Analytical Methods for the Performance Evaluation of Binary Linear Block Codes

by

Pragat Chaudhari

A thesis

presented to the University of Waterloo

in fulfilment of the

thesis requirement for the degree of

Master of Applied Science

in

Electrical & Computer Engineering

Waterloo, Ontario, Canada, 2000

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The modeling of the soft-output decoding of a binary linear block code using a Binary Phase Shift Keying (BPSK) modulation system (with reduced noise power) is the main focus of this work. With this model, it is possible to provide bit error performance approximations to help in the evaluation of the performance of binary linear block codes. As well, the model can be used in the design of communications systems which require knowledge of the characteristics of the channel, such as combined source-channel coding. Assuming an Additive White Gaussian Noise channel model, soft-output Log Likelihood Ratio (LLR) values are modeled to be Gaussian distributed. The bit error performance for a binary linear code over an AWGN channel can then be approximated using the Q-function that is used for BPSK systems. Simulation results are presented which show that the actual bit error performance of the code is very well approximated by the LLR approximation, especially for low signal-to-noise ratios (SNR). A new measure of the coding gain achievable through the use of a code is introduced by comparing the LLR variance to that of an equivalently scaled BPSK system. Furthermore, arguments are presented which show that the approximation requires fewer samples than conventional simulation methods to obtain the same confidence in the bit error probability value. This translates into fewer computations and therefore, less time is needed to obtain performance results.

Other work was completed that uses a discrete Fourier Transform technique to calculate the weight distribution of a linear code. The weight distribution of a code is defined by the number of codewords which have a certain number of ones in the codewords. For codeword lengths of small to moderate size, this method is faster and provides an easily implementable and methodical approach over other methods. This technique has the added advantage over other techniques of being able to methodically calculate the number of codewords of a particular Hamming weight instead of calculating the entire weight distribution of the code.

Acknowledgements

I would like to acknowledge and thank my supervisor Dr. A. K. Khandani for his guidance through all phases of my research. During my years here in the graduate program, his undying support and enthusiasm for the work were always present, even when things didn't always work out. Thank you for sharing your insight into research problems. And most of all, thank you for being a friend.

This research was supported by funds from Communications and Information Technology Ontario (CITO), then known as the Information Technology Research Centre, the University of Waterloo, and the Natural Sciences and Engineering Research Council of Canada (NSERC). Thank you for your financial support of the research.

I would also like to acknowledge Prof. A. Hasan and Prof. S. Safavi-Naeini for taking the time to carefully read my thesis and for their helpful feedback and suggestions.

My sincere gratitude also goes to all the supporting staff of the Electrical and Computer Engineering Department here at the University of Waterloo. You have all been patient, helpful and resourceful, and personable.

Many warm thanks to Shahram Yousefi, and Ghodrat Esmaeili for "living" with me in the office over the months and for the numerous insightful discussions we've had.

Finally, I would like to thank my family and close friends here in Waterloo and in Toronto for their continuous and relentless love and support of me and my work. I couldn't have accomplished this without you all behind me.

Contents

List of Tables

List of Figures

Chapter 1

Introduction and Background

In today's increasingly connected world, people are communicating more frequently, and transmitting trillions of bits of information to one another, whether as voice, video, or data. Transmitting and receiving this information reliably and efficiently is very important to the users. Without reliable communications techniques, the data transmitted may be corrupted by noise, and the value of the information may be lost. Efficient communication techniques also tend to lower the cost of communicating for everyone. These two quality of service requirements are met through the advent of new digital communications algorithms and methodology and through their eventual implementation.

The communications engineer aims to provide techniques and algorithms such that reliable communications can be realized. To this end, to gauge the reliability of a communications system, the bit error probability performance of a channel code is often used. The channel code is specifically designed to introduce known redundancy into the information bit stream so that the corrupting noise does not make the transmitted codeword unrecognizable at the receiver. However, errors in the decoding of a codeword can still occur. Bit error performance curves depicting the bit error probability for a given signal-to-noise power ratio (SNR) are useful in determining the reliability of a code.

Communications systems are complex, being comprised of many interacting components. To better the performance of a system, knowledge of channel characteristics can be used to better design the system components. Figure 1.1 depicts a communications system, where the question-marked boxes are the components in the system that should be designed with some prior knowledge of the characteristics of the channel. It would be desirable to be able to replace the complex channel encoder, channel and channel decoding system by a simpler model so that design engineers can concentrate on these question-marked components of the system. An example of this would be combined source-channel coding schemes where the source coder quantizes the incoming bit stream based upon the characteristics of the channel.

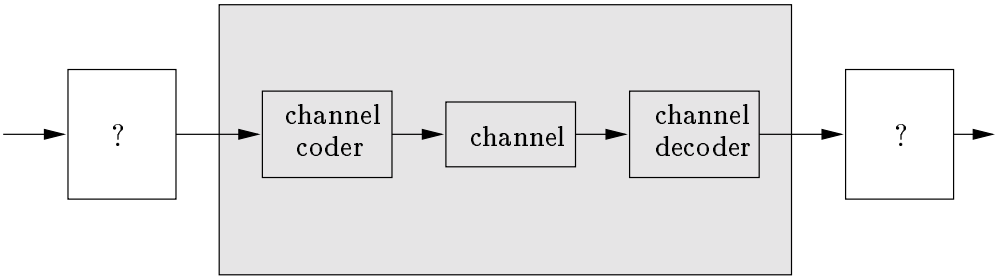

Figure 1.1: Replacement of the Channel Model with Simpler Model

This replacement is addressed in this thesis and is the motivation for this work. This work is completed by investigating soft-output decoding techniques and by studying the probabilistic behaviour of the output values.

To assist the communications engineer with the design of coding systems, mathematical bounds exist that provide a rough estimate of the performance of a code without having to actually simulate the code on computers. In many cases, the structure of the code is required to customize the bounds. A common property that is used is weight dis-

tribution of a code. The weight distribution of a code is a table of values of the number of codewords of the code which have a given number of ones in them. The number of low weight codewords is obtainable from the by the weight distribution of the code and these codewords are more likely to be confused with other codewords in the presence of noise, making their contribution to the error performance of a code large. A bound that is often used in digital communications is the Union Bound. This bound takes into account the sum effect of all possible errors which can occur for a code. Using the weight distribution of a code, the Union bound can be easily calculated [1, 2].

Methods to obtain the weight distribution of a code currently involve traversing the trellis of the code and accumulating the weight of the paths through the trellis by multiplying and accumulating polynomials representing the path weights. These methods prove to be tedious and complicated. An easier mathematical approach to this problem is desirable. One such approach is also presented in this thesis, based upon the raising a modified state transition matrix of the code to a power equal to its length, and performing an inverse Discrete Fourier Transform.

Before the work is presented, some background information on soft-output decoding techniques is presented as a foundation for further discussions. As well, the channel model used for this thesis is described below.

1.1 Assumed Channel Model

Suppose codeword $\boldsymbol{m}$ is to be transmitted over a channel as presented in figure 1.2. The codeword is a vector of n bits of value 0 or 1. The vector nature of the codeword is represented by the bold font of the vector name and will be consistently applied throughout this thesis. The original codeword $\boldsymbol{m}$ is then modulated using a Binary Phase Shift Keying modulation scheme. Essentially, this means that the zeros have been mapped to

-1's, with the ones unchanged, producing the modulated codeword $\boldsymbol{s}$. More information on BPSK modulation is provided in section 2.1. The modulated codeword is then transmitted over the channel and is disrupted by channel noise $\boldsymbol{\eta}$ [1,3].

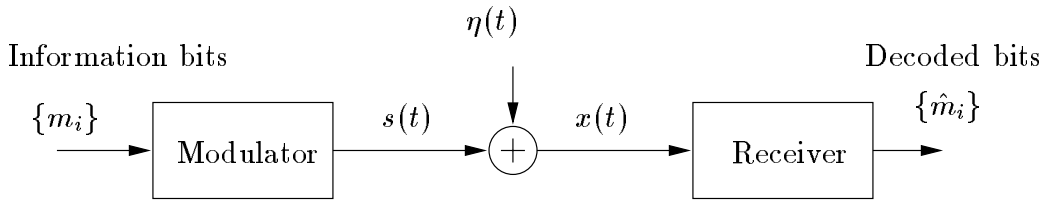

Figure 1.2: AWGN Channel Model

The noise vector $\boldsymbol{\eta}$ is comprised of independent and identically distributed (i.i.d.) Gaussian random samples with mean, 0, and variance, $\frac{N_0}{2}$. These facts characterize the channel as an Additive White Gaussian Noise (AWGN) channel. The codeword $\boldsymbol{x}$ is received at the receiver and is then decoded to obtain the decoded codeword $\hat{\boldsymbol{m}}$.

The probability density function of a Gaussian random variable becomes important in discussions of the probability of bit error for the received codeword since the AWGN channel is assumed. This probability density function is given by,

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma_x^2}} \exp\{-\frac{(x-\mu_x)^2}{2\sigma_x^2}\}. \tag{1.1}$$

In particular, for the noise, η, the probability density function is modified with a mean of 0 and a variance of $\sigma_\eta^2 = \frac{N_0}{2}$ to yield,

$$f_\eta(\eta) = \frac{1}{\sqrt{2\pi\sigma_\eta^2}} \exp\{-\frac{\eta^2}{2\sigma_\eta^2}\}. \tag{1.2}$$

The codewords are n bits in length, and the noise samples are independent and identically distributed with the same mean and variance. The multivariate Gaussian probability

density function is given by,

$$\begin{aligned} f_\eta(\boldsymbol{\eta}) &= \frac{1}{(2\pi)^{n/2}(\sigma_\eta^n)} \exp\left\{ -\frac{\sum_{i=0}^{n} \eta_i^2}{2\sigma_\eta^2} \right\} \\ &= \frac{1}{(2\pi)^{n/2}(\sigma_\eta^n)} \exp\left\{ -\frac{||\boldsymbol{\eta}||^2}{2\sigma_\eta^2} \right\} \\ &= \frac{1}{(2\pi)^{n/2}(\sigma_\eta^n)} \exp\left\{ -\frac{||\boldsymbol{x} - \boldsymbol{s}||^2}{2\sigma_\eta^2} \right\}, \end{aligned} \tag{1.3}$$

where the relationship $\boldsymbol{\eta} = \boldsymbol{x} - \boldsymbol{s}$, in vector form, has been used from figure 1.2.

There are many ways in which the received codeword may be decoded. A brief synopsis of decoding practices is provided in the next section to help tie the ideas of simulations and bit error performance to the approximation that is the main contribution of this thesis.

1.2 Decoding and the Use of the Log Likelihood Ratio

Three main classifications of decoding techniques exist for the decoding of a received codeword: hard decision decoding; soft-decision decoding; and soft-output decoding. Each of these will be briefly described for an understanding of decoding practices.

Hard decision decoding involves the immediate quantization of each component of the received codeword using a threshold value of 0. A value of 0 is assigned for a negative received component and a 1 for a positive component. This method, although simple in implementation, does not produce the best possible results.

Soft-decision decoding can be used where the actual decisions about the received bits of the codewords are not made until some further processing is carried out. Noting the structure of the Gaussian distribution, to minimize the block error probability, the further processing includes the search for a BPSK-modulated codeword which minimizes

the squared Euclidean distance between two codewords. The squared Euclidean distance between the received codeword $\boldsymbol{x}$ and the i^{th} modulated codeword of the code, $\mathcal{C}$, is given by

$$\begin{aligned} D &= \sum_{j=1}^{n} (x_j - c_{ij})^2 \\ &= ||\boldsymbol{x} - \boldsymbol{c}_i||^2. \end{aligned} \tag{1.4}$$

The smaller the squared Euclidean distance between two vectors, the more likely that the codewords are the same and therefore, the more likely the received codeword will be decoded correctly. Note that the term in the exponent of the probability density function of (1.3) contains the squared Euclidean distance measure explicitly.

Soft-decision techniques make a hard decision at the end of processing to determine the bit value, and these methods are known to minimize the probability of block error rather than the probability of bit error [1]. The Viterbi algorithm is an example of such a soft-decision decoding technique. This algorithm is applicable to any code which is representable by a trellis. Since linear block codes are representable by a trellis [4], the algorithm can be applied. The concatenated bits values of various joined trellis branches constitute a path through the trellis. This method is a maximum likelihood approach and calculates a metric along the paths and chooses the path with the minimum total metric at the end of the trellis [1]. Therefore, this minimizes the probability of codeword error, but not necessarily the bit probability error [4]. The soft-output Viterbi algorithm was later developed to not only provide the maximum likely path, but also an indication of the confidence in this path [5].

Soft-output decoding techniques produce the probability of a given transmitted bit being a certain value (0 or 1) and can be used as a level of confidence in the value of the

bit. These techniques differ from soft-decision techniques in two key ways:

1. these techniques minimize the bit error probability of a code, rather than minimizing the block error probability; and

2. the soft values obtained (probabilities) can be used in iterative decoding techniques, for example, as used with turbo codes.

Bahl *et al.* derived an optimal soft-output decoding method (BCJR) for codes representable by a trellis. The BCJR algorithm, which is also known as the "Forward-Backward" algorithm, traverses a trellis and attempts to calculate the *a posteriori probabilities* (APP) of the states and transitions in the trellis. That is, given the received codewords, what is the probability that the bit m_k of the original codeword was a value of i (i.e. $\Pr\{m_k = i|\boldsymbol{x}\}$). This is done by traversing the trellis in the forward direction and calculating transition probabilities based on the vector $\boldsymbol{x}$ received thus far. The multivariate probability density function of (1.3) is used for the transition probabilities. After the entire vector has been received (all n symbols or bits), the probabilities are updated backwards through the trellis to the beginning, where the APPs are finally calculated. With the APPs, the bits can be decoded by noting that the probability of a bit being either a 0 or 1 is greater than 0.5.

In 1993, Berrou *et al* [6] used the idea of the log likelihood ratio (LLR) to shift the decision threshold to 0, with the sign of the LLR determining the bit value. The LLR for the k^{th} bit of the original codeword $\boldsymbol{m}$ was defined as,

$$LLR,\ \Delta(m_k) = \log \frac{\Pr\{m_k = 1|\boldsymbol{x}\}}{\Pr\{m_k = 0|\boldsymbol{x}\}}. \tag{1.5}$$

Note that $\Pr\{m_k = i|\boldsymbol{x}\}$, $i = 0, 1$, is the APP of the original bit m_k. $\log(\cdot)$ will be assumed to be defined as $\log_e(\cdot)$, for this thesis unless otherwise stated.

If the LLR is positive, then the transmitted bit was most probably a 1 since the numerator probability is greater than the denominator; otherwise, it was most probably a 0. The magnitude of the LLR would indicate the confidence associated with making such a decision

It is through the use of the LLR that the new approximation is obtained. By approximating the LLR to have a Gaussian distribution, and using its mean and standard deviation, the soft-output decoding probability of bit error for the linear code can be obtained. This is the focus of this thesis. The idea is presented, expanded upon, and analyzed thoroughly in the following chapters.

1.3 Conventional Simulation Methods

Before continuing, current conventional simulation techniques must be mentioned. Today, when a code is to be computationally simulated, the decoding algorithm processes the noise-corrupted transmitted codewords, and produces bits it believes to be the original bits. Using soft-output decoding techniques, this requires that a decision be made on the LLR value of a particular bit. Any errors are detectable since what was transmitted is known in the simulations. The bit error probability for a given SNR is calculated as being the total number of errors detected divided by the total number of bits transmitted. Throughout this thesis, this simulation method will referred to as the conventional simulation method.

1.4 Thesis Outline

The outline of the thesis is as follows. Chapter 2 presents the soft-output technique used to obtain a value for the LLR of a bit in the transmitted codeword. The BCJR algorithm could have been used just as well. The chapter also reviews Binary Phase Shift Keying

modulation, with mention of the expression for bit error probability. The expression involves the use of the Q-function, which is well defined in literature [1,3]. Next, the LLR is expanded in powers of Gaussian variables, using a Taylor series expansion. The Gaussian approximation is introduced and the modeling of the soft-output decoding of binary linear codes using a BPSK modulated system is presented. A new definition for coding gain is also stated and described.

With the approximation presented, the following chapters present an analysis of the approximation based on the use of the mean and variance of the LLR values obtained via simulation. The eventual goal is to compare the variance in the results of the conventional simulation methods to the variance in results obtained from using the approximation, to discuss the relative number of samples required for the each approach.

Chapter 3 presents the independence of the two estimators used for the mean and variance of the LLR values and also presents the probability density functions of the mean and standard deviation. With the determination of these probability density function, the probability density function of the ratio, denoted Z, can be found[1].

Chapter 4 presents an expression for the r^{th} moments of the random variable Z which is then used to obtain variance of the Q-function using its Taylor series expansion and argument Z. An expression for the variance is provided.

In chapter 5, the comparison of the conventional simulation method to the Gaussian approximation is presented, based upon the value of variances for a given bit error probability and a given number of samples. It is here that the merit of the approximation is illustrated.

Simulations results are presented in chapter 6 and discrepancies between the approximation and the actual bit error curve obtained through conventional simulation methods

[1]Formulation of the mean and variance of the estimators is presented in Appendix A. Appendix B provides the formulation of the probability density function of the ratio Z.

are addressed.

Another contribution of this thesis is presented in chapter 7. This chapter presents the use of the discrete Fourier transform on a weighted state transition matrix to obtain the weight distribution of a code. The methodology is thoroughly presented, with examples.

Finally, the thesis is concluded with conclusions of the two major contributions of the thesis in chapter 8. Possible future research directions for the two contributions are also discussed in this final chapter.

The contributions made within this thesis are based upon well-known principles, however, the Gaussian approximation of the LLR, and its application to the modeling of the soft-output decoding of binary linear codes is novel.

Chapter 2

The Gaussian Approximation for the Log Likelihood Ratio

The Log Likelihood Ratio (LLR) was discussed as a method of determining the value of a transmitted bit of information and providing a measure of the confidence in that value. The confidence is measured through the size of the absolute value of the LLR and its sign determines the value of the bit: positive for a 1 and negative for a 0. Therefore, a decision on the originally transmitted bit can be easily made since the decision threshold is simply 0.

The LLR is used extensively in decoding. The value for a particular bit position of the received codeword can be found using the expression (1.5) from the previous chapter. It will be shown that by using the codewords of the code $\mathcal{C}$, these probabilities can be approximated and the value of the bit can be decoded.

Consider the bit in position k of the codewords of code $\mathcal{C}$. Depending on the value of these bits, the codewords can be divided into two subsets. All the codewords in one subset, $\mathcal{C}_0$, contain a 0 in that position and the other subset, $\mathcal{C}_1$, contains those codewords

with a 1 in that same position. Subset $\mathcal{C}_0$ is a sub-code of code $\mathcal{C}$. Subset $\mathcal{C}_1$ is a coset of $\mathcal{C}_0$, where a coset is obtained by adding a constant vector to every codeword of the sub-code.

Using a probability measure involving the received vector and the codewords of either subset, the probability of the sent bit m_k equaling either 0 or 1 can be emulated by summing these measures for each subset. The summation of the measures is justified since the codewords are distinct from one another, in that only one codeword is transmitted and received at any one time [7].

Since an AWGN channel model is considered in this thesis, the probabilities are those of Gaussian random variables. The multivariate probability density function for a vector of n noise samples is given in (1.3). Based upon this probability density function, one can define a probability measure incorporating the received codeword and a codeword of $\mathcal{C}$. Observing the exponential term of the distribution in (1.3), the two codewords are seen to be related through their squared Euclidean distance, as defined in equation (1.4). The squared Euclidean distance is used as a metric for the calculation of the pseudo-probabilities. The term pseudo is used since the measure are not proper probabilities and require normalization.

The pseudo-probabilities are calculated by exhaustively calculating the squared Euclidean distance of the received codeword to all of the ± 1 modulated codewords in the codebook and then dividing these quantities into two subsets based upon the bit value of position k. Those pseudo-probabilities which are computed using a codeword that contains a 1 in the given bit position are summed together to produce a quantity, A. For those codewords which have a 0 in that same position, the pseudo-probabilities are summed to form a quantity, B. By dividing A by B, and taking the log of the result, the

LLR for that bit position is formed. Mathematically, this procedure is,

$$\begin{aligned} LLR(m_k) &= \log \frac{\Pr(m_k = 1|\boldsymbol{x})}{\Pr(m_k = 0|\boldsymbol{x})} \\ &= \log \frac{\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\left\{-\frac{||\boldsymbol{x}-\boldsymbol{c}_i||^2}{2\sigma_\eta^2}\right\}}{\sum_{\boldsymbol{c}_j \in \mathcal{C}_0} \exp\left\{-\frac{||\boldsymbol{x}-\boldsymbol{c}_j||^2}{2\sigma_\eta^2}\right\}} \\ &= \log \frac{\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\left\{-\frac{||\boldsymbol{x}||^2-2\boldsymbol{x}\cdot\boldsymbol{c}_i+||\boldsymbol{c}_i||^2}{2\sigma_\eta^2}\right\}}{\sum_{\boldsymbol{c}_j \in \mathcal{C}_0} \exp\left\{-\frac{||\boldsymbol{x}||^2-2\boldsymbol{x}\cdot\boldsymbol{c}_j+||\boldsymbol{c}_j||^2}{2\sigma_\eta^2}\right\}} \\ &= \log \frac{\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\left\{\frac{\boldsymbol{x}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\right\}}{\sum_{\boldsymbol{c}_j \in \mathcal{C}_0} \exp\left\{\frac{\boldsymbol{x}\cdot\boldsymbol{c}_j}{\sigma_\eta^2}\right\}} \\ &= \log \frac{A}{B} \end{aligned} \tag{2.1}$$

Note that $||\boldsymbol{c}_q||^2 = \sum_{k=1}^{n} c_{qk}^2 = n$, since the codeword bits were modulated ± 1 for $q = i, j$, and where n is the length of the codeword. The constant terms can be cancelled from the numerator and denominator of the LLR expression.

With the LLR value for a bit, the bit error probability for that bit position can be found through simulation of a number of transmitted codewords. The focus of this chapter is to present a new modeling approximation method using the LLR and to discuss when such an approximation is valid. The statistical nature of the LLR is investigated to develop the model.

The remainder of the chapter is organized as follows. Firstly, Binary Phase Shift Keying (BPSK) modulation is reviewed and the calculation of the bit error probability of a BPSK system is discussed. With characteristics of a BPSK system established, the approximation of the LLR as being Gaussian distributed is presented in section 2.3. This approximation is based upon the Taylor series expansion of the LLR and is thoroughly described. The approximation can then be used to obtain the bit error performance of a

binary linear code. Finally, this chapter is concluded with a definition of new measure of coding gain, wherein the variance of the LLR approximation is compared to that of an equivalently scaled BPSK system. Later chapters expand upon the LLR approximation and present the mathematics to evaluate the precision of the approximation and why it would be a favoured method over the conventional simulation of the bit error performance of a linear code.

2.1 Binary Phase Shift Keying (BPSK) Modulation

Modulation is the process of mapping digital information (bits or symbols) into analog waveforms which match the characteristics of the channel. The waveforms used are deterministic and have finite energy. The modulated information is transmitted over the channel and is received by the receiver. By considering a modulation method with M possible waveform mappings, the mapping process can be described. Typically, the mapping is performed by taking a block of $k = \log_2 M$ bits at a time from the information sequence $\{m_i\}$ and selecting one of the $M = 2^k$ waveforms $\{s_k(t), k = 0, 1, \ldots, M-1\}$ for transmission. The waveforms are generally transmitted for a symbol duration of T seconds. This technique is widely used for transmissions over AWGN channels and is considered to be memoryless modulation since the current waveform to be transmitted does not depend on the previously transmitted waveforms.

Binary Phase Shift Keying (BPSK) is a modulation technique whereby the number of possible mappings, M, is 2. With any phase shift keying type of modulation, the information is transmitted within the phase of the signal. For BPSK modulation, the pair of signals, $s_0(t)$ and $s_1(t)$, representing the bits 0 and 1, respectively, can be represented

as [1]

$$\begin{aligned} s_k(t) &= Re\left[g(t)e^{j2\pi(k-1)/2}\, e^{j2\pi f_c t}\right], \quad k = 0, 1., \; 0 \le t \le T \\ &= g(t)\cos\Big[2\pi f_c t + \pi(k-1)\Big], \end{aligned} \tag{2.2}$$

where $g(t)$ is the signal pulse shape, which is nonzero in the interval $0 \le t \le T$ and zero elsewhere; f_c is the carrier frequency of the waveform; and, again, T is the bit duration[1]. The signals both have equal energy E, i.e.,

$$\begin{aligned} E &= \int_0^T s_k^2(t)dt \\ &= \frac{1}{2}\int_0^T g^2(t)dt = \frac{1}{2}E_g. \end{aligned} \tag{2.3}$$

Noting that the two signals of (2.2) have a common basis function with unit energy of $\phi(t) = \sqrt{2/E_g}g(t)\cos(2\pi f_c t)$, the signals become

$$\begin{aligned} s_0(t) &= \sqrt{\frac{E_g}{2}}\phi(t) \;\; = -\sqrt{E_b}\phi(t) \quad \text{`0'} \\ s_1(t) &= -\sqrt{\frac{E_g}{2}}\phi(t) \;\; = \sqrt{E_b}\phi(t) \quad \text{`1'} \end{aligned} \tag{2.4}$$

where the substitution $E_b = E_g/2$ was made to simplify the expressions.

It is clear that BPSK is characterized by a one-dimensional signal space with only one basis function required to represent both signals. The two signals are termed as being antipodal since the waveforms differ in their relative phase-shift by 180 degrees [3]. Signals, $s_0(t)$ and $s_1(t)$, can be represented in a signal space representation by their amplitudes of $-\sqrt{E_b}$ and $\sqrt{E_b}$, respectively, and are denoted s_0 and s_1. The signal space diagram for this modulation scheme is presented in figure 2.1 below.

[1] A symbol is comprised of one bit in a BPSK modulated system.

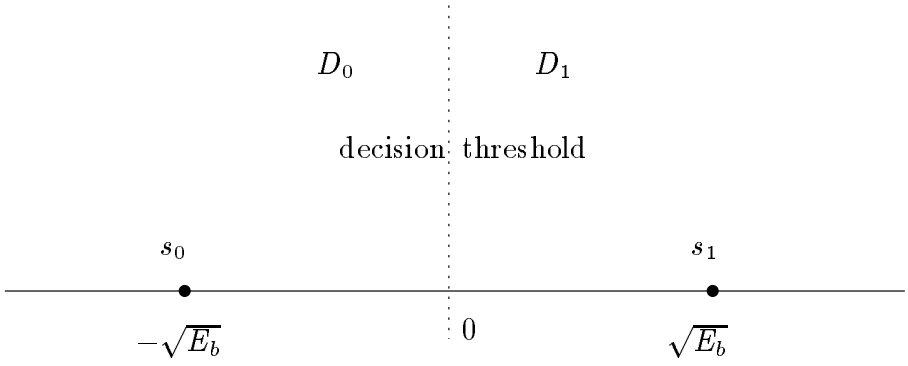

Figure 2.1: Signal Space Diagram for BPSK System

The two message points of figure 2.1 are separated by a distance of $2\sqrt{E_b}$.

The originally transmitted signals are not recovered at the receiver due to the introduction of AWGN. The noise can move the signal point to essentially any point in the signal space. To make a decision as to which bit was transmitted, 0 or 1, the signal space must be partitioned into two regions such that the following two scenarios are accounted for,

1. those received points more likely to be the message point at $\sqrt{E_b}$ are decided in favour of a 1 being transmitted, and

2. those received points more likely to be the message point at $-\sqrt{E_b}$ are decided in favour of a 0.

For two signals which are equally likely, as is usually the case, the midpoint between the two points denotes the decision region boundary as depicted in figure 2.1. A received signal point which is located in the decision region D_1 will be decided in favour of signal $s_1(t)$ and a signal point received in decision region D_0 of the figure will be decided in favour of signal $s_0(t)$. It is conceivable that an error may occur though.

Since the channel is modeled as being disturbed by AWGN, and assuming that signal point s_1 was transmitted, the received signal point r is of the form

$$r = s_1 + \eta = \sqrt{E_b} + \eta, \tag{2.5}$$

where η is the AWGN component with zero mean and variance $\sigma_\eta^2 = \frac{N_0}{2}$, as presented in section 1.1. The received component is Gaussian with mean $\sqrt{E_b}$ and variance $\frac{N_0}{2}$. The decision threshold for this system is 0 (due to equally likely signals), so that the decision rule comes down to observing r: if $r > 0$, decide in favour of s_1 and thus a 1 was transmitted; otherwise, decide in favour of s_0.

With this system, two possible types of errors can occur. If s_0 is transmitted and the noise component is such that the received signal point falls in region D_1, the receiver will then decide in favour of s_1 when in fact s_0 was transmitted. The second type of error occurs if s_1 is transmitted and the received signal point falls in D_0, causing the receiver to decide in favour of s_0. The probability of either error can be calculated and is presented in the next section.

2.2 Bit Error Probability for BPSK and the Q-function

It is customary to ask with what probability these two error types occur so that the performance of the system can be evaluated. From (2.5), it is intuitive to see that the larger that is the variance of the noise component η, for a given deterministic value of s_i, $i = 0, 1$, the large the variance of the received signal point r. This larger variance translates into more possible errors as crossings of the decision threshold can occur more frequently.

The bit error probability for BPSK will be formulated, assuming that the two signals

s_1 and s_0 are equally likely, each with probability 1/2, and are as depicted in figure 2.1.

The average bit error probability, P_e, can be found by averaging the error for the two error scenarios discussed earlier. The two error scenarios which can occur are:

1. error occurs given that s_1 was sent. This means that the received component r is less than 0 (i.e., $error|s_1$ is when $r < 0$);

2. error occurs given that s_0 was sent. This means that r is greater than 0 (i.e., $error|s_0$ is when $r > 0$).

Therefore,

$$\begin{aligned} P_e &= \Pr(s_1)\ \Pr(error|s_1) + \Pr(s_0)\ \Pr(error|s_0) \\ &= \frac{1}{2}\Pr(error|s_1) + \frac{1}{2}\Pr(error|s_0). \end{aligned} \tag{2.6}$$

To calculate the probabilities $\Pr(error|s_i), i = 0, 1$, the conditional probability density functions, $p(r|s_0)$ and $p(r|s_1)$, are needed. The conditional probability density functions are formed using equation (2.5). First, the conditional probability density function $p(r|s_1)$ is found, assuming that signal s_1 was sent. By manipulating $p(r|s_1)$, the form of the probability density function can be realized.

$$p(r|s_1) \sim p(r - s_1|s_1) \tag{2.7}$$

$$\sim p(\eta|s_1) \tag{2.8}$$

$$\sim p(\eta) \tag{2.9}$$

In (2.7) above, the shift of s_1 to r does not change the conditional probability density function. Using (2.5) and solving for η, (2.8) results. Noting that the noise component η is not dependent on the sent signal s_1, the conditioning of η on s_1 is not required.

The conditional probability density function then has the form of the probability density function of noise. Since the noise is Gaussian distributed, the probability density function is given by (1.1).

Therefore, the conditional probability density function, $p(r|s_1)$, is

$$\begin{aligned} p(r|s_1) &= \frac{1}{\sqrt{2\pi\sigma_\eta^2}} \exp\{-\frac{(r-s_1)^2}{2\sigma_\eta^2}\} \\ &= \frac{1}{\sqrt{\pi N_0}} \exp\{-\frac{(r-\sqrt{E_b})^2}{N_0}\}, \end{aligned} \tag{2.10}$$

where $\sigma_\eta^2 = N_0/2$ and $s_1 = \sqrt{E_b}$.

If $r = s_0 + \eta = -\sqrt{E_b} + \eta$, then $p(r|s_0)$ can be shown to have the same form as $p(r|s_1)$ in (2.10), with the only difference being the value of s_0 is substituted for the value of s_1. Therefore,

$$p(r|s_0) = \frac{1}{\sqrt{\pi N_0}} \exp\{-\frac{(r+\sqrt{E_b})^2}{N_0}\}. \tag{2.11}$$

If the two conditional probability density functions, $p(r|s_1)$ and $p(r|s_0)$, are superimposed on the signal space representation of figure 2.1, the following is observed.

