# SPHERE DECODING ALGORITHMS FOR DIGITAL COMMUNICATIONS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Haris Vikalo
October, 2002

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Thomas Kailath
(Principal Adviser)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Babak Hassibi

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Stephen P. Boyd

Approved for the University Committee on Graduate Studies:

# Abstract

The problem of finding the least-squares solution to a system of linear equations where the unknown vector is comprised of integers, while the given vector and the coefficient matrix are comprised of real entries, arises in many applications: digital communications, cryptography, GPS, to name a few. This problem is equivalent to the closest-point search in a lattice and is known to be NP-hard. However, in communications applications, where maximum-likelihood detection often reduces to solving an integer-least-squares problem, the given vector is not arbitrary, but rather is an unknown point in a finite lattice that has been perturbed by an additive noise vector of known statistical properties. We study the expected complexity of the integer-least-squares problem, averaged over the noise and over the lattice. For the "sphere decoding" algorithm of Fincke and Pohst we find analytic expressions for the expected complexity, both for the case when the lattice generator matrix is full (which models flat-fading multiple-antenna channels) and when the generator matrix is Toeplitz (which models frequency-selective channels). We show that for a wide range of signal-to-noise ratios the expected complexity is polynomial, in fact often sub-cubic. Many communication systems operate at noise levels for which this is the case, suggesting that the maximum-likelihood detection, which was hitherto thought to be computationally intractable, can in fact be implemented with the complexity similar to the heuristic methods, but with significant performance gains – a result with many practical implications.

In addition, extending the sphere-constrained search idea of the Fincke-Pohst algorithm, we develop algorithms that solve several important integer least-squares

related problems in the communications setting. For channels with memory, we propose a combination of the constrained search technique of sphere decoding and the dynamic programming principles of the Viterbi algorithm. The resulting algorithm has the worst-case complexity of the Viterbi algorithm but is significantly faster on average. Furthermore, motivated by the prospect of high reliability that iterative decoding techniques have been shown to provide on single-antenna channels, and by the absence of efficient techniques to obtain the soft-information (which is required of the iterative scheme) in multiple-antenna systems, we propose a modification of the Fincke-Pohst algorithm that solves the maximum a posteriori detection problem and efficiently approximates the required soft-information. We demonstrate that the algorithm provides near-capacity performance when used for soft iterative detection in multiple antenna communication systems that employ powerful turbo and low density parity check codes. Furthermore, we develop a new algorithm for solving integer least-squares problems when the lattice points are generated by a linear block transform of elements in a Galois field and employ it for joint maximum-likelihood detection and decoding of the linear block codes. The joint detection and decoding algorithm exhibits superior performance over traditional receivers whereas it has comparable complexity.

# Contents

# List of Figures

# Chapter 1

# Introduction

In his landmark work [1] which sprang the fields of digital communications and information theory, Claude Shannon stated that "the fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point." The design of efficient yet simple communications systems that are capable of solving this problem has been the ongoing focus of research in the communications and information theory fields ever since. The communication systems should be designed to be effective, i.e., to be capable of exploiting the available resources to the fullest extent possible. On the other hand, they should remain simple to be implementable in practice.

The measure of the effectiveness of a communication system on a given channel model is provided by information theory. The ultimate limit of the performance – the channel capacity – was first calculated by Shannon in [1]. As shown there, there exist codes such that, for any data transmission rate less than the channel capacity, allow the probability of the decoding error to be made arbitrarily small. However, the proof of this statement was non-constructive and the codes achieving the capacity were allowed to be arbitrarily long random codes. Gallager [2] later showed that the decoding error probability decreases exponentially with the length of the code. However, in both [1] and [2], it was assumed that the decoder was designed according to the maximum-likelihood criteria. Maximum-likelihood decoding of a

random code requires an exhaustive search over all the possible codewords, and so the computational complexity of the optimal decoding scheme is exponential in the length of the codeword. As it was shown in the years to come, the major obstacle to the design of the practical effective communications systems has been the complexity of the detection and decoding algorithms.

The most meaningful statistical measure of the performance of a digital communication system is the probability of the transmission error, i.e., the probability that a received message is mistaken for another one from the available ensemble of the messages. For equally likely codewords, maximum-likelihood decoding minimizes the probability of decoder error. Hence, maximum-likelihood naturally arises as the target criterion for the decoder design. Nevertheless, the existing algorithms for maximum-likelihood decoding have computational complexity that is not feasible for implementation in practical systems. Hence, in order to recover the transmitted message, designers often turn to other design criteria and to heuristic techniques. Traditionally, rather than trying to directly decode the transmitted message, the receivers are often designed to first attempt to invert the effect of the communication channel and to optimize for the recovered signal in some sense. For instance, when recovering the transmitted signal at the receiver, some form of the mean-square error criterion is often considered. Essentially, by exploiting the knowledge of the channel at the receiver, this technique performs the minimization of the mean-square error between the original transmitted signal and and its recovered estimate. If, in addition, the filters in the receiver are constrained to be linear, one obtains a low complexity solution readily implementable in practice. However, since the probability of error is the most important criterion, the performance of this (and other) low-complexity heuristic solution is inferior in comparison to the optimal decoding scheme.

The implementation details of the algorithm which solves for a particular target criterion depend on the channel model. The mathematical model of the channel needs to reflect the physical characteristics of the communication medium while, at the same time, remaining tractable and allowing for implementable algorithms of low complexity. This may be satisfied by assuming appropriate forms for the stochastic

processes that play a role in the transmission. For instance, relying on the central limit theorem, we often model the noise disturbance affecting transmission as a Gaussian process. As a result, maximum-likelihood detection is equivalent to a least-squares problem, and often to a so-called integer least-squares minimization problem, which we discuss later in this section.

There are various important aspects of the data transfer that need to be considered in the design of a communication system. Some of the most prominent ones are:

- Compressing the data to the most compact form which still conveys all the information contained in the original message, and, after the transmission, recovering the data to the original form.
- Embedding redundancy into the data in order to provide protection from the adversities that occur in the form of the channel and noise disturbances.
- Statistically modeling the effects of the transmission medium, i.e., modeling the communication channel and statistically describing the noise. Also, developing techniques for channel estimation and tracking.
- Designing a receiver which efficiently recovers the transmitted data.

These are all areas of study in their own right. However, in this thesis we shall focus on the design of efficient algorithms for the optimal data detection. The other modules of the digital communications system will be briefly addressed in this chapter. Furthermore, since the design of the elements of a communication system is often tightly intertwined, we shall occasionally revisit some of them in more details in the remainder of the thesis, when their closer examination contributes to the understanding of the structure and the derivation of the optimal receiver.

We continue this chapter by describing the basic setting of the digital communications problem. We proceed by discussing the capacity of various communication systems. Then we introduce the integer least-squares problem and give an overview of existing techniques for solving it. We close the chapter by providing an outline of the thesis.

## 1.1 Fundamentals of Digital Communications

The general scheme of data transmission in a digital communication system is illustrated in Figure 1.1. The information source in Figure 1.1 generates a message

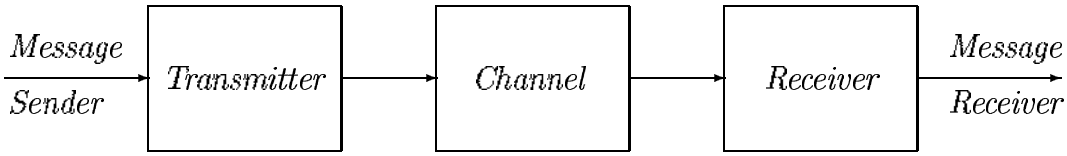


Figure 1.1: *Digital Communication System*

in the form of a digital sequence of bits. The transmitter then sends an appropriate continuous-time waveform representing the message through the communication channel. The receiver recovers the transmitted signal that was corrupted by the transmission media, and decides upon the message that has been sent.

Figure 1.2 shows the transmitter in more detail. The source encoder removes the

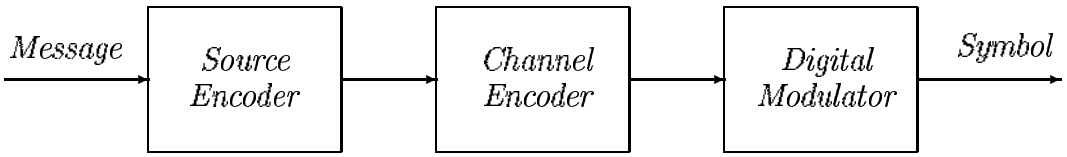


Figure 1.2: *Transmitter*

redundancies in the original message by compressing it into a more compact form. Then the channel encoder embeds some redundancies into the message in order to protect it from the errors caused by the communication channel. [We should note that the design of the source and channel encoder may also be done jointly. In fact, joint source-channel coding has been an active and fruitful area of research in the field of coding theory – see, e.g., [3] and references therein.] A digital modulator converts the output from the channel encoder to an analog waveform that is convenient for transmission through the communication medium. This is done in two stages. First,

the modulator maps the message into a symbol. The symbol is a vector with, in general, complex valued entries and it corresponds to a point in Euclidean space. The collection of all possible symbol points that are allowed to be transmitted across the channel is called the constellation. The constellation is typically designed by making a compromise between the opposing requirements that the constellation points be set as far apart from each other as possible while, at the same time, keeping the expected energy of the transmitted symbols as low as possible. Figure 1.3 illustrates a popular rectangular constellation – quadrature amplitude modulation (QAM) [4]. Typically, the spacing between any two constellation points in Figure 1.3 along either axis is an (possibly scaled) integer.

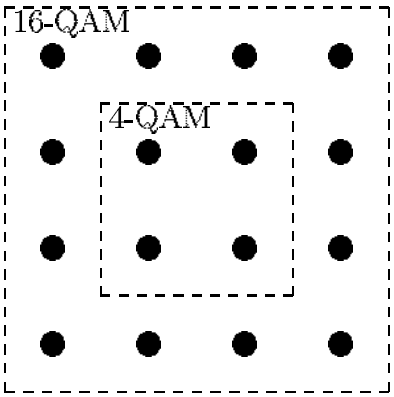


Figure 1.3: 4-*QAM and* 16-*QAM Constellations*

A possible mapping of the bits at the output of the channel encoder to the 16-QAM constellation from Figure 1.3 is shown in Figure 1.4. The bits (in groups of 4) are assigned to the symbol points using the so-called Gray mapping. The Gray mapping has the property that the nearest neighbors in the constellation are separated by a Hamming distance of 1. This mapping strategy is justified by the observation that for the uncoded transmission, the most likely error – confusing the transmitted symbol for its nearest neighbor – results in only one bit error. [For the best labeling strategies in coded systems, see, e.g., [5], [6].]

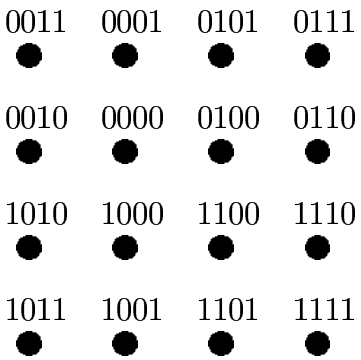


Figure 1.4: *The Gray mapping for* 16*-QAM constellation*

Once the message is mapped onto a constellation point, each component of the symbol vector is used to modulate an appropriate continuous-time basis function. The linear combination of the modulated components of the symbol vector is the analog waveform that is sent through the transmission media. The *rate* of the data transmission is determined by the number of bits sent across the channel each time the channel is used, i.e., by the total number of bits represented by the single modulated analog waveform sent. Note that in order to find the effective rate, or the *information* rate, we need to also factor in the amount of the redundancy that the channel encoder plants into the transmitted bit sequence.

The communication channel models the effects of the transmission medium. For example, in wireless communication systems, because of the interaction of the signal with the environment (e.g., scattering, reflection and refraction), the transmitted signal effectively propagates along a number of different paths. As a result of the propagation loss and fading, the signal gets attenuated during the transmission. These effects of the propagation medium are often modeled by a multiple path channel with interfering signals and additive white Gaussian noise (see Figure 1.5).

Primarily motivated by the desire to keep the model tractable, we frequently assume that the channel is linear time-invariant (LTI). Linearity is often justified by the practical limitations set on the dynamic range of the signals that are being transmitted. Because of these limitations, the non-linear distortions that might be caused by

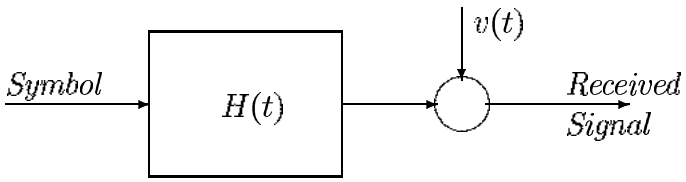


Figure 1.5: *General Channel Model*

the physical channel do not actually take place. On the other hand, time-invariance is an appropriate assumption for quasi-stationary channels, i.e., for scenarios where the channel changes slowly over the time interval that is being considered. For instance, the discrete-time "block-fading" model, where the channel is constant for some discrete interval $T$, after which it changes and remains constant for a duration of another interval $T$, is a tractable approximation of the continuously fading channel models, such as Jakes' model [7]. In general, the channel in Figure 1.5 has a memory and is mathematically described by an impulse response. Furthermore, the inputs and/or the outputs of the channel may be vector (rather than scalar) valued, as is the case in wireless communications systems employing multiple antennas. Therefore, under the LTI assumption, the most general mathematical description of the channel in Figure 1.5 is a (finite) sequence of matrices that comprise the channel impulse response.

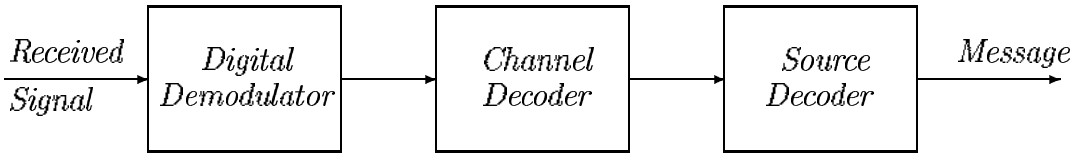


Figure 1.6: *Receiver*

The receiver is pictured in more detail in Figure 1.6. The task of the demodulator in Figure 1.6 is to recover the discrete message that was the output of the channel encoder. To do so, the received signal is first converted to baseband (if needed) and then filtered. The receive filter are matched either to the pulse shaping filter or to

the overall channel response, i.e., the combination of the pulse shaping filter and the channel. The output of the receive filters is then sampled and quantized to perform an analog to digital conversion. The resulting discrete sequence is processed by an algorithm that recovers the channel encoder output.

The last two stages of the receiver in Figure 1.6 use the output of the demodulator to recover the original message. First, the channel decoder recovers the compressed message that is the output of the source encoder. Then, the source decoder decompresses the original message.

The performance of the system in Figure 1.1 is typically measured in terms of a bit-error-rate. The bit-error-rate (BER) is the average (over time) fraction of the information bit sequence that was erroneously recovered at the receiver. Modern communication systems are characterized by ever increasing demands for high data rate service with high reliability (i.e., the low BER).

## 1.2 Limits of Performance

Channel capacity, error exponent, and cutoff rate are some of the information-theoretic concepts most often used to measure the efficiency of a communications scheme. We shall primarily focus on channel capacity as the most important one. Channel capacity provides a limit to the amount of the information that can be transmitted across the channel and still be recovered at the receiver with negligible probability of the error.

In information theory, communication channel is abstracted by a statistical description. Both the input and the output of the channel are random variables. Denoting the transmitted signal in Figure 1.5 by $x$ and the received signal by $y$, the channel is described by the conditional probability distribution $p_{Y|X}(y|x)$. To quantify the amount of information about transmitted signal that is gained by observing received signal, one considers mutual information between $X$ and $Y$, defined as

$$I(X;Y) = h(Y) - h(Y|X),$$

where $h(Y)$ is the entropy of the of the received signal, and $h(Y|X)$ is the entropy of the received signal conditioned that the signal $x$ was transmitted. Since the entropy of the random variable $X$,

$$h(X) = -\int p_X(x) \log\left(p_X(x)\right) dx,$$

is a measure of its uncertainty, the mutual information can be described as the reduction in the uncertainty about one random variable due to the acquired knowledge about the other [8].

Channel capacity of a single user system where the transmitter sends the signal drawn from the distribution described by the probability density function $p_X(x)$ is given by [1]

$$C = \max_{p_X(x)} I(X;Y),$$

i.e., channel capacity is mutual information between input and output signals, maximized over all possible distributions of the input. The significance of capacity is noted in Shannon's famous coding theorem. Essentially, assuming the communication scheme that employs the channel code at the transmitter, the theorem states that for all transmission rates $R < C$, there exist a coding scheme with arbitrarily low block and bit error rates as the length of the channel code goes to infinity. Conversely, for rates greater than channel capacity, error-free communication is not possible.

Information capacity is the maximum number of bits per channel use that can be reliably transmitted. i.e, information capacity is the actual communication capacity. For an additive white Gaussian noise (AWGN) channel $H$ that is constant and known to the receiver, the capacity can be found as

$$C = \max_{p(x)} I(X, Y|H) = \log(1+\rho) \text{ bits/sec/Hz}, \tag{1.1}$$

where $\rho$ denotes the signal-to-noise ratio (SNR), defined as the ratio of the transmit power and the power of the noise disturbance. From (1.1) it can be inferred that to transmit an additional bit, we require an increase of the transmit power corresponding

to the 3dB increase of the SNR.

The channel is not always constant. In wireless communications, for instance, the channel varies (fades) with time. The capacity, in that case, is a random variable. The statistics of the channel capacity provide insight to the system designer about the acceptable range of the system parameters. For an ergodic fading channel, expectation of mutual information can be achieved by coding which is long in comparison to the fading speed. For such a channel, the average mutual information is referred to as the *ergodic capacity* [9] and is defined as

$$C = E \log(1 + \rho|h|^2), \tag{1.2}$$

where $|h|^2$ is the squared value of the random, complex-valued channel gain, and the expectation is taken over all channel realizations. If the channel gain is a complex Gaussian random variable, $|h|^2$ is a random variable itself, described by a chi-square distribution with two degrees of freedom.

On the other hand, *outage capacity* is a frequently used measure of performance of practical systems [10]. Outage capacity is defined as the probability $\eta_o$ that the channel will not be able to support transmission at the given rate,

$$Prob(C \leq C_o) = \eta_o.$$

This can also be expressed as the lower bound: the outage capacity $C_o$ is the data rate that can be sustained with probability $1 - \eta_o$, i.e.,

$$Prob(C > C_o) = 1 - \eta_o.$$

For instance, the $\eta_o = 5\%$ outage capacity $C_o$ means that 95% of time the channel will be able to support data transmission at the rate $R = C_o$. We should note that the outage capacity is not a capacity in an information-theoretic sense but is still often used in practice.

Expression (1.2) gives the ergodic capacity for flat-fading (memoryless) channels.

Channels with memory (or the frequency-selective channels), are often represented by a tapped delay line model. Assuming $l$ taps, the channel impulse response is given by

$$h(t) = h_0\delta(t) + h_2\delta(t-1) + \ldots + h_{l-1}\delta(t-l+1).$$

The ergodic capacity for such a channel is given by

$$C = E\int_f \log\left(1 + \frac{S_x(f)|H(f)|^2}{S_v(f)}\right) df, \tag{1.3}$$

where $S_x(f)$ and $S_v(f)$ denote the power spectral densities of the input signal and the noise, respectively, $H(f)$ is the channel frequency response (i.e., the Fourier transform of the channel impulse response), and the integration is over the frequency range of signaling, that is, over the allotted bandwidth.

The capacity expressions often indicate the optimal transmitter signaling schemes. For instance, when the channel information is available at the transmitter, the capacity expression (1.3) is maximized by the so-called frequency water-pouring strategy [8]. However, the knowledge of the channel realization at the transmitter assumes that there is an information feedback from the receiver, which is not always readily available, as, for instance, in the mobile communications. Hence in this thesis, unless stated differently, we shall be making the assumption that the transmitter does not know the realization of the channel.

The variations in the frequency response of the channel provide the frequency branches that can mitigate the bad conditions (i.e., the low SNRs) in others. This frequency *diversity* is especially beneficial to the outage capacity – as the length of the channel impulse response increase while the power is being uniformly distributed over the channel taps, the outage capacity approaches the ergodic capacity in probability [11]. In general, diversity techniques exploit the variations (in time, frequency, etc.) of the channel to help improve the performance of the system. In the quest for the ever higher rates of the data transmission, the communication schemes seek for the various ways of exploiting the existing diversity provided by the channel or embedding the

additional ones by means of the system design. Recently, spatial diversity originating from the use of multiple antennas, and its effect on the channel capacity, have received significant attention ([12], [13]).

The ergodic capacity for systems with a single transmit and multiple (say, $N$) receive antennas, which is an often used configuration in practical wireless schemes, is given by

$$C = E\log(1+\rho\|h\|^2), \tag{1.4}$$

where $\|\cdot\|^2$ denotes the norm of its vector argument. Assuming that $h_i, i = 1,\ldots,N$ is a Gaussian $\mathcal{N}(0,1)$ random variable, we see from (1.4) that the increase in capacity that comes with the use of multiple antennas at the receiver is only logarithmic.

In [12], the capacity of systems employing multiple antennas at both the transmitter and the receiver was considered. The ergodic capacity was found as

$$C = E\log\left[\det\left(I_M + \frac{HR_{XX}H^*}{N_0}\right)\right],$$

where $R_{XX}$ is the signal covariance matrix and $N_0$ is the noise power. When the channel is white and its realization is not known at the transmitter, the ergodic capacity is given by

$$C = E\log\left[\det\left(I_N + \frac{\rho}{M}HH^*\right)\right], \tag{1.5}$$

where the SNR $\rho$ is defined via the total transmit energy $E_s$ as $\rho = \frac{E_s}{MN_0}$. Denoting the eigenvalues of $HH^*$ by $\lambda_i$, (1.5) can be written as

$$C = E\log\prod_{i=1}^{q}\left(1+\frac{\rho}{M}\lambda_i\right), \tag{1.6}$$

where $q \leq \min(M,N)$ is the rank of $H$. Assuming rich scattering, i.e., assuming that $H$ is not rank deficient, capacity of the multiple antenna system increases linearly with the minimum of the number of the transmit and receive antennas, $\min(M,N)$.

In Figure 1.7, we plot the ergodic capacity versus signal-to-noise ratio for a single-input single-output ($M = N = 1$), a single-input multiple-output ($M = 1, N = 4$),

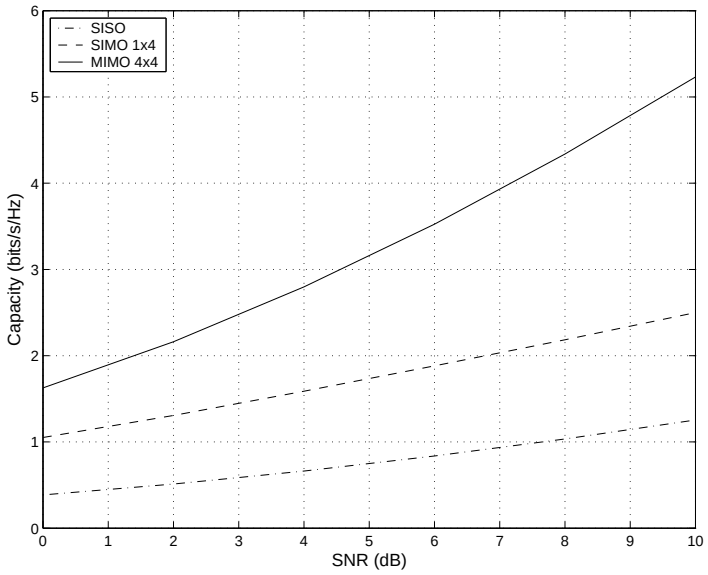


Figure 1.7: *Ergodic capacity comparison of the* $1 \times 1$, $1 \times 4$, *and* $4 \times 4$ *systems*

and a multiple-input multiple-output ($M = 4, N = 4$) systems. The linear increase in the capacity of the MIMO system over the capacity of the SISO system at the same SNR is evident.

We should mention that the capacity expressions discussed in this section are derived under the assumption that the transmitter can generate any distribution of the input signal – and, hence, it can also generate the Gaussian, being the optimal one. However, as we have discussed earlier, practical modulation schemes, such as the QAM modulation, impose a constraint on the achievable distributions of the transmitted signals. The mutual information for various constellation schemes was considered in [5]. Figure 1.8 shows the mutual information as a function of the SNR for the often used 4-QAM, 16-QAM, and 64-QAM constellations and compares them with the mutual information achieved with the Gaussian distribution of the input signal. The significance of these curves is that, as pointed out in [5], a QAM constellation of size $2^{R+1}$ is likely to yield a code with near-optimal performance at rate $R$ because, under those assumptions, the mutual information is within $1 - 2$ dB from the capacity curve. Therefore, when designing a system, one needs to consult

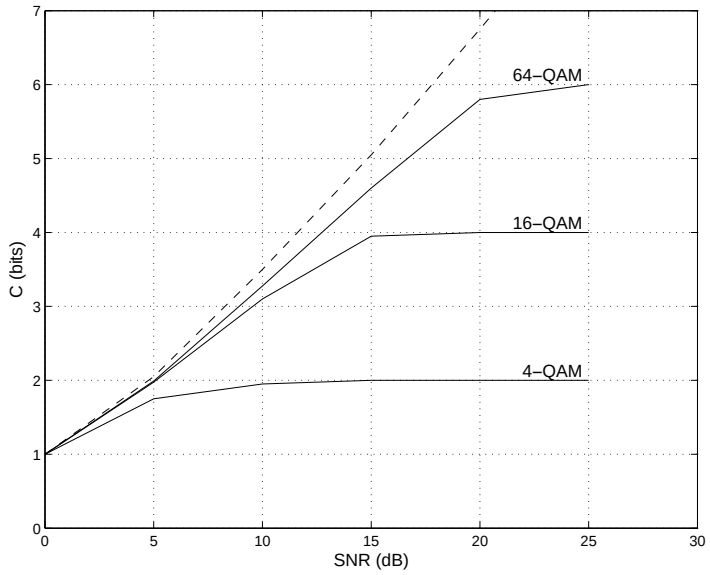


Figure 1.8: *Mutual information of the QAM modulation techniques on the single-input single-output channel. The dashed line is the mutual information for the input signal with Gaussian distribution.*

to Figure 1.8 for the proper choice of the modulation scheme.

## 1.3 Integer Least-Squares Problem

The main focus of this thesis is the design of demodulator algorithms that recover the channel encoder output or, in some cases, the channel encoder input. [In this text, we shall refer to the process of recovering the channel encoder output as *detection* and to the process of recovering the channel encoder input as *decoding*.] In the design, we shall consider those target criteria that minimize the probability of error: maximum-likelihood and maximum a posteriori criteria.

We consider transmission of data packets across linear time-invariant channels, corrupted by additive Gaussian noise. When the transmitted symbols are elements of a QAM constellation, the maximum-likelihood and the maximum a posteriori criteria can be expressed as equivalent non-linear optimization problems with integer

unknowns. We shall illustrate this equivalence for the case of the detection with the maximum-likelihood (ML) criterion. Assume that the received signal is given by

$$\mathbf{x} = \mathbf{H}\mathbf{s} + \mathbf{v}, \tag{1.7}$$

where $\mathbf{H}$ is the complex valued channel whose realization is known to the receiver (and is estimated, for instance, by means of sending a known training sequence), $\mathbf{s}$ is the transmitted symbol, and $\mathbf{v}$ is Gaussian $\mathcal{C}(0, \sigma^2)$ noise. Furthermore, assume that the entries in the transmitted symbol vectors $\mathbf{s}$ in (1.7) are coming from a QAM constellation. For later discussion, it will be useful to first find a real-valued equivalent of the equation (1.7). To this end, let the vector $s$ be comprised of the real and imaginary parts of the vector $\mathbf{s}$, organized as

$$s = \begin{bmatrix} \mathcal{R}(\mathbf{s})^T & \mathcal{I}(\mathbf{s})^T \end{bmatrix}^T.$$

Furthermore, let $x$ denotes

$$x = \begin{bmatrix} \mathcal{R}(\mathbf{x})^T & \mathcal{I}(\mathbf{x})^T \end{bmatrix}^T,$$

let $v$ denotes

$$v = \begin{bmatrix} \mathcal{R}(\mathbf{v})^T & \mathcal{I}(\mathbf{v})^T \end{bmatrix}^T,$$

and let $H$ be given by

$$H = \begin{bmatrix} \mathcal{R}(\mathbf{H}) & \mathcal{I}(\mathbf{H}) \\ -\mathcal{I}(\mathbf{H}) & \mathcal{R}(\mathbf{H}) \end{bmatrix}.$$

Then the real-valued equivalent of the model (1.7) is given by

$$x = Hs + v. \tag{1.8}$$

The ML detector maximizes the likelihood that $x$ was received given that $s$ was sent, i.e., performs the optimization

$$\max_s p_{x|s}(x|s). \tag{1.9}$$

The optimization in (1.9) is performed over the entire symbol space, i.e., over all constellation points **s**. Since we assumed additive Gaussian noise, the conditional distribution of $x$ given $s$ is

$$p_{x|s}(x|s) = \frac{1}{2\pi} e^{-\frac{\|x-Hs\|^2}{2\sigma^2}}.$$

Hence the maximization of the conditional probability in (1.9) is equivalent to the non-linear optimization problem

$$\min_s \|x - Hs\|^2, \tag{1.10}$$

where the components of the unknown vector $s$ are integer numbers. We now state the optimization problem (1.10) more formally.

**Integer least-squares problem over infinite lattice:**

*Given the vector $x \in \mathcal{R}^n$, matrix $H \in \mathcal{R}^{n\times m}$, and an $m$-dimensional integer lattice $\mathcal{Z}^m$, find the vector $s \in \mathcal{Z}^m$ that minimizes*

$$\min_{s\in\mathcal{Z}^m} \|x - Hs\|_2, \tag{1.11}$$

■

The optimization problem (1.11) is over an infinite space of $m$-dimensional vectors $s$ with integer entries. It is an appropriate search space for, e.g., spatial lattices, as in the case of finding coordinates of a point using the global positioning system (GPS) [31]. However, in digital communication applications where the lattice models symbol constellations, the power constraints on the transmitted signal means that the search space is a (finite) subset of the infinite lattice, $\mathcal{D} \subset \mathcal{Z}^m$. There we have the following problem.

**Integer least-squares problem over finite lattice:**

*Given the vector $x \in \mathcal{R}^n$, matrix $H \in \mathcal{R}^{n \times m}$, and a finite subset $\mathcal{D}$ of an $m$-dimensional infinite integer lattice $\mathcal{Z}^m$, find the vector $s \in \mathcal{D}^m$ that minimizes*

$$\min_{s \in \mathcal{D} \subset \mathcal{Z}^m} \|x - Hs\|_2 . \tag{1.12}$$

■

The integer least-squares problem has a simple geometric interpretation. As the entries of $s$ run over the integers, $s$ spans the "rectangular" $m$-dimensional lattice, $\mathcal{Z}^m$. However, for any given *lattice-generating matrix* $H$, the $n$-dimensional vector $Hs$ spans a "skewed" lattice. (When $n > m$, this skewed lattice lives in an $m$-dimensional subspace of $\mathcal{R}^n$.) Therefore, given the skewed lattice $Hs$, and given a vector $x \in \mathcal{R}^n$, the integer least-squares problem is to find the "closest" lattice point (in a Euclidean sense) to $x$, as is illustrated in Figure 1.9.

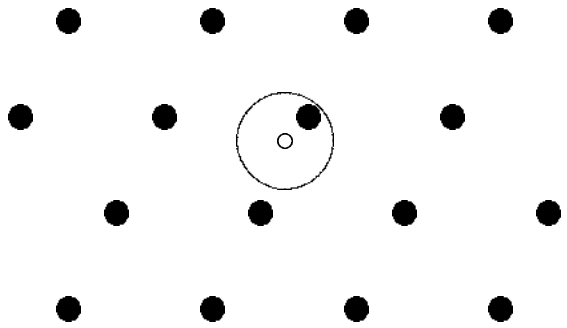

Figure 1.9: *Geometric interpretation of the integer least-squares problem*

Compared to the standard least-squares problem where the unknown vector $s$ is an arbitrary vector in $\mathcal{R}^m$, and the solution is obtained via a simple pseudo-inverse, it is much more difficult to find the solution to (1.11) or (1.12). The main reason is that the search space is discrete (whether it is finite or infinite). In fact, it is well known that problems (1.11) and (1.12) are, for a general $H$, exponentially complex, both in a worst-case sense [14], as well as in an average sense [25].

As we have just shown, when the channel is linear and the noise independent, identically distributed (i.i.d.) Gaussian, maximum-likelihood decoding leads to a

least-squares cost function. When the transmitted symbols are from a finite set, this can be often cast as an integer least-squares problem (1.12). However, integer least-squares problems do not arise only in the context of the simple uncoded transmission modeled by (1.7). It is also encountered in lattice codes [26, 38], CDMA systems [27, 28], multi-antenna systems possibly employing space-time codes [13, 29, 30], etc. In all these applications, the unknown vector $s$ represents the transmitted signal, the matrix $H$ represents the equivalent channel, and the vector $x$ represents the received signal. For example, in the multi-antenna context of V-BLAST [13] where we have $M$ transmit and $N$ receive antennas, $H$ is a $(m = 2M) \times (n = 2N)$ real channel matrix, and for linear space-time codes (such as those in [30]), $H$ is the equivalent channel matrix. The integer least-squares problem also arises when we have a frequency selective finite-impulse response (FIR) channel $H(z) = h_0 + h_1 z^{-1} + \ldots h_l z^{-l}$, in which case the channel matrix takes the form of a Toeplitz matrix

$$H = \begin{bmatrix} h_0 & & & \\ h_1 & h_0 & & \\ \vdots & \ddots & \ddots & \\ h_l & \ddots & \ddots & h_0 \\ & h_l & \ddots & h_1 \\ & & \ddots & \vdots \\ & & & h_l \end{bmatrix}. \tag{1.13}$$

We should point out that aside from the detection problem in the context of digital communications, integer least-squares problems appear in a host of other applications. Prominent applications include the GPS systems [31] and cryptography. In fact, there is a whole family of public-key cryptosystems based on the NP-hardness of the integer least-squares problem [32, 33, 34].

## 1.4 Overview of Solution Techniques

In this section, we review the existing techniques that are commonly used for solving integer least-squares problems. In particular, the solution techniques that we discuss are the following:

- heuristic techniques, which provide an approximate but readily implementable low-complexity solutions to the integer least-squares problem,
- lattice reduction methods that, when used for pre-processing of the channel matrix, improve the performance of the previously mentioned heuristics, and
- exact methods that, by exploiting the structure of the integer lattice, generally obtain the solution faster than straightforward exhaustive search.

### 1.4.1 Heuristic Techniques

Since the integer least-squares problem arises in many applications and finding the exact solution is, in general, NP hard, all practical systems employ some approximations, heuristics or combinations thereof with the manageable computational complexity. In communications applications, these approximations can be broadly grouped into several classes.

**Zero-forcing:**

Solve the unconstrained least-squares problem to obtain $\hat{s} = H^{\dagger}x$, where $H^{\dagger}$ denotes the pseudo-inverse of $H$. Since the entries of $\hat{s}$ will not necessarily be integers, round them off to the closest integer (a process referred to as slicing) to obtain

$$\hat{s}_B = \left[H^{\dagger}x\right]_{\mathcal{Z}}. \tag{1.14}$$

The above $\hat{s}_B$ is often called Babai estimate [14]. In the communications parlance, this procedure is referred to as the zero-forcing equalization.

The complexity of finding the Babai estimate is essentially determined by the complexity of finding pseudo-inverse of the matrix $H$ in (1.14). The simplest way of

calculating the pseudo-inverse is by means of QR factorization, $H = QR$. However, it can be calculated in a more stable way (which avoids inverting the upper triangular matrix $R$) by means of singular value decomposition (SVD) of $H$, $H = U\Sigma V^*$. The complexity of performing the SVD of $n \times m$ matrix $H$, as given in [15], is $2n^2m+11m^3$. The complexity of finding the pseudoinverse $H^\dagger = V\Sigma^{-1}U^*$ is $2nm^2$. Hence, assuming that $H$ is square (i.e, $n = m$), the complexity of finding the Babai estimate is of cubic order, $O(m^3)$.

**Nulling and cancelling:**

In this method, the Babai estimate is used for only one of the entries of $s$, say the first entry $s_1$. Then this entry, $s_1$, is assumed to be known and its effect is cancelled out to obtain a reduced-order integer least-squares problem with $m-1$ unknowns. The process is then repeated to find $s_2$, etc. In communications parlance this is known as the *decision-feedback equalization*.

We will find it convenient to denote the partition of the channel matrix $H$ into columns as

$$H = [\underline{h}_1 \;\; \underline{h}_2 \;\; \ldots \;\; \underline{h}_m].$$

The nulling and cancelling algorithm can be stated by the following pseudo-code:

$y_1 := x$
for $k = 0$ to $m-1$
  find weighting vector $w_{m-k}$
  $\hat{s}_{m-k} :=$ decision$(w_{m-k}y_{k+1})$
  $y_{k+2} = y_{k+1} - \underline{h}_{m-k}\hat{s}_{m-k}$
end

In the algorithm, for each value of the index $k$, the entries of the auxiliary vector $y_{k+1}$ are weighted by the components of the weight vector $w_{m-k}$ and linearly combined to account for the effect of the interference. Depending on the criterion chosen for the design of $w_{m-k}$ (i.e., for performing the nulling operation), we can distinguish between the following cases:

1. *Zero-forcing (ZF) nulling*

   In this case, interference from the yet undetected symbols is nulled. Denoting

   $$H_{m-k} = [\underline{h}_1 \;\; \underline{h}_2 \;\; \ldots \;\; \underline{h}_{m-k}],$$

   this condition can be stated as

   $$H^*_{m-k} w_{m-k} = e_{m-k},$$

   where $e_{m-k}$ is a $(m-k) \times 1$ column vector that consists of all zeros and has 1 for $(m-k)^{th}$ entry. The weighting vector $w_{m-k}$ for the zero-forcing nulling and cancelling algorithm is then given by the least-norm solution of the form

   $$w_{m-k} = H^{\dagger}_{m-k} e_{m-k},$$

   where $(\cdot)^{\dagger}$ denotes a pseudo-inverse, i.e., $H^{\dagger}_{m-k} = H_{m-k}(H^*_{m-k} H_{m-k})^{-1}$.

2. *Minimum mean-square error (MMSE) nulling*

   Objective in MMSE nulling is to minimize the expected mean-square error between the receiver's estimate and the transmitted symbol. This can be expressed as

   $$E\left[s_{m-k} y^*_{k+1}\right] = w^*_{m-k} E\left[y_{k+1} y^*_{k+1}\right].$$

   Furthermore, we shall assume that the previous decisions made by the detector were correct, i.e.,

   $$\hat{s}_{m-k} = s_{m-k}.$$

   Defining

   $$s_{1:m-k} = [s_1 \;\; s_2 \;\; \ldots \;\; s_{m-k}],$$

   we can write

   $$y_{k+1} = H_{m-k} s_{1:m-k} + v,$$

where $v$ is the noise vector in (1.8). Furthermore, assuming that the transmitted symbol sequence is white and has variance $\mathcal{E}_s$, we find

$$E\left[s_{m-k}y_{k+1}^*\right] = \mathcal{E}_s \underline{h}_{m-k}^*,$$

and

$$E\left[y_{k+1}y_{k+1}^*\right] = \mathcal{E}_s H_{m-k}H_{m-k}^* + \sigma^2 I,$$

where $\sigma^2$ denotes noise variance. Then the weighting vector $w_{m-k}$ for the MMSE nulling and cancelling algorithm is given by

$$w_{m-k} = \left(H_{m-k}^* H_{m-k} + \frac{1}{\rho}I\right)^{-1} \underline{h}_{m-k},$$

where $\rho = \frac{\mathcal{E}_s}{\sigma^2}$ denotes signal-to-noise ratio (SNR).

