ERROR-CORRECTION CODING FOR RELIABLE COMMUNICATION IN THE PRESENCE OF EXTREME NOISE

Thesis by

Chi-chao Chao

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

1989

(Submitted May 22, 1989)

ABSTRACT

This thesis is a study of error-correcting codes for reliable communication in the presence of extreme noise. We consider very noisy channels, which occur in practice by pushing ordinary channels to their physical limits. Both block codes and convolutional codes are examined.

We show that the family of triply orthogonal codes, defined and studied in this thesis, or orthogonal codes can be used to achieve channel capacity for certain classes of very noisy discrete memoryless channels. The performance of binary block codes on the unquantized additive white Gaussian noise channel at very low signal-to-noise ratios is studied. Expressions are derived for the decoder block error as well as bit error probabilities and the asymptotic coding gain near the point where the signal energy is zero.

The average distance spectrum for the ensemble of time-varying convolutional codes is computed, and the result gives a surprisingly accurate prediction of the growth rate of the number of fundamental paths at large distance for fixed codes. A Gilbert-like free distance lower bound is also given. Finally, a Markov chain model is developed to approximate burst error statistics of Viterbi decoding. The model is validated through computer simulations and is compared with the previously proposed geometric model.

ACKNOWLEDGEMENT

I am deeply indebted to my advisor, Prof. Robert J. McEliece, for his guidance and encouragement. His invaluable insight and advice greatly improved this work. I would like to thank Profs. Richard Wilson and Gary Lorden for helpful discussions. Other people who had a strong influence in my mathematical and engineering training are Profs. Joel Franklin, Edward Posner, P.P. Vaidyanathan, Yaser Abu-Mostafa, and Dr. Laif Swanson.

Special thanks are due to Dr. Fabrizio Pollara at JPL for his contribution and help. I am grateful to Mr. Ivan Onyszchuk for his assistance in simulations and his constructive criticism that improved the manuscript a lot. I also thank Mr. Kumar Sivarajan for his comments about this thesis. My gratitude also goes to Drs. Khaled Abdel-Ghaffar, Eric Majani, and Kar-Ming Cheung for helpful discussions.

My sister Ya-Yuan and brother-in-law Harry gave me a lot of help in all respects during my stay at Caltech. Finally, I would like to thank my girlfriend Pearl for her love and patience.

This work was supported by the Air Force Office of Scientific Research under Grant No. AFOSR-88-0247. I appreciate their support.

CONTENTS

to my parents,

Tung-Hsi and Bi-Hsia,

and

to Pearl

CHAPTER 1

INTRODUCTION

1.1 Overview

This thesis is a study of error-correcting codes for reliable communication in the presence of *extreme* noise. We consider "very noisy" channels, which occur in practice by pushing ordinary channels to their physical limits. Communication systems operating at very low signal-to-noise ratios and data storage systems with very high data densities are common examples of such channels. In the next two chapters, block codes are studied, while convolutional codes are examined in the last two chapters.

In Chapter 2, we consider discrete memoryless Class I and Class II very noisy channels as identified in Majani's thesis. It is shown that, for both classes of very noisy channels, orthogonal codes can be used to achieve the computational cutoff rate. Most importantly, we prove that the family of "triply orthogonal" codes, defined and studied in Section 2.3, achieves channel capacity for binary-input Class I very noisy channels, and so does the family of orthogonal codes if the channels are also symmetric. However, similar results do not hold for Class II channels. Generalizations to channels with more than two inputs are also given. The results obtained in this chapter are among the few, since the introduction of Shannon's channel coding theorem, to show explicitly codes that achieve channel capacity.

In Chapter 3, we study the behavior of binary block codes on the unquantized additive white Gaussian noise channel at very low signal-to-noise ratios with maximum-likelihood decoding. Expressions are derived for the decoder block error and bit error probabilities near the point where the signal energy is zero. Asymptotic coding gain at low signal-to-noise ratios is computed. Applications of the results to several block codes are discussed.

The path weight enumerator of a convolutional code, which counts all fundamental paths in the code's state diagram, is of great importance in performance estimation. In Chapter 4, we compute the average distance profile for the ensemble of time-varying convolutional codes, and find that the result gives a surprisingly accurate prediction of the growth rate of the number of fundamental paths at large distance for fixed codes. We also estimate the average free distance for time-varying codes and obtain, for each constraint length, a Gilbert-like free distance lower bound. A similar random coding analysis for the weight of information bits for fundamental paths is given. Examples of the performance of several convolutional codes at low signal-to-noise ratios are discussed.

In some applications such as concatenated coding systems, the burst error statistics of a Viterbi decoder are important. Best showed that any convolutional coding scheme with maximum-likelihood decoding on a discrete memoryless channel can be modeled exactly as a finite state Markov chain. It then becomes apparent that, for Viterbi decoding on discrete memoryless channels, output burst and guardspace lengths are distributed asymptotically geometrically. However, the excessive amount of computation required for Best's method makes it infeasible for practical codes. In Chapter 5, we develop a Markov chain model to approximate the burst error statistics of Viterbi decoding. Our Markov chain model is validated through computer simulations and is compared with the geo-

CHAPTER 2

ORTHOGONAL CODES AND CHANNELS WITH NOISE SCALING

2.1 Introduction

Shannon's channel coding theorem guarantees that, as long as the rate is below channel capacity, codes exist so that arbitrarily reliable communication is possible. However, little is known about which codes can achieve channel capacity. One of the few explicit results is that when communicating over an additive white Gaussian noise (AWGN) channel, provided the bit signal-to-noise ratio E_b/N_0 is greater than $\ln 2$, the error probability of orthogonal codes can be made arbitrarily small [12]:

$$\lim_{M\to\infty} P_E = \begin{cases} 0 & \text{if } E_b/N_0 > \ln 2 \\ 1 & \text{if } E_b/N_0 < \ln 2. \end{cases}$$

Since for AWGN channels, no bit signal-to-noise ratio less than $\ln 2$ can be achieved, orthogonal codes provide error-free transmission for rates up to channel capacity. In this chapter, we show that the family of "triply orthogonal" codes or orthogonal codes can be used to achieve capacity for certain other classes of very noisy channels.

In [2] the concept of *noise scaling* is introduced to model certain classes of noisy channels. The noise scaling parameter z is the *resource* per stored or transmitted bit, where the abstract resource could be energy, area, time, etc. For example, the scaling parameter could be the area per stored bit for a VLSI memory chip

or the symbol signal-to-noise ratio for a practical communication system. The parameter z directly affects the noisiness of the channel; the smaller z is, the noisier the channel will become. The channel capacity C is hence a function of the scaling parameter z. When the channel gets noisy, we use coding to combat noise. By Shannon's channel coding theorem, reliable communication is possible if and only if the code rate R is below the capacity $C(z)$. In many communication or information storage systems, we want to transmit or store information not only reliably but also efficiently or compactly. It will then be interesting to know what the ultimate limits of information density are for various channels. Let λ be the resource per information bit:

$$\lambda = \frac{z}{R}.$$

Then the ultimate minimum resource per information bit should be

$$\lambda_{\min} = \inf_{z>0} \frac{z}{C(z)}. \tag{2.1}$$

For many practical channels, it is preferable to push them to their very noisy limit to achieve the largest information density. We therefore define

$$\lambda_C = \lim_{z\to 0} \frac{z}{C(z)},$$

the minimum achievable resource per information bit as $z \to 0$. It will be the absolute minimum of λ if the infimum in (2.1) is achieved as $z \to 0$. Another quantity of interest is λ_0, a practical measure of the minimum resource per information bit needed as $z \to 0$, defined by

$$\lambda_0 = \lim_{z\to 0} \frac{z}{R_0(z)},$$

where R_0 is the channel's computational cutoff rate. It is shown in [2] that the family of orthogonal codes achieves λ_C for very noisy binary symmetric channels. We will generalize the result in this chapter.

The study of very noisy channels in [3] identifies two classes of discrete memoryless very noisy channels: Class I and Class II. We will use the model in [3] throughout this chapter. The rest of this chapter is divided into seven sections. In Section 2.2, λ_C and λ_0 are computed for both classes of binary-input very noisy channels. In Section 2.3, we study orthogonal codes and define triply orthogonal codes as well as generalized triply orthogonal codes. A construction of a sequence of generalized triply orthogonal codes with arbitrary symbol size is given. Section 2.4 shows that orthogonal codes can be used to achieve λ_0 for both classes of binary-input very noisy channels. In Section 2.5, we prove that the family of triply orthogonal codes achieves λ_C for binary-input Class I very noisy channels and orthogonal codes will also achieve λ_C if the channels are symmetric. Section 2.6 is about binary-input Class II channels. We generalize our results to symmetric Class I channels with more than two inputs in Section 2.7. Finally, possible further research is discussed in Section 2.8.

2.2 Binary-Input Channels with Noise Scaling

We are particularly interested in the behavior of channels when they are pushed to their "very noisy" limit, i.e., if noise scaling is possible, when the scaling parameter z becomes very small. In other words, we want to study the behavior of channels in the neighborhood of zero capacity. The model for very noisy channels studied in [3] will be used in this chapter.

Consider a discrete memoryless channel (DMC) with input alphabet $\mathcal{X}$ and output alphabet $\mathcal{Y}$.

Definition 2.1 [3] *A DMC is a very noisy channel (VNC) if its transition prob-*

abilities satisfy

$$p(y|x) = w(y) + \epsilon \cdot \sigma(x,y) + O(\epsilon^2), \quad \textit{for all } x \in \mathcal{X} \textit{ and } y \in \mathcal{Y}, \tag{2.2}$$

where $\epsilon \ll 1$, $w(y)$ *is a probability distribution, i.e.,*

$$w(y) \geq 0, \quad \textit{for all } y \in \mathcal{Y}, \textit{ and } \sum_{y \in \mathcal{Y}} w(y) = 1, \tag{2.3}$$

and $\sigma(x,y)$*'s are fixed numbers satisfying*

$$\sum_{y \in \mathcal{Y}} \sigma(x,y) = 0, \quad \textit{for all } x \in \mathcal{X}. \tag{2.4}$$

It is clear that

$$\lim_{\epsilon \to 0} p(y|x) = w(y),$$

and hence

$$\lim_{\epsilon \to 0} C = 0.$$

Note that if the $w(y)$'s and $\sigma(x,y)$'s are fixed, then the behavior of the channel is controlled by the single parameter ϵ. For our model, ϵ is a function of the noise scaling parameter z.

Let the set $\mathcal{Y}$ of channel outputs be partitioned into two sets $\mathcal{Y}_1$ and $\mathcal{Y}_2$, where

$$\mathcal{Y}_1 = \{y \in \mathcal{Y} : w(y) \neq 0\},$$

$$\mathcal{Y}_2 = \{y \in \mathcal{Y} : w(y) = 0\}.$$

In [3] two classes of VNCs are identified. Class I VNCs contain all VNCs for which $\mathcal{Y}_2$ is empty, covering those VNCs studied in [4] and [5]. Class II VNCs contain all VNCs for which $\mathcal{Y}_2$ is not empty. For example, a very noisy binary symmetric channel is of Class I and a very noisy Z-channel is of Class II. For Class I VNCs, the channel capacity is of the order ϵ^2, but, for Class II VNCs, the capacity is of the order ϵ.

In this section, only binary-input channels are considered. Without loss of generality, let the input alphabet $\mathcal{X}$ be $\{0,1\}$. From [3], for binary-input Class I VNCs, the capacity is achieved with uniform input probabilities, given by

$$C = \frac{\epsilon^2}{8 \ln 2} \cdot \left(\sum_{y \in y} \frac{1}{w(y)} (\sigma(0,y) - \sigma(1,y))^2 \right) + O(\epsilon^3).$$

For binary-input Class II VNCs, the capacity is

$$\begin{aligned} C &= \frac{\epsilon}{\ln 2} \cdot \left(\max_{0 \le \mu \le 1} \sum_{y \in y_2} \left((1-\mu)\sigma(0,y) \ln \frac{\sigma(0,y)}{(1-\mu)\sigma(0,y) + \mu\sigma(1,y)} \right.\right. \\ &\quad \left.\left. + \mu\sigma(1,y) \ln \frac{\sigma(1,y)}{(1-\mu)\sigma(0,y) + \mu\sigma(1,y)} \right) \right) + O(\epsilon^2). \end{aligned} \tag{2.5}$$

Definition 2.2 [6, p. 94] *A DMC is symmetric if the set of outputs can be partitioned into subsets in such a way that, for each subset in the matrix of transition probabilities, each row is a permutation of each other row and the same is true of columns.*

The capacity of a symmetric DMC is achieved with equiprobable inputs. Hence, for a symmetric binary-input Class II VNC, the maximum in (2.5) occurs at $\mu = 1/2$ and the capacity is

$$C = \frac{\epsilon}{\ln 2} \cdot \left(\sum_{y \in y_2} \sigma(0,y) \ln \frac{2\sigma(0,y)}{\sigma(0,y) + \sigma(1,y)} \right) + O(\epsilon^2).$$

We now define three cases of severity of noise for VNCs, which are generalizations of those in [7].

Definition 2.3 *A VNC is said to have moderate noise if the following limit is nonzero:*

$$\lim_{z \to 0} \frac{\epsilon^2(z)}{z} = k_1, \qquad \textit{for Class I,} \tag{2.6}$$

or

$$\lim_{z \to 0} \frac{\epsilon(z)}{z} = k_2, \qquad \textit{for Class II.} \tag{2.7}$$

For this case, a finite nonzero number of information bits per unit resource can be transmitted or stored as $z \to 0$, so λ_C is finite and nonzero.

Definition 2.4 *If the limit (2.6) or (2.7) approaches infinity, then a VNC is said to have light noise.*

For this case, an infinite number of information bits per unit resource can be transmitted or stored as $z \to 0$, so λ_C is zero. This is an unlikely case and rarely occurs in practice.

Definition 2.5 *A VNC is said to have severe noise if the limit (2.6) or (2.7) is zero.*

For this case, all the bits become redundant as $z \to 0$ and λ_C is infinity.

For the case of moderate noise, for binary-input Class I channels,

$$\lambda_C = \frac{8 \ln 2}{k_1} \cdot \left(\sum_{y \in \mathcal{Y}} \frac{1}{w(y)} \left(\sigma(0,y) - \sigma(1,y) \right)^2 \right)^{-1}, \tag{2.8}$$

and for symmetric binary-input Class II channels,

$$\lambda_C = \frac{\ln 2}{k_2} \cdot \left(\sum_{y \in \mathcal{Y}_2} \sigma(0,y) \ln \frac{2\sigma(0,y)}{\sigma(0,y) + \sigma(1,y)} \right)^{-1}. \tag{2.9}$$

We now compute λ_0 for both classes of channels, starting with the definition of R_0,

Definition 2.6 [8, p. 68] *For each pair $x_1, x_2 \in \mathcal{X}$, let*

$$J(x_1, x_2) = \sum_{y \in \mathcal{Y}} \sqrt{p(y|x_1) p(y|x_2)}$$

and

$$J_0 = \min \left\{ E \left(J(X_1, X_2) \right) \right\},$$

where the minimization is taken over all i.i.d. random variables assuming values in $\mathcal{X}$. The computational cutoff rate R_0 is defined by

$$R_0 = -\log_2 J_0.$$

For binary-input channels,

$$E(J(X_1,X_2)) = (1-\nu)^2+\nu^2+2\sum_{y\in\mathcal{Y}}\nu(1-\nu)\sqrt{p(y|0)p(y|1)}$$

$$= 1+2\nu\left(\sum_{y\in\mathcal{Y}}\sqrt{p(y|0)p(y|1)}-1\right)-2\nu^2\left(\sum_{y\in\mathcal{Y}}\sqrt{p(y|0)p(y|1)}-1\right),$$

where $\nu = p(X_1=0) = p(X_2=0)$. Since the minimum of $E(J(X_1,X_2))$ is achieved at $\nu = 1/2$,

$$J_0 = \frac{1}{2}\left(1+\sum_{y\in\mathcal{Y}}\sqrt{p(y|0)p(y|1)}\right)$$

and

$$R_0 = 1-\log_2\left(1+\sum_{y\in\mathcal{Y}}\sqrt{p(y|0)p(y|1)}\right). \tag{2.10}$$

The following are proved in Appendix 2.A. For binary-input Class I VNCs,

$$R_0 = \frac{\epsilon^2}{16\ln 2}\cdot\left(\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)+O(\epsilon^3). \tag{2.11}$$

For binary-input Class II VNCs,

$$R_0 = \frac{\epsilon}{4\ln 2}\cdot\left(\sum_{y\in\mathcal{Y}_2}\left(\sqrt{\sigma(0,y)}-\sqrt{\sigma(1,y)}\right)^2\right)+O(\epsilon^2). \tag{2.12}$$

Note that, for binary-input Class I VNCs, $R_0 \approx C/2$, originally found in [4].

Therefore, for the case of moderate noise as characterized in (2.6) or (2.7), for binary-input Class I channels,

$$\lambda_0 = \lim_{z\to 0}\frac{z}{R_0(z)} = \frac{16\ln 2}{k_1}\cdot\left(\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)^{-1}, \tag{2.13}$$

and, for binary-input Class II channels,

$$\lambda_0 = \lim_{z\to 0}\frac{z}{R_0(z)} = \frac{4\ln 2}{k_2}\cdot\left(\sum_{y\in\mathcal{Y}_2}\left(\sqrt{\sigma(0,y)}-\sqrt{\sigma(1,y)}\right)^2\right)^{-1}. \tag{2.14}$$

Also note that $\lambda_0 = 2\lambda_C$ for Class I channels.

Example. Consider the binary symmetric channel formed by binary phase-shift keying (BPSK) with output quantization on the AWGN channel. The matrix of transition probabilities is

$$\mathbf{P}_{Y|X} = \begin{bmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{bmatrix} + \epsilon \cdot \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}.$$

And

$$\epsilon(z) = \frac{1}{2} - Q(\sqrt{2z}),$$

where z is the symbol signal-to-noise ratio and $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-t^2/2}\, dx$. This channel becomes a Class I VNC as $z \to 0$. Since

$$k_1 = \lim_{z \to 0} \frac{\epsilon^2(z)}{z} = \frac{1}{\pi},$$

which is nonzero, it has moderate noise and $\lambda_C = \frac{\pi}{2} \ln 2$, which is the minimum achievable bit signal-to-noise ratio as $z \to 0$. Actually, it can be shown that λ_C is the minimum achievable bit signal-to-noise ratio for all z. For the case of no output quantization, it is well known that the minimum achievable bit signal-to-noise ratio is $\ln 2$, which corresponds to a $10 \log(\pi/2) \approx 1.96$ dB gain over hard quantization. Also, for this channel, $\lambda_0 = 2\lambda_C = \pi \ln 2$.

2.3 Orthogonal Codes and Triply Orthogonal Codes

We start with the definition of a Hadamard matrix.

Definition 2.7 [9, Chap. 14] *A Hadamard matrix of order M is an $M \times M$ matrix $\mathbf{H}_M$ of $+1$'s and -1's such that*

$$\mathbf{H}_M \mathbf{H}_M^T = M\mathbf{I}.$$

We may permute rows or columns of $\mathbf{H}_M$ or multiply rows or columns of $\mathbf{H}_M$ by -1 without disturbing the above property, and such matrices are considered

equivalent. Given a Hadamard matrix, we can always find one equivalent to it in normalized form, i.e., the first row and the first column of the matrix contain only +1's. Hereafter, only Hadamard matrices in normalized form are considered.

It is known that if $\mathbf{H}_M$ exists then M necessarily equals 1, 2, or a multiple of 4. For a Hadamard matrix $\mathbf{H}_M$ of order $M = 4t$, take any two distinct rows $\mathbf{u}$ and $\mathbf{v}$ of $\mathbf{H}_M$ other than the first row. Consider the following matrix:

$$\begin{bmatrix} 1 & 1 & \dots & 1 \\ u_1 & u_2 & \dots & u_M \\ v_1 & v_2 & \dots & v_M \end{bmatrix}.$$

Let a_1, a_2, a_3, a_4 be the number of columns of the form

$$\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix},$$

respectively.

Lemma 2.1 [9, Chap. 14] $a_1 = a_2 = a_3 = a_4 = M/4$.

Now take any three distinct rows of $\mathbf{H}_M$, say $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{w}$. Consider the matrix:

$$\begin{bmatrix} u_1 & u_2 & \dots & u_M \\ v_1 & v_2 & \dots & v_M \\ w_1 & w_2 & \dots & w_M \end{bmatrix}.$$

Let b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8 be the number of columns of the form

$$\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix}, \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix},$$

respectively.

Proposition 2.1 $b_1 + b_8 = b_2 + b_7 = b_3 + b_6 = b_4 + b_5 = M/4$.

Proof. If any one of u, v and w is the first row of $\mathbf{H}_M$, then this proposition follows directly from Lemma 2.1. If none of these is the first row, then from Lemma 2.1, $b_1+b_2 = b_3+b_4 = b_5+b_6 = b_7+b_8 = b_1+b_5 = b_2+b_6 = b_3+b_7 = b_4+b_8 = b_1+b_3 = b_2+b_4 = b_5+b_7 = b_6+b_8 = M/4$ which implies $b_2 = b_3 = b_5 = b_8, b_1 = b_4 = b_6 = b_7$. The proposition follows immediately. ∎

Definition 2.8 [10] *An orthogonal code is a code such that for any two codewords the number of bit-by-bit agreements equals the number of disagreements.*

Clearly, each row of an $M \times M$ Hadamard matrix can be taken as a codeword of an orthogonal code of M codewords via the mapping that the $+1$'s are changed to 0's and the -1's are changed to 1's. Thus orthogonal codes have the same correlation properties as those of Hadamard matrices.

Definition 2.9 [11] *An orthogonal array (n, M, s, t) of strength t, size n, M constraints, and s levels is an $M \times n$ matrix, with entries from a set Σ of $s \geq 2$ elements, such that each $t \times n$ submatrix contains all possible $t \times 1$ column vectors with the same frequency.*

It follows directly that n must be divisible by s^t. From Lemma 2.1, the Hadamard matrix $\mathbf{H}_M$ of order $M > 2$ with the first row deleted is an orthogonal array $(M, M-1, 2, 2)$.

Definition 2.10 *A triply orthogonal code of length n and M codewords is defined to be a code in which each codeword is a row of an orthogonal array $(n, M, 2, 3)$ with the symbol set $\Sigma = \{0, 1\}$.*

It is shown in [11] that for an orthogonal array $(n, M, s, 3)$ of strength 3, M must satisfy

$$M \leq \left\lfloor \frac{\frac{n}{s} - 1}{s - 1} \right\rfloor + 1, \tag{2.15}$$

where $\lfloor \alpha \rfloor$ is the largest integer not exceeding α. Substituting $s = 2$ in the above inequality, we obtain $M \leq n/2$ for triply orthogonal codes. A different but simpler proof is given as follows.

Proposition 2.2 *For triply orthogonal codes of length n and M codewords, M must satisfy $M \leq n/2$.*

Proof. For an orthogonal array $(n, M, 2, 3)$ with symbol set $\Sigma = \{-1, +1\}$, consider the $M \times n/2$ submatrix $\mathbf{B}$ formed by taking all columns with the first entry -1. Since the original orthogonal array is of strength 3, the inner product of any two distinct rows of $\mathbf{B}$ is zero. Let $\mathbf{V} = \mathbf{U}_1 \oplus \mathbf{U}_2 \oplus \cdots \oplus \mathbf{U}_M$, where $\mathbf{U}_i$ is the subspace spanned by the row i of $\mathbf{B}$. Then $\mathbf{V}$ is the row space of $\mathbf{B}$ and $\dim \mathbf{V} = M$ because the $\mathbf{U}_i$'s are orthogonal and, hence, independent. But, notice that $\mathbf{B}$ is an $M \times n/2$ matrix, and hence $\dim \mathbf{V} \leq n/2$. ∎

The proof suggests the following construction of the orthogonal array $(2M, M, 2, 3)$ with symbol set $\Sigma = \{-1, +1\}$ whenever the Hadamard matrix $\mathbf{H}_M$ exists:

$$\mathbf{A}_M = [\ \mathbf{H}_M \quad -\mathbf{H}_M\ ].$$

This construction happens to be the same as that in [12]. From Proposition 2.1, $\mathbf{A}_M$ is indeed an orthogonal array $(2M, M, 2, 3)$. Hence a triply orthogonal code of length $2M$ and M codewords can be obtained from $\mathbf{A}_M$ by changing all $+1$'s to 0's and all -1's to 1's. From [9, Theorem 14.1.1],

$$\mathbf{H}_{2M} = \begin{bmatrix} \mathbf{H}_M & \mathbf{H}_M \\ \mathbf{H}_M & -\mathbf{H}_M \end{bmatrix}$$

is a Hadamard matrix of order $2M$ if $\mathbf{H}_M$ is a Hadamard matrix of order M. This is called the Sylvester construction. A triply orthogonal code can hence be thought of as a coset of an orthogonal code. An example of a Hadamard matrix and an orthogonal code is illustrated in Figure 2.1. Also an example of a triply orthogonal code is shown in Figure 2.2.

We now generalize the definition of triply orthogonal codes to nonbinary cases.

Definition 2.11 *A generalized triply orthogonal code of s symbols, length n and M codewords is defined to be a code in which each codeword is a row of an orthogonal array* $(n, M, s, 3)$ *of strength* 3.

+	+	+	+	+	+	+	+
+	−	+	−	+	−	+	−
+	+	−	−	+	+	−	−
+	−	−	+	+	−	−	+
+	+	+	+	−	−	−	−
+	−	+	−	−	+	−	+
+	+	−	−	−	−	+	+
+	−	−	+	−	+	+	−

0	0	0	0	0	0	0	0
0	1	0	1	0	1	0	1
0	0	1	1	0	0	1	1
0	1	1	0	0	1	1	0
0	0	0	0	1	1	1	1
0	1	0	1	1	0	1	0
0	0	1	1	1	1	0	0
0	1	1	0	1	0	0	1

Figure 2.1: Hadamard Matrix and Orthogonal Code, $M = 8$.

0	0	0	0	1	1	1	1
0	1	0	1	1	0	1	0
0	0	1	1	1	1	0	0
0	1	1	0	1	0	0	1

Figure 2.2: Triply Orthogonal Code, $n = 8$ and $M = 4$.

A sequence of such codes for arbitrary s will be constructed. First, consider the case when s is p^n, a prime power. In [11], the following lemma is proved to be a sufficient condition of the existence of an orthogonal array.

Lemma 2.2 *If we can find a $k \times m$ matrix $\mathbf{C}$:*

$$\mathbf{C} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1m} \\ c_{21} & c_{22} & \cdots & c_{2m} \\ \vdots & \vdots & & \vdots \\ c_{k1} & c_{k2} & \cdots & c_{km} \end{bmatrix},$$

where $c_{ij} \in GF(p^r)$, and for which every submatrix obtained by taking t rows is of rank t, then we can construct an orthogonal array (s^m, k, s, t), where $s = p^r$.

Form an $m \times s^m$ matrix $\mathbf{D}$ whose columns are all possible $m \times 1$ column vectors whose coordinates are in $GF(p^r)$. Then an orthogonal array (s^m, k, s, t) is obtained

by multiplying $\mathbf{C}$ in the above lemma by $\mathbf{D}$. From a coding-theoretic point of view, we can think of $\mathbf{C}^T$ as a parity-check matrix of a $\lfloor t/2 \rfloor$-error correcting code. BCH-code-like constructions for $\mathbf{C}$'s are given in the following propositions.

Proposition 2.3 *If $s = p^r$, where p is an odd prime, let $N = s^m - 1$ and consider the following $(s^m + 1) \times 3m$ matrix:*

$$\mathbf{C} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathbf{1} & \mathbf{1} & \mathbf{1} \\ \mathbf{1} & \alpha & \alpha^2 \\ \vdots & \vdots & \vdots \\ \mathbf{1} & \alpha^{N-1} & \alpha^{2(N-1)} \end{bmatrix},$$

where α is a primitive element in $GF(s^m)$ and $\mathbf{0}, \mathbf{1}$, and α are expressed as $1 \times m$ row vectors in $GF(s)$. Then every submatrix obtained by taking 3 rows of $\mathbf{C}$ has rank 3.

Proof. Now consider 0, 1, and α as elements in $GF(s)$. Any submatrix obtained by taking three rows of $\mathbf{C}$ without the first 2 rows is of the form

$$\mathbf{C}' = \begin{bmatrix} 1 & \alpha^i & \alpha^{2i} \\ 1 & \alpha^j & \alpha^{2j} \\ 1 & \alpha^l & \alpha^{2l} \end{bmatrix}.$$

Then its determinant is

$$\det(\mathbf{C}') = (\alpha^j - \alpha^i)(\alpha^l - \alpha^i)(\alpha^l - \alpha^j) \neq 0.$$

Hence $\mathbf{C}'$ is of rank 3.

Any submatrix obtained by taking the first row of $\mathbf{C}$ and any two of the last N rows is of the form

$$\mathbf{C}' = \begin{bmatrix} 1 & 0 & 0 \\ 1 & \alpha^i & \alpha^{2i} \\ 1 & \alpha^j & \alpha^{2j} \end{bmatrix}$$

and its determinant is

$$\det(\mathbf{C}') = \alpha^i \alpha^j (\alpha^j - \alpha^i) \neq 0.$$

A similar result follows for any matrix obtained by taking the second row of $\mathbf{C}$ and any two of the last N rows.

Any submatrix obtained by taking the first two rows of $\mathbf{C}$ and one row of the last N rows is of the form

$$\mathbf{C}' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & \alpha^i & \alpha^{2i} \end{bmatrix}$$

and its determinant is

$$\det(\mathbf{C}') = -\alpha^i \neq 0,$$

which completes the proof. ∎

From Lemma 2.2 and this proposition, from $\mathbf{C}$ we can construct a sequence of orthogonal arrays $(s^{3m}, s^m + 1, s, 3)$ for arbitrary $m > 0$ if $s = p^r$, p an odd prime. If s is a power of 2, then we have the following construction.

Proposition 2.4 *If $s = 2^r$, let $N = s^m - 1$ and consider the following $(s^m+2)\times 3m$ matrix:*

$$\mathbf{C} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathbf{1} & \mathbf{1} & \mathbf{1} \\ \mathbf{1} & \alpha & \alpha^2 \\ \vdots & \vdots & \vdots \\ \mathbf{1} & \alpha^{N-1} & \alpha^{2(N-1)} \end{bmatrix},$$

where α is a primitive element in $GF(s^m)$ and $\mathbf{0}, \mathbf{1}$, and α are expressed as $1 \times m$ row vectors in $GF(s)$. Then every submatrix obtained by taking 3 rows of $\mathbf{C}$ has rank 3.

Proof. Again consider 0, 1, and α as elements in $GF(s)$. The submatrix obtained by taking the first three rows of $\mathbf{C}$ is an identity matrix and hence it has rank 3. Consider any submatrix obtained by taking the second row of $\mathbf{C}$ and any two of

the last N rows. It is of the form

$$\mathbf{C}' = \begin{bmatrix} 0 & 1 & 0 \\ 1 & \alpha^i & \alpha^{2i} \\ 1 & \alpha^j & \alpha^{2j} \end{bmatrix}$$

and its determinant is

$$\begin{aligned} \det(\mathbf{C}') &= \alpha^{2i} - \alpha^{2j} \\ &= \left(\alpha^i - \alpha^j\right)^2 \neq 0. \end{aligned}$$

All the remaining cases follow similarly from Proposition 2.3. ∎

Again from Lemma 2.2 and this proposition, from C we can construct a sequence of orthogonal arrays $(s^{3m}, s^m + 2, s, 3)$ for arbitrary $m > 0$ if $s = 2^r$.

Now consider the case when $s = s_1 s_2 \cdots s_u$, where $s_i = p_i^{r_i}$, and the p_i's are primes. In [13], a generalization of MacNeish's theorem is proved. We restate it here as the following lemma.

Lemma 2.3 *Let $n = n_1 n_2 \cdots n_u$. If orthogonal arrays (n_i, M_i, s_i, t) exist for every i, $i = 1, 2, \ldots, u$, then we can construct an orthogonal array (n, M, s, t), where $M = \min(M_1, M_2, \ldots, M_u)$.*

The construction can be found in [13]. From this lemma and the previous constructions, the following proposition is obtained.

Proposition 2.5 *If $s = s_1 s_2 \cdots s_u$, where $s_i = p_i^{r_i}$ and the p_i's are primes, then we can construct an orthogonal array $(s^{3m}, M, s, 3)$, where $M = \min(s_1^m + 1, s_2^m + 1, \ldots, s_u^m + 1)$, for arbitrary $m > 0$.*

From Equation (2.15), for an orthogonal array $(s^{3m}, M, s, 3)$, M must satisfy

$$M \leq \left\lfloor \frac{s^{3m-1} - 1}{s - 1} \right\rfloor,$$

while for our construction, $M = \min(s_1^m + 1, s_2^m + 1, \ldots, s_u^m + 1)$. An example of a generalized triply orthogonal code is shown in Figure 2.3.

0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2
0	1	2	1	2	0	2	0	1	1	2	0	2	0	1	0	1	2	2	0	1	0	1	2	1	2	0
0	1	2	2	0	1	1	2	0	1	2	0	0	1	2	2	0	1	2	0	1	1	2	0	0	1	2
0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	2

Figure 2.3: Generalized Triply Orthogonal Code, $n = 27$, $M = 4$ and $s = 3$.

2.4 λ_0-Theorem for Binary-Input Channels

In this section, we prove an important theorem about λ_0 for both classes of binary-input VNCs.

Theorem 2.1 *For both classes of binary-input VNCs, if moderate noise is assumed, the family of orthogonal codes can be used to achieve a minimum resource per information bit of* λ_0.

Proof. Let $\mathbf{C} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{M-1}\}$ be an orthogonal code of length n and M codewords, where $n = M$. The code $\mathbf{C}$ is used on a binary-input channel with maximum-likelihood decoding. Our goal is to prove that the decoder error probability $P_E \to 0$ as $M \to \infty$ if $\lambda > \lambda_0$. Assume the codeword $\mathbf{x}_i$ is transmitted and $\mathbf{y}$ is received. Let $P_E^{(i)}$ denote the decoder error probability given that $\mathbf{x}_i$ is transmitted. First, consider the case of two codewords: $\{\mathbf{x}_i, \mathbf{x}_j\}$. The decoder will make a correct decision if and only if $p(\mathbf{y}|\mathbf{x}_j) < p(\mathbf{y}|\mathbf{x}_i)$. Hence the decoder error probability Q_j for the two-codeword case is

$$Q_j = \sum_{\mathbf{y} \in Y_j} p(\mathbf{y}|\mathbf{x}_i),$$

where $Y_j = \{\mathbf{y} : p(\mathbf{y}|\mathbf{x}_j) \geq p(\mathbf{y}|\mathbf{x}_i)\}$. (Here we exaggerate the decoder error probability a little bit by assuming that the decoder makes an error in case of a

tie.) Since for all $\mathbf{y} \in Y_j$, $\sqrt{p(\mathbf{y}|\mathbf{x}_j)/p(\mathbf{y}|\mathbf{x}_i)} \geq 1$, Q_j can be Chernoff-bounded as

$$\begin{aligned} Q_j &\leq \sum_{\mathbf{y}\in Y_j} p(\mathbf{y}|\mathbf{x}_i) \cdot \sqrt{\frac{p(\mathbf{y}|\mathbf{x}_j)}{p(\mathbf{y}|\mathbf{x}_i)}} + \sum_{\mathbf{y}\in Y^n - Y_j} p(\mathbf{y}|\mathbf{x}_i) \cdot \sqrt{\frac{p(\mathbf{y}|\mathbf{x}_j)}{p(\mathbf{y}|\mathbf{x}_i)}} \\ &= \sum_{\mathbf{y}\in Y^n} \sqrt{p(\mathbf{y}|\mathbf{x}_i)p(\mathbf{y}|\mathbf{x}_j)} \\ &= \prod_{l=1}^{n} \sum_{y\in Y} \sqrt{p(y|x_{il})p(y|x_{jl})}, \end{aligned}$$

where $\mathbf{x}_i = (x_{i1}, \ldots, x_{in})$ and $\mathbf{x}_j = (x_{j1}, \ldots, x_{jn})$. But

$$\sum_{y\in Y} \sqrt{p(y|x_{il})p(y|x_{jl})} = \sum_{y\in Y} p(y|x_{il}) = 1 \quad \text{if } x_{il} = x_{jl},$$

and

$$\sum_{y\in Y} \sqrt{p(y|x_{il})p(y|x_{jl})} = \sum_{y\in Y} \sqrt{p(y|0)p(y|1)} \quad \text{if } x_{il} \neq x_{jl}.$$

Since $\mathbf{x}_i$ and $\mathbf{x}_j$ are codewords of an orthogonal code, the number of bit-by-bit agreements equals the number of disagreements and it is $n/2$. We obtain

$$Q_j \leq \left(\sum_{y\in Y} \sqrt{p(y|0)p(y|1)} \right)^{\frac{n}{2}}.$$

For the original M-codeword case, the decoder error probability is bounded above by the union bound:

$$P_E^{(i)} \leq \sum_{j\neq i} Q_j$$

which yields

$$P_E^{(i)} \leq M \left(\sum_{y\in Y} \sqrt{p(y|0)p(y|1)} \right)^{\frac{n}{2}}.$$

It is shown in Appendix 2.A that, for Class I VNCs,

$$\sum_{y\in Y} \sqrt{p(y|0)p(y|1)} = 1 - \frac{\epsilon^2}{8} \cdot \left(\sum_{y\in Y} \frac{1}{w(y)} \left(\sigma(0,y) - \sigma(1,y)\right)^2 \right) + O(\epsilon^3),$$

and, for Class II VNCs,

$$\sum_{y\in Y} \sqrt{p(y|0)p(y|1)} = 1 - \frac{\epsilon}{2} \cdot \left(\sum_{y\in Y_2} \left(\sqrt{\sigma(0,y)} - \sqrt{\sigma(1,y)} \right)^2 \right) + O(\epsilon^2).$$

Hence, for Class I channels,

$$\begin{aligned} P_E^{(i)} &\leq M\left(1-\frac{\epsilon^2}{8}\cdot\left(\sum_{y\in y}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)+O(\epsilon^3)\right)^{\frac{n}{2}} \\ &= M\cdot M^{\frac{n}{2\ln M}\ln\left(1-\frac{\epsilon^2}{8}\cdot\left(\sum_{y\in y}\frac{1}{w(y)}(\sigma(0,y)-\sigma(1,y))^2\right)+O(\epsilon^3)\right)} \\ &= M^{1-\frac{n}{\ln M}\left(\frac{\epsilon^2}{16}\cdot\left(\sum_{y\in y}\frac{1}{w(y)}(\sigma(0,y)-\sigma(1,y))^2\right)+O(\epsilon^3)\right)}. \end{aligned}$$

Since $z=\lambda R$, where the code rate $R=\ln M/(n\ln 2)$, for fixed λ, z decreases as M increases. If moderate noise is assumed, then

$$\frac{n}{\ln M}\cdot\epsilon^2=\frac{\epsilon^2}{z}\cdot\frac{\lambda}{\ln 2}\sim k_1\cdot\frac{\lambda}{\ln 2},\qquad \text{for large } M.$$

Thus, for large M,

$$P_E^{(i)}\leq M^{1-\lambda\cdot\frac{k_1}{16\ln 2}\sum_{y\in y}\frac{1}{w(y)}(\sigma(0,y)-\sigma(1,y))^2}.$$

Recalling the expression for λ_0 in (2.13), we get

$$P_E^{(i)}\leq M^{1-\frac{\lambda}{\lambda_0}},$$

which implies that, if $\lambda>\lambda_0$, $P_E^{(i)}\to 0$ as $M\to\infty$.