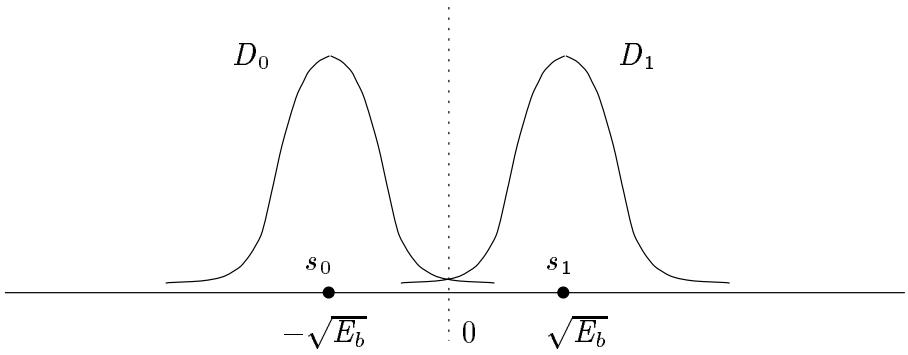

Figure 2.2: Conditional Probability Density Functions of Two Signals

Since the two signals were assumed to be equally likely, the decision threshold is simply 0. This is seen where the two functions in figure 2.2 intersect. Furthermore, the mean of the conditional probability density functions, μ_r, is $\sqrt{E_b}$ for $p(r|s_1)$ and $-\sqrt{E_b}$ for $p(r|s_0)$. This was can be seen implicitly through equation (2.5). Later in this chapter, these facts are used to demonstrate the modeling of soft-output decoding of binary linear codes using a BPSK system.

To calculate $\Pr(error|s_1) = \Pr(r < 0|s_1)$, the condition on r is imposed and the integral is calculated. Letting $\mu_r = \sqrt{E_b}$ and $\sigma_\eta^2 = N_0/2$, this leads to

$$\begin{aligned}\Pr(error|s_1) &= \int_{-\infty}^{0} p(r|s_1)\, dr \\ &= \int_{-\infty}^{0} \frac{1}{\sqrt{2\pi\sigma_\eta^2}} \exp\Big\{-\frac{(r-\mu_r)^2}{2\sigma_\eta^2}\Big\}\, dr. \end{aligned} \tag{2.12}$$

Substituting $t = \frac{r-\mu_r}{\sigma_\eta}$, $dt = dr/\sigma_\eta$, and appropriately changing the limits of integration, equation (2.12) becomes

$$\begin{aligned}\Pr(error|s_1) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{-\frac{\mu_r}{\sigma_\eta}} \exp\Big\{-\frac{t^2}{2}\Big\}\, dt \\ &= \frac{1}{\sqrt{2\pi}} \int_{\frac{\mu_r}{\sigma_\eta}}^{\infty} \exp\Big\{-\frac{t^2}{2}\Big\}\, dt \\ &= Q\big(\frac{\mu_r}{\sigma_\eta}\big), \end{aligned} \tag{2.13}$$

where the definition of the Q-function has been used. The Q-function is defined as [1]

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_{x}^{\infty} \exp\Big\{-\frac{t^2}{2}\Big\} dt. \tag{2.14}$$

Substituting the values of μ_r and σ_η into (2.13),

$$\Pr(error|s_1) = Q\left(\sqrt{\frac{2E_b}{N_0}}\right). \tag{2.15}$$

Similarly, if it is assumed that s_0 was transmitted, $r = -\sqrt{E_b} + \eta$ and the probability that $r > 0$ is also given as

$$\Pr(error|s_0) = Q\left(\sqrt{\frac{2E_b}{N_0}}\right). \tag{2.16}$$

Finally, the average bit error probability P_e, assuming equal probabilities for a 0 or 1, can be calculated using (2.6), and results in

$$P_e = Q\left(\sqrt{\frac{2E_b}{N_0}}\right) = Q\left(\frac{\mu_r}{\sigma_r}\right). \tag{2.17}$$

It is interesting to note that the average probability of bit error can be found with the Q-function, whose argument is simply the ratio of the mean of the received signal to the standard deviation of the received signal (which is the same as the variance of the AWGN noise). Remember, this formula can only be used for BPSK systems over an AWGN channel, producing Gaussian distributed received signal points. Gaussian samples are required for this calculation to be useful.

2.3 Gaussian Distributed LLR Values

Prior to this section, BPSK modulation was reviewed and the calculation of the average bit error probability was presented. Transmission of the modulated bits over an AWGN channel gives the received signal component, r, a Gaussian distribution. With a Gaussian distribution, the bit error probability can be calculated using the Q-function and the

ratio of the mean of the received component to the standard deviation of the received component. Since the signal constellation of a BPSK modulated system is simple, the bit error probability calculation is also straight forward.

It is the goal of this section to demonstrate that the LLR values can be approximated to be Gaussian distributed. This observation leads to the modeling of the soft-output decoding of binary linear codes by a BPSK modulation system. The Q-function is then useful in the approximation of the bit error performance of binary linear codes.

The Gaussian nature of the LLR values is shown through a Taylor series expansion of the LLR expression defined in (2.1). Since the codewords can be thought of as n dimensional vectors, the Taylor series expansion for vectors is required.

2.3.1 General Taylor Series Expansion for Vectors

Consider a vector $\boldsymbol{X}$ and a constant vector $\boldsymbol{a}$, each of dimensionality n. The general Taylor series expansion of function $f(\boldsymbol{X})$, about $\boldsymbol{a}$, is given as [8]

$$\begin{aligned} f(\boldsymbol{X}+\boldsymbol{a}) &= \sum_{v=0}^{\infty} \frac{[(\boldsymbol{X}\cdot\nabla)^v f](\boldsymbol{a})}{v!} \\ &= f(\boldsymbol{a}) + \boldsymbol{X}\cdot\nabla f(\boldsymbol{a}) + \frac{1}{2}(\boldsymbol{X}\cdot\nabla)^2 f(\boldsymbol{a}) + \ldots, \end{aligned} \tag{2.18}$$

where ∇ is the gradient operator. For a vector of length n, and $\hat{X}_i$, $i = 1, 2, \ldots, n$, used to denote unit components of the vector, ∇ is defined as

$$\nabla = \hat{X}_1\frac{\partial}{\partial X_1} + \hat{X}_2\frac{\partial}{\partial X_2} + \hat{X}_3\frac{\partial}{\partial X_3} + \ldots + \hat{X}_n\frac{\partial}{\partial X_n}. \tag{2.19}$$

The expression of (2.18) is directly applicable to the LLR function defined in (2.1), thereby yielding the Taylor series expansion of the LLR in powers of $\boldsymbol{\eta}$.

Using (2.1), the LLR for a bit m_k in arbitrary position k of the codeword is calculated,

by first dividing the code codebook into two subsets, $\mathcal{C}_1$ and $\mathcal{C}_0$, based upon the value of the bit in position k. Remember that the squared Euclidean distance is calculated between the received codeword $\boldsymbol{x}$ and the ± 1 modulated codewords of the two subsets, where 0 is mapped to -1 and 1 to 1. This point is crucial in order to illustrate that the LLR is approximately Gaussian distributed. With these facts in mind, the Taylor series expansion of the LLR is now presented.

2.3.2 Taylor Series Expansion of the LLR

The codeword vector $\boldsymbol{x}$ is obtained from the channel by the receiver. The received codeword differs from the assumed transmitted modulated codeword $\tilde{\boldsymbol{c}}$ by the addition of Additive White Gaussian Noise components to the deterministic codeword components. This means that the received codeword $\boldsymbol{x}$ can be written as the sum of $\tilde{\boldsymbol{c}}$ and $\boldsymbol{\eta}$. The n components of the received codeword are Gaussian distributed due to components of $\boldsymbol{\eta}$ being Gaussian distributed samples.

In order to facilitate the expansion of the LLR for a bit m_k, the expression of (2.1) is rewritten to separate the numerator and denominator into two similar terms using the properties of logarithms. These functions are a function of $\boldsymbol{\eta}$. Define function $H(\boldsymbol{\eta})$ as,

$$\begin{aligned} H(\boldsymbol{\eta}) = LLR(m_k) &= \log \frac{\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\{\frac{\boldsymbol{x}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\}}{\sum_{\boldsymbol{c}_j \in \mathcal{C}_0} \exp\{\frac{\boldsymbol{x}\cdot\boldsymbol{c}_j}{\sigma_\eta^2}\}} \\ &= \log \frac{\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\{\frac{(\boldsymbol{\eta}+\tilde{\boldsymbol{c}})\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\}}{\sum_{\boldsymbol{c}_j \in \mathcal{C}_0} \exp\{\frac{(\boldsymbol{\eta}+\tilde{\boldsymbol{c}})\cdot\boldsymbol{c}_j}{\sigma_\eta^2}\}} \\ &= \log\Big(\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_i + \tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\}\Big) - \log\Big(\sum_{\boldsymbol{c}_j \in \mathcal{C}_0} \exp\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_j + \tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_j}{\sigma_\eta^2}\}\Big) \qquad (2.20) \\ &= f(\boldsymbol{\eta}) - g(\boldsymbol{\eta}), \qquad (2.21) \end{aligned}$$

where, $\tilde{\boldsymbol{c}}$ is the assumed transmitted codeword,

$$f(\boldsymbol{\eta}) = \log\Big(\sum_{\boldsymbol{c}_i \,\epsilon\, \mathcal{C}_1} \exp\Big\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_i + \tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\Big\}\Big) \tag{2.22}$$

and,

$$g(\boldsymbol{\eta}) = \log\Big(\sum_{\boldsymbol{c}_j \,\epsilon\, \mathcal{C}_0} \exp\Big\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_j + \tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_j}{\sigma_\eta^2}\Big\}\Big). \tag{2.23}$$

The assumed transmitted codeword $\tilde{\boldsymbol{c}}$ can be any codeword of the code provided it does not alter the probability density function of the LLR for bit k. Assuming that the probability density of the noise is symmetrical about zero, the following theorem and proof justifies the arbitrary choice for the transmitted codeword.

Theorem 2.1 *The probability distribution of $LLR(m_k)$ is not affected by the choice of transmitted codeword $\tilde{\boldsymbol{c}}$ so long as the bit k remains unchanged.*

Proof: Assume the codeword $\tilde{\boldsymbol{c}}_\beta$ is transmitted, where the value of the bit in position k is $\beta = 0, 1$. For this proof, two properties will be used:

1. the distance invariance property of the code, and
2. the noise is symmetrical about the origin.

To show that the probability distribution of the LLR is not affected by a change in transmitted codeword, keeping the value of the bit in bit position k unchanged, the terms of (2.20) need to remain statistically unchanged. This can be realized by observing the terms $\boldsymbol{\eta}\cdot\boldsymbol{c}_j$ and $\tilde{\boldsymbol{c}}_\beta\cdot\boldsymbol{c}_j$ in the exponent of the expressions of (2.20), for different $\boldsymbol{c}_j \,\epsilon\, \mathcal{C}_1$ or $\mathcal{C}_0$ and different transmitted codeword $\tilde{\boldsymbol{c}}_\beta$.

For a given $\tilde{\boldsymbol{c}}_\beta$, the dot product takes on various integer values for different codewords of the sub-code $\mathcal{C}_0$ or coset $\mathcal{C}_1$. By changing the transmitted codeword $\tilde{\boldsymbol{c}}_\beta$ such that the

value of β is unchanged, the resulting values of the dot products are simply permuted values of those obtained using the originally assumed transmitted codeword. This is due to the Hamming distance profile between a transmitted codeword and any other codeword of the set being similar. Therefore, for different transmitted codewords with the value of β held constant, there is a simple reordering of the values for this part of the exponent expression. If the value of β was changed with the choice of another codeword to be transmitted, then a different Hamming distance profile would exist and would not necessarily equal that of the originally assumed transmitted codeword, changing the probability distribution of the LLR.

The noise vector in $\boldsymbol{\eta} \cdot \boldsymbol{c}_j$, for $\boldsymbol{c}_j \, \epsilon \, \mathcal{C}_1$ or $\mathcal{C}_0$, is unchanged for a given transmission instance and codeword $\boldsymbol{c}_j$. The dot product remains unchanged for different transmitted codewords $\tilde{\boldsymbol{c}}$. The association between the codewords $\boldsymbol{c}_j$ and noise vector is always maintained and results in sign changes of the components of the noise vector as dictated by the modulated codeword $\boldsymbol{c}_j$. The independence of $\boldsymbol{\eta} \cdot \boldsymbol{c}_j$ on the assumed transmitted codeword presents a problem that is resolvable by the properties assumed above.

The permutation of transmitted codeword dot product values for different assumed transmitted codewords needs to correspond to a similar permutation with the noise dot product values for the probability density function of the LLR to be unchanged. Then the summation of different exponentials still produces the same overall sum as values are simply permuted between the exponentials. The assumption of a symmetric probability density function about 0 helps in this respect. The signs of the noise components do not change statistical nature of the noise. Then, the statistical nature of the exponentials do not change with a change in the assumed transmitted codeword. The Gaussian distribution of the noise vectors is such a probability density function. Therefore, provided the value of bit k does not change with a change in the assumed transmitted codeword, any codeword can be assumed. ■

For the purpose of this thesis, the all-zero codeword $\mathbf{0} = (0, 0, \ldots, 0)$ is assumed to be transmitted, for simplicity. This means that $\tilde{\boldsymbol{c}} = -\mathbf{1}$ is transmitted, after considering BPSK modulation of the all-zero codeword. Therefore, for bit k, a zero is assumed to be transmitted.

The two functions $f(\boldsymbol{\eta})$ and $g(\boldsymbol{\eta})$ differ only in that the summations are carried out over different subsets of the same code, $\mathcal{C}_1$ and $\mathcal{C}_0$, respectively. The expansion involves linear operations on the function $H(\boldsymbol{\eta})$ and therefore, it is possible to carry out the expansion for one of the functions and then tailor the results for the other function.

For the Taylor series expansion of $f(\boldsymbol{\eta})$, the function is expanded about $\boldsymbol{a} = \mathbf{0}$ in powers of $\boldsymbol{\eta}$. The expansion of $f(\boldsymbol{\eta})$ is then,

$$\begin{aligned} f(\boldsymbol{\eta} + \mathbf{0}) &= f(\boldsymbol{\eta}) \\ &= \sum_{v=0}^{\infty} \frac{[(\boldsymbol{\eta} \cdot \nabla)^v f](\mathbf{0})}{v!} \\ &= f(\mathbf{0}) + \boldsymbol{\eta} \cdot \nabla f(\mathbf{0}) + \frac{1}{2}(\boldsymbol{\eta} \cdot \nabla)^2 f(\mathbf{0}) + \ldots . \end{aligned} \tag{2.24}$$

The mathematics of this expansion will follow the next section, starting first with the zeroth-order term, then the first-order term, and finally the second-order term of (2.24).

2.3.3 Useful Theorems and Definitions

Before continuing, a few theorems are introduced since they will be needed to simplify and evaluate the terms of the expansion. The theorems relate to the structure of linear codes and the multiplication of columns of bits within codebooks. The theorems are then extended to subsets of a code since the code $\mathcal{C}$ is divided into $\mathcal{C}_0$ and $\mathcal{C}_1$ in the LLR expression.

To simplify the presentation of the expressions to come, weight enumeration func-

tions will be defined. Weight enumeration functions are useful in providing a means to conveniently represent the weight distributions of a code, where the weight distribution is a set of all weights (or number of non-zero elements) of the codewords. Using weight distribution notation make the expressions easier to read, and make calculations of the LLR possible based upon the structure of the code.

Theorem 2.2 *In any column of the codebook of a binary linear code, there are an equal number of ones and zeros.*

Proof: The proof follows from the fact that binary linear codes form a closed group under modulo 2 addition. The trivial case of all 0's or 1's is not considered since these columns can be removed from the codebook without changing the properties of the code. ■

Due to the original codebook being divided into two subsets for the LLR calculations, a corollary to Theorem 2.2 is needed for the case of a coset. The case of a sub-code follows directly from Theorem 2.2.

Corollary 2.2.1 *The columns of a coset also have an equal number of ones and zeros.*

Proof: The coset is formed by adding a given codeword (coset leader) to all the elements of the sub-code. Therefore, the codewords in the coset exhibit the same properties of the original code. ■

The codewords used in the calculation of the LLR are all ± 1 modulated, and since the number of 0's and 1's in a column are equal, there are an equal number of -1's and 1's in a given column, not considering the trivial cases.

Another theorem which is required is one that relates to the multiplication of columns of the ± 1 modulated codebook.

Theorem 2.3 *The multiplication of two columns of a* ± 1 *modulated linear codebook yields equal numbers of* -1*'s and* 1*'s.*

Proof: The proof of this theorem is assisted by figure 2.3. Consider two arbitrary columns of a modulated code $\mathcal{C}$, i and j, $1 \leq i,j \leq n$. When the bits of the two columns

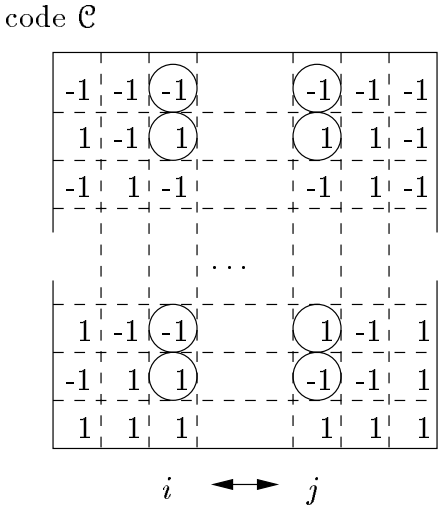

Figure 2.3: Multiplication of Two Columns of a ± 1 Modulated Code $\mathcal{C}$

are multiplied together, the following combinations are encountered: '$-1,-1$', '$-1,1$', '$1,-1$', and '$-1,-1$', yielding 1, -1, -1, and 1. Since the bits in any column of a code are equally probable to be a 0 or 1, and since the combinations produce quantities fitting this same probability distribution (equally 1 or -1), following from Theorem 2.2, there are an equal number of -1's and 1's. ∎

Corollary 2.3.1 *The multiplication of two columns of a sub-code or coset yields an equal number of* -1*'s and* 1*'s in the resulting column.*

Proof: The proof of this corollary follows from that of Theorem 2.3. If the two columns of sub-code or coset happen to be identical, then an all-one trivial valued column results. If one of the columns considered is comprised of all 1's or -1's, the resulting column still maintains it's equal 1's and -1's. ∎

The multiplication of two columns of a modulated code or sub-code yields an equal number of 1's and -1's by Theorem 2.3. The resulting combinations discussed in the proof of this theorem can be equally represented by considering the exclusive-or of the unmodulated bits of the columns. This operation will be denoted $\oplus$ in later expressions.

Weight Enumeration Function Definitions

It will be convenient to define a number of weight enumeration functions to simplify later expressions. Given an unmodulated codeword, it is known that the sum of the bits would produce the weight of the codeword, since the number of ones would be represented by the sum. A weight enumeration function is a convenient manner to present the weight distribution of a code. The set of weights is presented as a polynomial in powers of a dummy variable raised to an exponent. The exponent is the codeword weight, and the coefficient of the dummy variable is the number of codewords of this weight. The weight of codeword $\boldsymbol{c}$ is commonly denoted by w.

A typical expression for the weight enumeration function of a code $\mathcal{C}$, with codewords of length n, is given by

$$A^{\mathcal{C}}(Z) = \sum_{w=0}^{n} A_w^{\mathcal{C}} Z^w \tag{2.25}$$

where Z is the dummy variable and $A_w^{\mathcal{C}}$ is the number of codewords of $\mathcal{C}$ with weight w.

The codewords in the expression of the LLR and the subsequent Taylor series are all modulated ± 1. To represent this fact, the weight enumeration function above is slightly modified so that the exponent of the dummy variable Z is changed to reflect the modulated nature of the codewords. The integer sum of the bits of modulated codewords can range from between $-n$ (the all-zero codeword) to n (the all-one codeword). The weights of the codewords can be mapped into this range by the expression $2w - n$. Adjusting (2.25) as

stated, the following expression for code $\mathcal{C}$ is obtained.

$$A^{\mathcal{C}}(Z) = \sum_{w=0}^{n} A_w^{\mathcal{C}} \, Z^{2w-n} \, . \tag{2.26}$$

where $A_w^{\mathcal{C}}$ is the number codewords of code $\mathcal{C}$ with weight w. Note that this expression is for a code $\mathcal{C}$ with codewords of length n. This expression can also be used for sub-code $\mathcal{C}_0$ and its coset $\mathcal{C}_1$.

Also, a weight enumeration function that associates the weights of codewords with the value of the bit in position p equal to α is defined as,

$$B^{\mathcal{C}}(Z, p, \alpha) = \sum_{w=0}^{n} B_w^{\mathcal{C}}(p, \alpha) \, Z^{2w-n} \, . \tag{2.27}$$

The coefficients $B_w^{\mathcal{C}}(p, \alpha)$ denote the number of codewords of weight w for which the bit in position p is equal in value to $\alpha = 0, 1$. The coefficients here are obtained from the codebook in a similar manner to the coefficients $A_w^{\mathcal{C}}$ in (2.26), however, extra care must be taken to account for the bit value of bit position p.

The method by which the weight distribution of a linear code is found, in order to obtain the coefficients used in the weight enumeration function, is discussed in chapter 7, where a new method of finding the weight distribution of a binary linear code is introduced, along with the required background material.

In the following presentation of the expansion, $|\mathcal{C}_i|$ denotes the number of codewords in the subset $\mathcal{C}_i$, $i = 0, 1$. With the theorems and definitions above, the simplification of the terms of the Taylor series expansion is made possible.

2.3.4 Continuation of the Expansion

The function $f(\boldsymbol{\eta})$ was given in (2.24) as

$$f(\boldsymbol{\eta}) = f(\mathbf{0}) + \boldsymbol{\eta} \cdot \nabla f(\mathbf{0}) + \frac{1}{2}(\boldsymbol{\eta} \cdot \nabla)^2 f(\mathbf{0}) + \frac{1}{6}(\boldsymbol{\eta} \cdot \nabla)^3 f(\mathbf{0}) + \ldots .$$

The expressions involving the dot product of the noise vector, $\boldsymbol{\eta}$, and the gradient, ∇, are provided below. The noise vector, $\boldsymbol{\eta}$, has components which will be labeled as $(\eta_1, \eta_2, \ldots, \eta_n)$.

$$\begin{aligned} \boldsymbol{\eta} \cdot \nabla &= \eta_1 \frac{\partial}{\partial \eta_1} + \eta_2 \frac{\partial}{\partial \eta_2} + \ldots + \eta_n \frac{\partial}{\partial \eta_n} \\ &= \sum_{p=1}^{n} \eta_p \, \frac{\partial}{\partial \eta_p} \end{aligned} \tag{2.28}$$

$$\begin{aligned} (\boldsymbol{\eta} \cdot \nabla)^2 &= \big(\eta_1 \frac{\partial}{\partial \eta_1} + \eta_2 \frac{\partial}{\partial \eta_2} + \ldots + \eta_n \frac{\partial}{\partial \eta_n}\big)^2 \\ &= \sum_{v=1}^{n} \eta_v^2 \, \frac{\partial^2}{\partial \eta_v^2} + 2 \sum_{p \neq q,\, p<q \leq n}^{n} \sum^{n} \eta_p \, \eta_q \, \frac{\partial}{\partial \eta_p} \, \frac{\partial}{\partial \eta_q}. \end{aligned} \tag{2.29}$$

The expressions for the dot products obtained above, and the assumed transmitted modulated codeword $\tilde{\boldsymbol{c}} = -\mathbf{1}$ can be applied to $f(\boldsymbol{\eta})$ to obtain the terms of the expansion. The partial derivatives will be carried out for non-specific component variables initially, e.g. η_p, and then conditions, if necessary, will be placed on the variables to fit the expressions above in (2.28), and (2.29). The higher-order partial derivatives of the series are more complicated to express and no general closed form expression exists and therefore, the series is only presented to the second-order.

Calculating $f(\mathbf{0})$

Using the definition of $f(\boldsymbol{\eta})$ in equation (2.22), and substituting $\boldsymbol{\eta} = \mathbf{0}$, $f(\mathbf{0})$ is,

$$\begin{aligned} f(\mathbf{0}) &= \log\Big(\sum_{\boldsymbol{c}_i \,\epsilon\, \mathcal{C}_1} \exp\Big\{ \frac{\boldsymbol{\eta} \cdot \boldsymbol{c}_i + -\mathbf{1} \cdot \boldsymbol{c}_i}{\sigma_\eta^2} \Big\} \Big) \Bigg|_{\boldsymbol{\eta}=\mathbf{0}} \\ &= \log\Big(\sum_{\boldsymbol{c}_i \,\epsilon\, \mathcal{C}_1} \exp\Big\{ 0 - \frac{1}{\sigma_\eta^2} \sum_{v=1}^{n} c_{iv} \Big\} \Big). \end{aligned} \tag{2.30}$$

The summation above can be rewritten as a weight enumeration function, since the exponent is simply the modulated weights of the codewords of $\mathcal{C}_1$. Using the definition of (2.26), the term $f(\mathbf{0})$ is then

$$\begin{aligned} f(\mathbf{0}) = \log\Big(\sum_{w=0}^{n} A_w^{\mathcal{C}_1} \exp\Big\{ -\frac{1}{\sigma_\eta^2}(2w - n) \Big\}\Big) & \\ = \log\Big(\sum_{w=0}^{n} A_w^{\mathcal{C}_1} Z^{2w-n}\Big) & \\ = \log\big(A^{\mathcal{C}_1}(Z)\big) & \end{aligned} \tag{2.31}$$

where, $Z = \exp\{-\frac{1}{\sigma_\eta^2}\}$ and will be defined as such for the purpose of the Taylor series expansion. The above Taylor series term is simply a constant with no random component and would simply shift the resulting distribution of the LLR.

Calculating $\frac{\partial f}{\partial \eta_p}(\mathbf{0})$

For a given bit position p, $1 \leq p \leq n$,

$$\frac{\partial f(\boldsymbol{\eta})}{\partial \eta_p}\bigg|_{\boldsymbol{\eta}=\mathbf{0}} = \frac{\partial}{\partial \eta_p}\left[\log\Big(\sum_{\boldsymbol{c}_i \,\epsilon\, \mathcal{C}_1} \exp\Big\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_i + -\mathbf{1}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\Big\}\Big)\right]\Bigg|_{\boldsymbol{\eta}=\mathbf{0}}$$

$$= \left[\frac{1}{\sum_{\boldsymbol{c}_i \,\epsilon\, \mathcal{C}_1} \exp\Big\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_i + -\mathbf{1}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\Big\}}\Big(\sum_{\boldsymbol{c}_j \,\epsilon\, \mathcal{C}_1} \frac{c_{jp}}{\sigma_\eta^2} \exp\Big\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_j + -\mathbf{1}\cdot\boldsymbol{c}_j}{\sigma_\eta^2}\Big\}\Big)\right]\Bigg|_{\boldsymbol{\eta}=\mathbf{0}} \tag{2.32}$$

$$= \frac{1}{\sigma_\eta^2}\frac{\sum_{\boldsymbol{c}_j \,\epsilon\, \mathcal{C}_1} c_{jp} \exp\big\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n} c_{jv}\big\}}{\sum_{\boldsymbol{c}_i \,\epsilon\, \mathcal{C}_1} \exp\big\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n} c_{iv}\big\}} \tag{2.33}$$

Again, the denominator expression of (2.33) can be rewritten below in terms of weight enumeration function using the definition of (2.26). The numerator expression requires special care due to the component of the codewords multiplied with the exponentials. The weight enumeration function definition of (2.27) is suited for this situation and is used later for the different possible positions which arbitrary p can take.

$$\frac{\partial f(\boldsymbol{\eta})}{\partial \eta_p}\bigg|_{\boldsymbol{\eta}=\mathbf{0}} = \frac{1}{\sigma_\eta^2}\frac{\sum_{\boldsymbol{c}_j \,\epsilon\, \mathcal{C}_1} c_{jp} \exp\big\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n} c_{jv}\big\}}{\sum_{w=0}^{n} A_w^{\mathcal{C}_1} Z^{2w-n}}$$

$$= \frac{1}{\sigma_\eta^2}\frac{\sum_{\boldsymbol{c}_j \,\epsilon\, \mathcal{C}_1} c_{jp} \exp\big\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n} c_{jv}\big\}}{A^{\mathcal{C}_1}(Z)} \tag{2.34}$$

Considering different scenarios for bit position p in the codeword, the expression can

be further simplified. For example, if the bit position is one where all codewords of the subset have the same bit value, c_{jp} becomes $+1$ for $\mathcal{C}_1$, and further simplification is possible. However, this is a trivial case. The following expression results for bit position p.

By Theorem 2.2 and the corollaries above, when the bits are not all identical, the column of the subset contain an equal number of -1's and 1's. Using (2.27), the expression is simplified as,

$$\begin{aligned}
\left.\frac{\partial f(\boldsymbol{\eta})}{\partial \eta_p}\right|_{\boldsymbol{\eta}=\mathbf{0}} &= \frac{1}{\sigma_\eta^2}\frac{\sum_{\boldsymbol{c}_j \,\epsilon\, \mathcal{C}_1} c_{jp} \exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n} c_{jv}\}}{A^{\mathcal{C}_1}(Z)} \\
&= \frac{1}{\sigma_\eta^2}\frac{\sum_{w=0}^{n} B_w^{\mathcal{C}_1}(p,1)Z^{2w-n} - \sum_{w=0}^{n} B_w^{\mathcal{C}_1}(p,0)Z^{2w-n}}{A^{\mathcal{C}_1}(Z)} \\
&= \frac{1}{\sigma_\eta^2}\,\frac{B^{\mathcal{C}_1}(Z,p,1) - B^{\mathcal{C}_1}(Z,p,0)}{A^{\mathcal{C}_1}(Z)}. \qquad (2.35)
\end{aligned}$$

This expression of the coefficient is in terms of the weight distribution of the subset, which can be easily computed.

The first-order term using (2.28), and (2.35) becomes,

$$\begin{aligned}
[(\boldsymbol{\eta}\cdot\nabla)f](\mathbf{0}) &= \sum_{p=1}^{n} \eta_p \frac{\partial}{\partial \eta_p} f(\boldsymbol{\eta})\Big|_{\boldsymbol{\eta}=\mathbf{0}} \\
&= \frac{1}{\sigma_\eta^2}\sum_{p=1}^{n}\frac{\left[B^{\mathcal{C}_1}(Z,p,1) - B^{\mathcal{C}_1}(Z,p,0)\right]}{A^{\mathcal{C}_1}(Z)}\,\eta_p \qquad (2.36)
\end{aligned}$$

The trivial case of bit position p being one where all the bit values are identical results in $B^{\mathcal{C}_1}(Z,p,0)$ equaling 0.