Computational complexity is again determined by the complexity of solving the underlying unconstrained least-squares problem, i.e., calculating pseudo-inverse at each step of the algorithm. Since we need to evaluate the pseudo-inverse of a series of matrices with dimensions $(n+i) \times i$, where $i = m, m-1, \ldots, 1$, total computational complexity is

$$\sum_{i=1}^{m}\left[2(n+i)^2 i + 11i^3 + 2(n+i)i^2\right] = n^2m^2 + 2nm^3 + \frac{15}{4}m^4.$$

When $m = n$ this complexity expression simplifies and is clearly of the fourth order, i.e., $O(m^4)$. Therefore, computational complexity of the nulling and cancelling is by an order of magnitude higher than the complexity of finding the Babai estimate.

**Nulling and cancelling with optimal ordering.**

In the nulling and cancelling algorithm we assumed that the previously made decisions are correct. However, the nulling and cancelling algorithm can suffer from *error-propagation*: if $s_1$ is estimated incorrectly it can have an adverse effect on

estimation of the remaining unknowns $s_2$, $s_3$, etc. To minimize effects of error propagation, it is advantageous to perform nulling and cancelling from the "strongest" to the "weakest" signal. This is the method proposed for V-BLAST [13].

Consider, for instance, the MMSE nulling and cancelling algorithms. To perform optimal ordering, we consider the covariance matrix of the estimation error $s - \hat{s}$,

$$P = E(s - \hat{s})(s - \hat{s})^* = \left( H^* H + \frac{1}{\rho} I \right)^{-1}.$$

Consider entries of the estimated symbol $\hat{s}$, i.e., $\{\hat{s}_i, i = 1, 2, \ldots, m\}$. The "strongest" signal, corresponding to the "best" estimate, is the one with the smallest variance, i.e., $s_i$ for which $P_{ii}$ is the smallest. If we reorder the entries of $s$ so that the strongest signal is $s_m$, then the estimate $\hat{s}_m$ is going to be better than it would be for any other ordering of the entries in $s$. This ordering we perform at each step of the nulling and cancelling algorithm, i.e., for each entry $s_k$, $(k = m, m-1, \ldots, 1)$, of the symbol vector $s$.

Therefore, the MMSE nulling and cancelling with optimal ordering can be described by the following pseudo-code:

$y_1 := x$
For $k = 0$ to $m - 1$
  Find $P = \left( H_{m-k}^* H_{m-k} + \frac{1}{\rho} I \right)^{-1}$ and $P_{jj} = \min_{i=1,\ldots,m-k} P_{ii}$
  Reorder $[s_1 \ \ldots \ s_{m-k}]$ so that $s_{m-k}^{(\text{new})} = s_{jj}$
  $w_{m-k} = \left( H_{m-k}^* H_{m-k} + \frac{1}{\rho} I \right)^{-1} \underline{h}_{m-k}$
  $\hat{s}_{m-k} := \text{Decision}(w_{m-k} y_{k+1})$
  $y_{k+2} = y_{k+1} - \underline{h}_{m-k} \hat{s}_{m-k}$
End

The computational complexity of the algorithm is as same as the complexity of the standard nulling and cancelling, augmented by the complexity of the ordering operation, which, for the set of $k$ elements, is $O(k^3)$.

**Square-root algorithm for nulling and cancelling:**

In order to avoid repeatedly evaluating the pseudo-inverse of the deflated matrices $H_m, H_{m-1}, \ldots, H_1$, one can use square-root algorithm as in [16]. Using the square-root algorithm, the computational complexity can be decreased by roughly an order of magnitude and is cubic, i.e., $O(m^3)$.

**Solving relaxed convex optimization problems:**

Another heuristic approach to the maximum-likelihood detection is via convex optimization techniques. The integer least-squares problem is essentially transformed into an optimization problem with both objective and constraint being convex functions. To illustrate the technique, we consider the detection problem where the entries in the symbol $s$ are chosen from $4-QAM$ constellations, i.e., for each entry in the symbol vector $s$ it holds that $s_i^2 = 1$.

Note that

$$\begin{aligned} \|x - Hs\|^2 &= s^T H^T H s - 2x^T H^T s + x^T x \\ &= \mathrm{Tr} H^T H S - 2x^T H^T s + x^T x, \end{aligned}$$

where $S = ss^T$. Therefore, the integer least-squares problem can be expressed as

$$\min \mathrm{Tr} H^T H S - 2x^T H^T s + x^T x$$

$$\text{subject to } S_{ii} = 1, S \succeq ss^T, \ \mathrm{rank}(S) = 1$$

Using

$$S \succeq ss^T \Leftrightarrow \begin{bmatrix} S & s \\ s^T & 1 \end{bmatrix} \succeq 0$$

and relaxing the rank one constraint, one can obtain semi-definite program (with variables $S$, $s$) of the form

$$\min \mathrm{Tr} H^T H S - 2x^T H^T s + x^T x$$

$$\text{subject to } S_{ii} = 1, \begin{bmatrix} S & s \\ s^T & 1 \end{bmatrix} \succeq 0.$$

Solving this SDP for $s$, an approximate solution to the detection problem can be found as $\hat{s} = \text{sgn}(s)$.

### 1.4.2 Lattice Reduction

The aforementioned heuristic methods are exact only if the columns of $H$ are orthogonal. In this case $H$ can be diagonalized by a unitary transformation on the left, and so slicing the unconstrained least-squares solution yields the exact solution.

In practice, however, the columns of $H$ are rarely orthogonal. Orthogonalizing the columns of $H$ via a QR decomposition, or otherwise, generally destroys the lattice structure. (The reason being that, if $s$ has integer entries, $Rs$ need not have integer entries.) One method that attempts to alleviate this is *lattice reduction*. In these methods, one attempts to find an invertible $M \times M$ matrix $T$, such that $T$ and $T^{-1}$ have integer entries (thereby preserving the lattice structure), and such that the matrix $G = HT$ is as "orthogonal as possible". Having found such a $T$, rather than solve (1.11), one can solve the integer least-squares problem

$$\min_{t \in \mathcal{Z}^m} \; \|x - Gt\|_2 \,, \tag{1.15}$$

using the earlier mentioned heuristics and set $s = T^{-1}t$. However, lattice reduction is itself NP-hard.

Let $\Lambda$ denotes the lattice generated by the matrix $H$. The classic lattice reduced basis for $\Lambda$ are due to Minkowski [21] and Korkin and Zolotarev [22] (see also [23]).

**Minkowski-reduced basis:**

A basis $G = [\underline{g}_1 \;\; \underline{g}_2 \;\; \ldots \;\; \underline{g}_m]$ is Minkowski-reduced if

- $\underline{g}_1$ is the shortest nonzero vector in the lattice $\Lambda$,
- For each $k = 2, \ldots, m$, $\underline{g}_k$ is the shortest nonzero vector in $\Lambda$ such that $(\underline{g}_1, \ldots, \underline{g}_k)$ may be extended to a basis of $\Lambda$.

The reduced basis $G$ clearly contains the shortest nonzero vector in the lattice $\Lambda$,

$\underline{g}_1$. For every $k = 2, \ldots, m$, the basis vector $\underline{g}_k$ is the shortest nonzero vector in $\Lambda$ independent of $(\underline{g}_1, \ldots, \underline{g}_{i-1})$. Of course, finding shortest nonzero vector in a lattice is an NP-hard problem and there are no polynomial-time algorithms that could find the Minkowski-reduced basis.

**Korkine-Zolotarev reduction:**

A basis $G = [\underline{g}_1 \ \underline{g}_2 \ \ldots \ \underline{g}_m]$ is Korkin-Zolotarev-reduced if

- $\underline{g}_1$ is the shortest nonzero vector in the lattice $\Lambda$,
- For each $k = 2, \ldots, m$, define $\Lambda_k$ to be the $(k-1)$-dimensional subspace spanned by $G_{k-1} = [\underline{g}_1 \ldots \underline{g}_{k-1}]$, and denote the orthogonal complement of $\Lambda_k$ in $\mathcal{R}^m$ by $\Lambda_k^{\perp}$. Furthermore, let $S_k(\Lambda)$ denote the orthogonal projection of $\Lambda$ onto $\Lambda_k^{\perp}$. Then $\underline{g}_k$ is such that $S_k(\underline{g}_k)$ is the shortest nonzero vector in $\mathcal{S}_k(\Lambda)$.
- For $1 \leq k < j \leq m$,

$$\left|\left\langle S_k(\underline{g}_k), S_k(\underline{g}_j)\right\rangle\right| \leq \frac{1}{2}\|S_k(\underline{g}_k)\|^2,$$

and $S_1(\underline{g}_k) = \underline{g}_k$, for each $k$.

Therefore, in Korkine-Zolotarev reduction, the basis vector $\underline{g}_k$ is chosen based on its length in the orthogonal complement of the space spanned by $G_{k-1} = [\underline{g}_1 \ \ldots \ \underline{g}_{k-1}]$. It is known that every lattice has at least one KZ-reduced generator matrix [24]. As in the case of Minkowski-reduced basis, there are no known polynomial-time algorithms which find Korkin-Zolotarev reduced basis. Therefore, there is a need for lattice reduction techniques which could be performed in polynomial time, even if the orthogonality conditions are relaxed. The most famous such algorithm is the Lenstra-Lenstra-Lovasz (LLL) reduction. Permitting a gross oversimplification, LLL algorithm can be regarded as Gram-Schmidt over integers.

**Lenstra-Lenstra-Lovasz reduction:**

A basis $G = [\underline{g}_1 \ \underline{g}_2 \ \ldots \ \underline{g}_m]$ is LLL-reduced if

- $|\mu_{ij}| \leq \frac{1}{2}$ for $1 \leq j < i \leq m$,

- $||\underline{h}_i^* + \mu_{i,i-1}\underline{h}_{i-1}^*||^2 \geq \eta||\underline{h}_{i-1}^*||^2$ for $1 < i \leq m$,

where

$$\mu_{i,j} = \frac{(\underline{g}_i, \underline{h}_j^*)}{(\underline{h}_j^*, \underline{h}_j^*)},$$

$H^* = [\underline{h}_1^* \;\; \underline{h}_2^* \;\; \ldots \;\; \underline{h}_m^*]$ is a Gram-Schmidt orthogonalized basis generated from $H$, and parameter $\eta \in (\frac{1}{4}, 1)$.

Both the Minkowski and Korkin-Zolotarev reductions produce bases that are "more orthogonal" than the LLL-reduced basis (i.e., the orthogonality conditions are more strict). However, unlike for the other two, there is an efficient polynomial-time algorithm for finding the LLL-reduced basis [35]:

- Compute vectors $\underline{h}_1^*, \underline{h}_2^*, \ldots, \underline{h}_m$ by performing Gram-Schmidt orthogonalization procedure on the vectors $\underline{h}_1, \underline{h}_2, \ldots, \underline{h}_m$, i.e., perform the recursion

$$\underline{h}_j^* = \underline{h}_j - \sum_{i=1}^{j-1} \frac{\underline{h}_j^T \underline{h}_i^*}{||\underline{h}_j^*||^2} \underline{h}_i^*, \quad \underline{h}_1^* = \underline{h}_1, \quad j = 2, \ldots q. \tag{1.16}$$

  The vectors $\{\underline{h}_1^*, \underline{h}_2^*, \ldots, \underline{h}_m\}$ comprise an orthogonal basis for the subspace spanned by $\{\underline{h}_1^*, \underline{h}_2^*, \ldots, \underline{h}_m\}$. From (1.16), it is clear that any vector $\underline{h}_j$ can be expressed as a linear combination of vectors $\{\underline{h}_1^*, \underline{h}_2^*, \ldots, \underline{h}_m\}$ as

$$\underline{h}_j = \sum_{i=1}^{j} \mu_{ji} \underline{h}_i^*, \tag{1.17}$$

  where $\mu_{ji} = \underline{h}_j^T \underline{h}_i^* / ||\underline{h}_j^*||^2$ for $i = 1, 2, \ldots, j-1$, and $\mu_{jj} = 1$.

- For $j = 1, 2, \ldots, q$, and given $j$, for $i = 1, 2, \ldots, j-1$, replace $\underline{h}_j$ by $\underline{h}_j - \lceil \mu_{ji} \rfloor \underline{h}_i$, where $\lceil \mu_{ji} \rfloor$ is the integer nearest to $\mu_{ji}$. Update the $\underline{h}_j^*$'s and $\mu_{ji}$'s with the new $\underline{h}_j$'s according to (1.16) and (1.17).

- If there is a subscript $j = 1, \ldots, q$ violating

$$\|\underline{h}^*_{j+1} + \mu_{(j+1)j}\underline{h}^*_j\|^2 \geq \frac{3}{4}\|\underline{h}^*_j\|^2,$$

interchange $\underline{h}_j$ and $\underline{h}_{j+1}$ and return to the first step; otherwise, stop. The matrix $G = [\underline{h}_1 \ \underline{h}_2 \ \ldots \ \underline{h}_q]$ is the reduced generator matrix[1].

While lattice reduction may lead to some improvement in the solution of (1.11), the integer least-squares problem over infinite lattice, it is not useful for (1.12), where the solution belongs to a finite subset of the infinite lattice. The reason is that the lattice transforming matrix $T$ often destroys properties of the subset $\mathcal{D} \subset \mathcal{Z}^m$. In communications applications, we are typically interested in *finite* subsets of infinite integer lattice. Therefore, we shall not consider lattice reduction methods in this thesis.

### 1.4.3 Exact Methods

With such an abundance of heuristic methods, it is natural to ask what their performance is, and how close they come to the optimal solution? In [29] this question is studied in the context of V-BLAST where it is shown, by simulation, that an exact solution significantly outperforms even the best heuristics. This is shown in Figure 1.10, where a bit-error rate performance of an exact solution is compared with ordered nulling and cancelling for a multiple antenna system with $M = 8$ transmit and $N = 12$ receive antennas employing 16-QAM modulation scheme (the same system specifications as in V-BLAST [13]).

The above discussion shows that there is merit in studying exact solutions. The most obvious one is to search over the entire lattice which, although theoretically feasible for finite lattices, invariably requires an exponential search. When the channel matrix has the Toeplitz structure of (1.13), the Viterbi algorithm (see, e.g., [37]) can be used to exploit the special structure and provide the exact solution to the integer

[1]The coefficient $\frac{3}{4}$ is arbitrary; any number between $\frac{1}{4}$ and 1 is allowed

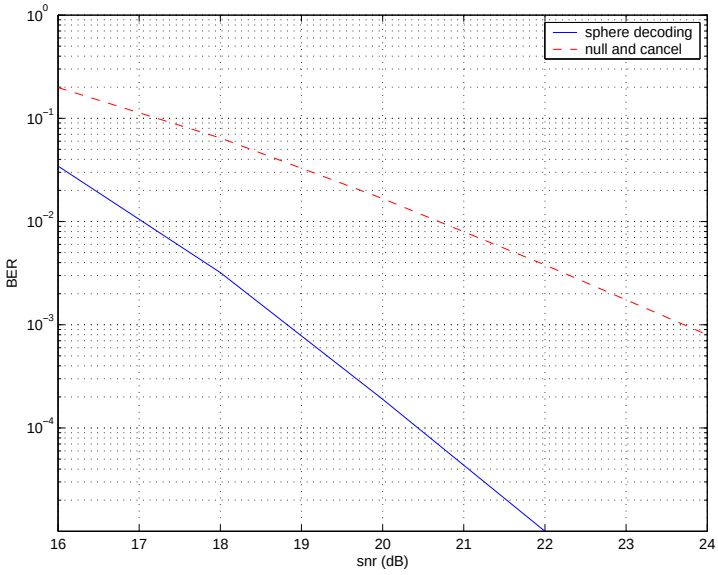


Figure 1.10: *Bit error performance of a sphere decoding vs. nulling and cancelling with optimal ordering,* $M = 8$, $N = 12$, 16-*QAM.*

least-squares problem with considerable computational savings. Nevertheless, these techniques can not help in the case of a channel matrix with no special structure. There do, however, exist exact methods that are more sophisticated than exhaustive search and can be employed for an arbitrary $H$. There are two such algorithms (along with some variations thereof [38]), Kannan's algorithm and the Fincke-Pohst algorithm:

- Kannan's algorithm [39], which searches only over those lattice points that belong to a restricted parallelogram, and
- Fincke and Pohst (or sphere decoding) algorithm [40], which performs search only over those lattice points that belong to a hyperdimensional sphere.

Although these exact algorithms do not perform exhaustive search over the entire lattice, intuition tells us that the complexity of the search may still be rather high. For instance, if the noise disturbance is unbounded, the point $x$ may be anywhere in space. Clearly, any bounded part of the space where the search is performed

needs to include the point $x$ as well. Therefore, in the case of unbounded noise, it is likely that any search region would contain quite a large number of lattice points and sorting through all of them may require close to the complexity of the exhaustive search. However, this scenario is unlikely in the context of digital communications. The point $x$ is not just any point in space – it is the lattice point perturbed by the noise of the known (often assumed i.i.d. Gaussian) statistics. As we will show in the next chapter, statistics of the channel and noise may be exploited to determine size of the sphere for the Fincke-Pohst algorithm. Solving integer least-squares (and related) problems by means of a localized search inside a sphere is the main focus of this thesis.

## 1.5 Outline of the Thesis

In this section, we give the motivation for the studies undertaken in the remainder of the thesis and describe the scope and contributions thereof.

In Chapter 2, we derive an analytical expression for the expected complexity of the sphere decoding algorithm. We start the chapter by giving a detailed description of the Fincke-Pohst procedure. Then we propose a tree pruning interpretation of the sphere decoding algorithm, and relate the complexity of the algorithm to the size of the tree. The radius of the sphere that needs to be searched is chosen according to the statistical properties of the noise. The statistical nature of the size of the region of search implies that computational complexity of the algorithm is a random variable. We use the intuition gained by the introduction of the previously mentioned tree structure to find the expected number of points in the search sphere and the first-order statistics of the computational complexity. As we show, over a wide range of data rates and signal-to-noise ratios, the expected complexity of sphere decoding is polynomial-time and, in fact, often sub-cubic, implying that in many cases of interest maximum-likelihood performance can be obtained with complexity similar to that of nulling and cancelling. The second-order statistics as well as the density function of the complexity are computed numerically.

The study of Chapter 2 was motivated by the current intensive research on the design of systems capable of providing high-speed wireless data services. In particular, physical limitations of the wireless medium present many challenges to the design of reliable communication systems. Multipath fading, interference, noise, as well as bandwidth restrictions, severely limit achievable transmission rates. As shown in [12], multiple antenna wireless communication systems are capable of providing data transmission at potentially very high rates. However, good decoding schemes for the multiple antenna systems may result in high complexity of the receiver. Therefore, in Chapter 2 we apply sphere decoding to the wireless systems with multiple antennas at both transmitter and receiver and compare its performance with that of the nulling and cancelling schemes. It will be shown that sphere decoding significantly outperforms the best among the existing heuristic techniques. Moreover, in the range of signal-to-noise ratios of interest, the expected complexity of the sphere decoding is shown to be polynomial.

In Chapter 3, we study sphere decoding for the channels with memory. The Viterbi algorithm [36] provides an elegant solution to the maximum-likelihood detection problem on frequency-selective channels but is computationally inefficient when employed for detection on long channels. Therefore, techniques that allow for easy implementation, albeit of inferior performance when compared to the Viterbi algorithm, are often considered in practice (see, e.g., [66]-[68]).

We first study how the sphere decoding algorithm can be employed on channels with memory without performing generally required QR factorization, and derive probabilistic quantities required to compute the expected complexity of the algorithm. Then we propose an algorithm that combines the sphere-constrained search strategy of the sphere decoding algorithm with the dynamic programming principles of the Viterbi algorithm. We consider computational complexity of the hybrid algorithm and show that it combines best of both worlds: it has polynomial expected complexity and deterministic worst-case complexity.

In Chapter 4, we derive a modification of the Fincke-Pohst algorithm to perform maximum a posteriori search and yields the probabilistic information needed

for decoding in soft iterative schemes. We start the chapter by a review of the most commonly used soft iterative decoding techniques. Soft iterative decoding has been reported to achieve impressive results for codes with long codeword length. Following the seminal paper by Berrou et al [49], there have been many results on turbo decoding, with performance approaching the Shannon limit on single-input single-output systems (see [46] and the references therein). More recently, low-density parity check (LDPC) codes, long neglected since their introduction by Gallager [47], have also been resurrected ([50]-[52]).

Crucial to both turbo and LDPC decoding techniques is the use of probabilistic ("soft") information about each bit in the transmitted sequence. For multiple antenna systems it is not clear what is the best way to obtain this soft-information with low complexity. As noted in [54], where turbo-coded modulation for multiple antenna systems has been studied, if soft information is obtained by means of an exhaustive search, the computational complexity grows exponentially in the number of transmit antennas and in the size of the constellation. Hence, for high-rate systems with large number of antennas, exhaustive search proves to be practically infeasible. Therefore, heuristic techniques are often employed to obtain the soft channel information ([54, 55]). To solve this problem, we modify the integer least-squares metric so that by optimizing for it, we are searching for the maximum a posteriori optimal point in the neighborhood of the received point $x$. The solution, along with the other points in the region, is used to approximate soft detector decisions and employ them as an input to the soft decoder. The performance of the algorithm is illustrated in systems employing both simple (convolutional) as well as turbo and LDPC error-correcting codes. When combined with the powerful channel codes, the algorithm results in near-capacity system performance. An upper bound on the expected complexity of the algorithm is also provided.

In Chapter 5, we consider the problem of joint detection and decoding in multi-input multi-output systems and develop algorithms that solve it for both the maximum-likelihood (JDD-ML) and the maximum a posteriori (JDD-MAP) criteria. Both algorithms that we propose in Chapter 5 draw on the ideas of sphere decoding. In

particular, they are based on searching the space in vicinity of the observed signal and looking for nearby lattice points. However, there is an important additional constraint imposed on the lattice points – they must be admissible codewords. Error-correcting codes that we consider are the linear block codes. Therefore, the symbol space is a subspace of the entire lattice, specified by the linear block transform.

Just as for sphere decoding, the computational complexity of the JDD algorithms is a random variable. We consider the expected complexity of the JDD-ML algorithm, which we find analytically. Furthermore, we provide an upper bound on the expected computational complexity of the JDD-MAP algorithm.

# Chapter 2

# Expected Complexity of Sphere Decoding

In this chapter, we study the computational complexity of sphere decoding algorithm. We assume a digital communication setup, in which channel model and noise statistics are readily available and are used to choose the size of the sphere to be searched. Due to probabilistic nature of the size of the sphere, complexity of the algorithm is a random variable. Thus, knowledge of the statistics of the complexity would provide a good indication of practical feasibility. To this end, we derive a closed-form expected complexity expressions for the closest point search in both infinite and finite lattices. We first find expected complexity for the lattice generating matrices that model flat-fading multiple antenna channels. Then we find the same for the lattice generating matrices that model frequency-selective channels. We show that the expected complexity is polynomial over a wide range of signal-to-noise ratios and often comparable to the complexity of the best known heuristic techniques. This implies that, contrary to common belief, maximum-likelihood detection may, in fact, be implemented in practical systems. The result is particularly interesting in multiple antenna systems because the benefits of finding the maximum-likelihood solution rather than an approximate solution provided by heuristic techniques, increase drastically with

the number of the antennas.[1] The findings are illustrated with extensive simulations of multiple antenna systems transmitting across both the flat-fading as well as the frequency-selective channels.

## 2.1 Sphere Decoding

The basic premise in sphere decoding is rather simple: we attempt to search over only those lattice points that lie in a certain hypersphere of radius $r$ around the given vector $x$, thereby reducing the search space and hence the required computations (see Figure 2.1). Clearly, the closest lattice point inside the hypersphere will also be the closest lattice point for the whole lattice. However, close scrutiny of this basic idea leads to two key questions.

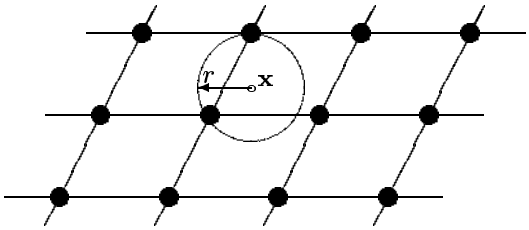


Figure 2.1: *Idea behind sphere decoder*

1. *How to choose r?* Clearly, if $r$ is too large, we obtain too many points and the search remains exponential in size, whereas if $r$ is too small, we obtain no points inside the hypersphere.

   A natural candidate for $r$ is the *covering radius* of the lattice, defined to be the smallest radius of spheres centered at the lattice points that cover the entire space. This is clearly the smallest radius that guarantees the existence of a

[1] This is due to increased number of degrees of freedom in the channel matrix, i.e., higher spatial diversity. Note that, at the same time, complexity of the full search grows exponentially with the number of antennas.

point inside the hypersphere for any vector $x$. The problem with this choice of $r$ is that determining the covering radius for a given lattice is itself NP hard [41]. Another choice is to use $r$ as the distance between the Babai estimate and the vector $x$, i.e., $r = \|x - H\hat{s}_B\|$, since this radius guarantees the existence of at least one lattice point (here the Babai estimate) inside the hypersphere. However, it is again not clear in general whether this choice of radius leads to too many lattice point lying inside the hypersphere.

2. *How can we tell which lattice points are inside the hypersphere?* If this requires testing the distance of each lattice point from $x$ (to determine whether it is less than $r$), then there is no point in sphere decoding as we will still need an exhaustive search.

Sphere decoding does not really address the first question. However, it does propose an efficient way to answer the second, and more pressing, one. The basic observation is the following. Although it is difficult to determine the lattice points inside a general $m$-dimensional hypersphere, it is trivial to do so in the (one-dimensional) case of $m = 1$. The reason is that a one-dimensional hypersphere is simply an interval and so the desired lattice points will be the integer values that lie in this interval. We can use this observation to go from dimension $k$ to $k+1$. Suppose we have determined all $k$-dimensional lattice points that lie in a hypersphere of radius $r$. Then for any such $k$-dimensional point, the set of admissible values of the $(k+1)$-th dimensional coordinate that lie in the higher dimensional sphere of the *same* radius $r$ can be expressed in the form of an interval.

The above means that we can determine all lattice points in a hypersphere of dimension $m$ and radius $r$ by successively determining all lattice points in hyperspheres of lower dimensions $1, 2, \ldots, m$ and the same radius $r$. Such an algorithm for determining the lattice points in an $m$-dimensional hypersphere essentially constructs and prunes a tree where the nodes in the $k$-th level of the tree correspond to the lattice points inside the hypersphere of radius $r$ and dimension $k$—see Figure 2.2. The paths that survive the tree pruning operation correspond to the vector points that belong

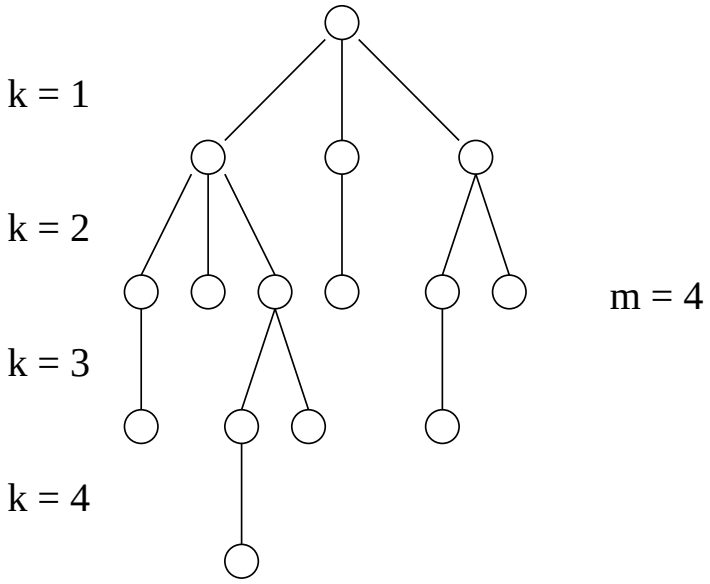


Figure 2.2: *Sample tree generated to determine lattice points in a 4-dimensional hypersphere.*

to the sphere of radius $r$ centered at the given point $x$. Moreover, the complexity of such an algorithm will depend on the *size* of the tree, i.e., on the number of nodes that the algorithm visits.

With this brief introduction we can now be more specific about the problem at hand. To this end, we shall assume that $n \geq m$, i.e., that there are at least as many equations as unknowns in $x \approx Hs$ (the case $n < m$ is considered further below). Note that the lattice point $Hs$ lies in a hypersphere of radius $r$ if, and only if,

$$r^2 \;\geq\; \|x - Hs\|^2 . \tag{2.1}$$

In order to break the problem into the subproblems described above, it is useful to consider the QR factorization of the matrix $H$

$$H = Q \begin{bmatrix} R \\ 0_{(n-m)\times m} \end{bmatrix},$$

where $R$ is an $m \times m$ upper triangular matrix and $Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}$ is an $n \times n$ orthogonal matrix. The matrices $Q_1$ and $Q_2$ represent the first $m$ and last $n-m$ orthonormal columns of $Q$, respectively. The condition (2.1) can therefore be written as

$$r^2 \geq \left\| x - \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R \\ 0 \end{bmatrix} s \right\|^2 = \left\| \begin{bmatrix} Q_1^* \\ Q_2^* \end{bmatrix} x - \begin{bmatrix} R \\ 0 \end{bmatrix} s \right\|^2 = \|Q_1^* x - Rs\|^2 + \|Q_2^* x\|^2 .$$

Or in other words,

$$r^2 - \|Q_2^* x\|^2 \geq \|Q_1^* x - Rs\|^2 . \tag{2.2}$$

Defining $y = Q_1^* x$ and $r'^2 = r^2 - \|Q_2^* x\|^2$ allows us to rewrite (2.2) as

$$r'^2 \geq \sum_{i=1}^{m} (y_i - \sum_{j=i}^{m} r_{ij} s_j)^2 .$$

Here is where the upper triangular property of $R$ comes in handy. The RHS of the above inequality can be expanded as

$$r'^2 \geq (y_m - r_{mm} s_m)^2 + (y_{m-1} - r_{m-1,m} s_m - r_{m-1,m-1} s_{m-1})^2 + \ldots \tag{2.3}$$

where the first term depends only on $s_m$, the second term on $\{s_m, s_{m-1}\}$ and so on. Therefore, considering the first term only, a necessary condition for $Hs$ to lie inside the hypersphere is that

$$r'^2 \geq (y_m - r_{mm} s_m)^2 .$$

This condition is equivalent to $s_m$ belonging to the interval

$$\left\lceil \frac{-r' + y_m}{r_{mm}} \right\rceil \leq s_m \leq \left\lfloor \frac{r' + y_m}{r_{mm}} \right\rfloor . \tag{2.4}$$

Of course, (3.3) is by no means sufficient. For every $s_m$ satisfying (3.3), defining

$$r'^2_{m-1} = r'^2 - (y_m - r_{mm} s_m)^2 ,$$

and

$$y_{m-1|m} = y_{m-1} - r_{m-1,m}s_m,$$

a stronger necessary condition can be found by looking at the first two terms in (3.2), which leads to $s_{m-1}$ belonging to the interval

$$\left\lceil \frac{-r'_{m-1} + y_{m-1|m}}{r_{m-1,m-1}} \right\rceil \leq s_{m-1} \leq \left\lfloor \frac{r'_{m-1} + y_{m-1|m}}{r_{m-1,m-1}} \right\rfloor. \tag{2.5}$$

One can continue in a similar fashion for $s_{m-2}$, and so on until $s_1$, thereby obtaining all lattice points satisfying (2.1).

### 2.1.1 The Sphere Decoding Algorithm

We can now formalize the algorithm.

*Input:* $Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}$, $R$, $x$, $y = Q_1^* x$, $r$.

1. Set $k = m$, $r'^2_m = r^2 - ||Q_2^* x||^2$, $y_{m|m+1} = y_m$
2. (Bounds for $s_k$) Set $UB(s_k) = \lfloor \frac{r'_k + y_{k|k+1}}{r_{k-1,k-1}} \rfloor$, $s_k = \lceil \frac{-r'_k + y_{k|k+1}}{r_{k-1,k-1}} \rceil - 1$
3. (Increase $s_k$) $s_k = s_k + 1$. If $s_k \leq UB(s_k)$ go to 5, else to 4.
4. (Increase $k$) $k = k + 1$; if $k = m + 1$ terminate algorithm, else go to 3.
5. (Decrease $k$) If $k = 1$ go to 6. Else $k = k - 1$, $y_{k|k-1} = y_k + \sum_{j=k+1}^{m} r_{kj}s_j$, $r'^2_k = r'^2_{k+1} - (y_{k+1} - r_{k+1,k+1}s_{k+1})^2$, and go to 2.
6. Solution found. Save $s$ and go to 3.

We should mention that the original paper of Fincke and Pohst [40] used slightly different notation to the one we have used. For completeness, we shall include it here. The paper [40] makes use of the unconstrained least-squares solution $\hat{s} = H^{\dagger}x = R^{-1}Q_1^* x$. In this case, it follows that

$$||Q_2^* x||^2 = ||x||^2 - ||H\hat{s}||^2,$$

and so inequality (2.2) becomes

$$r^2 - \|x\|^2 + \|H\hat{s}\|^2 \geq \|R(s - \hat{s})\|^2. \tag{2.6}$$

The expansion (3.2) becomes

$$\begin{aligned} r^2 &\geq r_{mm}^2 (s_m - \hat{s}_m)^2 \\ &+ r_{m-1,m-1}^2 \left( s_{m-1} - \hat{s}_{m-1} + \frac{r_{m-1,m}}{r_{m-1,m-1}} (s_m - \hat{s}_m) \right)^2 + \dots \end{aligned} \tag{2.7}$$

and the intervals (3.3) and (3.4)

$$\left\lceil \hat{s}_m - \frac{r'}{r_{mm}} \right\rceil \leq s_m \leq \left\lfloor \hat{s}_m + \frac{r'}{r_{mm}} \right\rfloor \tag{2.8}$$

and

$$\left\lceil \hat{s}_{m-1|m} - \frac{r'_{m-1}}{r_{m-1,m-1}} \right\rceil \leq s_{m-1} \leq \left\lfloor \hat{s}_{m-1|m} + \frac{r'_{m-1}}{r_{m-1,m-1}} \right\rfloor \tag{2.9}$$

respectively, where we have defined

$$\hat{s}_{m-1|m} = \hat{s}_{m-1} + \frac{r_{m-1,m}}{r_{m-1,m-1}} (s_m - \hat{s}_m).$$

We can now alternatively write the algorithm as

*Input:* $R$, $x$, $\hat{s}$, $r$.

1a. Set $k = m$, $r'^2_m = r^2 - \|x\|^2 + \|H\hat{s}\|^2$, $\hat{s}_{m|m+1} = \hat{s}_m$

2a. (Bounds for $s_k$) Set $z = \frac{r'_k}{r_{kk}}$, $UB(s_k) = \lfloor z + \hat{s}_{k|k+1} \rfloor$, $s_k = \lceil -z + \hat{s}_{k|k+1} \rceil - 1$

3a. (Increase $s_k$) $s_k = s_k + 1$. If $s_k \leq UB(s_k)$ go to 5a, else to 4a.

4a. (Increase $k$) $k = k + 1$; if $k = m + 1$, terminate algorithm, else go to 3a.

5a. (Decrease $k$) If $k = 1$ go to 6a. Else $k = k - 1$, $\hat{s}_{k|k-1} = \hat{s}_k + \sum_{j=k+1}^{m} \frac{r_{kj}}{r_{kk}} (s_j - \hat{s}_j)$, $r'^2_k = r'^2_{k+1} - r^2_{k+1,k+1} (s_{k+1} - \hat{s}_{k+1|k+2})^2$ and go to 2a.

6a. Solution found. Save $s$ and go to 3a.

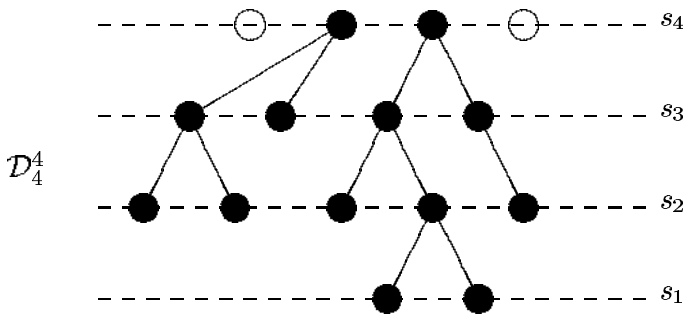


Figure 2.3: *A 4-dimensional example of the closest point search by Fincke-Pohst algorithm*

We can now elaborate on details of the tree pruning interpretation of sphere algorithm that we discussed earlier. Figure 2.3 illustrates the closest point search in a 4-dimensional lattice (i.e., $s$ is a 4-dimensional vector). The algorithm starts with determining an interval for $s_4$ (based on step [2$a$]) and descends down the tree (i.e., looks for $s_3$, $s_2$, $s_1$) as long as there are any points in the next dimension that satisfy condition [3$a$]. Whenever the algorithm reaches a point from which it cannot descend any further, it looks for other points in the same dimension that satisfy [3$a$]; if all the points in the particular dimension are exhausted, the parent node that led to this dimension is terminated and the search continues. If there are no points in 4-dimensional lattice that satisfy (2.1), the search radius is increased and the algorithm is restarted. Figure 2.3 implies that there are two solutions found for the example at hand (which is illustrated by two paths that reached the level $s_1$). In this case, the solution to the closest point search is the point with the smaller metric $\|x - Hs\|^2$ of the two.

### 2.1.2 A First Look at Complexity

The paper [40] gives a complexity analysis of the above algorithm. The main result is that the number of arithmetic operations of both of the previously described

algorithms (excluding Steps 1, 2, 3) is at most

$$\frac{1}{6}(2m^3+3m^2-5m)+\frac{1}{2}\left((2\lfloor\sqrt{r^2d}\rfloor+1)\begin{pmatrix}\lfloor 4r^2d\rfloor+m-1\\ \lfloor 4r^2d\rfloor\end{pmatrix}+1\right), \qquad (2.10)$$

where $d=\max(r_{11}^{-2},\ldots,r_{mm}^{-2})$. In practice $d$ grows proportional to $n$ ($r_{11}^2$, for example, is simply the squared norm of the first column of $H$, which has $n$ entries) and $r^2$ grows proportional to $m$ (for more on this see below) and so the upper bound on the number of computations in (2.10) can be quite large. Our experience with numerical implementations of the algorithm shows that the bound is quite loose. Moreover, although it does depend on the lattice-generating matrix $H$ (through the quantity $d$), it offers little insight into the complexity of the algorithm. We will therefore not further consider it.

In this chapter we study the complexity of sphere decoding algorithm using the geometric interpretation we have developed so far. As mentioned earlier, the complexity of sphere decoding algorithm depends on the size of the generated tree in Fig. 2.2, which is equal to the total number of lattice points in hyperspheres of radius $r$ and dimensions $k=1,\ldots,m$. The size of this tree depends on the matrix $H$, as well as on the vector $x$. Therefore, unlike the complexity of solving an unconstrained least-squares problem which only depends on $m$ and $n$ and *not* on the specific $H$ and $x$, complexity of sphere decoding algorithm is *data-dependent*.

## 2.2 Expected Complexity

Of course, since the integer least-squares problem is NP hard, the worst-case complexity of sphere decoding is exponential. However, if we assume that matrix $H$ and vector $x$ are generated randomly (according to some known distributions), then the complexity of the algorithm will itself be a random variable. In this case, it is meaningful to study expected (or average) complexity of sphere decoding, and perhaps even some of its higher order moments.

For a start, assume that there is no known model that relates $x$, $H$, and $s$, i.e., given $H$ and $s$, $x$ can be any point in $\mathcal{R}^n$. In what follows we will give a rough argument for the expected complexity of sphere decoding for this case, although it is not too difficult to make it rigorous. (For a rigorous treatment, albeit using a different approach, see [25].) For an arbitrary point $x$, and an arbitrary lattice $H$, it is not too difficult to show that the expected number of lattice points inside the $k$-dimensional sphere of radius $r$ is proportional to its volume,

$$V_r = \frac{\pi^{k/2}}{\Gamma(k/2+1)} r^k.$$

Therefore, the expected total number of points is

$$\sum_{k=1}^{m} \frac{\pi^{k/2}}{\Gamma(k/2+1)} r^k > \sum_{k=1}^{\frac{m}{2}} \frac{\pi^k}{\Gamma(k+1)} r^{2k} \approx e^{\pi r^2}, \text{ for large } m.$$

To have a nonvanishing probability of finding a point in the $m$-dimensional sphere, its volume must be

$$V_r = \frac{\pi^{m/2}}{(m/2)!} r^m = O(1).$$

But from Stirling's formula this implies that $r^2 = O(m)$ and that the expected complexity of the algorithm is exponential, $e^{O(m)}$.