For Class II channels,

$$\begin{aligned} P_E^{(i)} &\leq M\left(1-\frac{\epsilon}{2}\cdot\left(\sum_{y\in y_2}\left(\sqrt{\sigma(0,y)}-\sqrt{\sigma(1,y)}\right)^2\right)+O(\epsilon^2)\right)^{\frac{n}{2}} \\ &= M\cdot M^{\frac{n}{2\ln M}\ln\left(1-\frac{\epsilon}{2}\cdot\left(\sum_{y\in y_2}\left(\sqrt{\sigma(0,y)}-\sqrt{\sigma(1,y)}\right)^2\right)+O(\epsilon^2)\right)} \\ &= M^{1-\frac{n}{\ln M}\left(\frac{\epsilon}{4}\cdot\left(\sum_{y\in y_2}\left(\sqrt{\sigma(0,y)}-\sqrt{\sigma(1,y)}\right)^2\right)+O(\epsilon^2)\right)}. \end{aligned}$$

Again if moderate noise is assumed, then

$$\frac{n}{\ln M}\cdot\epsilon=\frac{\epsilon}{z}\cdot\frac{\lambda}{\ln 2}\sim k_2\cdot\frac{\lambda}{\ln 2},\qquad \text{for large } M.$$

Thus, for large M,

$$P_E^{(i)}\leq M^{1-\lambda\cdot\frac{k_2}{4\ln 2}\sum_{y\in y_2}\left(\sqrt{\sigma(0,y)}-\sqrt{\sigma(1,y)}\right)^2}.$$

Recalling the expression for λ_0 in (2.14), we obtain

$$P_E^{(i)} \leq M^{1-\frac{\lambda}{\lambda_0}},$$

which shows that, if $\lambda > \lambda_0$, $P_E^{(i)} \to 0$ as $M \to \infty$.

Since the above proof holds for every i, the theorem follows. ∎

One immediate corollary is that the theorem is still true if orthogonal codes are replaced by triply orthogonal codes, as seen from the proof.

2.5 λ_C-Theorem for Binary-Input Class I Channels

It is shown in [2] that the family of orthogonal codes achieves a minimum resource per information bit of λ_C for a binary symmetric channel if moderate noise is assumed. Using an extension of the method in [2], we generalize the result in this section.

Lemma 2.4 *Let $X_0, X_1, \ldots, X_{M-1}$ be i.i.d. normal random variables with mean 0 and variance 1. If $Y_1, Y_2, \ldots, Y_{M-1}$ are defined by*

$$Y_i = X_i + X_0, \quad \text{for } i = 1, 2, \ldots, M-1,$$

then the Y_i's are normal random variables with mean 0 and covariance matrix

$$\mathrm{Cov}(\mathbf{Y}) = \begin{bmatrix} 2 & 1 & \ldots & 1 \\ 1 & 2 & \ldots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \ldots & 2 \end{bmatrix}.$$

Proof. Since the sum of two normal random variables is still normal, the lemma follows from trivial verifications. ∎

The following theorem and corollary are the most important results in this chapter.

Theorem 2.2 *For binary-input Class I VNCs, if moderate noise is assumed, the family of triply orthogonal codes achieves λ_C, which is the minimum achievable resource per information bit as $z \to 0$.*

Proof. Let $\mathbf{C} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{M-1}\}$ be a triply orthogonal code of length n and M codewords, where $n = 2M$. The code $\mathbf{C}$ is used on a binary-input Class I channel. Without loss of generality, let the output alphabet $\mathcal{Y}$ of the channel be $\{0, 1, \ldots, L-1\}$. Suppose the codeword $\mathbf{x}_0$ is transmitted and the received vector is $\mathbf{y}$. Let P_E denote the decoder error probability and P_C denote the probability of correct decoding. A maximum-likelihood decoder will make a correct decision if and only if

$$p(\mathbf{y}|\mathbf{x}_0) > p(\mathbf{y}|\mathbf{x}_i), \qquad \text{for } i = 1, 2, \ldots, M-1$$

which yields

$$\prod_{j=1}^{n} p(y_j|x_{0j}) > \prod_{j=1}^{n} p(y_j|x_{ij}), \qquad \text{for } i = 1, 2, \ldots, M-1, \tag{2.16}$$

where x_{0j}'s, x_{ij}'s and y_j's are components of $\mathbf{x}_0$, $\mathbf{x}_i$ and $\mathbf{y}$, respectively. Let $N_i^0(y)$, $y \in \mathcal{Y}$, be the number of components of $\mathbf{y}$ equal to y when the corresponding $x_{0j} = 0$ and $x_{ij} = 1$. And let $N_i^1(y)$ denote the number of components of $\mathbf{y}$ equal to y when the corresponding $x_{0j} = 1$ and $x_{ij} = 0$. Since the codewords of a triply orthogonal code are rows of an orthogonal array of strength 3, which implies it is also of strength 2,

$$\sum_{y \in \mathcal{Y}} N_i^0(y) = \sum_{y \in \mathcal{Y}} N_i^1(y) = \frac{n}{4}. \tag{2.17}$$

After cancelling terms on both sides, (2.16) can be rewritten as

$$\prod_{y \in \mathcal{Y}} p(y|0)^{N_i^0(y)} p(y|1)^{N_i^1(y)} > \prod_{y \in \mathcal{Y}} p(y|1)^{N_i^0(y)} p(y|0)^{N_i^1(y)}.$$

Taking logarithms of both sides, we obtain

$$\sum_{y \in \mathcal{Y}} \left(N_i^0(y) \ln \frac{p(y|0)}{p(y|1)} + N_i^1(y) \ln \frac{p(y|1)}{p(y|0)} \right) > 0, \quad \text{for } i = 1, 2, \ldots, M-1. \tag{2.18}$$

Since the channel is of Class I, $w(y) > 0$ for all $y \in \mathcal{Y}$, which implies $p(y|0) > 0$ and $p(y|1) > 0$ for all y. Thus, $\ln(p(y|0)/p(y|1))$ or $\ln(p(y|1)/p(y|0))$ in (2.18) always exists.

In order to compute P_C, we define the following i.i.d. random variables. The first set of i.i.d. random variables $U_1, U_2, \ldots, U_n$ is defined by their common probability distribution: [1]

$$\Pr\left(U = \ln\frac{p(y|0)}{p(y|1)}\right) = p(y|0), \quad y \in \mathcal{Y}.$$

Since $p(y|0) > 0$ for all $y \in \mathcal{Y}$ and $\sum_{y\in\mathcal{Y}} p(y|0) = 1$, U is a valid probability distribution. The second set of i.i.d. random variables $V_1, V_2, \ldots, V_n$ is defined by their common probability distribution:

$$\Pr\left(V = \ln\frac{p(y|1)}{p(y|0)}\right) = p(y|1), \quad y \in \mathcal{Y}.$$

Similarly, V is a valid probability distribution. It is straightforward to calculate the means and variances of U and V:

$$\mathrm{E}(U) = \sum_{y\in\mathcal{Y}} p(y|0) \ln\frac{p(y|0)}{p(y|1)}, \tag{2.19}$$

$$\mathrm{E}(V) = \sum_{y\in\mathcal{Y}} p(y|1) \ln\frac{p(y|1)}{p(y|0)}, \tag{2.20}$$

also

$$\begin{aligned}
&\mathrm{Var}(U) = \mathrm{E}(U^2) - \mathrm{E}^2(U) \\
&= \sum_{y\in\mathcal{Y}} p(y|0) \ln^2\frac{p(y|0)}{p(y|1)} - \left(\sum_{y\in\mathcal{Y}} p(y|0) \ln\frac{p(y|0)}{p(y|1)}\right)^2 \\
&= \sum_{y\in\mathcal{Y}}\sum_{y'\in\mathcal{Y}} p(y|0)p(y'|0) \ln^2\frac{p(y|0)}{p(y|1)} - \sum_{y\in\mathcal{Y}}\sum_{y'\in\mathcal{Y}} p(y|0)p(y'|0) \ln\frac{p(y|0)}{p(y|1)} \ln\frac{p(y'|0)}{p(y'|1)}
\end{aligned}$$

[1] If $\ln(p(y|0)/p(y|1)) = \ln(p(y'|0)/p(y'|1))$ for some $y \neq y'$, then we define $\Pr(U = \ln(p(y|0)/p(y|1))) = p(y|0) + p(y'|0)$. All the derivations still hold. Similar modifications apply to the random variable V.

$$= \sum_{y\in\mathcal{Y}} \sum_{\substack{y'\in\mathcal{Y} \\ y'\neq y}} p(y|0)p(y'|0) \left(\ln^2 \frac{p(y|0)}{p(y|1)} - \ln \frac{p(y|0)}{p(y|1)} \ln \frac{p(y'|0)}{p(y'|1)} \right)$$

$$= \sum_{y\in\mathcal{Y}} \sum_{\substack{y'\in\mathcal{Y} \\ y'>y}} p(y|0)p(y'|0) \left(\ln^2 \frac{p(y|0)}{p(y|1)} + \ln^2 \frac{p(y'|0)}{p(y'|0)} - 2 \ln \frac{p(y|0)}{p(y|1)} \ln \frac{p(y'|0)}{p(y'|1)} \right)$$

$$= \sum_{y\in\mathcal{Y}} \sum_{\substack{y'\in\mathcal{Y} \\ y'>y}} p(y|0)p(y'|0) \ln^2 \frac{p(y|0)p(y'|1)}{p(y|1)p(y'|0)}, \tag{2.21}$$

and, similarly,

$$\mathrm{Var}(V) = \sum_{y\in\mathcal{Y}} \sum_{\substack{y'\in\mathcal{Y} \\ y'>y}} p(y|1)p(y'|1) \ln^2 \frac{p(y|1)p(y'|0)}{p(y|0)p(y'|1)}. \tag{2.22}$$

Now let $\mathbf{U} = (U_1, U_2, \ldots, U_n)$ and $\mathbf{V} = (V_1, V_2, \ldots, V_n)$. Then (2.18) becomes

$$\langle \mathbf{U}, \overline{\mathbf{x}_0} \cdot \mathbf{x}_i \rangle + \langle \mathbf{V}, \mathbf{x}_0 \cdot \overline{\mathbf{x}_i} \rangle > 0, \quad \text{for } i = 1, 2, \ldots, M-1,$$

where $\langle \mathbf{a}, \mathbf{b} \rangle$ denotes the real inner product $\sum_j a_j b_j$ of vectors $\mathbf{a}$ and $\mathbf{b}$, $\mathbf{a} \cdot \mathbf{b}$ is the bit-wise AND $(a_1 b_1, a_2 b_2, \ldots, a_n b_n)$ of $\mathbf{a}$ and $\mathbf{b}$, and $\bar{\mathbf{a}}$ is the bit-wise complement of vector $\mathbf{a}$. If we define $S_i, i = 1, 2, \ldots, M-1$, by

$$S_i = \langle \mathbf{U}, \overline{\mathbf{x}_0} \cdot \mathbf{x}_i \rangle + \langle \mathbf{V}, \mathbf{x}_0 \cdot \overline{\mathbf{x}_i} \rangle, \tag{2.23}$$

then the probability of correct decoding P_C is

$$P_C = \Pr\{S_i > 0, \quad i = 1, 2, \ldots, M-1\}. \tag{2.24}$$

If M is large and, hence, n is large, from the central limit theorem, the S_i's can be approximated by normal random variables. From (2.17) and (2.23), for $i = 1, 2, \ldots, M-1$,

$$\mathrm{E}(S_i) = \frac{n}{4} \left(\mathrm{E}(U) + \mathrm{E}(V) \right),$$

$$\mathrm{Var}(S_i) = \frac{n}{4} \left(\mathrm{Var}(U) + \mathrm{Var}(V) \right),$$

because U_i and V_i are independent. Define the following sets:

$$\mathcal{A}_0 = \{m : x_{0m} = 0 \text{ and } x_{im} = 1, \quad 1 \le m \le n\},$$

$$\mathcal{A}_1 = \{m : x_{0m} = 1 \text{ and } x_{im} = 0, \quad 1 \le m \le n\},$$

$$\mathcal{B}_0 = \{m : x_{0m} = 0 \text{ and } x_{jm} = 1, \quad 1 \le m \le n\},$$

$$\mathcal{B}_1 = \{m : x_{0m} = 1 \text{ and } x_{jm} = 0, \quad 1 \le m \le n\}.$$

Then from the properties of triply orthogonal codes,

$$|\mathcal{A}_0| = |\mathcal{A}_1| = |\mathcal{B}_0| = |\mathcal{B}_1| = \frac{n}{4},$$

and

$$|\mathcal{A}_0 \cap \mathcal{B}_0| = |\mathcal{A}_1 \cap \mathcal{B}_1| = \frac{n}{8}. \tag{2.25}$$

Thus,

$$\begin{aligned}
\mathrm{E}(S_i S_j) &= \mathrm{E}\left(\left(\sum_{\alpha\in\mathcal{A}_0} U_\alpha + \sum_{\alpha\in\mathcal{A}_1} U_\alpha\right)\left(\sum_{\beta\in\mathcal{B}_0} V_\beta + \sum_{\beta\in\mathcal{B}_1} V_\beta\right)\right) \\
&= \sum_{\alpha\in\mathcal{A}_0\cap\mathcal{B}_0} \mathrm{E}(U_\alpha^2) + \sum_{\alpha\in\mathcal{A}_0}\sum_{\substack{\beta\in\mathcal{B}_0\\ \beta\neq\alpha}} \mathrm{E}(U_\alpha)\mathrm{E}(U_\beta) + \sum_{\beta\in\mathcal{A}_1\cap\mathcal{B}_1} \mathrm{E}(V_\alpha^2) \\
&\quad + \sum_{\alpha\in\mathcal{A}_1}\sum_{\substack{\beta\in\mathcal{B}_1\\ \beta\neq\alpha}} \mathrm{E}(V_\alpha)\mathrm{E}(V_\beta) + \sum_{\alpha\in\mathcal{A}_0}\sum_{\beta\in\mathcal{B}_1} \mathrm{E}(U_\alpha)\mathrm{E}(V_\beta) + \sum_{\alpha\in\mathcal{A}_1}\sum_{\beta\in\mathcal{B}_0} \mathrm{E}(U_\beta)\mathrm{E}(V_\alpha) \\
&= \frac{n}{8}\left(\mathrm{E}(U^2) + \mathrm{E}(V^2)\right) + \left(\frac{n^2}{16} - \frac{n}{8}\right)\left(\mathrm{E}^2(U) + \mathrm{E}^2(V)\right) + \frac{n^2}{8}\mathrm{E}(U)\mathrm{E}(V),
\end{aligned}$$

and

$$\mathrm{E}(S_i)\mathrm{E}(S_j) = \frac{n^2}{16}\left(\mathrm{E}^2(U) + \mathrm{E}^2(V)\right) + \frac{n^2}{8}\mathrm{E}(U)\mathrm{E}(V).$$

Therefore, for $i \neq j$,

$$\begin{aligned}
\mathrm{Cov}(S_i, S_j) &= \mathrm{E}(S_i S_j) - \mathrm{E}(S_i)\mathrm{E}(S_j) \\
&= \frac{n}{8}\left(\mathrm{Var}(U) + \mathrm{Var}(V)\right).
\end{aligned}$$

Now define a new set of random variables $T_1, T_2, \ldots, T_{M-1}$ by

$$T_i = S_i - \frac{n}{4}\left(\mathrm{E}(U) + \mathrm{E}(V)\right).$$

All the T_i's have mean zero and covariance matrix

$$\mathrm{Cov}(\mathbf{T}) = \frac{n}{8}\left(\mathrm{Var}(U) + \mathrm{Var}(V)\right)\begin{bmatrix} 2 & 1 & \ldots & 1 \\ 1 & 2 & \ldots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \ldots & 2 \end{bmatrix}.$$

Similarly, for large M, T_i can be approximated by a normal random variable, which, from Lemma 2.4, is given by

$$T_i \approx \sqrt{\frac{n}{8}\left(\mathrm{Var}(U) + \mathrm{Var}(V)\right)}\,(X_i + X_0),$$

where the X_i's are i.i.d. $N(0,1)$ random variables. From (2.24), for large M, P_C can now be approximated by

$$\begin{aligned} P_C &\approx \Pr\left\{X_i + X_0 > \frac{-\frac{n}{4}\left(\mathrm{E}(U) + \mathrm{E}(V)\right)}{\sqrt{\frac{n}{8}\left(\mathrm{Var}(U) + \mathrm{Var}(V)\right)}}, \quad i = 1, 2, \ldots, M-1\right\} \\ &= \Pr\left\{X_i > \frac{-\frac{n}{4}\left(\mathrm{E}(U) + \mathrm{E}(V)\right)}{\sqrt{\frac{n}{8}\left(\mathrm{Var}(U) + \mathrm{Var}(V)\right)}} - X_0, \quad i = 1, 2, \ldots, M-1\right\} \end{aligned} \tag{2.26}$$

For Class I VNCs, the following are shown in Appendix 2.B:

$$\begin{aligned} \mathrm{E}(U) &= \sum_{y \in \mathcal{Y}} p(y|0) \ln \frac{p(y|0)}{p(y|1)} \\ &= \frac{\epsilon^2}{2} \cdot \left(\sum_{y \in \mathcal{Y}} \frac{1}{w(y)} \left(\sigma(0,y) - \sigma(1,y)\right)^2\right) + O(\epsilon^3), \end{aligned} \tag{2.27}$$

$$\begin{aligned} \mathrm{E}(V) &= \sum_{y \in \mathcal{Y}} p(y|1) \ln \frac{p(y|1)}{p(y|0)} \\ &= \frac{\epsilon^2}{2} \cdot \left(\sum_{y \in \mathcal{Y}} \frac{1}{w(y)} \left(\sigma(0,y) - \sigma(1,y)\right)^2\right) + O(\epsilon^3), \end{aligned} \tag{2.28}$$

$$\begin{aligned}\mathrm{Var}(U) &= \sum_{y\in\mathcal{Y}} \sum_{\substack{y'\in\mathcal{Y}\\ y'>y}} p(y|0)p(y'|0)\ln^2\frac{p(y|0)p(y'|1)}{p(y|1)p(y'|0)} \\ &= \epsilon^2\cdot\left(\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)+O(\epsilon^3), \qquad (2.29)\end{aligned}$$

$$\begin{aligned}\mathrm{Var}(V) &= \sum_{y\in\mathcal{Y}} \sum_{\substack{y'\in\mathcal{Y}\\ y'>y}} p(y|1)p(y'|1)\ln^2\frac{p(y|1)p(y'|0)}{p(y|0)p(y'|1)} \\ &= \epsilon^2\cdot\left(\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)+O(\epsilon^3). \qquad (2.30)\end{aligned}$$

For convenience, let D denote the quantity $\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2$. Then

$$\begin{aligned}P_C &\approx \Pr\left\{X_i > \frac{-\frac{n}{4}\left(\epsilon^2 D+O(\epsilon^3)\right)}{\sqrt{\frac{n}{4}\left(\epsilon^2 D+O(\epsilon^3)\right)}}-X_0,\quad i=1,2,\ldots,M-1\right\} \\ &= \Pr\left\{X_i > -\sqrt{n}\cdot\frac{\frac{\epsilon}{2}\sqrt{D}+O(\epsilon^2)}{\sqrt{1+O(\epsilon)}}-X_0,\quad i=1,2,\ldots,M-1\right\} \\ &= \int_{-\infty}^{\infty} Q\left(-\sqrt{n}\cdot\frac{\frac{\epsilon}{2}\sqrt{D}+O(\epsilon^2)}{\sqrt{1+O(\epsilon)}}-x\right)^{M-1} Z(x)\,dx \\ &= \int_{-\infty}^{\infty} P\left(\sqrt{n}\cdot\frac{\frac{\epsilon}{2}\sqrt{D}+O(\epsilon^2)}{\sqrt{1+O(\epsilon)}}+x\right)^{M-1} Z(x)\,dx, \qquad (2.31)\end{aligned}$$

where

$$Z(x)=\frac{1}{\sqrt{2\pi}}e^{-x^2/2},$$

$$P(x)=\int_{-\infty}^{x} Z(t)\,dt,$$

$$Q(x)=\int_{x}^{\infty} Z(t)\,dt = 1-P(x)=P(-x).$$

Now we want to investigate necessary and sufficient conditions for $P_C\to 1$ as $M\to\infty$. Since z decreases as M increases, it is understood that ϵ is a decreasing function of M. From (2.31), a necessary condition for $P_C\to 1$ is

$$\lim_{M\to\infty}\epsilon\sqrt{n}=\lim_{M\to\infty}\epsilon\sqrt{2M}=\infty. \qquad (2.32)$$

If (2.32) holds, for each finite x,

$$\lim_{M\to\infty} P\left(\sqrt{n}\cdot\frac{\frac{\epsilon}{2}\sqrt{D}+O(\epsilon^2)}{\sqrt{1+O(\epsilon)}}+x\right)^{M-1} = \lim_{M\to\infty} P\left(\frac{\epsilon}{2}\sqrt{D}\cdot\sqrt{n}\right)^{M}.$$

It follows that

$$\lim_{M\to\infty} P_C = \lim_{M\to\infty} P\left(\frac{\epsilon}{2}\sqrt{D}\cdot\sqrt{n}\right)^{M}.$$

By using the approximation $P(x)\approx 1-Z(x)/x$ as $x\to\infty$,

$$\lim_{M\to\infty} P_C = \lim_{M\to\infty}\left(1-\frac{Z\left(\frac{\epsilon}{2}\sqrt{D}\cdot\sqrt{n}\right)}{\frac{\epsilon}{2}\sqrt{D}\cdot\sqrt{n}}\right)^{M}.$$

Since it is easier to deal with $\ln P_C$, we have

$$\begin{aligned}\lim_{M\to\infty}\ln P_C &= \lim_{M\to\infty}\left(M\ln\left(1-\frac{Z\left(\frac{\epsilon}{2}\sqrt{D}\cdot\sqrt{n}\right)}{\frac{\epsilon}{2}\sqrt{D}\cdot\sqrt{n}}\right)\right)\\ &= \lim_{M\to\infty}\left(-\frac{2M}{\epsilon\sqrt{D}\cdot\sqrt{n}}Z\left(\frac{\epsilon}{2}\sqrt{D}\cdot\sqrt{n}\right)\right)\\ &= \lim_{M\to\infty}\left(-\frac{1}{\sqrt{2\pi}}\cdot\frac{2M}{\epsilon\sqrt{D}\cdot\sqrt{n}}e^{-\epsilon^2 Dn/8}\right).\end{aligned}$$

The condition (2.32) is satisfied for the case of moderate noise since

$$\epsilon^2\cdot n = \frac{\epsilon^2}{z}\cdot\lambda\cdot\log_2 M \sim k_1\cdot\lambda\cdot\log_2 M, \qquad \text{for large } M.$$

Also

$$\begin{aligned}\lim_{M\to\infty}\ln P_C &= \lim_{M\to\infty}\left(-\frac{2}{\sqrt{2\pi D}}\cdot\frac{M}{\sqrt{k_1\lambda\log_2 M}}e^{-\frac{\lambda k_1 D}{8\ln 2}\cdot\ln M}\right)\\ &= \lim_{M\to\infty}\left(-\frac{2}{\sqrt{2\pi D}}\cdot\frac{M^{1-\frac{\lambda k_1 D}{8\ln 2}}}{\sqrt{k_1\lambda\log_2 M}}\right)\\ &= \lim_{M\to\infty}\left(-\frac{2}{\sqrt{2\pi D}}\cdot\frac{M^{1-\frac{\lambda}{\lambda_C}}}{\sqrt{k_1\lambda\log_2 M}}\right)\end{aligned}$$

by recognizing that $\lambda_C = 8\ln 2/k_1 D$. Finally,

$$\lim_{M\to\infty}\ln P_C = \begin{cases} 0, & \text{if } \lambda>\lambda_C,\\ -\infty, & \text{if } \lambda<\lambda_C,\end{cases}$$

which is equivalent to

$$\lim_{M\to\infty} P_E = \begin{cases} 0, & \text{if } \lambda > \lambda_C, \\ 1, & \text{if } \lambda < \lambda_C. \end{cases}$$

From the symmetry of triply orthogonal codes, it follows that P_E will remain the same if any other codeword $\mathbf{x}_i$ other than $\mathbf{x}_0$ is transmitted. ∎

Corollary 2.1 *For symmetric binary-input Class I VNCs, if moderate noise is assumed, the family of orthogonal codes achieves a minimum resource per information bit of* λ_C.

Proof. If the channel is symmetric, then

$$\begin{aligned} \mathrm{E}(U) &= \sum_{y\in\mathcal{Y}} p(y|0) \ln \frac{p(y|0)}{p(y|1)} \\ &= \sum_{y\in\mathcal{Y}} p(y|1) \ln \frac{p(y|1)}{p(y|0)} \\ &= \mathrm{E}(V) \end{aligned}$$

because there exists $y' \in \mathcal{Y}$ such that $p(y|0) = p(y'|1)$ and $p(y|1) = p(y'|0)$ for every $y \in \mathcal{Y}$. Similarly,

$$\begin{aligned} \mathrm{Var}(U) &= \sum_{y\in\mathcal{Y}} \sum_{\substack{y'\in\mathcal{Y} \\ y'>y}} p(y|0)p(y'|0) \ln^2 \frac{p(y|0)p(y'|1)}{p(y|1)p(y'|0)} \\ &= \sum_{y\in\mathcal{Y}} \sum_{\substack{y'\in\mathcal{Y} \\ y'>y}} p(y|1)p(y'|1) \ln^2 \frac{p(y|1)p(y'|0)}{p(y|0)p(y'|1)} \\ &= \mathrm{Var}(V). \end{aligned}$$

If orthogonal codes are used instead of triply orthogonal codes, then, instead of (2.25), we get

$$|\mathcal{A}_0 \cap \mathcal{B}_0| + |\mathcal{A}_1 \cap \mathcal{B}_1| = \frac{n}{4}.$$

Following similar derivations for the covariance matrix of S_i, we obtain

$$\mathrm{Cov}(S_i S_j) = \begin{cases} \frac{n}{2}\mathrm{Var}(U) & i = j \\ \frac{n}{4}\mathrm{Var}(U) & i \neq j. \end{cases}$$

The rest of the proof for the theorem still works. ∎

Since a very noisy binary symmetric channel is of Class I, the theorem proved in [2] is a special case of this corollary.

2.6 Remarks about Binary-Input Class II Channels

In the previous section, we show that triply orthogonal codes achieve λ_C for binary-input Class I channels and orthogonal codes also achieve λ_C if the channels are symmetric. Similar results do not hold for general Class II VNCs since the capacity in (2.5) is usually not achieved with equiprobable inputs. The following examples show that orthogonal codes (or triply orthogonal codes) do not achieve λ_C for some symmetric binary-input Class II VNCs.

Consider the VNC with the transition probability matrix given by

$$\mathbf{P}_{Y|X} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} + \epsilon \cdot \begin{bmatrix} 2 & -3 & 1 \\ 1 & -3 & 2 \end{bmatrix}.$$

By our definition, it is a symmetric Class II VNC. If moderate noise is assumed, then from (2.9)

$$\begin{aligned} \lambda_C &= \frac{\ln 2}{k_2} \cdot \left(\sum_{y \in \mathcal{Y}_2} \sigma(0,y) \ln \frac{2\sigma(0,y)}{\sigma(0,y)+\sigma(1,y)} \right)^{-1} \\ &= \frac{\ln 2}{k_2} \left(5 \ln 2 - 3 \ln 3\right)^{-1} \\ &\approx \frac{4.079}{k_2}, \end{aligned}$$

and from (2.14)

$$\begin{aligned} \lambda_0 &= \frac{4 \ln 2}{k_2} \cdot \left(\sum_{y \in \mathcal{Y}_2} \left(\sqrt{\sigma(0,y)} - \sqrt{\sigma(1,y)} \right)^2 \right)^{-1} \\ &= \frac{2 \ln 2}{k_2} \left(3 - 2\sqrt{2}\right)^{-1} \\ &\approx \frac{8.080}{k_2}. \end{aligned}$$

From Theorem 2.1, we know that orthogonal codes can be used to achieve λ_0. Since, for this channel, no transition probability $p(x|y) = 0$, the same method used in proving Theorem 2.2 can be applied here. Proceeding as in the last section, we have (2.26):

$$P_C \approx \Pr\left\{X_i > \frac{-\frac{n}{4}\left(\mathrm{E}(U)+\mathrm{E}(V)\right)}{\sqrt{\frac{n}{8}\left(\mathrm{Var}(U)+\mathrm{Var}(V)\right)}} - X_0, \quad i = 1,2,\ldots,M-1\right\},$$

where $X_0, X_1, \ldots, X_{M-1}$ are i.i.d. $N(0,1)$ random variables, and the means and variances of U and V are found by

$$\begin{aligned}\mathrm{E}(U) &= \mathrm{E}(V) = \sum_{y\in y} p(y|0)\ln\frac{p(y|0)}{p(y|1)} \\ &= \epsilon\cdot(2\ln 2 - \ln 2) = \epsilon\cdot\ln 2,\end{aligned}$$

and

$$\begin{aligned}\mathrm{Var}(U) &= \mathrm{Var}(V) = \mathrm{E}(U^2) - \mathrm{E}^2(U) \\ &= \epsilon\cdot\left(2\ln^2 2 + + \ln^2 2\right) - (\epsilon\cdot\ln 2)^2 = \epsilon\cdot 3\ln^2 2 + O(\epsilon^2).\end{aligned}$$

Similar to (2.31),

$$\begin{aligned}P_C &\approx \Pr\left\{X_i > -\sqrt{n}\cdot\frac{\sqrt{\epsilon/3}}{\sqrt{1+O(\epsilon)}} - X_0, \quad i = 1,2,\ldots,M-1\right\} \\ &= \int_{-\infty}^{\infty} P\left(\sqrt{n}\cdot\frac{\sqrt{\epsilon/3}}{\sqrt{1+O(\epsilon)}} + x\right)^{M-1} Z(x)\,dx.\end{aligned}$$

A necessary condition for $P_C \to 1$ as $M \to \infty$ is

$$\lim_{M\to\infty}\sqrt{\epsilon}\sqrt{n} = \infty,$$

which is satisfied for the case of moderate noise since, for large M,

$$\epsilon\cdot n = \frac{\epsilon}{z}\cdot\lambda\cdot\log_2 M \sim k_2\cdot\lambda\cdot\log_2 M.$$

Following similar derivations as in the last section, we obtain

$$\begin{aligned}
\lim_{M\to\infty} \ln P_C &= \lim_{M\to\infty} \left(M \ln \left(P\left(\sqrt{\epsilon/3} \cdot \sqrt{n} \right) \right) \right) \\
&= \lim_{M\to\infty} \left(M \ln \left(1 - \frac{Z\left(\sqrt{\epsilon/3} \cdot \sqrt{n} \right)}{\sqrt{\epsilon/3} \cdot \sqrt{n}} \right) \right) \\
&= \lim_{M\to\infty} \left(-\frac{1}{\sqrt{2\pi}} \cdot \frac{\sqrt{3}M}{\sqrt{\epsilon}\sqrt{n}} e^{-\epsilon n/6} \right) \\
&= \lim_{M\to\infty} \left(-\sqrt{\frac{3}{2\pi}} \cdot \frac{M^{1-\frac{\lambda k_2}{6\ln 2}}}{\sqrt{k_2 \lambda \log_2 M}} \right),
\end{aligned}$$

which yields

$$\lim_{M\to\infty} P_C = \begin{cases} 1, & \text{if } \lambda > \frac{6\ln 2}{k_2}, \\ 0, & \text{if } \lambda < \frac{6\ln 2}{k_2}. \end{cases}$$

Since $6\ln 2/k_2 \approx 4.159/k_2$, $\lambda_C < 6\ln 2/k_2$. Therefore,

$$\lim_{M\to\infty} P_E = 1 \quad \text{if } \lambda_C < \lambda < \frac{6\ln 2}{k_2}$$

when orthogonal codes (or triply orthogonal codes) are used on this channel.

Remark. In general, for symmetric binary-input Class II VNCs, if the following is satisfied:

$$\sigma(x,y) \neq 0 \quad \text{for all } x \in \mathcal{X} \text{ and } y \in \mathcal{Y}_2,$$

then we can apply the same method as in the last section and obtain (2.26). The means and variances of U and V are now given by

$$\begin{aligned}
\mathrm{E}(U) &= \mathrm{E}(V) \\
&= \sum_{y\in\mathcal{Y}} p(y|0) \ln \frac{p(y|0)}{p(y|1)} \\
&= \epsilon \cdot \left(\sum_{y\in\mathcal{Y}_2} \sigma(0,y) \ln \frac{\sigma(0,y)}{\sigma(1,y)} \right) + O(\epsilon^2),
\end{aligned}$$

and

$$\mathrm{Var}(U) = \mathrm{Var}(V)$$

$$
\begin{aligned}
&= \mathrm{E}(U^2) - \mathrm{E}^2(U) \\
&= \epsilon \cdot \left(\sum_{y \in \mathcal{Y}_2} \sigma(0,y) \ln^2 \frac{\sigma(0,y)}{\sigma(1,y)} \right) + O(\epsilon^2).
\end{aligned}
$$

Following similar derivations, for the case of moderate noise, we can show that

$$
\lim_{M \to \infty} P_E = \begin{cases} 0, & \text{if } \lambda > \lambda_\star, \\ 1, & \text{if } \lambda < \lambda_\star, \end{cases}
$$

where $\lambda_\star$ is given by

$$
\lambda_\star = \frac{2 \ln 2}{k_2} \cdot \left(\sum_{y \in \mathcal{Y}_2} \sigma(0,y) \ln^2 \frac{\sigma(0,y)}{\sigma(1,y)} \right) \cdot \left(\sum_{y \in \mathcal{Y}_2} \sigma(0,y) \ln \frac{\sigma(0,y)}{\sigma(1,y)} \right)^{-2}.
$$

Comparing $\lambda_\star$ and λ_C given in (2.9), from Shannon's theorem, we have $\lambda_\star > \lambda_C$ in general. Hence if orthogonal codes (or triply orthogonal codes) are used on this type of Class II VNCs, then

$$
\lim_{M \to \infty} P_E = 1 \quad \text{if } \lambda_C < \lambda < \lambda_\star.
$$

Results similar to Theorem 2.2 or Corollary 2.1 do not hold for Class II VNCs.

2.7 Generalization to Symmetric Class I Channels with More than Two Inputs

Consider a symmetric Class I VNC with s inputs, where $s > 2$. Without loss of generality, let the input alphabet $\mathcal{X} = \{0, 1, \ldots, s-1\}$ and the output alphabet $\mathcal{Y} = \{0, 1, \ldots, L-1\}$. If the input probability is denoted by $p(x)$ for all $x \in \mathcal{X}$, from [3] the mutual information is given by

$$
I(X;Y) = \frac{\epsilon^2}{2 \ln 2} \cdot \left(\sum_{y \in \mathcal{Y}} \frac{1}{w(y)} \left(\sum_{x \in \mathcal{X}} p(x) \sigma^2(x,y) - \left(\sum_{x \in \mathcal{X}} p(x) \sigma(x,y) \right)^2 \right) \right) + O(\epsilon^3).
$$

Since the channel is symmetric, the capacity is achieved with uniform input probabilities. Setting $p(x) = 1/s$ for all $x \in \mathcal{X}$ in $I(X;Y)$, we obtain

$$
C = \frac{\epsilon^2}{2 \ln 2} \cdot \left(\sum_{y \in \mathcal{Y}} \frac{1}{w(y)} \left(\frac{1}{s} \sum_{x \in \mathcal{X}} \sigma^2(x,y) - \left(\frac{1}{s} \sum_{x \in \mathcal{X}} \sigma(x,y) \right)^2 \right) \right) + O(\epsilon^3)
$$

$$= \frac{\epsilon^2}{2\ln 2} \cdot \left(\sum_{y\in\mathcal{Y}} \frac{1}{w(y)} \left(\frac{s-1}{s^2} \sum_{x\in\mathcal{X}} \sigma^2(x,y) - \frac{2}{s^2} \sum_{x\in\mathcal{X}} \sum_{\substack{x'\in\mathcal{X} \\ x'>x}} \sigma(x,y)\sigma(x',y) \right) \right) + O(\epsilon^3)$$

$$= \frac{\epsilon^2}{2s^2\ln 2} \cdot \left(\sum_{y\in\mathcal{Y}} \frac{1}{w(y)} \sum_{x\in\mathcal{X}} \sum_{\substack{x'\in\mathcal{X} \\ x'>x}} (\sigma(x,y) - \sigma(x',y))^2 \right) + O(\epsilon^3).$$

Thus, if moderate noise is assumed, then

$$\lambda_C = \frac{2s^2\ln 2}{k_1} \cdot \left(\sum_{y\in\mathcal{Y}} \frac{1}{w(y)} \sum_{x\in\mathcal{X}} \sum_{\substack{x'\in\mathcal{X} \\ x'>x}} (\sigma(x,y) - \sigma(x',y))^2 \right)^{-1}.$$

The following lemma is similar to Lemma 2.4.

