. First, the spirit of the famous LLL-reduction algorithm is illuminated. Some practical variations are suggested which reduce the complexity from $O(n^4)$ to $O(n^3)$. As the LLL-reduction algorithm has been part of many important lattice algorithms, our improved versions in fact provide improvements on various applications (refer to page 37). Second, a unified treatment to various enumeration algorithms is introduced using the concept of isometric mapping. Based on this geometric interpretation, previously known enumeration algorithms can be derived and classified in a natural way. Improved enumeration algorithms for the CVP and the SVP are then suggested. Finally, a polynomial CVP algorithm for the norm induced by any orthogonalization is derived. Using the concept of norm approximation, efficient CVP algorithms for a general class of norms are proposed. Similar improvements can be obatined for the SVP algorithms in a straightforward manner.

Bibliography

[1] L. C. Barbosa. Maximum Likelkhood Sequence Estimators: A Geometric View. *IEEE Trans. Inform. Theory*, 35:419–427, 1989.

[2] C. Belfiore and J. Park. Decision Feedback Equalization. *Proc. IEEE*, 67:1143–1156, 1979.

[3] J. L. Bentle, B. Weide, and A. C. Yao. Optimal Expected-time Algorithms for Closest-point Problems. *ACM Trans. Math. Software*, 6:563–579, 1980.

[4] W. Bliss and L. L. Scharf. Algorithms and Architectures for Dynamic Programming on Markov Chains. *IEEE Trans. on A.S.S.P.*, 37:900–912, 1989.

[5] A. G. Burr. Block Codes on a Dispersive Channel. *IEE Proceedings, Part I*, 2:95–104, 1991.

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

[7] CCITT Red book, Recommendation V.32. Geneva, 1984.

[8] B. Chazelle. How to Search in History. *Inform. and Control*, 64:77–99, 1985.

[9] A. P. Clark and M. Clayden. Pseudobinary Viterbi Detector. *IEE Proceedings, Part F*, 131:208–218, 1984.

[10] A. P. Clark and H. Y. Najdi. Detection Process of a 9600 bit/s Serial Modem for HF Radio Links. *IEE Proceedings, Part F*, 130:368–376, 1983.

[11] K. L. Clarkson. A Randomized Algorithm for Closest-point Queries. *SIAM J. Comput.*, 17:830–848, 1988.

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

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

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

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

[16] D. Dobkin and R. J. Lipton. Multidimensional Searching Problems. *SIAM J. Comput.*, 5:181–186, 1976.

[17] A. Duel-Hallen and C. Heegard. Delayed Decision-feedback Sequence Estimation. *IEEE Trans. Commun.*, 37:428–436, 1989.

[18] H. Edelsbrunner, L. Guibas, and J. Stolfi. Optimal Point Location in a Planar Subdivision. *SIAM J. Comput.*, 15:317–340, 1986.

[19] H. Edelsbrunner and H. A. Maurer. Finding Extreme Points in Three Dimensions and Solving the Post-office Problem in the Plane. *Information Processing Letters*, 21:39–47, 1985.

[20] 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.

[21] M. V. Eyuboglu and S. U. H. Qureshi. Reduced-state Sequence Estimation with Set Partitioning and Decision Feedback. *IEEE Trans. Commun.*, 36:13–20, 1988.

[22] F. R. Magee, Jr. and John G. Proakis. Adaptive Maximum-Likelihood Sequence Estimation for Digital Signaling in the Presence of Intersymbol Interference. *IEEE Trans. Inform. Theory*, 19:120–124, 1973.

[23] D. D. Falconer and F. R. Magee. Adaptive Channel Memory Truncation for Maximum Likelihood Sequence Estimation. *Bell Syst. Tech. J.*, 52:1541–1562, 1973.

[24] H. G. Fehn and P. Noll. Multipath Search Coding of Stationary Signals with Applications to Speech. *IEEE Trans. Commun.*, 30:687–701, 1982.

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

[26] S. J. Fortune. A Sweepline algorithm for Voronoi Diagrams. *Proc. Second Symposium on Computational Geometry*, pages 313–322, 1986.

[27] G. D. Forney, Jr. Maximum-Likelihood Sequence Estimation of Digital Sequences in the Presence of Intersymbol Interference. *IEEE Trans. Inform. Theory*, 18:363–378, 1972.

[28] G. D. Forney, Jr. The Viterbi Algorithm. *IEEE Proc.*, 61:268–278, 1973.

