Implementation of Extrinsic Information Transfer Charts

by

Anupama Battula

Problem Report submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Electrical Engineering

Daryl S. Reynolds, Ph.D.
James D Mooney, Ph.D.
Matthew C. Valenti, Ph.D., Chair

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2008

Keywords: BICM-ID, Exit charts, mutual information, Modulation

Abstract

Implementation of Extrinsic Information Transfer Charts

by

Anupama Battula
Master of Science in Electrical Engineering

West Virginia University

Matthew C. Valenti, Ph.D., Chair

Many coded modulation schemes have been proposed where high order modulation and channel coding are combined for bandwidth-efficient transmission. In our project we will focus on BICM (Bit Interleaved Coded Modulation). The performance of BICM can be greatly improved through iterative information exchange between the demapper and the decoder at the receiver. This system is usually referred to as BICM with iterative decoding (BICM-ID). The convergence properties of iterative receivers in general, and BICM-ID in particular, may be investigated through the use of extrinsic information transfer (EXIT) charts. The EXIT chart is a simple, yet powerful tool for visualizing the exchange of mutual information in iterative systems. In this report, EXIT charts are used to predict the threshold signal-to-noise ratio (SNR) for which the bit error rate (BER) of a BICM-ID system will drop sharply given a particular convolutional code and modulation (including mapping).

Acknowledgements

I would first like to thank my committee chair and advisor, Dr. Matthew C. Valenti, for giving me the opportunity to do my problem report on a topic like EXIT charts. I would like to convey my sincere thanks to Dr. Matthew Valenti for helping me through in every tough situation. He is a very good advisor and a perfect professor to work with.

I would also like to thank Dr. Daryl S. Reynolds and Dr. James D. Mooney for being on my committee. I have been fortunate to have had the opportunity to take courses with all of my committee members.

Finally, I would like to thank my friends, my dad B. SambiReddy and my mom B. Vani who were of great support to me at all times.

Contents

List of Figures

Chapter 1

Introduction

Many coded modulation schemes have been proposed where high order modulation and channel coding are combined for bandwidth-efficient transmission. In our project we will focus on BICM (Bit Interleaved Coded Modulation) [1][2], which is the concatenation of an encoder, an interleaver and a mapper. Even though BICM is a convenient way to design, implement and provide high diversity in fading channels, due to the data-processing inequality its capacity is lower than that of coded-modulation (CM). However, by iteratively exchanging bit-wise extrinsic information between the demodulator and the decoder, we can mitigate the performance loss due to BICM and approach the CM capacity. Such a process is called Bit Interleaved Coded Modulation with Iterative Decoding (BICM-ID), a term coined by Li and Ritcey in [3]. A powerful tool for visualizing the process of iterative decoding and predicting the convergence threshold was developed by ten Brink in [4] and is known as Extrinsic Information Transfer (EXIT) Charts.

1.1 System Block Diagram

The block diagram of a system with BICM-ID is shown in Fig. 1.1. Serial concatenation of an encoder, a random bit interleaver π and a modulator forms the transmitter. The information bits u are first encoded by a convolutional code which produces a coded sequence b' which is then interleaved by a random bit interleaver π and finally that interleaved sequence b is mapped by the modulator into a symbol sequence x for transmission. According to some