### 2.2.1 A Random Model

Although not unexpected, the above is a discouraging result. In communications applications, however, the vector $x$ is not arbitrary, but rather is a lattice point perturbed by additive noise with known statistical properties. Thus, we will assume

$$x = Hs + v,$$

where the entries of $v$ are independent $N(0, \sigma^2)$ random variables. We will also assume that the lattice-generating-matrix $H$ is random and is comprised of independent

$N(0,1)$ entries. We further assume, for simplicity, that $m = n$. (The more general case of $m \neq n$ can also be studied without much more effort.)

The first by-product of these assumptions is a method to determine the desired radius $r$. Note that $\|v\|^2 = \|x - Hs\|^2$ is a $\chi^2$ random variable with $m/2$ degrees of freedom. Thus we may choose the radius to be a scaled variance of the noise,

$$r^2 = \alpha m \sigma^2,$$

in such a way that with a high probability $p_{\text{fp}}$ we find a lattice point inside the sphere,

$$\int_0^{\alpha m} \frac{\lambda^{m/2-1}}{\Gamma(m/2)} e^{-\lambda} d\lambda = p_{\text{fp}},$$

where $p_{\text{fp}}$ is set to a value close to 1, say, $p_{\text{fp}} = 0.99$. [If the point is not found, we can increase the probability $p_{\text{fp}}$, adjust the radius, and search again.]

As argued in the previous section of this chapter, the complexity of sphere decoding algorithm is proportional to the number of nodes visited on the tree in Figure 2.2 and, consequently, to the number of points visited in the spheres of radius $r$ and dimensions $k = 1, 2, \ldots, m$. Hence the expected complexity is proportional to the number of points in such spheres that the algorithm visits on average. Thus the expected complexity of sphere decoding algorithm is given by

$$C(m, \sigma^2) = \sum_{k=1}^{m} \underbrace{(\text{expected \# of points in } k\text{-sphere of radius } r)}_{\triangleq E_p(k, r^2 = \alpha m \sigma^2)} \cdot \underbrace{(\text{flops/point})}_{2k+17}. \quad (2.11)$$

The summation in (2.11) goes over the dimensions $k = 1$ through $k = m$. The coefficient $2k + 17$ is the number of elementary operations (additions, subtractions, and multiplications) that Fincke-Pohst algorithm performs per each visited point in dimension $k$.

To calculate (2.11), we need to compute the expected number of points in the sphere, $E_p(k, r^2)$. Note that if the lattice point $s_t$ was transmitted and the vector $x = Hs_t + v$ received, the probability that the lattice point $s_a$ lies in a hypersphere

of radius $r$ around $x$ is (see Appendix A.1)

$$\gamma\left(\frac{r^2}{\sigma^2+\|s_a-s_t\|^2},\frac{k}{2}\right)=\int_0^{\frac{r^2}{\sigma^2+\|s_a-s_t\|^2}}\frac{\lambda^{k/2-1}}{\Gamma(k/2)}e^{-\lambda}d\lambda, \tag{2.12}$$

where $\gamma(p,q)$ denotes an incomplete gamma function of argument $p$ and $q$ degrees of freedom.

Given this probability, one could in principle proceed by finding the argument of the gamma function in (2.12) for each pair of points $(s_a, s_t)$, and sum their contributions; however, even for a finite lattice this would clearly be a computationally formidable task (and not doable at all in the infinite lattice case). Therefore, we shall find it useful to enumerate the lattice, i.e., count the number of points with the same argument of the gamma function in (2.12). Enumeration of infinite and finite lattices is treated separately.

### 2.2.2 Infinite Lattice Case

The probability (2.12) depends only on Euclidean distance between the points $s_t$ and $s_a$, that is, on $\|s_a - s_t\|^2$. However, in an infinite lattice, $s_a - s_t = s$ is yet another lattice point and thus the probability (2.12) depends only on $\|s\|^2$, i.e., on the squared norm of an arbitrary lattice point in the $k$-dimensional lattice. It is thus straightforward to see that

$$E_p(k,r^2)=\sum_{l=0}^{\infty}\gamma\left(\frac{r^2}{\sigma^2+l},\frac{k}{2}\right)\cdot(\#\text{ of lattice points with }\|s\|^2=l).$$

Since $\|s\|^2 = s_1^2 + \ldots + s_k^2$, we basically need to figure out in how many ways a non-negative integer $l$ can be represented as a sum of $k$ squared integers. This is a classic problem in number theory and the solution is denoted by $r_k(l)$ [42]. There exist a plethora of results on how to compute $r_k(l)$. The simplest technique is by means of Jacobi's theta functions (which were originally proposed by Euler):

**Fact 1** *Consider the generating function*

$$\Phi(x) = 1 + \sum_{m=-\infty}^{\infty} x^{m^2} = 1 + 2\sum_{m=1}^{\infty} x^{m^2}. \tag{2.13}$$

*The number of ways a non-negative integer number $l$ can be represented as a sum of $k$ squares, $r_k(l)$, is given by the coefficient of $x^l$ in the expansion of the $k^{th}$ power of the generating function $\Phi(x)$,*

$$\Phi^k(x) = 1 + \sum_{l=1}^{\infty} r_k(l) x^l.$$

■

This can be illustrated by an example: consider the representations of an integer as a sum of $k = 2$ squares. $\Phi^2(x)$ is clearly a series in which each term has an exponent that is obtained as a sum of two squares; since the summation in (2.13) goes over all integer numbers, coefficients in front of each term in the series $\Phi^2(x)$ must be equal to the number of ways that the exponent in that same term can be represented as a sum of two squares.

[As an interesting sidenote, we should mention that the problem of finding $r_k(n)$ was first posed by Waring in 1770 and has attracted significant attention within number theorists community. In fact, Waring posed the problem in a more general form, not limiting representations on sum of squares but allowing powers higher than the second. Since then, many number theorists have tackled the problem but with a limited success. Closed-form solutions for $r_k(l)$ have been found for only several $k$'s. For instance, Jacobi has found analytical expressions for $k \in \{2, 4, 6, 8\}$ using relationships between elliptic and theta functions ([42], chapter 9). Solutions for $k = 10$ and $k = 12$ were found by Liouville and Eisenstein. Later, Ramanujan, Hardy, and Littlewood obtained formulas for even $k \leq 24$. For odd $k$, the only results are due to Dirichlet, who found $r_3(l)$, and Eisenstein, Smith, and Minkowski, who found $r_5(l)$ and $r_7(l)$. However, this exhausts known closed-form analytical solutions. There exist many asymptotic results (asymptotic both in $k$ and $l$). For our purpose, using

the original Euler's idea will suffice. Mathematical software, in particular Wolfram's Mathematica, has a built-in $r_k(l)$ function.]

The above arguments lead to the result on the expected complexity of the Fincke-Pohst algorithm for an infinite lattice which we summarize in the following theorem.

**Theorem 1 [Expected complexity of the Fincke-Pohst algorithm for an infinite lattice]** *Consider the model*

$$x = Hs + v,$$

*where* $v \in \mathcal{R}^{n\times 1}$ *is comprised of i.i.d.* $\mathcal{N}(0,\sigma^2)$ *entries,* $H \in \mathcal{R}^{n\times m}$ *is comprised of i.i.d.* $\mathcal{N}(0,1)$ *entries, and* $s \in \mathcal{Z}^m$ *is an* $m$*-dimensional vector whose entries are integer numbers. Then the expected complexity of the sphere decoding algorithm for the integer least-squares problem,*

$$\min_{s\in\mathcal{Z}^m} ||x - Hs||^2,$$

*is given by*

$$C(m,\sigma^2) = \sum_{k=1}^{m}(2k+17)\sum_{l=0}^{\infty} r_k(l)\gamma\left(\frac{\alpha m\sigma^2}{\sigma^2+l}, \frac{k}{2}\right),$$

*where* $\alpha$ *is such that* $\gamma(\alpha m, m) = 1-\varepsilon$, $\varepsilon \ll 1$.

**Proof:**

The proof follows from the discussion in this chapter.

■

As a measure of complexity, it is often useful to look at the *complexity exponent*, defined as

$$e_c = \frac{\log C(m,\sigma^2)}{\log m}.$$

Considering $e_c$ is visually appealing since complexity exponent approaches a constant if the expected complexity is polynomial, and grows like $\frac{m}{\log m}$ if $C(m,\sigma^2)$ is exponential. The complexity exponent is plotted as a function of $m$ for different values

of $\sigma^2$ in Figure 2.4. As it can be seen from the figure, for small enough noise the expected complexity is polynomial, indicated by the constant $e_c$ over a wide range of $m$. On the other hand, for large noise $e_c$ clearly exhibits the $\frac{m}{\log m}$ behavior and the computational complexity of the algorithm is exponential.

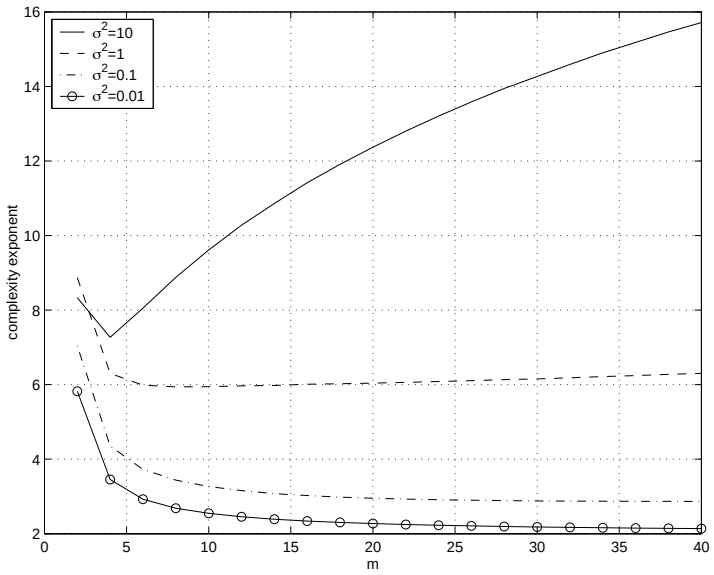


Figure 2.4: *The complexity exponent as a function of* $m$ *for* $\sigma^2 = 0.01, 0.1, 1, 10$.

### 2.2.3 Finite Lattice Case

In communications problems, rather than being unbounded integers, the entries in the unknown $m$-dimensional vector $s$ are often points that belong to an $L$-PAM constellations,

$$\mathcal{D}_L = \left\{ -\frac{L-1}{2}, -\frac{L-3}{2}, \ldots, \frac{L-3}{2}, \frac{L-1}{2} \right\}.$$

We say that the point $s$ then belongs to the lattice $\mathcal{D}_L^m$,

$$\mathcal{D}_L^m = \underbrace{\mathcal{D}_L \times \mathcal{D}_L \times \ldots \times \mathcal{D}_L}_{m\text{-times}},$$

where $\times$-operation denotes the Cartesian product.

Furthermore, in this case, rather than the noise variance $\sigma^2$, one is interested in the signal-to-noise ratio $\rho$,

$$\rho = \frac{m(L^2-1)}{12\sigma^2}.$$

The probability expression (2.12) for finding an arbitrary lattice point $s_a$ inside a sphere around the given point $s_t$, holds for a finite lattice case as well. However, counting lattice points which have the same argument of the gamma incomplete function in (2.12) is not as easy. The reason is that unlike in the infinite lattice case, the difference between two lattice points, $s_a - s_t$, is not necessarily another lattice point. Thus, the lattice enumeration that we used in the previous section needs to be performed over pairs of points, $(s_a, s_t)$.

For this, we propose a modification of the Euler's generating functions technique. In particular, for various finite lattices we will define generating polynomials that, when combined appropriately, perform the counting operations for us.

Let us do the case study for various $L$:

1. $\mathcal{D}_2^k$: The constellation $\mathcal{D}_2^k$ consists of the corners of a $k$-dim hypercube, as illustrated in Figure 2.5. Due to the symmetry, any point in the cube is as

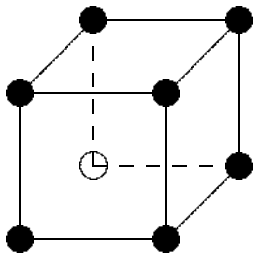

Figure 2.5: *Counting for* $\mathcal{D}_2^k$

likely to be transmitted as any other point. Therefore, we can assume that the "corner" point $s_t = \left[-\frac{1}{2}, -\frac{1}{2}, \ldots, -\frac{1}{2}\right]$ has been transmitted. Then the vector $s_a - s_t$ is comprised of zeros and ones. The number of such vectors whose

squared norm is $l$ is given by $\begin{pmatrix} k \\ l \end{pmatrix}$ and that is the number of points in $\mathcal{D}_2^k$ at distance $l$ from $s_t$.

2. $\mathcal{D}_4^k$: In this case, all points in the constellation are *not* the same. For each entry of $s_t$, we can distinguish between the "corner" and the "center" points, as illustrated in Figure 2.6. Extending the Euler's idea, for the corner points

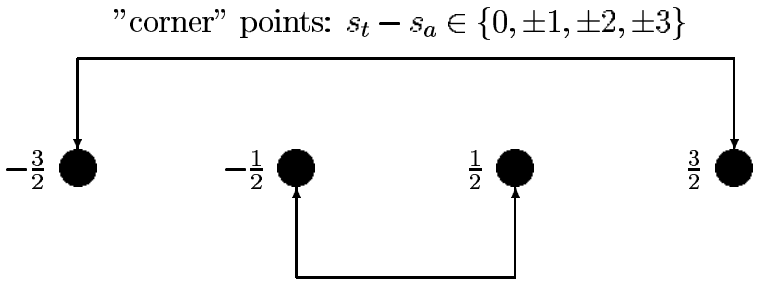


Figure 2.6: *Counting for* $\mathcal{D}_4^k$

we identify the generating polynomial

$$\phi_0(x) = 1 + x + x^4 + x^9,$$

and for the center points the polynomial

$$\phi_1(x) = 1 + 2x + x^4.$$

Essentially, the powers in the polynomials $\phi_0(x)$ and $\phi_1(x)$ contain information about possible squared distances between an arbitrary point $s_a$ and the transmitted point $s_t$. For instance, if an entry in the transmitted vector $s_t$, say $s_{t,1}$, is a corner point, then $s_{a,1} - s_{t,1} \in \{0, \pm 1, \pm 2, \pm 3\}$, depending on $s_{a,1} \in \mathcal{D}_4$. Thus the squared norm of their difference $|s_{a,1} - s_{t,1}|^2$ can be either 0, 1, 4, or 9, as described by powers of $\phi_0(x)$. On the other hand, if $s_{t,1}$, is a center point, then $s_{a,1} - s_{t,1} \in \{0, \pm 1, \mp 1, \pm 2\}$ (which explains coefficient 2 in front of term $x$

in $\phi_1(x)$). Now, if among the $k$ entries of $\mathbf{s}_t$, we choose a corner point $j$ times, the number of ways $\|s_t - s_a\|^2$ can add up to $l$ is given by the coefficient of $x^l$ in the polynomial

$$\begin{pmatrix} k \\ j \end{pmatrix} \phi_0^j(x)\phi_1^{k-j}(x).$$

3. $\mathcal{D}_8^k$: Note that

$$\mathcal{D}_8^k = \{-\frac{7}{2}, -\frac{5}{2}, -\frac{3}{2}, -\frac{1}{2}, \frac{1}{2}, \frac{3}{2}, \frac{5}{2}, \frac{7}{2}\}^k.$$

Let us denote subsets of $\mathcal{D}_8$,

$$\mathcal{S}_1 = \{-\tfrac{7}{2}, \tfrac{7}{2}\} \quad \mathcal{S}_2 = \{-\tfrac{5}{2}, \tfrac{5}{2}\}$$
$$\mathcal{S}_3 = \{-\tfrac{3}{2}, \tfrac{3}{2}\} \quad \mathcal{S}_4 = \{-\tfrac{1}{2}, \tfrac{1}{2}\}$$

Similarly to the $L = 4$ case, we identify the following polynomials for counting $s_a - s_t$ in $\mathcal{D}_8^k$ lattice:

$$\begin{aligned}
\psi_0(x) &= 1 + x + x^4 + x^9 + x^{16} + x^{25} + x^{36} + x^{49}, \\
\psi_1(x) &= 1 + 2x + x^4 + x^9 + x^{16} + x^{25} + x^{36}, \\
\psi_2(x) &= 1 + 2x + 2x^4 + x^9 + x^{16} + x^{25}, \\
\psi_3(x) &= 1 + 2x + 2x^4 + 2x^9 + x^{16}.
\end{aligned}$$

Therefore, if among $k$ entries of $s_t$, we choose $j_i$ points from $\mathcal{S}_i$, $i \in \{1, 2, 3, 4\}$, then the number of ways $\|s_t - s_a\|^2$ can add up to $l$ is given by the coefficient of $x^l$ in the polynomial

$$\begin{pmatrix} k \\ j_1, j_2, j_3, j_4 \end{pmatrix} \psi_0^{j_1}(x)\psi_1^{j_2}(x)\psi_0^{j_3}(x)\psi_1^{j_4}(x),$$

where $j_1 + j_2 + j_3 + j_4 = k$ and $\begin{pmatrix} k \\ j_1, j_2, j_3, j_4 \end{pmatrix} = \frac{k!}{j_1!j_2!j_3!j_4!}$.

Counting for $\mathcal{D}_{16}^k$ and higher order lattices is done similarly to the previously described

cases.

We can now summarize results for the computational complexity of the Fincke-Pohst algorithm for finite lattices in the following theorem:

**Theorem 2 [Expected complexity of the Fincke-Pohst algorithm for a finite lattice]** *Consider the model*

$$x = Hs + v,$$

*where* $v \in \mathcal{R}^{n\times 1}$ *is comprised of i.i.d.* $\mathcal{N}(0,1)$ *entries,* $H \in \mathcal{R}^{n\times m}$ *is comprised of i.i.d.* $\mathcal{N}(0,\rho/m)$ *entries, and* $s \in \mathcal{D}_L^m$ *is an* $m$*-dimensional vector whose entries are elements of an* $L$*-PAM constellation. Then the expected complexity of a sphere decoding algorithm for the integer least-squares problem*

$$\min_{s\in\mathcal{D}_L^m} \|x - Hs\|^2,$$

*for a 2-PAM constellation is*

$$C(m,\rho) = \sum_{k=1}^{m}(2k+17)\sum_{l=0}^{k}\binom{k}{l}\gamma\left(\frac{\alpha m}{1+\frac{12\rho l}{m(L^2-1)}},\frac{k}{2}\right). \tag{2.14}$$

*For a 4-PAM constellation it is*

$$C(m,\rho) = \sum_{k=1}^{m}(2k+17)\sum_{q}\frac{1}{2^k}\sum_{l=0}^{k}\binom{k}{l}g_{kl}(q)\gamma\left(\frac{\alpha m}{1+\frac{12\rho q}{m(L^2-1)}},\frac{k}{2}\right), \tag{2.15}$$

*where* $g_{kl}(q)$ *is the coefficient of* $x^q$ *in the polynomial*

$$(1+x+x^4+x^9)^l(1+2x+x^4)^{k-l}.$$

*Similar expressions can be obtained for 8-PAM, 16-PAM, etc., constellations.*

**Proof:**

The proof follows from the discussion in this chapter.

■

We shall illustrate the complexity calculations with a communications example. Figure 2.7 shows the multiple antenna system with $M$-transmit and $N$-receive antennas. The received signal $\mathbf{x}$ is related to the transmitted symbol $\mathbf{s}$ via

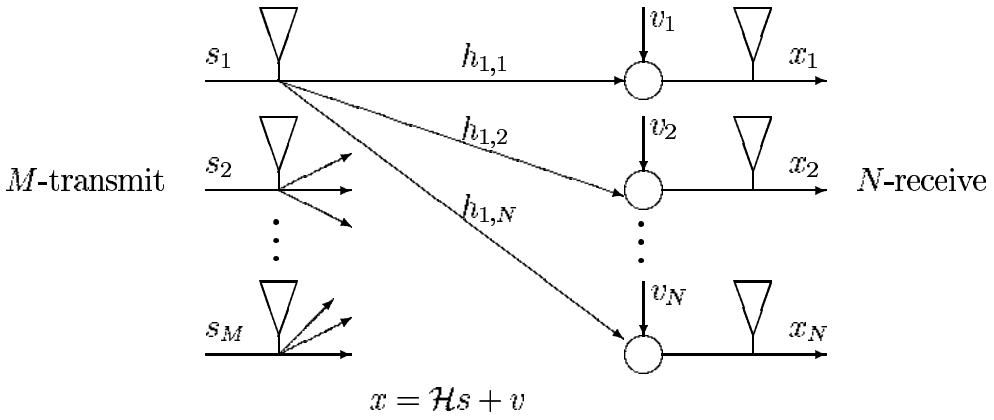


Figure 2.7: *Multiple antenna system*

$$\mathbf{x} = \sqrt{\frac{\rho}{M}}\mathbf{H}\mathbf{s} + \mathbf{v}, \tag{2.16}$$

where $\mathbf{H} \in \mathcal{C}^{N\times M}$ is the known channel matrix, and $\mathbf{v} \in \mathcal{C}^{N\times 1}$ is the additive noise vector, both comprised of independent, identically distributed complex-Gaussian entries $\mathcal{C}(0,1)$. Furthermore, entries in the symbol vector $\mathbf{s}$ are chosen from a complex-valued QAM constellation. If we assume that the entries of $\mathbf{s}$ and $\mathbf{H}$ have, on the average, unit variance, then $\rho$ is the expected received signal-to-noise ratio (SNR) at each receive antenna. To find a real-valued equivalent to the model (2.7), we let $m = 2M$, $n = 2N$, and let $s$ denote an $m$-dimensional real vector obtained from the $M$-dimensional complex vector $\mathbf{s}$,

$$s = \left[\mathcal{R}(\mathbf{s})^T \quad \mathcal{I}(\mathbf{s})^T\right]^T.$$

Furthermore, let $x \in \mathcal{R}^{n\times 1}$ denotes

$$x = \left[\mathcal{R}(\mathbf{x})^T \quad \mathcal{I}(\mathbf{x})^T\right]^T,$$

let $v \in \mathcal{R}^{n\times 1}$ denotes

$$v = \left[\mathcal{R}(\mathbf{v})^T \quad \mathcal{I}(\mathbf{v})^T\right]^T,$$

and let $H \in \mathcal{R}^{n\times m}$ be given by

$$H = \sqrt{\frac{\rho}{M}} \begin{bmatrix} \mathcal{R}(\mathbf{H}) & \mathcal{I}(\mathbf{H}) \\ -\mathcal{I}(\mathbf{H}) & \mathcal{R}(\mathbf{H}) \end{bmatrix}.$$

Then the real-valued equivalent of (2.7) is given by

$$x = Hs + v.$$

We consider the expected complexity of sphere decoding algorithm for signal detection in the system shown in Figure 2.7 for various QAM modulation schemes. The expected complexity $C(\rho, m)$ is a function of both the symbol vector size $m$ and the SNR $\rho$. We shall consider "snapshots" in each dimension, i.e., we keep either $m$ or $\rho$ variable fixed and plot the complexity as a function of the other variable.

Figure 2.8 shows the complexity exponent as a function of $m$ for a fixed SNR $\rho = 20$db and $L$-PAM constellations with $L = 2, 4, 8, 16$. For low rates (i.e., small constellations) the expected complexity is polynomial, whereas for high rates (i.e., large constellations) it is exponential. Simulation results suggest that the complexity is polynomial as long as the rate is sufficiently, but not necessarily all that much, below the Shannon capacity corresponding to the SNR. Since this is the regime at which most communication systems operate, it suggests that ML decoding can be feasible. For instance, the complexity exponents curves in Figure 2.8 that correspond to $L = 8$ and $L = 16$ modulation schemes appear to be in the exponential regime. However, as is illustrated in Figure 2.8 for $m = 10$, the data rates corresponding to the points on those two curves are larger than the corresponding ergodic capacity,

$$C_s = E\left[\log\det\left(I_m + \frac{\rho}{M} HH^T\right)\right].$$

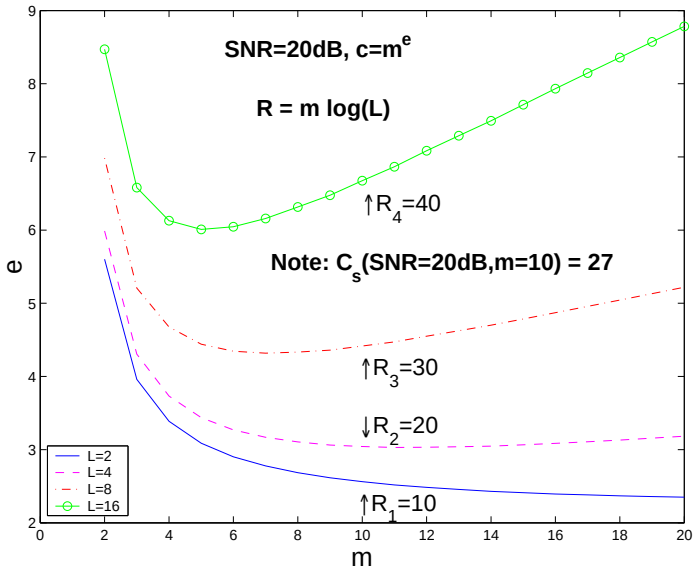


Figure 2.8: *The complexity exponent as a function of* $m$ *for* $\rho = 20db$ *and* $L = 2, 4, 8, 16$.

For instance, when $m = 10$ (and SNR $20dB$), ergodic capacity is $C_{\text{erg}} = 27$. For the same system parameters, only the rates provided by the modulation schemes corresponding to $L = 2$ and $L = 4$ ($R = 10$ and $R = 20$, respectively) can be supported by the channel. The other two modulation schemes cannot be employed (we assume uncoded transmission). Note that expected complexity exponent in the data transmission regime that is supportable by the channel complexity is roughly cubic – which, in fact, is the complexity of the heuristic techniques. Figure 2.9 shows the complexity as a function of SNR for a fixed $m = 10$ (i.e., $M = N = 5$ transmit and receive antennas) and $L$-PAM constellations with $L = 2, 4, 8, 16$. A particular modulation scheme can be used only in the range of SNRs that supports the transmission at the rate corresponding to that modulation scheme. We note that in such a range, complexity exponent is roughly cubic. For instance, although the complexity for $L = 16$ appears to be high over a wide range of SNR, it is only for $\rho > \rho_{40}$ that this modulation scheme can be employed ($\rho_{40}$ is the SNR for which the capacity $C_{\text{erg}} = 40 = R(L = 16)$). The complexity exponent at $\rho_{40}$ and $L =$

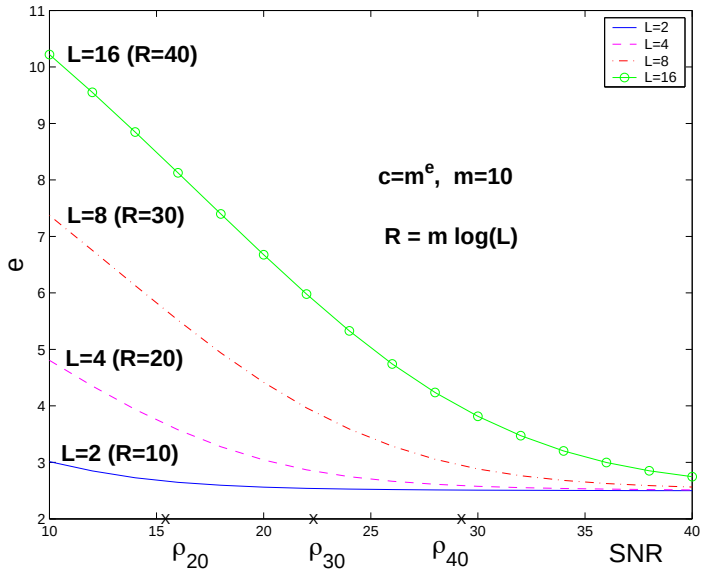


Figure 2.9: *The complexity exponent as a function of* $\rho$ *for* $m = 10$ *and* $L = 2, 4, 8, 16$.

16 is $c_e \approx 4$. The other SNRs marked on Figure 2.9, $\rho_{30}$, and $\rho_{20}$, have similar meanings (only for $L = 8$ and $L = 4$, respectively). Figures 2.8-2.9 show expected complexity, that is, the first-order statistics. In Figure 2.10, empirical distribution of the complexity is shown for $M = N = 5$ transmit and receive antennas, 16-QAM modulation scheme, and for 4 different SNR values. From Figure 2.9, we see that the lowest SNR in Figure 2.10 (16dB) roughly corresponds to the minimum SNR required for transmission on the particular system with the modulation scheme of choice. The outer dashed lines in each graph of Figure 2.10 correspond to three standard deviations of the corresponding distribution. The middle dashed line denotes the mean, i.e., the expected complexity that we previously obtained analytically. We can make the following observations in relation to the distributions as a function of the SNR:

- Expected complexity decreases, which was already anticipated from the results illustrated in Figure 2.9.
- Variance of the complexity decreases, as illustrated with tightening of the standard deviation.

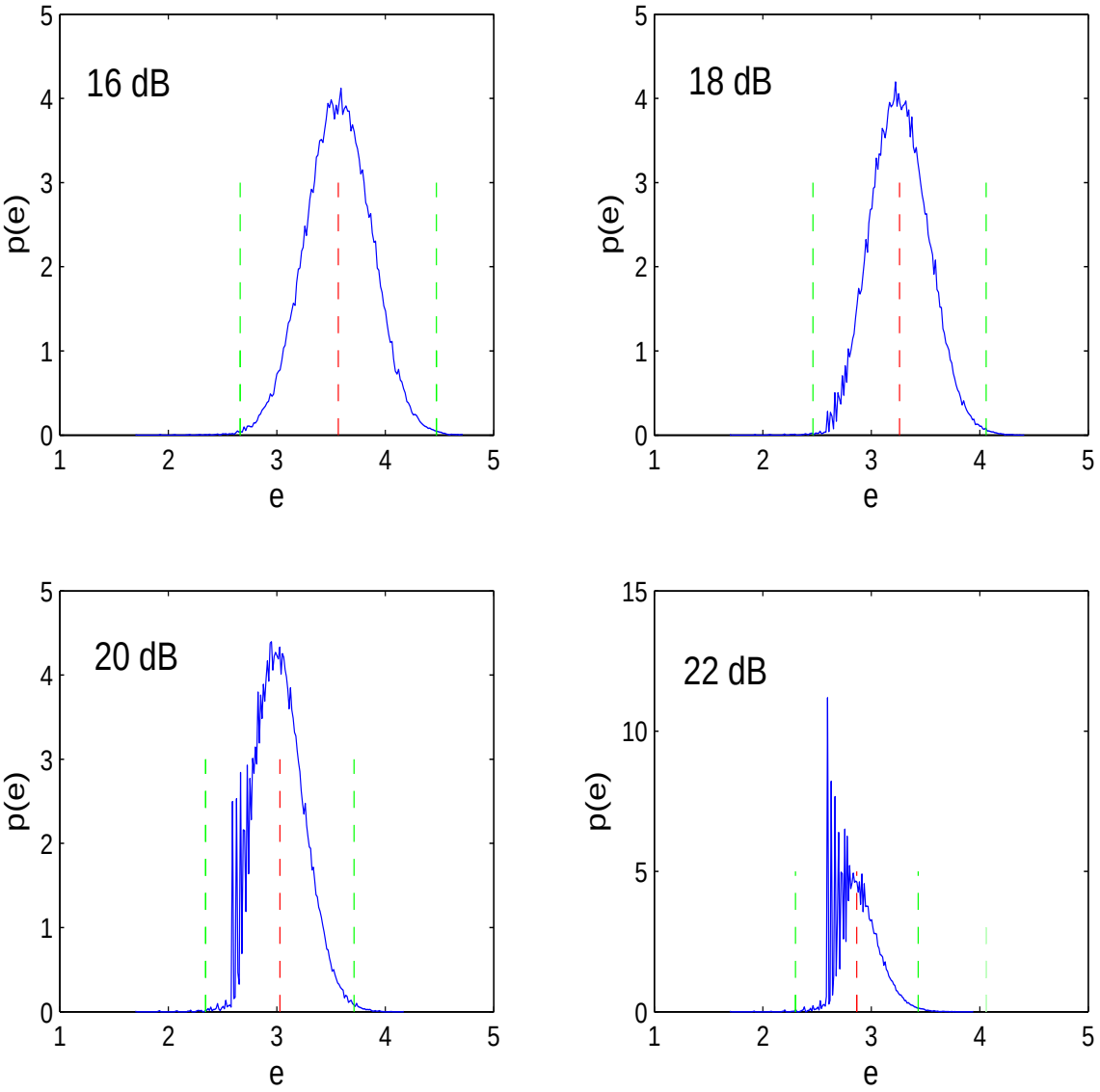


Figure 2.10: *The complexity exponent distribution for* $M = N = 5$, $L = 4$, *and* $SNR = 16, 18, 20, 22dB$.

- As SNR increases, some "point-mass" like segments occur in the distribution. This is expected: for large SNRs, the radius of the sphere will be small and only a small (discrete) number of lattice points are found inside.

For a comparison, exhaustive search in $M = N = 5$, 16-QAM system requires examining $k = 4^{10} \approx 10^6$ points, which is of the $O(m^{\log_{10} 10^6}) = O(m^6)$ order.

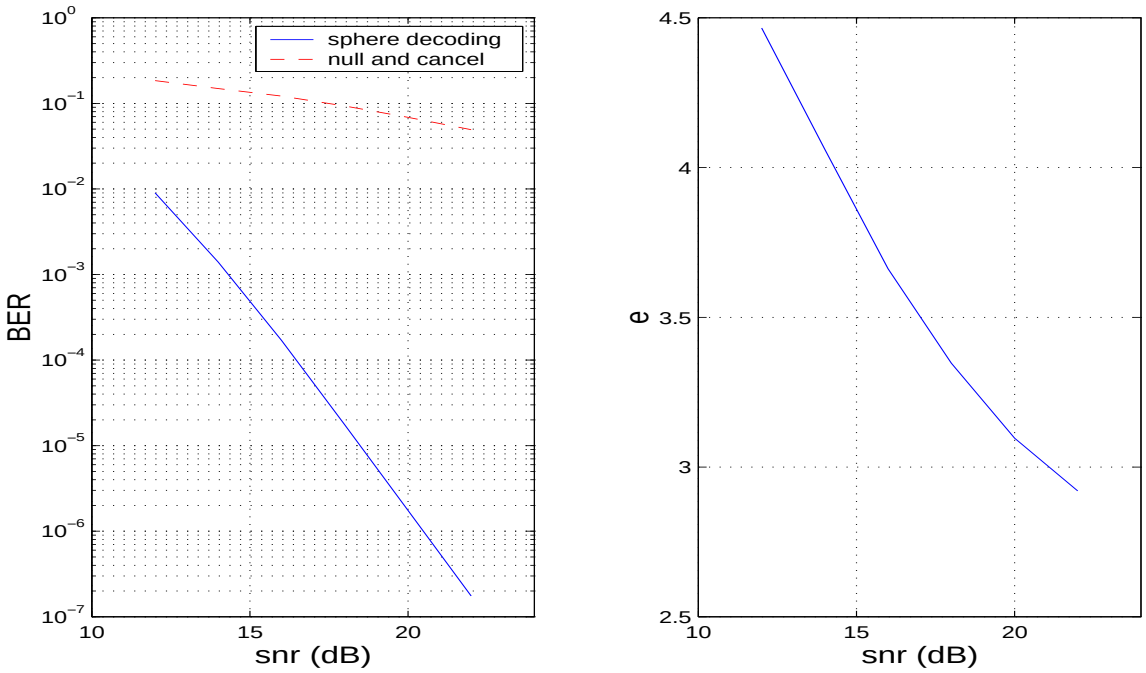


Figure 2.11: *Sphere decoder vs. nulling and cancelling, $M = N = 5$, $L = 4$.*

Figure 2.11 shows the improvement in performance of sphere decoding over nulling and cancelling for a multi-antenna system employing $M = N = 5$ transmit and receive antennas and 16-QAM modulation scheme. The complexity of ML decoding via sphere decoding here is comparable to that of nulling and cancelling, whereas the performance improvement is significant. The range of signal-to-noise ratios in Figure 2.11 is typical for indoor applications ([13]).

Figure 2.12 compares the performance of sphere decoding and nulling and cancelling for a multi-antenna system with $M = N = 2$ antennas, employing 16-QAM.

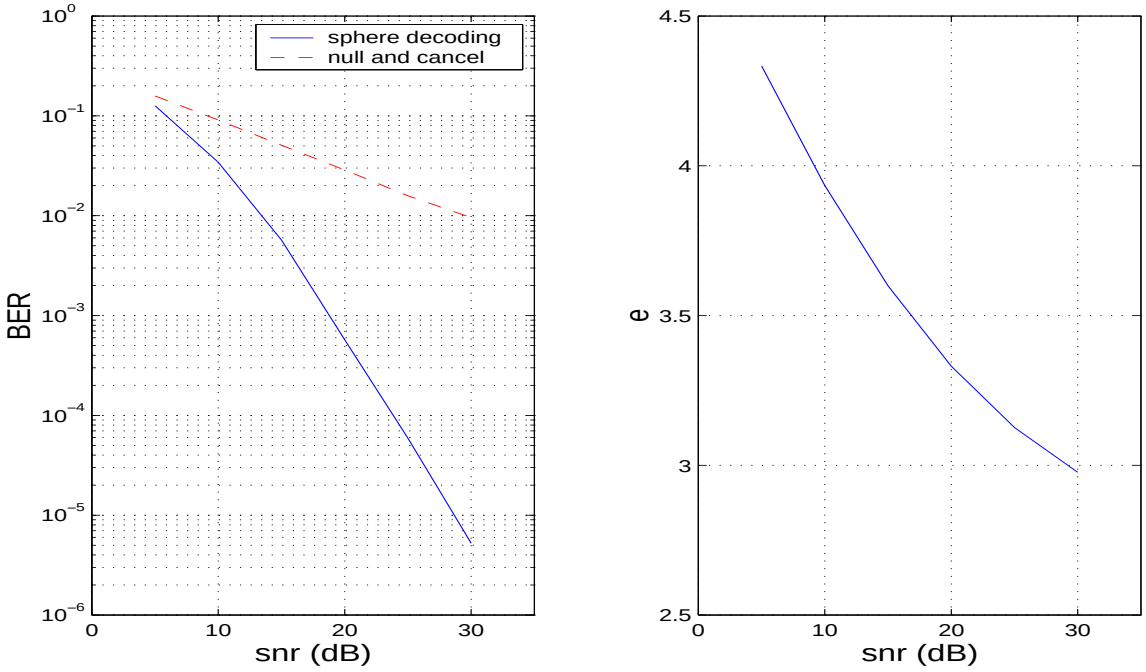


Figure 2.12: *Sphere decoder vs. nulling and cancelling,* $M = N = 2$, $L = 4$.

The expected complexity exponent is also shown in Figure 2.12. In the SNR range of interest, expected complexity of sphere decoding is again comparable to that of nulling and cancelling.