Calculating $\dfrac{\partial^2 f}{\partial \eta_p \, \partial \eta_q}(\mathbf{0})$

The second-order gradient term was given in equation (2.29). Again, for arbitrary bit positions, p and q, $1 \le p, q \le n$, the second-order partial derivative terms, starting with (2.32), are of the form,

$$
\begin{aligned}
\frac{\partial^2 f(\mathbf{0})}{\partial \eta_p \partial \eta_q} &= \frac{\partial}{\partial \eta_q} \frac{\partial f(\boldsymbol{\eta})}{\partial \eta_p} \Big|_{\boldsymbol{\eta}=\mathbf{0}} \\
&= \frac{\partial}{\partial \eta_q} \left[\frac{1}{\sigma_\eta^2} \frac{\sum_{\boldsymbol{c}_j \in \mathcal{C}_1} c_{jp} \exp\{\frac{\boldsymbol{\eta} \cdot \boldsymbol{c}_j + -\mathbf{1} \cdot \boldsymbol{c}_j}{\sigma_\eta^2}\}}{\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\{\frac{\boldsymbol{\eta} \cdot \boldsymbol{c}_i + -\mathbf{1} \cdot \boldsymbol{c}_i}{\sigma_\eta^2}\}} \right] \Bigg|_{\boldsymbol{\eta}=\mathbf{0}} \\
&= \frac{1}{\sigma_\eta^2} \left[\frac{\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\{\frac{\boldsymbol{\eta} \cdot \boldsymbol{c}_i + -\mathbf{1} \cdot \boldsymbol{c}_i}{\sigma_\eta^2}\} \sum_{\boldsymbol{c}_j \in \mathcal{C}_1} \frac{c_{jp}\, c_{jq}}{\sigma_\eta^2} \exp\{\frac{\boldsymbol{\eta} \cdot \boldsymbol{c}_j + -\mathbf{1} \cdot \boldsymbol{c}_j}{\sigma_\eta^2}\}}{\big[\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\{\frac{\boldsymbol{\eta} \cdot \boldsymbol{c}_i + -\mathbf{1} \cdot \boldsymbol{c}_i}{\sigma_\eta^2}\} \big]^2} \right. \\
&\qquad \left. - \frac{\sum_{\boldsymbol{c}_j \in \mathcal{C}_1} c_{jp} \exp\{\frac{\boldsymbol{\eta} \cdot \boldsymbol{c}_j + -\mathbf{1} \cdot \boldsymbol{c}_j}{\sigma_\eta^2}\} \sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \frac{c_{iq}}{\sigma_\eta^2} \exp\{\frac{\boldsymbol{\eta} \cdot \boldsymbol{c}_i + -\mathbf{1} \cdot \boldsymbol{c}_i}{\sigma_\eta^2}\}}{\big[\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\{\frac{\boldsymbol{\eta} \cdot \boldsymbol{c}_i + -\mathbf{1} \cdot \boldsymbol{c}_i}{\sigma_\eta^2}\} \big]^2} \right] \Bigg|_{\boldsymbol{\eta}=\mathbf{0}} \\
&= \frac{1}{\sigma_\eta^4} \frac{1}{\Big(\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\{-\frac{1}{\sigma_\eta^2} \sum_{v=1}^{n} c_{iv}\} \Big)^2} \times \\
&\qquad \Big[\Big(\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\{-\frac{1}{\sigma_\eta^2} \sum_{v=1}^{n} c_{iv}\} \Big) \Big(c_{1p} c_{1q} \exp\{-\frac{1}{\sigma_\eta^2} \sum_{v=1}^{n} c_{1v}\} \\
&\qquad\quad + c_{2p} c_{2q} \exp\{-\frac{1}{\sigma_\eta^2} \sum_{v=1}^{n} c_{2v}\} + \ldots + c_{|\mathcal{C}_1|p} c_{|\mathcal{C}_1|q} \exp\{-\frac{1}{\sigma_\eta^2} \sum_{v=1}^{n} c_{|\mathcal{C}_1|v}\} \Big) \\
&\qquad\quad - \Big(\sum_{\boldsymbol{c}_j \in \mathcal{C}_1} c_{jp} \exp\{-\frac{1}{\sigma_\eta^2} \sum_{v=1}^{n} c_{jv}\} \Big) \Big(\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} c_{iq} \exp\{-\frac{1}{\sigma_\eta^2} \sum_{v=1}^{n} c_{iv}\} \Big) \Big] \qquad (2.37)
\end{aligned}
$$

Similar to what was done in obtaining the first-order term coefficients, different scenarios are considered for the bit positions p and q so that the expression of (2.37) can be appropriately simplified. In the expansion of (2.29), the second-order partial derivatives can be seen to be for identical bit positions (i.e. $p = q$) or for different bit positions (i.e. $p \neq q$). These two cases produce different results when applied to (2.37). As well, care must be taken to include the effects of the case that one of the bit positions p or q is bit position k, since the values in this bit position are all identical within the subset of codewords. The resulting values of $\dfrac{\partial^2}{\partial\eta_p\,\partial\eta_q} f(\mathbf{0})$ can be found as shown below.

- $p = q$ and $p = k$ OR $p \neq q$, however all bit elements are one value (-1's or 1's): Since all the elements are the same in the bit positions, the product of $c_{jp}c_{jq}$ for any codeword j will be 1. Also, the single coefficients c_{jp} and c_{jq} will all be 1 (since $\mathcal{C}_1$ is considered). Recall that the denominator term can be replaced by the weight enumeration function of (2.26).

$$\begin{aligned}
\left.\frac{\partial^2 f(\boldsymbol{\eta})}{\partial\eta_k^2}\right|_{\boldsymbol{\eta}=\mathbf{0}} &= \frac{1}{\sigma_\eta^4}\left[\frac{\left(A^{\mathcal{C}_1}(Z)\right)\left(\sum\limits_{\boldsymbol{c}_j\,\epsilon\,\mathcal{C}_1} 1\exp\{-\frac{1}{\sigma_\eta^2}\sum\limits_{v=1}^{n} c_{jv}\}\right)}{\left(A^{\mathcal{C}_1}(Z)\right)^2}\right. \\
&\qquad \left. - \frac{\left(\sum\limits_{\boldsymbol{c}_j\,\epsilon\,\mathcal{C}_1} 1\exp\{-\frac{1}{\sigma_\eta^2}\sum\limits_{v=1}^{n} c_{jv}\}\right)\left(\sum\limits_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1} 1\exp\{-\frac{1}{\sigma_\eta^2}\sum\limits_{v=1}^{n} c_{iv}\}\right)}{\left(A^{\mathcal{C}_1}(Z)\right)^2}\right] \\
&= \frac{1}{\sigma_\eta^4}\left[\frac{\left(A^{\mathcal{C}_1}(Z)\right)\left(A^{\mathcal{C}_1}(Z)\right)}{\left(A^{\mathcal{C}_1}(Z)\right)^2} - \frac{\left(A^{\mathcal{C}_1}(Z)\right)\left(A^{\mathcal{C}_1}(Z)\right)}{\left(A^{\mathcal{C}_1}(Z)\right)^2}\right] \\
&= 0 \qquad (2.38)
\end{aligned}$$

- $p = q$ and $p \neq k$, OR $p \neq q$ but columns are identical: This case involves identical column considerations. When the columns are identical or if a column is considered

with itself, the product of the two bits in each codeword will be 1. The corresponding term is then

$$\begin{aligned}
\left.\frac{\partial^2 f(\boldsymbol{\eta})}{\partial\eta_p\partial\eta_p}\right|_{\boldsymbol{\eta}=\mathbf{0}} &= \frac{1}{\sigma_\eta^4}\frac{1}{\Big(\sum\limits_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1}\exp\{-\frac{1}{\sigma_\eta^2}\sum\limits_{v=1}^{n}c_{iv}\}\Big)^2}\times \\
&\quad\Bigg[\Big(\sum_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{iv}\}\Big)\Big(c_{1p}^2\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{1v}\} \\
&\qquad + c_{2p}^2\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{2v}\}+\ldots+c_{|\mathcal{C}_1|p}^2\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{|\mathcal{C}_1|v}\}\Big) \\
&\qquad -\Big(\sum_{\boldsymbol{c}_j\,\epsilon\,\mathcal{C}_1}c_{jp}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{jv}\}\Big)\Big(\sum_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1}c_{ip}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{iv}\}\Big)\Bigg] \\
&= \frac{1}{\sigma_\eta^4}\frac{1}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2}\Bigg[\Big(A^{\mathcal{C}_1}(Z)\Big)\Big(\sum_{\boldsymbol{c}_j\,\epsilon\,\mathcal{C}_1}1\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{jv}\}\Big) \\
&\qquad -\Big(\sum_{w=0}^{n}B_w^{\mathcal{C}_1}(p,1)Z^{2w-n}-\sum_{w=0}^{n}B_w^{\mathcal{C}_1}(p,0)Z^{2w-n}\Big)^2\Bigg] \\
&= \frac{1}{\sigma_\eta^4}\left[\frac{\Big(A^{\mathcal{C}_1}(Z)\Big)^2}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2}-\frac{\Big(B^{\mathcal{C}_1}(Z,p,1)-B^{\mathcal{C}_1}(Z,p,0)\Big)^2}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2}\right] \\
&= \frac{1}{\sigma_\eta^4}\left[1-\frac{\Big(B^{\mathcal{C}_1}(Z,p,1)-B^{\mathcal{C}_1}(Z,p,0)\Big)^2}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2}\right] \qquad (2.39)
\end{aligned}$$

The scenario considered before this scenario is a special case of this situation.

- $p \neq q$, either $p = k$ or $q = k$, and other column does not contain identical values: One of the bit positions considered contains values which are all the same. This only, at most, changes the sign of the coefficients $c_{ip}c_{iq}$. By Theorem 2.2 and the corollaries 2.2.1 and 2.3.1, the number of ones and zeros are equal in the other column. The bit-position-dependent weight enumeration function definition can be

used once again. Assuming $q = k$, and proceeding, these observations yield,

$$\begin{aligned}
\left.\frac{\partial^2 f(\boldsymbol{\eta})}{\partial\eta_k\partial\eta_p}\right|_{\boldsymbol{\eta}=\mathbf{0}} &= \frac{1}{\sigma_\eta^4}\frac{1}{\Big(\sum\limits_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1}\exp\{-\frac{1}{\sigma_\eta^2}\sum\limits_{v=1}^{n}c_{iv}\}\Big)^2}\times \\
&\quad\Bigg[\Big(\sum_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{iv}\}\Big)\Big(c_{1p}c_{1k}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{1v}\} \\
&\qquad + c_{2p}c_{2k}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{2v}\}+\ldots+c_{|\mathcal{C}_1|p}c_{|\mathcal{C}_1|k}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{|\mathcal{C}_1|v}\}\Big) \\
&\qquad - \Big(\sum_{\boldsymbol{c}_j\,\epsilon\,\mathcal{C}_1}c_{jp}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{jv}\}\Big)\Big(\sum_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1}c_{ik}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{iv}\}\Big)\Bigg] \\
&= \frac{1}{\sigma_\eta^4}\frac{1}{A^{\mathcal{C}_1}(Z)}\Bigg[\Big(A^{\mathcal{C}_1}(Z)\Big)\Big(c_{1p}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{1v}\} \\
&\qquad + c_{2p}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{2v}\}+\ldots+c_{|\mathcal{C}_1|p}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{|\mathcal{C}_1|v}\}\Big) \\
&\qquad - \Big(\sum_{\boldsymbol{c}_j\,\epsilon\,\mathcal{C}_1}c_{jp}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{jv}\}\Big)\Big(\sum_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1}1\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{iv}\}\Big)\Bigg] \\
&= \frac{1}{\sigma_\eta^4}\Bigg[\frac{\Big(A^{\mathcal{C}_1}(Z)\Big)\Big(\sum\limits_{w=0}^{n}B_w^{\mathcal{C}_1}(p,1)Z^{2w-n}-\sum\limits_{w=0}^{n}B_w^{\mathcal{C}_1}(p,0)Z^{2w-n}\Big)}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2} \\
&\qquad - \frac{\Big(\sum\limits_{w=0}^{n}B_w^{\mathcal{C}_1}(p,1)Z^{2w-n}-\sum\limits_{w=0}^{n}B_w^{\mathcal{C}_1}(p,0)Z^{2w-n}\Big)\Big(A^{\mathcal{C}_1}(Z)\Big)}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2}\Bigg] \\
&= 0 \qquad (2.40)
\end{aligned}$$

- $p \neq q$ and $p \neq k$ and $q \neq k$: Theorem 2.3 and its corollaries are used for this case. Since the two bit positions are not the same, and the bit values of the columns differ, the product of the bits will produce an equal number of ones and zeros much like the columns considered. Also, the coefficients c_{ip} (or c_{jq}) will exhibit a similar

behaviour, following from Theorem 2.2 and its corollaries. It is assumed for this case that the two positions have completely different values in their columns p and q. Therefore,

$$
\begin{aligned}
\left.\frac{\partial^2 f(\boldsymbol{\eta})}{\partial\eta_p\partial\eta_q}\right|_{\boldsymbol{\eta}=\mathbf{0}} &= \frac{1}{\sigma_\eta^4}\frac{1}{\Big(\sum_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{iv}\}\Big)^2}\times \\
&\quad\Bigg[\Big(\sum_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{iv}\}\Big)\Big(c_{1p}c_{1q}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{1v}\} \\
&\qquad + c_{2p}c_{2k}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{2v}\}+\ldots+c_{|\mathcal{C}_1|p}c_{|\mathcal{C}_1|q}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{|\mathcal{C}_1|v}\}\Big) \\
&\qquad -\Big(\sum_{\boldsymbol{c}_j\,\epsilon\,\mathcal{C}_1}c_{jp}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{jv}\}\Big)\Big(\sum_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1}c_{iq}\exp\{-\frac{1}{\sigma_\eta^2}\sum_{v=1}^{n}c_{iv}\}\Big)\Bigg] \\
&= \frac{1}{\sigma_\eta^4}\Bigg[\frac{\Big(A^{\mathcal{C}_1}(Z)\Big)\Big(\sum_{w=0}^{n}B_w^{\mathcal{C}_1}(p\oplus q,1)Z^{2w-n}-\sum_{w=0}^{n}B_w^{\mathcal{C}_1}(p\oplus q,0)Z^{2w-n}\Big)}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2} \\
&\qquad -\frac{\Big(\sum_{w=0}^{n}B_w^{\mathcal{C}_1}(p,1)Z^{2w-n}-\sum_{w=0}^{n}B_w^{\mathcal{C}_1}(p,0)Z^{2w-n}\Big)}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2}\times \\
&\qquad \frac{\Big(\sum_{w=0}^{n}B_w^{\mathcal{C}_1}(q,1)Z^{2w-n}-\sum_{w=0}^{n}B_w^{\mathcal{C}_1}(q,0)Z^{2w-n}\Big)}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2}\Bigg] \\
&= \frac{1}{\sigma_\eta^4}\Bigg[\frac{B^{\mathcal{C}_1}(Z,p\oplus q,1)-B^{\mathcal{C}_1}(Z,p\oplus q,0)}{A^{\mathcal{C}_1}(Z)} \\
&\qquad -\frac{\Big(B^{\mathcal{C}_1}(Z,p,1)-B^{\mathcal{C}_1}(Z,p,0)\Big)\Big(B^{\mathcal{C}_1}(Z,q,1)-B^{\mathcal{C}_1}(Z,q,0)\Big)}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2}\Bigg] \\
&= \frac{1}{\sigma_\eta^4}F. \qquad (2.41)
\end{aligned}
$$

In the above equation, the weight enumeration function denoted by $B^{\mathcal{C}_1}(Z, p \oplus q, 1)$ denotes that the weight of the codeword is only considered in this function if the result of the exclusive-or of the bits of position p and position q is a 1.

In the above, F is defined as

$$F = \frac{B^{\mathcal{C}_1}(Z, p \oplus q, 1) - B^{\mathcal{C}_1}(Z, p \oplus q, 0)}{A^{\mathcal{C}_1}(Z)} - \frac{\Big(B^{\mathcal{C}_1}(Z, p, 1) - B^{\mathcal{C}_1}(Z, p, 0)\Big)\Big(B^{\mathcal{C}_1}(Z, q, 1) - B^{\mathcal{C}_1}(Z, q, 0)\Big)}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2} \tag{2.42}$$

for convenience.

With the second-order derivatives formulated in equations (2.38 - 2.41) and (2.42), and using the second-order gradient term of (2.29), the complete second-order term of the Taylor series expansion in (2.24) is,

$$\begin{aligned}\frac{1}{2}(\boldsymbol{\eta} \cdot \nabla)^2 f(\mathbf{0}) &= \frac{1}{2}\sum_{v=1}^{n} \eta_v^2 \frac{\partial^2}{\partial \eta_v^2} f(\mathbf{0}) + \sum_{p \neq q,\, p<q \leq n}^{n}\sum^{n} \eta_p \, \eta_q \frac{\partial}{\partial \eta_p}\frac{\partial}{\partial \eta_q} f(\mathbf{0}) \\ &= \frac{1}{2\sigma_\eta^4} \sum_{v=1, v \neq k}^{n} \left[1 - \frac{\Big(B^{\mathcal{C}_1}(Z, v, 1) - B^{\mathcal{C}_1}(Z, v, 0)\Big)^2}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2}\right] \eta_v^2 \\ &\quad + \frac{1}{\sigma_\eta^4} \sum_{\substack{p=1 \\ p \neq q, p<q \leq n,\ p,q \neq k}}^{n}\sum_{q=1}^{n} F \; \eta_p \, \eta_q \end{aligned} \tag{2.43}$$

Expression for the Expansion of $f(\boldsymbol{\eta})$

Having formulated the individual terms which ultimately form the expansion in equations (2.31), (2.36), and equation (2.43), the expansion shown to the second-order is now

presented.

$$
\begin{aligned}
f(\boldsymbol{\eta}+\mathbf{0}) &= f(\mathbf{0}) + \boldsymbol{\eta}\cdot\nabla f(\mathbf{0}) + \frac{1}{2}(\boldsymbol{\eta}\cdot\nabla)^2 f(\mathbf{0}) + \ldots \\
&= \log\left(A^{\mathcal{C}_1}(Z)\right) + \frac{1}{\sigma_\eta^2}\sum_{p=1}^{n}\frac{\left[B^{\mathcal{C}_1}(Z,p,1) - B^{\mathcal{C}_1}(Z,p,0)\right]}{A^{\mathcal{C}_1}(Z)}\,\eta_p \\
&\quad + \frac{1}{2\sigma_\eta^4}\sum_{v=1,v\neq k}^{n}\left[1 - \frac{\left(B^{\mathcal{C}_1}(Z,v,1) - B^{\mathcal{C}_1}(Z,v,0)\right)^2}{\left(A^{\mathcal{C}_1}(Z)\right)^2}\right]\eta_v^2 \\
&\quad + \frac{1}{\sigma_\eta^4}\sum_{\substack{p=1\\ p\neq q, p<q\leq n,\ p,q\neq k}}^{n}\sum_{q=1}^{n} F\,\eta_p\,\eta_q + \ldots
\end{aligned}
\tag{2.44}
$$

Expression for $g(\boldsymbol{\eta})$

The expression for $g(\boldsymbol{\eta})$ can be found by noting some simple changes in the derivations of the terms of $f(\boldsymbol{\eta})$ given in (2.31), (2.36) and (2.43). The difference between the two functions is the subset over which the summation of the pseudo-probabilities is carried out. This change is nicely handled by the weight enumeration functions defined earlier. Therefore, the resulting expression for $g(\boldsymbol{\eta})$ is,

$$
\begin{aligned}
g(\boldsymbol{\eta}+\mathbf{0}) &= g(\mathbf{0}) + \boldsymbol{\eta}\cdot\nabla g(\mathbf{0}) + \frac{1}{2}(\boldsymbol{\eta}\cdot\nabla)^2 g(\mathbf{0}) + \ldots \\
&= \log\left(A^{\mathcal{C}_0}(Z)\right) + \frac{1}{\sigma_\eta^2}\sum_{p=1}^{n}\frac{\left[B^{\mathcal{C}_0}(Z,p,1) - B^{\mathcal{C}_0}(Z,p,0)\right]}{A^{\mathcal{C}_0}(Z)}\,\eta_p \\
&\quad + \frac{1}{2\sigma_\eta^4}\sum_{v=1,v\neq k}^{n}\left[1 - \frac{\left(B^{\mathcal{C}_0}(Z,v,1) - B^{\mathcal{C}_0}(Z,v,0)\right)^2}{\left(A^{\mathcal{C}_0}(Z)\right)^2}\right]\eta_v^2 \\
&\quad + \frac{1}{\sigma_\eta^4}\sum_{\substack{p=1\\ p\neq q, p<q\leq n,\ p,q\neq k}}^{n}\sum_{q=1}^{n} G\,\eta_p\,\eta_q + \ldots,
\end{aligned}
\tag{2.45}
$$

where, by changing the subset for the weight enumeration functions in the expression of F in (2.42), G is defined to be

$$G = \frac{B^{\mathcal{C}_0}(Z, p \oplus q, 1) - B^{\mathcal{C}_0}(Z, p \oplus q, 0)}{A^{\mathcal{C}_0}(Z)} - \frac{\Big(B^{\mathcal{C}_0}(Z, p, 1) - B^{\mathcal{C}_0}(Z, p, 0)\Big)\Big(B^{\mathcal{C}_0}(Z, q, 1) - B^{\mathcal{C}_0}(Z, q, 0)\Big)}{\Big(A^{\mathcal{C}_0}(Z)\Big)^2} \tag{2.46}$$

2.3.5 Complete Taylor Series Expansion Expression

Substituting the expressions of (2.44) and (2.45) into (2.21), the Taylor series expansion is obtained to the second-order.

$$\begin{aligned}
LLR(m_k) = H(\boldsymbol{\eta}) &= f(\boldsymbol{\eta}) - g(\boldsymbol{\eta}) \\
&= \log \frac{\big(A^{\mathcal{C}_1}(Z)\big)}{\big(A^{\mathcal{C}_0}(Z)\big)} \\
&\quad + \frac{1}{\sigma_\eta^2} \sum_{p=1}^{n} \frac{\big[B^{\mathcal{C}_1}(Z, p, 1) - B^{\mathcal{C}_1}(Z, p, 0)\big]}{A^{\mathcal{C}_1}(Z)}\, \eta_p \\
&\quad - \frac{1}{\sigma_\eta^2} \sum_{p=1}^{n} \frac{\big[B^{\mathcal{C}_0}(Z, p, 1) - B^{\mathcal{C}_0}(Z, p, 0)\big]}{A^{\mathcal{C}_0}(Z)}\, \eta_p \\
&\quad + \frac{1}{2\sigma_\eta^4} \sum_{v=1, v\neq k}^{n} \left[1 - \frac{\Big(B^{\mathcal{C}_1}(Z, v, 1) - B^{\mathcal{C}_1}(Z, v, 0)\Big)^2}{\Big(A^{\mathcal{C}_1}(Z)\Big)^2}\right] \eta_v^2 \\
&\quad - \frac{1}{2\sigma_\eta^4} \sum_{v=1, v\neq k}^{n} \left[1 - \frac{\Big(B^{\mathcal{C}_0}(Z, v, 1) - B^{\mathcal{C}_0}(Z, v, 0)\Big)^2}{\Big(A^{\mathcal{C}_0}(Z)\Big)^2}\right] \eta_v^2 \\
&\quad + \frac{1}{\sigma_\eta^4} \sum_{\substack{p=1}}^{n} \sum_{\substack{q=1 \\ p\neq q, p<q\leq n,\ p,q\neq k}}^{n} F\, \eta_p\, \eta_q - \frac{1}{\sigma_\eta^4} \sum_{\substack{p=1}}^{n} \sum_{\substack{q=1 \\ p\neq q, p<q\leq n,\ p,q\neq k}}^{n} G\, \eta_p\, \eta_q + \ldots
\end{aligned}$$

$$
\begin{aligned}
&= \log \frac{\left(A^{\mathcal{C}_1}(Z)\right)}{\left(A^{\mathcal{C}_0}(Z)\right)} \\
&\quad + \frac{1}{\sigma_\eta^2} \sum_{p=1}^{n} \Bigg[\frac{\left[B^{\mathcal{C}_1}(Z,p,1) - B^{\mathcal{C}_1}(Z,p,0)\right]}{A^{\mathcal{C}_1}(Z)} \\
&\quad - \frac{\left[B^{\mathcal{C}_0}(Z,p,1) - B^{\mathcal{C}_0}(Z,p,0)\right]}{A^{\mathcal{C}_0}(Z)} \Bigg] \eta_p \\
&\quad + \frac{1}{2\sigma_\eta^4} \sum_{v=1, v\neq k}^{n} \Bigg[\Bigg[1 - \frac{\left(B^{\mathcal{C}_1}(Z,v,1) - B^{\mathcal{C}_1}(Z,v,0)\right)^2}{\left(A^{\mathcal{C}_1}(Z)\right)^2} \Bigg] \\
&\quad - \Bigg[1 - \frac{\left(B^{\mathcal{C}_0}(Z,v,1) - B^{\mathcal{C}_0}(Z,v,0)\right)^2}{\left(A^{\mathcal{C}_0}(Z)\right)^2} \Bigg] \Bigg] \eta_v^2 \\
&\quad + \frac{1}{\sigma_\eta^4} \sum_{\substack{p=1 \\ p\neq q, p<q\leq n,\ p,q\neq k}}^{n} \sum_{q=1}^{n} \left(F - G\right) \eta_p \eta_q + \ldots
\end{aligned} \tag{2.47}
$$

$$
\begin{aligned}
&= K_0 + \frac{1}{\sigma_\eta^2} \Bigg(\sum_{p=1}^{n} K_1(p) \eta_p \Bigg) + \left(\frac{1}{\sigma_\eta^2}\right)^2 \Bigg(\sum_{v=1, v\neq k}^{n} K_2(v) \eta_v^2 \\
&\quad + \sum_{\substack{p=1 \\ p\neq q, p<q\leq n,\ p,q\neq k}}^{n} \sum_{q=1}^{n} K_3(p,q) \eta_p \eta_q \Bigg) + \ldots,
\end{aligned} \tag{2.48}
$$

where, K_0, K_1, K_2, and K_3 are constant terms dependent upon the weight distribution of the code. Recall that η_i, for $i = 1, 2, \ldots, n$, are Gaussian random variables with a mean of zero and variance of $\sigma_\eta^2 = \frac{N_0}{2}$.

2.3.6 The Gaussian Approximation

Examining the terms of the resulting expression, the Gaussian approximation can be shown. K_0 is a constant term and simply shifts the resulting distribution. The first-order term of (2.47) is composed of the summation of Gaussian random variables and therefore, this term is Gaussian distributed.

The distribution of higher-order terms is not easily observable. However, it can be

shown that the sum of the higher-order product terms are asymptotically Gaussian distributed using the results of Hoeffding [9]. Hoeffding assumes i.i.d. random variables and states theorems which are applicable to functions symmetric in their arguments. A symmetric function is one where interchanging the arguments does not change the expression of the function [10]. This means that the results of [9] do not make any assumptions on the underlying distribution to be Gaussian. Indeed, it turns out that in many cases (under a set of mild conditions), even the assumption of independence can be removed [11–15]. This implies that Gaussian nature of the probability distribution of the LLR will be valid for a much wider class of additive noise, not necessarily i.i.d. Gaussian.

In the following discussion, a generalization of Theorem 7.1 of [9] will be presented for the weighted sum of symmetric functions, with non-repeating arguments. Note that the repetition of arguments is permitted in the Taylor series expansion, and subsequently, another theorem will generalize the results to the weighted sum of all permutations. Conditions will be presented which show that if the weights of the symmetric functions are of similar orders of magnitude, then the sum will be Gaussian distributed. Theorem 7.1 of [9] is applicable to multivariate situations as well, however, for the purposes of this thesis, only the univariate case is considered. By properly defining the symmetric functions, and properly including the coefficients of the terms, this theorem can be useful for the Taylor series expansion. It will be shown that the asymptotic Gaussian distributions can be established for each of sum of terms of $k = 2, 3, \ldots$ degrees. With the Gaussian nature determined for the expressions of different degrees, using the well-known theorem [16, Theorem 17a] that the sum of Gaussian random variables produces another Gaussian random variable, the Gaussian distribution for the LLR is established.

The modified generalization of Theorem 7.1 is presented below, along with a proof based upon that found in [9]. Some notation needs to be defined before proceeding, following from the notation of [9].

Let $X_1, \ldots, X_n$ be n independent, identically distributed random variables. Let $x_1, \ldots, x_k$ be arbitrary fixed values or samples. Define

$$\Phi(x_1, \ldots, x_k), \quad k \leq n \tag{2.49}$$

to be a real-valued function symmetric in its k arguments and which does not involve n. Define

$$\theta = E\Big\{\Phi(X_1, \ldots, X_k)\Big\}. \tag{2.50}$$

Let

$$\Phi_c(x_1, \ldots, x_c) = E\Big\{\Phi(x_1, \ldots, x_c, X_{c+1}, \ldots, X_k)\Big\}, \qquad c = 1, \ldots, k, \tag{2.51}$$

where the expected value is taken with respect to the random variables $X_{c+1}, \ldots, X_k$, holding $x_1, \ldots, x_c$ fixed. Then,

$$E\Big\{\Phi_c(X_1, \ldots, X_c)\Big\} = \theta, \quad c = 1, \ldots, k. \tag{2.52}$$

Define

$$\Psi(x_1, \ldots, x_k) = \Phi(x_1, \ldots, x_k) - \theta \tag{2.53}$$

$$\Psi_c(x_1, \ldots, x_c) = \Phi_c(x_1, \ldots, x_c) - \theta, \quad c = 1, \ldots, k, \tag{2.54}$$

and therefore, it follows that

$$E\Big\{\Psi_c(x_1,\ldots,x_c)\Big\} = 0. \tag{2.55}$$

Suppose that the variance of $\Psi_c(X_1,\ldots,X_c)$ exists, and let

$$\zeta_0 = 0, \qquad \zeta_c = E\Big\{\Psi_c^2(X_1,\ldots,X_c)\Big\}, \quad c = 1,\ldots,k. \tag{2.56}$$

Therefore,

$$\zeta_c = E\Big\{\Phi_c^2(X_1,\ldots,X_c)\Big\} - \theta^2. \tag{2.57}$$

As well, by Hoeffding [9, pp. 299, (5.12)], if $(\alpha_1,\ldots,\alpha_k)$ and $(\beta_1,\ldots,\beta_k)$ are two sets of different integers, such that $1 \le \alpha_i, \beta_i \le n,\ i = 1,\ldots,k$, and c is the number of integers common to the two sets, by the symmetry of Ψ,

$$E\Big\{\Psi(X_{\alpha_1},\ldots,X_{\alpha_k})\Psi(X_{\beta_1},\ldots,X_{\beta_k})\Big\} = \zeta_c. \tag{2.58}$$

In applying this theorem to the Taylor series expansion expressions, care must be taken in defining the functions $\Phi(X_1,\ldots,X_k)$ for degree k. Here, the symmetric function will be defined as the product of the arguments $X_1,\ldots,X_k$, or $X_1X_2\ldots X_k$. The derivative coefficients found in the Taylor series will be defined in the function U of the theorem below.

Theorem: Let $X_1,\ldots,X_n$ be n independent, identically distributed random variables. Let

$$\Phi(x_1,\ldots,x_k), \quad k \le n \tag{2.59}$$

be a real-valued function symmetric in its k arguments, x_α, and which does not involve

n. Define

$$U = \binom{n}{k}^{-1} \sum{}' a_{\alpha_1,\ldots,\alpha_k} \Phi(X_{\alpha_1}, \ldots, X_{\alpha_k}), \tag{2.60}$$

where the summation $\sum'$ *is over all subscripts such that* $1 \leq \alpha_1 < \ldots < \alpha_k \leq n$, *and* $a_{\alpha_1,\ldots,\alpha_k}$ *are real-valued coefficients. Then, if the expected values*

$$\theta = E\Big\{\Phi(X_{\alpha_1}, \ldots, X_{\alpha_k})\Big\} \tag{2.61}$$

and

$$E\Big\{\Phi(X_{\alpha_1}, \ldots, X_{\alpha_k})\Big\}^2, \tag{2.62}$$

exist, the distribution function of $\sqrt{n}(U - \theta')$, *where*

$$\theta' = \binom{n}{k}^{-1} \sum{}' a_{\alpha_1,\ldots,\alpha_k} \theta \tag{2.63}$$

tends, as $n \to \infty$, *to the normal distribution function with mean* 0 *and variance* $k^2 C_1 \zeta_1$, *where* ζ_1 *is defined by (2.57), and* C_1 *is a coefficient based upon the normalized sum of squared coefficients* $a_{\alpha_1,\ldots,\alpha_k}$.

The proof of the above modified theorem follows by some generalization of Theorem 7.1 of [9, pp. 307]. Carrying through with the steps, it will be shown that the above theorem holds, with some restrictions on the values of the coefficients.

Proof: Define a new random variable Y, which is the sum of n independent random variables, i.e.

$$Y = \frac{k}{\sqrt{n}} \sum_{\beta=1}^{n} \binom{n-1}{k-1}^{-1} \sum\nolimits_{(\beta)} a_{\beta_1,\ldots,\beta_k} \Psi_1(X_\beta) \tag{2.64}$$

where, the summation $\sum_{(\beta)}$ is defined over all subscripts such that $1 \leq \beta_1 < \beta_2 < \ldots < \beta_k \leq n$ for those functions $\Phi(X_{\beta_1}, \ldots, X_{\beta_k})$ that contain X_β, $\beta = 1, \ldots, n$, and $\Psi_1(x)$ is defined by (2.54). This differs from the definition of Hoeffding due to the inclusion of coefficients. The variables of the sum are made independent by taking the expectation over all other random variables of the function with the exception of the argument of $\Psi_1(x)$, X_β. The summation of the coefficients represents the inclusion of the appropriate coefficients for each $\Phi(X_1, \ldots, X_k)$ included in the averaging of $\Psi_1(x_\beta)$. Using results of Le Cam [17], who restates Lévy's version of the Central Limit Theorem [18], for the sum of independent variables, the sufficient conditions for normality of Y is presented. This theorem contains two such conditions which must be satisfied to approach normality and are [17, Theorem 2]:

1. *Each summand that is not negligible compared to the dispersion of the entire sum has a distribution close to Gaussian.*

2. *The maximum of the absolute value of the negligible summands is itself negligible compared to the dispersion of the sum.*

Therefore, as long as the summands are not too large or, if large, possess a Gaussian-like distribution, the Central Limit Theorem can be applied. This can be applied to the definition of Y above for coefficients of similar magnitude. Using the above results, the distribution of Y tends to the normal distribution with mean 0, and variance $k^2 \left\{\sum_{(\beta)} a_{\beta_1,\ldots,\beta_k}\right\}^2 \zeta_1$.