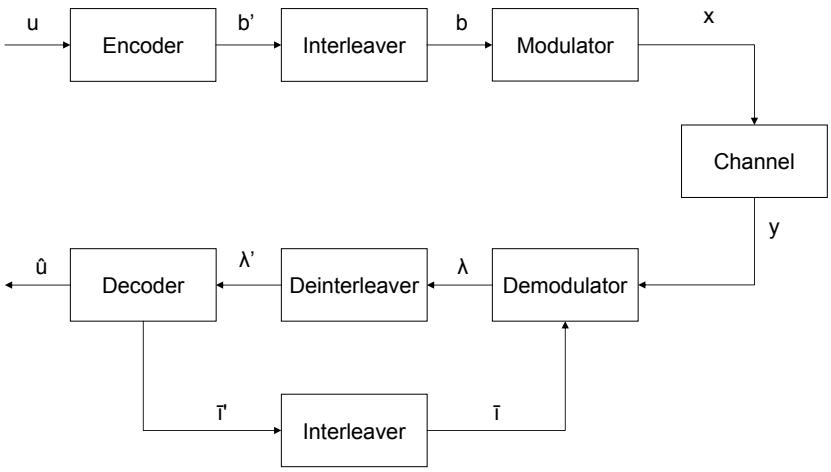

Figure 1.1: Block diagram of bit-interleaved coded modulation with iterative decoding.

mapping rule $\mu(.)$, each symbol x is chosen from a two-dimensional M-ary constellation.

The received signal is described by $y = x + n$, where n denotes the complex zero-mean Gaussian noise. At the receiver the demapper processes the received symbols y and their corresponding log-likelihood ratios λ are obtained which are deinterleaved and applied to the decoder. The estimates of the coded bits from the decoder are fed back as a priori information to the demodulator.

1.2 Problem Statement

The performance of coded-modulation is characterized by plotting the bit error rate (BER) as a function of the signal-to-noise ratio (SNR). Fig. 1.2 shows a typical BER curve for a system that uses a particular convolutional code [7,5], 16-QAM modulation (with set-partition mapping), and an interleaver of length 8004 bits. The SNR is represented by $\mathcal{E}_s/N_0$, which is the ratio of energy-per-symbol to one-sided noise spectral density. The curve was

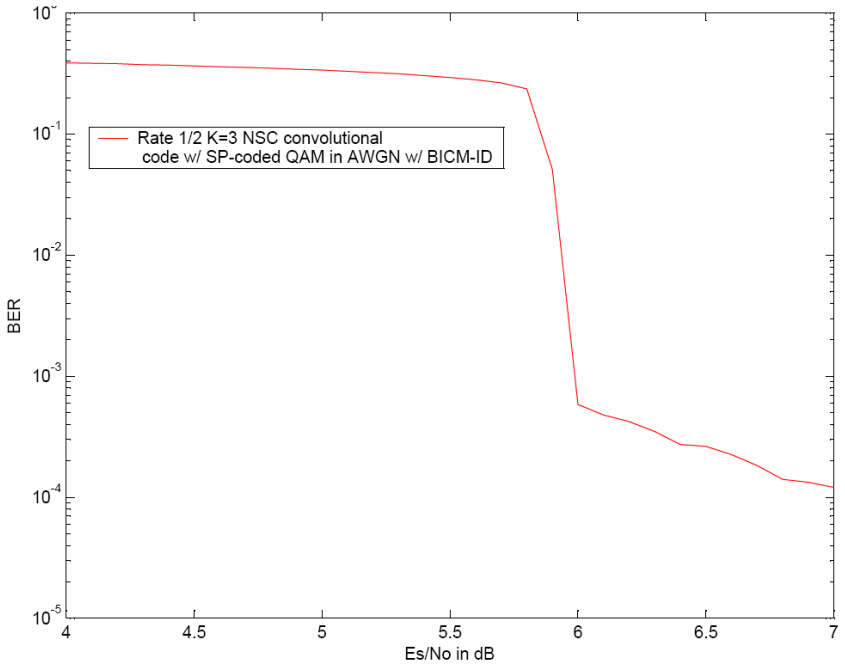

Figure 1.2: BER vs. $\mathcal{E}_s/N_0$ with iterative decoding for 16QAM AWGN channel, R = 1/2 K = 3 NSC convolutional code

produced through simulation using open-source software developed at WVU [1]. Notice that the curve drops steeply at $\mathcal{E}_s/N_0 \approx 5.8$ dB, and then flattens out into a BER "floor". The location of the error floor can be determined analytically by using the so-called error-free feedback (EFF) bound [3], but this floor is not within the scope of this report. Furthermore, the threshold SNR value for which the BER drops sharply can be determined analytically by using EXIT charts [4], which is the focus of this report.