Lemma 2.5 *Let $X_0, X_1, \ldots, X_{M-1}$ be i.i.d. $N(0,1)$ random variables. If Y_0, Y_1, $\ldots$,Y_{M-1} are defined by*

$$Y_i = \alpha X_i + \beta X_0, \quad \text{for } i = 1, 2, \ldots, M-1,$$

then the Y_i's are normal random variables with mean 0 and covariance

$$\mathrm{Cov}(Y_i Y_j) = \begin{cases} \alpha^2 + \beta^2, & \text{if } i = j, \\ \beta^2, & \text{if } i \neq j. \end{cases}$$

Recall that for arbitrary $s = s_1 s_2 \cdots s_u$, s_i a prime power, we have constructed a sequence of generalized triply orthogonal codes with s symbols, length $n = s^{3m}$ and $M = \min(s_1^m+1, s_2^m+1, \ldots, s_u^m+1)$ codewords for $m = 1, 2, \ldots$. The following theorem is a generalization of Theorem 2.2.

Theorem 2.3 *For symmetric Class I VNCs with s inputs, if moderate noise is assumed, the family of generalized triply orthogonal codes of s symbols achieves λ_C, which is the minimum resource per information bit needed as $z \to 0$.*

Proof. Let $C = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{M-1}\}$ be a generalized triply orthogonal code of length n and s symbols. This code is used on a symmetric Class I VNC with s

inputs. Suppose $\mathbf{x}_0$ is transmitted and $\mathbf{y}$ is the received. From the symmetry of generalized triply orthogonal codes, the decoder error probability is independent of which codeword is transmitted. A maximum likelihood decoder will successfully recover the originally transmitted codeword if and only if

$$p(\mathbf{y}|\mathbf{x}_0) > p(\mathbf{y}|\mathbf{x}_i), \qquad \text{for } i = 1, 2, \ldots, M-1.$$

After cancellations and taking logarithms, we obtain

$$\sum_{y\in\mathcal{Y}} \sum_{l\in\mathcal{X}} \sum_{\substack{m\in\mathcal{X} \\ m\neq l}} N_i^{l,m}(y) \ln \frac{p(y|l)}{p(y|m)} > 0, \qquad \text{for } i = 1, 2, \ldots, M-1, \tag{2.33}$$

where $N_i^{l,m}(y)$ is defined to be the number of components of $\mathbf{y}$ equal to y when the corresponding components of $\mathbf{x}_0$ and $\mathbf{x}_i$ are l and m, respectively. The value of $\ln(p(y|l)/p(y|m))$ always exists because this channel is of Class I. Since codewords of generalized triply orthogonal codes are rows of orthogonal arrays of strength 3,

$$\sum_{y\in\mathcal{Y}} N_i^{l,m}(y) = \frac{n}{s^2}, \qquad \text{for } l, m \in \mathcal{X} \text{ and } l \neq m. \tag{2.34}$$

Define $s(s-1)$ sets of i.i.d. random variables by their common probability distributions: [2]

$$\Pr\left(U^{l,m} = \ln \frac{p(y|l)}{p(y|m)}\right) = p(y|l), y \in \mathcal{Y},$$

where $l, m \in \mathcal{X}$ and $l \neq m$. If we define

$$S_i = \sum_{l\in\mathcal{X}} \sum_{\substack{m\in\mathcal{X} \\ m\neq l}} \langle \mathbf{U}^{l,m}, \mathbf{g}^{l,m}(\mathbf{x}_0, \mathbf{x}_i)\rangle, \qquad \text{for } i = 1, 2, \ldots, M-1, \tag{2.35}$$

where $\mathbf{U}^{l,m} = (U_1^{l,m}, U_2^{l,m}, \ldots, U_n^{l,m})$ and $\mathbf{g}^{l,m} = (g_1^{l,m}, g_2^{l,m}, \ldots, g_n^{l,m})$ are defined by

$$g_t^{l,m}(\mathbf{a}, \mathbf{b}) = \begin{cases} 1, & \text{if } a_t = l \text{ and } b_t = m, \\ 0, & \text{otherwise,} \end{cases}$$

[2] If $\ln(p(y|l)/p(y|m)) = \ln(p(y'|l)/p(y'|m))$ for some $y \neq y'$, then we define $\Pr(U^{l,m} = \ln(p(y|l)/p(y|m))) = p(y|l) + p(y'|l)$. All the derivations still hold.

for $t = 1, 2, \ldots, n$, then from (2.33) the probability of correct decoding becomes

$$P_C = \Pr\{S_i > 0, \quad i = 1, 2, \ldots, M-1\}. \tag{2.36}$$

Since S_i is the sum of $s(s-1)$ i.i.d. random variables, for large M, and hence for large n, P_C can be computed via the central limit theorem.

Now we want to find the first and second moments of the S_i's. By the definition of $U^{l,m}$, similar to (2.19), (2.20), (2.21), and (2.22),

$$\mathrm{E}(U^{l,m}) = \sum_{y \in \mathcal{Y}} p(y|l) \ln \frac{p(y|l)}{p(y|m)}, \tag{2.37}$$

and

$$\mathrm{Var}(U^{l,m}) = \sum_{y \in \mathcal{Y}} \sum_{\substack{y' \in \mathcal{Y} \\ y' > y}} p(y|l)p(y'|m) \ln^2 \frac{p(y|l)p(y'|m)}{p(y|m)p(y'|l)}. \tag{2.38}$$

Also from (2.34) and (2.35), for $i = 1, 2, \ldots, M-1$,

$$\mathrm{E}(S_i) = \frac{n}{s^2} \sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m \neq l}} \mathrm{E}(U^{l,m}) \tag{2.39}$$

and

$$\mathrm{Var}(S_i) = \frac{n}{s^2} \sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m \neq l}} \mathrm{Var}(U^{l,m}). \tag{2.40}$$

From the properties of generalized triply orthogonal codes, we show in Appendix 2.C that, for any $i \neq j$,

$$\mathrm{Cov}(S_i, S_j) = \frac{n}{s^3} \left(\sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m \neq l}} \mathrm{Var}(U^{l,m}) + \sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m \neq l}} \sum_{\substack{t \in \mathcal{X} \\ t \neq m \\ t \neq l}} \mathrm{Cov}(U^{l,m}, U^{l,t}) \right). \tag{2.41}$$

Since $\mathrm{E}(S_i)$, $\mathrm{Var}(S_i)$, and $\mathrm{Cov}(S_i, S_j)$ are independent of i and $j (\neq i)$, we change the notation to $\mathrm{E}(S)$, $\mathrm{Var}(S)$, and $\mathrm{Cov}(S, S')$. Define T_i, $i = 1, 2, \ldots, M-1$, by

$$T_i = S_i - \mathrm{E}(S).$$

Then the T_i's have mean zero and covariance matrix the same as that of the S_i's. From Lemma 2.5 and the central limit theorem, for large M, T_i can be approximated by

$$T_i \approx \alpha X_i + \beta X_0,$$

where the X_i's are $N(0,1)$ random variables and

$$\alpha = \sqrt{\mathrm{Var}(S) - \mathrm{Cov}(S, S')},$$

$$\beta = \sqrt{\mathrm{Cov}(S, S')}.$$

Thus, from (2.36),

$$\begin{aligned} P_C &\approx \Pr\left\{X_i > -\frac{\mathrm{E}(S) + \beta X_0}{\alpha}, \quad i = 1, 2, \ldots, M-1\right\} \\ &= \int_{-\infty}^{\infty} P\left(\frac{\mathrm{E}(S)}{\alpha} + \frac{\beta}{\alpha} x\right)^{M-1} Z(x)\, dx. \end{aligned} \tag{2.42}$$

Since the channel is of Class I, similar to (2.27), (2.28), (2.29), and (2.30), for $l \neq m$, (2.37) and (2.38) can be reduced to

$$\mathrm{E}(U^{l,m}) = \frac{\epsilon^2}{2} \cdot \left(\sum_{y \in \mathcal{Y}} \frac{1}{w(y)} \left(\sigma(l,y) - \sigma(m,y)\right)^2\right) + O(\epsilon^3),$$

and

$$\mathrm{Var}(U^{l,m}) = \epsilon^2 \cdot \left(\sum_{y \in \mathcal{Y}} \frac{1}{w(y)} \left(\sigma(l,y) - \sigma(m,y)\right)^2\right) + O(\epsilon^3).$$

Similarly, for $l \neq m \neq t$,

$$\mathrm{Cov}(U^{l,m}, U^{l,t}) = \epsilon^2 \cdot \left(\sum_{y \in \mathcal{Y}} \frac{1}{w(y)} \left(\sigma(l,y) - \sigma(m,y)\right)\left(\sigma(l,y) - \sigma(t,y)\right)\right) + O(\epsilon^3).$$

Therefore,

$$\mathrm{E}(S) = \frac{n\epsilon^2}{s^2} \cdot \left(\sum_{y \in \mathcal{Y}} \frac{1}{w(y)} \sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m > l}} \left(\sigma(l,y) - \sigma(m,y)\right)^2\right) + O(\epsilon^3),$$

$$\mathrm{Var}(S) = \frac{2n\epsilon^2}{s^2} \cdot \left(\sum_{y \in y} \frac{1}{w(y)} \sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m > l}} (\sigma(l,y) - \sigma(m,y))^2 \right) + O(\epsilon^3),$$

and

$$\begin{aligned}
\mathrm{Cov}(S, S') &= \beta^2 \\
&= \frac{2n\epsilon^2}{s^3} \cdot \left(\sum_{y \in y} \frac{1}{w(y)} \left(\sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m > l}} (\sigma(l,y) - \sigma(m,y))^2 \right.\right. \\
&\qquad \left.\left. + \sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m \neq l}} \sum_{\substack{t \in \mathcal{X} \\ t > m \\ t \neq l}} (\sigma(l,y) - \sigma(m,y))\,(\sigma(l,y) - \sigma(t,y)) \right) \right) + O(\epsilon^3).
\end{aligned}$$

Also

$$\begin{aligned}
\alpha^2 &= \mathrm{Var}(S) - \mathrm{Cov}(S, S') \\
&= \frac{2n\epsilon^2}{s^3} \cdot \left(\sum_{y \in y} \frac{1}{w(y)} \left(\sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m > l}} (s-1)\,(\sigma(l,y) - \sigma(m,y))^2 \right.\right. \\
&\qquad \left.\left. - \sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m \neq l}} \sum_{\substack{t \in \mathcal{X} \\ t > m \\ t \neq l}} (\sigma(l,y) - \sigma(m,y))\,(\sigma(l,y) - \sigma(t,y)) \right) \right) + O(\epsilon^3) \\
&= \frac{2n\epsilon^2}{s^3} \cdot \left(\sum_{y \in y} \frac{1}{w(y)} \left(\frac{s(s-1)}{2} \sum_{l \in \mathcal{X}} \sigma^2(l,y) - s \sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m > l}} \sigma(l,y)\sigma(m,y) \right) \right) + O(\epsilon^3) \\
&= \frac{n\epsilon^2}{s^2} \cdot \left(\sum_{y \in y} \frac{1}{w(y)} \left(\sum_{l \in \mathcal{X}} (s-1)\sigma^2(l,y) - \sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m > l}} 2\sigma(l,y)\sigma(m,y) \right) \right) + O(\epsilon^3) \\
&= \frac{n\epsilon^2}{s^2} \cdot \left(\sum_{y \in y} \frac{1}{w(y)} \sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m > l}} (\sigma(l,y) - \sigma(m,y))^2 \right) + O(\epsilon^3).
\end{aligned}$$

Let D denote the quantity $\sum_{y \in y} \frac{1}{w(y)} \sum_{l \in \mathcal{X}} \sum_{\substack{m \in \mathcal{X} \\ m > l}} (\sigma(l,y) - \sigma(m,y))^2$. Then

$$\frac{\mathrm{E}(S)}{\beta} = \frac{\sqrt{n}}{s} \cdot \frac{\epsilon^2 D + O(\epsilon^3)}{\sqrt{\epsilon^2 D + O(\epsilon^3)}} = \frac{\sqrt{n}}{s} \cdot \frac{\epsilon\sqrt{D} + O(\epsilon^2)}{\sqrt{1 + O(\epsilon)}}.$$

Note that both the numerator and the denominator of β/α are of the order $\sqrt{n}\cdot\epsilon$. From (2.42), a necessary condition for $P_C \to 1$ as $M \to \infty$ is

$$\lim_{M\to\infty} \epsilon\sqrt{n} = \infty,$$

which is satisfied for the case of moderate noise because, for large M,

$$\epsilon^2 \cdot n = \frac{\epsilon^2}{z} \cdot \lambda \cdot \log_2 M \sim k_1 \cdot \lambda \cdot \log_2 M.$$

For each finite x,

$$\begin{aligned}\lim_{M\to\infty} P\left(\frac{\mathrm{E}(S)}{\alpha} + \frac{\beta}{\alpha}x\right)^{M-1} &= \lim_{M\to\infty} P\left(\frac{\mathrm{E}(S)}{\alpha}\right)^{M} \\ &= \lim_{M\to\infty} P\left(\frac{\epsilon}{s}\sqrt{D}\cdot\sqrt{n}\right)^{M}.\end{aligned}$$

Therefore,

$$\begin{aligned}\lim_{M\to\infty} \ln P_C &= \lim_{M\to\infty} \left(M \ln\left(P\left(\epsilon\sqrt{D}/s\cdot\sqrt{n}\right)\right)\right) \\ &= \lim_{M\to\infty} \left(M \ln\left(1 - \frac{Z\left(\epsilon\sqrt{D}/s\cdot\sqrt{n}\right)}{\epsilon\sqrt{D}/s\cdot\sqrt{n}}\right)\right) \\ &= \lim_{M\to\infty} \left(-\frac{1}{\sqrt{2\pi}} \cdot \frac{sM}{\epsilon\sqrt{D}\cdot\sqrt{n}} e^{-\epsilon^2 Dn/2s^2}\right) \\ &= \lim_{M\to\infty} \left(-\frac{s}{\sqrt{2\pi D}} \cdot \frac{M^{1-\frac{\lambda k_1 D}{2s^2 \ln 2}}}{\sqrt{k_1\lambda\log_2 M}}\right) \\ &= \lim_{M\to\infty} \left(-\frac{s}{\sqrt{2\pi D}} \cdot \frac{M^{1-\frac{\lambda}{\lambda_C}}}{\sqrt{k_1\lambda\log_2 M}}\right)\end{aligned}$$

by recognizing that $\lambda_C = 2s^2 \ln 2/k_1 D$. Finally,

$$\lim_{M\to\infty} P_C = \begin{cases} 1, & \text{if } \lambda > \lambda_C, \\ 0, & \text{if } \lambda < \lambda_C, \end{cases}$$

which completes the proof. ∎

2.8 Discussions

In Section 2.6, we find counterexamples showing that results similar to Corollary 2.1 do not hold in general for symmetric Class II channels. However, one can show that if moderate noise is assumed, the family of orthogonal codes can be used to achieve λ_C for the very noisy binary erasure channel (which is of Class II). (The proof is omitted from this thesis.) This makes one suspect that results similar to Corollary 2.1 may hold for subclasses of symmetric Class II channels.

In Section 2.3, we construct a sequence of orthogonal arrays of strength 3 for an arbitrary number of symbols. However, the number of rows in the construction is away from the upper bound. We think that, by different choices of the matrix **C** in Lemma 2.2, one can make an improvement over the given construction.

Appendix 2.A Derivations of (2.11) and (2.12)

The computation of R_0 requires a longer expansion of $p(y|x)$:

$$p(y|x) = w(y) + \epsilon \cdot \sigma(x,y) + \frac{\epsilon^2}{2} \cdot \eta(x,y) + O(\epsilon^3),$$

where the $\eta(x,y)$'s satisfy the same condition as the $\sigma(x,y)$'s:

$$\sum_{y \in \mathcal{Y}} \eta(x,y) = 0, \quad \text{for all } x \in \mathcal{X}. \tag{2.43}$$

However, as we will see, the final expression for the first order approximation for R_0 does not contain the new term $\eta(x,y)$. For Class I VNCs, since $\mathcal{Y}_2 = \emptyset$,

$$\begin{aligned} & \sum_{y \in \mathcal{Y}} \sqrt{p(y|0)p(y|1)} \\ = & \sum_{y \in \mathcal{Y}} \sqrt{w(y) + \epsilon \cdot \sigma(0,y) + \frac{\epsilon^2}{2} \cdot \eta(0,y) + O(\epsilon^3)} \\ & \cdot \sqrt{w(y) + \epsilon \cdot \sigma(1,y) + \frac{\epsilon^2}{2} \cdot \eta(1,y) + O(\epsilon^3)} \end{aligned}$$

$$= \sum_{y\in\mathcal{Y}} w(y)\left(1+\epsilon\cdot\frac{1}{w(y)}\left(\sigma(0,y)+\sigma(1,y)\right)+\epsilon^2\cdot\left(\frac{1}{w^2(y)}\cdot\sigma(0,y)\sigma(1,y)\right.\right.$$
$$\left.\left.+\frac{1}{2w(y)}\left(\eta(0,y)+\eta(1,y)\right)\right)+O(\epsilon^3)\right)^{\frac{1}{2}}.$$

Using $(1+\delta)^{\frac{1}{2}} \sim 1+\delta/2-\delta^2/8$ as $\delta\to 0$, (2.3), (2.4), and (2.43), we have

$$\sum_{y\in\mathcal{Y}}\sqrt{p(y|0)p(y|1)}$$
$$= \sum_{y\in\mathcal{Y}} w(y)\left(1+\frac{\epsilon}{2}\cdot\frac{1}{w(y)}\left(\sigma(0,y)+\sigma(1,y)\right)+\frac{\epsilon^2}{2}\cdot\left(\frac{1}{w^2(y)}\cdot\sigma(0,y)\sigma(1,y)\right.\right.$$
$$\left.\left.+\frac{1}{2w(y)}\left(\eta(0,y)+\eta(1,y)\right)\right)-\frac{\epsilon^2}{8}\cdot\frac{1}{w^2(y)}\left(\sigma(0,y)+\sigma(1,y)\right)^2+O(\epsilon^3)\right)$$
$$= \sum_{y\in\mathcal{Y}} w(y)+\frac{\epsilon}{2}\cdot\sum_{y\in\mathcal{Y}}\left(\sigma(0,y)+\sigma(1,y)\right)+\frac{\epsilon^2}{2}\cdot\left(\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\cdot\sigma(0,y)\sigma(1,y)\right.$$
$$\left.+\frac{1}{2}\sum_{y\in\mathcal{Y}}\left(\eta(0,y)+\eta(1,y)\right)\right)-\frac{\epsilon^2}{8}\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2+O(\epsilon^3)$$
$$= 1-\frac{\epsilon^2}{8}\cdot\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2+O(\epsilon^3).$$

Substituting into (2.10) and using $\ln(1-\delta)\sim-\delta$ as $\delta\to 0$, we obtain

$$R_0=\frac{\epsilon^2}{16\ln 2}\cdot\left(\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)+O(\epsilon^3)$$

for binary-input Class I VNCs.

For Class II VNCs, since $\mathcal{Y}_2\neq\emptyset$,

$$\sum_{y\in\mathcal{Y}}\sqrt{p(y|0)p(y|1)}$$
$$= \sum_{y\in\mathcal{Y}_1}\left(\sqrt{w(y)+\epsilon\sigma(0,y)+O(\epsilon^2)}\cdot\sqrt{w(y)+\epsilon\sigma(1,y)+O(\epsilon^2)}\right)$$
$$+\sum_{y\in\mathcal{Y}_2}\sqrt{\epsilon\cdot\sigma(0,y)+O(\epsilon^2)}\cdot\sqrt{\epsilon\cdot\sigma(1,y)+O(\epsilon^2)}$$
$$= \sum_{y\in\mathcal{Y}_1} w(y)\sqrt{1+\epsilon\cdot\frac{1}{w(y)}\left(\sigma(0,y)+\sigma(1,y)\right)+O(\epsilon^2)}$$
$$+\sum_{y\in\mathcal{Y}_2}\sqrt{\epsilon^2\cdot\sigma(0,y)\sigma(1,y)+O(\epsilon^3)}$$

$$
\begin{aligned}
&= \sum_{y\in y_1} w(y) + \frac{\epsilon}{2}\cdot\left(\sum_{y\in y_1}(\sigma(0,y)+\sigma(1,y)) + \sum_{y\in y_2} 2\sqrt{\sigma(0,y)\sigma(1,y)}\right) + O(\epsilon^2) \\
&= 1 - \frac{\epsilon}{2}\cdot\sum_{y\in y_2}\left(\sigma(0,y)+\sigma(1,y) - 2\sqrt{\sigma(0,y)\sigma(1,y)}\right) + O(\epsilon^2) \\
&= 1 - \frac{\epsilon}{2}\cdot\left(\sum_{y\in y_2}\left(\sqrt{\sigma(0,y)} - \sqrt{\sigma(1,y)}\right)^2\right) + O(\epsilon^2).
\end{aligned}
$$

Therefore,

$$
R_0 = \frac{\epsilon}{4\ln 2}\cdot\left(\sum_{y\in y_2}\left(\sqrt{\sigma(0,y)} - \sqrt{\sigma(1,y)}\right)^2\right) + O(\epsilon^2).
$$

Appendix 2.B Derivations of (2.27), (2.28), (2.29), and (2.30)

To derive (2.27), (2.28), (2.29), and (2.30), we need a longer expansion of $p(y|x)$ as in the last appendix:

$$
p(y|x) = w(y) + \epsilon\cdot\sigma(x,y) + \frac{\epsilon^2}{2}\cdot\eta(x,y) + O(\epsilon^3).
$$

However, the final expressions for the first order approximations do not contain the new term $\eta(x,y)$. We have

$$
\begin{aligned}
\mathrm{E}(U) &= \sum_{y\in y} p(y|0)\ln\frac{p(y|0)}{p(y|1)} \\
&= \sum_{y\in y}\left(\left(w(y) + \epsilon\cdot\sigma(0,y) + \frac{\epsilon^2}{2}\cdot\eta(0,y) + O(\epsilon^3)\right)\right. \\
&\qquad \left.\cdot\ln\frac{w(y) + \epsilon\cdot\sigma(0,y) + \frac{\epsilon^2}{2}\cdot\eta(0,y) + O(\epsilon^3)}{w(y) + \epsilon\cdot\sigma(1,y) + \frac{\epsilon^2}{2}\cdot\eta(1,y) + O(\epsilon^3)}\right).
\end{aligned}
$$

By using $\ln(1+x) \sim x - x^2/2$ as $x \to 0$,

$$
\begin{aligned}
&\ln\left(w(y) + \epsilon\cdot\sigma(0,y) + \frac{\epsilon^2}{2}\cdot\eta(0,y) + O(\epsilon^3)\right) \\
&= \ln w(y) + \ln\left(1 + \epsilon\cdot\frac{\sigma(0,y)}{w(y)} + \frac{\epsilon^2}{2}\cdot\frac{\eta(0,y)}{w(y)} + O(\epsilon^3)\right) \\
&= \ln w(y) + \epsilon\cdot\frac{\sigma(0,y)}{w(y)} + \frac{\epsilon^2}{2}\cdot\left(\frac{\eta(0,y)}{w(y)} - \frac{\sigma^2(0,y)}{w^2(y)}\right) + O(\epsilon^3).
\end{aligned}
$$

Similarly,

$$\begin{aligned}
&\ln\left(w(y)+\epsilon\cdot\sigma(1,y)+\frac{\epsilon^2}{2}\cdot\eta(1,y)+O(\epsilon^3)\right)\\
=\;&\ln w(y)+\epsilon\cdot\frac{\sigma(1,y)}{w(y)}+\frac{\epsilon^2}{2}\cdot\left(\frac{\eta(1,y)}{w(y)}-\frac{\sigma^2(1,y)}{w^2(y)}\right)+O(\epsilon^3).
\end{aligned}$$

Thus

$$\begin{aligned}
&\ln\frac{w(y)+\epsilon\cdot\sigma(0,y)+\frac{\epsilon^2}{2}\cdot\eta(0,y)+O(\epsilon^3)}{w(y)+\epsilon\cdot\sigma(1,y)+\frac{\epsilon^2}{2}\cdot\eta(1,y)+O(\epsilon^3)}\\
=\;&\epsilon\cdot\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)+\frac{\epsilon^2}{2}\cdot\left(\frac{1}{w(y)}\left(\eta(0,y)-\eta(1,y)\right)\right.\\
&\left.+\frac{1}{w^2(y)}\left(\sigma^2(1,y)-\sigma^2(0,y)\right)\right)+O(\epsilon^3). \qquad (2.44)
\end{aligned}$$

Hence

$$\begin{aligned}
\mathrm{E}(U) \;=\;&\epsilon\cdot\sum_{y\in Y}\left(\sigma(0,y)-\sigma(1,y)\right)+\frac{\epsilon^2}{2}\cdot\left(\sum_{y\in Y}\left(\eta(0,y)-\eta(1,y)\right)\right.\\
&\left.+\sum_{y\in Y}\frac{1}{w(y)}\left(\sigma^2(1,y)-\sigma^2(0,y)+2\sigma^2(0,y)-2\sigma(0,y)\sigma(1,y)\right)\right)+O(\epsilon^3)\\
=\;&\frac{\epsilon^2}{2}\cdot\left(\sum_{y\in Y}\frac{1}{w(y)}\left(\sigma^2(0,y)-2\sigma(0,y)\sigma(1,y)+\sigma^2(1,y)\right)\right)+O(\epsilon^3)\\
=\;&\frac{\epsilon^2}{2}\cdot\left(\sum_{y\in Y}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)+O(\epsilon^3). \qquad (2.45)
\end{aligned}$$

Similarly,

$$\begin{aligned}
\mathrm{E}(V) \;&=\;\sum_{y\in Y}p(y|1)\ln\frac{p(y|1)}{p(y|0)}\\
&=\;\frac{\epsilon^2}{2}\cdot\left(\sum_{y\in Y}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)+O(\epsilon^3).
\end{aligned}$$

We proceed to find the first order approximations for $\mathrm{Var}(U)$ and $\mathrm{Var}(V)$. By definition

$$\mathrm{Var}(U)=\mathrm{E}(U^2)-\mathrm{E}^2(U).$$

We have

$$\begin{aligned}
\mathrm{E}(U^2) &= \sum_{y\in\mathcal{Y}} p(y|0)\ln^2\frac{p(y|0)}{p(y|1)} \\
&= \sum_{y\in\mathcal{Y}}\left(\left(w(y)+\epsilon\cdot\sigma(0,y)+\frac{\epsilon^2}{2}\cdot\eta(0,y)+O(\epsilon^3)\right)\right. \\
&\qquad \left.\cdot\ln^2\frac{w(y)+\epsilon\cdot\sigma(0,y)+\frac{\epsilon^2}{2}\cdot\eta(0,y)+O(\epsilon^3)}{w(y)+\epsilon\cdot\sigma(1,y)+\frac{\epsilon^2}{2}\cdot\eta(1,y)+O(\epsilon^3)}\right).
\end{aligned}$$

From (2.44)

$$\begin{aligned}
&\ln^2\frac{w(y)+\epsilon\cdot\sigma(0,y)+\frac{\epsilon^2}{2}\cdot\eta(0,y)+O(\epsilon^3)}{w(y)+\epsilon\cdot\sigma(1,y)+\frac{\epsilon^2}{2}\cdot\eta(1,y)+O(\epsilon^3)} \\
= \;& \epsilon^2\cdot\left(\frac{1}{w^2(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)+O(\epsilon^3).
\end{aligned}$$

So

$$\mathrm{E}(U^2)=\epsilon^2\cdot\left(\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)+O(\epsilon^3).$$

Therefore, from (2.45) we can obtain

$$\begin{aligned}
\mathrm{Var}(U) &= \mathrm{E}(U^2)-\mathrm{E}^2(U) \\
&= \mathrm{E}(U^2)+O(\epsilon^4) \\
&= \epsilon^2\cdot\left(\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)+O(\epsilon^3).
\end{aligned}$$

Similarly,

$$\mathrm{Var}(V)=\epsilon^2\cdot\left(\sum_{y\in\mathcal{Y}}\frac{1}{w(y)}\left(\sigma(0,y)-\sigma(1,y)\right)^2\right)+O(\epsilon^3).$$

Appendix 2.C Proof of (2.41)

We begin with the following definitions:

$$\mathcal{A}^{l,m}=\{\gamma : x_{0\gamma}=l \text{ and } x_{i\gamma}=m, \quad 1\le\gamma\le n\},$$

and

$$\mathcal{B}^{l,m}=\{\gamma : x_{0\gamma}=l \text{ and } x_{j\gamma}=m, \quad 1\le\gamma\le n\},$$

for $l, m \in \mathcal{X}$ and $l \neq m$, where $x_{0\gamma}$'s, $x_{i\gamma}$'s and $x_{j\gamma}$'s are components of $\mathbf{x}_0$, $\mathbf{x}_i$, and $\mathbf{x}_j$, respectively. Since the codewords of generalized triply orthogonal codes are rows of orthogonal arrays of strength 3, for $l \neq m \neq t$,

$$\left|\mathcal{A}^{l,m}\right| = \left|\mathcal{B}^{l,m}\right| = \frac{n}{s^2},$$

and

$$\left|\mathcal{A}^{l,m} \cap \mathcal{B}^{l,m}\right| = \left|\mathcal{A}^{l,m} \cap \mathcal{B}^{l,t}\right| = \frac{n}{s^3}.$$

By the definition of S_i in (2.23), for $i \neq j$,

$$\begin{aligned}
\mathrm{E}(S_i S_j) &= \mathrm{E}\left(\left(\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\alpha\in\mathcal{A}^{l,m}} U_\alpha^{l,m}\right)\left(\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\beta\in\mathcal{B}^{l,m}} U_\beta^{l,m}\right)\right) \\
&= \sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\alpha\in\mathcal{A}^{l,m}}\sum_{\beta\in\mathcal{B}^{l,m}} \mathrm{E}\left(U_\alpha^{l,m}U_\beta^{l,m}\right) + \sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\substack{t\in\mathcal{X}\\ t\neq m\\ t\neq l}}\sum_{\alpha\in\mathcal{A}^{l,m}}\sum_{\beta\in\mathcal{B}^{l,t}} \mathrm{E}\left(U_\alpha^{l,m}U_\beta^{l,t}\right) \\
&\quad + \sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\substack{l'\in\mathcal{X}\\ l'\neq l}}\sum_{\substack{t\in\mathcal{X}\\ t\neq l'}}\sum_{\alpha\in\mathcal{A}^{l,m}}\sum_{\beta\in\mathcal{B}^{l',t}} \mathrm{E}\left(U_\alpha^{l,m}U_\beta^{l',t}\right) \\
&= \sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\left(\sum_{\alpha\in\mathcal{A}^{l,m}\cap\mathcal{B}^{l,m}} \mathrm{E}\left(\left(U_\alpha^{l,m}\right)^2\right) + \sum_{\alpha\in\mathcal{A}^{l,m}}\sum_{\substack{\beta\in\mathcal{B}^{l,m}\\ \beta\neq\alpha}} \mathrm{E}\left(U_\alpha^{l,m}\right)\mathrm{E}\left(U_\beta^{l,m}\right)\right) \\
&\quad + \sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\substack{t\in\mathcal{X}\\ t\neq m\\ t\neq l}}\left(\sum_{\alpha\in\mathcal{A}^{l,m}\cap\mathcal{B}^{l,t}} \mathrm{E}\left(U_\alpha^{l,m}U_\beta^{l,t}\right) + \sum_{\alpha\in\mathcal{A}^{l,m}}\sum_{\substack{\beta\in\mathcal{B}^{l,t}\\ \beta\neq\alpha}} \mathrm{E}\left(U_\alpha^{l,m}\right)\mathrm{E}\left(U_\beta^{l,t}\right)\right) \\
&\quad + \sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\substack{l'\in\mathcal{X}\\ l'\neq l}}\sum_{\substack{t\in\mathcal{X}\\ t\neq l'}}\sum_{\alpha\in\mathcal{A}^{l,m}}\sum_{\beta\in\mathcal{B}^{l',t}} \mathrm{E}\left(U_\alpha^{l,m}\right)\mathrm{E}\left(U_\beta^{l',t}\right) \\
&= \frac{n}{s^3}\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}} \mathrm{E}\left(\left(U^{l,m}\right)^2\right) + \left(\frac{n^2}{s^4} - \frac{n}{s^3}\right)\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}} \mathrm{E}^2\left(U^{l,m}\right) \\
&\quad + \frac{n}{s^3}\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\substack{t\in\mathcal{X}\\ t\neq m\\ t\neq l}} \mathrm{E}\left(U^{l,m}U^{l,t}\right) + \left(\frac{n^2}{s^4} - \frac{n}{s^3}\right)\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\substack{t\in\mathcal{X}\\ t\neq m\\ t\neq l}} \mathrm{E}\left(U^{l,m}\right)\mathrm{E}\left(U^{l,t}\right) \\
&\quad + \frac{n^2}{s^4}\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\substack{l'\in\mathcal{X}\\ l'\neq l}}\sum_{\substack{t\in\mathcal{X}\\ t\neq l'}} \mathrm{E}\left(U^{l,m}\right)\mathrm{E}\left(U^{l',t}\right).
\end{aligned}$$

And, from (2.39),

$$\begin{aligned}
\mathrm{E}(S_i)\mathrm{E}(S_j) &= \mathrm{E}^2(S_i) \\
&= \frac{n^2}{s^4}\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\mathrm{E}^2\left(U^{l,m}\right)+\frac{n^2}{s^4}\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\substack{t\in\mathcal{X}\\ t\neq m\\ t\neq l}}\mathrm{E}\left(U^{l,m}\right)\mathrm{E}\left(U^{l,t}\right) \\
&\quad+\frac{n^2}{s^4}\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\substack{l'\in\mathcal{X}\\ l'\neq l}}\sum_{\substack{t\in\mathcal{X}\\ t\neq l'}}\mathrm{E}\left(U^{l,m}\right)\mathrm{E}\left(U^{l',t}\right).
\end{aligned}$$

Then, for $i \neq j$,

$$\begin{aligned}
\mathrm{Cov}(S_i,S_j) &= \mathrm{E}(S_iS_j)-\mathrm{E}(S_i)\mathrm{E}(S_j) \\
&= \frac{n}{s^3}\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\left(\mathrm{E}\left(\left(U^{l,m}\right)^2\right)-\mathrm{E}^2\left(U^{l,m}\right)\right) \\
&\quad+\frac{n}{s^3}\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\substack{t\in\mathcal{X}\\ t\neq m\\ t\neq l}}\left(\mathrm{E}\left(U^{l,m}U^{l,t}\right)-\mathrm{E}\left(U^{l,m}\right)\mathrm{E}\left(U^{l,t}\right)\right) \\
&= \frac{n}{s^3}\left(\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\mathrm{Var}\left(U^{l,m}\right)+\sum_{l\in\mathcal{X}}\sum_{\substack{m\in\mathcal{X}\\ m\neq l}}\sum_{\substack{t\in\mathcal{X}\\ t\neq m\\ t\neq l}}\mathrm{Cov}\left(U^{l,m},U^{l,t}\right)\right).
\end{aligned}$$

References

[1] A. J. Viterbi, "On coded phase-coherent communications," *IRE Trans. Space Electr. Teleme.*, vol. SET-7, pp. 3–14, Mar. 1961.

[2] K. Abdel-Ghaffar and R. J. McEliece, "The ultimate limits of information density," *Proc. NATO Advanced Study Institute on Performance Limits in Communication Theory and Practice*, Il Ciocco, Italy, July 1986.

[3] E. Majani, "A model for the study of 'very noisy' channels, and applications," Ph.D. dissertation, California Institute of Technology, Pasadena, 1988.

[4] B. Reiffen, "A note on 'very noisy' channels," *Inform. Contr.*, vol. 6, pp. 126–130, June 1963.

[5] R. G. Gallager, "A simple derivation of the coding theorem and some applications," *IEEE Trans. Inform. Theory*, vol. IT-11, pp. 3–18, Jan. 1965.

[6] R. G. Gallager, *Information Theory and Reliable Communication.* New York: Wiely, 1968.

[7] K. Abdel-Ghaffar and R. J. McEliece, "Soft-error correction for increased densities in VLSI memories," *Proc. 11th Annual International Symposium on Computer Architecture*, Ann Arbor, MI, pp. 248–250, June 1984.

[8] R. J. McEliece, *The Theory of Information and Coding.* Reading, MA: Addison-Wesley, 1977.

[9] M. Hall, *Combinatorial Theory.* Waltham, MA: Blaisdell, 1967.

[10] L. D. Baumert, "Codes with special correlation," in *Digital Communications with Space Applications.* S. W. Golomb, ed., Englewood Cliffs, NJ: Prentice-Hall, pp. 47–64, 1964.

[11] R. C. Bose and K. A. Bush, "Orthogonal arrays of strength two and three," *Ann. Math. Stat.*, vol. 23, pp. 508–524, 1952.

[12] E. Seiden, "On the problem of construction of orthogonal arrays," *Ann. Math. Stat.*, vol. 25, pp. 151–156, 1954.

[13] K. A. Bush, "A generalization of a theorem due to MacNeish," *Ann. Math. Stat.*, vol. 23, pp. 293–295, 1952.

CHAPTER 3

PERFORMANCE OF BINARY BLOCK CODES AT LOW SIGNAL-TO-NOISE RATIOS

3.1 Introduction

It is well known that for block codes of a given rate, the larger the minimum distance, the better the code will perform at *high* signal-to-noise ratios. An equally important problem is the behavior of block codes at *low* signal-to-noise ratios. In [1] Posner studies the properties of binary block codes over an AWGN channel at low signal-to-noise ratios. Most of the results in [1] assume hard decision on the channel output, and only results based on soft decision are for orthogonal codes. In this chapter we derive error probabilities of general binary block codes used on an unquantized AWGN channel at low signal-to-noise ratios, assuming maximum-likelihood decoding.

The formulation and derivation in this section are based on [2]. Let $\mathbf{C} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{M-1}\}$ be a binary block code (with components 0 and 1) of length n and rate $R = (\log_2 M)/n$. We shall evaluate the performance of C on an unquantized AWGN channel as a function of the bit signal-to-noise ratio E_b/N_0, which we denote by λ^2. Suppose each codeword is equally likely to be selected for transmission. The codes we are interested in are all "symmetric" in the sense that the error probabilities are independent of which codeword is transmitted (all linear codes have this property, for example). Therefore, we assume that $\mathbf{x}_0$ is transmitted.