Comparing Figure 2.11 and Figure 2.12 we note that the performance gap between ML detection (provided by sphere decoding) and nulling and cancelling increases as the system employs more antennas. The analytical expression for the average probability of error appears difficult to derive. As an alternative, a pairwise error probability, as in [43], may be considered instead. The pairwise error probability that the vector $\mathbf{s}^{(j)}$ was detected while $\mathbf{s}^{(i)}$ was transmitted can be upper bounded at high SNRs as [43]

$$P(\mathbf{s}^{(i)} \rightarrow \mathbf{s}^{(j)}) \leq \frac{1}{\left(\frac{\rho}{4M}\right)^{N} ||\mathbf{d}_{i,j}||_F^2}, \tag{2.17}$$

where $\mathbf{d}_{i,j} = \mathbf{s}^{(i)} - \mathbf{s}^{(j)}$. The bound (2.17) indicates that BER decreases exponentially with *receive diversity* – the number of receive antennas. ML decoder, as evident

from Figure 2.11, fully exploits the receive diversity – the slope of the BER curve implies improvement by $N = 4$ orders of magnitude per SNR decade. On the other hand, nulling and cancelling (assuming no error propagation) converts the channel into a set of parallel channels with increasing diversity [11]. However, due to the error propagation, the performance is dominated by the first stream decoded by the receiver. Thus to improve the performance of nulling and cancelling, the decoding is often ordered according to the signal-to-interference-and-noise ratio of the incoming data streams.

Finally, Figure 2.13 illustrates the symbol error rate performance comparison of the sphere decoding and nulling and cancelling for the system employing $M = 8$ transmit and $N = 12$ receive antennas and 16-QAM modulation, the system specifications of V-BLAST [13]. Corresponding expected complexity is sub-cubic over entire SNR range of interest.

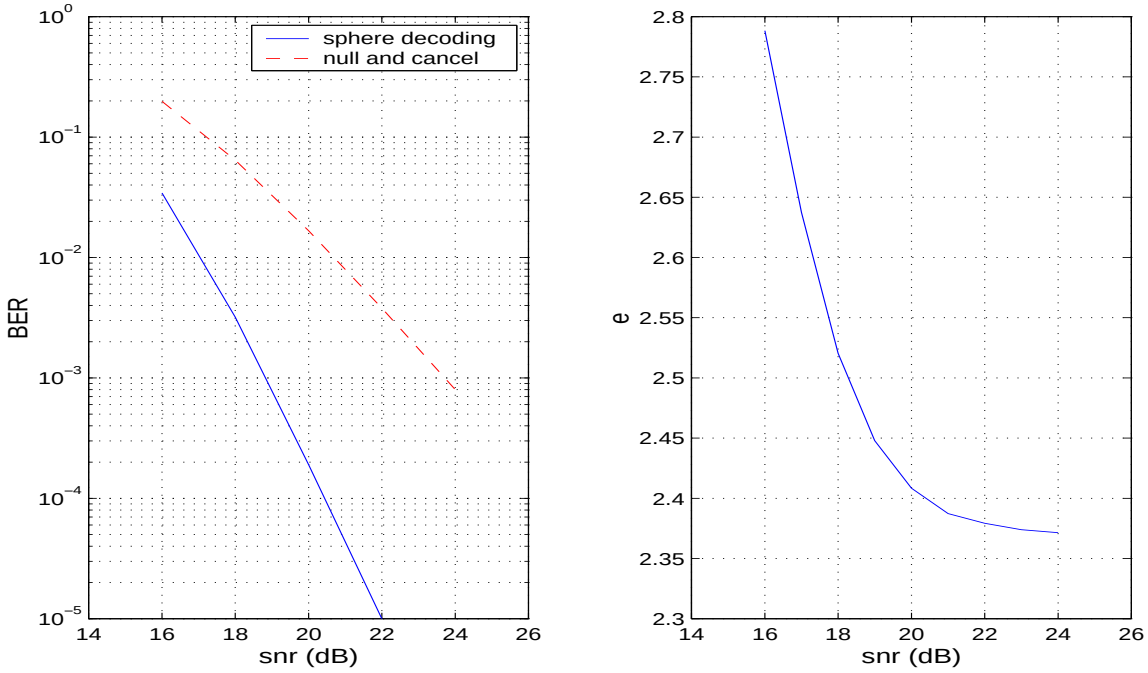


Figure 2.13: *Sphere decoder vs. nulling and cancelling,* $M = 8$, $N = 12$, *16-QAM.*

## 2.3 Some Remarks

The expected complexity that we discussed in this section accounts for finding all the lattice points in the sphere. The point among the found ones with the smallest metric (1.10) is the solution to maximum-likelihood problem. There are some variations on the basic sphere decoding algorithm which we briefly mention here:

- *Sphere decoding with radius update.*

  Every time the point $\mathbf{s}_i$ in the sphere is found, we set the new radius of the sphere $r^2 = ||x - H\mathbf{s}||^2$ and restart the algorithm. The radius update may be particularly useful at lower SNRs, where the number of points in the initial sphere is relatively large.

- *Schnorr-Euchner version of the sphere decoding.*

  This strategy was proposed in [45]. The likelihood that the point will be find early is maximized if the search at each dimension $k$ is performed in the order

  $$[\hat{s}_d], [\hat{s}_d] - 1, [\hat{s}_d] + 1, [\hat{s}_d] - 2, \ldots$$

  The expected complexity of the Schnorr-Euchner version of sphere decoding algorithm is no greater than the expected complexity of the basic algorithm that we derived in this chapter.

## 2.4 Conclusions

In this chapter, we considered the Fincke-Pohst (sphere decoding) algorithm for solving integer least-squares problems and its computational complexity. Geometrically, solving an integer least-squares problem is equivalent to searching for the closest point in a lattice. The sphere decoding algorithm performs such search only within a hypersphere centered at a given point. Based on the proposed tree-search geometric interpretation of the algorithm, we calculated its expected complexity, averaged

over the noise and the lattice. We obtained closed-form expressions for the expected complexity in terms of the noise variance and the dimension of the lattice for both the lattices with infinite span in each dimension as well as for the finite lattices (i.e., constellations). The expected complexity of the sphere decoding for finite lattices is, not surprisingly, also a function of the constellation size.

In many communication scenarios, maximum-likelihood detection reduces to solving an integer least-squares problem, i.e., to finding a point in a multidimensional finite lattice which is closest to the point observed at a receiver. In such applications, ML detection is rarely performed, on the grounds that it requires exponential complexity and is, therefore, computationally intractable.

It turns out that over a wide range of noise variances and dimensions, the expected complexity is often cubic or sub-cubic. Since many communications systems operate at noise levels for which this is the case, this suggests that maximum-likelihood detection, which was hitherto thought to be computationally intractable, can in fact be implemented with complexity similar to heuristic methods, but with significant performance gains – a result with many practical implications.

# Chapter 3

# Sphere Decoding for Channels with Memory

The Fincke-Pohst algorithm that we considered in Chapter 2 assumes no special structure of the channel matrix $H$ and requires computing its $QR$ factorization. In this chapter, we demonstrate how the sphere decoding idea can be employed for detection on frequency-selective channels directly, without performing $QR$ factorization of the equivalent (banded Toeplitz) channel matrix. We also consider the expected complexity of the algorithm for this special case of the lattice generator matrix.

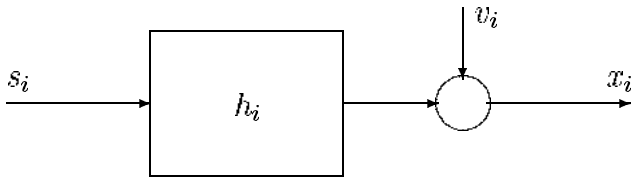


Figure 3.1: *Frequency-selective channel*

Consider the frequency-selective channel model in Figure 3.1, with the input/output relation given by

$$x_i = \sum_{j=1}^{l} h_j s_{i-j} + v_i,$$

where $h_i, i = 1, \ldots, l$ are the coefficients of the channel impulse response, $s_i$ is $i^{th}$ symbol in a transmitted sequence (chosen from an $L$-PAM constellation), and $v_i$ denotes Gaussian noise sample $\mathcal{N}(0,1)$. To employ the Fincke-Pohst algorithm, we write the channel model as

$$x = Hs + v, \tag{3.1}$$

where

$$s = [s_1 \; s_2 \; \ldots \; s_T]' \in \mathcal{D}_L^T$$

is the vector of the transmitted data sequence, and

$$v = [v_1 \; v_2 \; \ldots \; v_{T+l-1}]' \in \mathcal{R}^{(T+l-1)\times 1}$$

is the vector of an additive white Gaussian noise. The matrix $H \in \mathcal{R}^{(T+l-1)\times T}$ is given by

$$H = \begin{bmatrix} h_1 & & & & & & \\ h_2 & h_1 & & & & & \\ \vdots & \vdots & \ddots & & & & \\ h_l & \ldots & & h_1 & & & \\ & \ddots & & & \ddots & & \\ & & h_l & \ldots & & h_1 & \\ & & & \ddots & \vdots & \vdots & \\ & & & & h_l & h_{l-1} & \\ & & & & & h_l & \end{bmatrix}, \; h_i \in \mathcal{R}.$$

Here is where the banded Toeplitz structure of $H$ comes in handy. The RHS of (2.1) can be expanded as

$$r^2 \geq (x_{T+l-1} - h_l s_T)^2 + (x_{T+l-2} - h_l s_{T-1} - h_{l-1} s_T)^2 + \ldots \tag{3.2}$$

where the first term depends only on $s_T$, the second term on $\{s_T, s_{T-1}\}$ and so on. Therefore, considering the first term only, a necessary condition for $Hs$ to lie inside

the hypersphere is that

$$r^2 \geq (x_{T+l-1} - h_l s_T)^2.$$

This condition is equivalent to $s_T$ belonging to the interval

$$\left\lceil \frac{-r + x_{T+l-1}}{h_l} \right\rceil \leq s_T \leq \left\lfloor \frac{r + x_{T+l-1}}{h_l} \right\rfloor. \tag{3.3}$$

Of course, (3.3) is by no means sufficient. For every $s_T$ satisfying (3.3), defining

$$r_{T-1}^2 = r^2 - (x_{T+l-1} - h_l s_T)^2,$$

and

$$x_{T-1|T} = x_{T+l-1} - h_{l-1} s_T,$$

a stronger necessary condition can be found by looking at the first two terms in (3.2), which leads to $s_{T-1}$ belonging to the interval

$$\left\lceil \frac{-r_{T-1} + x_{T-1|T}}{h_l} \right\rceil \leq s_{T-1} \leq \left\lfloor \frac{r_{T-1} + x_{T-1|T}}{h_l} \right\rfloor. \tag{3.4}$$

One can continue in a similar fashion for $s_{T-2}$, and so on until $s_1$. However, these $T$ conditions used to find $s$ are necessary but still not sufficient. Only if an additional constraint,

$$r_1^2 \geq (x_{l-1} - h_{l-1} s_1 - \ldots - h_1 s_{l-1})^2 + \ldots + (x_1 - h_1 s_1)^2, \tag{3.5}$$

is satisfied, the point $s$ indeed does belong to the sphere, i.e., it satisfies condition (2.1). [Remark: one can immediately notice a potential drawback to the aforementioned algorithm. Additional constraint (3.5) means that the $T$ previously considered constraints might have not been particularly stringent. This would clearly have negative impact on the complexity. Indeed, as we shall argue shortly, we observe that there are scenarios where performing QR factorization and then employing sphere decoding as in Chapter 2 may, in fact, be more favorable approach. In general, one

should consult the calculated complexity expressions.]

## 3.1 The Algorithm for Channels with Memory

We can summarize the algorithm as follows:

*Input:* $H$, $x$, $r$.

(1b) Set $k = T$, $r_T^2 = r^2$, $x_{T|T+1} = x_{T+l-1}$

(2b) (Bounds for $s_k$) Set $UB(s_k) = \lfloor \frac{r_k + x_{k|k+1}}{h_l} \rfloor$, $s_k = \lceil \frac{-r_k + x_{k|k+1}}{h_l} \rceil - 1$

(3b) (Increase $s_k$) $s_k = s_k + 1$. If $s_k \leq UB(s_k)$ go to (5b), else to (4b).

(4b) (Increase $k$) $k = k + 1$ and go to (3b).

(5b) (Decrease $k$) If $k = 1$ go to (6b).
Else $k = k-1$, $x_{k|k-1} = x_{k+l-1} + \sum_{j=k+1}^{\min(k+l,T)} h_{l+k-j} s_j$, $r_k^2 = r_{k+1}^2 - (x_{k+l} - h_l s_{k+1})^2$.

(6b) If $r_1^2 \geq (x_{l-1} - h_{l-1}s_1 - \ldots - h_1 s_{l-1})^2 + \ldots + (x_1 - h_1 s_1)^2$, solution found. Save $s$ and go to (3b).

Just as we reasoned in Chapter 2 in the context of multi-antenna systems, vector $x$ in (3.1) is not arbitrary, but is a lattice point $Hs$ perturbed by an additive noise with known statistical properties. Thus, we can talk about average complexity of the sphere decoding algorithm. The expected complexity of the algorithm is proportional to the expected number of lattice points that the algorithm visits. To calculate it, we need a probability that an arbitrary lattice point $s_a$ belongs to a $k$-dimensional sphere of radius $r$ around the transmitted point $s_t$.

Suppose that the lattice point $s_t$ was transmitted and that the vector $x = Hs_t + v$ was observed. Then an arbitrary point $Hs_a$ belongs to the hypersphere of radius $r$ around $x$ if

$$\eta = r^2 - \|x - Hs_a\|^2 = r^2 - \|v + H(s_t - s_a)\|^2 > 0.$$

Therefore, probability that the point $Hs_a$ belongs to the hypersphere of radius $r$ around $x$ is

$$P(\eta > 0) = \int_0^\infty p(\eta) = \int_0^\infty \left[\frac{1}{2\pi}\int_{-\infty}^{\infty} \Phi(\omega)e^{-j\omega\eta}d\omega\right] d\eta, \tag{3.6}$$

where

$$\Phi(\omega) = Ee^{j\omega\eta}$$

is the characteristic function of $\eta$. It can be shown (see Appendix A.2) that

$$\Phi(\omega) = \frac{e^{j\omega r^2}}{(1+j\omega\sigma^2)^{T-l}\prod_{k=1}^{l}[1+j\omega(\sigma^2+\rho_k)]}, \tag{3.7}$$

where $\rho_k, k = 1, \ldots, l$ are the eigenvalues of the matrix

$$\Lambda^*\Lambda = (\mathcal{S}_a - \mathcal{S}_t)^*(\mathcal{S}_a - \mathcal{S}_t),$$

where

$$\mathcal{S} = \begin{bmatrix} s_l & \cdots & & s_1 \\ \vdots & \vdots & \ddots & \vdots \\ s_T & \cdots & & s_{T-l+1} \\ & s_T & & \vdots \\ & & \ddots & s_{T-1} \\ & & & s_T \end{bmatrix}_{T\times l}$$

By taking an inverse Fourier transform of the characteristic function, one can show that the probability in (3.6) is a linear combination of a number of incomplete gamma functions; the eigenvalues of $(\Lambda^*\Lambda)$ determine the arguments of the incomplete gamma functions whereas the degrees of freedom of the gamma functions depend on the channel and the data block length ($l$ and $T$). The general closed form expression for this probability appears hard to obtain, as the number of constituent incomplete gamma functions varies with $l$ and $T$. However, for the calculation purposes, one can use numerical techniques in, e.g., *Mathematica* or *MATLAB*. Summing the probabilities

in (3.6) over all possible eigenvalues of $(\Lambda^*\Lambda)$ yields the expected complexity. [Note that the calculation of the complexity via exhaustive summation is feasible only for small to moderate sizes of $l$ and $T$.]

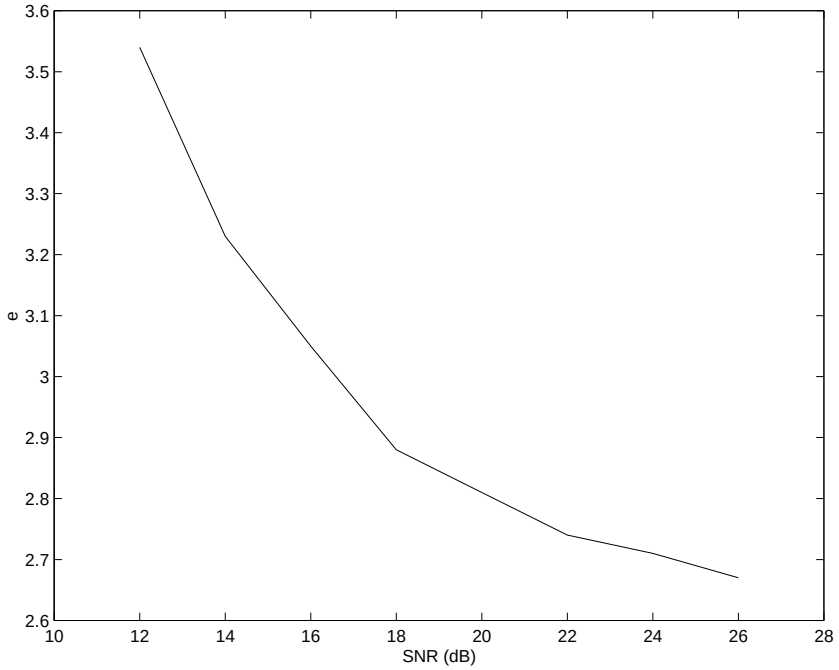


Figure 3.2: *Expected complexity exponent of SD,* $T = 20$, $l = 12$, $L = 2$.

For illustration, consider an example with the channel that has $l = 12$ taps, the data block of the length $T = 20$, and 2-PAM modulation scheme ($L = 2$). In Figure 3.2, we plot the (empirically calculated) complexity exponent $e$ as a function of SNR. Note that the complexity exponent is sub-cubic over a wide range of SNRs. Therefore, the algorithm is less complex than the standard sphere decoding (which requires $QR$ factorization).

We should repeat, however, that one can find sets of system parameters (i.e., $l$, $T$, $L$) for which it is better to first perform $QR$ factorization. This is due to the fact that the sphere decoding on $H$ imposes less strict conditions than the sphere decoding on $R$ – as evident by the need to pose the additional condition (6*b*) when doing the former.

A system for which this is the case is our next example: consider a channel of length $l = 8$, the data block with $T = 16$ symbols, and a 4-PAM modulation scheme ($L = 4$). Here the sphere decoding is employed on the matrix $R$ in the $QR$ factorization of $H$, rather than $H$ directly, as it is more favorable that way. The performance comparison with generalized decision-feedback equalization is given on the left plot in Figure 3.3. Note that in the range of SNRs for which the system performance is good (i.e., for which the bit-error-rate is small, say, $< 10^{-3}$), the expected complexity of the algorithm is roughly cubic and sub-cubic. In fact, for a wide range of $T$, $l$, $L$, and $\rho$,

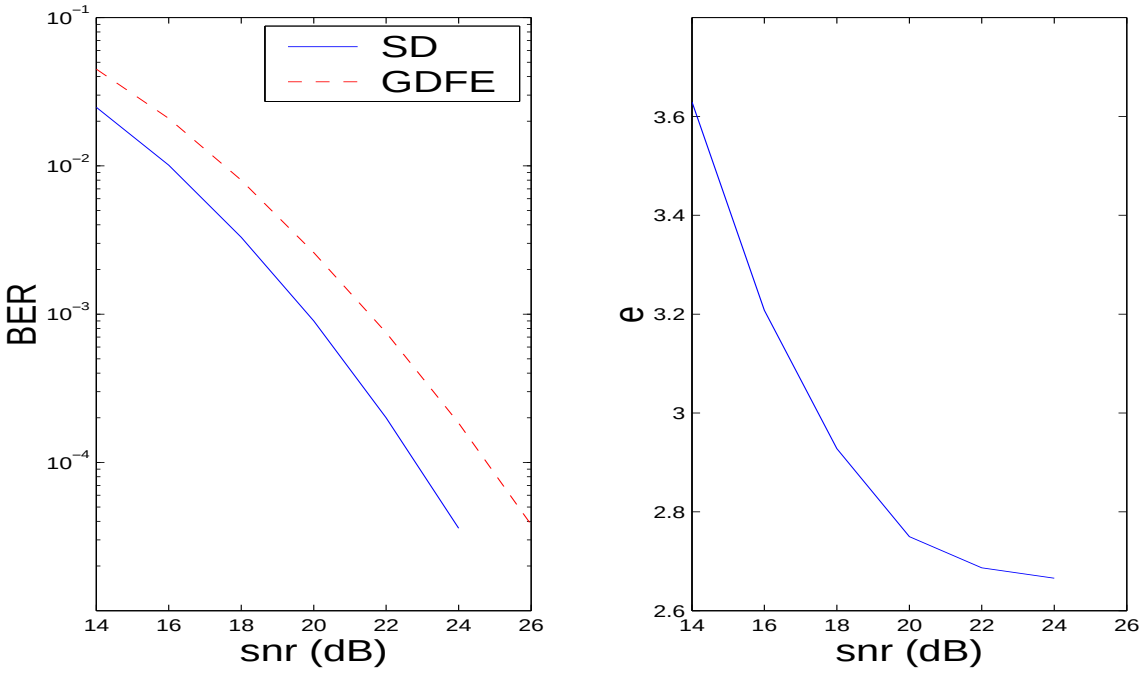


Figure 3.3: *Performance of SD compared to generalized DFE, $T = 16$, $l = 8$, $L = 4$.*

sphere decoding algorithm for frequency-selective channels has expected complexity comparable to cubic-time methods such as nulling and cancelling (cubic in $T$, the data block size). As a comparison, the Viterbi algorithm, which has the same performance as the sphere decoding, has complexity which is exponential in the channel length (i.e., $L^l$) and linear in the block length $T$. Therefore, for the example in Figure 3.3, the complexity of the Viterbi algorithm is $TL^l \sim 10^6$ flops. On the other hand, the

sphere decoding algorithm solves the same maximum-likelihood detection problem with $T^e \sim 4 \times 10^3$ flops on average, which is a considerable computational saving. The sphere decoding offers computational savings over the Viterbi algorithm for this particular set of parameters and, in general, for the cases where the length of the channel is large. However, for short channels, low modulation schemes, and very long block lengths, the Viterbi algorithm will have lower (essentially linear in the data block length) complexity than the sphere decoding.

## 3.2 Combining Sphere Decoding and Viterbi Algorithm

For channels with large memory, sphere decoding provides significant savings with respect to the Viterbi algorithm. On short channels, however, the Viterbi algorithm is often computationally superior. In practical schemes, a hybrid receiver structure that would combine the sphere decoding depth-first search strategy with the dynamic programming principles of the Viterbi algorithm, might be desired.

The Viterbi algorithm allows for exact ML sequence detection (MLSD). The Viterbi algorithm is commonly defined on a trellis, a directed graph that describes systems with memory. An example of the trellis is shown in Figure 3.4. It represents, for instance, a frequency selective channel of length $l = 3$ and a 2-PAM modulation scheme used for transmitting data on it.

The vectors of vertical black dots in the trellis in Figure 3.4 denote the state sets $\mathcal{S}$. These vectors are labeled by $k = 1, \ldots, T$, and arranged into an array of length $T$. The size of the state set $\mathcal{S}_k$ depends on the channel memory. In particular, each element of the set $\mathcal{S}_k$ represent one possible state of the channel memory. For instance, there are $L^{l-1} = 4$ states in the trellis in Figure 3.4, each one representing a possible content of the channel memory (00, 01, 10, and 11).

Adjacent state sets in the trellis are connected via branches. The branches are typically labeled to describe the input/output relation of the system corresponding

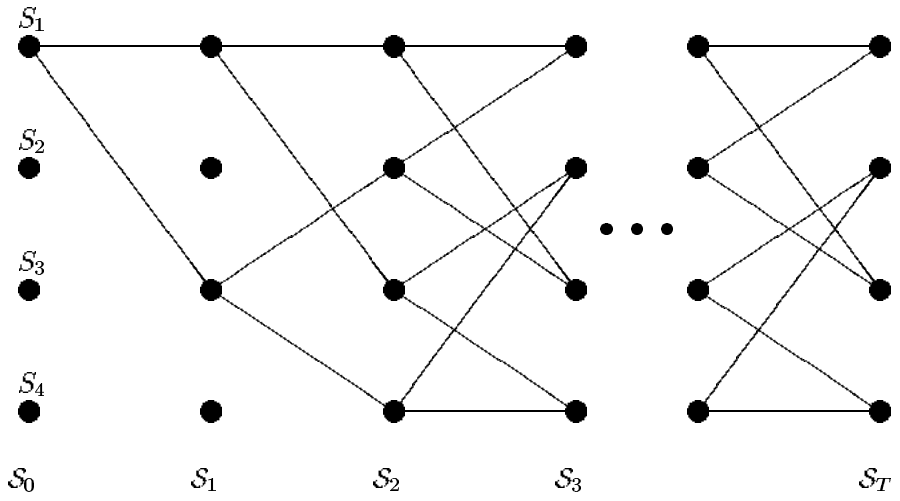


Figure 3.4: *Trellis example*

to the particular state transition. There is a total of $L$ branches emanating from each state, corresponding to $L$ possible values of the input. [Note that the trellis in Figure 3.4 starts from the all zeros state.] The state that a branch sinks in to depends upon the source state and the value of the current input bit. A concatenated sequence of trellis branches is called a *path*. The length of the path is determined by the number of branches in it. Each path corresponds to a distinct sequence of input symbols, indicated by the labels on the branches that comprise the path.

The Viterbi algorithm uses the trellis representation to find the solution to

$$\min_{\mathbf{s}} \|x - Hs\|^2.$$

In particular, it finds the trellis path corresponding to the smallest $\|x-Hs\|^2$ without actually performing an exhaustive search over the entire trellis. To this end, we describe the argument of the ML optimization problem by

$$\mathcal{C}_{k+1} = \sum_{j=1}^{k+1} |x_j - \sum_{m=1}^{l} h_m s_{j-m+1}|^2. \tag{3.8}$$

The value of $\mathcal{C}_{k+1}$ depends on the current state and the trellis path that led to that

particular state. To find the ML optimal sequence, one needs to determine the trellis path with the smallest cost. An exhaustive search over all possible paths would clearly not be feasible even for trellises with moderate number of states. However, $\mathcal{C}_{k+1}$ can be expressed as

$$\mathcal{C}_{k+1} = \mathcal{C}_k + |x_{k+1} - \sum_{j=1}^{l} h_j s_{k-j+1}|^2. \tag{3.9}$$

Clearly, the second term of the RHS of (3.9) does not depend on $s_{k-l}, \ldots, s_0$. Therefore, to find the smallest cost path to the state $S_j$ in the set $\mathcal{S}_{k+1}$, it is sufficient to consider all possible state transitions to $S_j$ (along the $L$ branches emanating from the states in set $\mathcal{S}_k$). This procedure can be done recursively. Finally, the trellis path of length $T$ that has the smallest cost $\mathcal{C}_T$ is the optimal path. The signal sequence that corresponds to the branch transitions along the optimal trellis path is the solution to the ML detection problem.

The complexity of the Viterbi algorithm is proportional to the number of states and thus grows exponentially with the length of the channel. On the other hand, it is linear in the length of the data sequence.

The Viterbi algorithm efficiently solves ML detection problem on trellises with a moderate number of states. However, for long channels and/or modulations with high cardinality constellations, the Viterbi algorithm is often inefficient and occasionally non-feasible. On the other hand, the sphere decoding algorithm has expected complexity often significantly below the complexity of the Viterbi algorithm. However, the worst-case complexity of the sphere decoding algorithm is exponential and corresponds to the exhaustive search. Therefore, a hybrid receiver structure that combines the sphere decoding constrained search strategy with the trellis based decoding of the Viterbi algorithm, is desired. This can be obtained by either modifying the sphere decoding algorithm to impose the channel memory state constraints or imposing a sphere-constrained search onto the Viterbi algorithm.

Before proceeding to explain the hybrid schemes, it will be useful to revisit (3.2) and re-examine it. In particular, we will find it useful to state the conditions on the components of $s$ in a reverse order.

We write (3.2) as

$$r^2 \geq (x_1 - h_1 s_1)^2 + (x_2 - h_1 s_2 - h_2 s_1)^2 + \dots \tag{3.10}$$

where the first term depends only on $s_1$, the second term on $\{s_1, s_2\}$ and so on. Therefore, considering the first term only, a necessary condition for $Hs$ to lie inside the hypersphere is

$$r^2 \geq (x_1 - h_1 s_1)^2. \tag{3.11}$$

This condition is equivalent to $s_1$ belonging to the interval

$$\left\lceil \frac{-r + x_1}{h_1} \right\rceil \leq s_1 \leq \left\lfloor \frac{r + x_1}{h_1} \right\rfloor. \tag{3.12}$$

Furthermore, for every $s_1$ satisfying (3.12), $s_2$ needs to satisfy

$$r^2 \geq (x_1 - h_1 s_1)^2 + (x_2 - h_1 s_2 - h_2 s_1)^2 \tag{3.13}$$

Defining

$$r_2^2 = r^2 - (x_1 - h_1 s_1)^2, \tag{3.14}$$

and $x_{2|1} = x_2 - h_2 s_1$, a stronger necessary condition can be found by looking at the first two terms in (3.10), which leads to $s_2$ belonging to the interval

$$\left\lceil \frac{-r_2 + x_{2|1}}{h_1} \right\rceil \leq s_2 \leq \left\lfloor \frac{r_2 + x_{2|1}}{h_1} \right\rfloor.$$

One can continue in a similar fashion for $s_3$, and so on until $s_T$. Note that these $T$ conditions used to find $s$ are necessary but still not sufficient. Only if an additional constraint,

$$r_{T+1}^2 \geq (x_{T+1} - h_l s_{T-l+2} - \dots - h_2 s_T)^2 + \dots + (x_{T+l-1} - h_l s_T)^2,$$

is satisfied, the point $s$ indeed does belong to the sphere, i.e., it satisfies condition (2.1).

Now, recall the tree search interpretation of the sphere decoding algorithm illustrated in Figure 2.2. The sphere decoding algorithm does not account for the special structure (banded Toeplitz) of the lattice generating matrix in (3.1). We propose the following modification to fix that: Assume that the algorithm is currently examining a point on the $k^{th}$ level of the tree. Based on the current, and up to $l-2$ points on levels $k-1, k-2, \ldots$, we identify the corresponding state $S_j$, $j = 1, 2, \ldots, L^{l-1}$ (where the state is defined as on the trellis). Furthermore, from (3.14), it is easy to see (by writing out the recursion for any $r_k$) that the cost associated with this state is given by

$$\mathcal{C}_k(S_j) = r^2 - r_{k+1}^2.$$

Now, in addition to the standard steps that the sphere decoding algorithm undertakes, it also compares this $\mathcal{C}_k(S_j)$ with the previously stored $\mathcal{C}_k(S_j)$ and, if the current one is greater, prunes the tree (i.e., discard the current tree node). If the current $\mathcal{C}_k(S_j)$ is smaller than the previously stored $\mathcal{C}_k(S_j)$ (or there are no previously stored $\mathcal{C}_k(S_j)$), it stores the current value of $\mathcal{C}_k(S_j)$ and proceeds with the other sphere decoding steps.

Alternatively, we modify the Viterbi algorithm by imposing the sphere-constrained trellis search. Consider the trellis representation of a frequency-selective channel and a finite data block transmission. We impose the constraint (2.1) that the transmitted signal belongs to a sphere of a radius $r$. As we have shown in the previous section, an obvious necessary condition that the transmitted signal needs to satisfy is given by (3.11). However, from (3.8), this condition is equivalent to the constraint $r^2 \geq \mathcal{C}_1$. Similarly, comparing (3.13) and (3.8), we obtain that $r^2 \geq \mathcal{C}_2$. In general,

$$r^2 \geq \mathcal{C}_k, \quad k = 1, 2, \ldots, T. \tag{3.15}$$

On the trellis, condition (3.15) means that the cost $\mathcal{C}_k$ should, for each state and time index $k$, be smaller than the radius of the sphere. The states that violate condition (3.15) can be neglected, i.e., no branches emanating from such states need to be considered when searching for the optimal trellis path. Therefore, the search on trellis can, on average, be performed faster that the Viterbi algorithm. The worst case

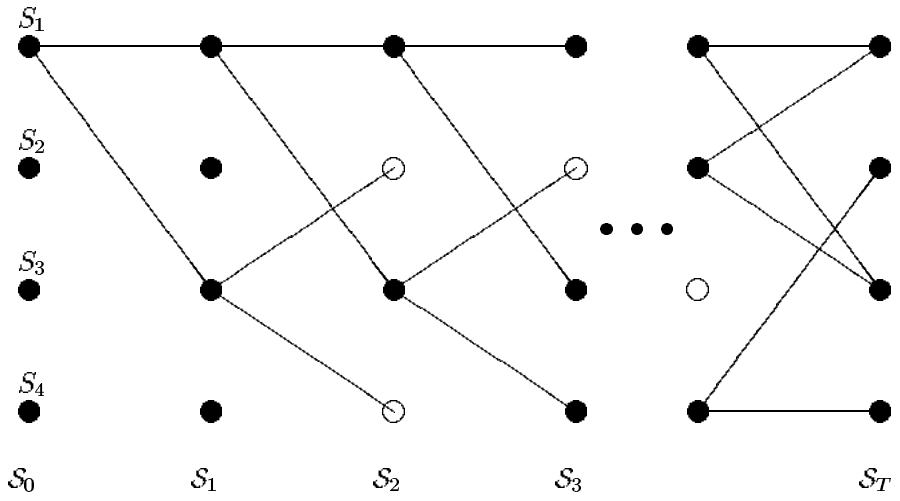


Figure 3.5: *Sphere-constrained search on trellis*

complexity, on the other hand, coincides with the complexity of the Viterbi algorithm.

The sphere-constrained trellis search is illustrated in Figure 3.5, where the "empty" states denote those from which no branch on optimal path can emanate.

As an example, consider a channel of length $l = 8$, transmitting BPSK modulated ($L = 2$) data in blocks of length $T = 20$ at $SNR = 10$dB and employ the sphere-constrained detection on the trellis. Figure 3.6 shows the empirical distribution of the complexity exponent.

As evident from Figure 3.6, the complexity exponent is always smaller than the complexity exponent corresponding to the Viterbi algorithm (denoted by the vertical dashed line). Figure 3.7 shows the expected complexity exponent as a function of SNR. The expected complexity is roughly cubic over the considered SNR range.

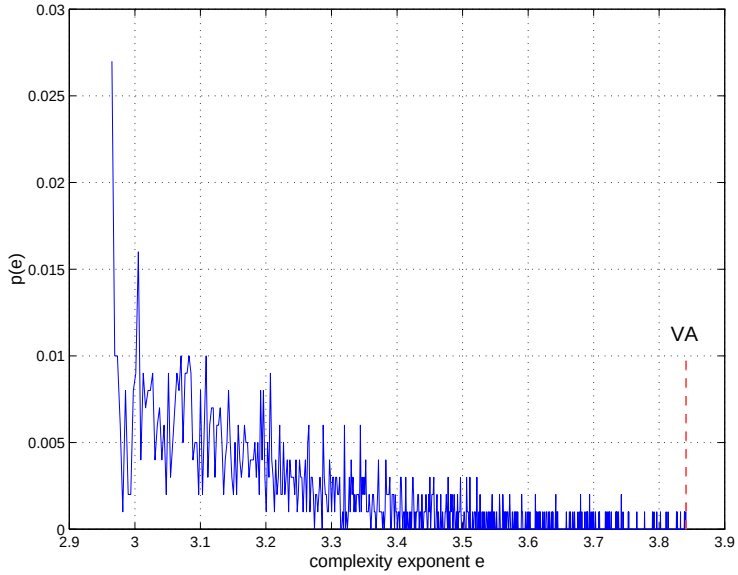


Figure 3.6: *Distribution of complexity exponent,* $l = 8$, $T = 20$, $L = 2$, $SNR = 10dB$.

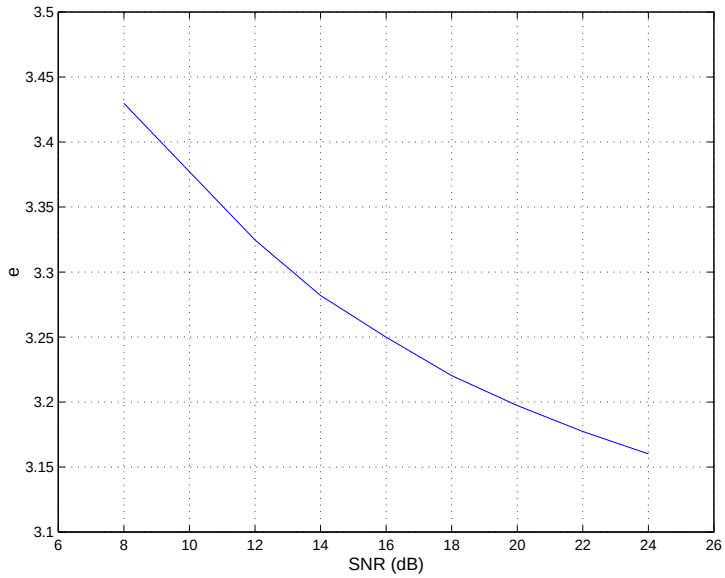


Figure 3.7: *The expected complexity exponent of the sphere-constrained ML detection on trellis,* $l = 8$, $T = 20$, $L = 2$.

## 3.3 Sphere Decoding for MIMO Channels with Memory

We should point out that the sphere decoding algorithm can be employed for detection in multi-antenna systems transmitting over frequency-selective channels. To this end, consider a multiple antenna system with $M$ transmit and $N$ receive antennas. The MIMO channel is modeled as block-fading frequency-selective, where the channel impulse response is constant for some discrete interval $T$, after which it changes to another (independent) impulse response that remains constant for another interval $T$, and so on. The additive noise is spatially and temporally independent identically distributed circularly-symmetric complex-Gaussian. The MIMO channel model is shown in Figure 3.8. The channel is represented by its complex baseband equivalent

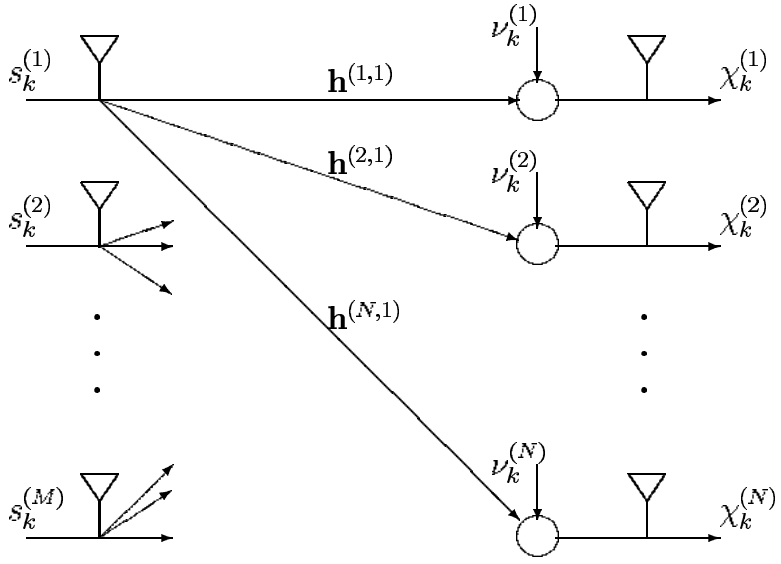


Figure 3.8: *FIR MIMO channel model*

model. Let the column vector

$$\mathbf{h}^{(i,j)} = \left[h_1^{(i,j)} \quad h_2^{(i,j)} \quad \dots \quad h_{l^{(i,j)}}^{(i,j)}\right]'$$

denote the single-input single-output (SISO) channel impulse response from the $j^{th}$ transmit to the $i^{th}$ receive antenna. For convenience, we shall make the following assumptions on the SISO channels $\mathbf{h}^{(i,j)}$:

1. $l^{(i,j)} = l, \quad 1 \leq i \leq N, \quad 1 \leq j \leq M$, that is, all SISO channels have impulse responses of the same length, and

2. the channel coefficients $h_p^{(i,j)}$, $1 \leq p \leq l, \quad 1 \leq i \leq M, \quad 1 \leq j \leq N$ are i.i.d. $\mathcal{C}(0,1)$.