The problem addressed in this report can therefore be summarized as follows. Given a BICM-ID system with a particular convolutional code and a particular modulation (including mapping), the goal of this report is to accurately predict the threshold SNR value at which the BER drops sharply. This threshold will be determined by using the recently developed EXIT charts.

[1]The Coded Modulation Library.

1.3 Structure of Report

The report is organized in five chapters. This first chapter has motivated the report and provided the Problem Statement. Chapter 2 discusses in detail about modulation, demodulation and mutual information. Chapter 3 discusses in detail about encoder and decoder. Chapter 4 deals with the demodulator and decoder characteristics of EXIT charts and the estimated threshold. The last chapter provides results obtained from the simulation of EXIT charts and also concludes with a summary of the report.

Chapter 2

Modulation

2.1 Modulation

In order to reliably transmit and receive information, most modern communication systems utilize digital modulation techniques. The process of mapping digital information onto analog waveforms that match the characteristics of the channel is called modulation. We can distinguish modulation schemes by their bandwidth efficiency, energy efficiency and complexity. The ability of a modulation scheme to accommodate data within a limited bandwidth is described by bandwidth efficiency. Similarly the ability of the system to reliably send information at the lowest practical power level is described by energy efficiency.

Let us now define the process of digital modulation. To transmit, the modulator selects from M available signal waveforms. In general, M is a power of 2. For *binary* modulation it can be as small as 2. Each individual bit is mapped to one of two possible *signal waveforms*, $s_0(t)$ or $s_1(t)$ in binary modulation. A bit value of 0 and 1 are mapped to signals $s_0(t)$ and $s_1(t)$ respectively. Generally, signal waveforms are also known as *modulated symbols*. A group of b bits is mapped to one of the M available signal waveforms, where $M = 2^b$ in M-ary modulation.

We can represent each modulated signal as a weighted linear combination of orthonormal functions, known as the *basis functions*. The cardinality, K is also called the dimensionality of the modulation for a set of basis functions. Let

$$K = |\Phi| \tag{2.1}$$

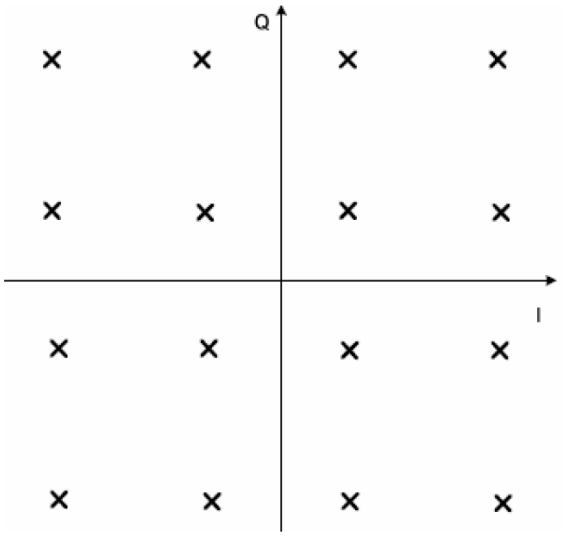

Figure 2.1: Constellation diagram for 16-QAM.

where

$$\Phi = \{\phi_0(t), \phi_1(t), ...\phi_{K-1}(t)\} \tag{2.2}$$

K is modulation-specific, and can vary from $K = 1$ to $K = M$. $K = 1$ in the case of Binary Phase-Shift Keying (BPSK) or Pulse-Amplitude-Modulation (PAM), while $K = M$ in the case of orthogonal Frequency-Shift Keying (FSK). For dimensionality $K = 2$ many common M-ary modulations are defined. It is convenient to display the set of modulated signals in the form of a *constellation diagram*, for modulations with dimensionality $K \leq 2$. An example of one such diagram is shown in Fig. 2.1 for Quadrature Amplitude Modulation (QAM) with $M = 16$ expressed in terms of in-phase (I) and quadrature (Q) components. I and Q present a rectangular representation of the polar diagram. I axis lies on the zero degree phase reference, and the Q axis is rotated by 90 degrees. Each of the axes corresponds to the projection of the signal onto one of the basis functions.