If $\hat{\mathbf{x}}_0$ is the counterpart of $\mathbf{x}_0$ with its components 0 changed to -1, then the output of the channel becomes

$$\mathbf{y} = \sqrt{S}\hat{\mathbf{x}}_0 + \mathbf{z},$$

where the quantity $\sqrt{S} = \lambda\sqrt{2R}$ and the vector $\mathbf{z} = (z_1, z_2, \ldots, z_n)$ has all components i.i.d. normal random variables with mean 0 and variance 1. (Here we normalize the noise power instead of the signal power as in some other formulations.) The maximum-likelihood decoder outputs the codeword with the minimum Euclidean distance to the received vector $\mathbf{y}$. This will be the correct decision if and only if the decoded codeword was actually transmitted, or equivalently,

$$|\mathbf{z}|^2 < |\mathbf{y} - \sqrt{S}\hat{\mathbf{x}}_i|^2, \qquad \text{for } i = 1, 2, \ldots, M-1.$$

This inequality can be rewritten as

$$\langle \mathbf{z}, \sqrt{S}(\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_0)\rangle < \frac{1}{2}|\sqrt{S}(\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_0)|^2.$$

Let d_i be the Hamming distance between $\mathbf{x}_i$ and $\mathbf{x}_0$ and $\mathbf{u}_i$ be the vector in the direction of $\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_0$ with magnitude $\sqrt{d_i}$. (Actually $\mathbf{u}_i$ is just $\mathbf{x}_i$ if $\mathbf{x}_0 = \mathbf{0}$.) Then $\sqrt{S}(\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_0) = 2\lambda\sqrt{2R}\,\mathbf{u}_i$. If we define the normal random variables

$$T_i = \langle \mathbf{z}, \mathbf{u}_i\rangle, \qquad \text{for } i = 1, 2, \ldots, M-1, \tag{3.1}$$

then P_C, the probability of correct decoding, is given by

$$P_C = \Pr\{\, T_i < \lambda\sqrt{2R}\, d_i, \quad \text{for } i = 1, 2, \ldots, M-1 \,\}.$$

If the distribution function of $T_1, T_2, \ldots, T_{M-1}$ is denoted by $F(x_1, x_2, \ldots, x_{M-1})$, then P_C can be written as

$$P_C = F(\lambda\sqrt{2R}\, d_1, \lambda\sqrt{2R}\, d_2, \ldots, \lambda\sqrt{2R}\, d_{M-1}). \tag{3.2}$$

Note that $T_1, T_2, \ldots, T_{M-1}$ are $M-1$ normal random variables with mean 0 and covariance matrix $\mathbf{V}$ with components

$$V_{ij} = \langle \mathbf{u}_i, \mathbf{u}_j \rangle.$$

If $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{M-1}$ are independent, then $\mathbf{V}$ is nonsingular and the density function of $T_1, T_2, \ldots, T_{M-1}$ is given by

$$p(x_1, x_2, \ldots, x_{M-1}) = \frac{1}{\sqrt{(2\pi)^{M-1}|\mathbf{V}|}} e^{-\frac{1}{2}\mathbf{x}^T\mathbf{V}^{-1}\mathbf{x}}.$$

We can therefore write P_C as an $M-1$-fold integral:

$$P_C = \int_{-\infty}^{\lambda\sqrt{2R}\,d_1} \int_{-\infty}^{\lambda\sqrt{2R}\,d_2} \cdots \int_{-\infty}^{\lambda\sqrt{2R}\,d_{M-1}} p(\mathbf{x})\, d\mathbf{x}.$$

However, if $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{M-1}$ are not independent, then $\mathbf{V}$ is singular and T_1, T_2, ..., T_{M-1} are "degenerate" in the sense in [3, p. 87], and we cannot convert P_C to an integral. This is true for most practical codes because usually $M \gg n$. For example, the $(24, 12)$ extended Golay code has $M = 4096$ and $n = 24$.

The approach we take is to view P_C in (3.2) as a function of λ and approximate P_C by $P_C(0) + \lambda P_C'(0)$ in the neighborhood of $\lambda = 0$. Since the codes we consider are "symmetric,"

$$P_C(0) = F(0, 0, \ldots, 0) = \frac{1}{M} \tag{3.3}$$

because each codeword is equally likely to be decoded if there is no signal at all. By the chain rule for partial differentiation,

$$P_C'(0) = \sum_{i=1}^{M-1} \sqrt{2R}\, d_i \cdot F_i(0, 0, \ldots, 0),$$

where

$$F_i(x_1, x_2, \ldots, x_{M-1}) = \frac{\partial F}{\partial x_i}(x_1, x_2, \ldots, x_{M-1}).$$

We can further express $F_i(x_1, x_2, \ldots, x_{M-1})$ as

$$F_i(x_1, x_2, \ldots, x_{M-1}) = G_i(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_{M-1}) \cdot f_i(x_i),$$

where G_i is the conditional distribution of $(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_{M-1})$ given that $T_i = x_i$, and $f_i(x_i)$ is the marginal density of T_i. Since each T_i is a normal random variable with mean 0 and variance d_i, $f_i(0) = 1/\sqrt{2\pi d_i}$. Therefore,

$$P_C \sim \frac{1}{M} + \lambda \cdot \sqrt{\frac{R}{\pi}} \sum_{i=1}^{M-1} \sqrt{d_i}\, P_i, \tag{3.4}$$

where P_i is the *conditional* probability that $T_1 < 0, \ldots, T_{i-1} < 0, T_{i+1} < 0, \ldots, T_{M-1} < 0$, given that $T_i = 0$. The block error probability $P_E = 1 - P_C$, and hence

$$P_E \sim 1 - \frac{1}{M} - \lambda \cdot \sqrt{\frac{R}{\pi}} \sum_{i=1}^{M-1} \sqrt{d_i}\, P_i. \tag{3.5}$$

In the next section we find a similar expression for the bit error probability at low signal-to-noise ratios. Some properties of P_i are explored in Section 3.3; we then discuss as examples orthogonal codes, bi-orthogonal codes, the $(24, 12)$ extended Golay code and the $(15, 6)$ expurgated BCH code in Section 3.4. The asymptotic coding gain at low signal-to-noise ratios is studied in Section 3.5. Finally, in Section 3.6 we make some conjectures.

3.2 Bit Error Probability

Maximum-likelihood decoding of binary *linear* block codes on an unquantized AWGN channel is now considered. We define P_b, the bit error probability, to be the ratio of the expected number of information bits in error to the length of information bits. Let $\mathbf{C} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{M-1}\}$ be a binary linear block code of length n and rate $R = k/n$, where $M = 2^k$. Assume $\mathbf{x}_0 = \mathbf{0}$ is transmitted so that

$$\mathbf{y} = \sqrt{S}\hat{\mathbf{x}}_0 + \mathbf{z}$$

is received, where $\hat{\mathbf{x}}_0$, $\mathbf{z}$, and $\sqrt{S}$ were defined in the preceding section. If the decoder chooses to output $\mathbf{x}_i$, then it will make w_i bit errors, where w_i is the

number of 1's in the information sequence corresponding to $\mathbf{x}_i$. The expected number of information bits in error is

$$b = \sum_{i=1}^{M-1} w_i \Pr\{\text{The decoder outputs } \mathbf{x}_i.\}.$$

Hence, the bit error probability $P_b = b/k$.

It remains to find the probability that the output codeword is $\mathbf{x}_i$. The maximum-likelihood decoder will output $\mathbf{x}_i$ if and only if the Euclidean distance between the received vector $\mathbf{y}$ and $\hat{\mathbf{x}}_i$ is the smallest among all the codewords, i.e.,

$$|\mathbf{y} - \sqrt{S}\hat{\mathbf{x}}_i| < |\mathbf{y} - \sqrt{S}\hat{\mathbf{x}}_j|, \qquad \text{for all } \; j \neq i.$$

This inequality is equivalent to

$$\langle \mathbf{z}, \hat{\mathbf{x}}_j - \hat{\mathbf{x}}_i \rangle < \sqrt{S}\langle \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j \rangle, \qquad \text{for all } \; j \neq i. \tag{3.6}$$

If we define d_{ij} to be the Hamming distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ and $\mathbf{u}_{ij}$ to be the vector in the direction of $\mathbf{x}_j - \mathbf{x}_i$ with magnitude $\sqrt{d_{ij}}$, then $\langle \mathbf{z}, \hat{\mathbf{x}}_j - \hat{\mathbf{x}}_i \rangle = 2\langle \mathbf{z}, \mathbf{u}_{ij} \rangle$. Also, $\sqrt{S}\langle \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j \rangle = 2\lambda\sqrt{2R}(d_j - d_i)$, where d_i is the Hamming distance between $\mathbf{x}_i$ and $\mathbf{x}_0$. Therefore, (3.6) is equivalent to

$$\langle \mathbf{z}, \mathbf{u}_{ij} \rangle < \lambda\sqrt{2R}(d_j - d_i), \qquad \text{for all } \; j \neq i.$$

For $i \neq j$, we define the normal random variable $T_{ij} = \langle \mathbf{z}, \mathbf{u}_{ij} \rangle$, which has mean 0 and variance d_{ij}. The bit error probability is then

$$P_b = \frac{1}{k} \sum_{i=1}^{M-1} w_i \Pr\{ T_{ij} < 2\sqrt{2R}(d_j - d_i), \quad \text{for all } j \neq i \}.$$

Using the same approach as in the previous section, we view P_b as a function of λ and approximate P_b by $P_b + \lambda P_b'(0)$ near $\lambda = 0$. By the chain rule for partial differentiation and then proceeding as in the last section, we can obtain

$$P_b(0) = \frac{1}{k} \sum_{i=1}^{M-1} w_i F^i(0, 0, \ldots, 0),$$

where F^i is the distribution function of $T_{i0}, \ldots, T_{i,i-1}, T_{i,i+1}, \ldots, T_{i,M-1}$. Also

$$P_b'(0) = \frac{1}{k}\sqrt{\frac{R}{\pi}} \sum_{i=1}^{M-1} w_i \sum_{j \neq i} \frac{(d_j - d_i)}{\sqrt{d_{ij}}} P_{ij},$$

where P_{ij} $(i \neq j)$ is the conditional probability that $T_{ij'} < 0$, for all $j' \neq i$ and $j' \neq j$, given that $T_{ij} = 0$. We now use the linearity of the code to simplify both the expressions of $P_b(0)$ and $P_b'(0)$. For every pair of codewords $\mathbf{x}_i$, $\mathbf{x}_j$, we can always find another codeword $\mathbf{x}_l$ such that $\mathbf{x}_i \oplus \mathbf{x}_j = \mathbf{x}_l$, where $\oplus$ means the modulo-2 addition. Since the normal distribution is symmetric about the origin, up to a permutation of the parameters, F^i for $i = 1, 2, \ldots, M-1$ are equivalent to F in the last section, and $P_{ij} = P_l$, where $\mathbf{x}_l = \mathbf{x}_i \oplus \mathbf{x}_j$. By (3.3), it follows that

$$P_b(0) = \frac{1}{kM} \sum_{i=1}^{M-1} w_i = \frac{1}{kM} \cdot \frac{Mk}{2} = \frac{1}{2},$$

as expected, because each information bit is right or wrong with equal probability when there is no signal at all. Now using the fact that $d_{ij} = d_l$ if $\mathbf{x}_i \oplus \mathbf{x}_j = \mathbf{x}_l$, we obtain

$$P_b \sim \frac{1}{2} - \lambda \cdot \frac{1}{k}\sqrt{\frac{R}{\pi}} \sum_{i=1}^{M-1} w_i \sum_{\substack{j \neq i \\ \mathbf{x}_i \oplus \mathbf{x}_j = \mathbf{x}_l}} \frac{(d_i - d_j)}{\sqrt{d_l}} P_l. \tag{3.7}$$

Note that the above approximation applies to all binary linear block codes. If we make more assumptions about the code $\mathbf{C}$, we can further simplify (3.7). Now suppose $\mathbf{C}$ is systematic and has the symmetry property such that each bit in the codeword is "permutationally equivalent" to each other bit, e.g., $\mathbf{C}$ is cyclic, or more generally, its automorphism group (see definition in Section 3.3) contains a transitive permutation group. Then the bit error probability can be found, alternatively, by dividing the expected number of codeword bits in error by the block length n. All the derivations remain the same as before except that k and w_i will now be replaced by n and d_i, respectively. Then,

$$P_b \sim \frac{1}{2} - \lambda \cdot \frac{1}{n}\sqrt{\frac{R}{\pi}} \sum_{i=1}^{M-1} d_i \sum_{\substack{j \neq i \\ \mathbf{x}_i \oplus \mathbf{x}_j = \mathbf{x}_l}} \frac{(d_i - d_j)}{\sqrt{d_l}} P_l. \tag{3.8}$$

After a few manipulations we can show that

$$\sum_{i=1}^{M-1} d_i \sum_{\substack{j \neq i \\ \mathbf{x}_i \oplus \mathbf{x}_j = \mathbf{x}_l}} \frac{(d_i - d_j)}{\sqrt{d_l}} P_l = \sum_{l=1}^{M-1} \frac{P_l}{\sqrt{d_l}} \sum_{i=0}^{M-1} d_i (d_i - d(\mathbf{x}_i \oplus \mathbf{x}_l)),$$

where $d(\mathbf{x}_i \oplus \mathbf{x}_l)$ denotes the Hamming distance between $\mathbf{x}_i \oplus \mathbf{x}_l$ and $\mathbf{x}_0$. If we further assume that the code $\mathbf{C}$ contains no repeated columns, i.e., there are no two positions in the block where the corresponding bits are the same for all codewords, then from Appendix 3.B

$$\sum_{i=1}^{M-1} d_i^2 = n(n+1)2^{k-2}, \tag{3.9}$$

and

$$\sum_{i=1}^{M-1} d_i \, d(\mathbf{x}_i \oplus \mathbf{x}_l) = n(n+1)2^{k-2} - d_l 2^{k-1}. \tag{3.10}$$

Equation (3.8) can hence be written as

$$P_b \sim \frac{1}{2} - \lambda \cdot \frac{M}{2n} \sqrt{\frac{R}{\pi}} \sum_{i=1}^{M-1} \sqrt{d_i} \, P_i. \tag{3.11}$$

The unknown quantities in both (3.5) and (3.11) are P_i, $i = 1, 2, \ldots, M-1$.

3.3 Properties of $\mathbf{P_i}$

The probability P_i for $i = 1, 2, \ldots, M-1$ is defined to be the conditional probability that $T_1 < 0, \ldots, T_{i-1} < 0$, $T_{i+1} < 0, \ldots, T_{M-1} < 0$, given that $T_i < 0$. In order to illustrate the calculation of the P_i's, consider the $M = 4$ orthogonal code $\{ \mathbf{x}_0 = 0000, \mathbf{x}_1 = 0101, \mathbf{x}_2 = 0011, \mathbf{x}_3 = 0110 \}$. By (3.1) we have the following random variables:

$$\begin{aligned} T_1 &= z_2 + z_4, \\ T_2 &= z_3 + z_4, \\ T_3 &= z_2 + z_3, \end{aligned}$$

where z_1, z_2, z_3, z_4 are i.i.d. $N(0,1)$ random variables. Therefore,

$$\begin{aligned} P_1 &= \Pr\{T_2 < 0, T_3 < 0 | T_1 = 0\}, \\ P_2 &= \Pr\{T_1 < 0, T_3 < 0 | T_2 = 0\}, \\ P_3 &= \Pr\{T_1 < 0, T_2 < 0 | T_3 = 0\}. \end{aligned}$$

Since z_2, z_3, z_4 are i.i.d., it is easy to see that $P_1 = P_2 = P_3$. It remains to find the probability that $z_3 + z_4 < 0$, $z_2 + z_3 < 0$, given that $z_2 + z_4 = 0$, which is the conditional probability that a random point with a normal distribution in 3-dimensional space falls in a region described by $z_3 + z_4 < 0$, $z_2 + z_3 < 0$ given that it is on the plane $z_2 + z_4 = 0$. We shall show in the next section that $P_1 = P_2 = P_3 = \tan^{-1}\sqrt{2}/\pi$. However, for most practical codes with $M \gg n$, a closed form expression for P_i is not expected to exist.

Definition 3.1 *The set of coordinate permutations that map every codeword in the code* C *into a (possibly different) codeword in* C *is called the automorphism group of* C*, denoted by Aut*(C).

It is not difficult to show that Aut(C) is indeed a group. The permutations in Aut(C) partition the codewords in C into equivalence classes. Codewords $\mathbf{x}_i$ and $\mathbf{x}_j$ are in the same equivalence class if there exists a permutation in Aut(C) that maps $\mathbf{x}_i$ to $\mathbf{x}_j$.

Theorem 3.1 *If* $\mathbf{x}_i$ *and* $\mathbf{x}_j$ *are in the same equivalence class partitioned by permutations in Aut*(C)*, then* $P_i = P_j$.

Proof. If $\mathbf{x}_i$ and $\mathbf{x}_j$ are in the same equivalence class, a permutation ϕ that maps $\mathbf{x}_i$ to $\mathbf{x}_j$ will map all the codewords other than $\mathbf{x}_0$ and $\mathbf{x}_i$ to codewords other than $\mathbf{x}_0$ and $\mathbf{x}_j$. It is impossible that two different codewords are mapped

to the same codeword because ϕ^{-1} is also in Aut(C). Recalling (3.1), we obtain that T_i will be accordingly mapped to T_j and $\{T_l : 1 \le l \le M-1,\ l \ne i\}$ to $\{T_l : 1 \le l \le M-1,\ l \ne j\}$, which implies that $P_i = P_j$. ∎

In the preceding orthogonal code example, the permutation $\phi = (2\ 3\ 4)$ maps $\mathbf{x}_1$ to $\mathbf{x}_3$, $\mathbf{x}_3$ to $\mathbf{x}_2$ and $\mathbf{x}_2$ to $\mathbf{x}_1$, so $P_1 = P_2 = P_3$. For many codes, all the codewords of the same weight are in one equivalence class (but this is not generally true), so their corresponding P_i's are equal. Thus, we can use the notation P_d for all the codewords of weight d (as we shall do in later sections). For this case, (3.5) can be simplified to

$$P_E \sim 1 - \frac{1}{M} - \lambda \cdot \sqrt{\frac{R}{\pi}} \sum_d A_d \sqrt{d}\, P_d, \tag{3.12}$$

where $A(z) = \sum_d A_d z^d$ is the weight enumerator. Similarly, we can simplify (3.11) to

$$P_b \sim \frac{1}{2} - \lambda \cdot \frac{M}{2n} \sqrt{\frac{R}{\pi}} \sum_d A_d \sqrt{d}\, P_d. \tag{3.13}$$

(Recall that the original assumption for (3.11) to hold is that C is linear systematic with no repeated columns and Aut(C) contains a transitive permutation group[1].) Automorphism groups of several block codes are discussed in [4] [5] [6]. There are computer search algorithms [7] [8] for finding the entire automorphism group of a code. Furthermore, the entire automorphism groups of all 2, 3, 4-error correcting binary primitive BCH codes have been determined algebraically in [9].

Definition 3.2 *Let* $\mathbf{u}$ *and* $\mathbf{v}$ *be binary vectors. If* $\mathbf{u}$ *has a* 1 *in every position that* $\mathbf{v}$ *has a* 1, *then we say that* $\mathbf{u}$ *covers* $\mathbf{v}$.

Theorem 3.2 *For a binary linear block code, if the codeword* $\mathbf{x}_i$ *covers another (different) nonzero codeword* $\mathbf{x}_j$, *then* $P_i = 0$.

[1] A permutation group G is transitive if, for any two symbols i and j, there is a permutation $\phi \in G$ such that $i\phi = j$.

Proof. Let $\mathbf{x}_l = \mathbf{x}_i \oplus \mathbf{x}_j$. It follows that $\mathbf{x}_l$ is covered by $\mathbf{x}_i$ because $\mathbf{x}_i$ covers $\mathbf{x}_j$. We can now have the random variable $T_i = T_j + T_l$. It is therefore impossible that both T_j and T_l are less than 0 given that $T_i = 0$. The theorem follows from the definition of P_i. ∎

For most practical codes, $M \gg n$, which means that there are many more random variables T_j in the definition of the P_i's than the code dimension n. Hence, it is desirable to eliminate some redundant random variables T_j to reduce the complexity of computing P_i. One simple result is that $T_l < 0$ can be eliminated from P_i if the codeword $\mathbf{x}_l$ covers another nonzero codeword $\mathbf{x}_j$ with $T_j < 0$. This is proved by letting $\mathbf{x}_m = \mathbf{x}_l \oplus \mathbf{x}_j$, and then $T_l = T_j + T_m$ and $T_j < 0, T_m < 0 (\text{or } = 0)$ guarantee that $T_l < 0$. The following theorem tells us in general how we can eliminate redundant T_j. We prove the theorem in Appendix 3.B by using the Farkas Alternative [10, p. 56].

Theorem 3.3 *Let the set* $\mathcal{A} = \{\mathbf{x} : \mathbf{A}\mathbf{x} < 0 \text{ and } \mathbf{d}^T\mathbf{x} = 0\}$ [2] [3] *be nonempty. The inequality* $\mathbf{b}^T\mathbf{x} < 0$ *holds for all* $\mathbf{x} \in \mathcal{A}$ *if and only if* $\mathbf{b} = \mathbf{b}' + \alpha\mathbf{d}$, *where* $\mathbf{b}' \in \{\mathbf{A}^T\mathbf{y} : \mathbf{y} \geq 0 \text{ and } \mathbf{y} \neq 0\}$ [4] *and* $\alpha \in R$.

To interpret this theorem, we view each $T_i < 0$ as an inequality in $z_1, z_2, \ldots, z_n$. The theorem implies that given $T_j < 0$, $j = 1, \ldots, i-1, i+1, \ldots, M-1$, and $T_i = 0$, the particular $T_l < 0$ is redundant and can be eliminated if and only if $T_l = \sum_{\substack{j=1 \\ j \neq i,l}}^{M-1} a_j T_j + \alpha T_i$, where $a_j \geq 0$ (not all zero) and $\alpha \in R$. Note that setting $\alpha = 0$ reduces to the case stated previously: $T_l < 0$ can be eliminated if the codeword $\mathbf{x}_l$ covers another nonzero codeword $\mathbf{x}_j$ with $T_j < 0$. On the other hand, if we somehow want to create another redundant inequality $T_l < 0$, then T_l

[2] Here $\mathbf{x}$, $\mathbf{y}$, $\mathbf{d}$, $\mathbf{b}$, $\mathbf{b}'$ are column vectors, and $\mathbf{A}$ is a matrix.

[3] We say a vector $\mathbf{x} < 0$ if all of its components < 0.

[4] We say a vector $\mathbf{x} \neq 0$ if there exists one component $\neq 0$.

must be in the form of $\sum_{\substack{j=1 \\ j\neq i}}^{M-1} a_j T_j + \alpha T_i$ with $a_j \geq 0$ (not all zero) and $\alpha \in R$.

Theorem 3.4

$$P_i = \Pr\{\, V_{ii}T_j - V_{ij}T_i < 0, \quad j = 1, \ldots, i-1, i+1, \ldots, M-1 \,\},$$

where $V_{ij} = \langle \mathbf{u}_i, \mathbf{u}_j \rangle$.

Proof. P_i is the conditional probability that $T_1 < 0, \ldots, T_{i-1} < 0, T_{i+1} < 0, \ldots, T_{M-1} < 0$, given that $T_i = 0$. Since $V_{ii} < 0$, given $T_i = 0$, P_i remains unchanged if each $T_j < 0$, $j = 1, \ldots, i-1, i+1, \ldots, M-1$, is replaced by $V_{ii}T_j - V_{ij}T_i < 0$:

$$P_i = \Pr\{\, V_{ii}T_j - V_{ij}T_i < 0, \quad j = 1, \ldots, i-1, i+1, \ldots, M-1 \mid T_i = 0 \,\}.$$

It should be noted that the covariance between $V_{ii}T_j - V_{ij}T_i$ and T_i is zero:

$$\begin{aligned} \mathrm{Cov}(V_{ii}T_j - V_{ij}T_i, T_i) &= V_{ii}\mathrm{E}(T_jT_i) - V_{ij}\mathrm{E}(T_i^2) \\ &= V_{ii}\langle \mathbf{u}_i, \mathbf{u}_j \rangle - V_{ij}\langle \mathbf{u}_i, \mathbf{u}_i \rangle \\ &= V_{ii}V_{ij} - V_{ij}V_{ii} \\ &= 0. \end{aligned}$$

Since uncorrelated normal random variables are independent, the condition $T_i = 0$ can be dropped without affecting P_i, completing the proof. ∎

Note that $V_{ii} = \langle \mathbf{u}_i, \mathbf{u}_i \rangle = d_i$, and for codes with $\mathbf{x}_0 = \mathbf{0}$, V_{ij} is the number of positions where $\mathbf{x}_i$ and $\mathbf{x}_j$ are both 1. As mentioned before, for most practical codes of interest, $M \gg n$; even after the redundant $T_j < 0$ are eliminated according to Theorem 3.3, the number of remaining conditions is still very large compared with the code dimension n. Hence, it is difficult to find P_i analytically, so Monte Carlo simulations are used to find approximate values. Since conditional probabilities are usually more difficult to simulate than unconditional ones, Theorem 3.4 gives us an easy way to simulate P_i. First n i.i.d. normal random variables

z_i, $i = 1, 2, \ldots, n$, with mean 0 and variance 1 are generated; then all necessary (nonredundant) conditions $V_{ii}T_j - V_{ij}T_i < 0$ are tested. If all are satisfied, we record this event as a "success." If any one of the conditions fails, we record this event as a "failure." The procedure is repeated a large number of times; then the relative frequency of "success" will be an approximate value for P_i.

3.4 Examples

We now apply the results in previous sections to orthogonal codes, bi-orthogonal codes, the $(24, 12)$ extended Golay code, and the $(15, 6)$ expurgated BCH code.

3.4.1 Orthogonal Codes

We consider orthogonal codes with $M = 2^k$ codewords, which may be obtained by the Sylvester construction [4, Chap. 2, §3] (or see Section 2.3). (Here all the codewords begin with 0.) It is easy to see that all the nonzero codewords are in the same equivalence class. By (3.12) near $\lambda = 0$, the block error probability can be approximated by

$$P_E \sim 1 - \frac{1}{2^k} - \lambda \cdot (2^k - 1)\sqrt{\frac{k}{2\pi}}\, P_{2^{k-1}}. \tag{3.14}$$

By using $P_b = (2^{k-1}P_E)/(2^k - 1)$ [11, pp. 100] or (3.7), then

$$P_b \sim \frac{1}{2} - \lambda \cdot 2^{k-1}\sqrt{\frac{k}{2\pi}}\, P_{2^{k-1}}. \tag{3.15}$$

We now want to compute the value of $P_{2^{k-1}}$, which is the conditional probability that $T_1 < 0, T_2 < 0, \ldots, T_{2^k-2} < 0$, given that $T_{2^k-1} < 0$. By the structures of orthogonal codes, T_i, $i = 1, 2, \ldots, 2^k - 1$, are normal random variables with mean 0 and covariance

$$V_{ij} = \begin{cases} 2^{k-1}, & \text{if } i = j, \\ 2^{k-2}, & \text{if } i \neq j. \end{cases}$$

With trivial verification, the random variables T_i, $i = 1, 2, \ldots, 2^k - 1$, can be modeled by

$$T_i = \sqrt{2^{k-2}}(X_i + X_0),$$

where $X_0, X_1, \ldots, X_{2^k-1}$ are i.i.d. $N(0,1)$ random variables. The probability $P_{2^{k-1}}$ is hence equivalent to the conditional probability that $X_1 + X_0 < 0, X_2 + X_0 < 0, \ldots, X_{2^k-2} + X_0 < 0$, given that $X_{2^K-1} + X_0 = 0$. Thus

$$\begin{aligned} & P_{2^{k-1}} \\ = & \lim_{\triangle x \to 0} \Pr\{ X_1 + X_0 < 0, \ldots, X_{2^k-2} + X_0 < 0 \mid 0 \le X_{2^k-1} + X_0 \le \triangle x \} \\ = & \lim_{\triangle x \to 0} \frac{\Pr\{ X_1 < -X_0, \ldots, X_{2^k-2} < -X_0, -X_0 \le X_{2^k-1} \le -X_0 + \triangle x \}}{\Pr\{ 0 \le X_{2^k-1} + X_0 \le \triangle x \}} \end{aligned}$$

Since $X_{2^k-1} + X_0$ is $N(0,2)$, $\Pr\{ 0 \le X_{2^k-1} + X_0 \le \triangle x \} = \triangle x \cdot Z(0)/\sqrt{2} = \triangle x/\sqrt{4\pi}$ as $\triangle x \to 0$, where $Z(t)$ is the density function of an $N(0,1)$ random variable. We also have

$$\begin{aligned} & \lim_{\triangle x \to 0} \Pr\{ X_1 < -X_0, \ldots, X_{2^k-2} < -X_0, -X_0 \le X_{2^k-1} \le -X_0 + \triangle x \} \\ = & \lim_{\triangle x \to 0} \int_{-\infty}^{\infty} \triangle x \cdot Z(-t) \left[P(-t) \right]^{2^k-2} Z(t)\, dt \\ = & \lim_{\triangle x \to 0} \frac{\triangle x}{\sqrt{2\pi}} \int_{-\infty}^{\infty} Z(\sqrt{2}\, t) \left[P(t) \right]^{2^k-2} dt, \end{aligned}$$

where $P(x) = \int_{-\infty}^{x} Z(t)\, dt$. Finally we obtain

$$P_{2^{k-1}} = \sqrt{2} \int_{-\infty}^{\infty} Z(\sqrt{2}\, t) \left[P(t) \right]^{2^k-2} dt. \tag{3.16}$$

The same result was obtained in [1] by directly expanding into a power series the expressions of the error probabilities for orthogonal codes from [12]. Our $P_{2^{k-1}}$ is equal to $\sqrt{2}\, A_{2^k-1}$ in [1]. In particular, for $k = 2$, A_3 was shown to be $\tan^{-1}\sqrt{2}/(\pi\sqrt{2})$; it follows that $P_2 = \tan^{-1}\sqrt{2}/\pi$. Since it was shown that $A_\nu \approx (2/\nu^2)\sqrt{\pi \ln \nu}$ for large ν,

$$P_{2^{k-1}} \approx \frac{2}{(2^k - 1)^2} \sqrt{2\pi \ln(2^k - 1)}, \qquad \text{for large} \quad k.$$

k	$P_{2^{k-1}}$	$(2^k-1)\sqrt{k/(2\pi)}\,P_{2^{k-1}}$	$2^{k-1}\sqrt{k/(2\pi)}\,P_{2^{k-1}}$
2	3.0409e-1	5.1469e-1	3.4312e-1
3	9.0117e-2	4.3589e-1	2.4908e-1
4	2.6084e-2	3.1219e-1	1.6650e-1
5	7.3959e-3	2.0453e-1	1.0556e-1
6	2.0606e-3	1.2686e-1	6.4436e-2
7	5.6580e-4	7.5845e-2	3.8221e-2
8	1.5351e-4	4.4170e-2	2.2171e-2
9	4.1242e-5	2.5222e-2	1.2636e-2
10	1.0991e-5	1.4185e-2	7.0995e-3

Table 3.1: $P_{2^{k-1}}$ for orthogonal codes.

(3.16) has been integrated numerically for $k = 2$ to 10, and the results are listed in Table 3.1, as are the quantities $(2^k-1)\sqrt{k/(2\pi)}\,P_{2^{k-1}}$ and $2^{k-1}\sqrt{k/(2\pi)}\,P_{2^{k-1}}$, which are the key elements of (3.14) and (3.15), respectively. Note that, for orthogonal codes at very low signal-to-noise ratios, the bit error probability increases with k, or the number of codewords M.

3.4.2 Bi-Orthogonal Codes

A bi-orthogonal code consists of the codewords of an orthogonal code and their complements. We consider bi-orthogonal codes with $M = 2^k$, $k \geq 2$, codewords. The bi-orthogonal code is the first-order Reed-Muller code if the corresponding orthogonal code is obtained by the Sylvester construction [4, Chap. 11, §3]. All the codewords except the all-zero and all-one codewords have weight 2^{k-2}.

Proposition 3.1 *All the codewords except the all-zero and all-one codewords in a bi-orthogonal code are in the same equivalence class.*

Proof. For a bi-orthogonal code with $M = 2^k$, $k \geq 2$, there are $2^k - 2$ codewords

of weight 2^{k-2}. $2^{k-1}-1$ of them begin with 0 because they are codewords of an orthogonal code, while the remainders begin with 1. Since all the nonzero codewords in an orthogonal code are in one equivalence class, it follows that there are at most two equivalence classes for codewords of weight 2^{k-2} in a bi-orthogonal code, one for each half. However, since the automorphism group of a Reed-Muller code contains the general_affine group that is triply transitive[5] [4, Chap. 13, §9], there exist permutations in the automorphism group of a bi-orthogonal code (which is the first-order Reed-Muller code) that map a nonzero codeword beginning with 0 to a codeword beginning with 1. It hence follows that all the codewords except the all-zero and all-one codewords lie in one equivalence class. ∎

The all-one codeword covers every codeword of weight 2^{k-2}, so from Theorem 3.2 the corresponding P_i is zero. Putting everything together, by (3.12) we now have

$$P_E \sim 1 - \frac{1}{2^k} - \lambda \cdot (2^k - 2)\sqrt{\frac{k}{2\pi}}\, P_{2^{k-2}}.$$

Since a bi-orthogonal code contains no repeated columns, can be encoded as a systematic code, and its automorphism group contains a triply transitive group, by (3.13) for λ near 0,

$$P_b \sim \frac{1}{2} - \lambda \cdot (2^k - 2)\sqrt{\frac{k}{2\pi}}\, P_{2^{k-2}}.$$

Now our goal is to find an analytical expression for $P_{2^{k-2}}$, the conditional probability that $T_2 < 0$, $T_3 < 0$,..., $T_{2^k-1} < 0$, given that $T_1 = 0$. (Here we number the codewords in such a way that $\mathbf{x}_i$, $i = 0, 1, \ldots, 2^{k-1}-1$, are codewords of a corresponding orthogonal code and $\mathbf{x}_{2^{k-1}+i}$, $i = 0, 1, \ldots, 2^{k-1}-1$, are the complements of $\mathbf{x}_i$.) Since the all-one codeword $\mathbf{x}_{2^{k-1}}$ covers every codeword of

[5] A permutation group G is t-fold transitive if, given t distinct symbols $i_1, i_2, \ldots, i_t$, and t distinct symbols $j_1, j_2, \ldots, j_t$, there is a permutation $\phi \in G$ such that $i_1\phi = j_1$, $i_2\phi = j_2$,..., $i_t\phi = j_t$.

weight 2^{k-2}, the condition $T_{2^{k-1}} < 0$ is redundant and can be discarded. From the structure of bi-orthogonal codes, the covariances between T_i and T_j, $i, j = 1, \ldots, 2^{k-1} - 1, 2^{k-1} + 1, \ldots, 2^k - 1$, are given by

$$V_{ij} = \begin{cases} 2^{k-2}, & \text{if } i = j, \\ 0, & \text{if } |i - j| = 2^{k-1}, \\ 2^{k-3}, & \text{otherwise.} \end{cases}$$

We then model the random variables $T_1, \ldots, T_{2^{k-1}-1}, T_{2^{k-1}+1}, \ldots, T_{2^k-1}$, by

$$T_i = \sqrt{2^{k-3}}(X_0 + X_i),$$

and

$$T_{2^{k-1}+i} = \sqrt{2^{k-3}}(X_0 - X_i),$$

where $i = 1, 2, \ldots, 2^{k-1} - 1$ and $X_0, X_1, \ldots, X_{2^k-1}$ are i.i.d. $N(0,1)$ random variables. Thus

$$\begin{aligned} & P_{2^{k-2}} \\ = & \lim_{\Delta x \to 0} \Pr\{ X_0 - X_1 < 0 \text{ and } X_0 + X_i < 0, X_0 - X_i < 0, i = 2, 3, \ldots, 2^{k-1} - 1, \\ & \qquad | \; 0 \le X_0 + X_1 \le \Delta x \} \\ = & \lim_{\Delta x \to 0} \Pr\{ X_0 < 0, -X_0 \le X_1 \le X_0 + \Delta x, \text{ and } X_0 < X_i < -X_0, \\ & \qquad i = 2, 3, \ldots, 2^{k-1} - 1 \} \; / \; \Pr\{ 0 \le X_0 + X_1 \le \Delta x \} \\ = & \lim_{\Delta x \to 0} \frac{\int_{-\infty}^{0} \Delta x \cdot Z(-t) \, [P(-t) - P(t)]^{2^{k-1}-2} \, Z(t) \, dt}{\Delta x / \sqrt{4\pi}} \\ = & \sqrt{2} \int_0^{\infty} Z(\sqrt{2}\, t) \, [P(t) - P(-t)]^{2^{k-1}-2} \, dt. \end{aligned}$$

The same result can be obtained if we expand, into a power series in λ, the expressions for error probabilities in [12]. We have integrated numerically the expressions for $P_{2^{k-2}}$, $k = 3, 4, \ldots, 11$, and listed the results in Table 3.2, along with the quantities $(2^k - 2)\sqrt{k/(2\pi)}\, P_{2^{k-2}}$. Again note that, for bi-orthogonal codes at very low signal-to-noise ratios, the bit error probability increases with the number of codewords.

k	$P_{2^{k-2}}$	$(2^k-2)\sqrt{k/(2\pi)}\,P_{2^{k-2}}$
3	1.0817e-1	4.4848e-1
4	2.8223e-2	3.1526e-1
5	7.6703e-3	2.0527e-1
6	2.0968e-3	1.2704e-1
7	5.7062e-4	7.5889e-2
8	1.5415e-4	4.4180e-2
9	4.1327e-5	2.5225e-2
10	1.1003e-5	1.4186e-2
11	2.9114e-6	7.8817e-3

Table 3.2: $P_{2^{k-2}}$ for bi-orthogonal codes.

3.4.3 The (24,12) Extended Golay Code

The $(24, 12)$ extended Golay code is obtained by adding an overall parity check bit to the perfect triple-error-correcting $(23, 12)$ Golay code. Its weight enumerator is $A(x) = 1 + 759x^8 + 2576x^{12} + 759x^{16} + x^{24}$. Note that the codeword of weight 24 is the all-one codeword, which covers all other nonzero codewords. The automorphism group of the $(24, 12)$ Golay code is the Mathieu group M_{24} [4, Chap. 20, §4, Corollary 5] which is five-fold transitive [4, Chap. 20, §3, Theorem 2].

Proposition 3.2 [4, Chap. 20, §3, Problem (6)] *All the codewords of weight* 8 *are in one equivalence class.*

Proposition 3.3 [4, Chap. 20, §4, Problem (11)] *All the codewords of weight* 12 *are in one equivalence class.*

Proposition 3.4 *All the codewords of weight* 16 *are in one equivalence class.*

Proof. The permutation that maps one codeword to another (possibly different) codeword will do the same to their complements. Since the complement of any codeword of weight 8 is a codeword of weight 16 and vice versa, the proposition follows from Proposition 3.2. ∎

Proposition 3.5 *Every codeword of weight* 16 *covers codewords of weight* 8.