The received signal at the $i^{th}$ antenna can then be expressed as

$$\chi_k^{(i)} = \sum_{j=1}^{M} \sum_{p=1}^{l} h_p^{(i,j)} s_{k-p}^{(j)} + \nu_k^{(i)}. \tag{3.16}$$

Equation (3.16) can be written in a matrix form as

$$\mathcal{X}_k = \sum_{p=1}^{l} H_p \mathcal{S}_{k-p} + \mathcal{V}_k, \tag{3.17}$$

where

- $\mathcal{V}_k \in \mathcal{C}^{N \times 1}$ is the additive noise vector defined as $\mathcal{V}_k = \begin{bmatrix} \nu_k^{(1)} & \nu_k^{(2)} & \dots & \nu_k^{(N)} \end{bmatrix}'$,

- $\mathcal{S}_k = \begin{bmatrix} s_k^{(1)} & s_k^{(2)} & \dots & s_k^{(M)} \end{bmatrix}'$ is the transmit vector, whose entries typically come from a QAM constellation, and

- $H_p \in \mathcal{C}^{N \times M}$ is the $p^{th}$ coefficient matrix in the MIMO channel impulse response,

$$H_p = \begin{bmatrix} h_p^{(1,1)} & h_p^{(1,2)} & \dots & h_p^{(1,M)} \\ h_p^{(2,1)} & h_p^{(2,2)} & \dots & h_p^{(2,M)} \\ \vdots & \vdots & \ddots & \vdots \\ h_p^{(N,1)} & h_p^{(N,2)} & \dots & h_p^{(N,M)} \end{bmatrix}$$

In other words, the $z$-transform of the MIMO channel impulse response is given by

$$H(z) = H_1 + H_2 z^{-1} + \ldots + H_l z^{-(l-1)}$$

Define the following vectors:

$$\begin{aligned} \mathcal{X} &= [\mathcal{X}_1 \ \mathcal{X}_2 \ \ldots \ \mathcal{X}_{T+l-1}]', \\ \mathcal{V} &= [\mathcal{V}_1 \ \mathcal{V}_2 \ \ldots \ \mathcal{V}_{T+l-1}]', \\ \mathcal{S} &= [\mathcal{S}_1 \ \mathcal{S}_2 \ \ldots \ \mathcal{S}_T]'. \end{aligned}$$

[Note that the random vector $\mathcal{V} \in \mathcal{C}^{N(T+l-1)}$ has unit variance complex Gaussian i.i.d. entries, $E[\mathcal{V}\mathcal{V}^*] = I_{N(T+l-1)}$.] Then from (3.17) we can write the input-output relation for the FIR MIMO channel in the matrix form as

$$\mathcal{X} = \mathbf{H}\mathcal{S} + \mathcal{V} \tag{3.18}$$

where $\mathbf{H} \in \mathcal{C}^{N(T+l-1)\times MT}$ is constructed as

$$\mathbf{H} = \begin{bmatrix} H_1 & & & & & & \\ H_2 & H_1 & & & & & \\ \vdots & \vdots & \ddots & & & & \\ H_l & \ldots & & H_1 & & & \\ & \ddots & & & \ddots & & \\ & & H_l & \ldots & & H_1 & \\ & & & \ddots & \vdots & \vdots & \\ & & & & H_l & H_{l-1} & \\ & & & & & H_l & \end{bmatrix} \tag{3.19}$$

We shall assume that symbol bursts are uncorrelated (which is an appropriate assumption when modeling, for instance, packet transmission in TDMA systems).

As earlier in this chapter, we will find it useful to rewrite (3.18) in terms of real quantities. To this end, define

$$\mathbf{x} = [\Re(\mathcal{X}) \quad \Im(\mathcal{X})]', \quad \mathbf{v} = [\Re(\mathcal{V}) \quad \Im(\mathcal{V})]',$$

and

$$\mathbf{H} = \begin{bmatrix} \Re(\mathbf{H}) & -\Im(\mathbf{H}) \\ \Im(\mathbf{H}) & \Re(\mathbf{H}) \end{bmatrix}.$$

Thus, with the previously defined $\mathbf{x} \in \mathcal{R}^{2NT\times 1}$, $\mathbf{v} \in \mathcal{R}^{2NT\times 1}$, and $\mathbf{H} \in \mathcal{R}^{2N(T+l-1)\times 2MT}$, we can rewrite (3.18) as

$$\mathbf{x} = \mathbf{H}\mathbf{s} + \mathbf{v}, \tag{3.20}$$

where the signal vectors $\mathbf{s}$ are typically obtained upon modulation input bits onto an $L$-PAM constellation $\mathcal{D}_L^{2MT}$,

$$\mathcal{D}_L^{2MT} = \left\{-\frac{L-1}{2}, -\frac{L-3}{2}, \ldots, \frac{L-3}{2}, \frac{L-1}{2}\right\}^{2MT}.$$

This particular structure of vector $\mathbf{s}$ stems from an assumption that the entries of $\mathcal{S}$ in (3.18) are points in $L \times L$ QAM constellation. Notice that $\mathcal{D}_L^{2MT}$ is a finite lattice carved from an infinite one, $\mathcal{Z}^{2MT}$.

With the notation introduced earlier in this chapter, since the noise is assumed to be additive Gaussian, we can express ML detection problem as the optimization

$$\min_{\mathbf{s}\in\mathcal{D}_L^{2MT}\subset\mathcal{Z}^{2MT}} \|\mathbf{x} - \mathbf{H}\mathbf{s}\|_2, \tag{3.21}$$

where the minimization is over all points in the constellation $\mathcal{D}_L^{2MT}$. This can be solved with sphere decoding algorithm.

To illustrate performance of the algorithm, we consider couple of examples. We first consider a communication system with $M = 2$ transmit and $N = 2$ receive antennas. The MIMO channel memory is assumed to be $l = 4$, and coherence interval time $T = 4$. Data is modulated onto 4-QAM constellation (corresponding to 2-PAM,

or $L = 2$, in the real-valued set of equation (3.20)). The resulting transmission rate is therefore 4 bits/channel use. Performance comparison of an uncoded transmission in terms of bit error rate (BER) of sphere decoding and nulling and canceling is shown in Figure 3.9. Corresponding complexity exponent as a function of SNR is shown in the same figure. Note that for SNRs above 7dB we obtain sub-cubic complexity.

As another example we consider the same $2 \times 2$ system ($M = 2$, $N = 2$), with

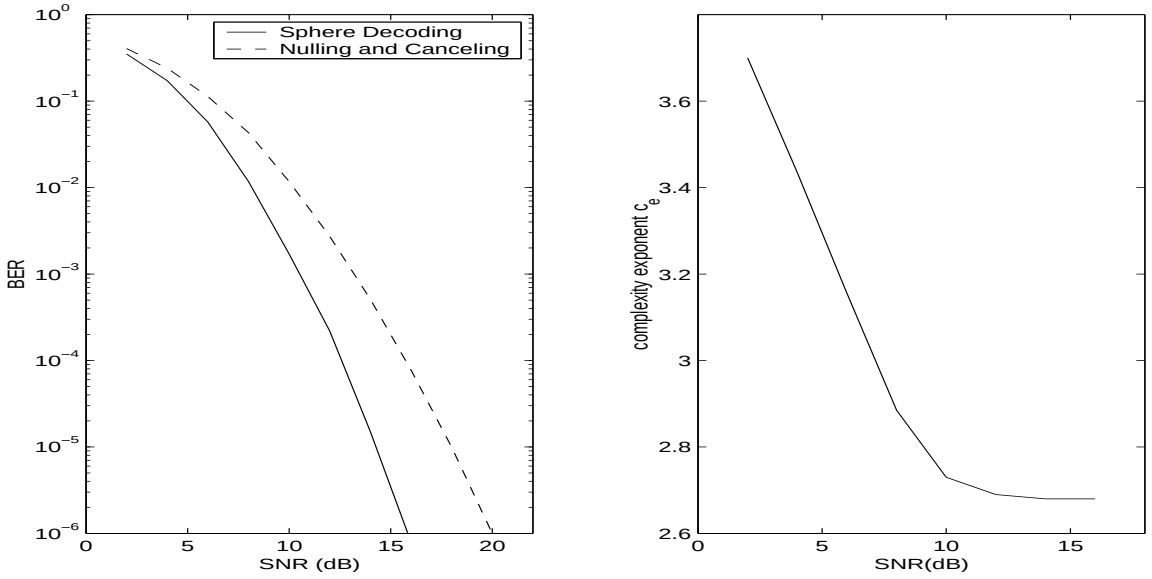


Figure 3.9: $M = 2$, $N = 2$, $l = 4$, $T = 4$, $L = 2$: *performance and complexity*

$l = 4$, but now increase the block length to $T = 8$, and the constellation to 16-QAM, corresponding to $L = 4$ and a transmission rate of 8 bits/channel use. Performance comparison of sphere decoding and nulling and cancelling is shown in Figure 3.10. Corresponding expected complexity exponent for this system is also shown.

As a final example, consider the $4 \times 4$ communication system ($M = 4$, $N = 4$), with $l = 4$ and block length $T = 8$ (and thus $m = 64$). The constellation used is 4-QAM (hence $L = 2$, and corresponding transmission rate is 8 bits/channel use). Performance comparison of sphere decoding and nulling and cancelling for this system is shown in Figure 3.11. The same figure shows corresponding complexity exponent

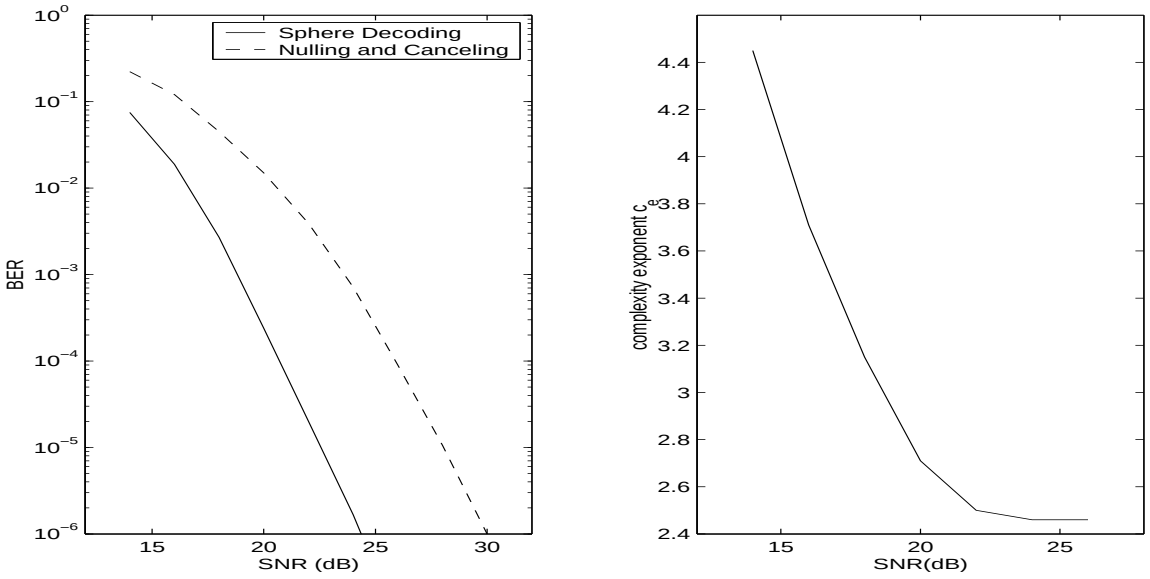


Figure 3.10: $M = 2$, $N = 2$, $l = 4$, $T = 8$, $L = 4$: *performance and complexity*

of sphere decoding, which is sub-cubic for SNRs above 12dB.

## 3.4 Conclusion

In this chapter, we considered sphere decoding algorithms for channels with memory. In particular, for a single-input single-output frequency-selective channels, we demonstrate that there is no need to perform QR factorization in order to employ the algorithm. The banded Toeplitz structure of the channel matrix can be exploited to directly perform the search. Furthermore, we have derived a necessary probabilistic description required for the calculation of the expected complexity in this case.

Moreover, we proposed algorithms that combine the principles of a constrained search of the sphere decoding and a dynamic programming of the Viterbi algorithm. The algorithm can be implemented either on the tree where additional constrains on the state of the channel memory improve the speed of the pruning. Alternatively, a combined algorithm can be employed on trellis, so that the optimal trellis path search

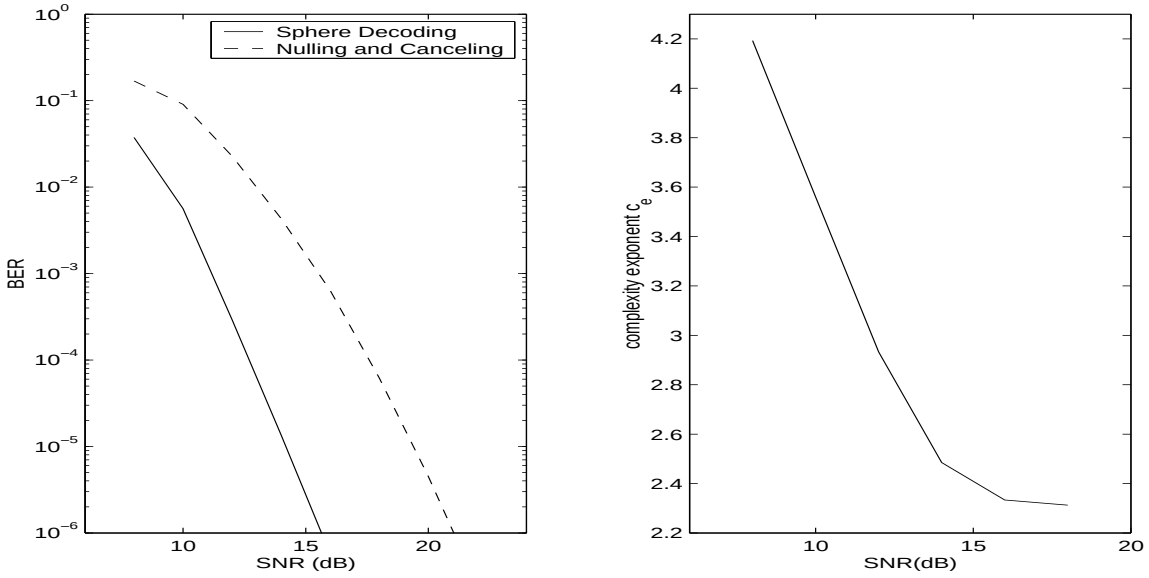


Figure 3.11: $M = 4$, $N = 4$, $l = 4$, $T = 8$, $L = 4$: *performance and complexity*

becomes sphere-constrained.

It turns out that, over a wide range of signal-to-noise ratios and channel/data block lengths, the expected complexity is often cubic or sub-cubic. Moreover, the combined sphere decoding and Viterbi algorithm has a worst-case complexity that equals the complexity of the Viterbi algorithm whereas is much faster on average.

# Chapter 4

# The FP-MAP Algorithm

In this chapter, we consider a design of soft iterative detection schemes in multi-input multi-output communication systems that employ error-correcting codes. In particular, we propose a new multi-input multi-output detector, based on a modification of the original Fincke-Pohst algorithm, which efficiently obtains the so-called *soft* information for the transmitted symbols and, hence, the soft information for the coded bit sequence. In the traditional *hard* detection, the only decision being made by the detector is a crisp decision for each received symbol. Soft detection, on the other hand, provides a *decision reliability* information – that is, it provides the probabilistic measure of the confidence that we have in the hard decisions. The proposed modified Fincke-Pohst algorithm uses information about the transmitted symbols that are available prior to the transmission to essentially perform a maximum a posteriori (MAP) search in the vicinity of the received signal. The points which are returned by the algorithm search (and those are the points that are "good" in MAP sense) are used to generate soft information for the transmitted coded bit sequence, which is then passed on to the channel decoder. The channel decoder, exploiting the properties of the particular error-correcting code that is being used, improves the reliability information and feeds it back to the Fincke-Pohst MAP (FP-MAP) detector for the next iteration. Note that, in general, the channel decoder may have an iterative structure as well, as, for instance, in the cases when the channel code is comprised

of serial or parallel concatenation of codes (as in the turbo codes) or is a low-density parity check code.

Employing error-correcting codes in the communication systems tends to significantly improve the system performance over an uncoded scheme. The error-correcting code protects the signal by

- embedding a redundancy in a form of additional bits to the coded word, hence distributing the information content and enabling the recovery of the bits in error; clearly, decreased information rate is the price one has to pay.

- systematically choosing the order in which the symbols obtained by modulating the codeword are sent across the channel; note that a single codeword may map into a sequence of symbols.

The good error-correcting codes have a wide separation among the codewords. In his work, Shannon [1] showed that one can achieve the channel capacity in systems employing very long random codes at the transmitted and maximum-likelihood decoding at the receiver. However, due to the lack of any structure in the codewords, the maximum-likelihood decoding of such random codes grows exponentially with the codeword length. Finding practical codes that approach capacity has been an open problem in the communications ever since.

Early steps towards designing the efficient coding schemes that can approach capacity were taken in the 1960's. In particular, Gallager [47] invented the *low-density parity check codes* (LDPC), which are essentially long random codes of a special structure that allows for the efficient decoding algorithms. Furthermore, Forney in his PhD thesis [48] studied classes of the concatenated codes with the optimal tradeoff of performance versus complexity. Some thirty years later, the code concatenation became an essential idea behind the *turbo codes*, first introduced in [49]. It is only after the advent of the turbo codes that the Gallager's LDPC codes regained significant attention ([50], [51], [52]). Both turbo and LDPC codes exhibit impressive performance that brings them very close to the channel capacity. For instance, the rate 1/2 LDPC codes in [53] get to 0.0045dB away from the channel capacity.

The common features of the decoders for both the turbo and LDPC codes are the iterative structure and the use of the soft information. The traditional detectors, such as the one shown in the receiver's scheme in Figure 1.6, make hard decisions based on the received signal and pass them on to the channel decoder. Making the hard decision corresponds to finding the symbol constellation point which is closest to the received noise-perturbed signal. The hard detector does not provide any information about the reliability of the hard decisions it makes. However, in order to fully exploit the information contained in the received signal as well as the protection provided by the error-correcting code, one needs to employ the detection algorithms which are capable of producing the soft information and passing them to the channel decoder. It is not clear, though, what is the best way of obtaining this soft information in the multi-input multi-output systems, such as the communications schemes that are employing multiple antennas.

We start the chapter by briefly explaining the coding structures in which the demand for the soft iterative decoding algorithms occurs. In particular, we revisit the turbo and the low-density parity check codes. We follow up by explaining the fundamentals of the iterative decoding techniques and the structure of the decoders for the previously mentioned coding schemes. Then we proceed by describing the coded multi-input multi-output communication system model and deriving the soft FP-MAP detection algorithm. Next, we discuss the expected complexity of the FP-MAP algorithm and show how it can be efficiently upper bounded by the complexity of the Fincke-Pohst algorithm discussed in the Chapter 2. Finally, we employ the FP-MAP algorithm for the soft detection in the multi-antenna communication systems. The simulation results illustrate the bit-error rate performance of the new iterative detection algorithm when it is employed in the combination with both the simple (convolutional) and the powerful (turbo and LDPC) codes. We demonstrate that the system using powerful error-correcting codes achieves near-capacity performance. The bounds for the corresponding expected complexity of the algorithm, which imply that the practical implementation is in fact feasible, are also provided.

## 4.1 Soft iterative decoding

Before deriving the FP-MAP algorithm for the MAP detection in the multi-input multi-output systems and using it to obtain the soft information for the transmitted symbols, it will be instructive to briefly review the probabilistic notions that are often encountered in the design of the soft iterative detection and decoding schemes. Furthermore, to shed some light on the interaction between the detector and the decoder, we provide brief overview of the two classes of the powerful error-correcting codes that we employ later in this chapter for the coding in the multi-antenna communication systems: the turbo and the LDPC codes.

Let $\mathcal{Y}$ denotes the set of all possible values that a random variable $Y$ can take. There are various types of the probabilistic information that, in a relation to a random event $E$, can be assigned to $Y$. In the context of the soft iterative decoding techniques, we shall find it useful to distinguish among the following three types of the probabilistic information:

- *A priori* or *intrinsic* information is the probability that a random variable $Y$ takes a particular value $y \in \mathcal{Y}$, i.e.,

$$P_E^{\text{int}}(Y = y) = P(Y = y)$$

  This probability is drawn prior to the observation of the event $E$ and does not depend on its outcome.

- *A posteriori* information is the conditional probability that a random variable $Y$ takes a particular value $y \in \mathcal{Y}$ upon observing the outcome of the random event $E$, i.e., it is the probability

$$P_E^{\text{post}}(Y = y) = P(Y = y|E)$$

- Given the a priori and the a posteriori information, the definition of the *extrinsic*

information follows from the Bayes' rule,

$$P(Y=y|E) = \frac{1}{P(E)} \underbrace{P(E|Y=y)}_{\frac{1}{c_y} P_E^{\text{ext}}} P(Y=y),$$

and, therefore,

$$P_E^{\text{ext}} = c_y P(E|Y=y).$$

The normalization factor $c_y$ is needed to ensure that the extrinsic probabilities sum up to 1, i.e., $\sum_y P_E^{\text{ext}}(Y=y) = 1$.

Furthermore, for the sake of the computational simplicity, we will find it useful to use a *log-likelihood ratio* as a way of probabilistic description of the random variable. For instance, for a binary random variable $Y$ (i.e., the variable which can only take values $y \in \{0,1\}$), the log-likelihood ratio is defined as

$$LLR(y) = \log \frac{P(Y=1)}{P(Y=0)}.$$

With this, the relation between a priori, extrinsic, and a posteriori information can be expressed as an additive, rather than the multiplicative one:

$$LLR_E^{\text{post}}(y) = LLR_E^{\text{int}}(y) + LLR_E^{\text{ext}}(y) \tag{4.1}$$

Aside from the computational, the log-likelihood ratio representation has an advantage when it comes to developing an intuition about the relations between the different types of the information. In particular, from the relation (4.1) it is clear that the a posteriori information for $Y$ is obtained by incrementally adjusting the previously existing knowledge – that is, it is obtained by adding the newly found extrinsic information to the a priori knowledge. This incremental correction of the existing knowledge is a calculation that is (in some form) performed by the decoder at each step of the soft iterative algorithm.

### 4.1.1 Turbo codes

Turbo codes are the class of codes that employ some form of a code concatenation. Figure 4.1 shows the turbo encoder where the constituent convolutional codes are concatenated in parallel. Serially concatenated structures, as well as the combinations of the two, may be used as well. For the simplicity, we shall focus on the basic parallel encoder structure of Figure 4.1.

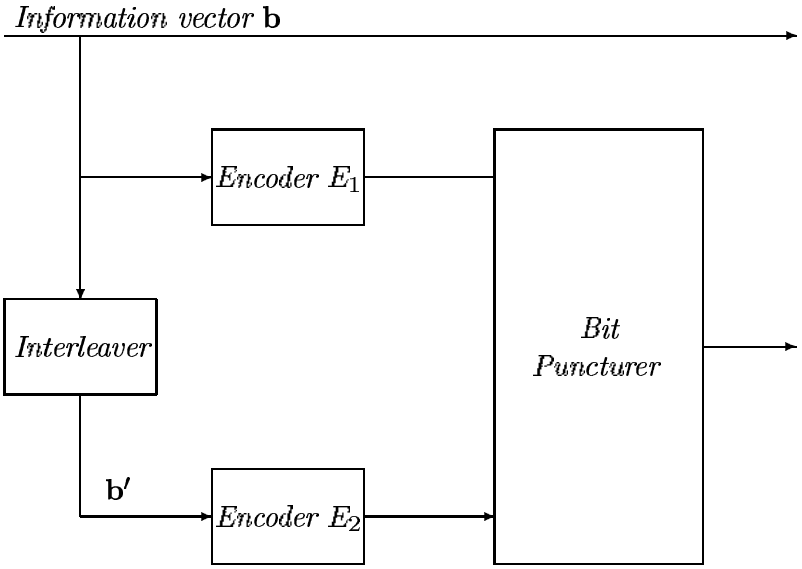


Figure 4.1: *Turbo Encoder*

The information bit vector that we want to transmit, $\mathbf{b} = [b_1 \quad b_2 \quad \ldots \quad b_{l_b}]$, is shown in Figure 4.1 being encoded twice. The first time it is encoded directly, with the convolutional encoder $E_1$. At the same time, the vector $\mathbf{b}'$, obtained upon interleaving the vector $\mathbf{b}$, is encoded with the convolutional encoder $E_2$. The interleaver performs the periodical reordering of the blocks of the information vector bits. The purpose of the interleaving is to protect against the error bursts that are due to, for instance, strong impulsive noise and are localized in time. The de-interleaver in the decoder then disperses the bits in error over time. Another benefit of the interleaving is in achieving pseudo-randomness of the obtained vector. This is exploited by the soft

decoder and shall be addressed later in this chapter.

The encoders $E_1$ and $E_2$ need not to be but we assume that they are employing the same convolutional code. In general, the generator matrix for an arbitrary rate $1/2$ convolutional code is given by

$$G'(D) = [g_1(D) \quad g_2(D)],$$

where $g_1(D)$ is a feedforward, while $g_2(D)$ is a feedback generating polynomial. Furthermore, any non-catastrophic convolutional code with a generator matrix $G'(D)$ can be employed as a recursive systematic convolutional encoder with a generator matrix given by

$$G(D) = \left[1 \quad \frac{g_2(D)}{g_1(D)}\right].$$

Figure 4.2 shows a typical rate $1/2$ recursive systematic convolutional (RSC) encoder. The feature of the "systematic" encoders is that, as illustrated in Figure 4.2, one of the two output bits of the RSC encoder, denoted by $c_1$ in the figure, is equal to the input bit. It is commonly referred to as the systematic (or the information) bit. The remaining output bit, $c_2$, is obtained as a combination of the bits from the information bit sequence and is referred to as the parity-check bit.

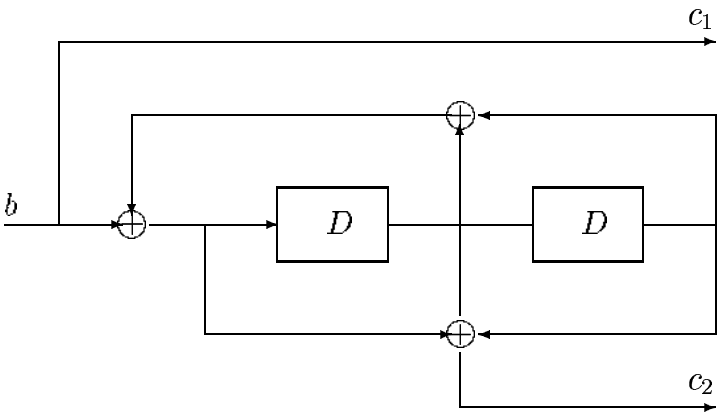


Figure 4.2: *Convolutional code*

The RSC encoder in Figure 4.2 is comprised of generating polynomials $g_1(D) = 1+D^2$ (feedforward) and $g_2(D) = 1+D+D^2$ (feedback). The memory elements $D$ in Figure 4.2 can be implemented by means of the simple shift registers which can store either a logical value 0 or 1. A state $\xi$ of the convolutional encoder is characterized by the content of its memory registers. In particular, for the RSC encoder in Figure 4.2, $\xi$ is a 2-dimensional vector (because there are two memory elements $D$) comprised of 0's and 1's. We shall denote the set of all possible encoder states by $\Xi$.

We should note that if all of the coded bits produced by the two RSC encoders in Figure 4.2 are actually used, then the rate of the resulting turbo code would be $1/3$ – for every time index, there are 3 bits at the output of the turbo encoder: the systematic bit and two parity-check bits. However, one can obtain the higher-rate codes by either using the higher-rates constituent convolutional codes or by performing puncturing. Essentially, puncturing is a periodic deletion of the parity-check bits. For instance, to obtain the rate $1/2$ code, it is sufficient to multiplex parity-check bits obtained by the encoders $E_1$ and $E_2$. In particular, at one instance use only the parity-check bit produced by the encoder $E_1$ while at the next instance use the parity-check bit produced by the encoder $E_2$. Therefore, the even-indexed entries in the coded vector $\mathbf{c} = [c_1 \quad c_2 \quad \ldots \quad c_{l_c}]$ at the output of the rate $1/2$ punctured turbo encoder in Figure 4.2 are equal to the information bits, while the odd-indexed entries in the coded vector $\mathbf{c}$ are the alternated parity-check bits of the two convolutional encoders $E_1$ and $E_2$.

Figure 4.3 illustrates the general structure of the turbo decoder for the soft iterative decoding of the parallel concatenated turbo codes. The a priori information for the coded bit vector, $\lambda(\mathbf{c})$, which is obtained from the channel detector, is used by the soft-input soft-output (SISO) decoder $D_1$ to produce the soft information for the information bit sequence $\mathbf{b}$, $\lambda_1(\mathbf{b})$. The extrinsic part of this information, $\lambda_1^{\text{ext}}(\mathbf{b})$ is first interleaved and then passed on to the soft decoder $D_2$. The decoder $D_2$ uses this information, along with the channel detector information $\lambda(\mathbf{c})$, to obtain its own vector of the soft information for the (interleaved) information bit vector, $\lambda_2(\mathbf{b}')$.

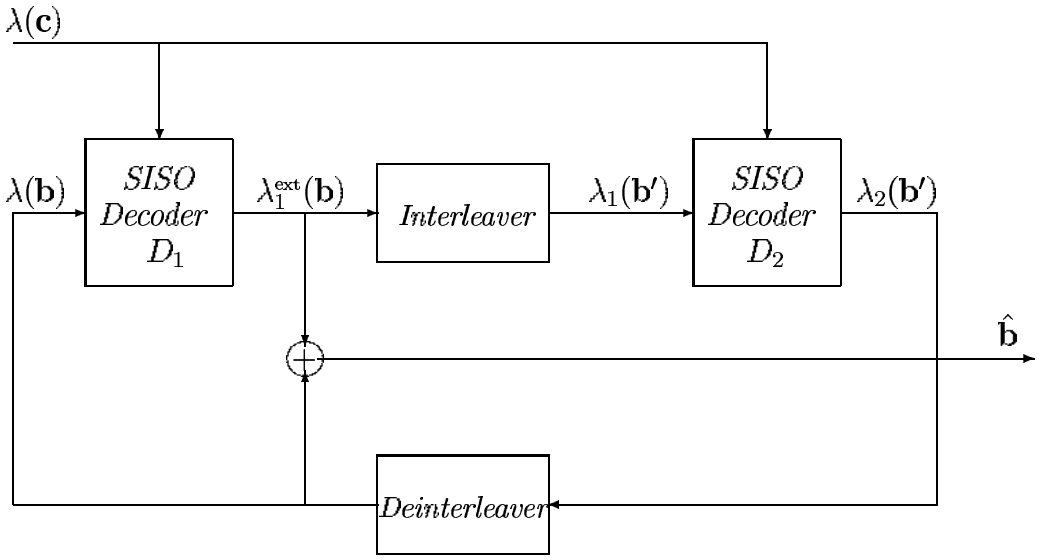


Figure 4.3: *Turbo Decoder*

The extrinsic part of $\lambda_2(\mathbf{b}')$, $\lambda_2^{\text{ext}}(\mathbf{b}')$, is then de-interleaved and used (in combination with $\lambda(\mathbf{c})$) by the decoder $D_1$ to run the soft decision algorithm yet again. The two decoders iterate until some chosen stopping criterion is satisfied, upon which the hard decisions are made by combining the soft information from both decoders, and rounding them.

There are various soft decision algorithms that may be employed as the soft-input soft-output decoding modules in Figure 4.3. A standard algorithm that may be used for that purpose is the MAP (or BCJR) algorithm [57] which performs the symbol-by-symbol maximum a posteriori (MAP) decoding and finds the log-likelihood ratio for each bit $b_k$ in the information bit vector $\mathbf{b}$ as

$$\lambda(b_k) = \ln\left(\frac{\sum_{\xi\in\Xi:\, b_k=1} p(\xi_{k-1}=\xi^{old}, \xi_k=\xi, \mathbf{c})/p(\mathbf{c})}{\sum_{\xi\in\Xi:\, b_k=0} p(\xi_{k-1}=\xi^{old}, \xi_k=\xi, \mathbf{c})/p(\mathbf{c})}\right). \tag{4.2}$$

The summation in the numerator of (4.2) is over all encoder state transitions $\xi_k \rightarrow \xi_{k+1}$ that are caused by an input $b_k = 1$, whereas the summation in the denominator is over all encoder state transitions $\xi_k \rightarrow \xi_{k+1}$ that are caused by an input $b_k = 0$.

The joint probability $p(\xi', \xi, \mathbf{c})$ can be found as

$$\begin{aligned} p(\xi', \xi, \mathbf{c}) &= p(\xi', \mathbf{c}_{1:k-1}) p(\xi, c_k | \xi') p(\mathbf{c}_{k+1:l_c}) \\ &= \alpha_{k-1}(\xi') \gamma_k(\xi', \xi) \beta_k(\xi), \end{aligned}$$

where $\mathbf{c}_{p:q} = [c_p \quad \dots c_q]$. Furthermore, using Bayes' rules, the quantities $\alpha_k(\cdot)$, $\beta_k(\cdot)$, and $\gamma_k(\cdot, \cdot)$ can be found through certain recursions. In particular, $\alpha_k(\cdot)$ can be found through the so-called *forward recursion* of the form

$$\alpha_k(\xi) = p(\xi_k, \mathbf{c}_{1:k}) = \sum_{\xi' \in \Xi} \alpha_{k-1}(\xi') \gamma_k(\xi', \xi).$$

The *backward recursion* for $\beta_k(\cdot)$ can be expressed as

$$\beta_{k-1}(\xi') = p(\mathbf{c}_{k+1:l_c}) = \sum_{\xi \in \Xi} \gamma_k(\xi', \xi) \beta_k(\xi).$$

We assume that the encoder both starts and terminates in the state 0, i.e., the initial conditions for the forward and backward recursions are set to

$$\alpha_0(0) = 1, \quad \alpha_0(1) = 0.$$

and

$$\beta_{l_c+1}(0) = 1, \quad \beta_{l_c+1}(1) = 0.$$

Finally, the quantity $\gamma_k(\cdot, \cdot)$ in (4.2) can be found as

$$\gamma_k(\xi', \xi) = p(c_k | b_k) p(b_k),$$

for each pair of the states $(\xi', \xi)$ and bits $(b_k, c_k)$.

This version of the MAP algorithm is referred to as multiplicative, due to the form of the recursions for $\alpha_k(\cdot)$, $\beta_k(\cdot)$, and $\gamma_k(\cdot, \cdot)$. As a computationally more attractive alternative to the multiplicative MAP algorithm, one can use the additive version

thereof. For instance, to calculate $\alpha_k(\cdot)$, one can use

$$\ln(\alpha_k(\xi)) = \ln(e^{\sum_{\xi' \in \Xi} \alpha_{k-1}(\xi')\gamma_k(\xi',\xi)}). \tag{4.3}$$

The logarithms in (4.3) can be efficiently computed using the standard *Max-Log-MAP* approximation (see, e.g., [59]) of the form

$$\ln(e^{\delta_1} + \ldots + e^{\delta_l}) \approx \max_{1 \le k \le l} \delta_k. \tag{4.4}$$

The approximation (4.4) is both simple and computationally easy but it may not provide desired accuracy. To fix this, one can use an additional correction term to obtain the *log-MAP* solution,

$$\begin{aligned} \ln(e^{\delta_1} + e^{\delta_2}) &= \max(\delta_1, \delta_2) + \ln(1 + e^{-|\delta_2 - \delta_1|}) \\ &= \max(\delta_1, \delta_2) + f_c(|\delta_2 - \delta_1|), \end{aligned} \tag{4.5}$$

where $f_c(\cdot)$ is a correction function found by a table look-up. Relation (4.5) may be used recursively to calculate (4.3). To show this, denote

$$\delta_{|k} = \ln(\sum_{i=1}^{k} e^{\delta_i}) \quad \text{and} \quad \Delta_k = e^{\delta_{|k}}.$$

Then we can write

$$\begin{aligned} \ln(\textstyle\sum_{i=1}^{k+1} e^{\delta_i}) &= \ln(\Delta_k + e^{\delta_{k+1}}) \\ &= \max(\ln \Delta_k, \delta_{k+1}) + f_c(|\ln \Delta_k - \delta_{k+1}|) \\ &= \max(\delta_{|k}, \delta_{k+1}) + f_c(|\delta_{|k} - \delta_{k+1}|) \end{aligned} \tag{4.6}$$

We should mention that another algorithm often used to obtain the soft information is the soft-output Viterbi algorithm (SOVA) proposed in [58]. SOVA is computationally less complex than the Log-MAP but is generally outperformed by the Log-MAP algorithm [59].

### 4.1.2 Low-density parity check codes

The low-density parity check (LDPC) codes do not share the concatenated code structure with the previously described turbo codes. Instead, the LDPC codes are the block error-correcting codes and, as such, are commonly defined via its parity-check matrix $P$. The codewords of the block code with the $p \times q$ parity-check matrix $P$ comprise the set of all $q$-dimensional vectors $\mathbf{c}$ that belong to the null space of $P$, i.e., the allowed code vectors $\mathbf{c}$ are those that satisfy

$$P\mathbf{c}^T = 0.$$

The length of the codeword of the block code with the $p \times q$ parity check matrix is $q$, while $p$ denotes the number of the parity check bits. Hence, the rate of the code is $R_c = (q-p)/p$.

An LDPC code is the block code whose parity-check matrix is sparse. For instance, the binary LDPC code has the parity-check matrix with 0's for most of its entries and only a few 1's. Imposing the sparseness, as we will discuss in this section, allows for an efficient soft iterative decoding of the LDPC codes. The positions of 1's in the parity-check matrix may be chosen randomly or, alternatively, they could be placed to impose some structure to the matrix. The randomness guarantees good performance in the probabilistic sense – the block error rate of the long random block code decreases with the code length [47]. The LDPC codes may be *regular* or *irregular*. The regular binary LDPC codes have a fixed number number of 1's in each column and row. The $(k, l)$ *LDPC code* is the common notation for the code whose parity-check matrix has $k$ non-zero elements in each column and $l$ non-zero elements in each row. On the other hand, the irregular LDPC codes have the parity-check matrix whose columns have a small, but not necessarily equal, number of non-zero elements.

The parity-check matrix defines the set of the allowed codewords but one needs to form a generator matrix $G$ in order to encode the information vector $\mathbf{b}$ as $\mathbf{c} = G\mathbf{b}$. Using the binary Gaussian elimination, we can write the parity-check matrix in the systematic form, $P_{sys} = [I \quad \tilde{P}]$. Then we use the relation between the parity-check

and the generator matrix, $G^T P = 0$ to find the generator matrix as $G = [-\tilde{P}^T \ \ I]$.

A block code can be conveniently represented by a graph which consists of nodes and edges ([62], [63]). The nodes are connected by the edges and the geometric structure of the graph captures the structure of the parity-check matrix. There are two types of the nodes:

- the bit nodes (or the equality nodes), which correspond to the columns of the parity-check matrix, and
- the check nodes (or the parity-check nodes), which correspond to the rows of the parity check matrix.

Each node of the graph may be connected to one or more edges. An edge may be connected to either one or two nodes. Depending on its connectivity, an edge may be

- the internal edge, connecting a bit node with a check node, and
- the external edge, connected to a bit node.

Let $N_i^{(c)}$, $i = 1, \ldots, p$ denote the check nodes, and $N_j^{(b)}$, $j = 1, \ldots, q$ denote the bit nodes. The check node $N_i^{(c)}$ and the bit node $N_j^{(b)}$ are connected by an edge if and only if the $(i, j)$ entry in the parity-check matrix $P$ is non-zero. Therefore, the graph for the LDPC codes has low density of the branches in the graph.

We associate a variable with each edge. An external edge variable corresponds to the soft information for the single bit in the coded bit vector – it is essentially the a priori information for that coded bit (in the log-likelihood ratio form) and was obtained from the detector. The variables on the internal edges are referred to as the state variables of the graph. We will find it useful to denote the internal edge variable on the edge $e_{i,j}$ (the edge which connects the check node $N_i^{(c)}$ with the bit node $N_j^{(b)}$) by $\lambda_{i,j}$. The external edge variable on the edge $e_j$ (the edge which is connected only to the bit node $N_j^{(b)}$) is denoted by $\lambda_j$.

To illustrate the graph representation of a block code, consider the simple example

of the code with the parity check matrix

$$P = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

The graph corresponding to the $P$ above is given in Figure 4.4.