Let $s_{m,k}$ be the coefficients(weights) of the basis functions, $\phi_k(t)$ for k from 0 to $K-1$ It is required to create the modulated signal, $s_m(t), 0 \leq m \leq M-1$. Each signal point, $s_m(t)$

is then

$$s_m(t) = \sum_{k=0}^{K-1} s_{m,k}\phi_k(t). \tag{2.3}$$

If $\phi_1(t)$ represent the real axis, and $\phi_2(t)$ represent the imaginary axis,

$$s_m = s_{m,0} + js_{m,1} \tag{2.4}$$

Let

$$\mathcal{S} = \{s_0, ..., s_{M-1}\} \tag{2.5}$$

be the set of all M possible modulated symbols. The constellation diagram shows the locations of all M symbols. The modulator selects $b = \log_2(M)$ bits and then maps them to the appropriate symbol from the set, $\mathcal{S}$ for each transmitted symbol. This symbol is then transmitted.

Mapping, or labeling, is how bits are mapped to signals. By determining how many bit errors occur each time there is a symbol error, labeling affects the bit error rate (BER). In our project, we focus on two different mappings namely *set-partitioning mapping* and *MSEW mapping*. MSEW stands for Maximum squared Euclidian weight [5]. Fig. 2.2 shows the constellation diagram for 16 QAM with SP mapping and Fig. 2.3 shows the constellation diagram for 16 QAM with SP mapping

2.2 Demodulation

Demodulator is used to process the received signal and determine the symbol/bit likelihoods. Then, a hard-decision may be made by selecting the symbol from the set, $\mathcal{S}$, which has the largest likelihood. The appropriate bit (binary) or bits (M-ary) can be selected from the mapping used, once the most likely symbol is selected. A receiver consists of a *front end* and a *back end*. The front end is known as a *demodulator*, and the back end is known as a *detector*.

The front end, or demodulator, transforms the received signal waveform, $r(t)$, into a K-dimensional vector $\mathbf{r}$. Here K is the dimensionality of the modulation, defined in (2.1).

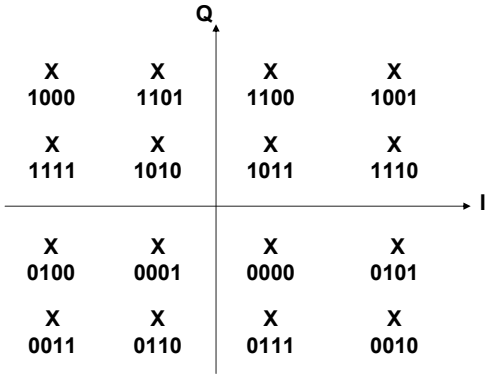

Figure 2.2: SP mapping for 16-QAM.

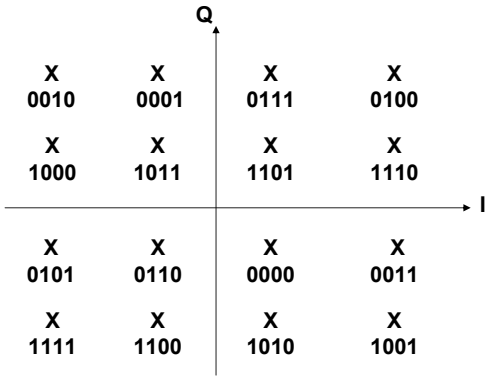

Figure 2.3: MSEW mapping for 16-QAM.

Each value, r_i, in the vector, $\mathbf{r}$, is the correlation of $r(t)$ and the i-th basis function. There are two types of demodulators namely, the *correlation demodulator* and the *matched-filter demodulator*.