Using Lemma 7.1 from [9, pp. 305], it will be shown that

$$\begin{aligned} Z &= \sqrt{n}(U - \theta') \\ &= \sqrt{n}\binom{n}{k}^{-1} \sum{}' a_{\alpha_1,\ldots,\alpha_k} \Psi(X_{\alpha_1}, \ldots, X_{\alpha_k}) \end{aligned} \tag{2.65}$$

has the same limiting distribution as Y. By the lemma, it is sufficient to show that

$$\lim_{n\to\infty} E\big(Z - Y\big)^2 = 0. \tag{2.66}$$

Proving (2.66) requires expansion of the square as

$$E\big\{Z - Y\big\}^2 = E\{Z\}^2 + E\{Y\}^2 - 2E\{ZY\}, \tag{2.67}$$

and by formulating each of the expressions of (2.67), separately.

Beginning with $E\{Z\}^2$,

$$\begin{aligned} E\{Z\}^2 &= nE\big\{U - \theta'\big\}^2 = n\sigma_U^2 \\ &= n\binom{n}{k}^{-2} E\Big\{\sum{}' a_{\alpha_1,\ldots,\alpha_k}\Psi(X_{\alpha_1},\ldots,X_{\alpha_k})\Big\}^2 \\ &= n\binom{n}{k}^{-2}\sum_{c=1}^{k}\sum{}^{(c)} a_{\alpha_1,\ldots,\alpha_k}a_{\beta_1,\ldots,\beta_k} \\ &\qquad\qquad E\Big\{\Psi(X_{\alpha_1},\ldots,X_{\alpha_k})\Psi(X_{\beta_1},\ldots,X_{\beta_k})\Big\} \end{aligned} \tag{2.68}$$

where the summation $\sum^{(c)}$, as defined by Hoeffding, represents the summation over all subscripts such that

$$1 \le \alpha_1 < \alpha_2 < \ldots < \alpha_k \le n, \;\; 1 \le \beta_1 < \beta_2 < \ldots < \beta_k \le n. \tag{2.69}$$

By (2.58), each term $E\Big\{\Psi(X_{\alpha_1},\ldots,X_{\alpha_k})\Psi(X_{\beta_1},\ldots,X_{\beta_k})\Big\}$ is equal to ζ_c. Hoeffding states that the number of terms in $\sum^{(c)}$ is

$$\binom{k}{c}\binom{n-k}{k-c}\binom{n}{k} \tag{2.70}$$

However, due to the coefficients $a_{\alpha_1,\ldots,\alpha_k}$, this is a scaled sum and results in the expression

$$\sum\nolimits^{(c)} a_{\alpha_1,\ldots,\alpha_k} a_{\beta_1,\ldots,\beta_k} = C_c \binom{k}{c}\binom{n-k}{k-c}\binom{n}{k} \tag{2.71}$$

where C_c is an appropriate scaling factor for equality to hold. Using (2.58), $\zeta_0 = 0$, and (2.71), in (2.68), $E\{Z\}^2$ becomes [9, pp. 308, (7.9)],

$$E\{Z\}^2 = k^2 C_1 \zeta_1 + O(n^{-1}) \tag{2.72}$$

Typically, it is desired that the variance of U to be normalized to 1 and therefore the constraint, as $n \to \infty$,

$$k^2 C_1 \zeta_1 = 1 \tag{2.73}$$

is imposed.

Continuing with the formulation of expressions,

$$\begin{aligned} E\{Y\}^2 &= \frac{k^2}{n} \sum_{\beta=1}^{n} \binom{n-1}{k-1}^{-2} \Big[\sum\nolimits_{(\beta)} a_{\beta_1,\ldots,\beta_k}\Big]^2 \zeta_1 \\ &= k^2 D_1 \zeta_1 \end{aligned} \tag{2.74}$$

where

$$\sum_{\beta=1}^{n} \Big[\sum\nolimits_{(\beta)} a_{\beta_1,\ldots,\beta_k}\Big]^2 = D_1 n \binom{n-1}{k-1}^2 \tag{2.75}$$

since there are $\binom{n-1}{k-1}^2$ terms in the square of the summation; D_1 is an appropriate constant for equality.

Finally, using (2.64) and (2.65),

$$\begin{aligned} E\{ZY\} = E\Big\{\sqrt{n}\binom{n}{k}^{-1}\sum{}'a_{\alpha_1,\ldots,\alpha_k}\Psi(X_{\alpha_1},\ldots,X_{\alpha_k})\cdot \\ \frac{k}{\sqrt{n}}\sum_{\beta=1}^{n}\binom{n-1}{k-1}^{-1}\sum\nolimits_{(\beta)}a_{\beta_1,\ldots,\beta_k}\Psi_1(X_\beta)\Big\} \\ = k\binom{n}{k}^{-1}\binom{n-1}{k-1}^{-1}\sum{}'a_{\alpha_1,\ldots,\alpha_k}\sum_{\beta=1}^{n}\sum\nolimits_{(\beta)}a_{\beta_1,\ldots,\beta_k} \\ E\Big\{\Psi(X_{\alpha_1},\ldots,X_{\alpha_k})\Psi_1(X_\beta)\Big\} \end{aligned} \tag{2.76}$$

where the summations, $\sum'$ and $\sum_{(\beta)}$, are as previously defined. Using conditions found in [9, pp. 308], the term

$$E\Big\{\Psi(X_{\alpha_1},\ldots,X_{\alpha_k})\Psi_1(X_\beta)\Big\} = \zeta_1 \tag{2.77}$$

if

$$\alpha_1 = \beta \text{ or } \alpha_2 = \beta \ldots \text{ or } \alpha_k = \beta, \tag{2.78}$$

and is 0 otherwise, due to cross multiplication of independent variables having zero mean. For some fixed β, the number of possible sets $\{\alpha_1,\ldots,\alpha_k\}$, such that $1 \le \alpha_1 < \ldots < \alpha_k \le n$ that satisfy (2.78) is $\binom{n-1}{k-1}$. As well, the number of coefficients in the summations $\sum_{\beta=1}^{n}\sum_{(\beta)}$ is $n\binom{n-1}{k-1}$. Defining B_1 as the scaling factor for the summation of coefficients,

$$\sum{}'a_{\alpha_1,\ldots,\alpha_k}\sum_{\beta=1}^{n}\sum\nolimits_{(\beta)}a_{\beta_1,\ldots,\beta_k} = B_1 n\binom{n-1}{k-1}^2 \tag{2.79}$$

where the conditions imposed in (2.78) determine which coefficients in (2.79) are summed.

Substituting (2.77) into (2.76),

$$\begin{aligned} E\{ZY\} &= k\binom{n}{k}^{-1}\binom{n-1}{k-1}^{-1} n\binom{n-1}{k-1}^{2} B_1\zeta_1 \\ &= k^2 B_1\zeta_1. \end{aligned} \tag{2.80}$$

Substituting (2.72), (2.74) and (2.80) into (2.67), the following is obtained.

$$E\{Z-Y\}^2 = k^2C_1\zeta_1 + O(n^{-1}) + k^2D_1\zeta_1 - 2k^2B_1\zeta_1 \tag{2.81}$$

and taking the limit as $n \to \infty$,

$$\begin{aligned} \lim_{n\to\infty} E\{Z-Y\}^2 &= k^2C_1\zeta_1 + k^2D_1\zeta_1 - 2k^2B_1\zeta_1 \\ &= k^2\zeta_1(C_1 + D_1 - 2B_1) \end{aligned} \tag{2.82}$$

results. For the two limiting distributions to be the same, the coefficients C_1, D_1 and B_1 must combine to yield 0. Note that if the derivative coefficients, $a_{\alpha_1,\ldots,\alpha_k}$, were unity, the situation encountered in [9] would apply, and the expression of (2.82) would be 0. It suffices to say that if

$$D_1 \approx C_1 \approx B_1, \tag{2.83}$$

then $E\{Z-Y\}^2 = 0$. Therefore, so long as the normalized sum of the coefficients is of similar order of magnitude, in the sense defined in (2.83), the distribution function of the U function approaches a Gaussian distribution. ■

The above conditions will be satisfied if the coefficient of a given degree in the Taylor series expansion of the LLR are of similar orders of magnitude. In practice, due to the combinatorial symmetries of linear codes, these coefficients are approximately equal,

satisfying the necessary conditions to obtain a Gaussian distribution.

The above theorem is applicable to the function U, where no repetition of the arguments is allowed in the functions. This differs from the Taylor series terms above since terms of multiplicity of arguments between 2 and k, the degree of the Taylor expression, occur often. Hoeffding addresses this issue with Theorem 7.3 of [9]. Using similar conditions to those discussed previously for Theorem 7.1 it can be shown that the distribution of the complete function including all permutations, i.e.

$$\theta(S) = \frac{1}{n^k} \sum_{\alpha_1=1}^{n} \cdots \sum_{\alpha_k=1}^{n} a_{\alpha_1,\ldots,\alpha_k} \Phi(X_{\alpha_1}, \ldots, X_{\alpha_k}), \tag{2.84}$$

has the same asymptotic normal distribution as U. Therefore, the results from Theorem 7.1 hold for $\theta(S)$ and therefore for the Taylor series expressions. Therefore, it has been shown that the sum of weighted terms of given degree possess an asymptotic Gaussian distribution and therefore the complete Taylor series expansion of the LLR is Gaussian distributed.

Theorem 7.1 can be applied successfully for degrees less than the blocklength of the code, n, however, as the degree of the term approaches the blocklength, the number of unique functions decreases. This leads to the function U above being composed of only a few summands and reduces the appropriateness of the theorem. Below, another observation about the derivative coefficients $a_{1,\ldots,k}$ can be useful to compensate for this shortcoming. Although the higher-order partial derivatives are difficult to express in a closed form, it can be shown, in Theorem 2.4, that these coefficients approach zero as the noise variance σ_η^2 becomes large in $f(\boldsymbol{\eta})$.

Theorem 2.4 *The higher-order partial derivative coefficients in the expansion of* $f(\boldsymbol{\eta})$ *approach* 0 *as the noise variance,* σ_η^2*, approaches infinity.*

Proof: The coefficients of the series are formed from partial derivatives and subsequently, $\boldsymbol{\eta} = \mathbf{0}$ is substituted. To prove this theorem, (2.22) containing $f(\boldsymbol{\eta})$ is rewritten as,

$$\begin{aligned} f(\boldsymbol{\eta}) &= \log\Big(\sum_{\boldsymbol{c}_i \,\epsilon\, \mathcal{C}_1} \exp\Big\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_i + \tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\Big\}\Big) \\ e^{f(\eta)} &= \sum_{\boldsymbol{c}_i \,\epsilon\, \mathcal{C}_1} \exp\Big\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_i + \tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\Big\} \\ e^{f(\eta)} &= A. \end{aligned} \tag{2.85}$$

where, A is defined for the convenience of representation.

In this form, the higher-order partial derivatives are easier to calculate by the repeated differentiation of the exponential function. The term on the right hand side of equation (2.85) with $\boldsymbol{\eta} = \mathbf{0}$ is finite valued for block lengths which are not infinitely long. The exponentials are well behaved as the noise variance σ_η^2 increases (since the exponential decreases) and the summation yields a finite result. These arguments will be used for higher-order partial derivatives below. The proof will be carried out by following an inductive thought process to yield the form of the expression

$$D^k_{m_1,m_2,\ldots,m_n}(f) = \frac{\partial^k f(\boldsymbol{\eta})}{\partial\eta_1^{m_1}\partial\eta_2^{m_2}\ldots\partial\eta_n^{m_n}}, \tag{2.86}$$

where, $m_1 + m_2 + \ldots + m_n = k$. $D^k_{m_1,m_2,\ldots,m_n}(f)$ denotes the k^{th}-order partial derivative of the function f.

First-order: Implicitly differentiating (2.85) which respect to η_p, where p is a given bit

position, the first-order derivative is given by

$$\frac{\partial e^{f(\eta)}}{\partial \eta_p} = \frac{\partial}{\partial \eta_p}\Big(\sum_{\boldsymbol{c}_i \,\epsilon\, \mathcal{C}_1} \exp\Big\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_i + \tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\Big\}\Big)$$

$$e^f \frac{\partial f}{\partial \eta_p} = \frac{1}{\sigma_\eta^2}\sum_{\boldsymbol{c}_i \,\epsilon\, \mathcal{C}_1} c_{ip} \exp\Big\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_i + \tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\Big\} \tag{2.87}$$

$$\frac{\partial f}{\partial \eta_p} = e^{-f}\frac{\partial A}{\partial \eta_p} \tag{2.88}$$

$$D^1_{m_1,m_2,\ldots,m_n}(f) = \frac{1}{\sigma_\eta^2} Y_1, \tag{2.89}$$

where, $m_p = 1$ and $m_i = 0$, for $i = 1, \ldots, n$, $i \neq p$.

Substituting $\boldsymbol{\eta} = \mathbf{0}$, $D^1_{m_1,m_2,\ldots,m_n}(A)$ becomes,

$$\frac{1}{\sigma_\eta^2}\sum_{\boldsymbol{c}_i \,\epsilon\, \mathcal{C}_1} c_{ip} \exp\Big\{\frac{\tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\Big\}. \tag{2.90}$$

The first-order partial derivative in (2.88) is a finite number multiplied by $1/\sigma_\eta^2$ and divided by e^f. It was determined above that e^f is a finite number with increasing noise variance. The factors c_{ip} are either 1 or -1, meaning the result of dividing (2.90) by e^f is less than or equal to 1 (as they share similar components for the subset C_1). Therefore, the expression of (2.90) tends to 0 with increasing σ_η^2. This is seen in the expression of (2.48) with the multiplicative inverse noise variance factor. Therefore, for increasing noise variance, the first-order coefficients decrease.

Second-order: Implicitly differentiating (2.87) with respect to η_q yields the second-

order expression,

$$\frac{\partial}{\partial \eta_q}\big(e^{f(\eta)}\frac{\partial f}{\partial \eta_p}\big) = \frac{\partial}{\partial \eta_q}\big(\frac{1}{\sigma_\eta^2}\sum_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1} c_{ip}\exp\big\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_i + \tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\big\}\big)$$

$$e^f\frac{\partial^2 f}{\partial\eta_p\partial\eta_q} + e^f\frac{\partial f}{\partial\eta_p}\frac{\partial f}{\partial\eta_q} = \frac{1}{\sigma_\eta^4}\sum_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1} c_{ip}c_{iq}\exp\big\{\frac{\boldsymbol{\eta}\cdot\boldsymbol{c}_i + \tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\big\} \tag{2.91}$$

$$\frac{\partial^2 f}{\partial\eta_p\partial\eta_q} = e^{-f}\frac{\partial^2 A}{\partial\eta_p\partial\eta_q} - \frac{\partial f}{\partial\eta_p}\frac{\partial f}{\partial\eta_q} \tag{2.92}$$

$$D^2_{m_1,m_2,\ldots,m_n}(f) = \frac{1}{\sigma_\eta^4}Y_2. \tag{2.93}$$

Substituting $\boldsymbol{\eta} = \mathbf{0}$ into $D^2_{m_1,m_2,\ldots,m_n}(A)$ yields,

$$\frac{1}{\sigma_\eta^4}\sum_{\boldsymbol{c}_i\,\epsilon\,\mathcal{C}_1} c_{ip}c_{iq}\exp\big\{\frac{\tilde{\boldsymbol{c}}\cdot\boldsymbol{c}_i}{\sigma_\eta^2}\big\}. \tag{2.94}$$

The second-order partial derivative in (2.92) is comprised of the expression of (2.94) and the terms of the first-order partial derivatives. Again, the above expression in (2.94) decreases towards zero due to the $1/\sigma_\eta^4$ multiplicative factor and since the summation is again finite (≤ 1) and decreasing with increasing noise variance. Therefore, the term $D^2_{m_1,m_2,\ldots,m_n}(A)$ tends to zero. It was determined above that the first-order partial derivatives tend to zero as well with increasing noise variance. A common multiplicative term of $1/\sigma_\eta^4$ can be factored and the rate of decrease of the second-order coefficients is greater than that of the first-order coefficients. This appears in (2.48) for the second-order term.

It can easily be verified that the general second-order expression takes the form of,

$$\frac{\partial^2 f}{\partial^{m_p}\eta_p\partial^{m_q}\eta_q} = e^{-f}\frac{\partial^2 A}{\partial\eta_p^{m_p}\partial\eta_q^{m_q}} - \big(\frac{\partial f}{\partial\eta_p}\big)^{m_p}\big(\frac{\partial f}{\partial\eta_q}\big)^{m_q} \tag{2.95}$$

where, $m_p + m_q = 2$.

The coefficients have multiplicative factors of $1/\sigma_\eta^2$ to the degree for which the coefficient is sought. It can be shown that this is the case for third order terms, however, a closed expression does not exist and so it is not presented here. Given a degree k, the resulting expression is a function of the k^{th} partial derivative of A and the partial derivatives of lower-orders $k-1, k-2, \ldots, 2, 1$. This was the situation encountered for the second and third-order expressions. Using this observations, the expression for the $k+1$-order expression can be deduced.

Assume that the k-order partial derivative coefficients approach zero as the noise variance increases. Assume they take the form below based upon the lower-order partial derivatives seen above.

$$D^k_{m_1,m_2,\ldots,m_n}(f) = \frac{\partial^k f}{\partial\eta_1^{m_1}\partial\eta_2^{m_2}\ldots\partial\eta_n^{m_n}} = \frac{1}{\sigma_\eta^{2k}}Y_k \tag{2.96}$$

$$= e^{-f}\frac{\partial^k A}{\partial\eta_1^{m_1}\partial\eta_2^{m_2}\ldots\partial\eta_n^{m_n}} - X_k \tag{2.97}$$

$$\tag{2.98}$$

where, $m_1 + m_2 + \ldots + m_n = k$ and Y_k is a constant expression comprised of lower-order coefficients expressions, and X_k is comprised of lower-order partial derivatives.

$\underline{k+1\text{-order:}}$ To form the $k+1$-order partial derivative, the expression of (2.97) is implicitly differentiated. It will then be shown that the $k+1$-order expression is comprised of lower-order partial derivatives and $k+1$-order partial derivative of A. To facilitate this, (2.97) is reorganized as

$$\frac{\partial^k f}{\partial\eta_1^{m_1}\partial\eta_2^{m_2}\ldots\partial\eta_n^{m_n}} = e^{-f}\frac{\partial^k A}{\partial\eta_1^{m_1}\partial\eta_2^{m_2}\ldots\partial\eta_n^{m_n}} - X_k$$

$$e^f\frac{\partial^k f}{\partial\eta_1^{m_1}\partial\eta_2^{m_2}\ldots\partial\eta_n^{m_n}} = \frac{\partial^k A}{\partial\eta_1^{m_1}\partial\eta_2^{m_2}\ldots\partial\eta_n^{m_n}} - e^f X_k. \tag{2.99}$$

Without loss of generality, it will be assumed that the $k+1^{st}$ derivative is taken with respect to η_1 for convenience, and therefore, $D^{k+1}_{m_1+1,m_2,\ldots,m_n}(f)$ is sought. Implicitly differentiating (2.99) with respect to η_1, and solving,

$$\begin{aligned}
\frac{\partial}{\partial \eta_1}\left(e^f \frac{\partial^k f}{\partial \eta_1^{m_1}\partial \eta_2^{m_2}\ldots\partial \eta_n^{m_n}}\right) &= \frac{\partial}{\partial \eta_1}\left(\frac{\partial^k A}{\partial \eta_1^{m_1}\partial \eta_2^{m_2}\ldots\partial \eta_n^{m_n}} - e^f X_k\right) \\
e^f \frac{\partial^{k+1} f}{\partial \eta_1^{m_1+1}\partial \eta_2^{m_2}\ldots\partial \eta_n^{m_n}} + e^f \frac{\partial^k f}{\partial \eta_1^{m_1}\partial \eta_2^{m_2}\ldots\partial \eta_n^{m_n}}\frac{\partial f}{\partial \eta_1} &= \frac{\partial^{k+1} A}{\partial \eta_1^{m_1+1}\partial \eta_2^{m_2}\ldots\partial \eta_n^{m_n}} \\
&\quad - \frac{\partial}{\partial \eta_1}\left(e^f X_k\right) \\
\frac{\partial^{k+1} f}{\partial \eta_1^{m_1+1}\partial \eta_2^{m_2}\ldots\partial \eta_n^{m_n}} &= e^{-f}\frac{\partial^{k+1} A}{\partial \eta_1^{m_1+1}\partial \eta_2^{m_2}\ldots\partial \eta_n^{m_n}} - e^{-f}\frac{\partial}{\partial \eta_1}\left(e^f X_k\right) \\
&\quad - \frac{\partial^k f}{\partial \eta_1^{m_1}\partial \eta_2^{m_2}\ldots\partial \eta_n^{m_n}}\frac{\partial f}{\partial \eta_1}
\end{aligned} \tag{2.100}$$

is obtained.

The expression on the right hand side of (2.100) is composed of the $k+1$-order partial derivative of A and other lower-order partial derivatives of the function $f(\boldsymbol{\eta})$ evaluated at $\boldsymbol{\eta} = \mathbf{0}$. The lower-order terms were assumed to tend towards zero with increasing noise variance σ_η^2. The first term of the expression contains a summations of exponentials which is finite and decreasing for increasing σ_η^2. The division by e^f normalizes the summation. A multiplicative factor of $1/\sigma_\eta^{2k+2}$ exists and diminishes this term quickly with increasing noise variance. Therefore, the $k+1$-order partial derivative coefficient tends to zero with increasing noise variance. ■

The above theorem shows that the coefficients for the expansion of $f(\boldsymbol{\eta})$ tend towards zero with increasing noise variance. This also holds for $g(\boldsymbol{\eta})$ as well, so that the combined coefficients of (2.48) also tend towards zero with increasing σ_η^2. Having established that the higher-order terms tend to zero with increasing noise variance and at a faster rate for higher-order terms, the lower-order terms remain to produce the Gaussian distribution.

This limits the effect of the terms with degree approaching the blocklength of the code. The lower-order terms are therefore asymptotically Gaussian and their sum in the series yields a Gaussian distribution for the LLR.

In the limit, as the noise variance values become quite large, the first-order terms remain and the approximation below is obtained

$$\begin{aligned} LLR(m_k) = H(\boldsymbol{\eta}) &= K_0 + \frac{1}{\sigma_\eta^2}\left(\sum_{p=1}^{n} K_1(p)\eta_p\right) \\ &= \log\frac{\left(A^{\mathcal{C}_1}(Z)\right)}{\left(A^{\mathcal{C}_0}(Z)\right)} \\ &\quad + \frac{1}{\sigma_\eta^2}\sum_{p=1}^{n}\left[\frac{\left[B^{\mathcal{C}_1}(Z,p,1) - B^{\mathcal{C}_1}(Z,p,0)\right]}{A^{\mathcal{C}_1}(Z)}\right. \\ &\quad \left. - \frac{\left[B^{\mathcal{C}_0}(Z,p,1) - B^{\mathcal{C}_0}(Z,p,0)\right]}{A^{\mathcal{C}_0}(Z)}\right]\eta_p \end{aligned} \tag{2.101}$$

The asymptotic approximation is comprised of first-order expressions in terms of the components of $\boldsymbol{\eta}$. Each are Gaussian distributed and the sum of scaled Gaussian random variables results in a Gaussian distributed variable. At high SNR, it is expected that the approximation will not be valid since the higher-order terms are not negligible. The usefulness of this approximation will be seen in the following chapters and with the simulation results in chapter 6.

2.4 LLR Approximation Comparison to a BPSK System

The LLR value is compared to the threshold value of 0 to determine whether a 0 or a 1 was transmitted. It was shown that the LLR can be approximated as a Gaussian random variable. This variable would have a certain mean and variance which can be measured by taking a number of samples for a given bit position. Alternately, the formula of (2.47)

can be used directly to calculate the mean and variance of the LLR. However, for the purpose of this thesis, simulation samples will be used. Figure 2.4 shows the distribution of the LLR to the right of the 0 threshold. Since the zero codeword was assumed to be transmitted, the LLR is centered about a negative mean. It was assumed throughout this

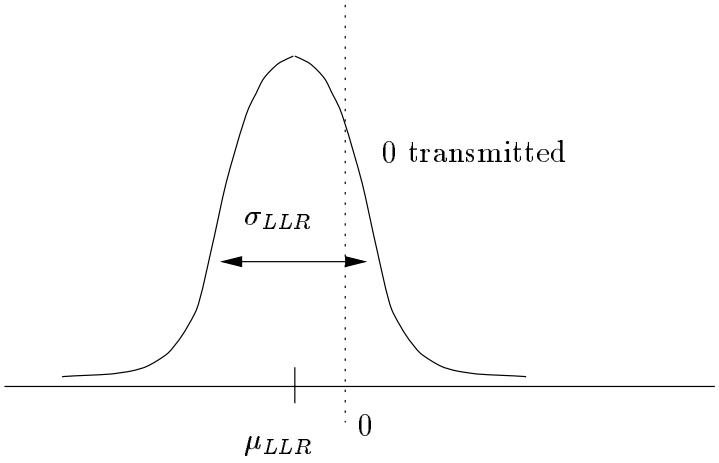

Figure 2.4: LLR Distribution Approximation

analysis that a 0 has been transmitted in bit position k of the codeword. Theorem 2.1 states that any codeword can be chosen without affecting the probability distribution of the LLR, provided the bit k remains unchanged. What if the bit value of 1 was assumed for transmission? How would the distribution differ from that of a transmitted 0? The following theorem addresses this matter.

Theorem 2.5 *The probability distribution of* $LLR(m_k)$ *for* $m_k = 0$ *and* $m_k = 1$ *are reflections of one another through the decision threshold of* 0 *(i.e., the origin).*

Proof: From the expression of (2.48), it can be seen that the mean of the LLR is comprised of K_0 and since η_i's have means of zero, those terms with even powers of η_i's, e.g. η_i^2. Assume the all-zero codeword $\mathbf{0}$ is transmitted. Then, the expression for K_0 can be

written as

$$K_0(-\mathbf{1}) = \log\left(\frac{\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\left\{\frac{1}{\sigma_\eta^2}\left(-\mathbf{1}\cdot\boldsymbol{c}_i\right)\right\}}{\sum_{\boldsymbol{c}_j \in \mathcal{C}_0} \exp\left\{\frac{1}{\sigma_\eta^2}\left(-\mathbf{1}\cdot\boldsymbol{c}_j\right)\right\}}\right) \tag{2.102}$$

Given the sub-code $\mathcal{C}_0$, and by adding the all-one codeword to all of the codewords in the sub-code, the coset $\mathcal{C}_1$ is obtained and more importantly, the Hamming distance of the all-zero codeword to all the codewords with a zero in a bit position k is equal to the Hamming distance profile of the all-one codeword to all codewords which have a one in that position.

Therefore, if the all-one codeword $\mathbf{1}$ is transmitted instead, and noting the change in the subset with the addition of the all-one codeword to all the codewords of $\mathcal{C}_0$, the mean of the LLR becomes

$$\begin{aligned} K_0(\mathbf{1}) &= \log\left(\frac{\sum_{\boldsymbol{c}_i \in \mathcal{C}_1} \exp\left\{\frac{1}{\sigma_\eta^2}\left(\mathbf{1}\cdot\boldsymbol{c}_i\right)\right\}}{\sum_{\boldsymbol{c}_j \in \mathcal{C}_0} \exp\left\{\frac{1}{\sigma_\eta^2}\left(\mathbf{1}\cdot\boldsymbol{c}_j\right)\right\}}\right) \\ &= \log\left(\frac{\sum_{\boldsymbol{c}_i \in \mathcal{C}_0} \exp\left\{\frac{1}{\sigma_\eta^2}\left(-\mathbf{1}\cdot\boldsymbol{c}_i\right)\right\}}{\sum_{\boldsymbol{c}_j \in \mathcal{C}_1} \exp\left\{\frac{1}{\sigma_\eta^2}\left(-\mathbf{1}\cdot\boldsymbol{c}_j\right)\right\}}\right) \\ &= -\log\left(\frac{\sum_{\boldsymbol{c}_j \in \mathcal{C}_1} \exp\left\{\frac{1}{\sigma_\eta^2}\left(-\mathbf{1}\cdot\boldsymbol{c}_j\right)\right\}}{\sum_{\boldsymbol{c}_i \in \mathcal{C}_0} \exp\left\{\frac{1}{\sigma_\eta^2}\left(-\mathbf{1}\cdot\boldsymbol{c}_i\right)\right\}}\right) \\ &= -K_0(-\mathbf{1}) \end{aligned} \tag{2.103}$$

The other terms with even powers of η_i's have coefficients, which with a change in the transmitted codeword, realize a sign change in a similar manner due to the distance profile of the code. Therefore, the transmission of a 1 in a given bit position is seen to produce a mirror image of the distribution of the transmitted 0 through the decision threshold of 0. ■

Therefore, if the all-one codeword was assumed for transmission so that the bit in position k was a 1, by Theorem 2.5, it would be expected that the distribution would be mirrored in the decision threshold of 0. The distribution would appear as the dotted-line form depicted in figure 2.5. It can be seen that the crossover point for the distributions would be the decision threshold of 0. The mean of the LLR distribution for a transmitted 1 is positive.

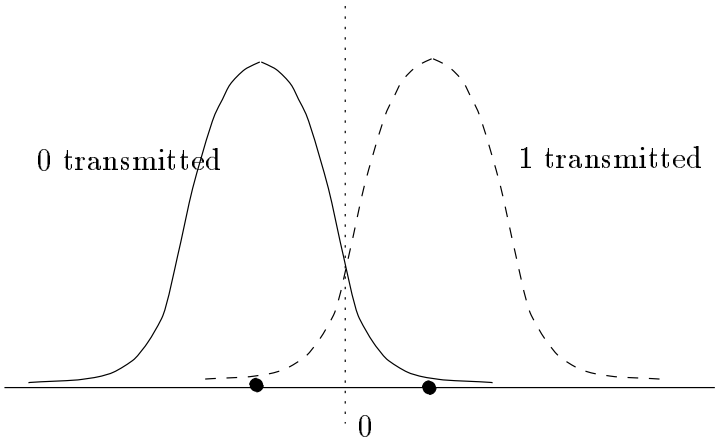

Figure 2.5: LLR Distributions for Transmitted 0 and 1

This diagram appears to be equivalent to that of the conditional distributions of figure 2.2 for a BPSK system. The only difference appears in the different means and variances for the distributions.

Since the LLR is approximately Gaussian with a measurable mean and variance, and its decision threshold is defined to be 0, the soft-output decoding of binary linear codes using LLR values can be modeled using a BPSK system. Therefore, it is conceivable that the error probability for the bit in position k of the codeword could be approximated by using the bit error probability expression for a BPSK system in (2.17).

2.5 Coding Gain

With the establishment of the model, a new coding gain can be defined. The coding gain is defined as the amount of improvement in performance that is obtained by coding information over not coding the information. Since the soft-output decoding of a binary linear block code has been shown to equivalent to a BPSK system, the coding gain can be measured against a BPSK-modulated system for the uncoded words with the energy per bit, E_b, set appropriately.

By scaling the LLR values such that the mean of them is equal to the set E_b value, the ratio of the variance of the noise from the channel to the variance of the LLR values can be calculated and is representative of the gain obtained from coding and soft-output decoding. Remember that the variance of the noise samples was used in the determination of the LLR values from (2.1) above and so the gain is from the code structure.

2.6 Chapter Summary

This chapter has presented the method by which the LLR of a given bit of a transmitted codeword is calculated. As mentioned in chapter 1, the BCJR method could have been used, but was not used for this thesis as it would have been more complicated to implement and analyze. Before presenting the methodology used to obtain the Gaussian approximation for the LLR distribution, a review of a BPSK modulation system was presented as a basis for the modeling of soft-output decoding of binary linear codes.

The approximation is derived from the Taylor series expansion of the LLR function defined in (2.1), in powers of Gaussian random variables. By noting that for high noise variance values, higher-order terms become negligible and do not greatly affect the distribution of the LLR, the remaining grouped terms of the series produce an approximately-Gaussian distribution. The requirement that the noise variance be high for the approxi-

mation to hold true corresponds with low SNR values. It is in this range of SNR values which an approximation would be useful in practice. Based on the Gaussian approximation, the modeling of the LLR (from soft-output decoding) using a BPSK modulated system was presented. A new definition for coding gain was presented.

In the following chapters, an in-depth study of the ratio of the mean of the LLR, M, to the standard deviation of the LLR, V (as used in the Q-function) is presented. The mean and variance of the LLR is calculated using samples from simulations. Expressions from this chapter could have been used to calculate these values using knowledge of the weight distribution of the code, but the simulation method was chosen. With the study of the ratio, it is possible to discuss the merits of the approximation, later in chapter 5.