Proof. The following is a generator matrix for the (24, 12) extended Golay code [4, Fig. 2.13]:

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

The codeword obtained by taking the modulo-2 sum of row 1, row 2,..., row 10 is of weight 16:

$$011111111110010111000101.$$

The modulo-2 sum of row 1, row 3, row 6 and row 8 gives a codeword of weight 8:

$$010100101000010111000000,$$

which is covered by the previous codeword of weight 16. Also note that the modulo-2 sum of these two codewords is a codeword of weight 8 covered by the first codeword. Now the proposition follows from Proposition 3.4. ∎

By the above propositions, along with theorems in the previous section, near

$\lambda = 0$ the block error probability can be approximated by

$$P_E \sim \frac{4095}{4096} - \lambda \cdot \sqrt{\frac{1}{2\pi}} \left(759\sqrt{8}\, P_8 + 2576\sqrt{12}\, P_{12} \right),$$

and the bit error probability can be approximated by

$$P_b \sim \frac{1}{2} - \lambda \cdot \frac{256}{3} \sqrt{\frac{1}{2\pi}} \left(759\sqrt{8}\, P_8 + 2576\sqrt{12}\, P_{12} \right).$$

Unlike the last two examples, we do not expect exact analytical expressions for P_8 and P_{12}. The procedure described in the last section is used to simulate P_8 and P_{12}. The results are $P_8 \approx 4.0 \times 10^{-6}$ and $P_{12} \approx 4 \times 10^{-8}$. (Since P_{12} is very small, the reliability of the exact value is doubtful but the magnitude is correct.) Then

$$\begin{aligned} P_E &\sim \frac{4095}{4096} - \lambda \cdot \sqrt{\frac{1}{2\pi}} \left(8.6 \times 10^{-3} + 3.6 \times 10^{-4} \right) \\ &\approx \frac{4095}{4096} - \lambda \cdot \left(3.6 \times 10^{-3} \right), \end{aligned}$$

and

$$\begin{aligned} P_b &\sim \frac{1}{2} - \lambda \cdot \frac{256}{3} \sqrt{\frac{1}{2\pi}} \left(8.6 \times 10^{-3} + 3.6 \times 10^{-4} \right) \\ &\approx \frac{1}{2} - \lambda \cdot (0.30). \end{aligned}$$

Note that the terms above for weight 8 codewords (codewords at the minimum distance) are much larger than the terms for P_{12}.

3.4.4 The (15,6) Expurgated BCH Code

We now consider the $(15, 6)$ expurgated BCH code with generator polynomial $(x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1)(x + 1) = x^9 + x^6 + x^5 + x^4 + x + 1$. Its weight distribution is as follows.

d:	0	6	8	10
A_d:	1	30	15	18

It is known [8] [9] that the complete automorphism group of the $(15, 7)$ primitive BCH code with $d_{\min} = 5$ is the group $\{bx^{2^i} + b'x^{2^{i+2}} : b, b' \in GF(2^4),\ b^{2^2+1} \neq$

b'^{2^2+1} and $i = 0, 1\}$. We also know that the automorphism group of an expurgated code contains that of the corresponding primitive code. With the above facts in mind, by examining the codewords of $(15, 6)$ expurgated BCH code, we find that all the codewords of equal weight are in the same equivalence classes. Therefore, the error probabilities near $\lambda = 0$ are approximately

$$P_E \sim \frac{63}{64} - \lambda \cdot \sqrt{\frac{2}{5\pi}} \left(30\sqrt{6}\, P_6 + 15\sqrt{8}\, P_8 + 18\sqrt{10}\, P_{10}\right),$$

and

$$P_b \sim \frac{1}{2} - \lambda \cdot \frac{32}{15}\sqrt{\frac{2}{5\pi}} \left(30\sqrt{6}\, P_6 + 15\sqrt{8}\, P_8 + 18\sqrt{10}\, P_{10}\right).$$

From a Monte Carlo simulation, $P_6 \approx 4.1 \times 10^{-3}$, $P_8 \approx 8.9 \times 10^{-4}$ and $P_{10} \approx 9.4 \times 10^{-5}$. Thus

$$\begin{aligned} P_E &\sim \frac{63}{64} - \lambda \cdot \sqrt{\frac{2}{5\pi}} \left(0.30 + 3.8 \times 10^{-2} + 5.4 \times 10^{-3}\right) \\ &\approx \frac{63}{64} - \lambda \cdot (0.12), \end{aligned}$$

and

$$P_b \sim \frac{1}{2} - \lambda \cdot (0.26).$$

Note again that the terms above for codewords at the minimum distance are much larger than the remaining terms.

3.5 Asymptotic Coding Gain

The *coding gain* is the ratio of the signal-to-noise ratio without coding to the signal-to-noise ratio required when using an error-correcting code to achieve the same error probability. We define the *asymptotic coding gain* as the limit, as the signal-to-noise ratio approaches zero, of the coding gain. Two theorems based on the criterions of P_E and P_b, respectively, will be given.

We now derive approximations to P_E and P_b at low signal-to-noise ratios when no coding is used. For an unquantized AWGN channel, if no coding is used, the bit error probability is

$$P_b = Q(\sqrt{2}\,\lambda),$$

where $Q(x) = \int_x^\infty e^{-t^2/2}/\sqrt{2\pi}\,dt$. Thus near $\lambda = 0$,

$$P_b \sim \frac{1}{2} - \lambda \cdot \frac{1}{\sqrt{\pi}}. \tag{3.17}$$

If we group k bits as a block, when no coding is used, a block error occurs when there is at least one erroneous bit and so

$$P_E = 1 - (1 - P_b)^k,$$

which gives the following approximation near $\lambda = 0$:

$$P_E \sim 1 - \frac{1}{2^k} - \lambda \cdot \frac{k}{2^{k-1}\sqrt{\pi}}. \tag{3.18}$$

Comparing (3.5), (3.7), (3.11), (3.17), and (3.18), we obtain the following:

Theorem 3.5 *For binary block codes, with the criterion based on block error probability, the asymptotic coding gain at low signal-to-noise ratios is given by*

$$G_E = \frac{2^{2(k-1)}}{nk}\left(\sum_{i=1}^{M-1}\sqrt{d_i}\,P_i\right)^2.$$

Theorem 3.6 *For binary linear block codes, with the criterion based on bit error probability, the asymptotic coding gain at low signal-to-noise ratios is given by*

$$G_b = \frac{1}{nk}\left(\sum_{i=1}^{M-1} w_i \sum_{\substack{j\neq i \\ \mathbf{x}_i\oplus\mathbf{x}_j=\mathbf{x}_l}} \frac{(d_i - d_j)}{\sqrt{d_l}}\,P_l\right)^2.$$

If the code used is systematic with no repeated columns and its automorphism group contains a transitive permutation group, then the asymptotic coding gain can be simplified to

$$G_b = \frac{k}{n}\left(\frac{2^{k-1}}{n}\sum_{i=1}^{M-1}\sqrt{d_i}\,P_i\right)^2,$$

which is equal to $(k/n)^2 G_E$.

We now apply the results in Theorems 3.5 and 3.6 to the codes discussed in the previous section. For orthogonal codes with 2^k codewords, based on P_E-criterion, the asymptotic coding gain is

$$G_E = \frac{2^{2(k-1)}(2^k-1)^2}{k}\left(\int_{-\infty}^{\infty} Z(\sqrt{2}\,t)\,[P(t)]^{2^k-2}\,dt\right)^2,$$

which approaches $\pi \ln 2 \approx 3.38$ dB as $k \to \infty$. For the P_b-criterion, the asymptotic coding gain becomes

$$G_b = k\,2^{2(k-1)}\left(\int_{-\infty}^{\infty} Z(\sqrt{2}\,t)\,[P(t)]^{2^k-2}\,dt\right)^2,$$

which is asymptotic in k to $(\pi \ln 2)k^2/2^{2k}$. The same results were obtained in [1]. We list the asymptotic coding gains based on criterions P_e and P_b for orthogonal codes in Table 3.3. Note that, based on the P_E-criterion, except for $k = 2$, orthogonal codes result in positive coding gain compared with no coding at low signal-to-noise ratios and the gain increases with the number of codewords. However, for the P_b-criterion, there is always a coding loss when using an orthogonal code and the loss increases with k.

For bi-orthogonal codes with 2^k codewords, based on the P_E-criterion,

$$G_E = \frac{2^{2(k-1)}(2^k-2)^2}{k}\left(\int_{0}^{\infty} Z(\sqrt{2}\,t)\,[P(t)-P(-t)]^{2^{k-1}-2}\,dt\right)^2,$$

For the P_b-criterion,

$$G_b = k(2^k-2)^2\left(\int_{0}^{\infty} Z(\sqrt{2}\,t)\,[P(t)-P(-t)]^{2^{k-1}-2}\,dt\right)^2.$$

We tabulate these asymptotic coding gains for bi-orthogonal codes in Table 3.4. It is observed that with the criterion P_E, there is a positive coding gain when using a bi-orthogonal code and the gain increases with the number of codewords.

k	G_E	G_b
2	-0.798 dB	-4.32 dB
3	0.258 dB	-7.10 dB
4	0.880 dB	-10.6 dB
5	1.29 dB	-14.6 dB
6	1.58 dB	-18.8 dB
7	1.79 dB	-23.4 dB
8	1.96 dB	-28.1 dB
9	2.09 dB	-33.0 dB
10	2.19 dB	-38.0 dB

Table 3.3: Asymptotic coding gain at low signal-to-noise ratios for orthogonal codes.

k	G_E	G_b
3	0.505 dB	-1.99 dB
4	0.965 dB	-5.06 dB
5	1.32 dB	-8.78 dB
6	1.59 dB	-12.9 dB
7	1.80 dB	-17.4 dB
8	1.96 dB	-22.1 dB
9	2.09 dB	-27.0 dB
10	2.19 dB	-32.0 dB
11	2.28 dB	-37.1 dB

Table 3.4: Asymptotic coding gain at low signal-to-noise ratios for bi-orthogonal codes.

Again, using the P_b-criterion, there is always a coding loss and the loss increases with k.

For the $(24, 12)$ extended Golay code,

$$\begin{aligned} G_E &= \frac{2^{22}}{24 \cdot 12} \left(759\sqrt{8}\, P_8 + 2576\sqrt{12}\, P_{12}\right)^2 \\ &\approx 1.16 \approx 0.66 \text{ dB}, \end{aligned}$$

which is a gain over no coding. Also

$$G_b = \left(\frac{k}{n}\right)^2 G_E \approx 0.291 \approx -5.3 \text{ dB},$$

which is a loss. For the $(15, 6)$ expurgated BCH code,

$$\begin{aligned} G_E &= \frac{2^{10}}{15 \cdot 6} \left(30\sqrt{6}\, P_6 + 15\sqrt{8}\, P_8 + 18\sqrt{10}\, P_{10}\right)^2 \\ &\approx 1.35 \approx 1.3 \text{ dB}, \end{aligned}$$

and

$$G_b = \left(\frac{k}{n}\right)^2 G_E \approx 0.216 \approx -6.6 \text{ dB}.$$

It was shown in [1] that if hard quantization is used on an AWGN channel, using the bit error probability criterion, any coding scheme results in a loss at low signal-to-noise ratios. Note that for all the codes discussed in the last section, based on the P_b-criterion, there is always a loss with respect to no coding, as we expect. However, one should understand that, at low signal-to-noise ratios, maximum-likelihood decoding is not the scheme that minimizes the bit error probability.

3.6 Discussions

The preceding sections show that the performance of binary block codes at low signal-to-noise ratios depends heavily on codes' geometries through the important

quantities P_i. Since the number of inequalities involved in P_i is much larger than the dimension for most codes of interest, closed form expressions for P_i are not expected to exist. We have tried some general lower and upper bounds, but all the resulting bounds are pretty loose. Further research can be done in finding good lower and upper bounds for P_i by using the codes' algebraic structures.

Now we state a conjecture about the property of P_i, which we cannot prove:

Conjecture 3.1 *For codewords* $\mathbf{x}_i, \mathbf{x}_j \in \mathbf{C}$, *if* $d_i < d_j$, *then* $P_i > P_j$.

Consider the quantity $\sum_{i=1}^{M-1} \sqrt{d_i}\, P_i$, which plays an important part in expressions of both the block error probability and the bit error probability. Based on what we have observed, we boldly make the following conjecture:

Conjecture 3.2 *The sum of terms* $\sqrt{d_i}\, P_i$ *at code's minimum distance is larger than the sum of the remaining terms.*

Appendix 3.A Derivations of (3.9) and (3.10)

We need several equalities before we can show (3.9) and (3.10). If the code $\mathbf{C}$ has no zero columns and no repeated columns, then it is easily shown that

$$\sum_{i=1}^{M-1} x_{ij}^2 = 2^{k-1}, \qquad \text{for } j = 1, 2, \ldots, n, \tag{3.19}$$

$$\sum_{i=1}^{M-1} x_{ij}\overline{x}_{ij} = 0, \qquad \text{for } j = 1, 2, \ldots, n, \tag{3.20}$$

$$\sum_{i=1}^{M-1} x_{ij}x_{il} = 2^{k-2}, \qquad \text{for } j, l = 1, 2, \ldots, n \text{ and } j \neq l, \tag{3.21}$$

$$\sum_{i=1}^{M-1} x_{ij}\overline{x}_{il} = 2^{k-2}, \qquad \text{for } j, l = 1, 2, \ldots, n \text{ and } j \neq l, \tag{3.22}$$

where $x_{ij}, j = 1, 2, \ldots, n$, are the components of $\mathbf{x}_i$ and $\overline{x}_{ij}$ is the complement of x_{ij}. (Note that the symmetric assumption we made about each bit position in the codeword implies that there are no zero columns.)

We have

$$\sum_{i=1}^{M-1} d_i^2 = \sum_{i=1}^{M-1} \left(\sum_{j=1}^{n} x_{ij} \right) \left(\sum_{m=1}^{n} x_{im} \right).$$

If $j = m$, then by (3.19)

$$\sum_{j=1}^{n} \sum_{i=1}^{M-1} x_{ij}^2 = n\, 2^{k-1}.$$

On the other hand, if $j \neq m$, then by (3.21)

$$\sum_{j=1}^{n} \sum_{\substack{m=1 \\ m \neq j}}^{n} \sum_{i=1}^{M-1} x_{ij} x_{im} = n(n-1)2^{k-2}.$$

Therefore,

$$\begin{aligned} \sum_{i=1}^{M-1} d_i^2 &= n\, 2^{k-1} + n(n-1)2^{k-2} \\ &= n(n+1)2^{k-2}. \end{aligned}$$

This ends the derivation of (3.9).

Similarly,

$$\sum_{i=1}^{M-1} d_i\, d(\mathbf{x}_i \oplus \mathbf{x}_l) = \sum_{i=1}^{M-1} \left(\sum_{j=1}^{n} x_{ij} \right) \left(\sum_{m=1}^{n} (x_{im} \oplus x_{lm}) \right).$$

If $j = m$, then by (3.19) and (3.20)

$$\sum_{i=1}^{M-1} x_{ij}(x_{ij} \oplus x_{lj}) = \begin{cases} 2^{k-1}, & \text{if } x_{lj} = 0, \\ 0, & \text{if } x_{lj} = 1. \end{cases}$$

Since there are $(n - d_l)$ of x_{lj}'s such that $x_{lj} = 0$,

$$\sum_{j=1}^{n} \sum_{i=1}^{M-1} x_{ij}(x_{ij} \oplus x_{lj}) = (n - d_l)2^{k-1}.$$

On the other hand, if $j \neq m$, then by (3.21) and (3.22)

$$\sum_{i=1}^{M-1} x_{ij}(x_{im} \oplus x_{lm}) = 2^{k-2}.$$

It follows that

$$\sum_{j=1}^{n} \sum_{\substack{m=1 \\ m \neq j}}^{n} \sum_{i=1}^{M-1} x_{ij}(x_{im} \oplus x_{lm}) = n(n-1)2^{k-2}.$$

Finally,

$$\begin{aligned}\sum_{i=1}^{M-1} d_i\, d(\mathbf{x}_i \oplus \mathbf{x}_l) &= (n-d_l)2^{k-1} + n(n-1)2^{k-2} \\ &= n(n+1)2^{k-2} - d_l 2^{k-1},\end{aligned}$$

which is the result of (3.10).

Appendix 3.B Proof of Theorem 3.3

In this appendix $\mathbf{x}$, $\mathbf{y}$, $\mathbf{b}$, $\mathbf{b}'$, $\mathbf{d}$ are all column vectors and $\mathbf{A}$ is a matrix. We say a vector $\mathbf{x} > 0$ if all its components > 0.

The Farkas Alternative [10, p. 56] *Either the equation*

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \textit{has a solution} \quad \mathbf{x} \geq 0 \tag{3.23}$$

or (exclusively)

$$\mathbf{y}^T\mathbf{A} \geq 0,\ \mathbf{y}^T\mathbf{b} < 0 \quad \textit{has a solution} \quad \mathbf{y}. \tag{3.24}$$

Lemma 3.1 *Either the equation*

$$\mathbf{A}\mathbf{x} + \alpha\mathbf{d} = \mathbf{b} \quad \textit{has a solution} \quad \mathbf{x} \geq 0,\ \alpha \in R \tag{3.25}$$

or (exclusively)

$$\mathbf{y}^T\mathbf{A} \leq 0,\ \ \mathbf{y}^T\mathbf{d} = 0,\ \mathbf{y}^T\mathbf{b} > 0 \quad \textit{has a solution} \quad \mathbf{y}. \tag{3.26}$$

Proof. The assertion (3.25) is not yet of the form (3.23) in the previous lemma. So we use a trick: we set the unconstrained $\alpha = u - v$ and require $u \geq 0$ and $v \geq 0$. Now (3.25) becomes

$$\mathbf{A}\mathbf{x} + (u-v)\mathbf{d} = \mathbf{b} \quad \text{has a solution} \quad \mathbf{x} \geq 0,\ u \geq 0, \text{ and } v \geq 0.$$

Multiplying both sides of the equality by -1, we now have

$$(-\mathbf{A})\mathbf{x} - (u - v)\mathbf{d} = -\mathbf{b} \quad \text{has a solution } \mathbf{x} \geq 0,\ u \geq 0, \text{ and } v \geq 0.$$

In partitioned form, this says

$$[-\mathbf{A}, -\mathbf{d}, \mathbf{d}] \begin{bmatrix} \mathbf{x} \\ u \\ v \end{bmatrix} = -\mathbf{b} \quad \text{has a solution } \mathbf{x} \geq 0,\ u \geq 0, \text{ and } v \geq 0.$$

Now we have a Farkas case (3.23). By (3.24), the alternative is this:

$$\mathbf{y}^T [-\mathbf{A}, -\mathbf{d}, \mathbf{d}] \geq 0,\ -\mathbf{y}^T\mathbf{b} < 0 \quad \text{has a solution } \mathbf{y}.$$

If we unpack the first inequality, we can obtain

$$-\mathbf{y}^T\mathbf{A} \geq 0,\ -\mathbf{y}^T\mathbf{d} \geq 0,\ \mathbf{y}^T\mathbf{d} \geq 0,$$

which is equivalent to

$$\mathbf{y}^T\mathbf{A} \leq 0,\ \mathbf{y}^T\mathbf{d} = 0,$$

completing the proof. ∎

Lemma 3.2 *Suppose the set* $\mathcal{A} = \{\mathbf{x} : \mathbf{A}\mathbf{x} < 0,\ \mathbf{d}^T\mathbf{x} = 0\}$ *is nonempty. If* $\mathbf{A}\mathbf{x} \leq 0$, $\mathbf{d}^T\mathbf{x} = 0$, $\mathbf{b}^T\mathbf{x} > 0$ *has a solution* $\mathbf{x}$, *then* $\mathbf{A}\mathbf{y} < 0$, $\mathbf{d}^T\mathbf{y} = 0$, $\mathbf{b}^T\mathbf{y} > 0$ *has a solution* $\mathbf{y}$.

Proof. Let $\mathbf{x} = \mathbf{x}^0$ be a solution of $\mathbf{A}\mathbf{x} \leq 0$, $\mathbf{d}^T\mathbf{x} = 0$, $\mathbf{b}^T\mathbf{x} > 0$. Choose $\mathbf{y} = \mathbf{x}^0 + \epsilon\,\mathbf{z}$, where $\mathbf{z} \in \mathcal{A}$ and $\epsilon > 0$. We will show that for small enough ϵ, $\mathbf{y}$ is a solution of $\mathbf{A}\mathbf{y} < 0$, $\mathbf{d}^T\mathbf{y} = 0$, $\mathbf{b}^T\mathbf{y} > 0$. First since $\mathbf{A}\mathbf{x}^0 \leq 0$ and $\mathbf{A}\mathbf{z} < 0$,

$$\mathbf{A}\mathbf{y} = \mathbf{A}\mathbf{x}^0 + \epsilon\,\mathbf{A}\mathbf{z} < 0.$$

Second we obtain

$$\mathbf{d}^T\mathbf{y} = \mathbf{d}^T\mathbf{x}^0 + \epsilon\,\mathbf{d}^T\mathbf{z} = 0.$$

Finally we can have

$$\mathbf{b}^T\mathbf{y} = \mathbf{b}^T\mathbf{x}^0 + \epsilon\,\mathbf{b}^T\mathbf{z}.$$

Since $\mathbf{b}^T\mathbf{x}^0 > 0$, the condition $\mathbf{b}^T\mathbf{y} > 0$ holds for sufficiently small ϵ. ∎

Theorem 3.3 *Let the set* $\mathcal{A} = \{\mathbf{x} : \ \mathbf{A}\mathbf{x} < 0, \ \mathbf{d}^T\mathbf{x} = 0\}$ *be nonempty. The inequality* $\mathbf{b}^T\mathbf{x} < 0$ *holds for all* $\mathbf{x} \in \mathcal{A}$ *if and only if* $\mathbf{b} = \mathbf{b}' + \alpha\mathbf{d}$, *where* $\mathbf{b}' \in \{\mathbf{A}^T\mathbf{y} : \ \mathbf{y} \geq 0$ *and* $\mathbf{y} \neq 0\}$ *and* $\alpha \in R$.

Proof. The proof for the sufficient condition is straightforward. We have

$$\mathbf{b}^T\mathbf{x} = \mathbf{b}'^T\mathbf{x} + \alpha\mathbf{d}^T\mathbf{x} = \mathbf{b}'^T\mathbf{x} = \mathbf{y}^T\mathbf{A}\mathbf{x},$$

which is < 0 because $\mathbf{y} \geq 0$, $\mathbf{y} \neq 0$ and $\mathbf{A}\mathbf{x} < 0$. Now comes the proof for the opposite direction. Suppose the necessary condition is wrong. First we assume that $\mathbf{b} = \mathbf{b}' + \alpha\mathbf{d}$ but with $\mathbf{b}' = 0$. Then $\mathbf{b}^T\mathbf{x} = \alpha\mathbf{d}^T\mathbf{x} = 0$, contradicting that $\mathbf{b}^T\mathbf{x} < 0$ for all $\mathbf{x} \in \mathcal{A}$. Second we assume $\mathbf{b}$ is not in the form of $\mathbf{b} = \mathbf{b}' + \alpha\mathbf{d}$, where $\mathbf{b}' \in \{\mathbf{A}^T\mathbf{y} : \ \ \mathbf{y} \geq 0\}$ and $\alpha \in R$. It follows that the equation $\mathbf{A}^T\mathbf{y} + \alpha\mathbf{d} = \mathbf{b}$ doesn't have a solution $\mathbf{y} \geq 0$, $\alpha \in R$. Therefore, the case (3.25) of Lemma 3.1 is wrong, and we must have the alternative:

$$\mathbf{x}^T\mathbf{A}^T \leq 0, \ \mathbf{x}^T\mathbf{d} = 0, \ \mathbf{x}^T\mathbf{b} > 0 \quad \text{has a solution} \ \ \mathbf{x},$$

which is the same as

$$\mathbf{A}\mathbf{x} \leq 0, \ \mathbf{d}^T\mathbf{x} = 0, \ \mathbf{b}^T\mathbf{x} > 0 \quad \text{has a solution} \ \ \mathbf{x}.$$

By Lemma 3.2 this implies

$$\mathbf{A}\mathbf{x} < 0, \ \mathbf{d}^T\mathbf{x} = 0, \ \mathbf{b}^T\mathbf{x} > 0 \quad \text{has a solution} \ \ \mathbf{x},$$

which contradicts the assumption that $\mathbf{b}^T\mathbf{x} < 0$ holds for all $\mathbf{x} \in \mathcal{A}$. ∎

References

[1] E. C. Posner, "Properties of error-correcting codes at low signal-to-noise ratios," *SIAM J. Appl. Math.*, vol. 15, pp. 775–798, July 1967.

[2] L. Swanson, R. J. McEliece and C.-C. Chao, "Behavior of codes at very low signal-to-noise ratios," *1988 IEEE International Symposium on Information Theory*, Kobe, Japan, June 1988.

[3] W. Feller, *An Introduction to Probability Theory and its Applications*, vol. 2, 2nd ed. New York: Wiley, 1971.

[4] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes.* Amsterdam: North Holland, 1977.

[5] J. H. van Lint, *Introduction to Coding Theory.* New York: Springer-Verlag, 1982.

[6] W. W. Peterson and E. J. Weldon, Jr., *Error Correcting Codes*, 2nd ed. Cambridge, MA: MIT Press, 1972.

[7] J. S. Leon, "Computing automorphism groups of error-correcting codes," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 496–511, May 1982.

[8] J. J. Costa, "An algorithm for finding the automorphism group of a linear code," Ph.D. dissertation, University of Southern California, Los Angeles, 1985.

[9] C.-C. Lu, "The automorphism groups of binary primitive BCH codes," Ph.D. dissertation, University of Southern California, Los Angeles, 1987.

[10] J. Franklin, *Methods of Mathematical Economics.* New York: Springer-Verlag, 1980.

[11] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding.* New York: McGraw-Hill, 1979.

[12] A. J. Viterbi, "On coded phase-coherent communications," *IRE Trans. Space Electr. Teleme.*, vol. SET-7, pp. 3–14, Mar. 1961.

CHAPTER 4

ON THE PATH WEIGHT ENUMERATORS OF CONVOLUTIONAL CODES

4.1 Introduction

In the study of convolutional codes, we are particularly interested in distance properties because they are of great importance in performance estimation. Usually we consider the convolutional code encoder as a finite state machine; then the behavior of the encoder can be completely described by the corresponding state diagram. There is a one-to-one correspondence between the possible output code sequences from the encoder and the paths through the state diagram beginning and ending in the all-zero state. We call a path beginning and ending in the all-zero state without intermediate returns a *fundamental path*, and for each d denote by A_d the number of fundamental paths of weight d. The *path weight enumerator* [1] $A(x)$, which is the generating function of A_d:

$$A(x) = \sum_{d=0}^{\infty} A_d x^d$$

provides weight distribution information of the corresponding convolutional code. The *free distance* of the code is defined to be the least d such that A_d is not zero:

$$d_{\text{free}} = \min\{\, d \geq 0 : \; A_d > 0 \,\}.$$

It is well-known that for a given code rate, the larger the free distance, the better the code will perform at *high* signal-to-noise ratios. In this chapter, we introduce

a quantity, called the *dominant root* of the code, which tells us the growth rate of A_d at large distance:

$$\alpha = \limsup_{d \to \infty} A_d^{1/d}. \tag{4.1}$$

The importance of α can be seen from the following transfer function bound for the first-event error probability (which is the probability that the correct path is excluded for the first time during Viterbi decoding):

$$P_E \leq A(x)|_{x=\gamma}, \tag{4.2}$$

where γ is the channel's Bhattacharyya parameter [1]. For a binary-input DMC with output alphabet $\mathcal{Y}$,

$$\gamma = \sum_{y \in \mathcal{Y}} \sqrt{p(y|0)p(y|1)}.$$

For the unquantized AWGN channel,

$$\gamma = e^{-E_s/N_0},$$

where E_s/N_0 is the channel symbol signal-to-noise ratio. The largest positive value of γ for which the bound 4.2 converges is the radius of convergence of $A(x)$, which by a well-known theorem [3, p. 213] is α^{-1}, where α is defined in (4.1).

In principle, the path weight enumerator $A(x)$ can be computed by applying Mason's gain rule, or some other standard combinatorial technique, to the code's labelled state diagram [1] [4]. In general, $A(x)$ is a rational function, with integer coefficients:

$$A(x) = \frac{N(x)}{D(x)}.$$

Since the coefficients A_d of $A(x)$ are nonnegative, it follows from [3, Theorem 7.21] that $x = \alpha^{-1}$ is a singularity of $A(x)$, i.e., that $D(\alpha^{-1}) = 0$. Thus α is the reciprocal of the least-magnitude root of the equation $D(x) = 0$.

In this chapter we shall investigate the dominant roots of various convolutional codes. In Table 4.1 we actually compute α for some $(2,1)$ convolutional codes of constraint length [1] K from 4 to 12. For $K \leq 7$, α is computed directly from the denominator of $A(z)$. For $K > 7$, we compute α as follows. From [5], it is known that the maximum number of consecutive all-zero branches that can occur in a nonzero fundamental path is $K-2$ for a rate $1/2$ code of constraint length K. Hence any fundamental path of weight d has length $\leq (K-1)d$ branches. We then compute A_d from the trellis diagram up to some large enough d for a given code and found α approximately by either A_d/A_{d-1} or $\sqrt{A_d/A_{d-2}}$ for some large d. (For some codes $A_d = 0$ for odd d, so we compute $\sqrt{A_d/A_{d-2}}$ for large even d instead of A_d/A_{d-1}.)

One should note that for a fixed constraint length, all dominant roots are close together, and, furthermore, they seem to approach a limit for very large K. We explain this interesting phenomenon in Section 4.2 by considering the ensemble of fixed convolutional codes as a subset of the ensemble of time-varying convolutional codes, and then computing the average distance profile for a random time-varying code. The result gives a surprisingly accurate prediction of the growth rate of the number of fundamental paths at large distance for fixed codes. In Section 4.3, the corresponding generating functions are found, and their pole locations are investigated. In Section 4.4, we estimate the average free distance for time-varying convolutional codes and obtain, for each finite constraint length, a Gilbert-like free distance lower bound that performs asymptotically as well as the asymptotic bound in [6]. A similar random coding analysis for the total weight of information bits for fundamental paths at each distance is given in Section 4.5. An interesting example of the performance of several convolutional codes at low signal-to-noise

[1] Following [1], we define the constraint length K to be $m+1$, where m is the code's memory.

	Generator polynomials			
K	g_1	g_2	d_{free}	α
4	1101	1111	6	2.20557
4	1011	1101	6	2.21750
4	1001	1011	5	2.20557
4	1110	1101	6	2.08931
4	1010	1101	5	2.07766
4	1111	0111	4	2.06709
4	1001	1000	3	1.93013
5	10011	11101	7	2.31266
5	10101	11101	6	2.29655
5	10111	11111	6	2.30147
5	11101	11001	7	2.29503
5	10001	11111	6	2.34632
5	11111	01111	4	2.25692
5	10101	10000	4	2.09504
6	110101	101111	8	2.35695
6	110001	101001	6	2.35830
6	100101	111111	8	2.36326
6	110011	111011	7	2.35927
6	100011	100111	7	2.34530
6	101111	011001	8	2.35089
6	110101	100000	5	2.18396
7	1011011	1111001	10	2.38762
7	1100101	1000101	7	2.37357
7	1001001	1110011	8	2.38751
7	1100111	1111111	6	2.38699
7	1001111	1100011	8	2.38929
7	1110011	0101001	8	2.38219
7	1111001	1000000	6	2.24920
8	11100101	10011111	10	2.40007
8	11011011	10011011	7	2.39711
8	10011101	11010011	8	2.39768
8	10001001	10101101	8	2.39200

Table 4.1: Dominant roots for some $(2, 1)$ convolutional codes.

K	g_1	g_2	d_{free}	α
8	10000101	10111001	8	2.40034
8	10111001	01110001	9	2.38478
8	11101111	10000000	6	2.32136
9	101110001	111101011	12	2.40733
9	100100101	100100001	7	2.41385
9	110000001	110101101	9	2.40651
9	101011001	110100111	11	2.40672
9	111100011	101001101	11	2.40582
9	101111011	011001111	10	2.40383
9	111010111	100000000	8	2.34476
10	1001110111	1101100101	12	2.41046
10	1011101001	1001010011	9	2.40925
10	1010011001	1001111011	12	2.41026
10	1101101001	1011110101	12	2.41019
10	1111100011	1000011011	10	2.40888
10	1100100011	0111010001	10	2.40999
10	1110111001	1000000000	8	2.35743
11	10011011101	11110110001	14	2.41212
11	11100110111	10001101111	11	2.41125
11	11010000001	10010011111	11	2.41247
11	11000010011	11001001101	11	2.41172
11	11001010101	10000101011	11	2.41251
11	11110101001	01101100001	12	2.40988
11	11110110001	10000000000	8	2.36820
12	100011011101	101111010011	15	2.41303
12	111110110111	111101011011	11	2.41191
12	101001001101	111000100111	12	2.41321
12	110110111001	100001100101	13	2.41318
12	101011011001	100100001001	10	2.41355
12	101100000101	010100010111	11	2.41259
12	110001100001	100000000000	6	2.40774

Table 4.1: (*Continued.*)

ratios is given in Section 4.6. Finally, possible extensions to this work are discussed in Section 4.7.

4.2 Average Distance Structure for Random Convolutional Codes

In this section we shall compute the average distance profile for the ensemble of time-varying convolutional codes. A time-varying convolutional code [1] [7] is a convolutional code whose generator polynomials may be changed after each time unit. In other words, the tap positions of modulo-2 adders in the shift register encoder are reselected after each shift of the bits. Now consider the ensemble of all time-varying convolutional codes, which include the ensemble of fixed convolutional codes as a subset. A uniform probability measure is imposed on each code by randomly reselecting the encoder tap positions after each shift. This can be done by hypothetically flipping a fair coin for each tap position. Then the encoder output will be a random binary vector after each shift.

Let $\overline{A}_d$ denote the average number of fundamental paths of weight d in the ensemble of all (n,k) time-varying convolutional codes of constraint length K. We consider only the (n,k) codes with $2^{k(K-1)}$ states, whose encoders have k shift registers all of the same length K. One of the main results in this chapter is the following theorem:

Theorem 4.1 *For $K > 2$,*

$$\overline{A}_d = \theta_d + \frac{1}{n} \sum_{i=1}^{K-1} \sum_{\delta:\delta^n=r_i} \frac{c_i}{1-\frac{\delta}{2}} \left(\frac{\delta}{2-\delta} \right)^d,$$

where

$$\theta_d = \begin{cases} 1 - \frac{1}{2^n}, & \text{if } d = 0, \\ -\frac{\binom{n}{d}}{2^n}, & \text{if } 1 \le d \le n, \\ 0, & \text{if } d > n, \end{cases}$$

and r_i, $i = 1, 2, \ldots, K-1$, *are the reciprocals of the roots of*

$$1 - (2^k - 1) \sum_{j=1}^{K-1} y^j = 0.$$

Proof. Let $A_{d,l}$ be the number of fundamental paths of weight d and length l so that

$$\overline{A}_d = \sum_l \overline{A}_{d,l}. \tag{4.3}$$

Under the uniform probability measure, each branch of a trellis path is a random n-dimensional binary vector, and each nl-dimensional binary vector has the same probability of being a fundamental path of length l. Let T_l denote the number of fundamental paths of length l for any (n, k) convolutional code of constraint length K. Then the probability of being a fundamental path for any nl-dimensional binary vector is $T_l/2^{nl}$. Since there are $\binom{nl}{d}$ nl-dimensional binary vectors of weight d,

$$\overline{A}_{d,l} = \binom{nl}{d} \frac{T_l}{2^{nl}}. \tag{4.4}$$

We show in Appendix 4.A that the generating function $T(y)$ of T_l for any (n, k) code of constraint length K being considered is given by

$$T(y) = \frac{\hat{P}(y)}{\hat{Q}(y)} = \frac{(2^k - 1)y^K(1 - y)}{1 - 2^k y + (2^k - 1)y^K}. \tag{4.5}$$

Cancelling the common factor $(1 - y)$ in $\hat{P}(y)$ and $\hat{Q}(y)$, we obtain

$$T(y) = \frac{P(y)}{Q(y)} = \frac{(2^k - 1)y^K}{1 - (2^k - 1)\sum_{j=1}^{K-1} y^j}. \tag{4.6}$$

The denominator $Q(y)$ can be shown to be squarefree for $K > 2$ by proving that $\hat{Q}(y) = (1 - y)Q(y) = 1 - 2^k y + (2^k - 1)y^K$ is squarefree:

$$\gcd\left(\hat{Q}(y), \hat{Q}'(y)\right) = 1, \qquad \text{for } K > 2, \tag{4.7}$$

The derivation of (4.7) can be found in Appendix 4.B. For $K > 2$, $T(y)$ can therefore be partial-fraction expanded to

$$T(y) = 1 - y + \sum_{i=1}^{K-1} \frac{c_i}{1 - r_i y},$$

where $r_i, i = 1, 2, \ldots, K-1$, are the reciprocals of the roots of $1-(2^k-1)\sum_{j=1}^{K-1} y^j = 0$ and

$$c_i = \frac{-r_i P(r_i^{-1})}{Q'(r_i^{-1})}.$$

Therefore,

$$T_l = \phi_l + \sum_{i=1}^{K-1} c_i r_i^l, \tag{4.8}$$

where

$$\phi_l = \begin{cases} 1, & \text{if } l = 0, \\ -1, & \text{if } l = 1, \\ 0, & \text{if } l \geq 2. \end{cases}$$

For $d > n$, from (4.3), (4.4) and (4.8)

$$\begin{aligned} \overline{A}_d &= \sum_{i=1}^{K-1} \sum_{l=\lceil \frac{d}{n} \rceil}^{\infty} \binom{nl}{d} \frac{c_i r_i^l}{2^{nl}} \\ &= \sum_{i=1}^{K-1} c_i \sum_{l=\lceil \frac{d}{n} \rceil}^{\infty} \frac{nl(nl-1)\cdots(nl-d+1)}{d!} \left(\frac{r_i^{\frac{1}{n}}}{2} \right)^{nl}, \end{aligned}$$

where $r_i^{1/n}$ is any n-th root of r_i. In order to compute the above series, define $f(t) = 1 + t^n + t^{2n} + \cdots + t^{ln} + \cdots$. Then the d-th derivative of $f(t)$ is

$$f^{[d]}(t) = t^{-d} \sum_{l=\lceil \frac{d}{n} \rceil}^{\infty} nl(nl-1)\cdots(nl-d+1)t^{nl}.$$

It thus follows that

$$\overline{A}_d = \sum_{i=1}^{K-1} c_i \cdot \frac{t^d}{d!} f^{[d]}(t) \Big|_{t=\frac{r_i^{\frac{1}{n}}}{2}}, \qquad \text{for } d > n. \tag{4.9}$$

It is not difficult to see that

$$f(t) = \frac{1}{1-t^n} = \frac{1}{n} \sum_{s=0}^{n-1} \frac{1}{1-\omega^s t},$$

where $\omega = e^{i\frac{2\pi}{n}}$. With $f(t)$ in this form, its d-th derivative is easily found to be

$$f^{[d]}(t) = \frac{1}{n}\sum_{s=0}^{n-1} d! \cdot \omega^{sd} \cdot (1-\omega^s t)^{-(d+1)}. \tag{4.10}$$

Combining (4.9) and (4.10), we get

$$\overline{A}_d = \frac{1}{n}\sum_{i=1}^{K-1}\sum_{s=0}^{n-1} \frac{c_i}{1-\omega^s \cdot \frac{r_i^{\frac{1}{n}}}{2}} \left(\frac{\omega^s r_i^{\frac{1}{n}}}{2-\omega^s r_i^{\frac{1}{n}}}\right)^d, \qquad \text{for } d > n.$$

Since $\omega^s r_i^{1/n}$, $s = 0, 1, \ldots, n-1$, are all the n-th roots of r_i, $\overline{A}_d$ can finally be expressed as the form in Theorem 4.1 for $d > n$. The proofs for $d = 0$ and $1 \le d \le n$ are the same as the above except that a few modifications for the offset θ_d are needed. ∎

Now we want to investigate the behavior of $\overline{A}_d$ when d is large. Let δ be any n-th root of any r_i and let r be the reciprocal of the least-magnitude root of $1-(2^k-1)\sum_{j=1}^{K-1} y^j$, the denominator of $T(y)$. From [3, Theorem 7.21], r is real and positive because $T(y)$ is a nonnegative series. Since

$$\left|\frac{\delta}{2-\delta}\right| \le \frac{|\delta|}{||2|-|\delta||} \le \frac{\sqrt[n]{r}}{2-\sqrt[n]{r}},$$

it follows from Theorem 4.1 that for large d, $\overline{A}_d$ satisfies

$$\overline{A}_d \approx \beta\hat{\alpha}^d,$$

where β is some constant independent of d and

$$\hat{\alpha} = \frac{\sqrt[n]{r}}{2-\sqrt[n]{r}}.$$

For large K, r is very close to 2^k, and then $\hat{\alpha}$ is very close to $2^R/(2-2^R)$, where $R = k/n$ is the code rate.