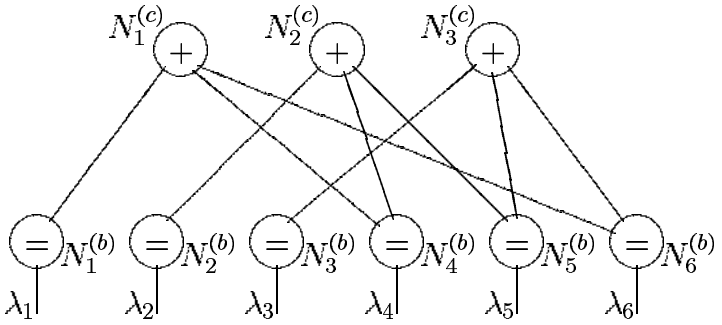


Figure 4.4: *Graph representation of the block code*

Decoding of the LDPC codes is based on the message passing on the graph, i.e., on the iterative updating of the edge variables. There are two essential steps in the message passing algorithm:

- Step 1: *Bit-to-Check Message.* This step is performed in the bit nodes of the graph. All the variables at the edges connected to the particular bit node are supposed to be equal (because the bit nodes carry information about 1's in the columns of the parity-check matrix). Hence, for each internal edge connected to the bit node under consideration, we calculate the bit-to-check message based on the values of the variables on all other edges connected to the same node. The expression for the bit-to-check message can be found by employing the Bayes' rule on the set of variables under the equality constraint (see, e.g., [60]),

$$\lambda_{i,j}^{(c-b)} = \sum_{P(k,j)=1, k \neq i} \lambda_{k,j} + \lambda_j,$$

where $\lambda_{i,k}$ is the current value of the variable on the internal edge $e_{i,k}$ and $\lambda_i$ is the value of the variable on the external edge $e_i$. Once the bit-to-check message calculation is done for all the edges, we update the values of the internal edge variables, $\lambda_{i,j}^{(c-b)} \rightarrow \lambda_{i,j}$.

Figure 4.5 shows the relevant edge variables needed for the calculation of the bit-to-check messages in the bit node $N_4^{(b)}$ of the code graph in Figure 4.4.

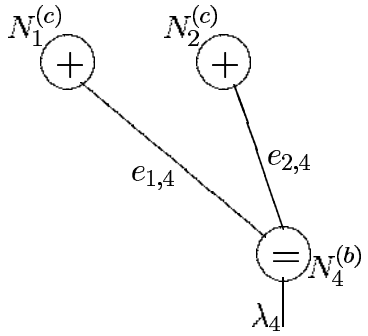


Figure 4.5: *Bit-to-check message in node* $N_4^{(b)}$

- Step 2: *Check-to-Bit Message.* This step is performed in the check nodes of the graph. The check nodes specify the parity check constraints imposed by the parity-check matrix $P$. For any edge variable connected to the check node under consideration, we find a check-to-bit message employing the Bayes' rule on the set of variables under the parity-check constraint ([60]),

$$\lambda_{i,j}^{(b-c)} = \Phi^{-1}\left(\prod_{P(i,k)=1, k\neq j} \Phi(\lambda_{i,k})\right),$$

where $\Phi(x) = \tanh(-x/2)$, and $\lambda_{i,k}$ is, as before, the current value of the variable on the edge $e_{i,k}$. After completing the check-to-bit calculation for all the edge variables, their value is updated, $\lambda_{i,j}^{(b-c)} \rightarrow \lambda_{i,j}$.

Figure 4.6 shows the relevant edge variables needed for the calculation of the check-to-bit messages in the bit node $N_1^{(c)}$ of the code graph in Figure 4.4.

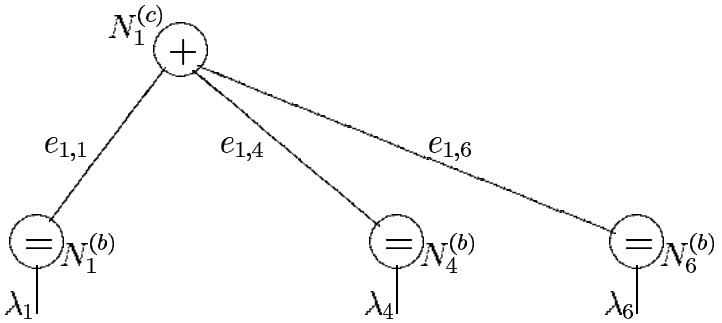


Figure 4.6: *Parity check node*

The sum-product algorithm evaluates the steps above until the certain stopping criteria (e.g., the number of the iterations) is satisfied. The decoder output is then calculated as

$$\lambda_j^{(\text{out})} = \sum_{P(i,j)=1} \lambda_{i,j} + \lambda_j.$$

The resulting values of the variables on the external edges, $\lambda_j^{(\text{out})}$, are then used to make hard decisions for the information vector bits. Note that for the initialization of the sum-product algorithm, we require the a priori information for the bits in the coded vector (i.e., the a priori information $\lambda_i$ for the external edges). This need to be provided by the soft detector.

The sum-product algorithm essentially breaks a potentially rather complex problem into a set of simple calculations which are executed iteratively. The algorithm gives an exact MAP decision when the graph contains no loops. If the graph in fact does contain a loop, the approximation is shown empirically to be good. The decoding complexity of the sum-product algorithm is linear in the length of the codeword, i.e., it is $O(q)$.

We should note that the sparseness condition on the parity-check matrix will, in general, result in the non-sparse generator matrix $G$, placing a burden on the encoder. To this end, some efficient encoding structures have been proposed in [61]. Another way to ensure the efficient encoding is to require that the parity-check matrix is both systematic and sparse, which yields generator matrix of the low density. However,

placing such constraint deteriorates the performance of the LDPC code [60].

The soft decoding of both the turbo and the LDPC codes may be considered in the same framework using, for instance, the belief propagation networks [64]. However, in this review we focused on the originally proposed and currently most commonly used iterative decoder structures for the two families of the codes.

## 4.2 System Model

We shall consider the multiple antenna communication system which employs an error-correcting code to protect the transmitted data against the channel disturbances. We assume that the number of the transmit antennas, $M$, is not greater than the number of the receive antennas, $N$, i.e., that $M \leq N$. This assumption is due to the fact that the detection problem translates into solving a system of equations and this condition ($M \leq N$) ensures that there are at least as many equations as unknowns. When $M > N$, a space-time code such as an LD code [30] can be used to get the appropriate number of equations and unknowns (more on the space-time codes will be said in the following chapters).

Furthermore, we assume a discrete-time block-fading multiple antenna channel model, where the channel is known to the receiver. This is a reasonable assumption for communication systems where the signaling rate is much faster than the pace at which the propagation environment changes, so that the channel may be learned, e.g., via transmitting known training sequences. We do not assume that the knowledge of the channel is available at the transmitter.

During any channel use, the transmitted signal $\mathbf{s} \in \mathcal{Z}^{M\times 1}$ and the received signal $\mathbf{x} \in \mathcal{C}^{N\times 1}$ are related by

$$\mathbf{x} = \sqrt{\frac{\rho}{M}}\mathbf{H}\mathbf{s} + \mathbf{v}, \tag{4.7}$$

where $\mathbf{H} \in \mathcal{C}^{N\times M}$ is the known channel matrix, and $\mathbf{v} \in \mathcal{C}^{N\times 1}$ is the additive noise vector, comprised of the independent, identically distributed complex-Gaussian entries $\mathcal{C}(0,1)$. If we constrain the entries in $\mathbf{s}$ and $\mathbf{H}$ to have, on the average, unit

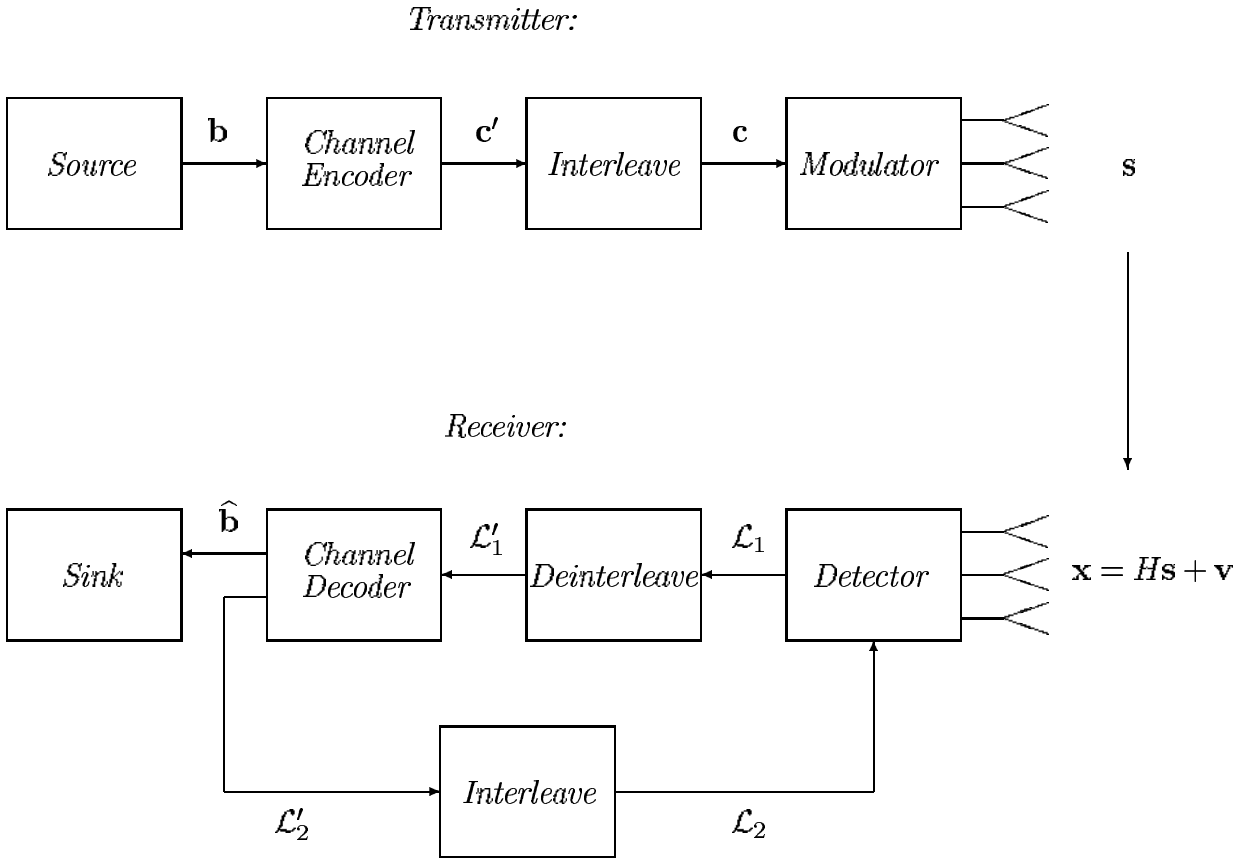


Figure 4.7: *System model*

variance, then $\rho$ is the expected received signal-to-noise ratio.

The system model is shown in Figure 4.7. The vector of information bits $\mathbf{b}$ is encoded with the error-correcting code to obtain the vector of coded bits $\mathbf{c}'$, which is then interleaved to result in the coded vector $\mathbf{c}$. The vector $\mathbf{c}$ is modulated onto a QAM-constellation to form the sequence of symbols which need to be transmitted across the channel. The rate of the transmission is determined by the number of the transmit antennas and the number of points in the constellation. If we assume a $Q$-QAM modulation scheme, the number of the bits transmitted across the channel per each channel use is given by

$$R = M \log Q. \tag{4.8}$$

To see this, assume that each constellation symbol may represent $p_m$ coded bits. For instance, in the $Q$-QAM constellation, the number of the coded bits modulated onto the single constellation symbol is

$$p_m = \log_2 Q.$$

Thus the modulation is performed by taking the blocks of vector $\mathbf{c}$, each one of the length $p_m M$, and mapping these blocks into the $M$-dimensional complex-valued symbol vectors. We will assume that the bits are mapped according to the simple gray mapping discussed in Chapter 1. The resulting symbol vectors are transmitted across the channel as described by the model (4.7). Therefore, a block of $R = p_m M$ coded bits (corresponding to a single symbol vector) is sent across the channel per each channel use.

Clearly, the size of the coded vector $\mathbf{c}$ may be quite long and can not, in general, be represented by a single symbol vector $\mathbf{s}$ in (4.7). Therefore, to send the entire vector $\mathbf{c}$ across the channel, the channel has to be used multiple times. Let us denote the previously described blocks of coded bits as $\mathbf{c}^{[1]}, \mathbf{c}^{[2]}, \ldots \mathbf{c}^{[p_c]}$, each one of the length $p_m M$. Furthermore, we assume for simplicity that the total length of the vector $\mathbf{c}$ is $l_c = p_c p_m M$. Then the entire vector $\mathbf{c}$ can be blocked as

$$\mathbf{c} = [\mathbf{c}^{[1]} \quad \mathbf{c}^{[2]} \quad \ldots \quad \mathbf{c}^{[p_c]}], \tag{4.9}$$

and therefore transmitted in $p_c$ channel uses.

Consider the $k^{th}$ channel use, that is, consider the time index $k$ when the block $\mathbf{c}^{[k]}$ has been modulated onto the symbol vector $\mathbf{s}$ and sent across the channel. The receiver observes the noise corrupted and multi-antenna channel altered signal $\mathbf{x}$ in (4.7). Assume that the a priori probabilities for the components of the transmitted symbol vector $\mathbf{s}$, $\{p(\mathbf{s}_1), p(\mathbf{s}_2), \ldots, p(\mathbf{s}_M)\}$, are known to the receiver. The multi-input multi-output detector in Figure 4.7 processes the received vector $\mathbf{x}$ and the available a priori probabilities to obtain the soft information for the coded bits in the current

block $\mathbf{c}^{[k]}$. [Note that for the first iteration of the detector, the a priori information for the symbols is generally not available; if that is the case, we assume that all symbols are equally likely.]

Let us denote the bits in the block $\mathbf{c}^{[k]}$ by $c_i$, $i = 1, 2, \ldots, p_m M$. We consider the log-likelihood ratio form of the soft information for the bits in the block $\mathbf{c}^{[k]}$, given by

$$L_1(c_i|\mathbf{x}) = \log \frac{P[c_i = +1|\mathbf{x}]}{P[c_i = -1|\mathbf{x}]}. \tag{4.10}$$

Let us denote the reliability information for the block $\mathbf{c}^{[k]}$ by

$$\mathcal{L}_1^{[k]} = [L_1(c_1|\mathbf{x}) \quad L_1(c_2|\mathbf{x}) \quad \ldots \quad L_1(c_{p_m M}|\mathbf{x})],$$

and let $\mathcal{L}_1$ denotes a vector of concatenated blocks of the reliabilities,

$$\mathcal{L}_1 = [\mathcal{L}_1^{[1]} \quad \mathcal{L}_1^{[2]} \quad \ldots \quad \mathcal{L}_1^{[p_c]}],$$

collected over all $p_c$ uses of the channel. Then $\mathcal{L}_1$ is a vector of LLRs corresponding to all the bits in the coded vector $\mathbf{c}$.

As illustrated in Figure 4.7, the vector $\mathcal{L}_1$ is de-interleaved to obtain vector $\mathcal{L}_1'$, which is then passed to the channel decoder. The channel decoder forms the estimate of the information bit vector, $\widehat{\mathbf{b}}$, and provides $\mathcal{L}_2'$, the new posteriori reliability information for the coded bits vector $\mathbf{c}'$. Posteriori reliability information for the vector $\mathbf{c}$ is obtained by de-interleaving $\mathcal{L}_2'$ into $\mathcal{L}_2$. Let us denote the posteriori reliability information for the block $\mathbf{c}^{[k]}$ by $\mathcal{L}_2^{[k]}$. Furthermore, assume that the bits $c_i$, $i = 1, 2, \ldots, p_m M$, in the block $\mathbf{c}^{[k]}$ are mutually independent. [This is a valid approximation for a long vector $\mathbf{c}$ and an efficient interleaver which makes its output pseudo-random.] Then the a posteriori probabilities for the components of the symbol vector $\mathbf{s}$ (which is the symbol vector corresponding to the block $\mathbf{c}^{[k]}$) can easily be found from $\mathcal{L}_2^{[k]}$ using the modulator mapping function and the independence assumption for the bits in the vector $\mathbf{c}^{[k]}$ – the a posteriori probability for the component $s_i$ of the symbol vector $\mathbf{s}$ is simply an appropriate product of the a posteriori probabilities

of those bits of $\mathbf{c}^{[k]}$ which are represented by the constellation point $s_i$.

The probabilities $\{p(s_1), p(s_2), \ldots, p(s_M)\}$ can now be used to run the multi-input multi-output detector algorithm (i.e., evaluate (4.10)) once again. All the previously described operations now can be repeated as we use the described scheme for the iterative joint detection and decoding in the multi-antenna communication system.

The structure of the channel decoder in Figure 4.7 clearly depends upon the choice of the error-correcting code. If we employ a simple convolutional code, then the channel decoder is a simple soft-in soft-out decoder, such as the MAP, the Max-Log-MAP, or the log-MAP algorithm described earlier in this chapter. When the channel code is a powerful turbo or LDPC code, then the channel decoder has an iterative structure itself, as we discussed in the previous section of this chapter.

The computational complexity of traditional algorithms for evaluating (4.10) can be prohibitive for the practical implementation in the systems employing the multiple antennas due to the sheer size of the symbol space. On the other hand, as we have shown in Chapter 2, the sphere decoding algorithm of Fincke and Pohst can supply us with the (otherwise computationally rather demanding) maximum-likelihood estimate of $\mathbf{s}$ with reasonable complexity. Therefore, one may speculate whether a modification can be devised to yield the required soft information with low complexity. In the next section, we show that this is indeed possible to do. Moreover, we show that the expected complexity of the algorithm remains low – in particular, bounded above by the expected complexity of the Fincke-Pohst algorithm – indicating that it may as well be employed in the practical systems.

## 4.3 Modified FP Algorithm for MAP Detection

The transmitted symbols $\mathbf{s}$ in (4.7) are chosen from a QAM constellation and, therefore, are complex-valued. However, since we wish to state the MAP detection problem in a form that has an integer least-square problem in its core, we need to first find the real-valued equivalent of the equation (4.7). This can be done following the procedure described in Chapter 2, where we found that the real-valued equivalent of (4.7) may

be given by

$$x = Hs + v.$$

Recall that $s = [s_1 \;\; s_2 \;\; \ldots \;\; s_m]$ is comprised of the entries that belong to an $L$-PAM constellation $\mathcal{D}_L$. Therefore, the unknown vector $s$ is a point in the $m$-dimensional integer lattice $\mathcal{D}_L^m$.

The MAP detector maximizes the posterior probability $p_{s|x}(s|x)$, i.e., it solves the optimization problem

$$\max_{s \in \mathcal{D}_L^m} p_{s|x}(s|x). \tag{4.11}$$

Using the Bayes' rule, we can write

$$\arg\max_{s \in \mathcal{D}_L^m} p_{s|x}(s|x) = \arg\max_{s \in \mathcal{D}_L^m} \frac{p_{x|s}(x|s)p_s(s)}{p_x(x)} = \arg\max_{s \in \mathcal{D}_L^m} p_{x|s}(x|s)p_s(s)$$

Furthermore, by assuming that the entries $s_1, s_2, \ldots, s_m$ of the symbol vector $s$ are mutually independent, we can write

$$p_s(s) = \prod_{k=1}^{m} p(s_k) = e^{\sum_{k=1}^{m} \log p(s_k)}.$$

Then, for a known channel in AWGN, the maximization problem (4.11) is equivalent to the optimization problem

$$\min_{s \in \mathcal{D}_L^m} \left[ \|x - Hs\|^2 - \sum_{k=1}^{m} \log p(s_k) \right]. \tag{4.12}$$

Therefore, by solving (4.12), we can obtain the solution to the MAP detection problem (4.11). However, for an iterative detection and decoding scheme, we also require the soft information, i.e., the probability that each bit in the transmitted vector is decoded correctly. To this end, consider the log-likelihood ratio defined in (4.10) and, as in the previous section, consider the $k^{th}$ channel use and assume that the current symbol vector $s$ is obtained by modulating coded block $\mathbf{c}^{[k]} = [c_1 \;\; c_2 \;\; \ldots \;\; c_{p_m M}]$ onto an $L$-PAM constellation. Then the log-likelihood ratio for each bit $c_i$ in the block $\mathbf{c}^{[k]}$

can be expressed as

$$L_1(c_i|x) = \log \frac{p[c_i = +1|x]}{p[c_i = -1|x]} = \log \frac{p[x, c_i = +1]}{p[x, c_i = -1]},$$

where $i = 1, 2, \ldots, p_m M$. By considering the set of all possible blocks $\mathbf{c}^{[k]}$, we note that the LLR can be found as

$$L_1(c_i|x) = \log \frac{\left[ \sum_{\mathbf{c}^{[k]} : c_i = +1} p[x, \mathbf{c}^{[k]}] \right]}{\left[ \sum_{\mathbf{c}^{[k]} : c_i = -1} p[x, \mathbf{c}^{[k]}] \right]}$$

and, using the Bayes' rule again, as

$$L_1(c_i|x) = \log \frac{\left[ \sum_{\mathbf{c}^{[k]} : c_i = +1} p[x|\mathbf{c}^{[k]}] p[\mathbf{c}^{[k]}] \right]}{\left[ \sum_{\mathbf{c}^{[k]} : c_i = -1} p[x|\mathbf{c}^{[k]}] p[\mathbf{c}^{[k]}] \right]}. \tag{4.13}$$

Assuming mutual independence of the bits $c_1, c_2, \ldots, c_{p_m M}$, the expression (4.13) can be written as the sum of two terms,

$$L_1(c_i|x) = \underbrace{\log \frac{p[c_i = +1]}{p[c_i = -1]}}_{L_1^{\text{int}}(c_i)} + \underbrace{\log \frac{\left[ \sum_{\mathbf{c}^{[k]} : c_i = +1} p[x|\mathbf{c}^{[k]}] \prod_{j, j \neq i} p[c_j] \right]}{\left[ \sum_{\mathbf{c}^{[k]} : c_i = -1} p[x|\mathbf{c}^{[k]}] \prod_{j, j \neq i} p[c_j] \right]}}_{L_1^{\text{ext}}(c_i)},$$

where $L_1^{\text{int}}(c_i)$ and $L_1^{\text{ext}}(c_i)$ denote the intrinsic (a priori) and extrinsic part of the total soft information, respectively. In an iterative detection and decoding scheme, only the extrinsic part of the soft information, $L_1^{\text{ext}}(c_i)$, is passed to the other decoding block(s) in the scheme.

Since the block $\mathbf{c}^{[k]}$ is uniquely mapped into the symbol vector $s$, it follows that the terms in the numerator and the denominator of (4.13) may be summed over the symbol space, i.e., (4.13) can be written as

$$L_1(c_i|x) = \log \frac{\left[ \sum_{s:c_i=+1} p[x|s] \prod_j p[s_j] \right]}{\left[ \sum_{s:c_i=-1} p[x|s] \prod_j p[s_j] \right]}. \tag{4.14}$$

For the known channel in additive white Gaussian noise, the log-likelihood ratio of (4.14) can be written as

$$L_1(c_i|x) = \log \frac{\left[ \sum_{s:c_i=+1} e^{-||x-Hs||^2+\sum_j \log p[s_j]} \right]}{\left[ \sum_{s:c_i=-1} e^{-||x-Hs||^2+\sum_j \log p[s_j]} \right]} \tag{4.15}$$

The summations in the numerator and the denominator in (4.15) are over the entire symbol space $\mathcal{D}_L^m$, which is often of prohibitive complexity. Therefore, instead of calculating (4.15) exactly, we constrain ourselves to include only those $s \in \mathcal{D}_L^m$ for which the argument of the optimization problem (4.12) is small. The reasoning for this decision is that the symbol vectors which yield small argument in (4.12) are the symbol vectors whose contribution to the numerator and the denominator in (4.15) are significant. This implies that the chosen set of symbol vectors will provide a good approximation of the exact log-likelihood ratio while, at the same time, requiring much lower number of computations as compared to the exhaustive summation over the entire symbol space.

We note that the optimization problem (4.12) is reminiscent of the standard integer least-squares problem, except that is has an additional non-linear term – the sum of logarithms of the a priori probabilities of the unknowns. Applying the idea of the Fincke-Pohst algorithm, we search for the points $s$ that belong to the geometric

body described by

$$r^2 \geq (s-\hat{s})^* R^* R(s-\hat{s}) - \sum_{k=1}^{n} \log p(s_k), \tag{4.16}$$

where $R$ is the lower triangular matrix following QR factorization of channel matrix $H$. Note that this geometric body is no longer a hypersphere. The search parameter $r$ in (4.16) can be chosen according to the statistical properties of the noise and the a priori distribution of the symbol vector $s$.

Expanding the right-hand side of (4.16) gives

$$\begin{aligned} r^2 \;&\geq\; r_{mm}^2 (s_m - \hat{s}_m)^2 + \log p(s_m) \\ &+\; r_{m-1,m-1}^2 \left( s_{m-1} - \hat{s}_{m-1} + \frac{r_{m-1,m}}{r_{m-1,m-1}} (s_m - \hat{s}_m) \right)^2 + \log p(s_{m-1}) \ldots \end{aligned}$$

Looking at the first term in the above expansion, a necessary condition that $s_m$ has to satisfy in order for (4.16) to hold readily follows,

$$r_{mm}^2 (s_m - \hat{s}_m)^2 - \log p(s_m) \leq r^2. \tag{4.17}$$

Moreover, for every $s_m$ satisfying (4.17), we define

$$r_{m-1}^2 = r^2 - r_{mm}^2 (s_m - \hat{s}_m)^2 + \log p(s_m),$$

and from the second term in the expansion obtain a stronger necessary condition for (4.16) to hold,

$$r_{m-1,m-1}^2 \left( s_{m-1} - \underbrace{\hat{s}_{m-1} + \frac{r_{m-1,m}}{r_{m-1,m-1}} (s_m - \hat{s}_m)}_{\hat{s}_{m-1|m}} \right)^2 - \log p(s_{m-1}) \leq r_{m-1}^2.$$

The procedure continues until all components of the vector $s$ are found. The FP-MAP algorithm can be summarized as follows:

*Input:* $R$, $x$, $\hat{s}$, $r$, $p_s(\mathbf{s})$.

1. Set $k = m$, $r'^2_m = r^2 - \|x\|^2 + \|H\hat{s}\|^2$, $\hat{s}_{m|m+1} = \hat{s}_m$

2. (Bounds for $s_k$) Set $z = \frac{r'_k}{r_{kk}}$, $UB(s_k) = \lfloor z + \hat{s}_{k|k+1} \rfloor$, $s_k = \lceil -z + \hat{s}_{k|k+1} \rceil - 1$.

3. (Increase $s_k$) $s_k = s_k + 1$. If $r^2_{kk}(s_k - \hat{s})^2 > r'^2_k + \log p(s_k)$ and $s_k \leq UB(s_k)$, go to 3, else proceed. If $s_k \leq UB(s_k)$ go to 5, else to 4.

4. (Increase $k$) $k = k + 1$; if $k = m + 1$, terminate algorithm, else go to 3.

5. (Decrease $k$) If $k = 1$ go to 6. Else $k = k - 1$, $\hat{s}_{k|k-1} = \hat{s}_k + \sum_{j=k+1}^{m} \frac{r_{kj}}{r_{kk}}(s_j - \hat{s}_j)$, $r'^2_k = r'^2_{k+1} - r^2_{k+1,k+1}(s_{k+1} - \hat{s}_{k+1|k+2})^2 + \log p(s_{k+1})$, and go to 2.

6. Solution found. Save $s$ and go to 3.

The FP-MAP algorithm returns all points $s$ in the symbol space $\mathcal{D}_L^m$ which satisfy the constraint (4.16). The number of the points found by the algorithm depends on the search parameter $r$ in (4.16). Assume that the algorithm returned $l_s$ points and denote their set by $\mathcal{S}$,

$$\mathcal{S} = \{s^{(1)}, s^{(2)}, \ldots, s^{(l_s)}\} \subset \mathcal{D}_L^m.$$

The vector $s \in \mathcal{S}$ which minimizes (4.12) is the solution to the MAP detection problem (4.11). The soft information for each bit $c_i$ can be estimated from (4.15), by only summing those terms in the numerator and the denominator for which the symbol vector $s$ belongs to $\mathcal{S}$, i.e., we approximate the log-likelihood ratio as

$$L_1(c_i|x) \approx \log \frac{\left[ \sum_{s \in \mathcal{S} : c_i = +1} e^{-\|x - Hs\|^2 + \sum_j \log p[s_j]} \right]}{\left[ \sum_{s \in \mathcal{S} : c_i = -1} e^{-\|x - Hs\|^2 + \sum_j \log p[s_j]} \right]}.$$

This is the soft information which is used by the channel decoder in the fashion described earlier in the chapter.

Before discussing the expected computational complexity of the FP-MAP algorithm, we should note that the logarithms in (4.15) can be efficiently computed using the efficient max-Log-MAP or the Log-MAP implementations.

## 4.4 Computational Complexity of FP-MAP Algorithm

The expected computational complexity of the FP-MAP algorithm could, in principle, be found by following the outline of the calculation of the expected complexity of the original Fincke-Pohst algorithm which was given in Chapter 2. Recall that there we needed two key ingredients in order to calculate the expected number of lattice points visited by the algorithm (and, consecutively, the expected complexity of the algorithm); those are:

(1) a probability that an arbitrary lattice point $s_a$ belongs to a $k$-dimensional sphere of radius $r$ around the transmitted point $s_t$.

(2) a technique for enumerating points $s_a$ in the lattice with respect to the transmitted point $s_t$.

We used the enumeration in (2) to count the number of the points visited by the algorithm which have the same argument of the probability in (1) and to efficiently calculate the expected complexity. Therefore, one would suspect that if these two listed ingredients could be determined for the FP-MAP algorithm as well, it would be possible to exactly compute its expected complexity.

However, the probability that an arbitrary point $s_a = [s_{a,1} \quad s_{a,2} \quad \dots \quad s_{a,m}]$ belongs to the $k$-dimensional sphere of radius $r$ around the transmitted point $s_t$ is an incomplete gamma function of $k$ degrees of the freedom,

$$p_{\mathbf{s}_a} = \gamma\left(\frac{\alpha n + \frac{12\rho}{m(L^2-1)}\sum_{j=1}^{m}\log p(s_{a,j})}{1+\frac{12\rho}{m(L^2-1)}||s_t - s_a||^2}, \frac{k}{2}\right). \tag{4.18}$$

The argument of the incomplete gamma function in (4.18) is the function of both the Euclidean distance between the points $s_t$ and $s_a$, as well as the a priori probabilities of the components of $s_a$.[1] This, in fact, makes the computation of the expected number of points quite cumbersome. The reason is as follows: since each point $s_a$ in a lattice has a distinct a priori probability affiliated with it, the argument of the probability function in (4.18) will, in general, be different for each pair of the points $(s_t, s_a)$. Hence, there is no efficient way to collect the points in order to ease the calculation of the expected complexity – in fact, to compute the expected number of points that the FP-MAP algorithm visits, one needs to consider all possible pairs of points $(s_t, s_a)$ and the corresponding probabilities (4.18). This, as the size of the problem increases, quickly becomes a rather cumbersome calculation. Furthermore, due to the iterative nature of the algorithm, this calculation does not make much sense nor is necessarily introspective – at each iteration, the a priori probabilities of the points $s_a$ change and unless we knew the time progression of the a priori probabilities of all lattice points ahead of the iterative process (which we do not), we can not calculate the exact expected complexity.

In fact, we may get more insight in the expected complexity of the FP-MAP algorithm by considering some upper bounds. Note that since

$$\sum_{j=1}^{m} \log p(s_j) \leq 0,$$

we have

$$\alpha n + \frac{12\rho}{m(L^2-1)} \sum_{j=1}^{m} \log p(s_j) \leq \alpha n.$$

Hence from the expressions (2.12) and (4.18) and the fact that the incomplete gamma function is monotonically increasing with its argument, it follows that for the same choice of the search parameter $r$,

$$p_{s_a}^{FP-MAP} \leq p_{s_a}^{FP},$$

[1]Recall that the corresponding probability for the Fincke-Pohst algorithm in (2.12) was only a function of the Euclidean distance between $s_t$ and $s_a$.

and we may conclude the following:

**Lemma 1** *For the same choice of the search parameter $r$, the expected complexity of the FP-MAP algorithm is upper bounded by the expected complexity of the original sphere decoding algorithm.*

■

In Chapter 2, we have shown that the expected complexity of the sphere decoding algorithm is polynomial over a wide range of SNRs. This suggests that the expected complexity of the FP-MAP algorithm is also polynomial in $m$ (and often cubic) over a wide range of rates and SNRs, which implies that it may be employed in the practical communication systems.

We should point out that every time the FP-MAP algorithm runs, it is initiated with the different set of the a priori probabilities for the components of the symbol vector in (4.16). Therefore, from the detector's point of view, the iterations are concerned with redefining the search regions. In particular, each new iteration defines (through the new set of the a priori probabilities and the search parameter $r$) the shape and the size of the geometric body that the algorithm searches over for the lattice point which are "good" in the MAP sense, that is, the points with small metric of the optimization in (4.12).

Since the expected complexity of the Fincke-Pohst algorithm bounds the expected complexity of the FP-MAP, the Theorem 1 of Chapter 2 can be used to choose the search parameter $r$ so that the expected number of points found by the FP-MAP does not exceed certain predetermined number. However, one should also require that the FP-MAP algorithm does not end up finding too few points since. Otherwise, the approximation of the log-likelihood ratio (4.14) may not be very good. In fact, if the number of points from one iteration to another drops significantly, it is conceivable that the algorithm may in fact diverge. Therefore, when setting the parameter search $r$ in (4.16), one should guarantee that a certain minimum number of points is found. If not, the search parameter $r$ should be increased and the algorithm run again.

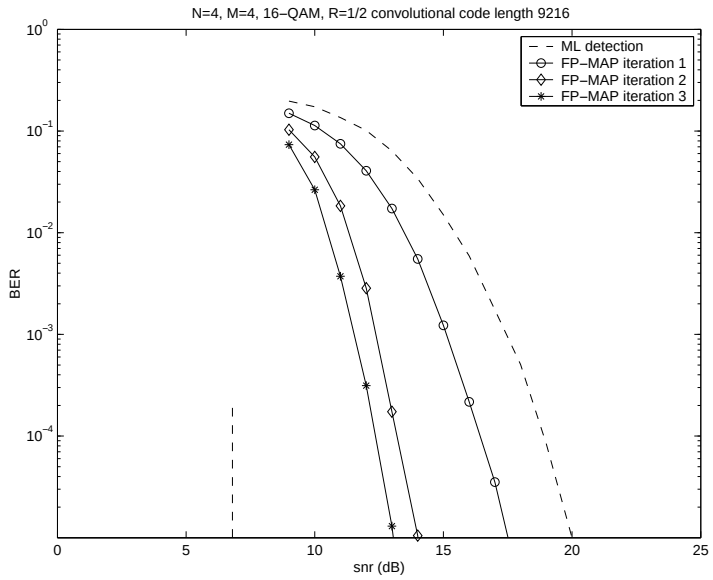


Figure 4.8: *Rate* $1/2$ *convolutional code,* $4 \times 4$ *system,* 16-*QAM*

The procedure of selecting the set of points which are considered for the decoding that the FP-MAP algorithm employs, is conceptually similar to the principles of the traditional *list decoding* algorithms encountered in the theory and practice of decoding for block codes.

## 4.5 Simulation Results

In this section we present the simulation results illustrating the performance and the complexity of the FP-MAP algorithm employed in the multi-antenna communication system of Figure 4.7.

Figure 4.8 shows the bit-error rate (BER) performance of the multi-antenna system which employs the rate $R_c = 1/2$ convolutional code with generating polynomials $g_1(D) = 1 + D^2$ (feedforward polynomial) and $g_2(D) = 1 + D + D^2$ (feedback polynomial) and the code length $l_c = 9216$. The multi-antenna system has $M = 4$ transmit and $N = 4$ receive antennas and employs 16-QAM modulation scheme. The outermost (dashed) curve in Figure 4.8 shows BER performance for the system in

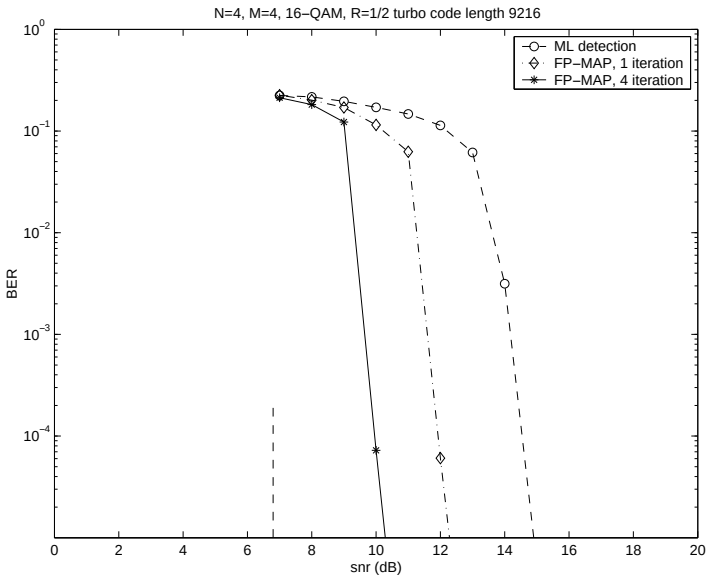


Figure 4.9: *Rate* 1/2 *turbo code,* $4 \times 4$ *system,* 16-*QAM*

Figure 4.7 when the detector provides only the ML optimal hard decisions. These hard decisions are then passed on to the channel decoder which performs hard decoding to retrieve the information vector. The remaining curves in Figure 4.8 are obtained with the FP-MAP algorithm in place of the detector. We notice that with more iterations, the performance improves and, furthermore, the improvements start to saturate after only $2-3$ iterations. The vertical dashed line denotes the channel capacity. We note that with the required reliability $BER = 10^{-5}$, the system gets to 6dB away from the channel capacity after only $k = 3$ iterations.

Figure 4.9 shows the bit-error (BER) performance of the parallel concatenated turbo coded system with the constituent convolutional codes described above ($R_c = 1/2$, $g_1(D) = 1 + D^2$, $g_2(D) = 1 + D + D^2$, $l_c = 9216$). The code is punctured to obtain the overall code rate $R_c = 1/2$. The multi-antenna system has $M = N = 4$ transmit and receive antennas and employs 16-QAM modulation scheme. The total transmission rate of the system is

$$R = R_c 2M \log L = 8 \text{ bits/channel use.}$$

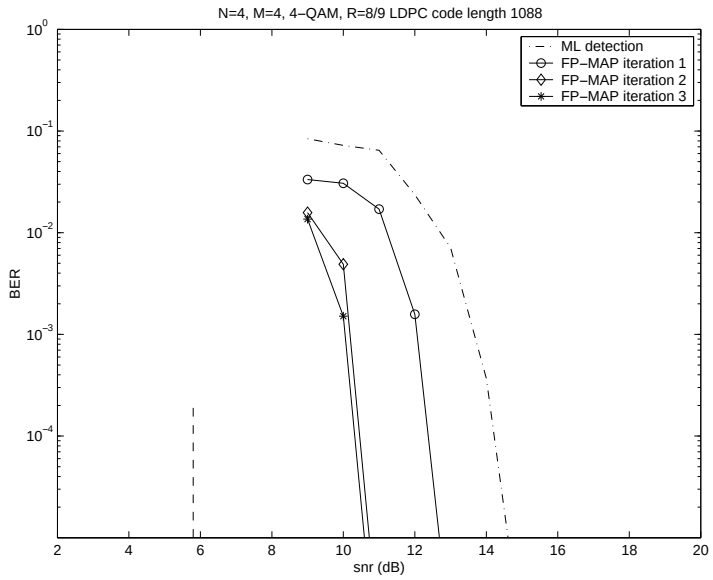


Figure 4.10: *Rate* 8/9 *LDPC code,* $4 \times 4$ *system,* 4*-QAM*

For each run of the detector, the turbo decoder performs 8 iterations of its own. As the Figure 4.9 illustrates, when the FP-MAP algorithm is used for the detector, at $BER = 10^{-5}$ the systems gets to only 3.3dB away from the channel capacity. It is interesting to note that when the FP-MAP runs only once (i.e., there are no iterations between the detector and the turbo decoder), the performance of the system already improves by $\approx$ 3dB with respect to the system that employs the hard ML detection followed by the turbo decoder.