Chapter 3

Mean and Variance Estimators of the LLR

In the last chapter, it was proposed that the statistical behaviour of the log likelihood ratio can be approximated by a Gaussian distribution. The modeling of the LLR values using a Binary Phase Shift Keying (BPSK) system was described. This model makes possible the use of the statistical properties of the LLR in obtaining an approximation of the bit error performance of a code. In particular, the ratio of the mean to standard deviation is used with the Q-function defined earlier in section 2.2, in the evaluation of the bit error probability of a BPSK system.

In practice, the LLR values will be computed numerically through Additive White Gaussian Noise (AWGN) channel simulations. Then, the mean and variance of the LLR can be measured using sample estimators. The question then arises as to how many samples are needed to obtain numerically stable results. Before this question can be addressed, the sample estimators are further examined in this chapter.

This chapter will present the sample estimators, and their distributions will be dis-

cussed. As well, the independence of the sample estimators will be shown, attributable to the samples being independent and normally distributed. By obtaining the sample mean and sample variance, the ratio of the two quantities can be found for use in the Q-function. Using their distributions and manipulating the ratio of the two estimators, a probability density function for the ratio will be formulated. With these facts, the precision of the approximation and an idea of the number of samples required will be presented in the next two chapters to follow.

3.1 Sample Estimators of Mean and Variance

The common estimators for the mean and variance of a set of samples are simple and well-known. Suppose there are N independent and identically distributed (i.i.d.) Gaussian samples x_i , $i = 1, 2, \ldots, N$, with mean μ and variance σ^2. Then, the sample estimators for the mean and variance are as shown in equations (3.1) and (3.2), respectively. Since the mean and variance estimators are functions of samples of Gaussian random variables, these estimators can be viewed as random variables. Therefore, the mean estimator will be referred to by the random variable M, and likewise, the variance will be referred to by the random variable V^2.

$$\text{Mean } M,\ \overline{x} = \frac{1}{N}\sum_{i=0}^{N} x_i \tag{3.1}$$

$$\text{Variance } V^2,\ \sigma^2 = \frac{1}{N}\sum_{i=0}^{N} (x_i - \overline{x})^2 \tag{3.2}$$

With any estimator, the more information that is available about the statistical nature of a set of samples allows for closer estimation. Since the samples are from a population exhibiting a Gaussian distribution, the sample estimators above are in fact maximum likelihood estimators derived from the joint probability density function of many Gaussian

random variables [19]. Having presented the two estimators to be used, a discussion of their statistical nature follows in the next section.

3.2 Statistical Nature of the Mean and Variance Estimators

In order to be able to study the statistical properties of the ratio of the mean to standard deviation for use in approximating the performance of a linear code, the statistical properties of the estimators must be first analyzed. The mean and variance of the two estimators (of the mean, M and variance, V^2 of i.i.d. normal samples) must be found and examined for any bias. Is the mean of the sample estimator in fact equal to the quantity to be estimated [19,20]? As well, in order to be able to obtain a probability density function for the ratio of the estimators easily, their independence must be first established.

It can be mathematically shown (see Appendix A) that the mean and variance of the M and V^2 are as shown below [21] for i.i.d. Gaussian samples with mean μ and variance σ^2.

$$\overline{M} = \mu \qquad Var(M) = \frac{\sigma^2}{N} \tag{3.3}$$

$$\overline{V^2} = \frac{N-1}{N}\sigma^2 \qquad Var(V^2) = \frac{2(N-1)}{N^2}\sigma^4 \tag{3.4}$$

The estimator of the mean is seen to be unbiased. However, it is evident that the estimator for the variance is not an unbiased estimator. The expected value of the variance estimator is not equal to the variance of the samples. It is, nonetheless, an asymptotically unbiased estimator; as the number of samples N approaches infinity, the estimator effectively becomes equal to the correct expected value of σ^2. To remove the

bias, the estimator can be multiplied by an appropriate factor, but this is not expected to be necessary for the purposes here. The number of samples that are to be dealt with will be large. In the next section, more reason for not changing the estimator will become evident. The biased estimator of V^2 helps in the formulation of the probability density function of the ratio of M to V.

If the estimators are in fact independent, then the probability density function of the ratio will be much easier to formulate. Multiplication of the two probability density functions will constitute the joint distribution, and a variable transformation and its Jacobian will be all that is required. It can be shown that the two estimators are in fact uncorrelated for samples of any statistical nature. The estimators are independent for the case of i.i.d. Gaussian samples [21], since they are uncorrelated, and for very large N, they become Gaussian random variables. This is presented in Appendix A. Therefore, the task of obtaining the probability density function of the ratio is not a daunting one.

M is a Gaussian random variable; it is the normalized sum of i.i.d Gaussian samples. Its probability density function is a Gaussian distribution with a mean and variance as shown in the equations of (3.3). The distribution of V^2, and more importantly V, is not so straight forward. This will require some modification to the ratio to produce a denominator with a known and recognizable distribution and will be presented in the following section.

3.3 The Probability Density Function of the Ratio

With the establishment of the independence of the two estimators M and V^2 in the previous section, the probability density function of the ratio of M to V can be found.

The ratio Z is given by

$$Z = \frac{M}{V} = \frac{M}{\sqrt{V^2}}$$

As mentioned in the previous section, the numerator of the ratio, M, is a Gaussian random variable. The distribution of the denominator is not so straight forward. By modifying the numerator and denominator by a constant multiplicative factor, without changing the overall ratio, a distribution for the denominator can also be realized.

The factor by which the numerator and denominator of the ratio is to be multiplied is $\frac{\sqrt{N}}{\sigma}$. This quantity is the reciprocal of the standard deviation of the random variable M obtained from (3.3). The value of the ratio is unchanged, however, the factor allows the denominator random variable to have a recognizable probability density function. The results of this multiplication are

$$\begin{aligned} Z = \frac{M}{V} &= \frac{(\frac{\sqrt{N}}{\sigma})M}{(\frac{\sqrt{N}}{\sigma})V} \\ &= \frac{D}{S} \end{aligned}$$

where,

$$D = (\frac{\sqrt{N}}{\sigma})M \qquad \text{and} \qquad S = (\frac{\sqrt{N}}{\sigma})V.$$

The distribution of D is a scaled version of the distribution of M. The probability density function is still Gaussian, however, rather than the mean and variance of (3.3), the distribution now has a mean of $\frac{\mu\sqrt{N}}{\sigma}$ and a variance equal to 1. The Gaussian probability density function is given in (1.1).

For the denominator, S, some mathematical manipulation is required before its dis-

tribution can be recognized. Starting with the definition of S and expanding,

$$\begin{aligned} S &= \left(\frac{\sqrt{N}}{\sigma}\right)V \\ &= \left(\frac{\sqrt{N}}{\sigma}\right)\sqrt{V^2} \\ &= \left(\frac{\sqrt{N}}{\sigma}\right)\sqrt{\frac{1}{N}\sum_{i=0}^{N}\left(x_i - \overline{x}\right)^2} \\ &= \sqrt{\frac{N}{N}\sum_{i=0}^{N}\left(\frac{x_i - \overline{x}}{\sigma}\right)^2} \\ &= \sqrt{\sum_{i=0}^{N} X_i^2} \end{aligned} \tag{3.5}$$

It can be seen that the random variable S is in fact a Chi distributed random variable. The Chi distribution is formed by square-rooting a Chi-squared distribution, which is the sum of squared Gaussian random variables. The probability density function of a Chi random variable is also found in literature [22,23]. The random variables X_i, which are squared and then summed, are standard Gaussian distributed with mean 0 and variance 1. Chi and Chi-squared random variables formed of standard Gaussian random variables are fully characterized by the degrees of freedom of the variable (i.e. the number of squared variables which are summed), in this case N. As N approaches infinity (i.e. infinite degrees of freedom), the Chi and Chi-squared variables become Gaussian. This is where the implication of independence from Gaussian variables can be made to find the probability density function of the ratio.

Having recognized the distributions of the numerator and denominator after the scaling of each, the probability density function of D and S [22, pp. 417] are presented below.

Note that the degrees of freedom of S is denoted by the subscript N.

$$f_D(x) = \frac{1}{\sqrt{2\pi}} \exp\Big\{-\frac{(x - \frac{\mu\sqrt{N}}{\sigma})^2}{2}\Big\} \tag{3.6}$$

$$f_{S_N}(y) = \frac{y^{N-1} \exp\Big\{-\frac{y^2}{2}\Big\}}{2^{\frac{N}{2}-1}\Gamma\big(\frac{N}{2}\big)} \tag{3.7}$$

Therefore, the probability density function of the ratio Z can now be formulated. It can be formed by substituting variable transformations of $z = \frac{x}{y}$ and $w = x$ into the product of the two individual probability density functions of D and S (due to independence) and dividing the result by the absolute value of the Jacobian of the variable transformations. This procedure is detailed mathematically [24] as

$$f_{ZW}(z,w) = f_{DS}(w, \frac{w}{z}) \frac{1}{|J(x,y)|} \tag{3.8}$$

$$= f_D(w)\, f_S(\frac{w}{z}) \frac{1}{|J(x,y)|} \tag{3.9}$$

where $|J(x,y)|$ is the absolute value of the Jacobian defined as,

$$J(x,y) = \begin{vmatrix} \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} \\ \frac{\partial z}{\partial x} & \frac{\partial z}{\partial y} \end{vmatrix}.$$

Carrying through with this procedure, and integrating from $-\infty$ to ∞ with respect to the dummy variable, w, the resulting probability density function of the ratio is given as

$$f_Z(z) = \frac{\exp\big\{-\frac{\gamma^2}{2(z^2+1)}\big\}}{2^{\frac{N-2}{2}}\Gamma\big(\frac{N}{2}\big)} \frac{1}{\sqrt{z^2+1}^{N+1}} \Big[\sum_{k=0}^{\lfloor \frac{N}{2} \rfloor} \binom{N}{2k} \big(\frac{\gamma z}{\sqrt{z^2+1}}\big)^{N-2k} \frac{(2k)!}{k!2^k}\Big] \tag{3.10}$$

where $\gamma = \frac{\mu\sqrt{N}}{\sigma}$. (The mathematics behind this result are presented in Appendix B).

3.4 Chapter Summary

This chapter presented the estimators of the mean and variance of a set of Gaussian distributed samples. The estimators can be shown to be uncorrelated and for the special case of Gaussian distributed samples, the two estimators are independent. This case is encountered here due to the Gaussian approximation of chapter 2. The probability density function of the ratio of the two quantities to be estimated (i.e., the mean to standard deviation) was derived using their probability density functions. By adjusting the numerator and denominator of the ratio, the distributions were recognizable as a Gaussian distribution for the numerator and a Chi distribution for the denominator. The independence of the two estimators was then used to obtain the probability density function for the ratio.

With the probability density function of the ratio Z of the mean to the standard deviation of the LLR, an analysis of the Q-function as a function of the random variable Z can be investigated in the next chapter to obtain its variance.

Chapter 4

Analysis of the Ratio $Z = \frac{D}{S}$

It was established in the last chapter that the numerator and denominator of the ratio $\frac{D}{S}$ are indeed independent. The independence follows from the fact that the estimators of the numerator and denominator are uncorrelated, and since the samples are Gaussian distributed [21]. The independence was used in the previous chapter to determine the probability density function of the ratio.

Although the new method is only an approximation for the bit error probability performance of a linear code, the method does possess its own merits. However, additional expressions need to be formulated before discussing the merits in chapter 5. This chapter will present these expressions, including the r^{th} moment of the ratio which will later be used to formulate the mean and variance of $Q(Z)$, where $Z = \frac{D}{S}$. The variance of $Q(Z)$ can be used to discuss the precision of the approximation.

4.1 r^{th} Moment of the Ratio Z

The calculation of the r^{th} moment of the random variable Z is possible using its probability density function. This would require multiplying the expression of (3.10) by z^r and

integrating with respect to z from 0 to ∞. This was attempted, however, the resulting integral proved to be difficult and a simple closed form expression was not attainable [25–27]. This called for another approach to obtain a closed-form expression of the r^{th} moment.

Using the fact of independence between the numerator random variable, D, and the denominator random variable, S, the r^{th} moment of the ratio can be found. It is known that the mean of the product of two independent random variables is the product of the means of the individual random variables [24]. This is to say that,

$$\overline{AB} = \bar{A}\bar{B}, \tag{4.1}$$

where A and B are independent random variables.

With this property, the r^{th} moment follows,

$$E[Z^r] = E[S^{-r}]E[D^r], \quad \text{where } E[\cdot] \text{ denotes expectation.} \tag{4.2}$$

Now the individual expectations from above can be formulated separately. The probability density functions for D and S are provided in equations (3.6) and (3.7), respectively. Recall that D is a Gaussian random variable $N\left(\frac{\mu\sqrt{N}}{\sigma}, 1\right)$ and S is a central Chi distributed random variable.

First, $E[S^{-r}]$ is formulated, followed by the formulation of $E[D^r]$.

$$\begin{aligned} E[S^{-r}] &= \int_0^\infty y^{-r} f_{S_N}(y)\, dy \\ &= \int_0^\infty \frac{y^{-r} y^{N-1} e^{-\frac{y^2}{2}}}{2^{\frac{N}{2}-1}\Gamma\left(\frac{N}{2}\right)}\, dy \\ &= \int_0^\infty \frac{y^{(N-r)-1} e^{-\frac{y^2}{2}}}{2^{\frac{N}{2}-1}\Gamma\left(\frac{N}{2}\right)}\, dy. \end{aligned}$$

Letting $t = \frac{y^2}{2}$, $y = \sqrt{2t}$, $dt = y\ dy$, and continuing,

$$\begin{aligned} E[S^{-r}] &= \int_0^\infty \frac{(2t)^{\frac{(N-r)}{2}-1}\, e^{-t}}{2^{\frac{N}{2}-1}\Gamma\left(\frac{N}{2}\right)}\, dt \\ &= \frac{\left(\frac{1}{2}\right)^{\frac{r}{2}}}{\Gamma\left(\frac{N}{2}\right)} \int_0^\infty t^{\frac{N-r}{2}-1}\, e^{-t}\, dt \\ &= \left(\frac{1}{2}\right)^{\frac{r}{2}} \frac{\Gamma\left(\frac{1}{2}(N-r)\right)}{\Gamma\left(\frac{N}{2}\right)}. \end{aligned} \tag{4.3}$$

where $\Gamma\left(x\right) = \int_0^\infty y^{x-1}\, e^{-y}\, dy$ by definition [28]. Similarly,

$$\begin{aligned} E[D^r] &= \int_{-\infty}^{\infty} x^r\ f_D\left(x\right) dx \\ &= \int_{-\infty}^{\infty} \frac{x^r}{\sqrt{2\pi}}\, e^{-\frac{(x-\frac{\mu\sqrt{N}}{\sigma})^2}{2}}\, dx. \end{aligned}$$

Letting $t = x - \frac{\mu\sqrt{N}}{\sigma}$, $dt = dx$, and continuing,

$$\begin{aligned} E[D^r] &= \int_{-\infty}^{\infty} \frac{\left(t + \frac{\mu\sqrt{N}}{\sigma}\right)^r}{\sqrt{2\pi}}\, e^{-\frac{t^2}{2}}\, dt \\ &= \sum_{k=0}^{r} \binom{r}{k} \left(\frac{\mu\sqrt{N}}{\sigma}\right)^{r-k} \int_{-\infty}^{\infty} \frac{t^k}{\sqrt{2\pi}}\, e^{-\frac{t^2}{2}}\, dt \\ &= \sum_{k=0}^{\lfloor r/2 \rfloor} \binom{r}{2k} \left(\frac{\mu\sqrt{N}}{\sigma}\right)^{r-2k} \frac{(2k)!}{2^k k!}. \end{aligned} \tag{4.4}$$

In the above expression, it was noted that

$$\int_{-\infty}^{\infty} \frac{t^k}{\sqrt{2\pi}}\, e^{-\frac{t^2}{2}}\, dt = E[t^k].$$

$E[t^k]$ is the k^{th} moment of a Gaussian random variable $N(0,1)$ and can be expressed

as [1,7,24],

$$E[t^k] = \begin{cases} \frac{(2v)!}{2^v v!}, & v = \frac{k}{2} \text{ when } k \text{ is even} \\ 0, & \text{when } k \text{ is odd} \end{cases}$$

Now, the resulting expression for $E[Z^r]$ can be formed using (4.3) and (4.4) from above, to yield,

$$E[Z^r] = \left(\frac{1}{2}\right)^{\frac{r}{2}} \frac{\Gamma\left(\frac{1}{2}(N-r)\right)}{\Gamma\left(\frac{N}{2}\right)} \sum_{k=0}^{\lfloor \frac{r}{2} \rfloor} \binom{r}{2k} \left(\frac{\mu\sqrt{N}}{\sigma}\right)^{r-2k} \frac{(2k)!}{2^k k!}. \tag{4.5}$$

It is evident that the expression for the r^{th} moment is a function of the ratio of the individual expected means of the two random variables, namely μ and σ. Also, it is a function of N, the number of samples used to obtain the values of D and S numerically. Since the r^{th} moment depends on number of samples, it can be viewed as an estimator [19]. This leads to the questions: does the first moment (i.e., the mean) and other moments approach the true expected values for the ratio? Are the expressions asymptotically unbiased? Or are they biased?

Unfortunately, the expression is not a simple function of N for general r, otherwise the limit of (4.5) as N approaches infinity could be taken to answer these questions. However, the limit can be taken for predetermined values of r. For example, with $r = 1$, the first moment is [23, pp. 513]

$$\begin{aligned} \lim_{N\to\infty} & \left(\frac{1}{2}\right)^{\frac{1}{2}} \frac{\Gamma\left(\frac{1}{2}(N-1)\right)}{\Gamma\left(\frac{N}{2}\right)} \frac{\mu\sqrt{N}}{\sigma} \\ &= \lim_{N\to\infty} \left(\frac{N}{2}\right)^{\frac{1}{2}} \frac{\Gamma\left(\frac{1}{2}(N-1)\right)}{\Gamma\left(\frac{N}{2}\right)} \frac{\mu}{\sigma} \\ &\approx \frac{\mu}{\sigma}. \end{aligned} \tag{4.6}$$

Likewise, for the second moment, $r = 2$,

$$\begin{aligned}
&\lim_{N\to\infty} \frac{1}{2} \frac{\Gamma\left(\frac{1}{2}(N-2)\right)}{\Gamma\left(\frac{N}{2}\right)} \left(\left(\frac{\mu\sqrt{N}}{\sigma} \right)^2 + 1 \right) \\
&= \lim_{N\to\infty} \frac{1}{2} \frac{\Gamma\left(\frac{1}{2}(N-2)\right)}{\frac{1}{2}(N-2)\Gamma\left(\frac{1}{2}(N-2)\right)} \left[\frac{\mu^2 N}{\sigma^2} + 1 \right] \\
&= \lim_{N\to\infty} \frac{1}{N-2} \left[\frac{\mu^2 N}{\sigma^2} + 1 \right] \\
&\approx \frac{\mu^2}{\sigma^2}. \qquad (4.7)
\end{aligned}$$

The answers to the questions is that the expressions for the first and other moments are in fact asymptotically unbiased. This was determined empirically for N very large, for different values of r. The first moment approaches the ratio of the expected means, $\frac{\mu}{\sigma}$, and the second moment approaches the square of $\frac{\mu}{\sigma}$ as N increases, etc. The importance of this observation is that no multiplicative factor needs to be applied to the expression; the expressions do not drift from the true values of the ratio and therefore do not introduce errors into the approximation.

Observing (4.6), it is seen that the first moment of the ratio is simply the ratio of the first moments of the two quantities D and S. From (4.7), the second moment of the ratio is seen to be equal to the ratio of the squares of the means of the estimated quantities. The situation holds for higher moments as well and is indicative of the invariance property of maximum likelihood estimation. This property states that the maximum likelihood estimate of a function g with parameter θ, $\widehat{g(\theta)}$, is equal to the function of the maximum likelihood estimate of the parameter θ, i.e. $g(\hat{\theta})$ [19,20]. The r^{th} moment of Z is therefore equal to the ratio of the mean of D raised to the r^{th} power, to the mean of S raised to the r^{th} power.

The ratio $Z = \frac{D}{S}$ will be used as an argument of the Q-function in the sections to come, and since the expressions are unbiased, it is known that the estimation error will

approach zero as N increases, without further modifications to the expressions.

4.2 Expansion of $Q(Z)$

Having formulated expressions for the r^{th} moments of Z, focus can now be shifted to the use of Z as an argument to the Q-function. This relates back to a main idea of this thesis of modeling the LLR values using a BPSK modulation scheme in section 2.2.

In order to obtain the mean and variance of the function, a series expansion is used. The Q-function can be written in terms of the complementary error function [25, 27], $erfc(\cdot)$, which can be expressed by a series expansion. This yields,

$$\begin{aligned}
Q(z) &= \frac{1}{2} erfc\left(\frac{z}{\sqrt{2}}\right) \\
&= \frac{1}{2}\left(1 - erf\left(\frac{z}{\sqrt{2}}\right)\right) \\
&= \frac{1}{2} - \frac{1}{2}\frac{2x}{\sqrt{\pi}}\left[1 - \frac{x^2}{1!\,3} + \frac{x^4}{2!\,5} - \frac{x^6}{3!\,7} + \frac{x^8}{4!\,9} - \ldots\right]\Bigg|_{x=\frac{z}{\sqrt{2}}} \\
&= \frac{1}{2} - \frac{z}{\sqrt{2\pi}}\left[1 - \frac{z^2}{1!\,3\cdot 2} + \frac{z^4}{2!\,5\cdot 4} - \frac{z^6}{3!\,7\cdot 8} + \frac{z^8}{4!\,9\cdot 16} - \ldots\right] \\
&= \frac{1}{2} - \frac{z}{\sqrt{2\pi}}\left[\sum_{k=0}^{\infty}(-1)^k \frac{z^{2k}}{k!\,(2k+1)\cdot 2^k}\right] \\
&= \frac{1}{2} - \frac{1}{\sqrt{2\pi}}\left[\sum_{k=0}^{\infty}(-1)^k \frac{z^{2k+1}}{k!\,(2k+1)\cdot 2^k}\right].
\end{aligned} \tag{4.8}$$

4.3 Mean and Variance of $Q(Z)$

With the series expansion of $Q(z)$ above, the mean and variance can be formulated. By taking the expectation of $Q(z)$, the first and second moments of $Q(z)$ can be calculated. The variance can be expressed using the second moment of $Q(z)$ and the square of the mean. These expressions are now presented.

4.3.1 Mean of $Q(Z)$

The mean of $Q(Z)$, using the series expansion of (4.8), is given by,

$$\begin{aligned}
E[Q(z)] =& \overline{Q(z)} \\
=& \frac{1}{2} - \frac{1}{\sqrt{2\pi}} \left[\sum_{k=0}^{\infty} (-1)^k \frac{E[\, z^{2k+1}\,]}{k!\,(2k+1)\cdot 2^k} \right] \\
=& \frac{1}{2} - \frac{1}{\sqrt{2\pi}} \sum_{k=0}^{\infty} (-1)^k \left(\frac{1}{2}\right)^{\frac{2k+1}{2}} \frac{\Gamma\left(\frac{1}{2}(N-2k-1)\right)}{\Gamma\left(\frac{N}{2}\right) k!\,(2k+1)\cdot 2^k} \cdot \\
& \sum_{i=0}^{\lfloor \frac{2k+1}{2} \rfloor} \binom{2k+1}{2i} \left(\frac{\mu\sqrt{N}}{\sigma}\right)^{2k+1-2i} \frac{(2i)!}{2^i\, i!}.
\end{aligned} \tag{4.9}$$

4.3.2 Variance of $Q(Z)$

For the variance of $Q(Z)$, the second moment of $Q(Z)$ needs to be formulated and then the squared-mean is subtracted. To formulate the second moment of $Q(z)$, the series expansion of $Q(z)$ from (4.8) must be squared, i.e.,

$$\begin{aligned}
Q(z)^2 =& \left(\frac{1}{2} - \frac{1}{\sqrt{2\pi}} \sum_{k=0}^{\infty} (-1)^k \frac{z^{2k+1}}{k!\,(2k+1)\cdot 2^k} \right)^2 \\
=& \frac{1}{4} - \frac{1}{\sqrt{2\pi}} \sum_{k=0}^{\infty} (-1)^k \frac{z^{2k+1}}{k!\,(2k+1)\cdot 2^k} + \frac{1}{2\pi} \left[\sum_{k=0}^{\infty} (-1)^k \frac{z^{2k+1}}{k!\,(2k+1)\cdot 2^k} \right]^2 \\
=& \frac{1}{4} - \frac{1}{\sqrt{2\pi}} \sum_{k=0}^{\infty} (-1)^k \frac{z^{2k+1}}{k!\,(2k+1)\cdot 2^k} + \frac{1}{2\pi} \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} (-1)^{i+j} \cdot \\
& \frac{z^{2(i+j)+2}}{i!\,j!\,(2i+1)\,(2j+1)\cdot 2^{i+j}}.
\end{aligned} \tag{4.10}$$

Taking the expectation of the above, the second moment is obtained as a function of various moments of Z. Substituting the moments of Z from (4.5), the second moment is

obtained as,

$$E[Q(z)^2] = \frac{1}{4} - \frac{1}{\sqrt{2\pi}} \sum_{k=0}^{\infty} (-1)^k \frac{E[\, z^{2k+1}\,]}{k!\,(2k+1)\cdot 2^k} + \frac{1}{2\pi} \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} (-1)^{i+j} \cdot \frac{E[\, z^{2(i+j)+2}\,]}{i!\,j!\,(2i+1)\,(2j+1)\cdot 2^{i+j}}$$

$$= \frac{1}{4} + \overline{Q(z)} - \frac{1}{2} + \frac{1}{\sqrt{2\pi}} \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \frac{(-1)^{i+j}}{i!\,j!\,(2i+1)(2j+1)\cdot 2^{i+j}} \left(\frac{1}{2}\right)^{i+j+1} \cdot \frac{\Gamma\left(\frac{1}{2}\left(N-2(i+j)-2\right)\right)}{\Gamma\left(\frac{N}{2}\right)} \sum_{m=0}^{i+j+1} \binom{2(i+j)+2}{2m} \left(\frac{\mu\sqrt{N}}{\sigma}\right)^{2(i+j)+2-2m} \frac{(2m)!}{m!\,2^m}. \tag{4.11}$$

Taking the expression $E[\,Q(Z)^2\,]$ in (4.11) and subtracting $E[Q(z)]^2$, the variance is found. For the sake of completeness, the expression is shown here, where $\overline{Q(z)}$ was defined in equation (4.9).

$$Var\big(Q(Z)\big) = E[\,Q(z)^2\,] - E[\,Q(z)\,]^2$$

$$= \frac{1}{4} + \overline{Q(z)} - \frac{1}{2} + \frac{1}{\sqrt{2\pi}} \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \frac{(-1)^{i+j}}{i!\,j!\,(2i+1)\,(2j+1)\cdot 2^{i+j}} \left(\frac{1}{2}\right)^{i+j+1} \cdot \frac{\Gamma\left(\frac{1}{2}\left(N-2(i+j)-2\right)\right)}{\Gamma\left(\frac{N}{2}\right)} \sum_{m=0}^{i+j+1} \binom{2(i+j)+2}{2m} \left(\frac{\mu\sqrt{N}}{\sigma}\right)^{2(i+j)+2-2m} \cdot \frac{(2m)!}{m!\,2^m} - \overline{Q(z)}^2. \tag{4.12}$$

4.3.3 Comments on the Mean and Variance of $Q(Z)$

Examining the expressions above, it can be seen that the series are comprised of an infinite number of terms. It is not practical nor feasible to include a large numbers of terms in the series expansion due to the computation time and memory requirements of

such a task, and thus the series must be truncated. It is necessary to examine when the individual terms become small enough so that they can be neglected. This approximation must be done numerically as it is difficult to analytically estimate the number of terms required for different values of N and $\frac{\mu}{\sigma}$. A good rule of thumb was found; if the number of terms included in the series is 4 times the ratio $\left(\frac{\mu}{\sigma}\right)^2$, then stable results are obtained for $\frac{\mu}{\sigma} > 2$ and $N > 1000$.

4.4 Chapter Summary

In this chapter, the expression for the r^{th} moment of Z was formulated and then later used in the determination of the mean and variance of $Q(Z)$. With these quantities calculated, the merits of the new approximation method can be explained in the next chapter.

Chapter 5

Variance Comparison of Simulation Methods

In the previous chapters, the reasoning and mathematics behind the new approximation method for the bit error performance for linear block codes have been explained thoroughly. Using these arguments, the approximation can be obtained. However, the merits of the method are not yet clear.

Conventional simulation methods tend to be a lengthy process, requiring hours of computation time, especially for low bit error probabilities. This chapter will focus on the merits of the method, prior to seeing any simulation results. This will be done by comparing the number of samples required to produce the approximation to the number required for conventional bit error performance simulations. As it will be shown, this new method requires far fewer samples than the conventional simulation methods to obtain the same error performance, and therefore less time.

To compare the performance of the methods, the variance in the bit error probability is calculated and used. The variance in the bit error probability obtained from conventional

simulation methods is defined and calculated in the next section. The expression for the variance of the approximation method was defined in equation (4.12). The comparisons are made in the last section of this chapter.

5.1 Analysis of Conventional Simulation Methods

In this analysis, a random variable E is defined to encapsulate the error totaling done using conventional simulation methods. These simulation methods were described earlier, in chapter 1. Suppose that a bit b_i is transmitted, in position i of a bit stream, and that the received bit in position i is decoded as $\hat{b}_i$. Then the random variable ϵ_i can be defined as follows:

$$\epsilon_i = \begin{cases} 1 & \text{when } b_i \neq \hat{b}_i \\ 0 & \text{otherwise.} \end{cases} \tag{5.1}$$

ϵ_i indicates an error event. The random variable E is defined as

$$\mathrm{E} = \frac{1}{N} \sum_{i=1}^{N} \epsilon_i. \tag{5.2}$$

The probability of bit error, P_e, is defined as the number of errors that occur in a stream of bits divided by the total number of bits transmitted in the stream. By this definition, E is a random variable representing the probability of bit error.

Insight into E and therefore conventional simulation methods, can be obtained from observing its mean and variance. The mean and variance of E are derived below.

$$\begin{aligned} \bar{\mathrm{E}} &= \overline{\frac{1}{N} \sum_{i=1}^{N} \epsilon_i} \\ &= P_e, \end{aligned} \tag{5.3}$$

and,

$$\begin{aligned}
\sigma_{\mathrm{E}}^2 &= \overline{\left(\frac{1}{N}\sum_{i=1}^{N}\epsilon_i\right)^2} - \left(\overline{\frac{1}{N}\sum_{i=1}^{N}\epsilon_i}\right)^2 \\
&= \frac{1}{N^2}\left[\overline{\sum_{i=1}^{N}\epsilon_i^2} + \sum_{\substack{i=1\\ i\neq j}}^{N}\sum_{j=1}^{N}\bar{\epsilon_i}\bar{\epsilon_j}\right] - P_e^2 \\
&= \frac{1}{N^2}\sum_{i=1}^{N}\bar{\epsilon_i^2} + \frac{1}{N^2}\sum_{\substack{i=1\\ i\neq j}}^{N}\sum_{j=1}^{N}\bar{\epsilon_i}\bar{\epsilon_j} - P_e^2 \\
&= \frac{P_e}{N} + \frac{N(N-1)}{N^2}P_e^2 - P_e^2 \\
&= \frac{P_e}{N}(1-P_e) \qquad (5.4) \\
&\approx \frac{P_e}{N} \quad \text{when } P_e \text{ is small.} \qquad (5.5)
\end{aligned}$$

Note that $\bar{\epsilon_i} = P_e$. The variance in the probability of bit error is approximately equal to $\frac{P_e}{N}$ as shown in (5.5). It is evident that the variance of E decreases as N increases. This is expected since the inclusion of more samples reduces uncertainty in the bit error probability for a given SNR.

5.2 Variance of Random Variable $Q(Z)$ Revisited

The variance of the random variable was formulated in the previous chapter (see section 4.3.2). Given the number of samples N, and the value of the ratio $\frac{\mu}{\sigma}$, the variance can be calculated by implementing equation (4.12). It can easily be verified using (4.7) and (4.6) that the variance of Z approaches zero with increasing N. Therefore, the due to the one-to-one mapping between Z and $Q(Z)$, the variance of $Q(Z)$ also approaches zero.