[29] G. D. Forney, Jr. Multidimensional Constellations — Part II: Voronoi Constellations. *IEEE Journal Sel. Areas Commun.*, 7:941–958, 1989.

[30] G. D. Forney, Jr. et al. Efficient Modulation for Bandlimited Channels. *IEEE Journal Sel. Areas Commun.*, 2:632–647, 1984.

[31] E. N. Gilbert. Random Subdivisions of Space into Crystals. *Ann. Math. Statist.*, 33:958–972, 1962.

[32] M. Grotschel, L. Lovasz, and A. Schrijver. Geometric Methods in Combinatorial Optimization. *Proc. Silver Jubilee Conf. on Comb. Univ. Waterloo*, 1:167–183, 1982.

[33] J. F. Hayes. The Viterbi Algorithm Applied to Digital Data Transmission. *IEEE Commun. Soc. Magazine*, 13(2):15–20, 1975.

[34] J. A. Heller and I. M. Jacobs. Viterbi Decoding for Satellite and Space Communication. *IEEE Trans. Commun. Technol.*, 5:835–848, 1971.

[35] B. Hughes. On the Error Probability of Signals in Additive White Gaussian Noise. *IEEE Trans. Inform. Theory*, 37:151–155, 1991.

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

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

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

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

[40] D. Kirkpatrick. Optimal Search in Planar subdivisions. *SIAM J. Comput.*, 12:28–35, 1983.

[41] V. Klee. On the Complexity of d-dimensional Voronoi Diagrams. *Arch. Math.*, 34:75–80, 1980.

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

[43] D. E. Knuth. *The Art of Computer Programming*, volume 2, pages 290–291. Addison-Wesley, Reading, Mass., second edition, 1981.

[44] H. Kobayashi. Application of Probabilistic Decoding to Digital Magnetic Recording Systems. *IBM Journal Res. Develop.*, 15:64–74, 1971.

[45] T. S. Koubantitsas. Application of the Viterbi Algorithm to Adaptive Delta Modulation with Delayed Decision. *Proc. IEEE*, 62:1076–1077, 1975.

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

[47] D. T. Lee and F. P. Preparata. Computational Geometry — A Survey. *IEEE Trans. Comput.*, 33:1072–1101, 1984.

[48] W. U. Lee and F. S. Hill. A Maximum-likelihood Sequence Estimator with Decision-feedback Equalization. *IEEE Trans. Commun.*, 25:971–979, 1977.

[49] A. Lender. The Duobinary Technique for High Speed Data Transmission. *IEEE Trans. Commun. Electron.*, 82:214–218, 1963.

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

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

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

[53] D. A. Leonard and C. A. Rodger. Limiting Error Propagation in Viterbi Decoding of Convolutional Codes. *IEEE Trans. Inform. Theory*, 35:1295–1299, 1989.

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

[55] H. Nyquist. Certain Topics in Telegraph Transmission Theory. *AIEE Trans.*, 47:617–644, 1928.

[56] J. O'Rourke. Computational Geometry. *Ann. Rev. Comput. Sci.*, 3:389–411, 1988.

[57] S. Pasupathy. Correlative Coding: A Bandwidth-efficient Signalling Scheme. *IEEE Commun. Soc. Magazine*, 15(4):4–11, 1977.

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

[59] M. I. Shamos and D. Heoy. Closest-point Problems. *Proc. 16th IEEE Annu. Symp. Found. Comput. Sci.*, pages 151–162, 1975.

[60] K. Trondle and G. Soder. *Optimization of Digital Transmission Systems.* Springer-Verlag, Heidelberg, 1987.

[61] G. Ungerboeck. Adaptive Maximum-Likelihood Receiver for Carrier-Modulated Data-Transmission Systems. *IEEE Trans. Commun.*, 22:624–636, 1974.

[62] G. Ungerboeck. Channel Coding with Multilevel/Phase Signals. *IEEE Trans. Inform. Theory*, 28:55–67, 1982.

[63] K. Wesolowski. An Efficient DFE & ML Suboptimum Receiver for Data Transmission over Dispersive Channels using Two-dimensional Signal Constellations. *IEEE Trans. Commun.*, 35:336–339, 1987.

[64] R. W. Wood and D. A. Petersen. Viterbi Detection of Class IV Partial Response on a Magnetic Recording Channel. *IEEE Trans. Commun.*, 34:454–461, 1986.