Figure 4.10 shows the BER performance of the same system ($N = M = 4$), employing 4-QAM modulation and the rate $R_c = 8/9$ LDPC code of length $l_c = 1088$, with the column weight 4. When the LDPC decoder receives information from the detector, it performs 8 iterations before passing what it inferred about the coded bits back to the detector. The total transmission rate of the system is $R = 7.1$ bits per channel use. We see from Figure 4.10 that at $BER = 10^{-5}$, the system gets to $\approx$ 4.5dB away from the channel capacity.

It is worthwhile comparing the last two schemes. The parameters of the two systems are chosen on purpose to result in the similar schemes. In particular, the

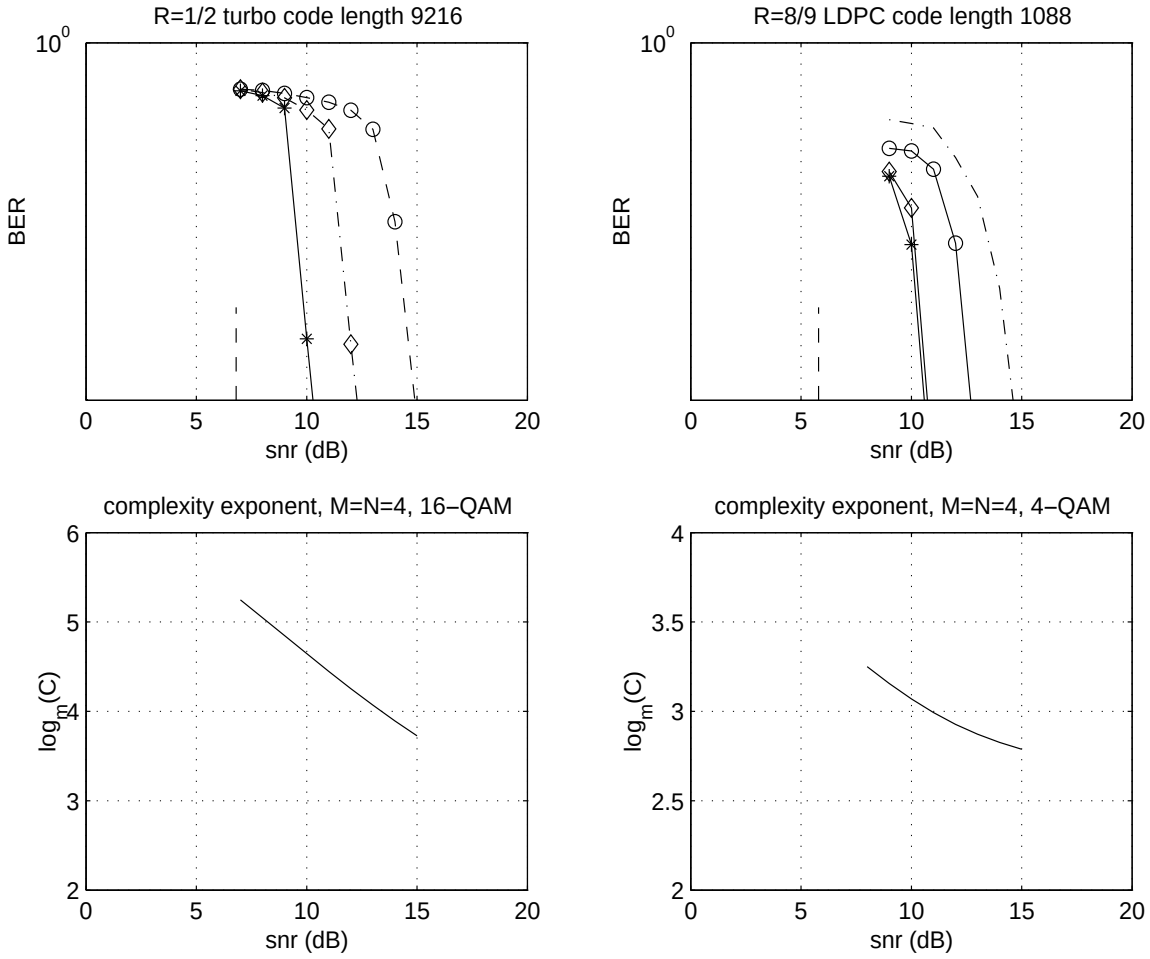


Figure 4.11: *Complexity exponents for rate* $1/2$ *turbo and rate* $8/9$ *LDPC codes*

rates of the systems are comparable (8 vs. 7.1 bits per channel use) while the capacity gap favors the scheme employing the turbo code (3.3dB vs. 4.5dB). Although the LDPC code is therefore seemingly outperformed by the turbo code, it proves to be an interesting alternative, especially in light of the complexity exponents $\log_m \mathcal{C}(m, L, \rho)$ shown in Figure 4.11. As indicated here, the expected complexity bound of the detection in the system employing the (high rate) LDPC code is approximately cubic in $m$, while the complexity in the system with the (1/2 rate) turbo code is significantly higher.[2] This goes to show that, not surprisingly, one needs to carefully consider all aspects of the system before deciding on the elements of the receiver.

In particular, the FP-MAP algorithm may not be appropriate for the detection scheme in the systems which employs low-rate error-correcting codes, due to relatively high expected complexity, as was illustrated in the previous examples. However, the reason that designers build the multi-antenna systems is their capability to provide the high-rate data transmissions, while the capacity gap (at least in the current state of the research) is not quite the highest priority. As the previous examples illustrate, the multi-antenna systems geared towards delivering high data rates, which also implies not-too-high error-correcting code rates, are exactly the scenarios in which the FP-MAP has the expected complexity which is acceptable in practice.

The reason why the expected complexity is higher for the low-rate error-correcting codes may be inferred from Figure 4.12, where the mutual informations of the system with $M = 4$ transmit and $N = 4$ receive antennas employing 4-QAM and 16-QAM schemes are shown. The dashed line curve in the same figure represents the capacity of the channel (i.e., the mutual information optimized over unconstrained symbol distributions and maximized for the Gaussian). Assume that an error-correcting code and a soft decoding scheme may be employed. Furthermore, assume that the expected SNR at the channel is, for instance, $SNR = 10$dB. From Figure 4.12 is evident that, to obtain a spectral efficient scheme at this SNR and 4-QAM and 16-QAM schemes, we have two options:

[2]The bound is clearly not an exact measure of the complexity in this case but is indicative of the true complexity.

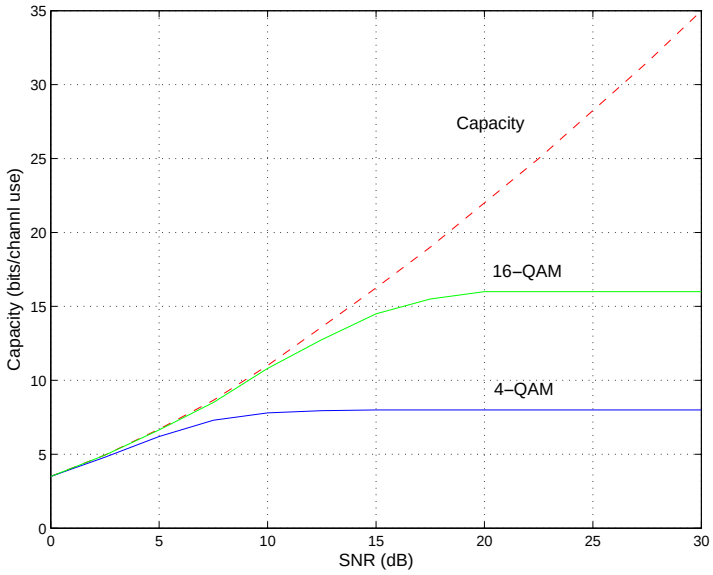


Figure 4.12: *Capacity for* $4 \times 4$ *system, and mutual information for the system of same size and* 4*-QAM and* 16*-QAM modulation schemes.*

($a$) Employ the 4-QAM scheme and (relatively) high-rate error-correcting code, or

($b$) Employ the 16-QAM scheme and (relatively) low-rate error-correcting code.

The option ($b$) is (potentially, if the mutual information is achieved) more efficient, as is implied by the small gap to the channel capacity. However, as we concluded in Chapter 2, for the given modulation scheme, the expected complexity of the sphere decoding is polynomial over the range of SNRs which supports an uncoded transmission at the rate implied by the particular modulation scheme. From Figure 4.12, the SNR for the mutual information achieving ($b$) is quite below the range of the SNRs where that (low expected complexity) would be the case. The option ($a$), although not as efficient, is much more feasible from the complexity standpoint, because the SNR belongs to the range where the sphere decoding has low polynomial expected complexity. Therefore, we conclude that in coded systems, the spectral efficiency and the low ML/MAP detection complexity via sphere decoding are two competing requirements.

## 4.6 Conclusion

In this chapter, we developed a modification of the Fincke-Pohst algorithm to perform maximum a posteriori detection. The Fincke-Pohst maximum a posteriori (FP-MAP) algorithm performs a search in the region of the symbol space whose side and shape is determined by both the statistical properties of the noise and a priori probabilities of the elements of the symbol space. The algorithm finds a set of points used to estimate the soft information for the transmitted coded bit sequence. This soft information is then used as an input to the soft decoder for the channel code.

When applied for the soft detection in the multi-antenna systems employing error-correcting codes, the FP-MAP algorithm provides reliable performances of the systems which are close to the channel capacity. This was demonstrated on the systems employing both the turbo and the low-density parity check codes. As we have shown, for instance, when combined with the standard off-shelf turbo decoder, the proposed algorithm allows one to get to approximately 3.3dB away from the channel capacity while achieving the bit-error rate performance on the order of $10^{-5}$.

We considered the expected complexity of the algorithm and found that, under some simple conditions, it is bounded above by the expected complexity of the original sphere decoding algorithm. Hence, over a wide range of rates and SNRs, the FP-MAP algorithm has the polynomial-time (often cubic) expected complexity in the number of transmit antennas. Moreover, the examples that we considered clearly indicate the need for careful examination of all system parameters before selecting the detection scheme that is (complexity-wise) most appropriate for the particular system at hand. In particular, the expected complexity of the FP-MAP is an indicative parameter that can serve as a guide of the appropriateness of employing this algorithm in a scheme that is being considered. For a given set of system parameters, one should analyze the expected complexity and, as a result, may or may not find the FP-MAP appropriate for the practical implementation in the system at hand. However, we can state this differently from the synthesis point of view: if one decides to make the FP-MAP algorithm of choice for the soft detection in the communication system

under design, then the other parameters of the system may (and should) be chosen to minimize the computational complexity of the FP-MAP algorithm (as well as the overall system).

Although the simulations we presented were for the equal number of the transmit and receive antennas $M = N$, once can still use the methodology for $M > N$, provided one uses, for instance, an LD code [30] to guarantee that the integer least-squares problem has at least as many equations as unknowns.

# Chapter 5

# Joint Detection and Decoding

When designing practical communication systems, the signal detection problem is often considered separately from the data decoding problem. The disjoint approach to the design of these two receiver's components is primarily motivated by the desire to alleviate the otherwise heavy computational burden. However, though the computational complexity of the receiver which employs detection and decoding in separate stages is potentially lower (with the existing algorithm, at least) than the complexity of the receiver which does them jointly, there is a performance price to pay. To overcome the performance losses, soft decoding techniques, such as those in Chapter 3, use the probabilistic information at the output of the first stage (i.e., the soft information about the detected symbols) as the input to the second stage – the decoder. The subsequent iterative exchange of the soft information between the receiver's stages attempts to extract all the information about the original uncoded message that is contained in the received signal.

In this chapter, we will take a non-iterative approach to the detection/decoding problem in multi-input multi-output communication systems which employ error-correcting codes. In particular, we consider the joint maximum-likelihood and maximum a posteriori detection and decoding problems in the multi-antenna systems, where the data is encoded by the linear block error-correcting codes and then transmitted across the channel in AWGN. To this end, we develop an algorithm for solving

the integer least-squares problem where not all the points in the underlying "rectangular" lattice are potential solutions; instead, only those of them generated by a linear block transform of the elements in a finite field are allowed. The algorithm draws on the Fincke-Pohst idea and performs a search for the lattice points inside a sphere. However, an important additionally imposed constraint needs to be satisfied: the admissible lattice points must not only be inside the sphere but also have to be valid codewords.

The criteria for the choice of the radius of the sphere is statistical and hence the computational complexity of the algorithm is a random variable. We quantify it by means of its first moment – the expected complexity – which we find analytically, in a closed form. The expected complexity of the joint maximum-likelihood detection and decoding (JDD-ML) algorithm is polynomial in the length of the uncoded information word over a wide range of SNRs. Simulation results illustrating the performance of the algorithm and its expected complexity are also presented. Furthermore, for the case when the a priori information for the information data are known, we consider the maximum a posteriori joint detection and decoding problem and derive an algorithm (JDD-MAP) for solving it.

## 5.1 The JDD-ML Algorithm

We consider the communications over the multi-input multi-output channel in the AWGN. The information data vector is first encoded by a linear block code. The coded vectors are modulated onto a QAM-constellation and transmitted across the channel. As we have shown in Chapter 2, the maximum-likelihood problem is equivalent to the integer least-squares problem or, geometrically, to the closest point search in the lattice. Obviously, in the communication setup that we just described, not all lattice points are allowed – the code selects the subset of the lattice (corresponding to the codebook of the block code) which may be transmitted. Therefore, we are hopeful that this "sparseness" of the allowed lattice points imposed by the code may be exploited to our advantage and lead to the low complexity solution of otherwise

rather cumbersome problem. We examine that in this section.

Assume a real-valued discrete-time block-fading channel model (or, if the channel model is originally complex, find its real-valued equivalent as described in the previous chapters) of the form

$$x = Hs + v, \tag{5.1}$$

where $H \in \mathcal{R}^{n \times m}$ is an equivalent channel matrix with i.i.d. Gaussian entries, and $v \in \mathcal{R}^{n \times 1}$ is a noise vector with i.i.d. $\mathcal{N}(0,1)$ entries. Entries in the symbol vector $s$ are obtained upon modulating the coded word $\mathbf{c}$ onto an $L$-PAM constellation. The vector $\mathbf{c}$, on the other hand, is obtained by encoding the information vector $\mathbf{b}$ by the linear block code.

The linear block code is defined by the code generator matrix $\mathbf{G}$. The dimensions of the matrix $\mathbf{G}$ determine the rate of the code. For instance, if the size of the generator matrix is $k \times m$, the rate of the code is $R_c = k/m$. The coded vector

$$\mathbf{c} = [c_1 \; c_2 \; \ldots \; c_m]^T$$

is constructed from the information vector

$$\mathbf{b} = [b_1 \; b_2 \; \ldots \; b_p]^T$$

by encoding it as

$$\mathbf{c} = G \otimes \mathbf{b}. \tag{5.2}$$

The entries in $\mathbf{c}$, $\mathbf{b}$, and $\mathbf{G}$, are assumed to be elements from a Galois field $GF(L)$. We chose the size of the Galois field to be as same as the size of the PAM constellation for simplicity. Operation $\otimes$ in (5.2) denotes a multiplication over Galois field, i.e., multiplication modulo $L$.

The maximum-likelihood decoding problem in the communications system described by (5.1),(5.2) can be stated as the following optimization problem,

$$\min_{\mathbf{b} \in GF(L)^p} \|x - Hs\|^2. \tag{5.3}$$

The space over which we optimize is the information vector space $GF(L)^p$. One way of obtaining the solution to the maximum-likelihood decoding problem is by means of an exhaustive search. However, expanding on the basic idea of the Fincke-Pohst approach, we propose an efficient alternative to the exhaustive search: the algorithm that performs the optimization (5.3) by searching only over those points in the lattice $Hs$ that belong to a hypershere around the observed point $x$. In addition, only those lattice points inside the hypersphere that are also admissible codewords (i.e., those points which, when demodulated, satisfy relation (5.2) for some $\mathbf{b} \in GF(L)$) are considered to be the candidates for the solution.

We essentially seek a procedure that would, just like the Fincke-Pohst does for the ordinary integer least-squares, decouple the problem (5.3) – i.e., turn the maximum-likelihood search into a set of $p$ ($p$ being the size of the vector $\mathbf{b}$) linear inequality conditions which, when all satisfied, guarantee that the point $s$ generated by $\mathbf{b}$ belongs to the sphere. These conditions ought to be stated in a convenient "nested" form – that is, when a bit $b_{p-i+1} \in GF(L)$ is determined from the $i^{th}$ inequality condition, that particular bit is substituted for in the remaining $(i-1)$ inequalities; then the same is repeated for $b_{p-i}$ and so on. In the end, each vector $\mathbf{b}_s$ that satisfies the set of $p$ conditions should be such that the vector $\mathbf{c}_s = \mathbf{G} \otimes \mathbf{b}_s$, when modulated onto an $L$-PAM constellation and processed by the lattice generator matrix (i.e., the channel), results in a lattice point $Hs$ that belongs to the sphere around $x$. If more than one vector $\mathbf{b}_s$ is found by the search, then the vector $\mathbf{b}$ among them that minimizes (5.3) is the solution to the ML decoding problem.

For convenience, henceforth we shall denote the modulation operation by $[\cdot]$, i.e., the fact that the point $s$ is obtained by modulating the code vector $\mathbf{c} = \mathbf{G} \otimes \mathbf{b}$ onto an $L$-PAM constellation will be denoted by

$$s \triangleq [\mathbf{G} \otimes \mathbf{b}].$$

As mentioned earlier, we need to state a set of the one-dimensional conditions that the components of a vector $\mathbf{b}$ need to satisfy so that $[\mathbf{G} \otimes \mathbf{b}]$ belongs to the searching

sphere. To do so, we first require some pre-processing of the matrix $\mathbf{G}$. In particular, we transform the code generator matrix $\mathbf{G}$ into an *approximately upper-triangular* form, that is, starting from a given matrix $\mathbf{G}$, we find its equivalent generator matrix $G$ of the form

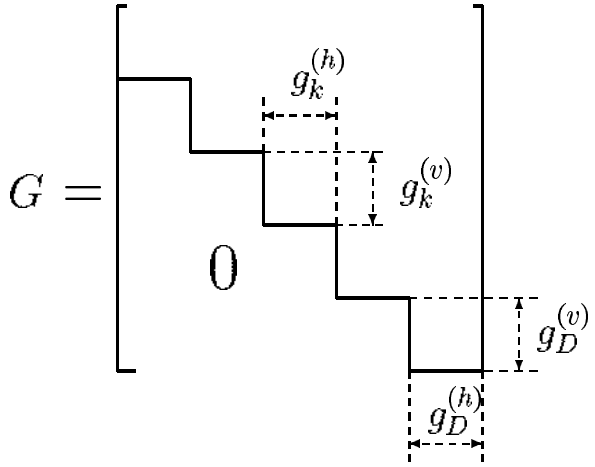


Figure 5.1: *Approximately upper-triangular form of generator matrix G*

The operations that are allowed in the process of transforming $\mathbf{G}$ to the approximately upper triangular form of Figure 5.1 are permutations and additions of the rows. Then the resulting matrix $G$ generates the same code as the starting $\mathbf{G}$, even though a particular information vector $\mathbf{b}$ may, in general, result in a different codeword upon encoding. As illustrated in Figure 5.1, after the preprocessing is done, the resulting matrix $G$ has the following structure:

- the columns $\left(\sum_{i=1}^{D-1} g_i^{(h)} + 1\right)$ to $\sum_{i=1}^{D} g_i^{(h)}$ have no fixed zero entries,
- the columns $\left(\sum_{i=1}^{k} g_i^{(h)} + 1\right)$ to $\sum_{i=1}^{k+1} g_i^{(h)}$ have $\sum_{i=k+2}^{D} g_i^{(v)}$ fixed zero entries, $k = 1, \ldots, D-2$,
- finally, the columns 1 to $g_1^{(h)}$ have $\sum_{i=2}^{D} g_i^{(v)}$ fixed zero entries.

Note that

$$\sum_{i=1}^{D} g_i^{(h)} = p, \quad \sum_{i=1}^{D} g_i^{(v)} = m,$$

where $D$ is the number of "diagonal steps" in $G$, i.e., the number of different column weights (the weight of the column is the average value of its entries). Furthermore, we shall refer to the set of the ratios,

$$\{g_1^{(h)}/g_1^{(v)}, \ldots, g_D^{(h)}/g_D^{(v)}\},$$

as a *diagonal profile* of the matrix $G$. When $g_k^{(h)}/g_k^{(v)} = p/m$, $k = 1, \ldots, D$, we say that the diagonal profile is *regular*. The reason for introducing the notion of the profile is that by focusing first on the matrices with the regular profiles, we can derive a simpler version of the JDD-ML algorithm and develop some intuition that we shall find useful when later considering the general case.

We shall start the description of the algorithm by considering the special case of a rather simple block code. Assume that the information bit vector $\mathbf{b}$ is encoded by the rate $R_c = 1/2$ binary code, whose generator matrix $G$ has the regular diagonal profile. Then the entries $c_{2i-1}$ and $c_{2i}$ of the coded vector $\mathbf{c}$ can be expressed as a linear combination of the bits $(b_i,\ b_{i+1}\ \ldots\ b_p)$ only,

$$c_{2i-1} = \sum_{j=1}^{i} G(2i-1, j) \otimes b_j, \quad \text{and} \quad c_{2i} = \sum_{j=1}^{i} G(2i, j) \otimes b_j.$$

Recall how we earlier assumed that the size of the Galois field, occupied by the elements of the uncoded and coded vectors, is as same as the size of the PAM constellation from which the elements of the transmitted symbol vector are chosen. As a consequence, the symbols $s_{2i-1}$ and $s_{2i}$ shall, just like the coded bits $c_{2i-1}$ and $c_{2i}$, depend only on the information bits $(b_i,\ b_{i+1}\ \ldots\ b_p)$.

Now, recall that the point $s$ lies in a sphere of radius $r$ around $x$ if and only if it holds that

$$r^2 \geq \|x - Hs\|^2 = (s - \hat{s})^* H^* H (s - \hat{s}) + \|x\|^2 - \|H\hat{s}\|^2, \tag{5.4}$$

where $\hat{s} = H^{\dagger}\mathbf{x}$ is the unconstrained least-squares estimate. As in the sphere decoding

algorithm, we shall chose the radius of the sphere according to the statistics of the noise,

$$r^2 = \alpha m \sigma^2,$$

where the scaling coefficient $\alpha$ is chosen so that there is a high probability that we find a lattice point inside the sphere.

Introducing the $QR$ decomposition $H = QR$, and defining

$$r'^2 = r^2 - \|x\|^2 + \|H\hat{s}\|^2,$$

we can write condition (5.4) as

$$\begin{aligned} r'^2 &\geq (s-\hat{s})^* R^* R (s-\hat{s}) \\ &= \sum_{i=1}^{2p} r_{2i-1,2i-1}^2 \left( s_{2i-1} - \hat{s}_{2i-1} + \sum_{j=2i}^{2p} \frac{r_{2i-1,j}}{r_{2i-1,2i-1}} (s_j - \hat{s}_j) \right)^2 \\ &+ \sum_{i=1}^{2p} r_{2i,2i}^2 \left( s_{2i} - \hat{s}_{2i} + \sum_{j=2i+1}^{2p} \frac{r_{2i,j}}{r_{2i,2i}} (s_j - \hat{s}_j) \right)^2 \end{aligned} \tag{5.5}$$

Expanding the summations on the right-hand side of the inequality (5.5) and considering only the first term in it, we can state an obvious necessary condition for $s$ to belong to the sphere,

$$r_{2p,2p}^2 (s_{2p} - \hat{s}_{2p}) + \left[ r_{2p-1,2p-1}^2 (s_{2p-1} - \hat{s}_{2p-1}) + \frac{r_{2p-1,2p}}{r_{2p-1,2p-1}} (s_{2p} - \hat{s}_{2p}) \right]^2 \leq r'^2 \tag{5.6}$$

The terms on the LHS of (5.6) comprise the part of the summation in (5.5) which depends on the bit $b_p$ only. Therefore, condition (5.6) need to be tested for every $b_p \in GF(L)$. When, for some $b_p$, the inequality (5.6) is satisfied, that particular $b_p$ value is substituted for in (5.5). Then by considering the second terms (i.e., the terms for $i = 2$) in the expanded summations (5.5), one can state a (stronger) condition that $b_{p-1}$ (assuming the already fixed value of $b_p$) needs to satisfy in order that the

point $s$ belongs to the sphere,

$$r^2_{2p-3,2p-3}\left(s_{2p-3}-\hat{s}_{2p-3}+\sum_{j=2p-2}^{2p}\frac{r_{2p-3,j}}{r_{2p-3,2p-3}}(s_j-\hat{s}_j)\right)^2 + r^2_{2p-2,2p-2}\left(s_{2p-2}-\hat{s}_{2p-2}+\sum_{j=2p-1}^{2p}\frac{r_{2p-2,j}}{r_{2p-2,2p-2}}(s_j-\hat{s}_j)\right)^2 \leq r''^2,$$

where $r''^2$ is evaluated for the previously chosen $b_p$,

$$r''^2 = r'^2 - r^2_{2p,2p}(s_{2p}-\hat{s}_{2p}) + \left[r^2_{2p-1,2p-1}(s_{2p-1}-\hat{s}_{2p-1}) + \frac{r_{2p-1,2p}}{r_{2p-1,2p-1}}(s_{2p}-\hat{s}_{2p})\right]^2.$$

When such $b_{p-1}$ is found, it is fixed and substituted for in (5.5). If no such $b_{p-1}$ is found, we need to take one step back, discard the previously chosen higher-indexed bit (in this case, $b_p$), chose another one and proceed likewise. By continuing in the same way to state the conditions on the remaining bits $(b_{p-2},\ldots,b_1)$, we define the total of $p$ nested necessary conditions from which all components of the vector $\mathbf{b}$ may consecutively be found.

One can think of the JDD algorithm performing the search on a tree (which, for the special case that we considered up to this point, would be the binary tree), as illustrated in Figure 5.2. The maximum depth of the tree is $p$. The conditions that might result in the pruning of the tree are tested with respect to the integer lattice, which is generated by the matrix $H$. Whenever a point falls outside the sphere, which is centered at the received point $x$ in the Euclidean space containing the lattice, the current node in the tree is discarded.

Every node at any level of the tree corresponds to some point in $GF(L)$. Therefore, the paths on the tree that survive the pruning, correspond to the information vectors. The lattice points in the Euclidean space are related to the tree via both the block code $G$ and the (lattice generating) channel matrix $H$. Not all the points in the lattice are allowed (as is illustrated by the "empty" points in Figure 5.2). This is clear since the code maps the space with $L^p$ elements (the information vector space)

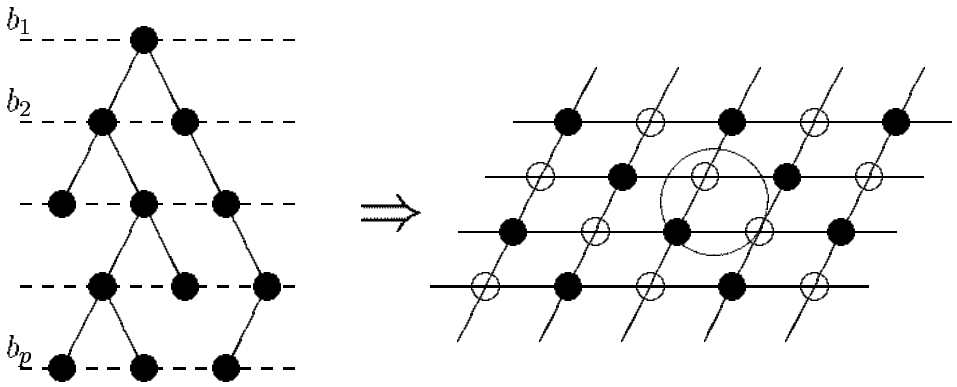


Figure 5.2: The tree-pruning interpretation of the JDD-ML algorithm

to the lattice with $L^m$ points (the symbol space).

The description of the JDD-ML algorithm based on (5.5) assumed a rate $R_c = 1/2$ binary block code with a regular profile of the generator matrix $G$. The algorithm can be generalized to accommodate for an arbitrary diagonal profile of the arbitrary rate code generator matrix $G$. To this end, we will find it useful to express the condition (5.5) (still specialized for 1/2 rate code with regular diagonal profile) in a "matrix form", that is, write it as

$$\sum_{j=1}^{p} \left\| R_{jj} \begin{pmatrix} s_{2j-1} - \hat{s}_{2j-1} \\ s_{2j} - \hat{s}_{2j} \end{pmatrix} + \sum_{k=j+1}^{p} R_{jk} \begin{pmatrix} s_{2k-1} - \hat{s}_{2k-1} \\ s_{2k} - \hat{s}_{2k} \end{pmatrix} \right\|^2 \leq r^2, \qquad (5.7)$$

where

$$R_{jj} = R(2j-1:2j; 2j-1:2j), \qquad R_{jk} = R(2j-1:2j; 2k-1:2k).$$

To state the JDD-ML algorithm in a matrix form similar to (5.7) but for the general structure of $G$, it will be convenient to denote

$$d_i^{(h)} = \sum_{j=1}^{i} g_j^{(h)}, \quad \text{and} \quad d_i^{(v)} = \sum_{j=1}^{i} g_j^{(v)}, \quad 1 \leq i \leq D.$$

The $d_i^{(h)}$ and $d_i^{(h)}$, $i = 1, 2, \ldots D$ are illustrated in Figure 5.3.

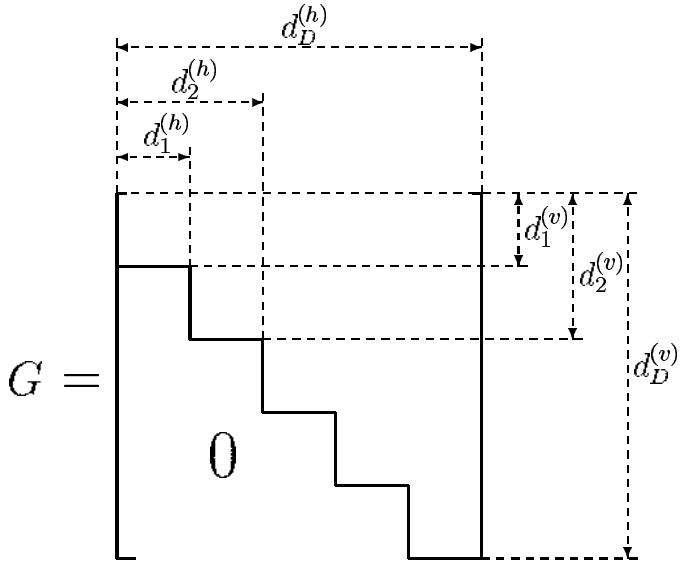


Figure 5.3: *Generator matrix G*

Now, the condition (5.7) can be generalized to the case of a code with any rate and any diagonal profile as

$$\sum_{j=1}^{D} \left\| R_{jj} \left( \mathbf{s}_j - \hat{\mathbf{s}}_j \right) + \sum_{k=j+1}^{D} R_{jk} \left( \mathbf{s}_k - \hat{\mathbf{s}}_k \right) \right\|^2 \leq r^2, \tag{5.8}$$

where

$$R_{jk} = R(d_{j-1}^{(h)} + 1 : d_j^{(h)}; d_{j-1}^{(v)} + 1 : d_j^{(v)}),$$

and where

$$\mathbf{s}_j = \left[ s_{d_{j-1}^{(h)}} \ldots s_{d_j^{(h)}} \right]^T \quad \hat{\mathbf{s}}_j = \left[ \hat{s}_{d_j^{(h)}} \ldots \hat{s}_{d_{j-1}^{(h)}} \right]^T,$$

$j = 1, 2, \ldots, D$, $j < k \leq D$.

For (5.8) to hold, the entry $b_p$ in the information vector needs to satisfy

$$\| R_{DD} (\mathbf{s}_D - \hat{\mathbf{s}}_D) \|^2 \leq r^2 \tag{5.9}$$

For every $b_p \in GF(L)$ which satisfies condition (5.16), we go back to (5.8) and substitute in that particular $b_p$. Then a new, more strict necessary condition on $b_{p-1}$ (and already chosen $b_p$) can be stated as

$$\sum_{j=1}^{D} \left\| R_{jj} \left( \mathbf{s}_j - \hat{\mathbf{s}}_j \right) + \sum_{k=j+1}^{D} R_{jk} \left( \mathbf{s}_k - \hat{\mathbf{s}}_k \right) \right\|^2 \leq r'^2,$$

where

$$r'^2 = r^2 - \| R_{DD} (\mathbf{s}_d(b_p) - \hat{\mathbf{s}}_D) \|^2$$

is the updated radius.

The procedure is continued until all the components of the information vector $\mathbf{b}$ that satisfy (5.8) are found. If no vectors $\mathbf{b}$ that satisfy (5.8) are found, the radius $r$ is increased and the algorithm is restarted. On the other hand, in general, there may be more than one information vector $\mathbf{b}$ found by the algorithm. The one that minimizes (5.3) is the solution to the joint ML detection and decoding problem.

The JDD-ML algorithm can be summarized as follows:

1. *Input:* $G$, $R$, $\mathbf{x}$, $\hat{\mathbf{s}}$, $r$.

2. Set $k = p$, $r_m'^2 = r^2 - \|x\|^2 + \|H\hat{\mathbf{s}}\|^2$.

3. Set $b_k = -1$.

4. $b_k = b_k + 1$; if $b_k > L$, go to 6.

5. (Increase $k$) $k = k + 1$; if $k = p + 1$, terminate algorithm, else go to 3.

6. (Decrease $k$) If $k = 1$ go to 6. Else $k = k - 1$, and calculate

$$\mathbf{s}_k = \sum_{j=k}^{p} \left[ \mathrm{mod} \left( G \left( d_k^{(h)} + 1 : d_{k+1}^{(h)}, j \right) \otimes b_k, L \right) - \frac{L-1}{2} \cdot \mathbf{1}_{g_k^{(v)}} \right],$$

where $\mathbf{1}_i$ denotes an $i$-dimensional vector with all entries 1. Also, calculate

$$r'^2_k = r'^2_{k+1} - \left\| R_{kk}\left(\mathbf{s}_k - \hat{\mathbf{s}}_k\right) + \sum_{j=k+1}^{D} R_{kj}\left(\mathbf{s}_j - \hat{\mathbf{s}}_j\right) \right\|^2,$$

and go to 2.

7. Solution found. Save $\mathbf{b}$ and go to 3.

## 5.2 The Computational Complexity of JDD-ML

The choice of the radius in (5.5) of the sphere that we search for the ML optimal information vector $\mathbf{b}$ is a function of the SNR. The number of the lattice points that lie inside the sphere depends on the particular instantiation of both the channel matrix $H$ and the noise vector $v$. On the other hand, the complexity of the JDD-ML algorithm depends on the number of points that the algorithm examines. Since the number of these points varies from one channel use to another, the complexity is a random variable. A way to characterize it is by means of its moments. In this section, we derive the closed form analytical expression for the expected complexity of the JDD-ML algorithm.

Let $s_t = [G\mathbf{b}_t]$ denotes the lattice point obtained by encoding the information vector $\mathbf{b}_t$ and modulating it. Furthermore, let $s_a$ denotes a lattice point obtained by encoding and modulating an arbitrary vector $\mathbf{b}_a$. The expected complexity of the JDD-ML algorithm is proportional to the expected number of the lattice points inside the sphere. Following the outline of the computation of the expected complexity for the original sphere decoder in Chapter 2, we shall need two elements in order to calculate the expected number of the lattice points inside the sphere:

(a) A probabilistic description of the event that both $s_t$ and $s_a$ belong to the sphere of radius $r$ around $x$.

(b) A method for enumerating information vectors space, i.e., a way of counting all the information vectors for which the probability in (a) takes the same value.

We start by calculating the probability in (a) above. Suppose that the vector $\mathbf{b}_t$ was encoded and modulated onto $s_t$, that the lattice point $s_t$ was transmitted and that the vector $\mathbf{x} = Hs_t + \mathbf{v}$ was observed. The probability that another lattice point $s_a$ lies in a hypersphere of radius $r$ around $\mathbf{x}$ can be expressed as

$$\gamma\left(\frac{r^2}{2(\sigma^2 + \|s_a - s_t\|^2)}, \frac{m}{2}\right) = \int_0^{\frac{r^2}{2(\sigma^2+\|s_a-s_t\|^2)}} \frac{\lambda^{m/2-1}}{\Gamma(m/2)} e^{-\lambda} d\lambda.$$

This probability is only a function of the Euclidean distance between $s_t$ and $s_a$, $\|s_a - s_t\|^2$. Therefore, all we need now is to count the number of the points $s_a$ which are both admissible codewords and lie inside a sphere of radius $r$ around $Hs_t$. We count this number of points by enumerating the information vector space. In particular, we want to enumerate

$$\|s_a - s_t\|^2 = \|[G \otimes b_a] - [G \otimes b_t]\|^2, \tag{5.10}$$

where $[\cdot]$ denotes modulation operation.

We will now demonstrate the enumeration procedure for $L = 2$, i.e., for the case of the binary block codes. That is, we consider all the possible values that the Euclidean distance between $s_a$ and $s_t$ can take depending on the pairs of the binary information vectors $(\mathbf{b}_t, \mathbf{b}_a)$. We shall also, in the following discussion, assume the regular diagonal profile of the generator matrix $G$. The generalization will follow directly and will be stated without discussion. With the previously mentioned assumptions, the following observations can be made:

- If $\mathbf{b}_a = \mathbf{b}_t$, then $\|s_a - s_t\|^2 = 0$.
- Assume that $\mathbf{b}_a$ differs from $\mathbf{b}_t$ in only one entry (that is, they differ in only one of the $p$ possible spots), and note that

$$\|s_t - s_a\|^2 = (s_{t,1} - s_{a,1})^2 + \ldots + (s_{t,m} - s_{a,m})^2. \tag{5.11}$$

  Then if $\mathbf{b}_t$ and $\mathbf{b}_a$ have the $i^{th}$ bit different, there will be $w_i$ non-zero terms in the sum on the RHS of (5.11), where $w_i$ $(i = 1, \ldots, p)$ denotes the weight of

the columns of $G$, i.e., $w_i$ denotes the sum of all entries in the $i^{th}$ column of $G$. Since $L = 2$, each such non-zero entry is 1. Therefore,

$$||s_a - s_t||^2 \in \{w_1, w_2, \ldots, w_p\}.$$

Therefore, in this case $||s_a - s_t||^2$ may take at most $p$ different values – it takes less when some columns of $G$ have the same weights.

- Assume that $\mathbf{b}_a$ differs from $\mathbf{b}_t$ in two positions, say, $j$ and $k$ (for which there are $\binom{p}{2}$ possible cases). From (5.10), it is clear that if $G(i,j) = G(i,k)$, then the $i^{th}$ component of the symbol vector $s_t$ and $s_a$ will be different. However, if $G(i,j) \neq G(i,k)$, the $i^{th}$ component will be the same for both $s_t$ and $s_a$. Therefore, to enumerate all the possible values of $||s_a - s_t||^2$, one needs to consider exclusive-or sums of any two columns of $G$. Consider the vectors that result by taking all such exclusive-or sums. Let $\Upsilon(2)$ denotes the set of distinct weights (sum of the entries) of such vectors. Clearly, in this case,

$$||s_a - s_t||^2 \in \Upsilon(2).$$

- We proceed alike for the cases when $\mathbf{b}_a$ differs from $\mathbf{b}_t$ in more than two entries. In general, if $\mathbf{b}_a$ and $\mathbf{b}_t$ differ in $k$ entries, we consider set of the vectors obtained by taking exclusive-or sums of all possible combinations of $k$ columns of $G$ (there are $\binom{p}{2}$ such combinations). Let $\Upsilon(k)$ denotes the set of distinct weights of such vectors. Then

$$||s_a - s_t||^2 \in \Upsilon(k).$$

The enumeration procedure we just described can obviously become rather involving as the dimension of the problem (i.e., $p$) increases. However, rather than counting for a specific $G$, we can average over all generator matrices $G$ of a given diagonal profile. For a large dimension $p$ (which, in fact, is the computationally difficult case),

the complexity of the algorithm for a random $G$ (with a given diagonal profile) will be represented well by the complexity of the algorithm averaged over all $G$'s with the same diagonal profile.