With the value of the variance of the function $Q(Z)$, and the variance obtained by

conventional methods above, comparisons can be made to show the merits of the new approximation method. However, prior to this, the relationship between $\frac{\mu}{\sigma}$ and P_e must be established.

5.3 Relationship Between $\frac{\mu}{\sigma}$ and P_e

Since the new approximation method is to be used for the evaluation of the performance of linear codes in the presence of AWGN, the relationship between $\frac{\mu}{\sigma}$ and P_e is a simple and well-known one. Suppose a bit is transmitted over an AWGN channel using BPSK modulation, with bit energy E_b and the noise power (and hence variance) on the channel is $\frac{N_0}{2}$. The probability of bit error for this setup is [1, pp. 258],

$$P_e = Q\left(\sqrt{\frac{2\,E_b}{N_0}}\right)$$

The transmitted bits are disturbed by independent Gaussian noise samples, and each received bit has a mean amplitude, μ, and variance, σ^2, due to noise. Considering the ratio $\frac{\mu}{\sigma}$ as a signal-to-noise ratio, the probability of bit error for a bit with energy μ^2, disturbed by AWGN, with noise power σ^2, is then

$$P_e = Q\left(\frac{\mu}{\sigma}\right).$$

Therefore, the P_e is in fact a function of the value of $\frac{\mu}{\sigma}$ via the Q-function. The value μ is seen to be equal to the signal amplitude $\sqrt{E_b}$ and σ is equal to $\sqrt{\frac{N_0}{2}}$.

With the relationship stated, the variances can be compared for different values of $\frac{\mu}{\sigma}$ and number of samples, N. These two quantities will be varied below in the expressions of the variances in equations (4.12) and (5.5).

5.4 Comparison of Variances and the Merits

The variance of $Q(Z)$ is a complicated expression and not a simple expression of N or $\frac{\mu}{\sigma}$. This means that the comparison must be done empirically using (4.12) and (5.5), and not analytically. Tables 5.1 and 5.2 contain variance values obtained for varying values of $\frac{\mu}{\sigma}$ and N.

$\frac{\mu}{\sigma}$	P_e	σ^2_{E}	$\sigma^2_{Q(Z)}$	Ratio $\frac{\sigma^2_{\mathrm{E}}}{\sigma^2_{Q(Z)}}$
1	1.5866e-1	1.5866e-5	8.78160e-6	1.81
2	2.2750e-2	2.2750e-6	8.746385e-7	2.60
3	1.3499e-3	1.3499e-7	1.0841e-8	12.45
4	3.1671e-5	3.1671e-9	1.6355e-11	193.6

Table 5.1: Comparison of Variances, $N = 10000$

$\frac{\mu}{\sigma}$	P_e	σ^2_{E}	$\sigma^2_{Q(Z)}$	Ratio $\frac{\sigma^2_{\mathrm{E}}}{\sigma^2_{Q(Z)}}$
1	1.5866e-1	1.5866e-6	8.7823e-7	1.81
2	2.2750e-2	2.2750e-7	8.7452e-8	2.60
3	1.3499e-3	1.3499e-8	1.0807e-9	12.49
4	3.1671e-5	3.1671e-10	1.6143e-12	196.2
5	2.8665e-7	2.8665e-12	2.9958e-16	9568.5

Table 5.2: Comparison of Variances, $N = 100000$

From these results, the merits of the new approximation can be presented. Using Chebyshev's inequality [29], it can be concluded that with smaller variance, there is a smaller chance of a large deviation from the mean value of a quantity. This relates directly to the precision of the quantity. This precision is relevant for discussing the merit of the approximation.

The fifth column of the tables is of most interest. The ratio of the variances, as well as

the variances themselves, show that the variance of $Q(Z)$ is always smaller. This means that this method is more precise. However, another interpretation of the ratio values can be made. If the result of the ratio of variances is equal to A, it can also be said that in order to achieve the same P_e, the number of samples needed by the approximation method can be reduced by a factor A. Therefore, A times *fewer* samples are required.

This was observed in practice. Fewer samples were needed to obtain a value of P_e since there is less variation in the result. For example, observing the data in tables 5.1 and 5.2, for a $\frac{\mu}{\sigma}$ value of 3, using conventional methods, the variance of 1.35e-8 is obtained with 100000 samples, while an even smaller variance of $1.08e-8$ can be obtained with 10000 samples using the approximation. This is a factor of 10 reduction in the number of samples needed to obtain similar precision in the bit error probability.

Unlike the conventional method, where a large number of samples is required to ensure a smaller variance in the resulting probability of bit error, the approximation method requires fewer. This translates into a savings in the time required to carry out computations. And although this method is only an approximation, it is quite close to the actual bit error curves. This will be seen in the following chapter. So the reduced number of samples required is the main advantage of this method, producing a good approximation to the conventionally-simulated bit error performance curves.

5.5 Chapter Summary

The approximation using LLR values was shown to require fewer samples than conventional simulation techniques to obtain the performance results of a linear code. This merit was illustrated by comparing the variance of conventional simulations methods to the approximation method. The variance of the Q-function, with Z as an argument, is always smaller than the variance in the results for conventional methods for a given bit

error probability and number of samples, N. Therefore, to obtain the same precision in the probability of bit error, fewer samples are required.

The next chapter presents the results of simulations using the approximation and compares it to the bit error performance of linear codes obtained through conventional simulation methods.

Chapter 6

Simulation Results and Discussion

This chapter presents the simulations results of using the approximation that has been analyzed in the preceding chapters. Two simple codes are simulated over an AWGN channel, and then the LLR is calculated for each bit position using the definition of the LLR found in equation (2.1). The conventional simulation method of calculating the bit error probability was used. This method involves performing hard decisions on the LLR values to decide upon a bit value, and the accumulation of errors determines the bit error performance of the code. This is done for every component of the received codeword. Then, making use of the LLR values for one bit position, the mean and standard deviation of them are calculated and used to gauge the performance of the linear code.

Simulation results are presented for two simple codes. Comments on the approximation are made when compared to those curves of the conventionally simulated bit error probability. The first-order approximations are presented using the expressions of chapter 2, for both codes. The implications of the results and appropriateness of the Gaussian approximation are discussed to conclude the chapter.

6.1 Simulation Parameters and Setup

For any linear code, the parameters of the code can be represented as (n, k, d), where n is the number of bits per codeword, k is the number of bits of information which are encoded (i.e. the dimensionality of the code), and d is the minimum Hamming distance[1] between the codewords. The Hamming distance d relates directly to the performance of the code. The larger the value of d, the more errors that can be detected and corrected during decoding [1, 3], yielding better performance results.

The two codes employed to illustrate the results of the approximation against the conventionally simulated bit error performance were chosen to be the $(8, 4, 4)$ Reed-Muller code and the $(24, 12, 8)$ Golay code. Both of these codes are well-documented in literature. The two codes were chosen since they are prime examples of binary linear codes and are easy to implement.

The generator matrix, G_{RM}, of the Reed-Muller code that was used, is shown below. The codebook of the code consists of 16 codewords ($2^k = 2^4$). The modulated codewords are then used in the LLR definition of (2.1) as code $\mathcal{C}$.

$$G_{RM} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{6.1}$$

The generator matrix, G_G, of the Golay code that was used is presented below. The

[1] The Hamming distance is defined as the number of bits positions in which two codewords differ.

Golay code contains 4096 codewords ($2^k = 2^{12}$) in its codebook.

$$G_G = \left[\begin{array}{c|cccccccccccc} & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ I_{12} & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{array}\right] \tag{6.2}$$

I_{12} is a 12-by-12 identity matrix which contains ones only on the diagonal of the matrix and zeros elsewhere.

Following from figure 1.2, the codewords are ± 1 modulated and then transmitted. These modulated bits were simulated over an AWGN channel with noise samples with mean 0 and noise variance $\sigma_\eta^2 = \frac{N_0}{2}$. The values of the noise variance were independent parameters in the simulation, and the resulting bit error probabilities for the conventional simulation method and the approximation method were plotted against the SNR $\frac{E_b}{N_0}$, in dB. The results are now presented in the following section.

6.2 Simulation Results

Based upon the simulation setup described above and the calculation of the LLR using (2.1), the performance of the codes was simulated. First, the Reed-Muller code performance curves are presented followed by the Golay code performance curves.

6.2.1 Reed-Muller Code Performance

Figure 6.1 presents the simulation results of the Reed-Muller code over an AWGN channel. The solid line represents the conventionally simulated bit error probability curve while the dotted line represents the approximation proposed by this thesis.

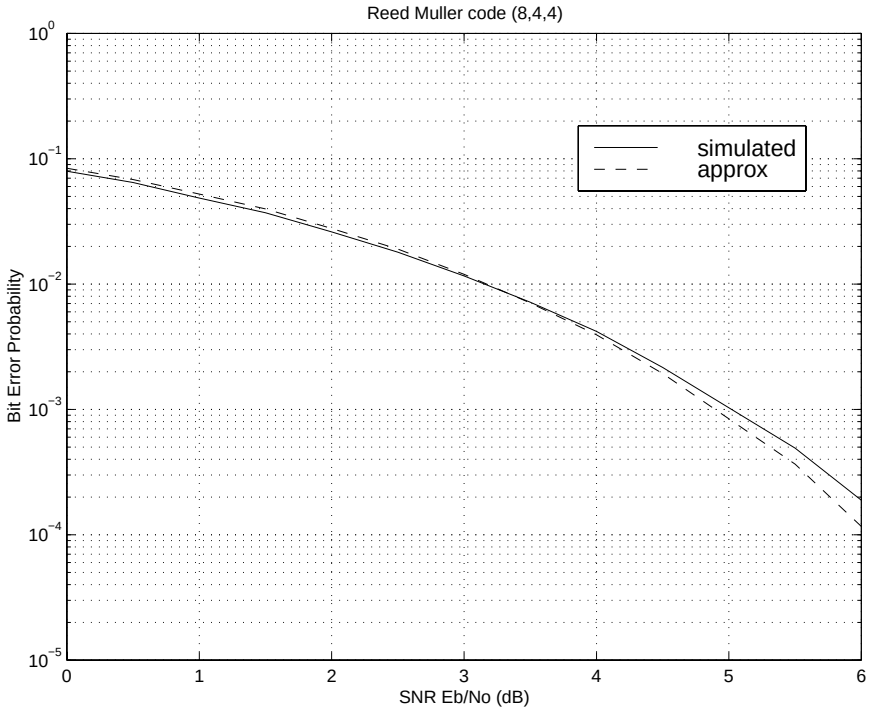

Figure 6.1: Bit Error Performance Comparison for the Reed-Muller code

The first impression obtained from figure 6.1 is that the approximation is remarkably

close to the simulated bit error performance of the code, more so for lower SNR values than higher SNR values. The close approximation at low SNR follows exactly from the approximation made in chapter 2. The LLR values were approximated to be Gaussian in situations of high noise variances since the higher-order terms of the expansion become negligible. For high SNR (i.e., low noise variance), the higher-order terms become more significant, and therefore cannot be neglected. The Gaussian nature discussed earlier breaks down. The approximation curve is seen to deviate, as expected.

For the low SNR values, which are the range of interest and of practical use in industry, the approximation is excellent. Even for higher SNR values, the amount of divergence is not catastrophic since the numbers in this region are small and the actual differences are small.

6.2.2 Golay Code Performance

Figure 6.2 presents the simulation results of the Golay code over an AWGN channel. The solid line represents the conventionally simulated bit error probability curve while the dotted line represents the approximation proposed by this thesis.

Again, the approximation is seen to be remarkably close to the simulated bit error performance of the code. The deviation in the approximation from the conventionally simulated performance curve is consistent with that seen for the Reed-Muller code. For higher SNR values, the approximation made in chapter 2 is not valid and therefore, the curves separate. The approximation is excellent for low SNR values and is still good for high SNR values.

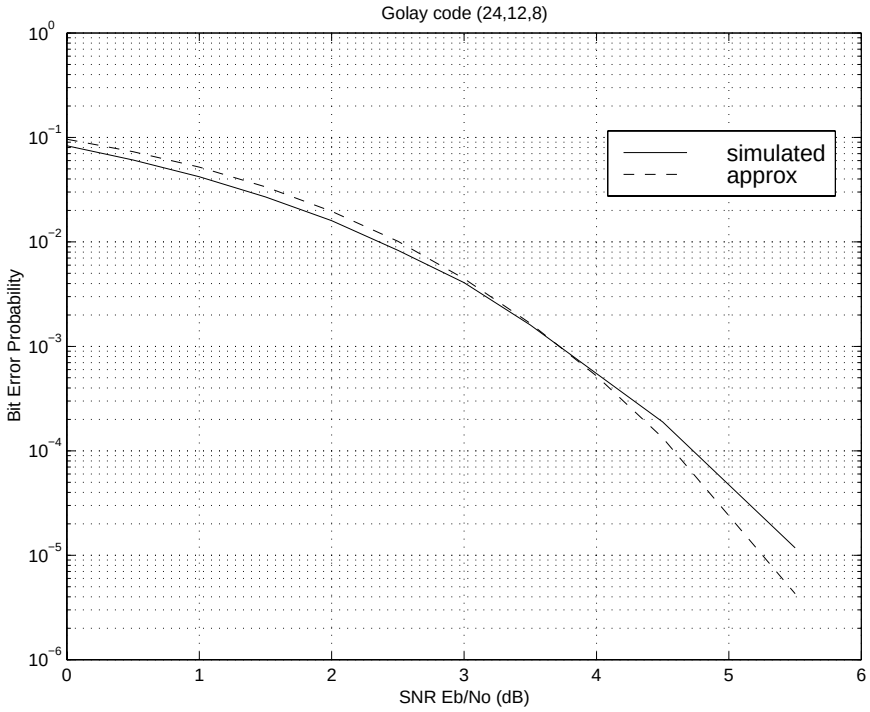

Figure 6.2: Bit Error Performance Comparison for the Golay code

6.3 First-Order Approximation Results

Chao *et al.* [30] evaluated the performance of binary block codes at low SNR values using a series expansion for the probability of correct decoding and considering only the first two terms of the series (zeroth and first-order). They give numerical results for the performance of a biorthogonal code with 16 codewords. It can be shown through a rotation of coordinates that this code is equivalent to the $(8, 4, 4)$ Reed-Muller code considered above. A comparison can thus be made between the two codes. The results of figure 1 of [30, pp. 1686] can be directly compared with the results shown graphically in figure 6.3. Results obtained via the approximation described in this thesis are more precise over a wider range of SNR values. The results are shown graphically in figure 6.3

for the Reed-Muller code.

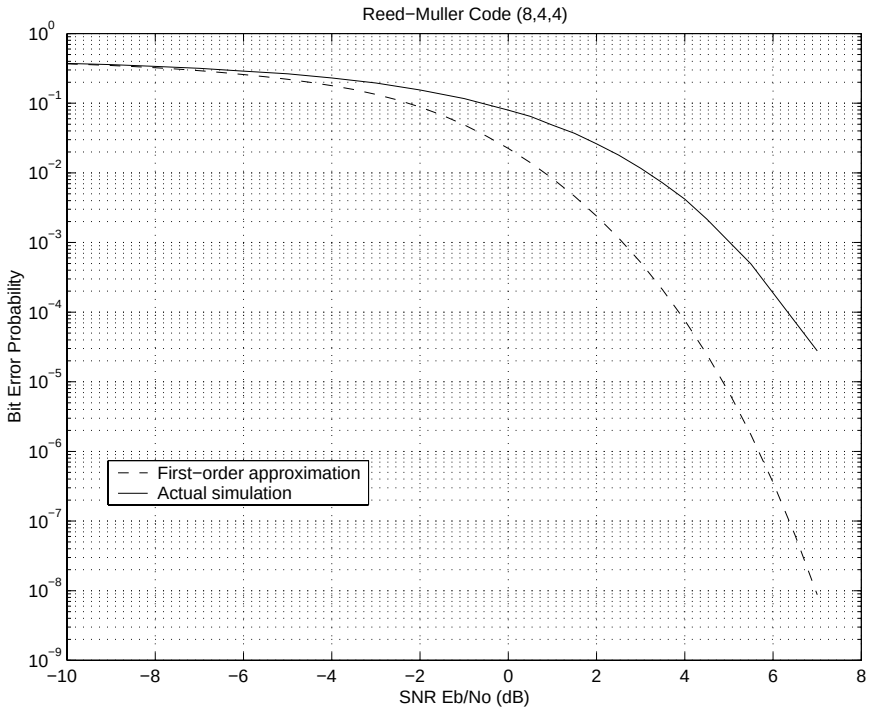

Figure 6.3: First-Order Approx. for the Bit Error Performance of the $(8, 4, 4)$ Reed-Muller code

Similar first-order results can be shown for the Golay code in figure 6.4. This first-order approximation deviates quicker than that of the Reed-Muller code since n is larger, producing more higher-order groupings which appear Gaussian and contribute to the overall approximation.

6.4 Implications of the Results

The approximation is an excellent one for low SNR; the performance of binary linear codes can be accurately approximated for these SNR values. As was shown in the previ-

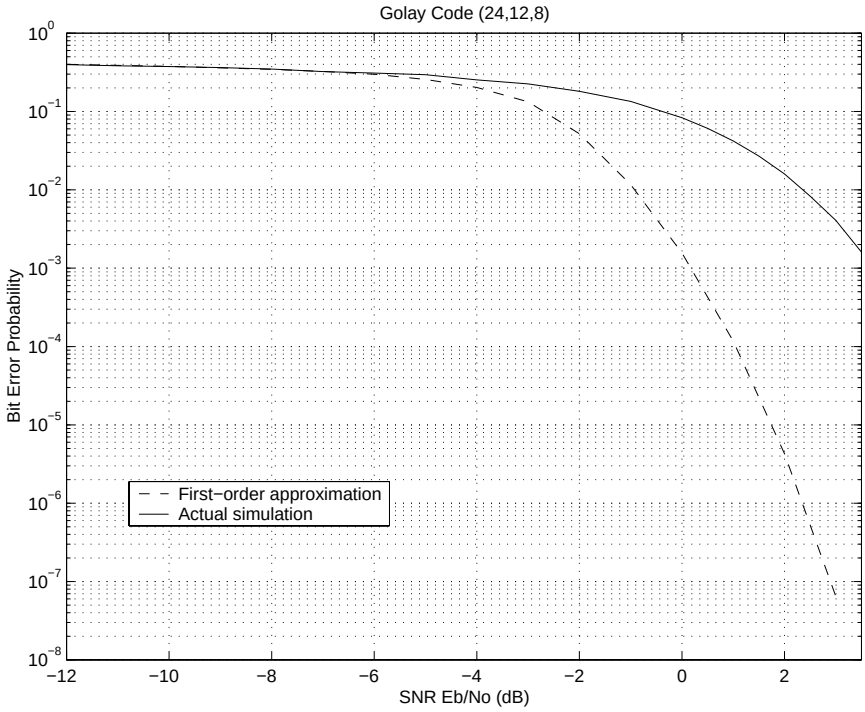

Figure 6.4: First-Order Approx. for the Bit Error Performance for the $(24, 12, 8)$ Golay code

ous chapter, fewer sample LLR values are required to obtain a relatively precise bit error probability value for a given SNR $\frac{E_b}{N_0}$. This was the result of the analysis done on $Q(Z)$, where Z is considered to be a random variable and was defined to be the ratio of mean to standard deviation of the approximated Gaussian-distributed LLR values. The requirement of fewer samples was also observed in practice. Therefore, less time was required to simulate the approximation curves. The results are quite good and comparable to the bit error performance of the code obtained via conventional simulation methods.

The approximation is very good in the area of interest to most designers. The error probabilities in the range of 10^{-3} to 10^{-4} are important when considering the transmission of analog signals (e.g. speech). Generally, these bit error values are associated with low

SNR values and therefore, the approximation is appropriate for the range over which it will be most useful.

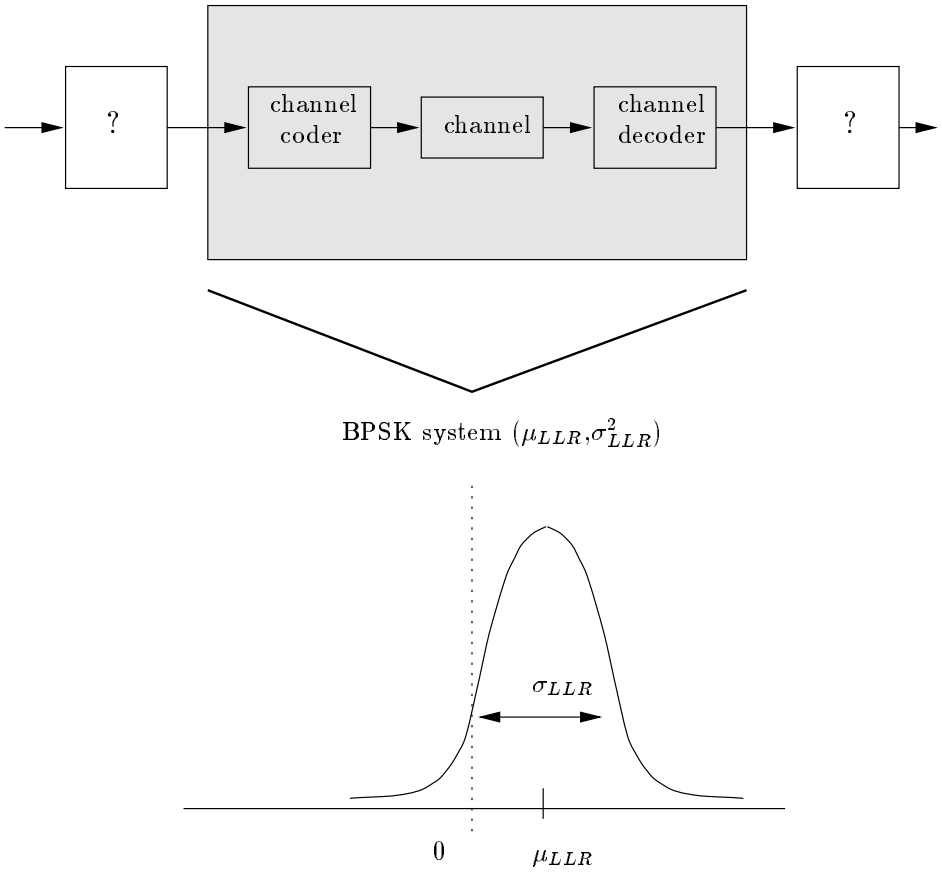

Figure 6.5: Channel Coding Components Replaced by a BPSK System

The close approximation of the bit error performance of a binary linear code justifies the modeling of the soft-output decoding of binary linear codes by a BPSK system. With this model, it is possible to replace the a complex channel coder, channel, and channel decoder by a BPSK system operating with signal points at $\pm\mu_{LLR}$ and noise variance equal to σ^2_{LLR} as seen in figure 6.5. The simple BPSK system is far less complicated and once the mean and variance of LLR values have been calculated, the original channel

coding structure can be replaced.

This type of simplification to the channel model simplifies the life of a designer of a system which requires knowledge of the channel's characteristics. Such is the case for the designer of a combined source-channel coding scheme [31]. Channel characteristics should be known such that the quantizers in the source coder can be optimized. The simple model allows the designer to concentrate more fully on this task.

6.5 Chapter Summary

This chapter presented the results of simulations of the bit error performance of two binary linear codes: Reed-Muller and Golay codes. The approximation described and analyzed in the preceding chapters and the modeling of soft-output LLR values using a BPSK system was used and compared against the conventionally simulated soft-output bit error performance of the linear codes. The approximation is found to be excellent for low SNR and deviates at higher SNR values. This was expected since the higher-order terms of the Taylor series expansion cannot be neglected at higher SNR values (i.e., low noise variance) and the Gaussian approximation falters. The approximation method requires fewer samples to obtain such close performance, as well.

Therefore, it was shown that complex channel coding and decoding systems can be replaced by a simple BPSK system, thereby simplifying the system interactions and allowing designers to concentrate on other components of the communications systems. The BPSK system would be characterized by signal points at $\pm\mu_{LLR}$ and noise variance of σ^2_{LLR}.

This ends the discussion on the modeling of the soft-output decoding of linear block codes using a BPSK system. The next chapter presents an interesting methodology for calculating the weight distribution of a code, or the number of codewords with a given

weight (i.e., number of ones). The weight distribution of a code can be used directly in the expression of chapter 2, in calculating the coefficients of the Taylor series expansion.

Chapter 7

Weight Distribution Using the Discrete Fourier Transform

The Hamming weight of a codeword is defined as the number of nonzero elements in the codeword [3, pp. 376]. For a given code, the various Hamming weights of the codewords form the weight distribution of the code. This distribution is typically presented via a weight enumeration function, which is a table comprised of the number of codewords which have a certain Hamming weight. Knowledge of the weight distribution of a linear code is important in carrying out an error performance analysis. Due to this fact, numerous research works have addressed the problem of computing the weight distribution of general or specific code constructions.

The techniques known for computing the weight distribution of a general linear code are based on representing the code by a state diagram in the case of convolutional codes [32,33], or by a trellis diagram[1] in the case of block codes [2,34–38]. These methods are based on assigning a partial weight enumeration function to the transitions of a state

[1] A trellis diagram differs from a state diagram in that a time axis is associated with the transitions.

(or trellis) diagram, where the partial weight distributions are appropriately multiplied and summed (reflecting the concept of state in traversing the allowed paths) to yield the complete weight distribution of the code. Similar computational techniques have been used in conjunction with constrained coding systems as well [39].

The focus of this chapter is to present the use of the Discrete Fourier Transform (DFT) to calculate the weight distribution of a linear block code using a modified state transition matrix. The matrix is modified in such a manner as to include the contribution in weight (by the input and output bits) for each state-to-state transition. This method can be used to calculate the coefficients of the terms of the Taylor series expansion of chapter 2. With the coefficients, the approximation of the earlier chapters can be calculated directly using the expression of (2.47).

The chapter is organized as follows. Section 7.2 presents background information on state transition matrices and the information they encapsulate. Section 7.3 presents the formation of the modified state transition matrix, called the *weighted state transition matrix*. The use of the Fourier analysis is presented in Section 7.4 to calculate the weight enumeration function coefficients of a general binary linear code through examples of a recursive convolutional code and single-parity check codes. Also, standard weight enumeration function notation is presented and the calculation of the weight enumeration function of a parallel concatenated code is given. The application of the weight enumeration function to the calculation of a bound on bit error probability is illustrated in Section 7.5. Once the weight distribution is known, it can directly be used in the calculation of the Union bound for the probability of error. Finally, the chapter is concluded with a discussion of this method's advantages and disadvantages in Section 7.6.

7.1 Notational Changes

In the discussion that follows, two key notational changes have been made from the previous chapters of this thesis. Firstly, all bold capitalized characters in equations now refer to matrices, rather than the lower-cased vectors of earlier chapters. Secondly, no references are made to random variables in this chapter, as well. Thirdly, the variable N is now the number of transitions through a trellis, and is, therefore, the number of input bits used to produce a codeword of a given code. This is different from the earlier definition where N was defined to be the number of samples considered in finding the mean and variance of the LLR. Keeping these changes in mind, confusion can be avoided in the following sections.

7.2 State Transitions Matrices

It is known that if a code can be represented by a trellis, it can equivalently be represented by a state diagram. Using either of these forms, the state-to-state transitions of a code with varying input, producing different output, are known. Consider a trellis $\mathcal{T}$ with K states, $s_0, \ldots, s_{K-1}$, where each transition between a pair of states (s_i, s_j) is distinguished by one or several input bit(s), as well as one or several output bit(s). A state transition matrix of dimensions $K \times K$ can be defined for the trellis, where the existence of a state-to-state transition denoted by a '1' in the appropriate location. The K rows of the matrix can each represent beginning in one of the K states, and the K columns can represent ending in any state. For instance, the element in location (1,2) of the matrix is associated with starting in state 1 and ending in state 2.

Once the transition matrix has been formed, it is easy to obtain the number of paths from one state to another state for a given set of N input transitions by raising this transition matrix to the power N. The resulting matrix will contain element values

which represent the number of paths that exist between any two states for the N input transitions [39]. However, there is no indication of the weights associated with the bits of these paths. Using this simple technique, and making some modifications, the number of paths through a trellis of a given input weight and given parity (output) weight will be found. The bits on each path form a codeword of the code being considered of a specific input and output weight. The modifications to the state transition matrix produce a *weighted state transition matrix.*

7.3 Weighted State Transition Matrices

As the name would suggest, each entry in the matrix would not only represent the presence of a transition from state-to-state, but would also incorporate the weight of the associated input and output bits of that transition. The formation of the matrix is now presented, and the method to represent the weight of the transition is clearly explained.

By considering the trellis $\mathcal{T}$, partial state transition matrices are defined as a set of $K \times K$ matrices $\mathbf{T}_{m,n}^{(k)}$, where the (i,j)th element of $\mathbf{T}_{m,n}^{(k)}$, namely $T_{m,n}^{(k)}(i,j)$, is equal to the number of transitions of input weight m and output weight n between states i and j, after k transitions. The placement of a '1' in any position of the partial matrices dictates that a transition of that weight exists.

$\mathbf{T}_{m,n}^{(k)}$, $m,n = 0,1,\ldots$, will be considered as two-dimensional discrete series elements. Through the convolution of the $K \times K$ matrices, and using the two-dimensional discrete Fourier transform defined below, the correctness of the method is addressed.

The number paths through the trellis after k transitions between states i and j with input weight, m, and output weight, n, can be found using the expression,

$$T_{m,n}^{(k)}(i,j) = \sum_{p} \sum_{q} \sum_{\alpha} T_{m-p,n-q}^{(k-1)}(i,\alpha)\, T_{p,q}^{(1)}(\alpha,j). \tag{7.1}$$

In matrix form, the number of paths is found using the following recursive relationship.

$$\mathbf{T}^{(k)}_{m,n} = \sum_{p} \sum_{q} \mathbf{T}^{(k-1)}_{m-p,n-q}\, \mathbf{T}^{(1)}_{p,q}. \tag{7.2}$$

The operation in (7.2) involves the multiplication and accumulation of paths through the trellis of given input and output weights, and is indeed the convolution of $\mathbf{T}^{(k-1)}_{m-p,n-q}$ and $\mathbf{T}^{(1)}_{p,q}$. The weighted state transition matrix, $\mathbf{X}(u,v)$, is defined as

$$\mathbf{X}(u,v) = \sum_{m=0}^{L_1-1} \sum_{n=0}^{L_2-1} \mathbf{T}_{m,n} U^{mu} V^{nv}, \tag{7.3}$$

where,

$$U = \exp\left(-\frac{j2\pi}{L_1}\right),\ V = \exp\left(-\frac{j2\pi}{L_2}\right),\ j = \sqrt{-1}, \tag{7.4}$$

and L_1, L_2 are selected as arbitrary integers larger than the maximum possible input weight and maximum possible output weight, respectively, to avoid aliasing. We usually have $L_1 = L_2$ resulting in $U = V$, in which case L^2 is used to represent the common value of $L_1 = L_2$ and W to represent the common value of $U = V$. Note that W is a transform variable, similar to that used in the discrete Fourier Transform [40,41](DFT). Also, note that $\mathbf{T}^{(1)}_{m,n}$ and $\mathbf{X}(u,v)$ are related through the discrete Fourier transform, i.e.

$$\mathbf{T}^{(1)}_{m,n} \overset{\mathcal{F}}{\longleftrightarrow} \mathbf{X}(u,v). \tag{7.5}$$

The matrix $\mathbf{X}(u,v)$ contains information of the weights of transitions through a trellis, with varying input weights u, and output weights v, and is therefore called the weighted state transition matrix.

[2] N, the block length of the code, is a convenient choice for the value of L.

Using $\mathbf{X}(u,v)$ defined above, the number of paths from one state to another state can be found, for a given input weight, m, and given output weight, n. This is done using an inverse type of transform similar to the inverse DFT [41] and is shown in the next section.

7.4 Fourier Analysis to Obtain Coefficients

It can be shown that the discrete Fourier transform of the convolutional operation in (7.2) yields a product of the discrete Fourier transforms of the two transitions matrices. Furthermore, by recursively applying this property, it can be established that

$$\mathbf{T}_{m,n}^{(N)} \overset{\mathcal{F}}{\longleftrightarrow} \mathbf{X}^N(u,v), \tag{7.6}$$

where $\mathcal{F}$ denotes the discrete Fourier transform operation.