We compute $\hat{\alpha}$ for $(2,1)$ codes of constraint lengths from 3 to 12 in Table 4.2. Comparing Tables 4.1 and 4.2, we can see that for a given K, the α's in Table

K	$\hat{\alpha}$
3	1.7473273
4	2.1065695
5	2.2699329
6	2.3451684
7	2.3806466
8	2.3977176
9	2.4060505
10	2.4101566
11	2.4121921
12	2.4132048

Table 4.2: Average $\hat{\alpha}$ for $(2,1)$ convolutional codes.

4.1 are quite close to the $\hat{\alpha}$'s in Table 4.2, especially for $K \geq 7$. Some kind of law of large numbers appears to be in operation. If we choose a convolutional code randomly, then its α is expected to be close to the average $\hat{\alpha}$. Note that for $(2,1)$ codes, $\hat{\alpha}$ will approach $\sqrt{2}/(2-\sqrt{2}) = 1+\sqrt{2}$, which is 2.4142136..., for large constraint length K. All the above results can facilitate predicting A_d at large distance for codes of moderate to large constraint length. Also note that the α's for the codes usually used in practice, of which the encoders have both ends of shift registers tapped to the modulo-2 adders, are closer to the average $\hat{\alpha}$ than those for systematic codes (the last row of each constraint length in Table 4.1).

4.3 The Generating Function $\overline{A}(x)$

In this section we compute the generating function $\overline{A}(x)$ of $\overline{A}_d$ and then investigate the pole locations. From (4.3) and (4.4),

$$\overline{A}(x) = \sum_{d=0}^{\infty} \overline{A}_d x^d = \sum_{d=0}^{\infty} \sum_{l=\lceil \frac{d}{n} \rceil}^{\infty} \binom{nl}{d} \frac{T_l}{2^{nl}} x^d.$$

Changing the order of summations, we obtain

$$\overline{A}(x) = \sum_{l=0}^{\infty} \frac{T_l}{2^{nl}} \sum_{d=0}^{nl} \binom{nl}{d} x^d = \sum_{l=0}^{\infty} \frac{T_l}{2^{nl}} (1+x)^{nl}$$

by recognizing $\sum_{d=0}^{nl} \binom{nl}{d} x^d = (1+x)^{nl}$. Therefore $\overline{A}(x)$ can be found to be

$$\overline{A}(x) = T\left(\left(\frac{1+x}{2}\right)^n\right) \tag{4.11}$$

$$= \frac{(2^k-1)(1+x)^{nK}}{2^{nK} - (2^k-1)\sum_{j=1}^{K-1} 2^{n(K-j)}(1+x)^{nj}}. \tag{4.12}$$

All the poles of $\overline{A}(x)$ for $n = 2$, $k = 1$, and $K = 7$ are plotted in Figure 4.1. Observe that there is only one pole inside the unit circle, a fact that is proven for $n \leq 4$ and $K > 2$ by using the well-known Rouché's Theorem:

Rouché's Theorem [3, Theorem 3.42] *If $f(z)$ and $g(z)$ are analytic inside and on a closed contour C, and $|g(z)| < |f(z)|$ on C, then $f(z)$ and $f(z)+g(z)$ have the same number of zeros inside C.*

Lemma 4.1 *The polynomial $\hat{Q}(y) = 1 - 2^k y + (2^k - 1)y^K$ has only one zero inside the unit circle for $K > 2$.*

Proof. Choose $r < 1$ but sufficiently close to 1 such that $1 - 2^k r + (2^k - 1)r^K < 0$. This is possible because $\hat{Q}(y)|_{y=1} = 0$ and

$$\left.\frac{d\hat{Q}(y)}{dy}\right|_{y=1} = -2^k + K(2^k - 1) > 0, \qquad \text{for } K > 2.$$

Let C be the contour $|y| = r$, $f(y) = -2^k y$, and $g(y) = 1 + (2^k - 1)y^K$. On C

$$|g(y)| = |1 + (2^k - 1)y^K| < 1 + (2^k - 1)r^K < 2^k r = |f(y)|.$$

Since $f(y)$ has only one zero inside the contour C, then by Rouché's Theorem, $\hat{Q}(y)$ has only one zero inside C. The lemma follows from the fact that any zero inside the unit circle will be inside the contour C for r sufficiently close to 1. ∎

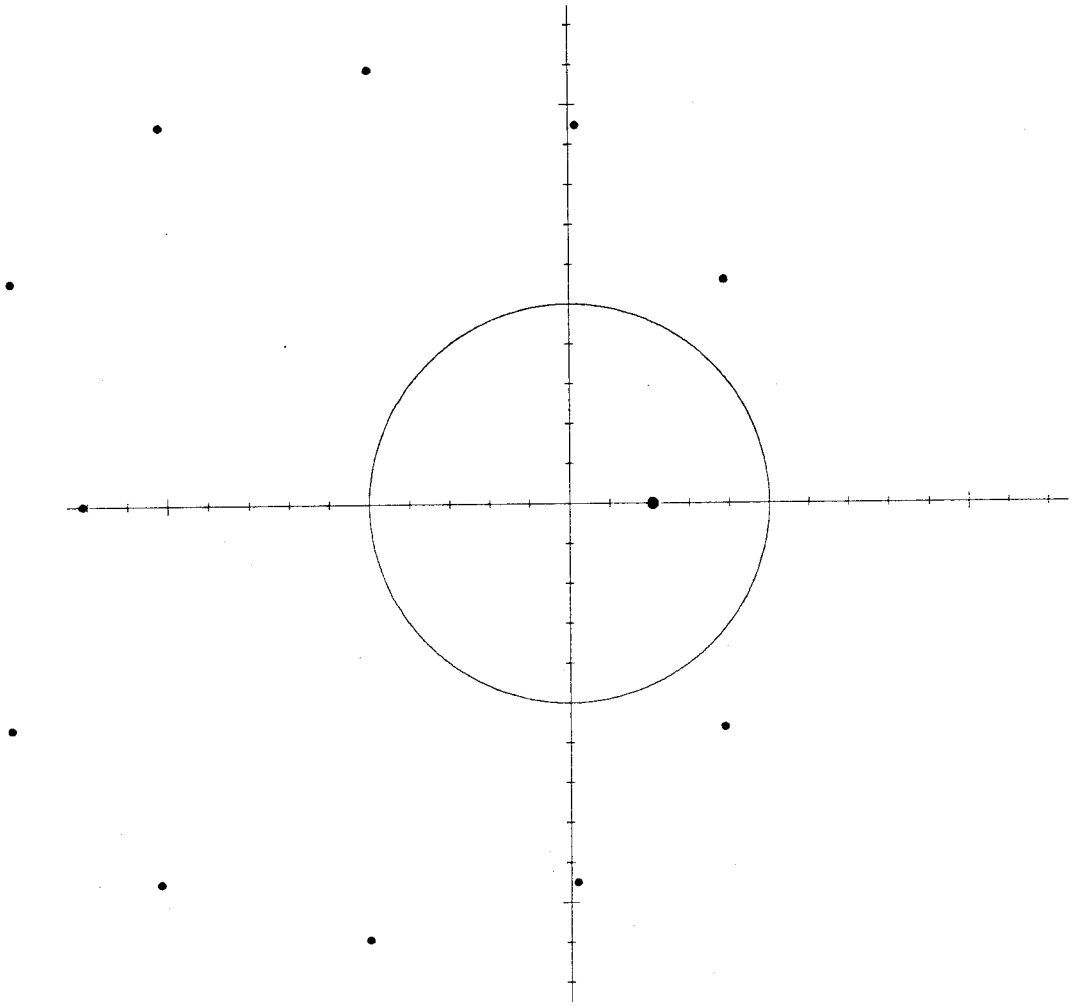

Figure 4.1: Pole locations of $\overline{A}(x)$ for $n = 2$, $k = 1$, and $K = 7$. (The circle shown is the unit circle.)

Since $\hat{Q}(y) = (1-y)Q(y)$, where $Q(y)$ is the denominator of $T(y)$, it follows from the lemma that $T(y)$ has only one pole inside the unit circle for $K > 2$. Since $T_l \geq 0$ for all l, the pole inside the unit circle is real and positive. Finally, we state the theorem as follows.

Theorem 4.2 *For $n \leq 4$ and $K > 2$, there is only one pole of $\overline{A}(x)$ inside the unit circle.*

Proof. From (4.11),

$$\overline{A}(x) = T\left(\left(\frac{1+x}{2}\right)^n\right).$$

If x is a pole of $\overline{A}(x)$ and y is a pole of $T(y)$, then

$$\left(\frac{1+x}{2}\right)^n = y,$$

or equivalently

$$x = \left(2y^{\frac{1}{n}} - 1\right).$$

It follows that

$$|x| < 1 \qquad \text{if and only if} \qquad \left|y^{\frac{1}{n}} - \frac{1}{2}\right| < \frac{1}{2}.$$

From Lemma 4.1, for $K > 2$ there is only one pole y (real and positive) of $T(y)$ inside the unit circle, and hence for $n \leq 4$ there is only one n-th root of y inside the circle $|z - (1/2)| = 1/2$ as seen from Figure 4.2. ∎

Thus for $n \leq 4$ and $K > 2$, the approximation

$$\overline{A}_d \approx \beta\hat{\alpha}^d$$

is very accurate for large d because $\hat{\alpha}$ is the reciprocal of the only pole inside the unit circle for $\overline{A}(x)$ and all the other terms neglected in the approximation approach 0 as $d \to \infty$.

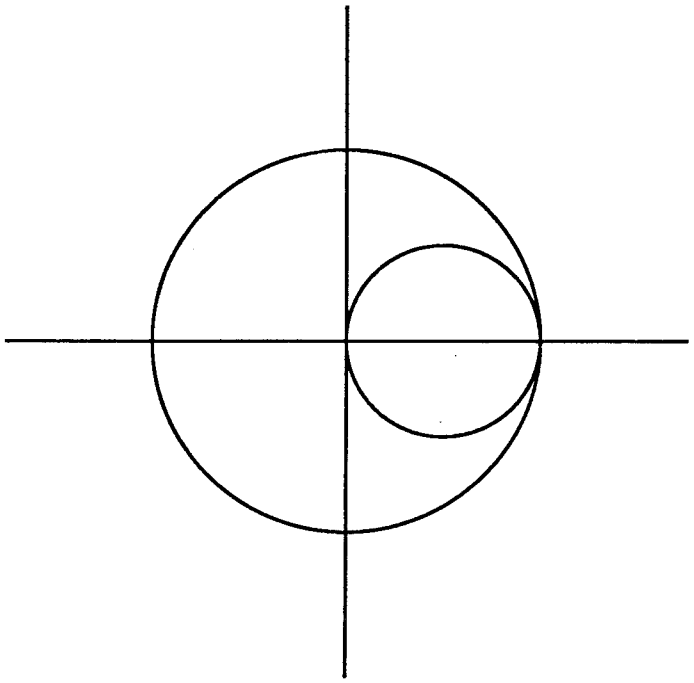

Figure 4.2: The unit circle $|z| = 1$ and the circle $|z - (1/2)| = 1/2$.

4.4 Free Distance Bound

In this section we first estimate the average free distance and then present a Gilbert-like free distance lower bound for time-varying convolutional codes. If we define $\hat{d}_{\text{free}}$ to be

$$\hat{d}_{\text{free}} = \min\left\{ d : \ \overline{A}_d \geq 1 \right\}, \tag{4.13}$$

then intuitively $\hat{d}_{\text{free}}$ constitutes an estimate of the average free distance for a random time-varying convolutional code. Furthermore, the following theorem gives a lower bound on free distance:

Theorem 4.3 *There exists an (n, k) time-varying convolutional code with con-*

straint length K and free distance d_{free} satisfying

$$\sum_{i=0}^{d_{\text{free}}} \overline{A}_i \geq 1.$$

Proof. The sum $\sum_{i=0}^{d_{\text{free}}} A_i$ must be greater than or equal to 1 for any time-varying code of given rate and constraint length if d_{free} is the corresponding maximum free distance. Since $\overline{A}_d$ is the average of A_d in the ensemble of time-varying convolutional codes, the theorem follows. ∎

This lower bound guarantees the existence of at least one time-varying code such that the free distance is bounded below. We can easily see that the lower bound is not larger than $\hat{d}_{\text{free}}$. It will become clear that the bound in Theorem 4.3 is Gilbert-like if we consider its analogy for binary block codes. For random (n,k) block codes, $\binom{n}{d}/2^{n-k}$ is the average number of codewords of weight d. The inequality $\sum_{i=0}^{d} \overline{A}_i \geq 1$ is hence equivalent to

$$\sum_{i=0}^{d} \binom{n}{i} \geq 2^{n-k},$$

which is just the Gilbert-Varshamov bound for block codes except that the upper limit of the summation should be $d-1$ for the bound.

Costello [6] obtained a Gilbert-type asymptotic lower bound on free distance for nonsystematic time-varying convolutional codes:

$$\lim_{K\to\infty} \frac{d_{\text{free}}}{nK} \geq \frac{R(1-2^{R-1})}{H(2^{R-1})+R-1}, \tag{4.14}$$

where $H(x)$ is the binary entropy function. Based on random coding arguments, Forney [8] gave a bound that is not restricted to linear convolutional codes. For the binary case, it says that there exists an (n,k) trellis code with memory m and free distance d_{free} satisfying

$$d_{\text{free}} \geq \max_{\substack{2/(1+e^{-\alpha})>2^R \\ \alpha\geq 0}} \left(nm \frac{\ln 2 - \ln(1+e^{-\alpha})}{\alpha} + \frac{\theta}{\alpha} \right), \tag{4.15}$$

where θ is a constant independent of m. It is shown in [8] that asymptotically (i.e., as $m \to \infty$), the bound (4.15) can be put in Costello's form (4.14). The following theorem is about the asymptotic behavior of the bound in Theorem 4.3.

Theorem 4.4 *Asymptotically, the bound in Theorem 4.3 can be put in Costello's form (4.14).*

Proof. We only need to show that the lower bound in Theorem 4.3 can be put in the form of Forney's bound (4.15). By using the Chernoff bound and the generating function $\overline{A}(x)$ of $\overline{A}_d$,

$$\sum_{i=0}^{d_{\text{free}}} \overline{A}_i \leq \sum_{i=0}^{\infty} \overline{A}_i e^{-\alpha(i-d_{\text{free}})} = e^{\alpha d_{\text{free}}} \overline{A}(e^{-\alpha}), \qquad \alpha \geq 0. \tag{4.16}$$

Thus, the bound in Theorem 4.3 can be given by

$$\min_{\alpha \geq 0} e^{\alpha d_{\text{free}}} \overline{A}(e^{-\alpha}) \geq 1,$$

where the minimization occurs because we want the inequality in (4.16) as tight as possible. The above inequality can be rewritten as

$$d_{\text{free}} \geq \max_{\alpha \geq 0} \frac{1}{\alpha} \left(\ln \frac{1}{\overline{A}(e^{-\alpha})} \right). \tag{4.17}$$

Using (4.12), we obtain

$$\begin{aligned} \ln \frac{1}{\overline{A}(e^{-\alpha})} &= nK[\ln 2 - \ln(1+e^{-\alpha})] - \ln(2^k - 1) \\ &\quad + \ln \left(1 - (2^k - 1) \sum_{j=1}^{K-1} 2^{-nj} (1+e^{-\alpha})^{nj} \right). \end{aligned} \tag{4.18}$$

We now upperbound $\sum_{j=1}^{K-1} 2^{-nj}(1+e^{-\alpha})^{nj}$ by

$$\begin{aligned} \sum_{j=1}^{K-1} \left(\frac{1+e^{-\alpha}}{2} \right)^{nj} &\leq \sum_{j=1}^{\infty} \left(\frac{1+e^{-\alpha}}{2} \right)^{nj} \\ &= \left(\frac{1+e^{-\alpha}}{2} \right)^{n} \Big/ \left[1 - \left(\frac{1+e^{-\alpha}}{2} \right)^{n} \right]. \end{aligned} \tag{4.19}$$

The inequality (4.19) will become an equality as $K \to \infty$. For (4.18) to be valid, we must have

$$(2^k - 1) \sum_{j=1}^{K-1} \left(\frac{1 + e^{-\alpha}}{2} \right)^{nj} < 1,$$

and by (4.19) this yields

$$2^k \left(\frac{1 + e^{-\alpha}}{2} \right)^n < 1,$$

which is equivalent to the required condition in the maximization domain of (4.15):

$$\frac{2}{1 + e^{-\alpha}} > 2^R. \tag{4.20}$$

If we set θ, a constant independent of m, to be

$$\begin{aligned} \theta &= n[\ln 2 - \ln(1 + e^{-\alpha})] - \ln(2^k - 1) \\ &\quad + \ln \left\{ 1 - (2^k - 1) \left(\frac{1 + e^{-\alpha}}{2} \right)^n \Big/ \left[1 - \left(\frac{1 + e^{-\alpha}}{2} \right)^n \right] \right\}, \end{aligned}$$

then from (4.18),

$$\ln \frac{1}{A(e^{-\alpha})} \geq nm \left[\ln 2 - \ln(1 + e^{-\alpha}) \right] + \theta, \tag{4.21}$$

which will become an equality as $m \to \infty$. Combining (4.17), (4.20), and (4.21), we have put the bound in Theorem 4.3 into Forney's form (4.15). ∎

Based on the idea in [8], a more general lower bound on free distance for trellis-coded modulation schemes was derived in [9] [10]. The bound can be applied to time-varying convolutional codes. For the binary case, it says that there exists an (n, k) time-varying convolutional code with memory m and free distance d_{free} satisfying

$$d_{\text{free}} \geq \max_{\substack{\mathrm{E}(\alpha,\rho) > k \ln 2 \\ \alpha \geq 0 \\ 0 \leq \rho \leq 1 \\ p(\mathbf{y})}} \left(m \frac{\mathrm{E}(\alpha, \rho)}{\alpha} + \frac{\theta[\mathrm{E}(\alpha, \rho)]}{\alpha} \right), \tag{4.22}$$

where

$$\mathrm{E}(\alpha, \rho) = -\ln \left[\sum_{\mathbf{y} \in S} p(\mathbf{y}) \left[\sum_{\mathbf{y}' \in S} p(\mathbf{y}') e^{-\alpha d(\mathbf{y}, \mathbf{y}')} \right]^{\rho} \right]^{1/\rho}, \tag{4.23}$$

and

$$\theta[\mathrm{E}(\alpha,\rho)] = \ln\left(e^{\rho[\mathrm{E}(\alpha,\rho)-k\ln 2]}-1\right)^{1/\rho} - \ln\left(\frac{2^k-1}{2^k}\right). \tag{4.24}$$

The signal set S consists of all binary n-tuples, and $p(\mathbf{y})$ is the probability of a particular n-tuple $\mathbf{y}$; also $d(\mathbf{y},\mathbf{y}')$ denotes the Hamming distance between $\mathbf{y}$ and $\mathbf{y}'$. Compared with Forney's bound (4.15), this bound has extra parameters ρ and $p(\mathbf{y})$ in the maximization domain. However, we find that if $\mathrm{E}(\alpha,\rho)$ is maximized by equiprobable signals, i.e., $p(\mathbf{y}) = 1/2^n$ for all $\mathbf{y} \in S$, then

$$\begin{aligned}
\mathrm{E}(\alpha,\rho) &= -\ln\left[\sum_{\mathbf{y}\in S}\frac{1}{2^n}\left(\frac{1}{2^n}\sum_{\mathbf{y}'\in S}e^{-\alpha d(\mathbf{y},\mathbf{y}')}\right)^{\rho}\right]^{1/\rho} \\
&= -\ln\left[\left(\frac{1+e^{-\alpha}}{2}\right)^{n\rho}\right]^{1/\rho} \\
&= n[\ln 2 - \ln(1+e^{-\alpha})]
\end{aligned}$$

regardless of what the value of ρ is. Since $\theta[\mathrm{E}(\alpha,\rho)]$ is independent of m, we can hence put the bound (4.22) in Forney's form if $\mathrm{E}(\alpha,\rho)$ is maximized by equal probabilities. For this case, the asymptotic behavior of the bound (4.22) will be the same as Costello's bound (4.14). We do not need the extra condition $\rho = 1$ as required in [9]. Although it is claimed in [9] that $\partial\mathrm{E}(\alpha,\rho)/\partial\rho < 0$ and the asymptotic form of the bound (4.22) is tighter than Costello's bound (4.14), actually we will not get improvement by having ρ if $\mathrm{E}(\alpha,\rho)$ is maximized by equal probabilities $p(\mathbf{y})$. We conjecture that the asymptotic behavior of the bound (4.22) will always be the same as Costello's bound (4.14).

In [9] another free distance lower bound is derived on expurgated sets of codes that meet some "adjacent distance" requirements. This bound has the same asymptotic behavior as the random coding bound (4.22) but gives better results for small constraint lengths. The random coding bound in [9] (Equation (4.22)), the expurgated bound in [9], our lower bound in Theorem 4.3, $\hat{d}_{\mathrm{free}}$ in (4.13), and

K	rand. bound [1]	exp. bound [2]	lower bound [3]	$\hat{d}_{\text{free}}$	max d_{free} [4]
3	1	5	3	3	5
4	2	5	4	4	6
5	2	5	4	5	7
6	3	6	5	5	8
7	4	6	6	6	10
8	4	7	7	7	10
9	5	8	7	8	12
10	6	8	8	9	12
11	6	9	9	9	14
12	7	10	10	10	15
13	8	10	10	11	16
14	8	11	11	12	16
15	9	12	12	13	18
16	10	12	13	13	19

[1] Random coding bound in [9].

[2] Expurgated bound in [9].

[3] Lower bound in Theorem 4.3.

[4] Maximum free distance for noncatastrophic fixed codes [11].

Table 4.3: Lower bounds on free distance for $(2,1)$ convolutional codes.

the maximum d_{free} of fixed noncatastrophic codes for $(2,1)$ and $(4,1)$ convolutional codes are listed in Table 4.3 and 4.4. For small constraint lengths, our lower bound is better than the random coding bound in [9] but slightly worse than the expurgated bound in [9]. As K increases, our lower bound becomes closer to the expurgated bound in [9]. It should be noted that either the random coding bound or the expurgated bound in [9] involves maximization over several parameters, while our lower bound is simpler and requires only computation of series expansion of a known rational function, which can be done by either long division or iterative methods.

K	rand. bound [1]	exp. bound [2]	lower bound [3]	$\hat{d}_{\text{free}}$	max d_{free} [4]
3	4	9	6	10	10
4	5	10	8	11	13
5	7	13	10	13	16
6	8	14	12	14	18
7	10	16	14	16	20
8	12	17	16	18	22
9	13	19	17	20	24
10	15	20	19	22	27
11	16	22	21	23	29
12	18	23	23	25	32
13	20	25	25	27	33
14	21	27	26	29	36

[1]Random coding bound in [9].

[2]Expurgated bound in [9].

[3]Lower bound in Theorem 4.3.

[4]Maximum free distance for noncatastrophic fixed codes [11].

Table 4.4: Lower bounds on free distance for $(4, 1)$ convolutional codes.

4.5 Average Weight of Information Bits for Fundamental Paths

In this section we give a similar random coding analysis for the total Hamming weight of information bits for fundamental paths at each distance. The *complete path enumerator* [1] is defined by

$$A(x, y, z) = \sum_d \sum_l \sum_i A_{d,l,i} x^d y^l z^i,$$

where $A_{d,l,i}$ is the number of fundamental paths of distance d, length l, and input weight i. In principle we can compute $A(x, y, z)$ by the same method in obtaining $A(x)$. The complete path enumerator $A(x, y, z)$ is in general a rational function

with integer coefficients:

$$A(x,y,z) = \frac{N(x,y,z)}{D(x,y,z)}.$$

Define $B_d = \sum_l \sum_i i A_{d,l,i}$, the total number of information bits for fundamental paths at distance d. Then the generating function $B(x)$ of B_d is

$$\begin{aligned} B(x) &= \left.\frac{\partial A(x,y,z)}{\partial z}\right|_{y=z=1} \\ &= \frac{N_z(x,1,1)D(x,1,1) - N(x,1,1)D_z(x,1,1)}{D^2(x,1,1)} \\ &= \frac{N_z(x,1,1)D(x) - N(x)D_z(x,1,1)}{D^2(x)}, \end{aligned} \tag{4.25}$$

where $N(x)$ and $D(x)$ are the numerator and the denominator of $A(x)$, and the subscript z means partial differentiation with respect to z. The values B_d are important because for maximum-likelihood decoding the bit error probability can be bounded above by

$$P_b \leq \frac{1}{k} \left.\frac{\partial A(x,y,z)}{\partial z}\right|_{x=\gamma,y=z=1} = \frac{1}{k} B(x)\Big|_{x=\gamma}, \tag{4.26}$$

where γ is the channel's Bhattacharyya parameter [1]. It is important to note that the bound diverges at α^{-1}, as does that for the first-event error probability.

Now we want to find the average $\overline{B_d}$ in the ensemble of all (n,k) time-varying convolutional codes of constraint length K. Let $T_{l,i}$ denote the number of fundamental paths of length l and input weight i. By using a uniform probability measure,

$$\overline{A}_{d,l,i} = \binom{nl}{d} \frac{T_{l,i}}{2^{nl}}.$$

Thus,

$$\begin{aligned} \overline{B}_d &= \sum_{l=0}^{\infty} \sum_{i=0}^{\infty} i \overline{A}_{d,l,i} \\ &= \sum_{l=\lceil \frac{d}{n} \rceil}^{\infty} \frac{\binom{nl}{d}}{2^{nl}} \sum_{i=0}^{\infty} i T_{l,i}. \end{aligned} \tag{4.27}$$

The generating function $T(y,z)$ of $T_{l,i}$ is found in Appendix 4.A. Define

$$S_l = \sum_{i=0}^{\infty} iT_{l,i}. \tag{4.28}$$

Then its generating function is

$$S(y) = \left.\frac{\partial T(y,z)}{\partial z}\right|_{z=1} = \frac{k2^{k-1}y^K}{\left[1-(2^k-1)\sum_{j=1}^{K-1} y^j\right]^2}.$$

Similar to Section 4.2, $S(y)$ can be partial-fraction expanded to

$$S(y) = \sum_{i=1}^{K-1}\left[\frac{e_i}{1-r_i y} + \frac{f_i}{(1-r_i y)^2}\right],$$

where r_i, $i = 1,2,\ldots,K-1$, are the reciprocals of the roots of $1-(2^k-1)\sum_{j=1}^{K-1} y^j = 0$ and

$$e_i = -\frac{1}{r_i}\frac{d}{dy}\left[S(y)(1-r_i y)\right]\Big|_{y=r_i^{-1}}, \tag{4.29}$$

$$f_i = S(y)(1-r_i y)^2\Big|_{y=r_i^{-1}}. \tag{4.30}$$

Therefore,

$$S_l = \sum_{i=1}^{K-1}\left[e_i r_i^l + f_i(l+1)r_i^l\right]. \tag{4.31}$$

Substituting (4.28) and (4.31) back to (4.27) and using the same technique in Section 4.2, we finally obtain the following theorem:

Theorem 4.5 *For $K > 2$,*

$$\overline{B}_d = \frac{1}{n}\sum_{i=1}^{K-1}\sum_{\delta:\delta^n=r_i}\left\{\frac{e_i+f_i}{1-\frac{\delta}{2}}\left(\frac{\delta}{2-\delta}\right)^d + \frac{1}{n}\left[\frac{f_i}{1-\frac{\delta}{2}}\left(\frac{\delta}{2-\delta}\right)^{d+1} + \frac{f_i}{\left(1-\frac{\delta}{2}\right)^2}\cdot d\left(\frac{\delta}{2-\delta}\right)^d\right]\right\},$$

where r_i, $i = 1,2,\ldots,K-1$, are the reciprocals of the roots of $1-(2^k-1)\sum_{j=1}^{K-1} y^j = 0$ and e_i, f_i are given in (4.29) and (4.30), respectively.

Hence for large d,

$$\overline{B}_d \approx \beta_1 \left(\frac{\sqrt[n]{r}}{2 - \sqrt[n]{r}} \right)^d + \beta_2 \cdot d \left(\frac{\sqrt[n]{r}}{2 - \sqrt[n]{r}} \right)^d,$$

where r is the reciprocal of the least-magnitude root of $1 - (2^k - 1) \sum_{j=1}^{K-1} y^j = 0$. This result can be expected from the expression of (4.25).

Similar to $\overline{A}(x)$, the generating function $\overline{B}(x)$ of B_d is found by

$$\begin{aligned} \overline{B}(x) &= \sum_{l=0}^{\infty} \sum_{d=0}^{nl} \frac{\binom{nl}{d}}{2^{nl}} S_l x^d = \sum_{l=0}^{\infty} \frac{S_l}{2^{nl}} (1+x)^{nl} \\ &= S\left(\left(\frac{1+x}{2} \right)^n \right) = \frac{k 2^{nK+k-1} (1+x)^{nK}}{\left[2^{nK} - (2^k - 1) \sum_{j=1}^{K-1} 2^{n(K-j)} (1+x)^{nj} \right]^2}. \end{aligned}$$

4.6 An Example of the Behavior of Convolutional Codes at Low Signal-to-Noise Ratios

In this section we shall give an interesting example about the behavior of convolutional codes at low signal-to-noise ratios. In [12] we conjecture that for a given code rate, the smaller the value of the dominant root, the better the convolutional code will perform at low signal-to-noise ratios. However, after extensive computer simulations, the conjecture is proved to be false by the following counterexample. Here we have three $(2, 1)$ convolutional codes all with constraint length 5: Codes A, B, and C. Their generator polynomials, free distances, dominant roots, and distance spectrums are shown in Table 4.5.

Results of computer simulations for these three codes with Viterbi decoding (with 32-bit truncation length) on an unquantized AWGN channel with binary phase-shift keying (BPSK) are illustrated in Figure 4.3. The bit error probability P_b is plotted as a function of the bit signal-to-noise ratio E_b/N_0 in decibels. Shown also in Figure 4.3 is the no-coding curve. Code A has the largest free distance

	Code A		Code B		Code C	
g_1, g_2	11001, 11011		10001, 11111		10001, 10101	
d_{free}	7		6		5	
α	2.30034		2.34632		2.32637	
d	A_d	B_d	A_d	B_d	A_d	B_d
5	0	0	0	0	1	1
6	0	0	1	2	2	4
7	2	4	1	1	4	12
8	4	12	3	10	8	32
9	6	26	5	15	16	80
10	15	74	12	52	34	196
11	37	205	27	123	75	481
12	83	530	61	346	170	1192
13	191	1369	144	926	392	2984
14	442	3504	334	2492	912	7520
15	1015	8849	789	6675	2129	18995
16	2334	22180	1847	17594	4973	47924
17	5371	55235	4347	46091	11609	120509
18	12353	136720	10203	119278	27074	301708
19	28414	336732	23963	306475	63084	751860
20	65364	825768	56246	781096	146889	1865284
21	150359	2017233	132005	1978601	341870	4608678
22	345876	4911042	309773	4983836	795453	11345518
23	795636	11919854	726856	12494136	1850572	27840404
24	1830234	28852304	1705495	31191560	4304973	68123240

Table 4.5: Distance spectrums for Codes A, B, and C.

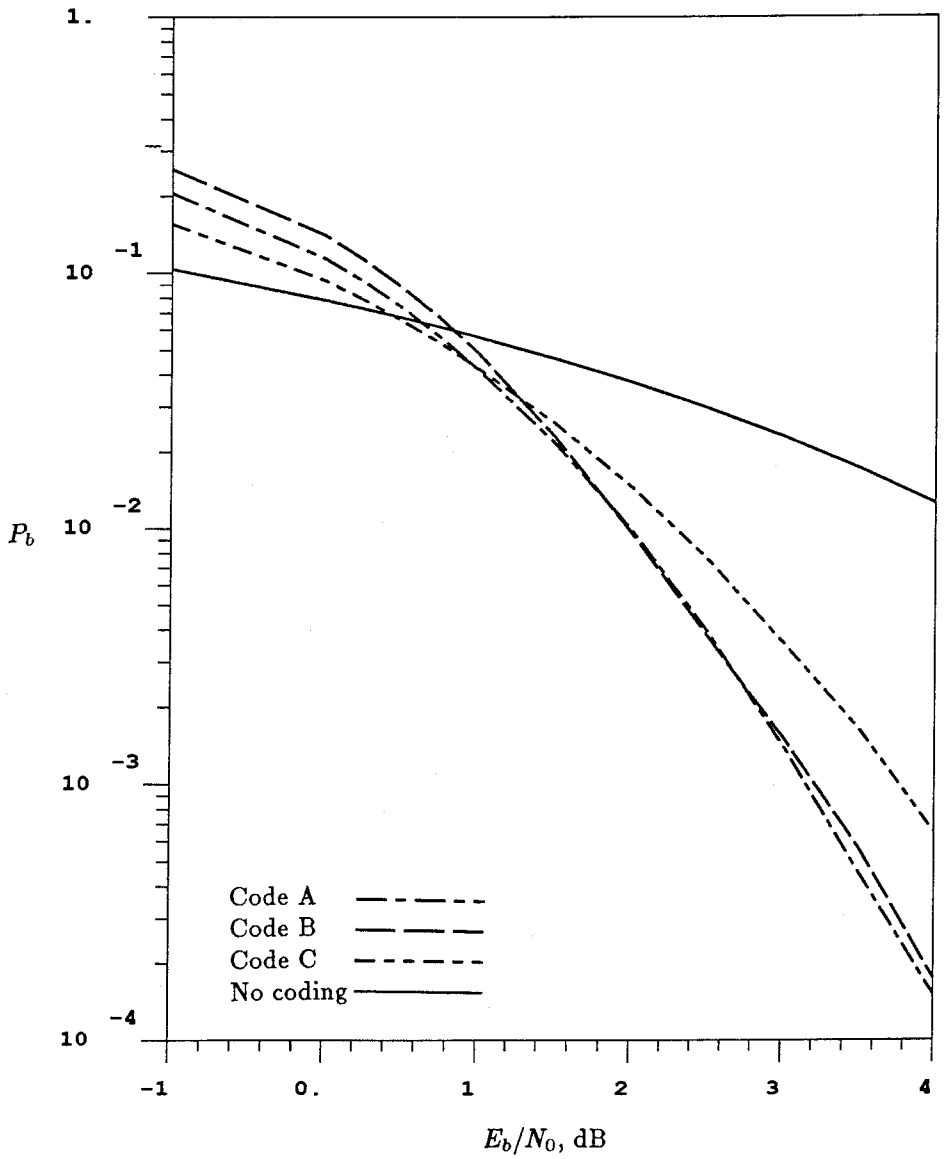

Figure 4.3: Performance curves for Codes A, B, C, and no coding.

among the three codes, so it will perform the best at sufficiently high signal-to-noise ratios. Note that Code C outperforms the other two codes for E_b/N_0 less than 0.8 dB, although it has a larger dominant root than Code A (and has the smallest free distance). This hence gives a counterexample to the conjecture in [12]. Also note that Code C has both A_d and B_d larger than Code A for all d. Regarding A_d and B_d at free distance, Code A has $A_7 = 2$ and $B_7 = 4$, which are larger than $A_6 = 1$ and $B_6 = 2$, respectively, for Code B. However, for E_b/N_0 less than 1.8 dB, Code A performs better than Code B, showing that the so-called "error coefficient" at free distance (or the number of "nearest neighbors") is not a good criterion for the performance of convolutional codes at low signal-to-noise ratios.

The transfer function bounds (4.2) and (4.26) diverge at $E_b/N_0 = \ln \alpha/R \approx 2.2$ dB for these three codes. This example indicates that, at signal-to-noise ratios where the transfer function bound diverges, conclusions drawn from the bound cannot be used to estimate or predict the code performance.

4.7 Extensions

In this chapter we only deal with binary convolutional codes, and hence an immediate generalization of this work is to general q-ary convolutional codes. Another possibility is extension to trellis-coded modulation schemes, which find important applications to band-limited channels. But then the distance measure between codewords is Euclidean distance (real number) instead of Hamming distance (integer), and the codes are nonlinear. Nevertheless, we think that a similar random coding analysis can be done for trellis-coded modulation schemes.