Assume the upper triangular form of a generator matrix $G$ as illustrated in Figure 5.1, and that the entries in a non-zero part of $G$ (that is, the entries in the upper-triangular part of $G$) are drawn from a Bernoulli($\frac{1}{2}$) distribution. Let $w_1, w_2, \ldots, w_p$ denote the *average* weights of the columns of $G$ (averaged over all possible generator matrices $G$ with entries randomly drawn from Bernoulli ($\frac{1}{2}$) and with a given profile), and note that

$$w_1 < w_2 < \ldots < w_p.$$

We have previously described the enumeration of the vector pairs $(\mathbf{b}_t, \mathbf{b}_a)$ for the fixed $G$. We shall now follow the same enumeration procedure but average over random $G$'s. Before proceeding any further, we will find useful to recall the fact that the distribution of exclusive-or sum of any number of Bernoulli($\frac{1}{2}$) variables remains Bernoulli($\frac{1}{2}$). Thus we can make the following observations:

- There are $\begin{pmatrix} p-1 \\ k-1 \end{pmatrix}$ combinations of $k$ ($k = 1, 2, \ldots, p$) columns of $G$ where the weight of the "heaviest" column in the combination is $w_p$ (i.e., there are $\begin{pmatrix} p-1 \\ k-1 \end{pmatrix}$ combinations of $k = 1, 2, \ldots, p$ columns which include column $p$). Hence there is a total of

$$\begin{pmatrix} p-1 \\ 0 \end{pmatrix} + \begin{pmatrix} p-1 \\ 1 \end{pmatrix} + \ldots + \begin{pmatrix} p-1 \\ p-1 \end{pmatrix} = 2^{p-1}$$

  possible combinations of columns whose exclusive-or sum results in a vector with weight $w_p$.

- There are $\begin{pmatrix} p-2 \\ k-1 \end{pmatrix}$ combinations of $k$ columns ($k = 1, 2, \ldots, p-1$) of $G$ where the weight of the "heaviest" column in the combination is $w_{p-1}$ (i.e., there are

$\begin{pmatrix} p-2 \\ k-1 \end{pmatrix}$ combinations of $k = 1, 2, \ldots, p-1$ columns which include column $p-1$ but do not include column $p$). Thus there is a total of

$$\begin{pmatrix} p-2 \\ 0 \end{pmatrix} + \begin{pmatrix} p-2 \\ 1 \end{pmatrix} + \ldots + \begin{pmatrix} p-2 \\ p-2 \end{pmatrix} = 2^{p-2}$$

combinations of the columns whose exclusive-or sum results in a vector with weight $w_{p-1}$.

- In a similar way we determine number of possible combinations of columns whose exclusive-or sum results in vectors of weights $w_{p-2}, w_{p-3}, \ldots, w_1$.

In summary, we identify the following generating polynomial:

$$\Psi(x) = 1 + \sum_{k=1}^{p} 2^{p-k} x^{w_{p-k+1}}.$$

The generating polynomial $\Psi(x)$ should be interpreted as follows: each coefficient $\psi_k$ in the expansion $\Psi(x) = \sum_{k=0}^{p} \psi_k x^{w_k}$ is the number of pairs of information vectors $(\mathbf{b}_t, \mathbf{b}_a)$ for which $\|s_t - s_a\|^2 = w_k$.

Remind that, in the previous discussion, we assumed the regular diagonal profile of $G$. For the codes with general profile (still assuming the binary codes), the generating polynomial takes the form

$$\Psi(x) = 1 + \sum_{k=1}^{D} \sum_{j=1}^{g_k^{(h)}} 2^{p-k-j+1} x^{w_{p-k+1}}. \tag{5.12}$$

The inner summation in (5.12) is due to possible multiple columns with the same average weights. For instance, assume that there are two "heaviest" columns, i.e., two columns with the weight $w_D$. Then (5.12) means that there is a total of $2^{p-1} + 2^{p-2}$ combinations of the columns of $G$ whose exclusive-or sum results in a vector with weight $w_D$.

We can summarize the complexity results in the following theorem:

**Theorem 3 [Expected complexity of JDD-ML algorithm for binary codes]**

*Consider the model*

$$\mathbf{x} = Hs + \mathbf{v},$$

*where* $v \in \mathcal{R}^{n\times 1}$ *is comprised of i.i.d.* $\mathcal{N}(0,1)$ *entries,* $H \in \mathcal{R}^{n\times m}$ *is comprised of i.i.d.* $\mathcal{N}(0,\rho/m)$ *entries, and* $s \in \mathcal{D}_L^m$ *is an* $m$*-dimensional vector whose entries are obtained by modulating coded vector* $\mathbf{c}$ *onto an* $L$*-PAM constellation. Furthermore, the vector* $\mathbf{c}$ *is obtained by block coding information vector* $\mathbf{b}$*,*

$$\mathbf{c} = G \otimes \mathbf{b},$$

*where* $G \in GF(2)^{m\times p}$*. Then the expected complexity of the JDD-ML algorithm for the integer least-squares problem*

$$\min_{\mathbf{b}\in GF(L)^p} ||\mathbf{x} - Hs||^2,$$

*averaged over* $H$*,* $G$*, and* $\mathbf{v}$*, for a 2-PAM constellation is*

$$C(m,\rho) = \sum_{k=1}^{m}(2k+17)\sum_{q=0}^{k} g(q)\gamma\left(\frac{r^2}{2(\sigma^2+q)}, \frac{m}{2}\right) \tag{5.13}$$

*where* $g(q)$ *is the coefficient of* $x^q$ *in the polynomial*

$$\Psi(x) = 1 + \sum_{k=1}^{D}\sum_{j=1}^{g_k^{(h)}} 2^{p-k-j+1} x^{w_{p-k+1}}.$$

**Proof:**

The proof follows from the previous discussion in this section.

■

## 5.3 The JDD-MAP Algorithm and its Complexity

The joint ML detection and decoding problem (5.3) assumed no prior knowledge about the information vector $\mathbf{b}$. There are scenarios, however, when we may have the access to the a priori information, that is, when we know the set of the a priori probabilities

$$\{p(b_1), p(b_2), \ldots, p(b_p)\}.$$

This a priori information may be exploited in order to obtain the maximum a posteriori estimate of the information vector $\mathbf{b}$. The joint MAP detection and decoding problem may be stated as

$$\min_{\mathbf{b}\in GF(L)^p} \left[ \|x - Hs\|^2 - \sum_{i=1}^{p} \log p(b_i) \right]. \tag{5.14}$$

To solve (5.14), we take the same approach as we did for the joint ML detection and decoding problem. In particular, we search for vectors $\mathbf{b}$ such that

$$\sum_{j=1}^{D} \left\| R_{jj}\left(\mathbf{s}_j - \hat{\mathbf{s}}_j\right) + \sum_{k=j+1}^{D} R_{jk}\left(\mathbf{s}_k - \hat{\mathbf{s}}_k\right) \right\|^2 \leq r^2 + \sum_{i=1}^{p} \log p(b_i), \tag{5.15}$$

where $R_{jk}$ are the blocks of the upper triangular matrix $R$ is the $QR$ decomposition of the channel matrix $H$,

$$R_{jk} = R(d_{j-1}^{(h)} + 1 : d_j^{(h)}; d_{j-1}^{(v)} + 1 : d_j^{(v)}),$$

and where

$$\mathbf{s}_j = \left[ s_{d_{j-1}^{(h)}} \ldots s_{d_j^{(h)}} \right]^T \quad \hat{\mathbf{s}}_j = \left[ \hat{s}_{d_j^{(h)}} \ldots \hat{s}_{d_{j-1}^{(h)}} \right]^T,$$

$j = 1, 2, \ldots, D$, $j < k \leq D$. From (5.15) it is clear that we search for the lattice points that no longer belong to the sphere but rather lie in some distorted object in the Euclidean space. This object can be thought of as the sphere stretched or compressed in various dimensions, depending on the a priori confidence that we have

about the corresponding components of the vector $\mathbf{b}$.

For (5.15) to hold, the entry $b_p$ in the information vector needs to satisfy

$$\|R_{DD}(\mathbf{s}_D - \hat{\mathbf{s}}_D)\|^2 \leq r^2 + \log p(b_p), \tag{5.16}$$

For every $b_p \in GF(L)$ which satisfies condition (5.16), we go back to (5.8) and substitute in that particular $b_p$. Then a new, more strict necessary condition on $b_{p-1}$ (and already chosen $b_p$) can be stated as

$$\sum_{j=1}^{D} \left\| R_{jj}\,(\mathbf{s}_j - \hat{\mathbf{s}}_j) + \sum_{k=j+1}^{D} R_{jk}\,(\mathbf{s}_k - \hat{\mathbf{s}}_k) \right\|^2 \leq r'^2 + \log p(b_{p-1}),$$

where

$$r'^2 = r^2 - \|R_{DD}(\mathbf{s}_d(b_p) - \hat{\mathbf{s}}_D)\|^2 - \log p(b_p)$$

is the updated radius.

The procedure is continued until all the components of the information vector $\mathbf{b}$ that satisfy (5.15) are found. If no vectors $\mathbf{b}$ that satisfy (5.15) are found, the radius $r$ needs to be increased and the algorithm starts again. Of course, in general, there may be more than one information vector $\mathbf{b}$ found by the algorithm. Then the one among them which minimizes (5.14) is the solution to the joint MAP detection and decoding problem.

We state the JDD-MAP algorithm as follows:

1. *Input:* $G$, $R$, $\mathbf{x}$, $\hat{\mathbf{s}}$, $\{\log p(b_1), \ldots, \log p(b_p)\}$, $r$.

2. Set $k = p$, $r_m'^2 = r^2 - \|x\|^2 + \|H\hat{\mathbf{s}}\|^2 + \log p(b_p)$,

3. Set $b_k = -1$.

4. $b_k = b_k + 1$; if $b_k > L$, go to 6.

5. (Increase $k$) $k = k + 1$; if $k = p + 1$, terminate algorithm, else go to 3.

6. (Decrease $k$) If $k = 1$ go to 6. Else $k = k - 1$, and calculate

$$\mathbf{s}_k = \sum_{j=k}^{p} \left[ \mathrm{mod}\left( G\left( d_k^{(h)} + 1 : d_{k+1}^{(h)}, j \right) \otimes b_k, L \right) - \frac{L-1}{2} \cdot \mathbf{1}_{g_k^{(v)}} \right],$$

where $\mathbf{1}_i$ denotes an $i$-dimensional vector with all entries 1. Also, calculate

$$r_k'^2 = r_{k+1}'^2 - \left\| R_{kk}\left(\mathbf{s}_k - \hat{\mathbf{s}}_k\right) + \sum_{j=k+1}^{D} R_{kj}\left(\mathbf{s}_j - \hat{\mathbf{s}}_j\right) \right\|^2 + \log p(b_k),$$

and go to 2.

7. Solution found. Save $\mathbf{b}$ and go to 3.

The computational complexity of the JDD-MAP relates to the computational complexity of the JDD-ML in the same way that the complexity of FP-MAP algorithm from Chapter 3 relates to the complexity of the original Fincke-Pohst algorithm. In particular, the probability that an arbitrary information vector $\mathbf{b}_a$ generates a lattice point $s_a$ inside the sphere around $Hs_t$, where $s_t$ is the transmitted symbol (generated by $\mathbf{b}_t$), is given by

$$p_{\mathbf{b}_a} = \gamma\left( \frac{r^2 + \sum_{i=1}^{p} \log p(b_i)}{2(\sigma^2 + \|s_a - s_t\|^2)}, \frac{m}{2} \right) = \int_0^{\frac{r^2 + \sum_{i=1}^{p} \log p(b_i)}{2(\sigma^2 + \|s_a - s_t\|^2)}} \frac{\lambda^{m/2-1}}{\Gamma(m/2)} e^{-\lambda} d\lambda.$$

Since $\sum_{j=1}^{m} \log p(b_j) \leq 0$, we have

$$\frac{r^2 + \sum_{i=1}^{p} \log p(b_i)}{2(\sigma^2 + \|s_a - s_t\|^2)} \leq \frac{r^2}{2(\sigma^2 + \|s_a - s_t\|^2)}.$$

Since the incomplete gamma function is monotonically increasing with its argument, it follows that for the same choice of $r$,

$$p_{\mathbf{b}_a}^{JDD-MAP} \leq p_{\mathbf{b}_a}^{JDD-ML},$$

and we may conclude the following:

**Lemma 2** *For the same choice of the search parameter $r$, the expected complexity of the JDD-MAP algorithm is upper bounded by the expected complexity of the JDD-ML algorithm.*

■

The JD-MAP algorithm is particularly promising for implementation in the communication schemes employing concatenated coding (with a block inner code) and iterative decoding.

## 5.4 Performance Simulations

We illustrate the calculation of the expected computational complexity and the performance of the JDD-ML algorithm on a few simple examples. Figure 5.4 shows the expected complexity exponent for the rate $R = 1/2$ linear block binary code with $p = 5$ and $m = 10$ and with the regular diagonal profile of the generator matrix $G$. The complexity exponent is defined as

$$e = \log_m C,$$

where $C$ represents the total flop count in (5.13).

The analytically calculated expected complexity, plotted as a dashed line in Figure 5.4, coincides with the empirically obtained complexity. Moreover, the expected complexity is roughly cubic over the considered range of SNRs.

Furthermore, we consider the performance of the JDD-ML algorithm on the system employing Golay 24 code. In Figure 5.5, we compare the performance of the JDD-ML algorithm with that of the receiver which employs hard detection followed by the hard decoding. It is evident that the performance improvement is significant over a wide range of SNRs.

In Figure 5.6, the expected complexity exponent of the JDD-ML algorithm for Golay 24 code is shown. In the SNR region where the BER is low, the expected

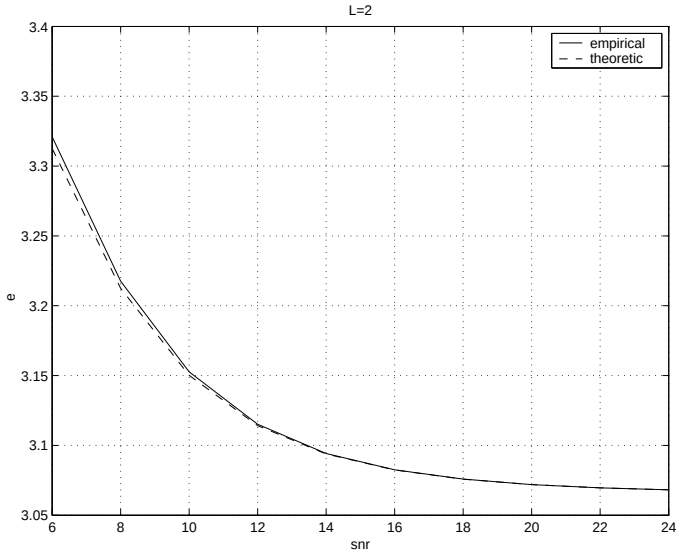


Figure 5.4: *An illustration of the expected complexity of the JDD-ML algorithm: a rate* $1/2$ *binary code,* $m = 10$

complexity of the algorithm is $e \approx 4$ and it decreases with the SNR. The complexity exponent is relatively high, which is due to the fact that the Golay 24 code can not be transformed into a form that allows for low complexity of JDD-ML; loosely speaking, the diagonal profile of the Golay 24 generator matrix is highly irregular.

## 5.5 Conclusion

In this chapter, we considered the problem of the joint detection and decoding in multi-input multi-output systems. The target criteria that we focused on are maximum-likelihood and maximum a priori. Due to the potentially rather high complexity of the joint solution, the two aforementioned problems – detection and decoding – are commonly considered separately in practice. However, the performance price that then has to be paid justifies studying the algorithms that treat the problems jointly. To this end, drawing on the ideas encountered in solving the standard integer least-squares problems (in particular, the sphere decoding idea), we developed

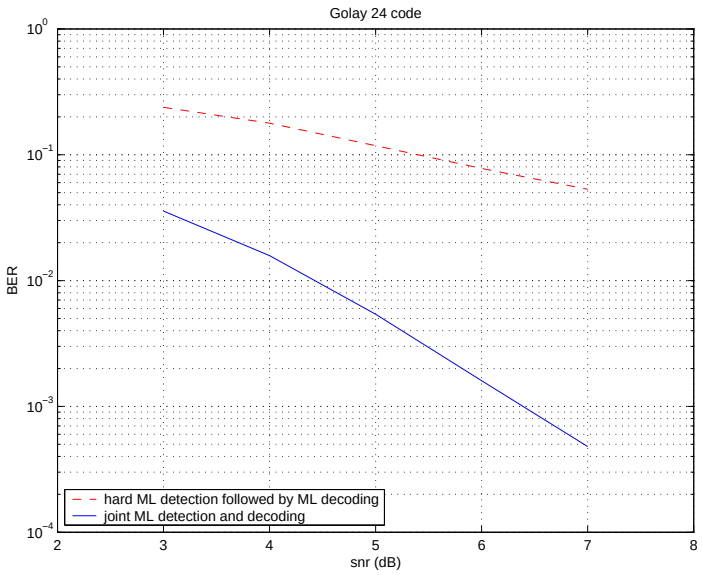


Figure 5.5: *BER performance of the JDD-ML for Golay 24 code compared to the hard detection followed by the hard decoding.*

algorithms that solve both the joint ML and the joint MAP detection and decoding problems. In particular, we studied the joint ML and MAP detection and decoding problems in the systems where the data is encoded by the linear block error-correcting codes and then transmitted across the matrix channel in AWGN. The algorithm that we developed solves the integer least-squares problem where the only allowed lattice points are those generated by a linear block transform of the elements in a finite field. The algorithm performs a search for the lattice points inside a sphere but added an important constraint to the search: the admissible lattice must be a valid codeword.

Due to the search strategy, the computational complexity of the algorithm is a random variable. We quantified it by means of the expected complexity, which we found analytically, in a closed form. The expected complexity of the joint maximum-likelihood detection and decoding (JDD-ML) algorithm was shown to be polynomial in the length of the uncoded information word over a wide range of SNRs. Furthermore, we considered the maximum a posteriori joint detection and decoding problem for the case when the a priori information for the information data are provided to the

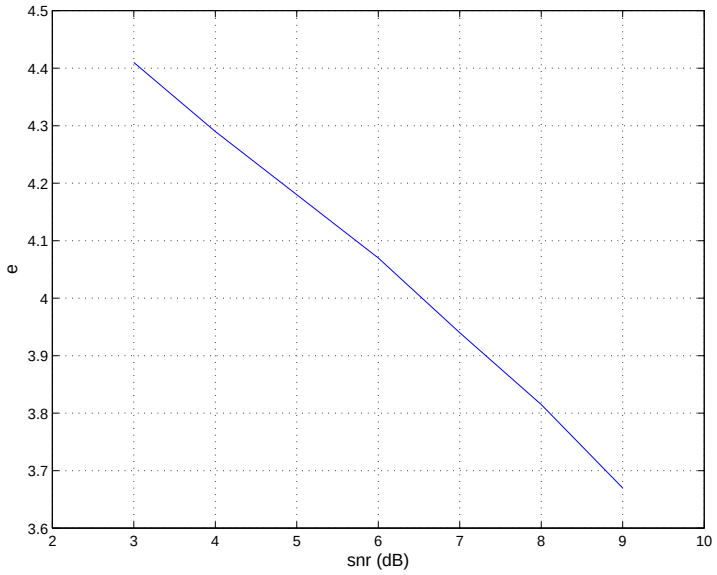


Figure 5.6: *Complexity exponent of the JDD-ML for Golay 24 code.*

receiver. We derived the JDD-MAP algorithm for solving it and derived the bound on the expected computational complexity of the JDD-MAP algorithm.

# Chapter 6

# Conclusion

We considered integer least-squares and related problems that are traditionally thought to be computationally intractable. Geometrically, the integer least-square problem is equivalent to the closest-point search in a lattice and is, indeed, NP-hard. However, we focused on the case where the given point in a space is not random but is a lattice point perturbed by the noise whose statistical properties are known. Furthermore, we limited our study to the so-called sphere decoding algorithm, which finds the closest point inside a sphere of a radius chosen according to the aforementioned statistics of the noise. We did not dwell on the worst-case but considered the expected complexity of the algorithm. A closed-form expression for the expected complexity of the sphere decoding algorithm is derived in Chapter 2. It was shown that the expected complexity of the algorithm is polynomial over a wide range of signal-to-noise ratios and, in fact, often cubic or sub-cubic. The maximum-likelihood detection often amounts to the closest-point in a lattice. Since the ML detection is considered intractable, heuristic techniques, which typically have cubic complexity, are often employed. The results in this thesis imply that the ML detection can be implemented at the expected complexity comparable to that of the heuristics techniques while significantly outperforming even the best among them.

Furthermore, for channels with memory, we have derived "hybrid" algorithms that combine the dynamic programming principles of the Viterbi algorithm with

the sphere-constrained search strategy of the sphere decoding. For instance, the Viterbi algorithm idea can be exploited to improve the complexity of the sphere constrained search in a lattice by tracking the costs associated with the channel memory state. Alternatively, the computational burden of the maximum-likelihood detection on trellis via the Viterbi algorithm can be alleviated by imposing a sphere constraint on the trellis paths. Such an algorithm has the worst-case complexity of the Viterbi algorithm but is significantly faster on the average.

A modification of the Fincke-Pohst algorithm which performs the maximum a posteriori search was proposed in Chapter 4. This algorithm, the FP-MAP, performs a search in the vicinity of the received point. However, the search is not constrained within a sphere but within a region defined by both the statistics of the noise and by the a priori information about the coordinates of the lattice points. The search, in general, returns a number of nearby points which are then used to form the soft probabilistic information required for the decoding in soft iterative schemes. It was argued that the complexity of the FP-MAP algorithm can be bounded by the complexity of the related sphere decoding algorithm. The performance of the algorithm was illustrated in the systems employing both simple convolutional as well as the powerfull turbo and LDPC error-correcting codes. When combined with the powerful channel codes, the algorithm results in the near-capacity system performance.

Finally, we considered the problem of the joint detection and decoding in multi-input multi-output systems. Extending the sphere-constrained idea of the Fincke-Pohst algorithm, we developed algorithms that solve the joint problem for both the maximum-likelihood and the maximum a posteriori criteria when a linear block code is used. The algorithms still perform a search within some constrained volume in the space. However, an important additional constraint is imposed on the lattice points – they must be admissable codewords. The expected computational complexity of the joint detection and decoding algorithms was found analytically for the case of binary error-correcting block codes. It was demonstrated that this complexity is polynomial and often of the cubic order.

## 6.1 Future Research

The discussion in Chapter 2 rises some fundamental questions about the complexity of the optimal detection algorithms. We have observed that when the channel is capable of supporting the data transmission at a particular rate, the corresponding expected complexity of the sphere decoding algorithm is polynomial. Thus there is merit in studying the relation of the computational complexity of sphere decoding (and other algorithms) and the basic system parameters. Finding such a connection would be rather illuminating and would contribute to the understanding of the limits of performance at tractable receiver complexities.

With the statistical choice of the search radius, the sphere decoding algorithm finds the exact solution to the maximum-likelihood detection problem with a chosen probability. We have shown that the expected complexity of the algorithm is polynomial over a wide range of SNRs and moderate sizes of the transmitted symbol vectors. It is of interest, however, to develop the sphere-constrained search techniques capable of solving the maximum-likelihood detection problem for long symbol vectors. To this end, the requirement that the solution has to be exact may be alleviated in order to keep the computational complexity of the algorithm tractable. The probability of error (as a function of the search parameters) would concisely express the complexity/performance tradeoff of such an algorithm.

In Chapter 5, we presented an algorithm that solves the joint maximum-likelihood detection and decoding problem for linear block codes and multi-input multi-output channels. Study of an extension to other error-correcting schemes appears promissing. So does the study of an extension to the joint detection and decoding over channels with memory and relating the current algorithm to the material presented in Chapter 3.

Finally, we should point out that the sphere decoding algorithm is a standard, well-known technique for solving integer least-squares problems. It is the availability of the well-defined statistical models of the detection problem in digital data communications that has made our expected complexity analysis of the algorithm possible.

Therefore, it would be of great interest to revisit problems in, e.g., graph theory, operation research, computational biology, etc., that are NP-hard but for which statistical models exist or can be built. Studying the complexity of the structured exact algorithms in such problems could lead to analysis and results comparable to those presented in this thesis.

# Appendix A

# Probability of Finding a Point Inside a Sphere

## A.1 Full Channel Matrix

Suppose that the lattice point $s_t$ was transmitted and that the vector $x = Hs_t + v$ was observed. Then an arbitrary point $Hs_a$ belongs to the hypersphere of radius $r$ around $x$ if

$$\eta = r^2 - \|x - Hs_a\|^2 = r^2 - \|v + H(s_t - s_a)\|^2 > 0.$$

Therefore, probability that the point $Hs_a$ belongs to the hypersphere of radius $r$ around $x$ is

$$P(\eta > 0) = \int_0^\infty p(\eta) = \int_0^\infty \left[\frac{1}{2\pi}\int_{-\infty}^{\infty} \Phi(\omega)e^{-j\omega\eta}d\omega\right] d\eta, \tag{A.1}$$

where

$$\Phi(\omega) = Ee^{j\omega\eta}$$

is the characteristic function of $\eta$. Define a $n \times mn$ matrix $\mathcal{S}$ as

$$\mathcal{S} = (s_1 I \; s_2 I \;\; \ldots \;\; s_m I),$$

where $I$ denotes $n \times n$ identity matrix. Furthermore, define a $nm \times 1$ vector $h$ as

$$h = \text{vec}(H),$$

i.e., $h$ is a vector of the stacked columns of $H$. Then we can write

$$\eta = r^2 - \left\| \begin{bmatrix} I & \Lambda \end{bmatrix} \begin{bmatrix} v \\ h \end{bmatrix} \right\|^2,$$

where

$$\Lambda = \mathcal{S}_t - \mathcal{S}_a.$$

The characteristic function can now be written as

$$\Phi(\omega) = \xi \int d\mathbf{h} d\mathbf{v} e^{j\omega r^2} e^{-\begin{bmatrix} \mathbf{v}^* & \mathbf{h}^* \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma^2} I_m + j\omega I_m & +j\omega\Lambda \\ j\omega\Lambda^* & I_{m^2} + j\omega\Lambda^*\Lambda \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \mathbf{h} \end{bmatrix}}, \quad \text{(A.2)}$$

where

$$\frac{e^{-j\omega r^2}}{\pi^{2m(m+1)}\sigma^{2m}}.$$

Evaluating (A.2) and taking the inverse Fourier transform gives

$$p(\eta) = \begin{cases} \frac{1}{(m-1)!} \frac{\sigma^{2m}}{(||s_t - s_a||^2 + \sigma^2)^m} (\eta - r^2)^{m-1} e^{\frac{\eta - r^2}{\sigma^2 + ||s_t - s_a||^2}} & , \eta \leq r^2 \\ 0 & , \eta > r^2 \end{cases}$$

Substituting $p(\eta)$ in (A.1) yields (2.12), i.e.,

$$P(\eta > 0) = \gamma\left(\frac{r^2}{\sigma^2 + ||s_a - s_t||^2}, \frac{k}{2}\right) = \int_0^{\frac{r^2}{\sigma^2 + ||s_a - s_t||^2}} \frac{\lambda^{k/2-1}}{\Gamma(k/2)} e^{-\lambda} d\lambda,$$

where $\gamma(p, q)$ denotes an incomplete gamma function of argument $p$ and $q$ degrees of freedom.

## A.2 Toeplitz Channel Matrix

Denote

$$\mathcal{S} = \begin{bmatrix} s_l & \cdots & & s_1 \\ \vdots & \vdots & \ddots & \vdots \\ s_T & \cdots & & s_{T-l+1} \\ & s_T & & \vdots \\ & & \ddots & s_{T-1} \\ & & & s_T \end{bmatrix}_{T \times l}$$

Following the outline in Appendix A.1, we obtain the characteristic function

$$\Phi(\omega) = \xi \int d\mathbf{h} d\mathbf{v} e^{j\omega r^2} e^{-\begin{bmatrix} \mathbf{v}^* & \mathbf{h}^* \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma^2} I_{T+l-1} + j\omega I_{T+l-1} & j\omega\Lambda \\ j\omega\Lambda^* & I_l + j\omega\Lambda^*\Lambda \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \mathbf{h} \end{bmatrix}},$$

where

$$\xi = \frac{1}{\pi^{T+2l-1}\sigma^{2(T+l-1)}},$$

and

$$\Lambda = \mathcal{S}_t - \mathcal{S}_a.$$

Evaluating the above integral, we obtain (3.7),

$$\Phi(\omega) = \frac{e^{j\omega r^2}}{(1+j\omega\sigma^2)^{T-1} \prod_{k=1}^{l} [1 + j\omega(\sigma^2 + \rho_k)]},$$

where $\rho_k, k = 1, \ldots, l$ are the eigenvalues of the matrix $\Lambda^*\Lambda$.

# Bibliography


[1] C. E. Shannon, "A mathematical theory of communications," *Bell Sys. Tech. J.*, vol. 27, pp. 379-423 and 623-656, 1948.

[2] R. G. Gallager, "A simple derivation of the coding theorem and some applications," *IEEE Trans. on Information Theory*, vol. IT-11, pp. 3-18, January 1965.

[3] R. E. Van Dyck and D. J. Miller, "Transport of wireless video using separate, concatenated, and joint source-channel coding," *Proceedings of the IEEE*, vol. 87, no. 10, October 1999, pp. 1734-1750.

[4] J. M. Cioffi, *Lecture Notes on Digital Communications*, Stanford University, 2002.

[5] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Transactions on Information Theory*, vol. 28, pp. 55-67, January 1982.

[6] R. D. Wesel, *Trellis Code Design for Correlated Fading and Achievable Rates for Tomlinson-Harashima Precoding*, Ph.D. Thesis, Stanford University, 1996.

[7] W. C. Jakes, *Microwave Mobile Communications*, Piscataway, NJ: IEEE Press, 1993.

[8] T. Cover and J. Thomas, *Elements of Information Theory*, Wiley & Sons, New York, 1991.

[9] E. Bigillieri, J. Proakis, and S. Shamai, "Fading channels: information-theoretic and communications aspects," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2619-2692, October 1998.

[10] G, J. Foschini and M. J. Gans, "On limits of wireless communications in a fading environment when using multiple antennas," *Wireless Personal Communications*, vol. 6, pp. 311-335, 1998.

[11] A. Paulraj, R. Nabar, and D. Gore, *Introduction to Space-time Wireless Communications*, Cambridge University Press, to appear February 2003.

[12] I. E. Telatar, "Capacity of multi-antenna gaussian channels," *Bell Labs technical memorandum*, 1995.

[13] G. J. Foschini, "Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas," *Bell Labs. Tech. Journal*, vol. 1, no. 2, pp. 41-59, 1996.

[14] M. Grotschel, L. Lovasz, and A. Schriver, *Geometric Algorithms and Combinatorial Optimization*, Springer Verlag, 2nd ed., 1993.

[15] G. H. Golub and C. F. Van Loan, *Matrix Computations*, John Hopkins University Press, Baltimore, MD, $3^{rd}$ edition, 1996.

[16] B. Hassibi, "An efficient square-root algorithm for BLAST," *submitted to IEEE Transactions on Signal Processing*, 2000.

[17] P. H. Tan, L. K. Rasmussen and T. J. Lim, "The application of semidefinite programming for detection in CDMA," *IEEE Journal on Selected Areas in Communications*, vol. 19, pp. 1442-1449, August 2001.

[18] P. H. Tan, L. K. Rasmussen and T. J. Lim, "Constrained maximum-likelihood detection in CDMA," *IEEE Transactions on Communications*,vol. 49, pp. 142-153, January 2001.

[19] S. Boyd and L. Vandenberghe, "Advances in Convex Optimization: Theory, Algorithms, and Applications," *IEEE International Symposium on Information Theory*, Lausanne, 2002.

[20] S. Boyd and L. Vandenberghe, "Convex Optimization," *to be published 2003.* Available on-line http://www.stanford.edu/class/ee364/.

[21] H. Minkowski, "Diskontinuitsbereich fur arithmetic Aquivalenz," *J. reine Angew.*, 129, 220-224.

[22] A. Korkin and G. Zolotarev, "Sur les formes quadratiques," *Mathematische Annalen*, vol. 6, pp. 366-389, 1873.

[23] B. A. LaMacchia, *Basis Reduction Algorithms and Subset Sum Problems*, S.M. Thesis, MIT.

[24] S. S. Ryshkov and E. P. Baranovskii, "Classical methods in the theory of lattice packings," *Russian Mathematical Surveys*, vol. 34, no. 4, pp. 1-68, 1979.

[25] M. Ajtai, "The shortest vector problem in $L_2$ is NP-hard for randomized reductions," *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pp. 10-19, 1998.

[26] A. Banihashemi and A. Khandani, "On the complexity of decoding lattices using the Korkin-Zolotarev reduced basis," *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 162-171, 1998.

[27] C. Brutel and J. Boutros, "Euclidean space lattice decoding for joint detection in CDMA systems," *Proceedings of the 1999 IEEE Information Theory and Communications Workshop*, p. 129, 1999.

[28] E. Viterbo and J. Boutros, "A universal lattice code decoder for fading channels," *IEEE Trans. on Information Theory*, vol. 45, pp. 1639-1642, July 2000.

[29] M. O. Damen, A. Chkeif, and J. C. Belfiore, "Lattice codes decoder for space-time codes," *IEEE Communications Letters*, vol. 4, pp. 161-163, May 2000.

[30] B. Hassibi and B. Hochwald, "High-rate codes that are linear in space and time," *submitted to IEEE Trans. Info. Theory*, 2000. Download available at `http://mars.bell-labs.com`.

[31] A. Hassibi and S. Boyd, "Integer parameter estimation in linear models with applications to GPS," *IEEE Transactions on Signal Processing*, vol. 46, pp. 2938-2952, November 1998.

[32] M. Ajtai, "Generating hard instances of lattice problems," *Proc. 28th Annual ACS Symposium on the Theory of Computing*, 1996.

[33] O. Goldreich, S. Goldwasser, and S. Halevi, "Public-key cryptosystems from lattice reduction problems," *Advances in Cryptology – CRYPTO97. 17th Annual International Cryptography Conference*, pp. 112-131, 1997.

[34] R. Fischlin and J. Seifert, "Tensor-based trapdoors for cvp and their application to public key cryptography," *Cryptography and Coding. 7th IMA International Conference*, pp. 244-257, 1999.

[35] A. K. Lenstra, H. W. Lenstra, and L. Lovasz, "Factoring polynomials with rational coefficients," *Math. Ann.*, pp. 515-534, 1982.

[36] A. D. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithms," *IEEE Transactions on Information Theory*, vol. 13, pp. 260-269, April 1967.

[37] G. D. Forney, "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," *IEEE Transaction of Information Theory*, vol. 18, no. 3, pp. 363-378, May 1972.

[38] E. Agrell, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Transactions on Information Theory*, vol. 48, no. 8, August 2002.

[39] R. Kannan, "Improved algorithms on integer programming and related lattice problems," *Proc. 15th Annual ACM Symposium on Theory of Computing*, pp. 193-206, 1983.

[40] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Mathematics of Computation*, vol. 44, pp. 463-471, April 1985.

[41] J. Conway and N. Sloane, *Sphere Packings, Lattices and Graphs*, Springer-Verlag, 1993.

[42] G. Hardy, *Ramanujan: Twelve Lectures*, Chelsea Publishing, 1940.

[43] V. Tarokh V, N. Seshadri, and A. R. Calderbank, "Space-time Codes for High Data Rate Wireless Communication: Performance Criterion and Code Construction," IEEE Transactions on Information Theory, Vol. 44, No. 2, pp. 744-765, Mar. 1998.

[44] M. O. Damen, K. Abed-Meraim, and M. S. Lemdani, "Further results on the sphere decoder algorithm," *submitted to the IEEE Trans. on Information Theory*, 2000.

[45] C. P. Schnorr and M. Euchner, "Lattice basis reduction: improved practical algorithms and solving subset sum problems,: *Mathematical Programming*, vol. 66, pp. 181-191, 1994.

[46] J. P. Woodard and L. Hanzo, "Comparative study of turbo decoding techniques: an overview," *IEEE Trans. on Vehicular Technology*, vol. 49, no. 6, November 2000.

[47] R. G. Gallager, *Low-density parity-check codes*, Cambridge, MA: MIT Press, 1963.

[48] G. D. Forney, Jr., *Concatenated Codes*, Cambridge, MA: MIT Press, 1966.

[49] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding. Turbo codes," in *Proc. Int. Conf. Communications*, May 1993, pp. 1064-1070.

[50] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low-density parity-check codes," *Electron. Lett.*, vol. 32, pp. 1645-1646, 1996.

[51] M. C. Davey and D. J. C. MacKay, "Low-density parity-check codes over GF(q)," *IEEE Commun. Lett.*, vol. 2, pp. 165-167, June 1998.

[52] D. J. C. MacKay, "Good error correcting codes based on very sparce matrices," *IEEE Trans. Inform. Theory,* vol. 45, pp. 399-431, March 1999.

[53] S.-Y. Chung, G. D. Forney, Jr., T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Communications Letters*, vol. 5, pp. 58-60, February 2001.

[54] A. Stefanov and T. M. Duman, "Turbo-coded modulation for systems with transmit and receive antenna diversity over block fading channels: system model, decoding approaches, and practical considerations," *IEEE Journal on Sel. Areas in Commun.*, vol. 19, no. 5, May 2001.

[55] V. Tarokh, A. Naguib, N. Seshadri, and A. R. Calderbank, "Combined array processing and space-time coding," *IEEE Trans. Information Theory*, vol. 45, pp. 1121-1128, May 1999.

[56] B. M. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," submitted to *IEEE Journal on Selec. Areas in Commun.*, 2001.

[57] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Info. Theory*, pp. 284-287, March 1974.

[58] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," *Proceedings GLOBECOM '89*, pp. 1680-1686, November 1989.

[59] P. Robertson, P. Hoeher and E. Villebrun, "Optimal and suboptimal maximum a posteriori algorithms suitable for turbo decoding", *European Trans. Telecommun.*, vol. 8, pp. 119-125, Mar-Apr. 1997.

[60] John Fan, *Constrained Coding and Soft Iterative Decoding*, Kluwer Academic Publishers, July 2001.

[61] T. J. Richardson and R. L. Urbanke, "Efficient encoding of low-density parity check codes," *IEEE Transactions on Information Theory*, vol. 47, pp. 638-656, February 2001.

[62] R. M. Tanner, "A Recursive Approach to Low Complexity Codes," *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533-547, September 1981.

[63] J. D. Forney, Jr., "Codes on graphs: normal realizations," *IEEE Transactions on Information Theory*, vol. 47, pp:520-548, February 2001.

[64] R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, "Turbo decoding as an istance of Pearl's 'beliefe propagation' algorithm," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 140-152, February 1998.

[65] M. Bossert, *Channel Coding for Telecommunications*, Wiley & Sons, 1999.

[66] M. Austin, "Decision feedback equalization for digital communication over dispersive channels," *MIT Research Laboratory of Electronics Technical Report 461*, August 1967.

[67] N. Al-Dhahir and J. M. Cioffi, "MMSE decision-feedback equalizers: finite-length results," *IEEE Transactions on Information Theory*, vol. 41, no. 4, pp. 961-975, July 1995.

[68] N. Al-Dhahir and A. H. Sayed, "The finite length multi-input multi-output MMSE-DFE," *IEEE Transactions on Signal Processing*, vol. 48, no. 10, pp. 2921-2936, October 2000.