Therefore, using this property and the orthogonality property of the Fourier operator, the matrix $\mathbf{X}(u,v)$ is raised to the power N (to encapsulate that N transitions have occurred) in order to compute the weight enumeration function coefficients over N consecutive stages of this trellis. The inverse transform is then applied to the N-raised weighted state transition matrix. The results of this operation is a matrix $\mathbf{A}_{m,n}$, with elements which indicate the number of paths of given weight from any starting state of the trellis to any ending state. $\mathbf{A}_{m,n}$ is computed, using the expression of (7.3), as,

$$\mathbf{A}_{m,n} = \frac{1}{L_1 L_2} \sum_{u=0}^{L_1-1} \sum_{v=0}^{L_2-1} \mathbf{X}^N(u,v) U^{-mu} V^{-nv}. \tag{7.7}$$

The (i,j)th element of the $K \times K$ matrix $\mathbf{A}_{m,n}$ indicates the number of paths of input weight m and output weight n starting at state i and ending at state j after traversing N consecutive stages of the trellis. By setting values of n and m, such that $0 \leq n \leq N$

and $0 \leq m \leq N$, the number of paths of different weights are obtained.

The main computational step in computing (7.7) is to raise the $K \times K$ matrix $\mathbf{X}(u, v)$ to the power of N. This can be achieved easily by using an eigenvalue decomposition of $\mathbf{X}(u, v)$ and raising the eigenvalues to the power of N [42].

In general, the entries of matrix $\mathbf{A}_{m,n}$ have an exponential growth with N. As a result, for large values of N, one may encounter numerical difficulties in using (7.7). This problem can be easily handled by performing the calculations on shorter sub-blocks, truncating the resulting partial weight distributions, combining the results through multiplication of the corresponding weighted state transition matrices, and finally performing the inverse Fourier operation on the result. Note that similar precautions are needed in any other method used to compute the weight distribution.

For a linear block code, it is required that the trellis begins and ends in the 'zero' state. This usually corresponds to the element found in location $(0, 0)$ of the matrix $\mathbf{A}_{m,n}$. The element in the $(0, 0)$ location of $\mathbf{A}_{m,n}$ will simply be referred to as $A_{m,n}$ for the remainder of the chapter. The other entries of the matrix provide the weight distributions of the cosets of this linear code.

The above formulation accounts for the contributions of the input and output weights, separately. In some situations, only the weight of the output may be of interest, in which case the variable u can be omitted in (7.3) and (7.7), and the Fourier transform pair can be expressed in terms of a single summation, as in the pair of equations below.

$$\mathbf{X}(v) = \sum_{n=0}^{L-1} \mathbf{T}_n^{(1)} \, W^{nv}, \tag{7.8}$$

$$\mathbf{A}_n = \frac{1}{L} \sum_{v=0}^{L-1} \mathbf{X}^N(v) \, W^{-nv}. \tag{7.9}$$

The mathematics of the method are straightforward and the method can be easily

implemented using commercially available software packages. The calculation of raising $\mathbf{X}$ to the N^{th} power is not difficult to carry out, using eigenvalue decomposition and similar matrix properties. Again, software applications exist to do this efficiently. Computational complexity and memory requirements are not concerns here as the main objective of the method is to provide for an easily-implementable methodology.

This methodology is quite versatile and can be applied to any code which is representable by a trellis diagram, including convolutional codes, Turbo codes, and many other linear block codes. For the case of Turbo codes, the coefficients of conditional weight enumeration functions can be easily calculated for the error performance analysis [2]. A simple example is presented below to illustrate the formation of the partial state transition matrices and the weighted state transition matrix of a recursive convolutional code.

7.4.1 Weighted State Transition Matrix Formation Example

Consider the simple $(5,7)_8$ recursive convolutional code, where 5_8 represents the taps on the memory elements for the output bits, and 7_8 represents the feedback taps. This is a 2 memory element code, with 4 states. The state diagram is as shown below in figure 7.1. From the state diagram, the following partial state transition matrices are formed as,

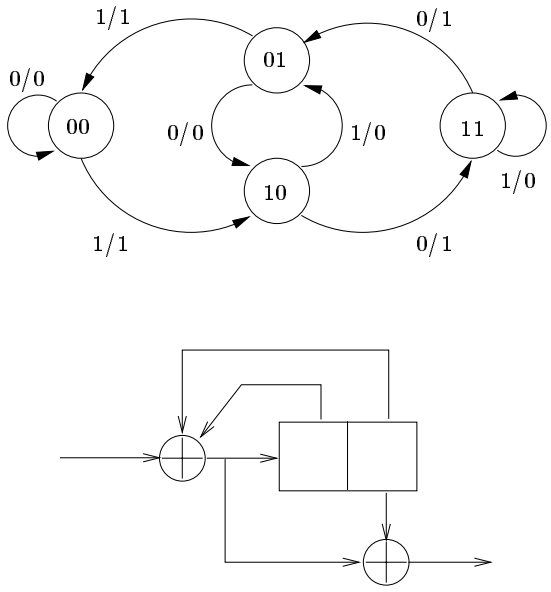

Figure 7.1: State Diagram of $(5,7)_8$ Recursive Convolutional Code

$$\mathbf{T}_{0,0}^{(1)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{T}_{0,1}^{(1)} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$

$$\mathbf{T}_{1,0}^{(1)} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{T}_{1,1}^{(1)} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{7.10}$$

Using (7.3),

$$\mathbf{X}(u,v) = \begin{bmatrix} 1 & 0 & W^{u+v} & 0 \\ W^{u+v} & 0 & 1 & 0 \\ 0 & W^{u} & 0 & W^{v} \\ 0 & W^{v} & 0 & W^{u} \end{bmatrix}, \tag{7.11}$$

is obtained, where,

$$W = \exp\left(-\frac{j2\pi}{L}\right), \quad L > N. \tag{7.12}$$

Using (7.7) and (7.11), the weight distribution of the code can be found.

The above example accounts for both the input and output weights of the paths through the trellis. Neglecting the contributions of u, only the weights of the output bits are found, as in the following example.

7.4.2 Simple Example of the Method

Consider a simple $(N, N-1)$ single-parity check code. Using the state diagram of the code, provided in figure 7.2, and using (7.8), the following weighted state transition matrix is obtained, considering the output weights only.

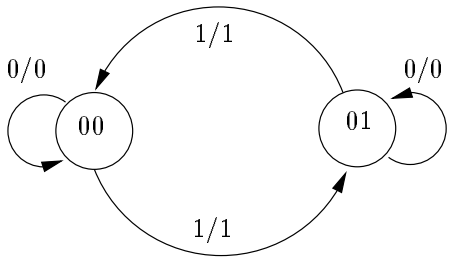

Figure 7.2: State Diagram of a Generic Single-Parity Check Code

$$\mathbf{X}(v) = \begin{bmatrix} 1 & W^v \\ W^v & 1 \end{bmatrix}, \tag{7.13}$$

where W is defined as in example 7.4.1. In (7.9), the coefficient is calculated by raising $\mathbf{X}(v)$ to the N^{th} power and the inverse DFT is then performed. The eigenvalues of $\mathbf{X}(v)$, with corresponding eigenvectors, can be verified to be,

$$\lambda_1 = 1 - W^v \qquad p_1 = \begin{bmatrix} 1 & -1 \end{bmatrix}^T, \tag{7.14}$$

$$\lambda_2 = 1 + W^v \qquad p_2 = \begin{bmatrix} 1 & 1 \end{bmatrix}^T. \tag{7.15}$$

Therefore, with these quantities, $\mathbf{X}^N$ can be calculated as

$$\mathbf{X}^N = \mathbf{P}\mathbf{\Lambda}^N\mathbf{P}^{-1} \tag{7.16}$$

where, $\mathbf{P} = [\, p_1 \; p_2 \,]$ and $\mathbf{\Lambda} = diag(\lambda_1, \; \lambda_2)$. Using (7.9), the coefficients can be calculated for different n values. Implementing this procedure for the $(5, 4)$ single-parity check code, the weight enumeration below is obtained.

Weight	Weight coefficient, A_n
0	1
2	10
4	5

The validity of this method has been verified by computer calculations and by comparing the resulting weight distributions of various codes to those obtained by-hand calculation or to those found in literature.

7.4.3 Weight Enumeration Functions

In order to represent the function in closed form, a notation is adopted where dummy variables are used to represent the weight of a code. These dummy variables can be viewed as terms of a polynomial, where $A_{m,n}$ is the coefficient of that term.

The conventional weight enumeration function of a generic systematic (N, k) linear block code is given by

$$A_{conv}(H) \triangleq \sum_{d=0}^{N} A_d H^d \tag{7.17}$$

where A_d is the number of codewords with Hamming weight d, and H is the dummy variable used in this representation, similar to what was defined in section 2.3.3. The individual contributions of the input bits are not clearly stated.

The conventional weight enumeration function can be calculated using the methodology above, by forming partial state transition matrices for the sum of the weights of the transitions. The separate contributions of the input and output bits can be represented by w in the weighted state transition matrix, taking into account for the largest possible value of w being $2N$ with the definition of W. With this change, the inverse DFT can be carried out with a single summation to obtain the coefficients of the weight enumeration function.

The separate contributions of the input and output bits are not evident with the conventional weight enumeration function and thus prompted Benedetto and Montorsi to define the *input-redundancy weight enumerating function* (IRWEF) of the code, C, as,

$$A^C(V, Z) = \sum_{m,n} A_{m,n} G^m H^n \ . \tag{7.18}$$

G and H are the dummy variables for the input and output weights. Here, the overall

Hamming weight of the path, or codeword, is therefore $d = n + m$. The separate contributions of the input and output bits to the total Hamming weight of the codeword are made explicit. This change was shown to be crucial for dealing with parallel concatenated codes, such as turbo codes.

7.4.4 Weight Enumeration Function for Parallel Concatenated Codes

Since the concatenated parity bits from the constituent codes are produced by the same input bit-stream, the enumeration functions of the constituent codes in the concatenation must be combined in such a manner so as to reflect this fact. This necessitated the definition of the conditional weight enumeration function.

Conditional weight enumeration functions have the form

$$A_m^C(Z) = \sum_n A_{m,n} Z^n. \tag{7.19}$$

It is conditional in the sense that the output bit weights only correspond to input bits of weight m. Thus, for a given input weight m, the combinations for the parity bits of the constituent codes can be found by multiplying their conditional weight enumeration functions.

As an aside, it is interesting to note that the IRWEF can therefore be obtained from the conditional weight enumeration as follows to obtain (7.7).

$$A^C(V, Z) = \sum_m V^m A_m^C(Z) \tag{7.20}$$

In the case of turbo codes, an inter-leaver is used to permute the input bits between the two constituent recursive convolutional codes, here called C_1 and C_2. It was theorized that if a uniform inter-leaver of length N was used to permute the information bits for the

second encoder C_2, then the second code is independent of the first code C_1 [2, pp. 412]. The conditional weight enumeration function of the parallel concatenated code becomes

$$A_m^{C_p}(Z) = \frac{A_m^{C_1}(Z) \cdot A_m^{C_2}(Z)}{\binom{N}{m}} \tag{7.21}$$

where $A_m^{C_1}$ and $A_m^{C_2}$ are conditional weight enumerating functions of the parity check bits produced by input words of weight m, and division by $\binom{N}{m}$ presents the uniform nature of the inter-leaver. With this background on the determination of the weight enumeration functions of parallel concatenated codes (PCC), the use of the methodology presented in this chapter above can be applied to PCCs below.

Consider a code with weighted transition matrix $\mathbf{X}_{C_1}(u, v)$ formed in the same manner as in (7.3). By considering the inverse DFT over only the u variable, the resulting matrix dictates the transitions of varying output weight v for a given input weight m. All the possible output weight transitions for the weight m are described by

$$\mathbf{Y}_m(v) = \frac{1}{N+1} \sum_{u=0}^{N} \mathbf{X}^N(u, v) W^{-mu} \tag{7.22}$$

Here, $L = N$ was chosen for convenience and W is as defined in (7.12). This is the conditional weight enumerating function in the transform domain. Following from (7.21), the conditional WEF in the transform domain can be obtained for the parallel concatenation of the two codes. This is to say that,

$$\begin{aligned} \mathbf{Y}_m^{C_p}(v) &= \frac{\mathbf{Y}_m^{C_1}(v) \cdot \mathbf{Y}_m^{C_2}(v)}{\binom{N}{m}} \\ &= \frac{1}{\binom{N}{m}} \frac{1}{(N+1)^2} \sum_{u=0}^{N} \sum_{u'=0}^{N} \mathbf{X}_{C_1}^N(u, v) \otimes \mathbf{X}_{C_2}^N(u', v)\, W^{-m(u+u')} \end{aligned} \tag{7.23}$$

where the operator $\otimes$ calls for an element-by-element matrix multiplication. To obtain

the number of paths (codewords) with a given input weight m and output weight n, simply take the inverse transform of $\mathbf{Y}_m^{C_p}(v)$ for a given value of m. Assuming that the number of output bits is equal to the number of systematic bits for each constituent code (usually the case for turbo codes), the number of paths of weight m, n is given by

$$\begin{aligned}\mathbf{A}_{m,n}^{C_p} &= \frac{1}{2N+1}\sum_{v=0}^{2N}\mathbf{Y}_m^{C_p}(v)\mathcal{W}^{-nv} \\ &= \frac{1}{\binom{N}{m}}\frac{1}{(2N+1)(N+1)^2}\sum_{v=0}^{2N}\sum_{u=0}^{N}\sum_{u'=0}^{N}\mathbf{X}_{C_1}^N(u,v)\otimes\mathbf{X}_{C_2}^N(u',v)\cdot \\ &\qquad W^{-m(u+u')}\mathcal{W}^{-nv}\ . \end{aligned} \tag{7.24}$$

Here, $0 \le m \le N$ and $0 \le n \le 2N$, and

$$\mathcal{W} = \exp\left(-\frac{j2\pi}{L}\right), \quad L > 2N. \tag{7.25}$$

Obtaining the value in the (0,0) location of the matrix $\mathbf{A}_{m,n}^{C_p}$ yields the number of codewords with input weight m, and output weight n for the parallel concatenated code.

7.5 Bound on Bit Error Probability and the Weight Enumeration Function

The bit error probability of a code can be upper bounded by the Union bound. From [2], the probability of bit error is bounded by

$$P_b(e) \le \frac{1}{2}\sum_d D_d\ erfc\left(\sqrt{d\frac{R_c E_b}{N_0}}\right) \tag{7.26}$$

where the term D_d is defined as

$$D_d \triangleq \sum_{m+n=d} \frac{m}{N} A^C_{m,n}, \tag{7.27}$$

R_c is the defined as the code rate, and E_b is the energy per bit. Note that the number of input bits is still N. This holds for any code, C, with weight enumeration function $A^C_{m,n}$.

For turbo codes specifically, the expression is found by combining (7.24), and the Union bound in (7.26), with D_d defined in (7.27). The probability of bit error for a given signal-to-noise ratio can be found using the resulting expression in (7.28) below.

$$P_b(e) \leq \frac{1}{2} \sum_{n=0}^{N} \sum_{m=0}^{2N} \frac{1}{(N+1)^2 (2N+1)} \frac{1}{\binom{N}{n}} \sum_{v=0}^{2N} \sum_{u=0}^{N} \sum_{u'=0}^{N} \mathbf{X}^N(u,v)_{(0,0)} \cdot \mathbf{X}^N(u',v)_{(0,0)} \cdot W^{-m(u+u')} \mathcal{W}^{-nv} \frac{m}{N} erfc\left(\sqrt{(m+n)R_c \frac{E_b}{N_0}}\right) \tag{7.28}$$

$\mathbf{X}^N(u,v)_{(0,0)}$ denotes the $(0,0)$ location of the matrix $\mathbf{X}^N(u,v)$.

7.6 Advantages and Disadvantages of the DFT Method

The presented methodology works well, and produces the correct number of codewords of a given input and output weight. The proposed method has been verified by computer simulations and the results have been compared with those found in literature. This method has the advantage of being able to calculate the number of paths of a given weight, without having to traverse the trellis or carry out tedious analytical work. The number of codewords at a given distance can be helpful for determining the contribution of low weight codewords to the probability of error. As well, the method can be used to calculate the weight distributions of the cosets of the linear code.

As one can see, this method involves many summations, which tend to be time consuming and computationally intensive for large block lengths. This methodology is well suited for codes of shorter lengths or on shorter sub-blocks. However, since this is an inverse DFT, it is believed that special optimized algorithms exist [40, 41] to obtain the results quickly and is therefore not prohibitive to use.

7.7 Chapter Summary

This chapter presented a systematic method to calculate the weight distribution of a linear block code expressed in terms of its trellis structure. By using the weighted state transition matrix of the code, the number of codewords of a given input and output weight can be found methodically using a simple implementable equation. The method not only provides a simple methodology, but is useful to calculate the number of codewords of a certain Hamming weight, without having to calculate the entire weight distribution of the code, as is done currently. The proposed method is general, easy to implement, and unlike other known methods, does not require traversing the trellis or performing tedious analytical work.

Chapter 8

Conclusions and Future Research

Through the course of this thesis, two major concepts have been presented: the modeling of the soft-output decoding of binary linear codes using a Binary Phase Shift Keying (BPSK) modulated system; and a new methodology to calculate the weight distribution was described which uses the discrete Fourier Transform on a weighted state transition matrix of the linear code. The conclusions obtained from these works are presented in the next two sections, followed by future research avenues made available to other researchers using the results of this thesis.

8.1 Conclusions

8.1.1 Equivalence Between Soft-Output Decoding of Binary Linear Codes and a BPSK System

This thesis presented the modeling of the soft-output decoding of binary linear codes using a BPSK system. A common soft-output measure used to help in the decoding of a received codeword is the Log Likelihood Ratio (LLR). The LLR is defined as the log of the ratio of *a posteriori* probabilities of a given transmitted bit being 0 or 1. By

obtaining the Taylor series expansion of an appropriately defined LLR, and generalizing some results found in literature, the distribution of the LLR can be approximated to be Gaussian for high noise variance values. It is seen that higher-order summations of the products of random variables in the Taylor series expansion possess an asymptotic Gaussian distribution, adding to the overall Gaussian approximation. With Gaussian distributed LLR values, the soft-output decoding systems using the LLR for decoding could be modeled as a BPSK system. The mean and variance of sample LLR values can be calculated and then used with the probability of bit error expression of a BPSK system to obtain an approximation to the bit error performance of the code. This involved forming the ratio of the mean of the LLR values to the standard deviation of the LLR values. Alternately, the expressions derived for the approximated LLR can be used to calculate its mean and variance using the weight distribution of the code. However, the simulation to obtain sample LLR values was the method used in this thesis.

Through an in-depth analysis of the estimators of the mean and variance, the probability density function of the ratio of the two estimators was found. This was done by noting that the two estimators are uncorrelated, and in the special case of Gaussian samples, the two estimators are independent. Therefore, the product of their probability density functions and the Jacobian of the variable transformations were used to find the probability density function of the ratio.

The use of the ratio in the bit error probability expression for a BPSK system required the determination of the variance of this quantity so that a comparison with the variance of the bit error probability obtained through conventional simulation methods could then be made. The conventional simulation method is one where a hard decision is made on the LLR of a bit and then the errors are totaled to obtain an estimate of the probability of bit error for the code. The variance of the approximation method was always shown to be smaller than that of the conventional method for a given number of samples N and

at a given bit error probability. This implied that fewer samples were needed to obtain the same precision in the bit error probability values. The requirement of fewer samples translates into a time-savings in obtaining performance results, and the reduction factors can be quite substantial.

Simulations results were shown for two codes (Reed-Muller and Golay codes) which were remarkably close to the conventionally simulated bit error probability curves. The curves of the two methods were closer together for the low SNR values, as was expected due to the approximation made earlier (higher-order terms cannot be neglected with decreasing noise variance values, i.e. at higher SNR). Although there was divergence at higher SNR, the approximation was still quite good. Comparing this method to another by Chao *et al.*, this approximation provides more robust results.

With these results, complex channel coding schemes can be replaced with simpler BPSK models with the known mean and variance of soft- output LLR values. The designers of systems requiring knowledge of the channel characteristics can use this simple channel coding model.

8.1.2 Weight Distribution using the DFT

Another contribution made by this thesis is the calculation of the weight distribution of a code based upon the discrete Fourier Transform (DFT) of a weighted state transition matrix. Each linear code can be represented by a state transition matrix, where a non-zero entry in the matrix represents the existence of a transition between different states of the code. A state transition matrix was defined where the weights of the inputs and outputs of the transitions are included in the representation. By performing the inverse DFT on the matrix raised to the required length of the codeword, the number of codewords of a given input and output weight resulted.

The advantage of such a methodology is that the number of codewords of a specific

weight can be found, rather than having to calculate the entire weight distribution of the code. Also, the methodology is straight forward and can be applied to many codes mathematically (rather than through computer simulation of the code). This method is better suited to codes of small to moderate length as the algorithm can require many summations and also to avoid numerical difficulties.

8.2 Future Research

There are many possible research avenues open to researchers to pursue in relation to the contributions made by this thesis.

8.2.1 Equivalence Between Soft-Output Decoding of Binary Linear Codes and a BPSK System

Only the tip of the iceberg for this topic has been explored here. There are many different directions in which the research can be extended. These can include:

- the investigation into the application of the results to convolutional codes using soft-output decoding techniques;
- use of the simplified BPSK channel model in the design of systems requiring knowledge of the channel. For example, in combined source-channel coding where the channel coding structure and channel can be replaced with a BPSK channel model with transmitted amplitude of μ_{LLR} and noise variance of σ^2_{LLR}; and
- further study of the approximation obtained by truncating the Taylor series expansion of the LLR and obtaining an expression to calculate the mean and variance of the LLR based upon the truncated series.

8.2.2 Weight Distribution using Inverse DFT

Future research which makes use of the methodology outlined for calculating the weight distribution of a code is first and foremost. The applications of the method is well suited for use in finding bounds for codes of small length. It is also possible to use this method in calculating the bound on error for schemes employing tailbiting schemes for convolutional codes, where the number of paths from one state, ending in the same state after N transitions is important. The methodology can also be adjusted to include the effects of multiply collapsed sections of a trellis.

It is known that the procedure of raising a matrix to a given power can be carried out by using the eigenvalues and eigenvectors of the matrix. Closed form expressions for the eigenvalues and eigenvectors can be useful, and studying the effects of only considering the largest eigenvalue to the power of the block length can be further explored. This may simplify the procedure outlined in this thesis.

Appendix A

Mean and Variance of the Sample Estimators

In this appendix, the mathematics behind the determination of the mean and variance of the estimators is presented. The samples used, x_i, $i = 1, 2, \ldots, N$, are independent Gaussian samples, with mean μ and variance σ^2.

Firstly, the mean and variance of the mean estimator M are presented, followed by the mean and variance of the variance estimator V^2. Lastly, the lack of correlation between the two estimators is shown. Independence is implied due to the consideration of the quantities as Gaussian random variables with increasing number of samples, N. Please note that $E[\cdot]$ denotes taking the expectation of the argument.

A.1 Mean and Variance of the Mean Estimator M

The mean estimator is given by equation (3.1) and is reproduced below for convenience.

$$M,\ \overline{x} = \frac{1}{N}\sum_{i=0}^{N} x_i \tag{A.1}$$

The mean of (A.1), μ_M, is formulated below as

$$\begin{aligned}
\mu_M &= \overline{\frac{1}{N}\sum_{i=0}^{N} x_i} \\
&= \frac{1}{N}\sum_{i=0}^{N} \overline{x_i} \\
&= \frac{1}{N}\sum_{i=0}^{N} \mu \\
&= \mu.
\end{aligned} \tag{A.2}$$

This is the expression of the mean that is found in (3.3).

The variance of (A.1), σ_M^2, is also found in a similar manner. Forming the expression for the variance, and solving,

$$\begin{aligned}
\sigma_M^2 &= E[(M-\mu_M)^2] \\
&= E[M^2] - (\mu_M)^2 \\
&= \overline{\Big(\frac{1}{N}\sum_{i=0}^{N} x_i\Big)^2} - \mu^2 \\
&= \frac{1}{N^2}\Big(\sum_{i=0}^{N}\overline{x_i^2} + \sum_{\substack{i=0\\ i\neq j}}^{N}\sum_{j=0}^{N}\overline{x_i\,x_j}\Big) - \mu^2
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{N^2}\Big(\sum_{i=0}^{N}(\mu^2+\sigma^2) + \sum_{\substack{i=0\\ i\neq j}}^{N}\sum_{j=0}^{N}\overline{x_i}\,\overline{x_j} - N^2\mu^2\Big) \\
&= \frac{1}{N^2}\Big(N\mu^2 + N\sigma^2 + N(N-1)\mu^2 - N^2\mu^2\Big) \\
&= \frac{1}{N^2}\big(N\sigma^2\big) = \frac{\sigma^2}{N}. \qquad \text{(A.3)}
\end{aligned}
$$

Again, this is the expression of the variance of M found in (3.3).

A.2 Mean and Variance of the Variance Estimator V^2

The variance estimator is given by equation (3.2) and is reproduced below for convenience.

$$V^2 = \frac{1}{N}\sum_{i=0}^{N}(x_i - \overline{x})^2 \qquad \text{(A.4)}$$

The expressions for the mean and variance of the variance estimator are more complicated since the mean estimator of $\overline{x}$ is needed in the determination of the variance estimate. The mean of (A.4), μ_{V^2}, is expressed below as

$$
\begin{aligned}
\mu_{V^2} &= \overline{\frac{1}{N}\sum_{i=0}^{N}(x_i - \overline{x})^2} \\
&= \frac{1}{N}\sum_{i=0}^{N}\overline{\big(x_i^2 - 2x_i\overline{x} + \overline{x}^2\big)} \\
&= \frac{1}{N}\sum_{i=0}^{N}\big(\overline{x_i^2} - 2\overline{x_i\overline{x}} + \overline{\overline{x}^2}\big) \\
&= \frac{1}{N}\sum_{i=0}^{N}\left((\sigma^2+\mu^2) - 2\overline{\Big[x_i\frac{1}{N}\sum_{l=0}^{N}x_l\Big]} + \overline{\Big[\frac{1}{N}\sum_{j=0}^{N}x_j\Big]^2}\right) \\
&= \frac{1}{N}\sum_{i=0}^{N}\left((\sigma^2+\mu^2) - \frac{2}{N}\big[\sigma^2+\mu^2+(N-1)\mu^2\big] + \frac{1}{N^2}\overline{\Big[\sum_{j=0}^{N}x_j^2 + \sum_{\substack{j=0\\ j\neq l}}^{N}\sum_{l=0}^{N}x_jx_l\Big]}\right)
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{N}\sum_{i=0}^{N}\Big(\sigma^2+\mu^2-\frac{2}{N}[\sigma^2+\mu^2]-\frac{2(N-1)}{N}\mu^2+\frac{1}{N^2}\big[N(\sigma^2+\mu^2)+N(N-1)\mu^2\big]\Big)\\
&= \sigma^2+\mu^2-\frac{2}{N}(\sigma^2+\mu^2)-\frac{2(N-1)}{N}\mu^2+\frac{1}{N}(\sigma^2+\mu^2)+\frac{N-1}{N}\mu^2\\
&= \big(1-\frac{2}{N}+\frac{1}{N}\big)\sigma^2+\mu^2\big(1-\frac{2}{N}-\frac{2(N-1)}{N}+\frac{1}{N}+\frac{N-1}{N}\big)\\
&= \frac{N-1}{N}\sigma^2. \qquad \text{(A.5)}
\end{aligned}
$$

This is the expression of the mean that is found in (3.4).

The variance of (A.4), $\sigma^2_{V^2}$ is also found in a similar manner. Forming the expression for the variance, and solving,

$$
\begin{aligned}
\sigma^2_{V^2} &= E[(V^2-\overline{V^2})^2]\\
&= E\big[\big(V^2\big)^2\big]-\big(E[V^2]\big)^2\\
&= E\Big[\big(\frac{1}{N}\sum_{i=0}^{N}(x_i-\overline{x})^2\big)^2\Big]-\Big[\frac{(N-1)}{N}\sigma^2\Big]^2 \qquad \text{(A.6)}
\end{aligned}
$$

is obtained. Taking the first expression in (A.6) and expanding it separately,

$$
\begin{aligned}
E\big[\big(V^2\big)^2\big] &= E\Big[\big(\frac{1}{N}\sum_{i=0}^{N}(x_i-\overline{x})^2\big)^2\Big]\\
&= E\Big[\frac{1}{N^2}\Big(\sum_{i=0}^{N}(x_i^2-2x_i\overline{x}+\overline{x}^2)\Big)^2\Big]\\
&= \frac{1}{N^2}E\Big\{\big(\sum_{i=0}^{N}x_i^2\big)^2+\big(2\overline{x}\sum_{i=0}^{N}x_i\big)^2+\big(\sum_{i=0}^{N}(\overline{x})^2\big)^2-4\overline{x}\sum_{i=0}^{N}x_i\sum_{j=0}^{N}x_j^2\\
&\qquad +2\sum_{i=0}^{N}\overline{x}^2\sum_{j=0}^{N}x_j^2-4\overline{x}\sum_{i=0}^{N}x_i\sum_{j=0}^{N}\overline{x}^2\Big\} \qquad \text{(A.7)}
\end{aligned}
$$

is obtained. In order to simplify the notation, $\sum_{i}^{N}$ will be used to denote the summation

of N terms over index i. This leads to the simplification of $\sum_{\substack{i=0 \\ i\neq j}}^{N}\sum_{j=0}^{N}$ to $\sum_{i}^{N}\sum_{j}^{N-1}$. The terms are broken up in such a way such that the arguments of the summations become independent of one another so that when the expectation is taken, the expectation of the product of two arguments can be replaced by the product of the expectations of the arguments. Expanding the terms of (A.7),

$$
\begin{aligned}
E\left[(V^2)^2\right] &= \frac{1}{N^2}E\Big\{(\sum_{i}^{N}x_i^2)^2 + (\frac{2}{N}\sum_{i}^{N}x_i\sum_{j}^{N}x_j)^2 + (\sum_{i}^{N}[\frac{1}{N}\sum_{j}^{N}x_j]^2)^2 \\
&\quad - \frac{4}{N}\sum_{l}^{N}x_l\sum_{i}^{N}x_i\sum_{j}^{N}x_j^2 + 2\sum_{i}^{N}[\frac{1}{N}\sum_{l}^{N}x_l]^2\sum_{j}^{N}x_j^2 \\
&\quad - \frac{4}{N}\sum_{l}^{N}x_i\sum_{i}^{N}x_i\sum_{j}^{N}[\frac{1}{N}\sum_{k}^{N}x_k]^2\Big\} \\
&= \frac{1}{N^2}E\Big\{\sum_{i}^{N}x_i^4 + \sum_{i}^{N}\sum_{j}^{N-1}x_i^2x_j^2 + \frac{4}{N^2}[\sum_{i}^{N}x_i^4 + 4\sum_{i}^{N}\sum_{j}^{N-1}x_i^3x_j \\
&\quad + 6\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}x_i^2x_jx_k + 3\sum_{i}^{N}\sum_{j}^{N-1}x_i^2x_j^2 + \sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}\sum_{l}^{N-3}x_ix_jx_kx_l] \\
&\quad + [\frac{1}{N^2}\sum_{m}^{N}\sum_{i}^{N}\sum_{j}^{N}x_ix_j]^2 - \frac{4}{N}\sum_{i}^{N}\sum_{j}^{N}\sum_{k}^{N}x_ix_jx_k^2 \\
&\quad + \frac{2}{N^2}\sum_{l}^{N}\sum_{i}^{N}\sum_{j}^{N}\sum_{k}^{N}x_ix_jx_k^2 - \frac{4}{N^3}\sum_{m}^{N}\sum_{i}^{N}\sum_{j}^{N}\sum_{k}^{N}\sum_{l}^{N}x_ix_jx_kx_l\Big\} \\
&= \frac{1}{N^2}E\Big\{\sum_{i}^{N}x_i^4 + \sum_{i}^{N}\sum_{j}^{N-1}x_i^2x_j^2 + \frac{4}{N^2}\sum_{i}^{N}x_i^4 + \frac{16}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}x_i^3x_j \\
&\quad + \frac{24}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}x_i^2x_jx_k + \frac{12}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}x_i^2x_j^2 \\
&\quad + \frac{4}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}\sum_{l}^{N-3}x_ix_jx_kx_l + \frac{1}{N^2}\sum_{i}^{N}\sum_{j}^{N}\sum_{k}^{N}\sum_{l}^{N}x_ix_jx_kx_l
\end{aligned}
$$

$$
\begin{aligned}
&\quad - \frac{2}{N}\sum_{i}^{N}\sum_{j}^{N}\sum_{k}^{N} x_i x_j x_k^2 - \frac{4}{N^2}\sum_{i}^{N}\sum_{j}^{N}\sum_{k}^{N}\sum_{l}^{N} x_i x_j x_k x_l \Big\} \\
&= \frac{1}{N^2}E\Big\{\sum_{i}^{N} x_i^4 + \sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2 + \frac{4}{N^2}\sum_{i}^{N} x_i^4 + \frac{16}{N^2}\sum_{i}^{N}\sum_{j}^{N-1} x_i^3 x_j \\
&\quad + \frac{24}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2} x_i^2 x_j x_k + \frac{12}{N^2}\sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2 \\
&\quad + \frac{4}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}\sum_{l}^{N-3} x_i x_j x_k x_l - \frac{2}{N}\Big(\sum_{i}^{N} x_i^4 + 2\sum_{i}^{N}\sum_{j}^{N-1} x_i^3 x_j \\
&\quad + \sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2 + \sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2} x_i^2 x_j x_k\Big) \\
&\quad - \frac{3}{N^2}\sum_{i}^{N}\sum_{j}^{N}\sum_{k}^{N}\sum_{l}^{N} x_i x_j x_k x_l\Big\} \\
&= \frac{1}{N^2}E\Big\{\sum_{i}^{N} x_i^4 + \sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2 + \frac{4}{N^2}\sum_{i}^{N} x_i^4 + \frac{16}{N^2}\sum_{i}^{N}\sum_{j}^{N-1} x_i^3 x_j \\
&\quad + \frac{24}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2} x_i^2 x_j x_k + \frac{12}{N^2}\sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2 \\
&\quad + \frac{4}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}\sum_{l}^{N-3} x_i x_j x_k x_l - \frac{2}{N}\sum_{i}^{N} x_i^4 - \frac{4}{N}\sum_{i}^{N}\sum_{j}^{N-1} x_i^3 x_j \\
&\quad - \frac{2}{N}\sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2 - \frac{2}{N}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2} x_i^2 x_j x_k - \frac{3}{N^2}\Big(\sum_{i}^{N} x_i^4 \\
&\quad + 4\sum_{i}^{N}\sum_{j}^{N-1} x_i^3 x_j + 6\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2} x_i^2 x_j x_k + 3\sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2 \\
&\quad + \sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}\sum_{l}^{N-3} x_i x_j x_k x_l\Big)\Big\} \\
&= \frac{1}{N^2}E\Big\{\sum_{i}^{N} x_i^4 + \sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2 + \frac{4}{N^2}\sum_{i}^{N} x_i^4 + \frac{16}{N^2}\sum_{i}^{N}\sum_{j}^{N-1} x_i^3 x_j \\
&\quad + \frac{24}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2} x_i^2 x_j x_k + \frac{12}{N^2}\sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2
\end{aligned}
$$

$$
\begin{aligned}
&+ \frac{4}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}\sum_{l}^{N-3} x_i x_j x_k x_l - \frac{2}{N}\sum_{i}^{N} x_i^4 - \frac{4}{N}\sum_{i}^{N}\sum_{j}^{N-1} x_i^3 x_j \\
&- \frac{2}{N}\sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2 - \frac{2}{N}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2} x_i^2 x_j x_k - \frac{3}{N^2}\sum_{i}^{N} x_i^4 \\
&- \frac{12}{N^2}\sum_{i}^{N}\sum_{j}^{N-1} x_i^3 x_j - \frac{18}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2} x_i^2 x_j x_k - \frac{9}{N^2}\sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2 \\
&- \frac{3}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}\sum_{l}^{N-3} x_i x_j x_k x_l \Big\}. \qquad \text{(A.8)}
\end{aligned}
$$