Appendix 4.A Derivation of (4.5)

In this appendix we shall derive $T(y)$ given in (4.5) by extending the argument in [2, Sec. 4.6], where $T(y)$ for any $(n,1)$ code is found. Actually we shall find $T(y,z)$ first, where the z terms describe the Hamming weights of the corresponding input bits. Consider any (n,k) convolutional code of constraint length K with $2^{k(K-1)}$ states, whose encoder has k shift registers all of the same length K. Then $T(y,z)$ is independent of n and the generator polynomials. It is a function only of k and the constraint length K. Since we shall use recursions on K to get the expression for $T(y,z)$, we change the notation to $T_K(y,z)$ to emphasize the constraint length. We claim that the following recursion holds, which is the same as [2, (4.6.2)]:

$$T_K(y,z) = y \cdot T_{K-1}(y,z) + T_{K-1}(y,z)T_K(y,z), \quad K \geq 2. \tag{4.32}$$

Consider all $2^k - 1$ states with a branch into the all-zero state in the state diagram. Suppose that all paths reaching any one of these states are absorbed. Then the generating function enumerating those paths is just $T_{K-1}(y,z)$ because we may ignore the initial inputs in all shift registers as if we had a code of constraint length $K-1$. If the following input is all-zero, then we get a fundamental path and this case constitutes the first term of (4.32). If the following input is anything other than all-zero, then we are in the same situation as leaving the all-zero state. These paths are enumerated by $T_K(y,z)$, justifying the second term of (4.32).

The initial condition for (4.32) is

$$T_1(y,z) = \sum_{i=1}^{k} \binom{k}{i} yz^i = \left[(1+z)^k - 1\right] y$$

because any input other than all-zero produces a fundamental path of length 1 for

a code of constraint length 1. Then from (4.32) by induction we can easily show

$$T_K(y,z) = \frac{\left[(1+z)^k - 1\right] y^K (1-y)}{1 - y\left\{1 + \left[(1+z)^k - 1\right](1 - y^{K-1})\right\}}.$$

We therefore obtain the expression for $T_K(y)$:

$$T_K(y) = T_K(y,z)|_{z=1} = \frac{(2^k-1)y^K(1-y)}{1 - 2^k y + (2^k-1)y^K},$$

which reduces to [2, (4.6.6)] for $k = 1$.

Appendix 4.B Derivation of (4.7)

In this appendix we prove that $\hat{Q}(y)$ and its derivative $\hat{Q}'(y)$ are relatively prime for $K > 2$, where $\hat{Q}(y) = 1 - 2^k y + (2^k - 1)y^K$, by showing that

$$\gcd\left(\hat{Q}(y), \hat{Q}'(y)\right) \neq 1 \qquad \text{if and only if} \quad K = 2 \text{ and } k = 1.$$

If $K = 1$, then $\hat{Q}(y) = 1 - y$, which is squarefree. For $K \geq 2$, $\hat{Q}'(y) = K(2^k - 1)y^{K-1} - 2^k$. We can now find their g.c.d. by Euclid's algorithm. By long division,

$$\hat{Q}(y) = \frac{1}{k} y \cdot \hat{Q}'(y) + \left(-\frac{2^k(K-1)}{K} y + 1\right).$$

Hence $\gcd(\hat{Q}(y), \hat{Q}'(y)) \neq 1$ holds if and only if

$$-\frac{2^k(K-1)}{K} y + 1 \;\Big|\; \hat{Q}'(y),$$

or equivalently,

$$\left[K(2^k-1)y^{K-1} - 2^k\right]\Big|_{y=\frac{K}{2^k(K-1)}} = 0,$$

which yields

$$K^K(2^k - 1) = 2^{kK}(K-1)^{K-1}. \tag{4.33}$$

K cannot be odd and in fact (4.33) holds if and only if $K = 2$ and $k = 1$. Suppose $K = p2^t$, where p is an odd integer. Then (4.33) becomes

$$p^K 2^{tK}(2^k - 1) = 2^{kK}(p2^t - 1)^{K-1}. \tag{4.34}$$

As p is odd, $\gcd(p^K, 2^{kK}) = 1$. It follows that for (4.34) to hold we must have $p^K | (p2^t - 1)^{K-1}$. However, $p \not| (p2^t - 1)^{K-1}$ unless $p = 1$. Substituting $K = 2^t$ back into (4.34), we obtain

$$2^{t2^t}(2^k - 1) = 2^{k2^t}(2^t - 1)^{2^t - 1},$$

which implies

$$t = k \quad \text{and} \quad 2^t - 1 = 1,$$

so $t = k = 1$ and $K = 2$.

References

[1] R. J. McEliece, *The Theory of Information and Coding.* Reading, MA: Addison-Wesley, 1977.

[2] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding.* New York: McGraw-Hill, 1979.

[3] E. C. Titchmarsh, *The Theory of Functions.* London: Oxford University Press, 1939.

[4] S. Lin and D. J. Costello, Jr., *Error-Control Coding: Fundamentals and Applications.* Englewood Cliffs, NJ: Prentice-Hall, 1983.

[5] G. D. Forney, Jr., "Structural analysis of convolutional codes via dual codes," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 512–518, July 1973.

[6] D. J. Costello, Jr., "Free distance bounds for convolutional codes," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 356–365, May 1974.

[7] A. J. Viterbi, "Convolutional codes and their performance in communication systems," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 751–772, Oct. 1971.

[8] G. D. Forney, Jr., "Convolutional codes: II — Maximum-likelihood decoding," *Inform. Contr.*, vol. 25, pp. 222–266, 1974.

[9] M. Rouanne, "Distance bounds and construction algorithms for trellis codes," Ph.D. dissertation, University of Notre Dame, Notre Dame, IN, 1988.

[10] M. Rouanne and D. J. Costello, Jr., "A lower bound on the minimum Euclidean distance of trellis-coded modulation schemes," *IEEE Trans. Inform. Theory*, vol. IT-34, pp. 1011–1020, Sep. 1988.

[11] K. J. Larsen, "Short convolutional codes with maximum free distance for rates 1/2, 1/3, and 1/4," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 371–372, May 1973.

[12] C.-C. Chao and R. J. McEliece, "On the path weight enumerators of convolutional codes," *Proc. 26th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Sep. 1988, pp. 1049–1058.

CHAPTER 5

ERROR STATISTICS OF VITERBI DECODING AND A MARKOV CHAIN MODEL

5.1 Introduction

The Viterbi algorithm is an effective way of decoding convolutional codes. It recursively finds the maximum-likelihood path through the code's trellis diagram. One of the characteristics of the Viterbi algorithm is that the decoding errors tend to occur in clusters or bursts. In some applications, burst error statistics can be important design considerations. One example is a concatenated coding system in which the inner convolutional code is Viterbi decoded and the outer code (sometimes interleaved) should correct most of the Viterbi decoder error bursts. In this chapter, a Markov chain model for the burst error statistics of a Viterbi decoder is developed.

For simplicity, only $(n, 1)$ convolutional codes are considered here; the results are easily generalized to (n, k) codes. The notation $(n, 1, K)$ $g_1, g_2, \ldots, g_n$ denotes a rate $1/n$ convolutional code with constraint length K and octal generator polynomials $g_1, g_2, \ldots, g_n$. Consider a sequence of Viterbi decoder output bits of the form

$$\underbrace{cc\cdots c}_{K-1}\overbrace{exx\cdots xe}^{B}\underbrace{cc\cdots c}_{K-1},$$

where the letter c represents a correctly decoded bit, e represents a decoder bit error, and x is either c or e. If the string $xx\cdots x$ does not contain $K-1$ consecutive

c's, then the string $exx\cdots xe$ is called a *burst* of length B. The string of c's between two bursts is called a *guardspace* (or *waiting time*) with length $G \geq K-1$.

It is shown in [1] that the average burst length distribution for time-varying convolutional codes can be upperbounded by a geometric distribution. In [2] geometric distributions are used to model both the burst length and guardspace length distributions. In [3] [4] Best shows that any convolutional coding scheme with maximum-likelihood decoding on a discrete memoryless channel can be modelled exactly by a finite state Markov chain (a metric-state diagram), and hence the Viterbi decoder output burst and guardspace lengths are distributed asymptotically (but not exactly) geometrically. Although this approach yields exact Viterbi decoder output characteristics, the excessive amount of computation required makes it infeasible for practical codes. For example, the $(2,1,3)$ $3,5$ convolutional code on a binary symmetric channel has a 104-state Markov chain. The (approximate) Markov chain model described in this chapter has the same 2^{K-1} states as the encoder. The resulting burst length distribution is asymptotically geometric and the guardspace length is distributed (exactly) geometrically.

In Section 5.2, we review some results from [3] [4]. A Markov chain model to approximate Viterbi decoder output error statistics is developed in Section 5.3. In Section 5.4, our Markov chain model is validated by computer simulations and it is compared with the geometric model. Based upon distance measures for burst length and guardspace length distributions, our Markov chain model is a better approximation to the actual simulation of the Viterbi decoder than the geometric model in [2]. However, both models closely approximate the overall decoder output error probabilities of a concatenated Reed-Solomon/convolutional coding system.

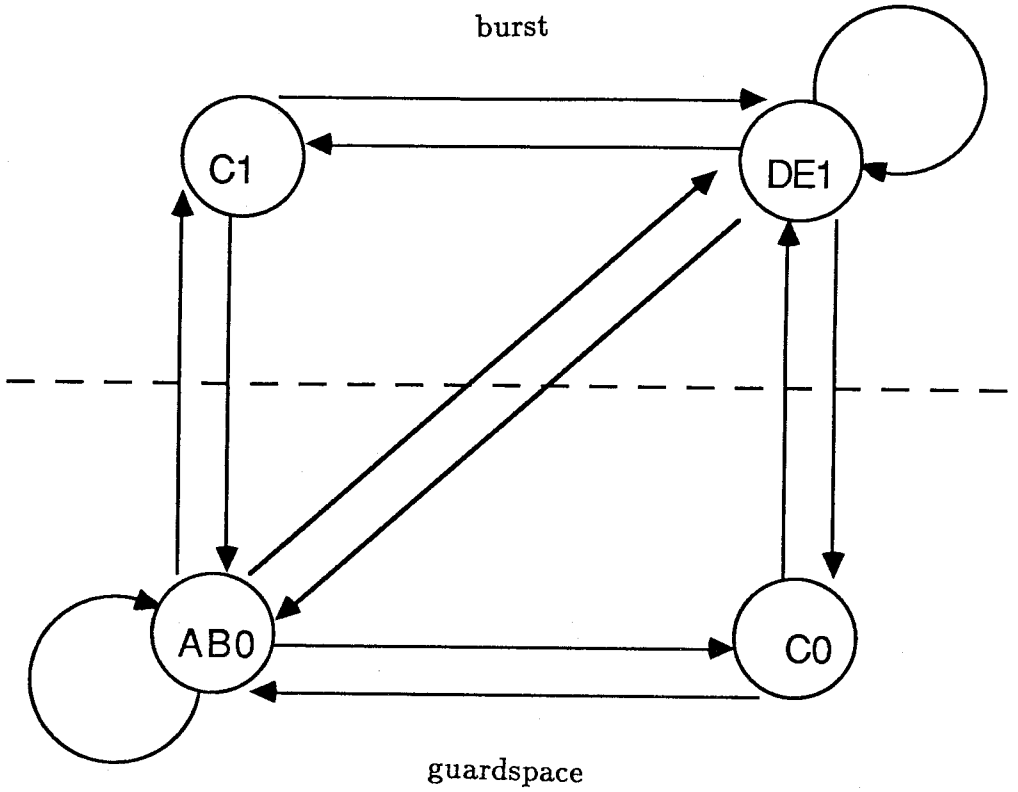

Figure 5.1: Markov chain for the $(2,1,2)$ $1,3$ convolutional code on a binary symmetric channel.

5.2 Review of Best's Results

In [3] [4] a method is presented by which any convolutional coding scheme with maximum-likelihood decoding on a discrete memoryless channel can be modelled as a finite state Markov chain (a metric-state diagram) whose transition probabilities can be computed. Some states in the Markov chain correspond to error bursts, the others to guardspaces. An example of the Markov chain is illustrated in Figure 5.1 for the $(2,1,2)$ $1,3$ convolutional code on a binary symmetric channel. The chain shown in Figure 5.1 is actually a metric-state diagram after merging

and deleting redundant nodes. There are two states 0 and 1 in the encoder's state diagram and five effective metrics A, B, C, D, and E for Viterbi decoding, which are [0 2], [0 1], [0 0], [0 −1], [0 −2], respectively. The notation C0 corresponds to metric C and state 0. All the transition probabilities in Figure 5.1 can be calculated explicitly in terms of the channel's crossover probability. Based on this model, the first-event error probability, the bit error probability, the burst length distribution, and the guardspace length distribution may be derived exactly.

For this example, the burst length distribution is

$$\mathrm{P}(B = b) = \alpha_0 \lambda_0^b + \alpha_1 \lambda_1^b,$$

where α_0 and α_1 are scaling factors, and $|\lambda_0| > |\lambda_1|$. Hence for large b, $\mathrm{P}(B = b)$ is approximated by the dominant term $\alpha_0 \lambda_0^b$, which is a geometric distribution. Similarly, the guardspace length distribution is

$$\mathrm{P}(G = g) = \beta_0 \mu_0^g + \beta_1 \mu_1^g,$$

where β_0 and β_1 are scaling factors, and $|\mu_0| > |\mu_1|$. The guardspace length distribution is approximated by the geometric distribution $\beta_0 \mu_0^g$ for large g. Since this method can be applied in principle to any convolutional code with Viterbi decoding on any discrete memoryless channel, the burst length and guardspace length distributions will be asymptotically geometrically distributed:

$$\mathrm{P}(B = b) = \sum_i \alpha_i \lambda_i^b,$$

and

$$\mathrm{P}(G = g) = \sum_j \beta_j \mu_j^g.$$

Although the above analysis is exact, the number of the nodes in the model grows enormously with the code's constraint length, which makes the method infeasible for practical codes.

5.3 A Markov Chain Model

The output of the Viterbi algorithm is a maximum-likelihood path traversing through the code's state diagram. Our model approximates the Viterbi decoder by a finite state Markov chain with the same configuration as that of the code's state diagram. This model is not strictly accurate, since the decoding state at time unit i depends not only on the decoding state at time unit $i-1$ and the channel noise but also on the metrics at time unit $i-1$. However, computer simulations of decoder error statistics in the next section show that this approximation is good for some practical applications. Hereafter, the "Markov chain model" corresponds to the model developed in this section.

An example of the Markov chain model for the $(2,1,3)$ $3,5$ convolutional code is shown in Figure 5.2. The four decoder states are 0, 1, 2, and 3. The most recent bit in the encoder shift-register is the last bit in the state's binary representation. The p_{ij} is the transition probability that the decoder will go to state j at the next time unit, given that it is presently in state i . The transitions not shown all have probability 0.

All the output statistics of the Markov chain model can be computed from the transition probabilities. In the following we compute some of them: the bit error probability, the guardspace length distribution, and the burst length distribution. Suppose a $(n,1,K)$ convolutional code is used and the all-zero code sequence is transmitted. (Since a convolutional code is linear, on a symmetric channel the decoding errors are independent of the transmitted code sequence.) The model has 2^m states called $0,1,\ldots,2^m-1$, where $m=K-1$ is the code's memory.

5.3.1 Bit Error Probability

Let $\mathbf{P}=[p_{ij}]$ be the model's transition probability matrix. Then the column

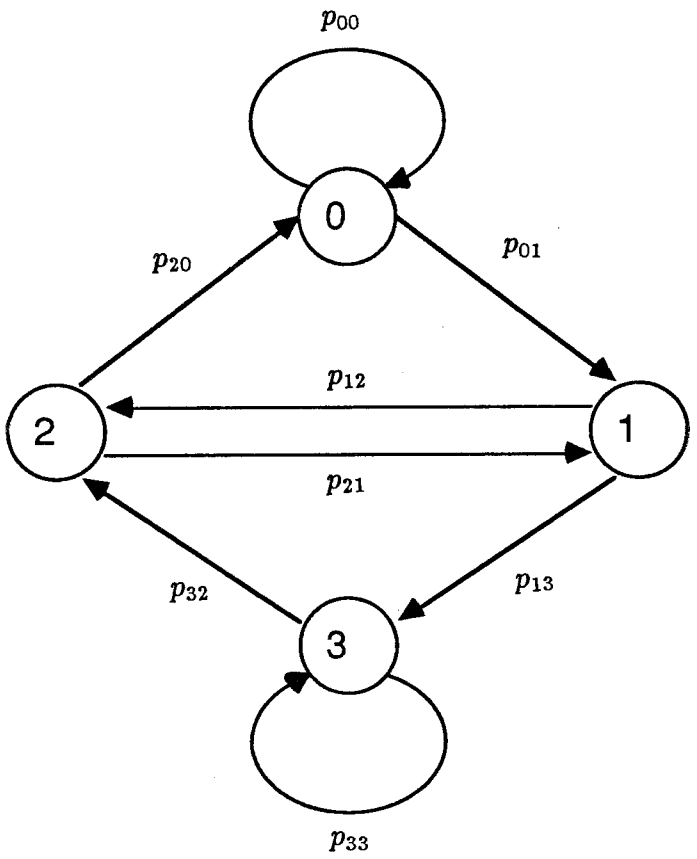

Figure 5.2: Markov chain model for the $(2,1,3)$ 3,5 code.

vector of stationary probabilities of the states, $\pi = [\ \pi_0 \quad \pi_1 \quad \cdots \quad \pi_{2^m-1}\ ]^T$, is just the eigenvector of $\mathbf{P}^T$ with eigenvalue 1:

$$\pi = \mathbf{P}^T \pi.$$

The bit error probability P_b is the sum of the stationary probabilities of the states whose binary representations have a 1 in the last bit:

$$P_b = \sum_{i=1}^{m} \pi_{2^i-1}.$$

5.3.2 Guardspace Length Distribution

The guardspace length $G \geq m$ and is geometrically distributed:

$$\mathrm{P}(G = g) = \begin{cases} p_{01}p_{00}^{g-m}, & \text{for } g \geq m, \\ 0, & \text{otherwise.} \end{cases}$$

The average guardspace length is therefore

$$\overline{G} = \frac{1}{p_{01}} + m - 1.$$

5.3.3 Burst Length Distribution

Let $\mathbf{x}^{(i)}$, $i = 0, 1 \ldots, 2^m - 1$, be eigenvectors of the transition probability matrix $\mathbf{P}$ with corresponding eigenvalues λ_i. Also let $\mathbf{y}^{(i)}$, $i = 0, 1, \ldots, 2^m - 1$, be eigenvectors of $\mathbf{P}^T$ with corresponding eigenvalues λ_i. Assume all the eigenvalues are distinct. Define u_l to be the probability that a path starts from the zero state and arrives at the zero state (with possible intermediate returns) in l branches. For convenience, set $u_0 = 1$. From [5], for $l \geq 1$, u_l is given by

$$u_l = \sum_{i=0}^{2^m-1} c_i x_0^{(i)} y_0^{(i)} \lambda_i^l, \tag{5.1}$$

where

$$c_i = \frac{1}{\sum_{j=0}^{2^m-1} x_j^{(i)} y_j^{(i)}}$$

is a normalization factor and $x_j^{(i)}, y_j^{(i)}$, $j = 0, 1, \ldots, 2^m - 1$, are components of $\mathbf{x}^{(i)}$, $\mathbf{y}^{(i)}$, respectively. Since u_l must remain bounded for all l, it follows that $|\lambda_i| \leq 1$ for all i. The matrix $\mathbf{P}$ always has an eigenvalue 1, so without loss of generality we may put $\lambda_0 = 1$. All the other eigenvalues must satisfy $|\lambda_i| < 1$. Now define f_l to be the probability that a path starts from the zero state and arrives at the zero state for the first time in l branches. It will be convenient to let $f_0 = 0$. From [5], f_l and u_l are related by

$$f_l = u_l - (f_1 u_{l-1} + \cdots + f_{l-1} u_1), \qquad \text{for } l \geq 1, \tag{5.2}$$

or, equivalently, their generating functions $F(z)$ and $U(z)$ satisfy

$$F(z) = 1 - \frac{1}{U(z)}.$$

For our model, $f_1 = u_1 = p_{00}$ and $f_2 = f_3 = \ldots = f_m = 0$. From [5], the average $\overline{f}$ is

$$\overline{f} = \sum_{l=0}^{\infty} l f_l = \frac{1}{c_0 x_0^{(0)} y_0^{(0)}}. \tag{5.3}$$

Since by the definition in Section 5.1, the length of a burst is m branches shorter than that of a fundamental path through the state diagram, the burst length distribution is

$$\mathrm{P}(B = b) = \begin{cases} \frac{f_{b+m}}{1-p_{00}}, & \text{for } b \geq 1, \\ 0, & \text{otherwise}, \end{cases} \tag{5.4}$$

where $1/(1 - p_{00})$ is a normalization factor. From (5.1), (5.2) and (5.4), we can compute the burst length distribution, which is asymptotically geometric. The average burst length is

$$\overline{B} = \sum_{b=0}^{\infty} b\mathrm{P}(B = b) = \sum_{b=1}^{\infty} \frac{b f_{b+m}}{1 - p_{00}} = \frac{\overline{f} - m + (m-1)p_{00}}{1 - p_{00}},$$

where $\overline{f}$ is given in (5.3).

5.4 Simulation Results

In this section, our Markov chain model is validated by computer simulations and it is compared to the geometric model by using distance measures for burst length and guardspace length distributions, and output error probabilities of concatenated Reed-Solomon/convolutional coding schemes. The description of the geometric model proposed in [2] is given in Appendix 5.A.

5.4.1 Distance Measure

A function ρ is a distance function for probability distributions [6] if $\rho(F, G)$ is defined for every pair of F, G of probability distributions and has the following three properties:

$$\rho(F, G) \geq 0 \quad \text{and} \quad \rho(F, G) = 0 \ \text{ if and only if } \ F = G,$$

$$\rho(F, G) = \rho(G, F),$$

and finally the triangle inequality

$$\rho(F_1, F_2) \leq \rho(F_1, G) + \rho(F_2, G).$$

Two types of distance functions will be used here to measure the closeness of two distributions. The total distance TD is

$$TD(F, G) = \sum_{i=0}^{\infty} |f_i - g_i|,$$

where f_i and g_i are densities of discrete probability distributions F and G, respectively. It satisfies

$$0 \leq TD(F, G) \leq 2.$$

The maximum distance MD is

$$MD(F, G) = \max_i |f_i - g_i|,$$

and it satisfies

$$0 \leq MD(F, G) \leq 1.$$

Computer simulations of the Viterbi software decoder, the Markov chain model, and the geometric model on a unquantized AWGN channel have been done for the $(2, 1, 7)$ $177, 133$ convolutional code and the $(2, 1, 5)$ $23, 35$ convolutional code. First, the Viterbi software decoder generates parameters, at several different bit

signal-to-noise ratios needed for both models, such as transition probabilities for the Markov chain model, and the average burst length, the average guardspace length, and the burst error density for the geometric model. Then distributions of the burst length and the guardspace length are computed for both models. Finally, other simulations for the Viterbi software decoder [1] produce statistics for the simulated burst and guardspace length distributions.

The burst length distribution and the guardspace length distribution are compared from three different sources: the Viterbi software decoder, the Markov chain model, and the geometric model, by using TD and MD measures. The results are listed in Tables 5.1, 5.2, 5.3, and 5.4. For the burst length distribution, the Markov chain model gives a better approximation to the actual data from the Viterbi software decoder than the geometric model for both convolutional codes at all bit signal-to-noise ratios tested. For the guardspace length distribution, the Markov chain model and the geometric model are almost the same since both result in geometric distributions. Note that, at relatively high bit signal-to-noise ratios (above 2.5 dB for the $(2,1,7)$ code and 3.0 dB for the $(2,1,5)$ code), the geometric distribution is not a good approximation for the guardspace length. This is because at those bit signal-to-noise ratios, short guardspaces are much less probable than one might expect from a geometric distribution. Based on the distance measure, we conclude that, for Viterbi burst error statistics, our Markov chain model performs better than the geometric model.

5.4.2 Concatenated Coding Scheme

One application of the Markov chain model is that, when different concatenated schemes (all with an inner Viterbi decoder) are studied, one may use the

[1] Here different seeds (from those of former simulations) in the random number generator of the Viterbi software decoder are used.

E_b/N_0,	M & G		M & S		G & S	
dB	TD	MD	TD	MD	TD	MD
0.0	0.09590	0.04013	0.1824	0.04451	0.2318	0.08464
0.5	0.1245	0.05222	0.1618	0.03350	0.2235	0.08572
1.0	0.1415	0.05982	0.1350	0.02276	0.2192	0.08258
1.5	0.1562	0.06139	0.1557	0.03695	0.2555	0.09834
2.0	0.1581	0.06008	0.1010	0.02735	0.2310	0.08743
2.5	0.1737	0.05533	0.1050	0.02225	0.2317	0.05748
3.0	0.2002	0.04970	0.1629	0.03347	0.2485	0.07204

Table 5.1: Distance measure of burst length distributions for the $(2, 1, 7)$, $171, 133$ convolutional code. (S: Viterbi software decoder, M: Markov chain model, G: Geometric model.)

E_b/N_0,	M & G		M & S		G & S	
dB	TD	MD	TD	MD	TD	MD
0.0	0.1778	0.04105	0.09617	0.02019	0.2025	0.05009
0.5	0.2114	0.04873	0.09874	0.01809	0.2334	0.05211
1.0	0.2516	0.05837	0.08217	0.01510	0.2637	0.05040
1.5	0.2971	0.07078	0.08021	0.02015	0.2989	0.05845
2.0	0.3425	0.08406	0.06660	0.01018	0.3354	0.07387
2.5	0.4010	0.09831	0.06406	0.01045	0.3914	0.09594
3.0	0.4636	0.1308	0.08732	0.01334	0.4462	0.1205
3.5	0.5667	0.1750	0.1572	0.03529	0.5356	0.1567
4.0	0.6264	0.1739	0.2222	0.08532	0.5732	0.1849

Table 5.2: Distance measure of burst length distributions for the $(2, 1, 5)$ $23, 35$ convolutional code. (S: Viterbi software decoder, M: Markov chain model, G: Geometric model.)

E_b/N_0,	M & G		M & S		G & S	
dB	TD	MD	TD	MD	TD	MD
0.0	4.708e-6	1.368e-7	0.3062	0.04928	0.3062	0.04928
0.5	4.816e-6	8.864e-8	0.3200	0.03973	0.3200	0.03973
1.0	6.620e-6	6.752e-8	0.3519	0.03623	0.3519	0.03623
1.5	1.022e-4	4.946e-7	0.4437	0.02670	0.4438	0.02670
2.0	3.869e-4	7.667e-7	0.5529	0.01674	0.5529	0.01674
2.5	9.379e-4	6.171e-7	1.258	0.01362	1.258	0.01362
3.0	1.471e-4	3.183e-8	1.907	0.02560	1.907	0.02560

Table 5.3: Distance measure of gap length distributions for the $(2, 1, 7)$ $171, 133$ convolutional code. (S: Viterbi software decoder, M: Markov chain model, G: Geometric model.)

E_b/N_0,	M & G		M & S		G & S	
dB	TD	MD	TD	MD	TD	MD
0.0	8.689e-8	2.416e-9	0.1425	0.02420	0.1425	0.02420
0.5	5.266e-7	9.657e-9	0.1542	0.01794	0.1542	0.01794
1.0	1.610e-5	1.775e-7	0.1850	0.01487	0.1850	0.01487
1.5	1.305e-5	7.852e-8	0.2773	0.01148	0.2773	0.01148
2.0	1.971e-5	5.756e-8	0.3270	0.005874	0.3270	0.005874
2.5	4.262e-4	5.673e-7	0.4962	0.003859	0.4962	0.003858
3.0	5.100e-4	2.800e-7	1.078	0.001656	1.078	0.001656
3.5	3.024e-3	6.467e-7	1.797	0.01259	1.797	0.01259
4.0	1.570e-3	1.888e-7	1.734	0.2254	1.736	0.2254

Table 5.4: Distance measure of gap length distributions for the $(2, 1, 5)$ $23, 35$ convolutional code. (S: Viterbi software decoder, M: Markov chain model, G: Geometric model.)

model to generate error sequences instead of simulating the Viterbi software decoder. The advantage is that large amounts of data can be generated quickly and inexpensively. For example, only about 2.2 minutes per million bits of computer time on a VAX 11/750 are needed for the Markov chain model of the $(2, 1, 7)$ code, compared to 2.9 hours per million bits required for the Viterbi software decoder.

Consider concatenated Reed-Solomon/convolutional codes on an unquantized AWGN channel. Monte Carlo software routines for the Markov chain model and the geometric model are used to generate Viterbi error sequences. (The required parameters for those two models are taken from simulations of the Viterbi software decoder as in last subsection.) Then concatenated Reed-Solomon word error and bit error probabilities are computed by using outputs from the Viterbi software decoder, the Markov chain model, and the geometric model. This is done for the $(2, 1, 7)$ $177, 133$ and $(2, 1, 5)$ $23, 35$ convolutional codes concatenated with the $(255, 223)$, $(63, 47)$, $(31, 23)$ Reed-Solomon codes with ideal interleaving or no interleaving. The computation of concatenated Reed-Solomon output word error and bit error probabilities is given in Appendix 5.B.

Simulation results are shown in Figures 5.3 to 5.14, where the Reed-Solomon performance curves are plotted versus the concatenated bit signal-to-noise ratios. For all cases considered, both the Markov chain model and the geometric model give good approximations to the actual data from the Viterbi software decoder. In this regard, the geometric model seems advantageous because it is simpler than the Markov chain model.

5.5 Discussion

The required parameters for the Markov chain model (or the geometric model) are obtained from simulations of the Viterbi software decoder. It is useful to have

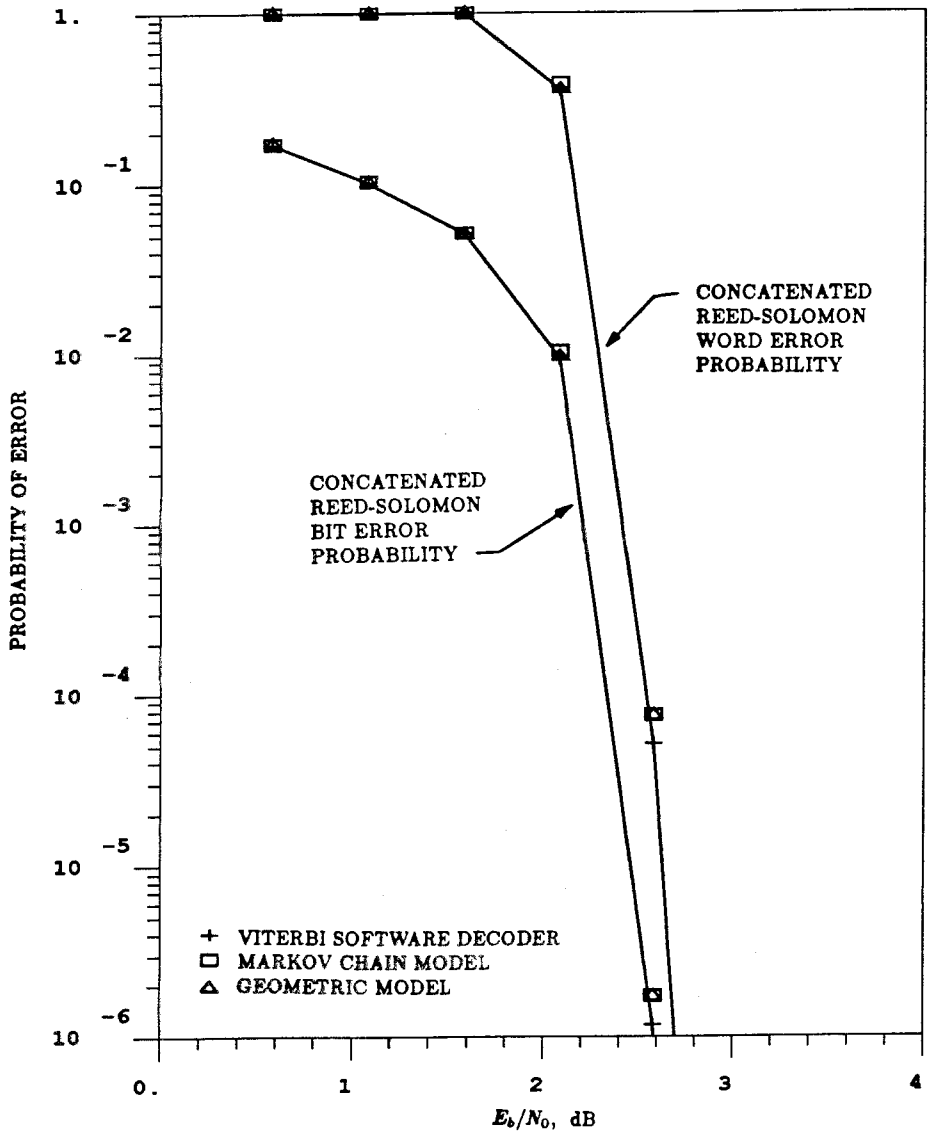

Figure 5.3: Performance statistics for the $(2, 1, 7)$ $171, 133$ convolutional code concatenated with the $(255, 223)$ Reed-Solomon code, ideally interleaved.

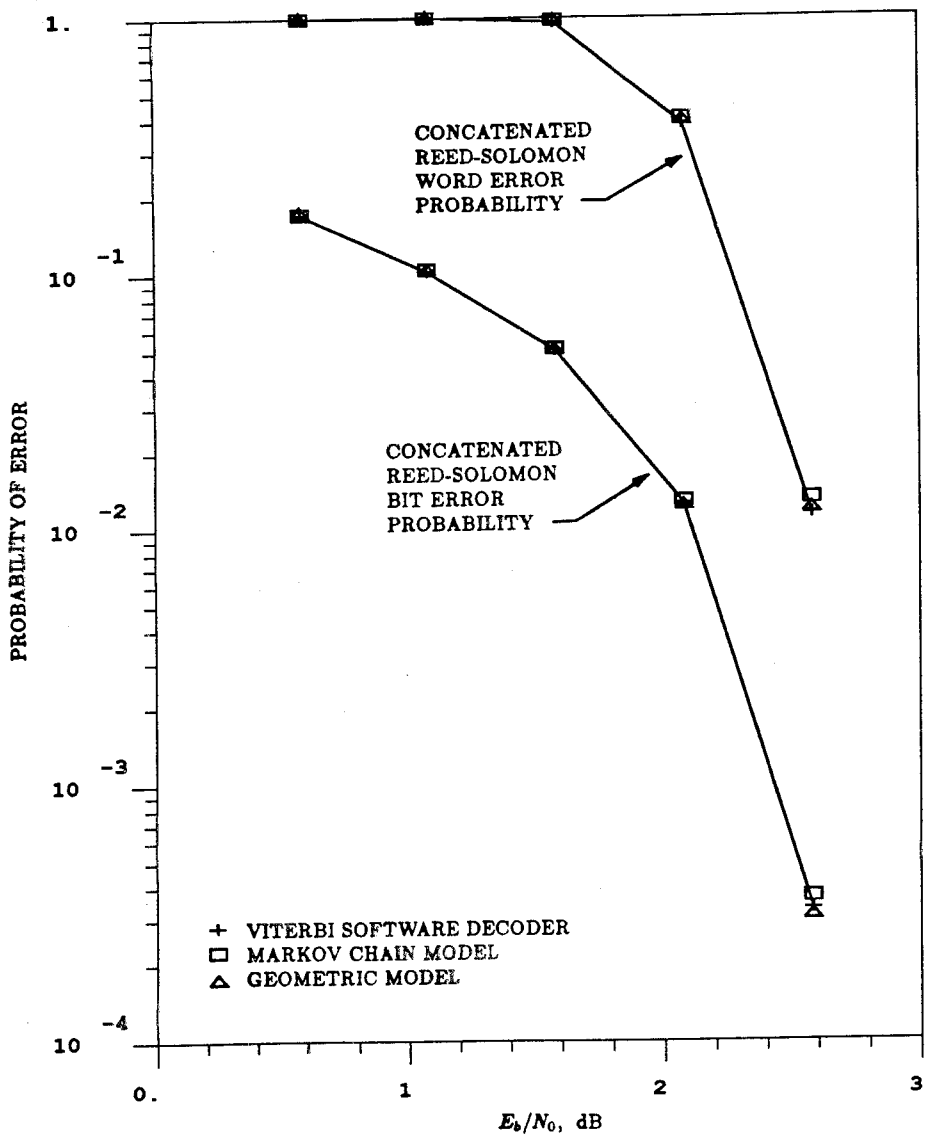

Figure 5.4: Performance statistics for the $(2, 1, 7)$ $171, 133$ convolutional code concatenated with the $(255, 223)$ Reed-Solomon code, noninterleaved.