Gathering terms and then distributing the expectation, produces

$$
\begin{aligned}
E\left[\left(V^2\right)^2\right] &= \frac{1}{N^2}E\Big\{\sum_{i}^{N} x_i^4\left(1+\frac{4}{N^2}-\frac{3}{N^2}-\frac{2}{N}\right) + \sum_{i}^{N}\sum_{j}^{N-1} x_i^3 x_j\left(\frac{16}{N^2}-\frac{12}{N^2}-\frac{4}{N}\right) \\
&\quad + \sum_{i}^{N}\sum_{j}^{N-1} x_i^2 x_j^2\left(1+\frac{12}{N^2}-\frac{9}{N^2}-\frac{2}{N}\right) \\
&\quad + \sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2} x_i^2 x_j x_k\left(\frac{24}{N^2}-\frac{18}{N^2}-\frac{2}{N}\right) \\
&\quad + \sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}\sum_{l}^{N-3} x_i x_j x_k x_l\left(\frac{4}{N^2}-\frac{3}{N^2}\right)\Big\} \\
&= \frac{1}{N^2}\Bigg[\left(\frac{N^2-2N+1}{N^2}\right)\sum_{i}^{N}\overline{x_i^4} + \frac{4(1-N)}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\overline{x_i^3}\,\overline{x_j} \\
&\quad + \left(\frac{N^2-2N+3}{N^2}\right)\sum_{i}^{N}\sum_{j}^{N-1}\overline{x_i^2}\,\overline{x_j^2} + \frac{2(3-N)}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}\overline{x_i^2}\,\overline{x_j}\,\overline{x_k} \\
&\quad + \frac{1}{N^2}\sum_{i}^{N}\sum_{j}^{N-1}\sum_{k}^{N-2}\sum_{l}^{N-3}\overline{x_i}\,\overline{x_j}\,\overline{x_k}\,\overline{x_l}\Bigg].
\end{aligned}
\qquad \text{(A.9)}
$$

Substituting the values of the moments for Gaussian random variables with mean μ and

variance σ^2 [24], and then simplifying, the following is obtained.

$$\begin{aligned}
E\left[\left(V^2\right)^2\right] = \frac{1}{N^4}\Big[&(N-1)(N-1)N\left(3\sigma^4+6\sigma^2\mu^2+\mu^4\right)\\
&-4(N-1)N(N-1)\left(3\sigma^2\mu+\mu^3\right)\mu\\
&+(N^2-2N+3)N(N-1)\left(\sigma^2+\mu^2\right)^2\\
&-2(N-3)N(N-1)(N-2)\left(\sigma^2+\mu^2\right)\mu^2\\
&+N(N-1)(N-2)(N-3)\mu^4\Big]\\
= \frac{1}{N^3}\Big[&\sigma^4\Big(3(N-1)(N-1)+(N-1)(N^2-2N+3)\Big)\\
&+\sigma^2\mu^2\Big(6(N-1)(N-1)+2(N-1)(N^2-2N+3)\\
&-12(N-1)(N-1)-2(N-3)(N-2)(N-1)\Big)\\
&+\mu^4\Big((N-1)(N-1)+(N-1)(N^2-2N+3)\\
&-4(N-1)(N-1)-(N-1)(N-2)(N-3)\Big)\Big]\\
= \frac{1}{N^3}&\left(\sigma^4(N^3-N)\right)\\
= \frac{1}{N^2}&(N+1)(N-1)\sigma^4. \qquad \text{(A.10)}
\end{aligned}$$

Now, substituting the expression of (A.10) into (A.6), the expression for the variance of the variance estimator V^2 is found.

$$\begin{aligned}
\sigma^2_{V^2} = E\left[V^{2\,2}\right] - \left[\frac{(N-1)}{N}\sigma^2\right]^2&\\
= \frac{1}{N^2}(N+1)(N-1)\sigma^4 - \frac{(N-1)^2}{N^2}\sigma^4&\\
= \frac{\sigma^4}{N^2}\left(N^2-1-N^2+2N-1\right)&\\
= \frac{2(N-1)}{N^2}\sigma^4. \qquad \text{(A.11)}&
\end{aligned}$$

Again, this is the expression of the variance of V^2 found in (3.4).

A.3 Independence of the Estimators

Since the samples are independent and Gaussian distributed, it can be shown that the estimators are uncorrelated. By considering the two random variables, M and V^2, as being Gaussian for a large number of samples, their lack of correlation implies independence. The two estimators will be shown to be uncorrelated by calculating the correlation coefficient of the two estimators. The correlation coefficient of the two estimators is given by

$$\rho_{M,V^2} = \frac{Cov(M,V^2)}{\sigma_M \, \sigma_{V^2}}.$$

$Cov(M,V^2)$ is the covariance between the two estimators and is defined as

$$\begin{aligned} Cov(M,V^2) &= E\big[(M-\mu_M)(V^2-\mu_{V^2})\big] \\ &= E\big[\,M\,V^2\,\big] - \mu_M \mu_{V^2} \\ &= E\big[\,M\,V^2\,\big] - \frac{N-1}{N}\mu\sigma^2. \end{aligned} \tag{A.12}$$

If the covariance is shown to be equal to 0, then the two estimators will be uncorrelated. It is this quantity which is formulated below and shown to equal 0. Starting with $E\big[\,M\,V^2\,\big]$ and expanding,

$$\begin{aligned} E\big[\,M\,V^2\,\big] &= E\Big\{\Big(\frac{1}{N}\sum_i^N x_i\Big)\Big(\frac{1}{N}\sum_j^N \big(x_j - \frac{1}{N}\sum_k^N x_k\big)^2\Big)\Big\} \\ &= E\Big\{\Big(\frac{1}{N}\sum_i^N x_i\Big)\Big(\frac{1}{N}\sum_j^N \big(x_j^2 - \frac{2}{N}\sum_k^N x_k x_j + \frac{1}{N^2}\sum_k^N\sum_l^N x_k x_l\big)\Big)\Big\} \end{aligned}$$

$$
\begin{aligned}
&= E\Big\{ \frac{1}{N^2} \sum_i^N \sum_j^N x_i x_j^2 - \frac{2}{N^3} \sum_i^N \sum_j^N \sum_k^N x_i x_j x_k \\
&\qquad + \frac{1}{N^4} \sum_i^N \sum_j^N \sum_k^N \sum_l^N x_i x_k x_l \Big\} \\
&= E\Big\{ \frac{1}{N^2} (\sum_i^N x_i^3 + \sum_i^N \sum_j^{N-1} x_j^2 x_i) - \frac{1}{N^3} (\sum_i^N x_i^3 + 3 \sum_i^N \sum_j^{N-1} x_i^2 x_j \\
&\qquad + \sum_i^N \sum_j^{N-1} \sum_k^{N-2} x_i x_j x_k) \Big\} \\
&= E\Big\{ (\frac{1}{N^2} - \frac{1}{N^3}) \sum_i^N x_i^3 + (\frac{1}{N^2} - \frac{3}{N^3}) \sum_i^N \sum_j^{N-1} x_j^2 x_i \\
&\qquad - \frac{1}{N^3} \sum_i^N \sum_j^{N-1} \sum_k^{N-2} x_i x_j x_k \Big\} \\
&= \frac{1}{N^2} \Big((1 - \frac{1}{N}) \sum_i^N \overline{x_i^3} + (1 - \frac{3}{N}) \sum_i^N \sum_j^{N-1} \overline{x_j^2}\, \overline{x_i} \\
&\qquad - \frac{1}{N} \sum_i^N \sum_j^{N-1} \sum_k^{N-2} \overline{x_i}\, \overline{x_j}\, \overline{x_k} \Big) \\
&= \frac{1}{N^2} \Big((N-1) \big(3\sigma^2 \mu + \mu^3\big) + (N-3)(N-1)\mu(\sigma^2 + \mu^2) \\
&\qquad - (N-1)(N-2)\mu^3 \Big) \\
&= \frac{1}{N^2} \Big(\sigma^2 \mu \Big(3(N-1) + (N-3)(N-1) \Big) \\
&\qquad + \mu^3 \Big((N-3)(N-1) - (N-1)(N-2) + (N-1) \Big) \Big) \\
&= \frac{1}{N^2}\, N\,(N-1)\sigma^2 \mu \\
&= \frac{N-1}{N}\, \mu \sigma^2 . \qquad \text{(A.13)}
\end{aligned}
$$

Substituting the expression of (A.13) into the equation for the covariance of the two estimators (A.12),

$$\begin{aligned} Cov(M, V^2) &= E\left[M\, V^2 \right] - \frac{N-1}{N}\mu\sigma^2 \\ &= \frac{N-1}{N}\mu\sigma^2 - \frac{N-1}{N}\mu\sigma^2 \\ &= 0. \end{aligned}$$

The covariance of the two estimators is equal to 0. Therefore, the correlation coefficient ρ_{M,V^2} is equal to 0. The two estimators are uncorrelated. Since M is Gaussian and for a large number of samples (and therefore, a large number of degrees of freedom), V^2 is Gaussian, the estimators are independent.

Appendix B

Obtaining Probability Density Function of $Z = \frac{D}{S}$

The ratio Z was defined in chapter 3 as being comprised of a Gaussian random variable D, with mean $\frac{\mu\sqrt{N}}{\sigma}$ and variance 1, as the numerator and the denominator is a Chi distributed random variable S with N degrees of freedom. The two random variables have the probability density functions shown in equations (3.6) and (3.7) and are reproduced below for convenience.

$$f_D(x) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{(x - \frac{\mu\sqrt{N}}{\sigma})^2}{2}\right\}$$

$$f_{S_N}(y) = \frac{y^{N-1} \exp\left\{-\frac{y^2}{2}\right\}}{2^{\frac{N}{2}-1}\Gamma\left(\frac{N}{2}\right)}$$

Before carrying out the actual formulation for the probability density function of the ratio, the procedure [7,24] to be followed is described below.

Define $z = \frac{x}{y}$ and define a dummy variable $w = x$, for convenience. Referring again

to (3.9), the transformation which will be carried out is

$$\begin{aligned} f_{ZW}(z,w) &= f_{DS}(w, \frac{w}{z}) \frac{1}{|J(x,y)|} \\ &= f_D(w)\, f_S(\frac{w}{z}) \frac{1}{|J(x,y)|}. \end{aligned} \tag{B.1}$$

$|J(x,y)|$ is the absolute value of the Jacobian, given by

$$J(x,y) = \begin{vmatrix} \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} \\ \frac{\partial z}{\partial x} & \frac{\partial z}{\partial y} \end{vmatrix}.$$

By integrating with respect to the dummy variable w from $-\infty$ to ∞, the probability density function of Z remains.

First, the Jacobian is determined for the variable substitutions $z = \frac{x}{y}$ (i.e. $y = \frac{w}{z}$) and $w = x$.

$$\begin{aligned} J(x,y) &= \begin{vmatrix} \frac{\partial x}{\partial x} & \frac{\partial x}{\partial y} \\ \frac{\partial \left(\frac{x}{y}\right)}{\partial x} & \frac{\partial \left(\frac{x}{y}\right)}{\partial y} \end{vmatrix} \\ &= \begin{vmatrix} 1 & 0 \\ \frac{1}{y} & -\frac{x}{y^2} \end{vmatrix} \\ &= -\frac{x}{y^2} = -\frac{z^2}{w} \end{aligned} \tag{B.2}$$

The formation of the joint distribution f_{ZW} is outlined in (B.1). Substituting for the mean of D, $\frac{\mu\sqrt{N}}{\sigma}$, by γ for convenience and gathering terms, the following is obtained.

$$f_{ZW}(z,w) = f_D(w)\, f_S(\frac{w}{z}) \frac{1}{|J(x,y)|}$$

$$
\begin{aligned}
&= \frac{1}{\sqrt{2\pi}} \exp\Big\{-\frac{(w-\gamma)^2}{2}\Big\} \cdot \frac{\left(\frac{w}{z}\right)^{N-1} \exp\Big\{-\frac{\left(\frac{w}{z}\right)^2}{2}\Big\}}{2^{\frac{N}{2}-1}\Gamma\left(\frac{N}{2}\right)} \cdot \frac{w}{z^2} \\
&= \frac{w^N}{z^{N+1}} \frac{1}{2^{\frac{N-2}{2}}\sqrt{2\pi}\Gamma\left(\frac{N}{2}\right)} \exp\left\{-\frac{\left(w^2 - 2\gamma w + \gamma^2 + \frac{w^2}{z^2}\right)}{2}\right\} \\
&= \frac{w^N}{\sqrt{2\pi}\, 2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)} \exp\left\{-\frac{\left(w^2\left(1+\frac{1}{z^2}\right) - 2\gamma w + \gamma^2\right)}{2}\right\} \\
&= \frac{w^N}{\sqrt{2\pi}\, 2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)} \exp\left\{-\frac{\left(1+\frac{1}{z^2}\right)}{2}\Big[w^2 - \frac{2\gamma w}{\left(1+\frac{1}{z^2}\right)}\right. \\
&\qquad\qquad \left. + \frac{\gamma^2}{\left(1+\frac{1}{z^2}\right)} + \frac{\gamma^2}{\left(1+\frac{1}{z^2}\right)^2} - \frac{\gamma^2}{\left(1+\frac{1}{z^2}\right)^2}\Big]\right\} \\
&= \frac{w^N}{\sqrt{2\pi}\, 2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)} \exp\left\{-\frac{\left(1+\frac{1}{z^2}\right)}{2}\Big[w - \frac{\gamma}{\left(1+\frac{1}{z^2}\right)}\Big]^2\right. \\
&\qquad\qquad \left. + \frac{\gamma^2}{2\left(1+\frac{1}{z^2}\right)} - \frac{\gamma^2}{2}\right\} \\
&= \frac{w^N}{\sqrt{2\pi}\, 2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)} \exp\left\{-\frac{\left(\frac{z^2+1}{z^2}\right)}{2}\Big[w - \frac{\gamma z^2}{z^2+1}\Big]^2\right\} \cdot \\
&\qquad\qquad \exp\left\{-\frac{\gamma^2}{2(z^2+1)}\right\} \\
&= \frac{w^N \exp\left\{-\frac{\gamma^2}{2(z^2+1)}\right\}}{\sqrt{2\pi}\, 2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)} \exp\left\{-\frac{\left(\frac{z^2+1}{z^2}\right)}{2}\Big[w - \frac{\gamma z^2}{z^2+1}\Big]^2\right\}. \qquad \text{(B.3)}
\end{aligned}
$$

Integrating with respect to the dummy variable w to obtain the probability density function of Z,

$$
f_Z(z) = \frac{\exp\left\{-\frac{\gamma^2}{2(z^2+1)}\right\}}{\sqrt{2\pi}\, 2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)} \int_{-\infty}^{\infty} w^N \exp\left\{-\frac{\left(\frac{z^2+1}{z^2}\right)}{2}\Big[w - \frac{\gamma z^2}{z^2+1}\Big]^2\right\} dw. \qquad \text{(B.4)}
$$

In order to simply the mathematics and the presentation, let $\sigma_Z^2 = \frac{z^2}{z^2+1}$ and $\eta_Z =$

$\frac{\gamma z^2}{z^2+1} = \gamma\sigma_Z^2$. Making these substitutions in (B.4) and continuing,

$$f_Z(z) = \frac{\exp\{-\frac{\gamma^2}{2(z^2+1)}\}}{\sqrt{2\pi}\,2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)} \int_{-\infty}^{\infty} w^N \exp\Big\{-\frac{1}{2\sigma_Z^2}\big[w-\eta_Z\big]^2\Big\} dw$$

$$= \frac{\exp\{-\frac{\gamma^2}{2(z^2+1)}\}}{2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)} \sigma_Z \int_{-\infty}^{\infty} \frac{w^N}{\sqrt{2\pi\sigma_Z^2}} \exp\{-\frac{(w-\eta_Z)^2}{2\sigma_Z^2}\} dw. \qquad \text{(B.5)}$$

In (B.5), consider the variable substitution $t = \frac{w-\eta_Z}{\sigma_Z}$. Therefore, $w = \sigma_Z\, t + \eta_Z$ and $dw = \sigma_Z dt$ producing,

$$f_Z(z) = \frac{\exp\{-\frac{\gamma^2}{2(z^2+1)}\}\,\sigma_Z}{2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)} \int_{-\infty}^{\infty} \frac{\big(\sigma_Z\, t + \eta_Z\big)^N}{\sqrt{2\pi}} \exp\{-\frac{t^2}{2}\} dt$$

$$= \frac{\exp\{-\frac{\gamma^2}{2(z^2+1)}\}\,\sigma_Z}{2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)} \sigma_Z^N \int_{-\infty}^{\infty} \frac{\big(t + \frac{\eta_Z}{\sigma_Z}\big)^N}{\sqrt{2\pi}} \exp\{-\frac{t^2}{2}\} dt$$

$$= \frac{\exp\{-\frac{\gamma^2}{2(z^2+1)}\}\,\sigma_Z^{N+1}}{2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)} \sum_{k=0}^{N} \binom{N}{k} \big(\frac{\eta_Z}{\sigma_Z}\big)^{N-k} \int_{-\infty}^{\infty} \frac{t^k}{\sqrt{2\pi}} \exp\{-\frac{t^2}{2}\} dt \qquad \text{(B.6)}$$

The binomial expansion was used in (B.6) to obtain separate t^k terms. The integral on the right is simply the k^{th} moment of a Gaussian random variable with mean 0 and variance 1. As seen in chapter 4, the k^{th} moment, $E[t^k]$, of a such a random variable $N(0,1)$ is expressible [1,7,24] as

$$E[t^k] = \begin{cases} \frac{(2v)!}{2^v v!}, & v = \frac{k}{2} \text{ when } k \text{ is even} \\ 0, & \text{when } k \text{ is odd} \end{cases}$$

Substituting this expression into the expression of (B.6), then substituting the expressions

of σ_Z and η_Z, and simplifying, the probability density function of Z is obtained in (B.7).

$$\begin{aligned} f_Z(z) &= \frac{\exp\{-\frac{\gamma^2}{2(z^2+1)}\}\,\sigma_Z^{N+1}}{2^{\frac{N-2}{2}} z^{N+1}\Gamma\left(\frac{N}{2}\right)}\Big[\sum_{k=0}^{\lfloor\frac{N}{2}\rfloor}\binom{N}{2k}\left(\frac{\eta_Z}{\sigma_Z}\right)^{N-2k}\frac{(2k)!}{k!2^k}\Big] \\ &= \frac{\exp\{-\frac{\gamma^2}{2(z^2+1)}\}}{2^{\frac{N-2}{2}}\Gamma\left(\frac{N}{2}\right)}\frac{\sigma_Z^{N+1}}{z^{N+1}}\Big[\sum_{k=0}^{\lfloor\frac{N}{2}\rfloor}\binom{N}{2k}(\gamma\sigma_Z)^{N-2k}\frac{(2k)!}{k!2^k}\Big] \\ &= \frac{\exp\{-\frac{\gamma^2}{2(z^2+1)}\}}{2^{\frac{N-2}{2}}\Gamma\left(\frac{N}{2}\right)}\frac{1}{\sqrt{z^2+1}^{N+1}}\Big[\sum_{k=0}^{\lfloor\frac{N}{2}\rfloor}\binom{N}{2k}\left(\frac{\gamma z}{\sqrt{z^2+1}}\right)^{N-2k}\frac{(2k)!}{k!2^k}\Big] \end{aligned} \tag{B.7}$$

where $\gamma = \frac{\mu\sqrt{N}}{\sigma}$.

Bibliography

[1] J. G. Proakis, *Digital Communications*, McGraw-Hill, New York, 3rd edition, 1995.

[2] S. Benedetto and G. Montorsi, "Unveiling Turbo Codes: Some Results on Parallel Concatenated Coding Schemes," *IEEE Trans. Inform. Theory*, vol. IT-42, no. 2, pp. 409–428, Mar. 1996.

[3] S. Haykin, *Digital Communications*, John Wiley & Sons, New York, 1988.

[4] L.R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284–287, Mar. 1974.

[5] J. Hagenauer and P. Hocher, "A Viterbi algorithm with soft-decision outputs and its applications," in *Proc. GLOBECOM '89*, Dallas, TX, Nov. 1989, pp. 47.1.1–47.1.7.

[6] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-codes(1)," in *Proc., IEEE Int. Conf. on Communications*, Geneva, Switzerland, May 1993, pp. 1064–1070.

[7] I. F. Blake, *An Introduction to Applied Probability*, Robert E. Krieger Publishing, Florida, 2nd edition, 1979.

[8] D. Zwillinger, Ed., *Standard Mathematical Tables and Formulae*, CRC Press, Boca Raton, 30th edition, 1996.

[9] W. Hoeffding, "A Class of Statistics with Asymptotically Normal Distribution," *Annals of Math. Stat.*, vol. 19, no. 3, pp. 293–325, 1948.

[10] D. Nelson, Ed., *Dictionary of Mathemathetics*, Penguin Books, Toronto, 2nd edition, 1998.

[11] P. de Jong, *Central limit theorems for generalized multilinear forms*, Centrum voor Wiskunde en Informatica, Netherlands, 1989.

[12] P. de Jong, "A Central Limit Theorem for Generalized Quadratic Forms," *Probability Theory and Related Fields*, vol. 75, pp. 261–277, 1987.

[13] R. von Mises, "On the Asymptotic Distribution of Differentiable Statistical Functions," *Annals of Math. Stat.*, vol. 18, no. 3, pp. 309–348, 1947.

[14] L. Zhengyan and L. Chuanrong, *Limit Theory and Mixing Dependent Random Variables*, Science Press and Kluwer Academic Publisers, New York, 1996.

[15] J. K. Patel and C. B. Read, *Handbook of the Normal Distribution*, Marcel Dekker Inc., New York, 2nd edition, 1996.

[16] Harald Cramér, *Random Variables and Probability Distributions*, Number 36 in Cambridge Tracts in Mathematics and Mathematical Physics. Cambridge University Press, 1963.

[17] L. Le Cam, "The Central Limit Theorem around 1935," *Statistical Science*, vol. 1, no. 1, pp. 78–96, 1986.

[18] P. Lévy, *Théorie de l'Addition des Variables Aléatoires*, Gauthier-Villars, Paris, 1937.

[19] J. M. Mendel, *Lessons in Estimation Theory for Signal Processing, Communications, and Control*, Prentice Hall, New Jersey, 1995.

[20] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*, vol. 1, Prentice Hall, New Jersey, 1993.

[21] Harald Cramér, *Mathematical Methods of Statistics*, Princeton University Press, Princeton, 1999.

[22] N. L. Johnson, S. Kotz, and N. Balakrishnan, *Continuous Univariate Distributions*, vol. 1, John Wiley & Sons, New York, 2nd edition, 1994.

[23] N. L. Johnson, S. Kotz, and N. Balakrishnan, *Continuous Univariate Distributions*, vol. 2, John Wiley & Sons, New York, 2nd edition, 1995.

[24] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill, Boston, 3rd edition, 1991.

[25] H. B. Dwight, *Tables of Integrals and Other Mathematical Data*, MacMillan, New York, 4th edition, 1961.

[26] I.S. Gradshteyn, I.M. Ryzhik, and A. Jeffrey, *Tables of Integrals, Series and Products*, Academic Press, San Diego, 5th edition, 1994.

[27] A. Jeffrey, *Handbook of Mathematical Formulas and Integrals*, Academic Press, San Diego, 1995.

[28] N. L. Johnson, S. Kotz, and A. W. Kemp, *Univariate Discrete Distributions*, John Wiley & Sons, New York, 2nd edition, 1992.

[29] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, New York, 1991.

[30] C.-C. Chao, R. J. McEliece, L. Swanson, and E. R. Rodemich, "Performance of Binary Block Codes at Low Signal-to-Noise Ratios," *IEEE Trans. Inform. Theory*, vol. 38, no. 6, pp. 1677–1687, Nov. 1992.

[31] J. Bakus and A. K. Khandani, "Quantizer Design for Turbo-code Channels," Tech. Rep., University of Waterloo, Waterloo, Ontario, Canada, July 1999, E&CE#99-04.

[32] S. J. Mason, "Feedback Theory – Further Properties of Signal Flow Graphs," in *Proceedings of IRE*, July 1956, vol. 44, pp. 920–926.

[33] S. J. Mason and H. J. Zimmerman, *Electronic Circuits, Signals, and Systems*, John Wiley & Sons, New York, 1960.

[34] J. K. Wolf and A. J. Viterbi, "On the Weight Distribution of Linear Block Codes formed from Convolutional Codes," *IEEE Trans. Commun.*, vol. IT-44, no. 9, pp. 1049–1051, Sept. 1996.

[35] Y. Desaki, T. Fujiwara, and T. Kasami, "A method for computing the weight distribution of a block code by using its trellis diagram," *IEICE Trans. Fundamentals*, vol. E77-A, pp. 1230–1237, Aug. 1994.

[36] Y. Desaki, T. Fujiwara, and T. Kasami, "The Weight Distributions of Extended Binary Primitive BCH Codes of Length 128," *IEEE Trans. Inform. Theory*, vol. IT-43, no. 4, pp. 1364–1371, July 1997.

[37] O. T. Takeshita, M. P. C. Fossorier, and D. J. Costello, "A New Technique for Computing the Weight Spectrum of Turbo-Codes," *IEEE Communications Letters*, vol. 3, no. 8, pp. 251–253, Aug. 1999.

[38] M. P. C. Fossorier, S. Lin, and D. J. Costello, "On the Weight Distribution of Terminated Convolutional Codes," *IEEE Trans. Inform. Theory*, vol. IT-45, no. 5, pp. 1646–1648, July 1999.

[39] R. E. Blahut, *Principles and Practice of Information Theory*, Addison-Wesley, New York, 1987.

[40] B. Porat, *A Course in Digital Signal Processing*, John Wiley & Sons, New York, 1997.

[41] A. V. Oppenheim and R. W. Shafer, *Discrete-Time Signal Processing*, Prentice Hall, New Jersey, 1989.

[42] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, 1993.

[43] T. Fujiwara, Y. Desaki, T. Sugita, and T. Kasami, "The Weight Distributions of Several Extended Binary Primitive BCH Codes of Length 256," in *Proc. Int. Symp. Inform. Theory*, Ulm, Germany, June 1997, p. 363.

[44] T. Sugita, T. Kasami, and T. Fujiwara, "The Weight Distributions of the Third-Order Reed-Muller Code of Length 512," *IEEE Trans. Inform. Theory*, vol. IT-42, no. 5, pp. 1622–1625, Sept. 1996.

[45] C. Fontaine, "A method to find cosets of the first-order Reed-Muller code with a high minimum weight," in *Proc. Int. Symp. Inform. Theory*, Cambridge, MA, USA, Aug. 1998, p. 464.

[46] M. J. Moisio and K. O. Väänänen, "Two Recursive Algorithms for Computing the Weight Distribution of Certain Irreducible Cyclic Codes," *IEEE Trans. Inform. Theory*, vol. IT-45, no. 4, pp. 1244–1249, May 1999.

[47] T. Klove, "The Weight Distribution of Cosets," *IEEE Trans. Inform. Theory*, vol. IT-40, no. 3, pp. 911–913, May 1994.

[48] J. Wolfmann, "Weight Distributions of Some Binary Primitive Cyclic Codes," *IEEE Trans. on Inform. Theory*, vol. IT-40, no. 6, pp. 2068–2071, Nov. 1994.

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

[50] J. Hagenauer, "Source-Controlled Channel Decoding," *IEEE Trans. Comm*, vol. 43, no. 9, pp. 2449–2457, Sept. 1995.

[51] A. Leon-Garcia, *Probability and Random Processes for Electrical Engineering*, Addison Wesley, Reading, Massachusetts, 2nd edition, 1993.

[52] I. Olkin, L. J Gleser, and C. Derman, *Probability Models and Applications*, MacMillan College Publishing, New York, 2nd edition, 1994.

[53] H. O. Lancaster, *The Chi-squared Distribution*, John Wiley & Sons, New York, 1969.

[54] J.W. Harris and H. Stocker, *Handbook of Mathematics and Computational Science*, Springer-Verlag, New York, 1998.

[55] B. Ostle and L. C. Malone, *Statistics in Research*, Iowa State University Press, Ames, Iowa, 4th edition, 1954.

[56] P. J. Bickel and K. A. Doksum, *Mathematical Statistics: Basic Ideas and Selected Topics*, Holden-Day, San Francisco, 1977.

[57] J. E. Freund and R. E. Walpole, *Mathematical Statistics*, Prentice Hall, New Jersey, 4th edition, 1987.

[58] R. Deutsch, *Estimation Theory*, Prentice Hall, New Jersey, 1965.

[59] L. R. Shenton and K. O. Bowman, *Maximum Likelihood Estimation in Small Samples*, Charles Griffin and Company, London, 1977.

[60] S. M. Kay, *Fundamentals of Statistical Signal Processing: Detection Theory*, vol. 2, Prentice Hall, New Jersey, 1998.

[61] R. K. Burdick and F. A. Graybill, *Confidence Intervals on Variance Components*, Marcel Dekker, New York, 1992.

[62] J. Bracken and A. Schleifer, *Tables for Normal Sampling With Unknown Variance*, Harvard Business School, Boston, 1964.

[63] H. Bateman, *Tables of Integral Transforms*, vol. 1, McGraw-Hill, New York, 1954.