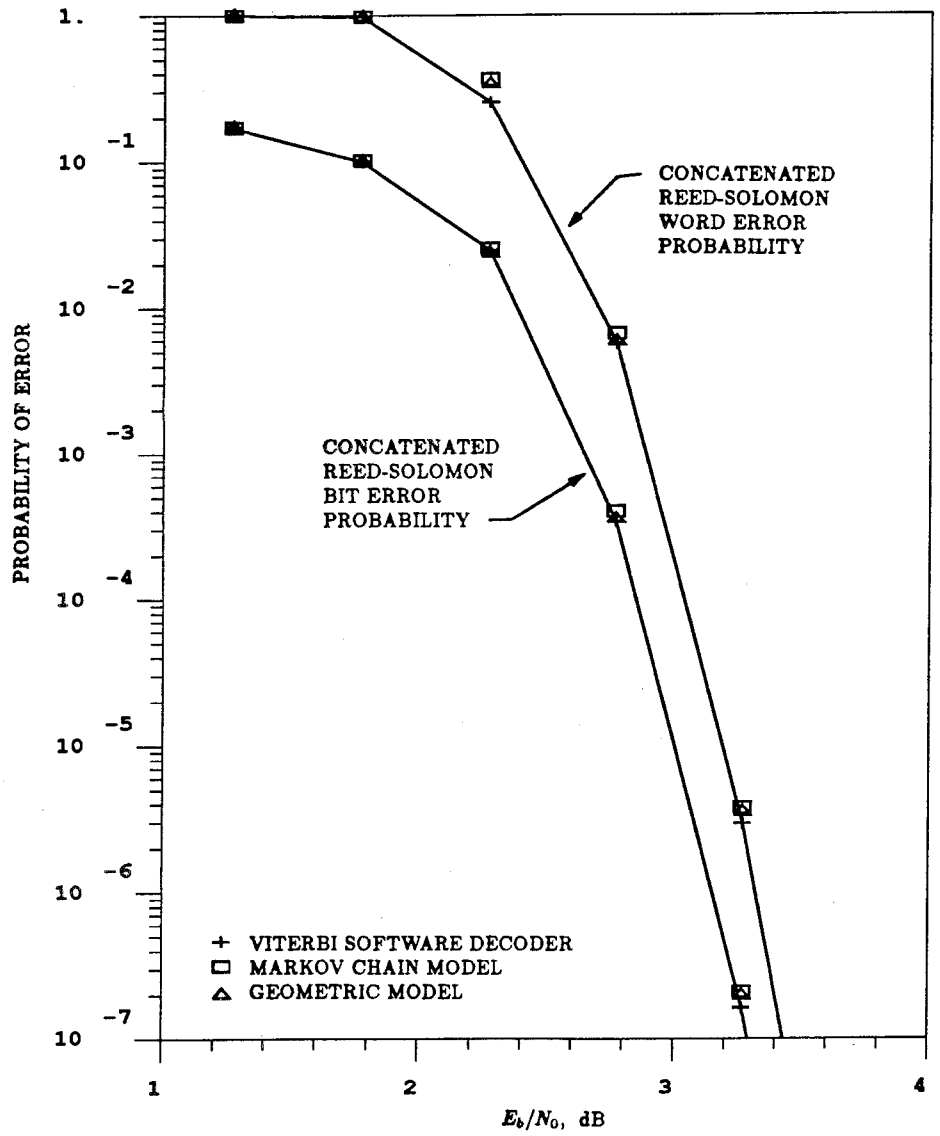

Figure 5.5: Performance statistics for the $(2, 1, 7)$ $171, 133$ convolutional code concatenated with the $(63, 47)$ Reed-Solomon code, ideally interleaved.

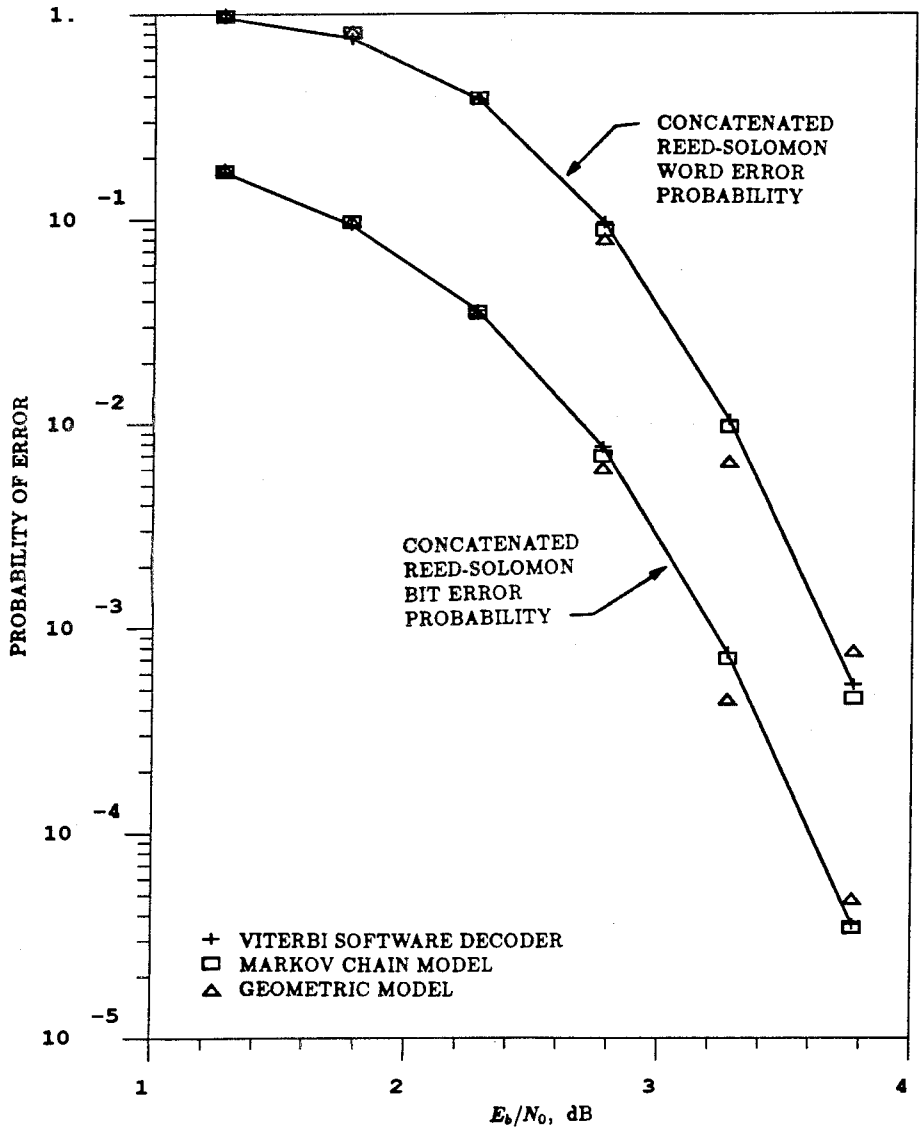

Figure 5.6: Performance statistics for the $(2, 1, 7)$ $171, 133$ convolutional code concatenated with the $(63, 47)$ Reed-Solomon code, noninterleaved.

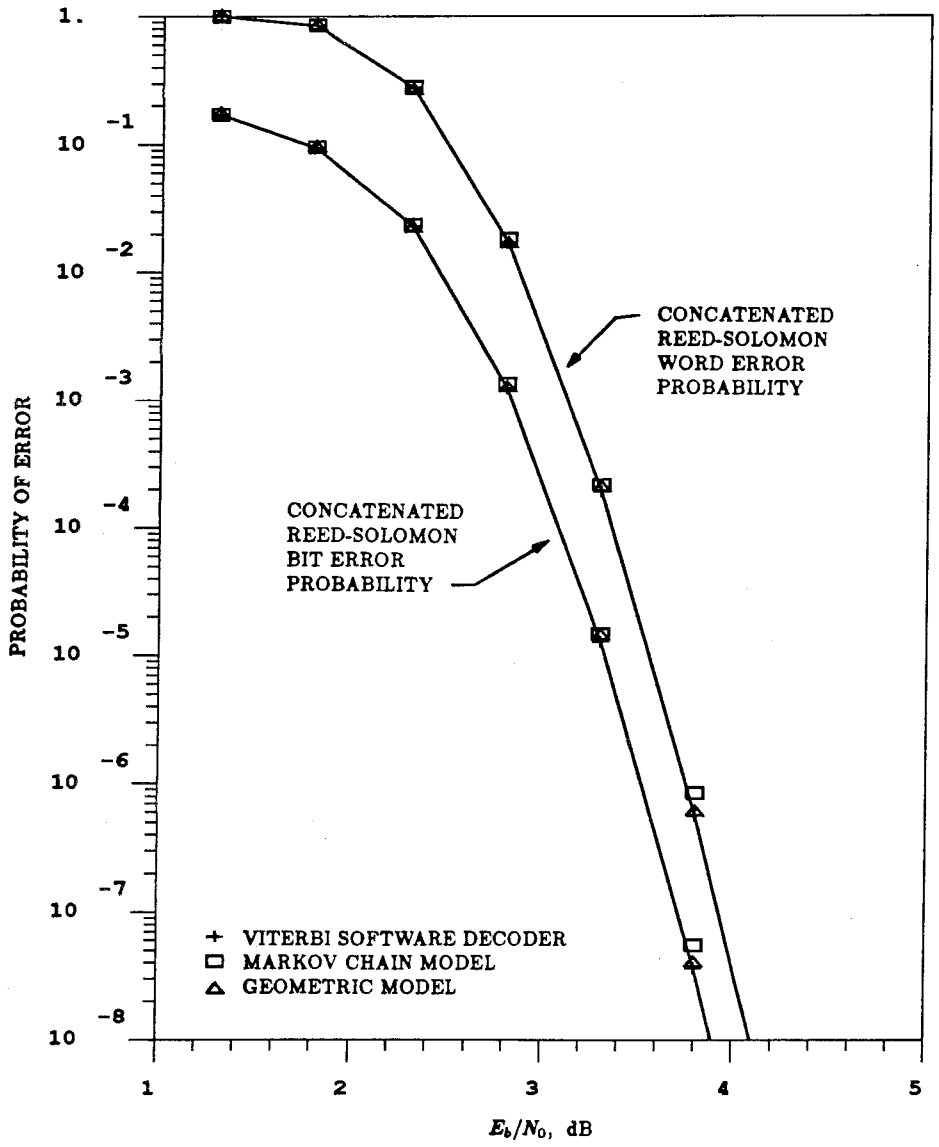

Figure 5.7: Performance statistics for the $(2, 1, 7)$ $171, 133$ convolutional code concatenated with the $(31, 23)$ Reed-Solomon code, ideally interleaved.

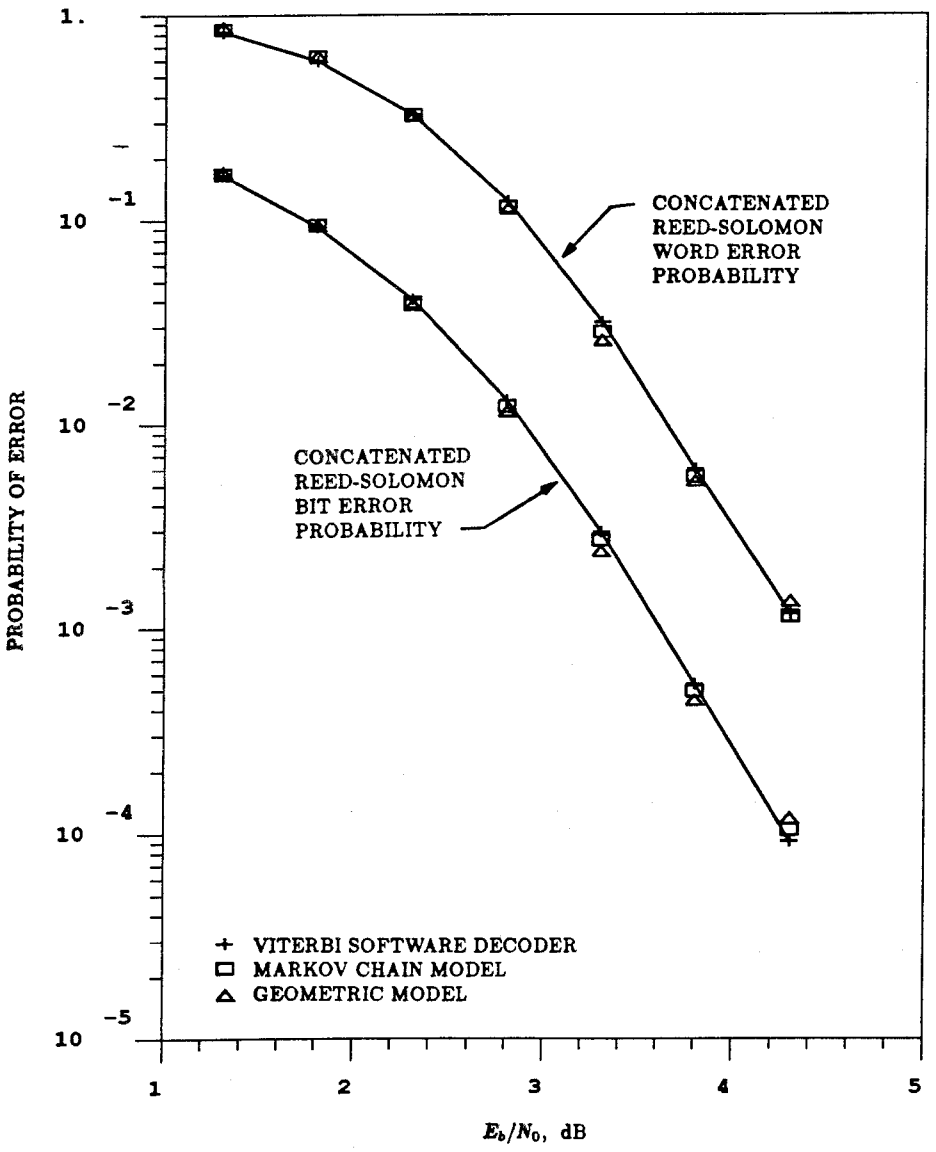

Figure 5.8: Performance statistics for the $(2, 1, 7)$ $171, 133$ convolutional code concatenated with the $(31, 23)$ Reed-Solomon code, noninterleaved.

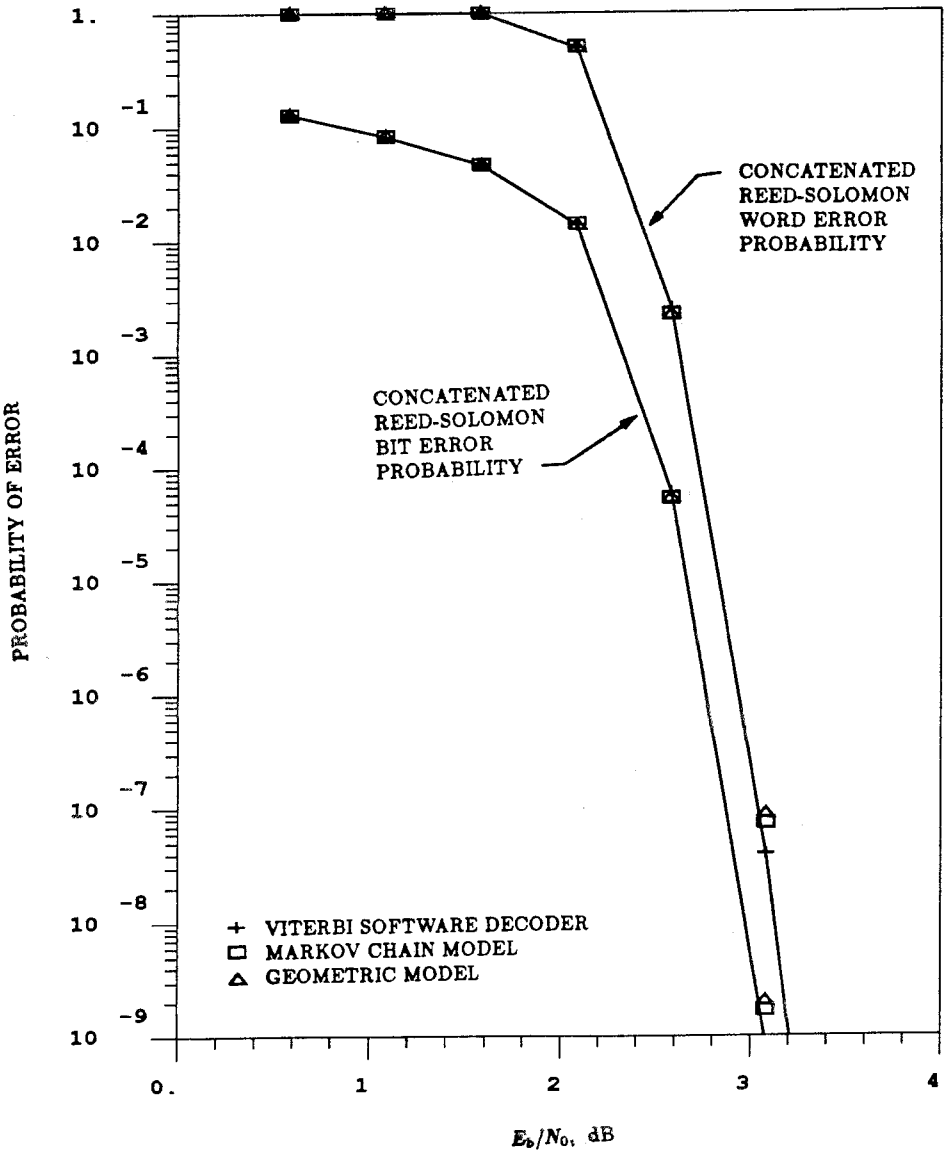

Figure 5.9: Performance statistics for the $(2, 1, 5)$ $23, 35$ convolutional code concatenated with the $(255, 223)$ Reed-Solomon code, ideally interleaved.

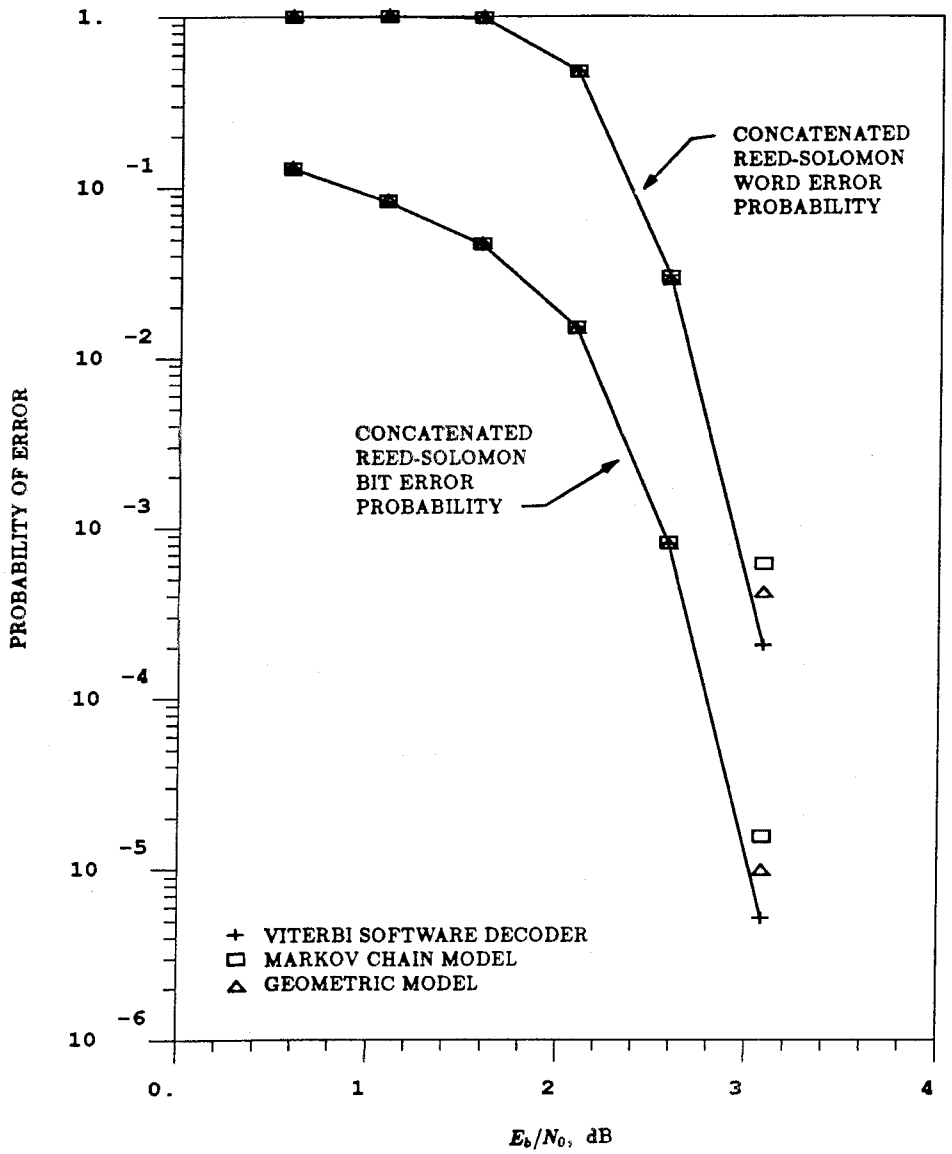

Figure 5.10: Performance statistics for the $(2, 1, 5)$ $23, 35$ convolutional code concatenated with the $(255, 223)$ Reed-Solomon code, noninterleaved.

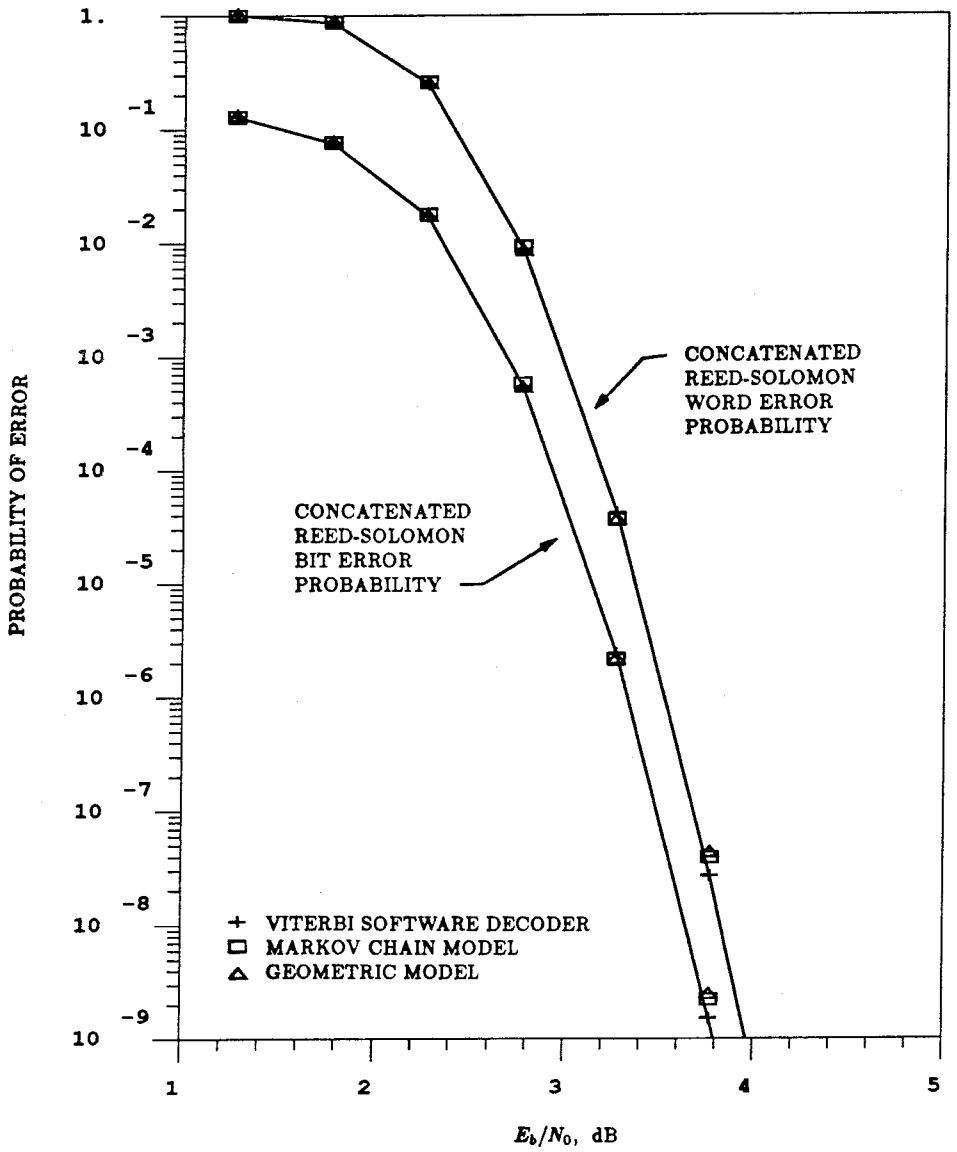

Figure 5.11: Performance statistics for the $(2, 1, 5)$ $23, 35$ convolutional code concatenated with the $(63, 47)$ Reed-Solomon code, ideally interleaved.

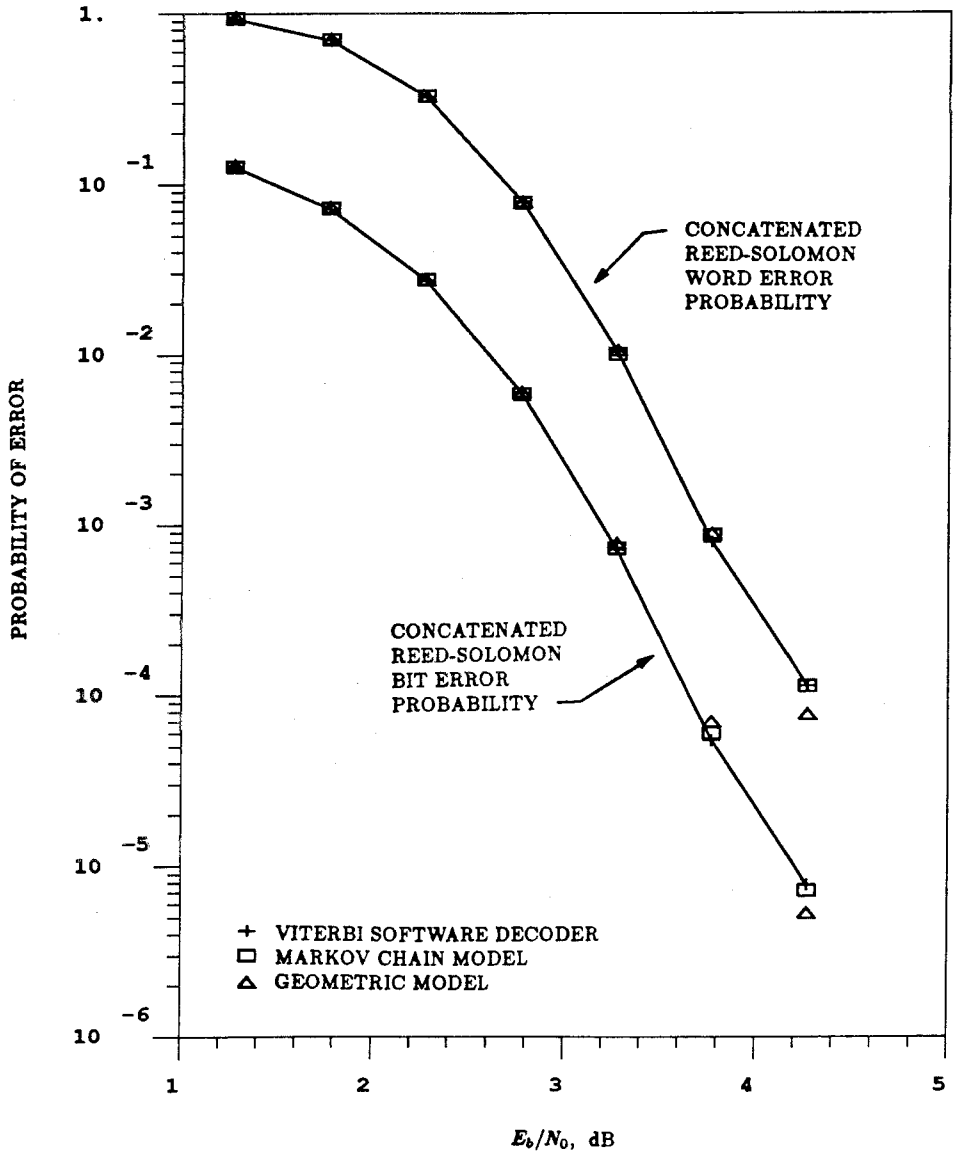

Figure 5.12: Performance statistics for the $(2, 1, 5)$ 23, 35 convolutional code concatenated with the $(63, 47)$ Reed-Solomon code, noninterleaved.

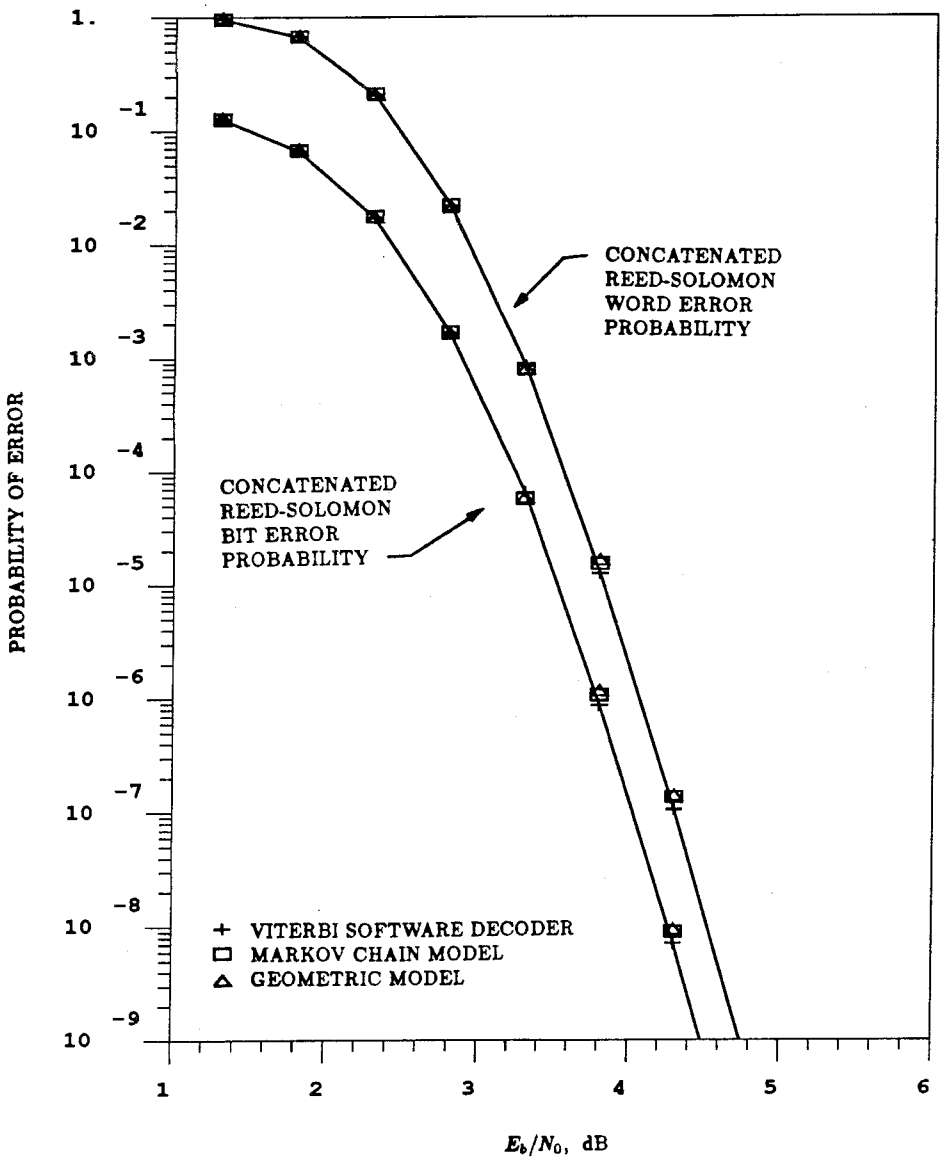

Figure 5.13: Performance statistics for the $(2, 1, 5)$ $23, 35$ convolutional code concatenated with the $(31, 23)$ Reed-Solomon code, ideally interleaved.

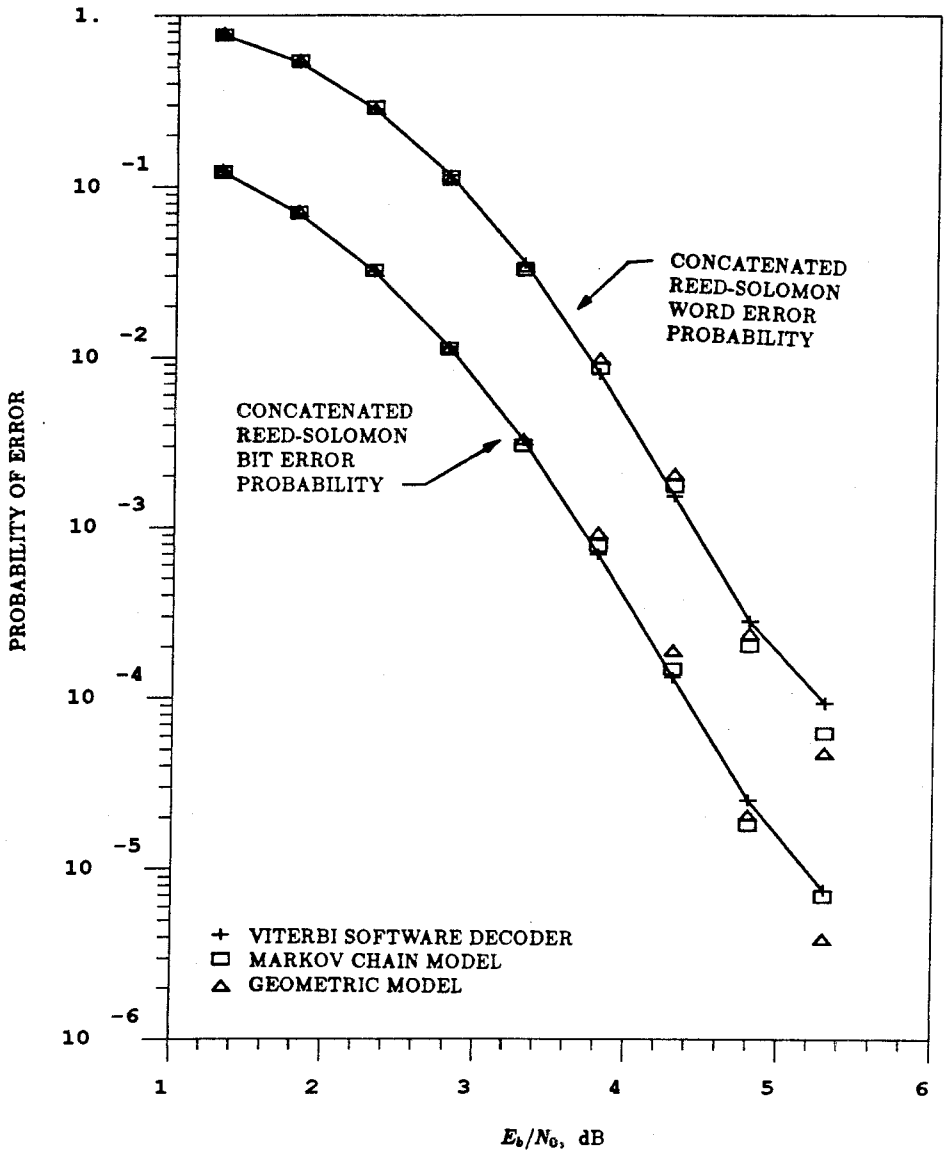

Figure 5.14: Performance statistics for the $(2, 1, 5)$ $23, 35$ convolutional code concatenated with the $(31, 23)$ Reed-Solomon code, noninterleaved.

approximate values for the transition probabilities for the Markov chain model analytically. Actually, the transition probability p_{01} can be upperbounded the same way as the first-event error probability. However, nothing similar is known for the other transition probabilities.

Appendix 5.A Geometric Model

A geometric model was proposed in [2] to model the Viterbi decoder burst error statistics. There are three parameters needed for the model: the average burst length $\overline{B}$, the average guardspace length $\overline{G}$, and the burst error density θ. The burst length is modelled as distributed geometrically according to

$$\mathrm{P}(B = b) = \begin{cases} q(1-q)^{b-1}, & \text{for } b \geq 1, \\ 0, & \text{otherwise,} \end{cases}$$

where $q = 1/\overline{B}$. Errors within bursts occur randomly with probability θ (except that each burst begins and ends with an error). The guardspace length distribution is modelled as

$$\mathrm{P}(G = g) = \begin{cases} r(1-r)^{g-m}, & \text{for } g \geq m, \\ 0, & \text{otherwise,} \end{cases}$$

where $r = 1/(\overline{G} - m + 1)$.

Appendix 5.B Computation of Reed-Solomon Error Probabilities

Consider an (n, k) Reed-Solomon code with symbols from $GF(2^b)$ that corrects $t = (n-k)/2$ symbol errors. Suppose this code is used as an outer code in a concatenated coding system with a convolutional inner code. The Reed-Solomon input symbol error probability V_s is found by partitioning the Viterbi output bit sequences into disjoint b-bit sets and counting how many of the sets contain bit errors. A word error occurs when there are more than t (out of possible n) symbols

in error for a Reed-Solomon codeword. For the case of ideal interleaving, i.e., the symbols are interleaved at a sufficient depth so that symbol errors are independent at the Reed-Solomon decoder input, the word error probability is

$$P_w = \sum_{i=t+1}^{n} \binom{n}{i} V_s^i (1 - V_s)^{n-i}.$$

When more than t symbol errors occur, the decoder either detects the presence of more than t symbol errors but is unable to correct them, or miscorrects the received pattern into a codeword other than the transmitted codeword. Since the probability of decoder miscorrecting is very low [7] [8], assume the decoder can always detect the presence of more than t symbol errors. The Reed-Solomon output symbol error probability is then approximated by

$$P_s \approx \sum_{i=t+1}^{n} \frac{i}{n} \binom{n}{i} V_s^i (1 - V_s)^{n-i}.$$

If the bit error probability at the output of the Viterbi decoder is denoted by V_b, then the Reed-Solomon output bit error probability is approximately

$$P_b \approx \frac{V_b}{V_s} P_s.$$

For the case of no interleaving, the Reed-Solomon output word error probability is calculated by partitioning the b-bit sets at the input of the Reed-Solomon decoder into n-set blocks and counting how many of the blocks contain more than t sets in error. The bit error probability can be found by examining how many bit errors there are for those blocks with more than t sets in error.

References

[1] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding.* New York: McGraw-Hill, 1979.

[2] R. L. Miller, L. J. Deutsch, and S. A. Butman, *On the Error Statistics of Viterbi Decoding and the Performance of Concatenated Codes.* JPL Publication 81-9, Jet Propulsion Laboratory, Pasadena, CA, Sep. 1981.

[3] M. R. Best, "A Markov chain model for a convolutional coding scheme," *Proc. Sixth Symp. Inform. Theory*, Benelux, Mierlo, The Netherlands, May 1985.

[4] M. R. Best, "The exact analysis of a convolutional coding scheme," to appear in *IEEE Trans. Inform. Theory.*

[5] W. Feller, *An Introduction to Probability Theory and its Applications*, vol. 1, 3rd ed. New York: Wiley, 1968.

[6] W. Feller, *An Introduction to Probability Theory and its Applications*, vol. 2, 2nd ed. New York: Wiley, 1971.

[7] R J. McEliece and L. Swanson, "On the decoder error probability for Reed-Solomon Codes," *IEEE Trans. Inform. Theory*, vol. IT-32, pp. 701-703, Sep. 1986.

[8] K. M. Cheung, "Error-correction coding in data storage systems," Ph.D. dissertation, California Institute of Technology, Pasadena, 1987.