The received signal, $r(t)$, is passed through a bank of K correlators in a correlation demodulator. The received signal is first multiplied by the i-th basis function in the i-th correlator, then integrated over the period of the basis function, and finally sampled once at the end of the period. Then each correlator finds the time-domain inner product of the received signal and the i-th basis function:

$$\begin{aligned} r_i &= \int_0^{T_s} r(t)\phi_i(t)dt \\ &= \langle r(t), \phi_i(t) \rangle \end{aligned} \tag{2.6}$$

The received signal, $r(t)$, is passed through K matched-filters in a matched-filter demodulator, with impulse responses given by

$$h_i(t) = \phi_i\left(T_s - t\right), \qquad 0 \leq t \leq T_s, i \in [0 : K-1] \tag{2.7}$$

where T_s is the duration of the i-th basis function, $\phi_i(t)$. The result is sampled at the end of the epoch. The output of a filter is obtained by the convolution of its input with its impulse response.Therefore the output of the i-th matched-filter is

$$\begin{aligned} r_i &= r(t) * h_i(t)|_{t=T_s} \\ &= \int_{\infty}^{-\infty} r(\lambda)h_i(T_s - \lambda)d\lambda \\ &= \langle r(t), \phi_i(t) \rangle \end{aligned} \tag{2.8}$$

We see that the output of the matched-filter, shown in (2.8) is equivalent to the output of the correlation demodulator, shown in (2.6).

The back end, or detector, picks the most likely transmitted symbol, $\hat{\mathbf{s}}$. There are two types of detectors namely *MAP detector* and the *ML detector*. The two detectors are similar. The only exception is that the MAP detector can handle input distributions that are not uniform.

A *maximum a posteriori* (MAP) detector determines which signal was most likely sent. This can be done by picking the estimated symbol, $\hat{\mathbf{s}}$, that is most probable given the received

symbol, $\mathbf{r}$. The MAP rule for selecting the estimated symbol is

$$\hat{\mathbf{s}} = \arg\max_{\mathbf{s}_i \in \mathcal{S}} p(\mathbf{s}_i|\mathbf{r}). \tag{2.9}$$

The final expression for the MAP rule is then

$$\hat{\mathbf{s}} = \arg\max_{\mathbf{s}_i \in \mathcal{S}} p(\mathbf{r}|\mathbf{s}_i)\, p(\mathbf{s}_i). \tag{2.10}$$

A *maximum likelihood* (ML) detector uses the same rule in (2.10). It is simplified by assuming that the input distribution is uniform. Since the probability of each symbol is equally likely it can be removed from the expression. The ML rule is thus

$$\hat{\mathbf{s}} = \arg\max_{\mathbf{s}_i \in \mathcal{S}} p(\mathbf{r}|\mathbf{s}_i). \tag{2.11}$$

When the noise is Gaussian, then p(r—s) is a Gaussian probability density function, and the *s*ymbol likelihood ratio is

$$\hat{\mathbf{s}} = \arg\max_{\mathbf{s}_i \in \mathcal{S}} -\frac{1}{N_o} \|\mathbf{r} - \mathbf{s}_i\|^2. \tag{2.12}$$

The final expression can be reduced to

$$\begin{aligned} \hat{\mathbf{s}} &= \arg\max_{\mathbf{s}_i \in \mathcal{S}} -\|\mathbf{r} - \mathbf{s}_i\| \\ &= \arg\min_{\mathbf{s}_i \in \mathcal{S}} \|\mathbf{r} - \mathbf{s}_i\| \end{aligned} \tag{2.13}$$

Let λ_n be the bit LLR of the n^{th} code bit of a given group of $b = \log_2(M)$ bits.

$$\lambda_n = \log \frac{p(c_n = 1|\mathbf{r})}{p(c_n = 0|\mathbf{r})} \tag{2.14}$$

where $p(c_n = 1|\mathbf{r})$ is the probability that the bit is a 1 given the received vector $\mathbf{r}$. From (2.14) we can see that the resulting LLR will be positive, if the bit is more likely to be 1.

The LLR expression in (2.14) can thus be represented as

$$\begin{aligned} \lambda_n &= \log \frac{\sum_{\mathbf{s}_i \in \mathcal{S}_n^{(1)}} p(\mathbf{s}_i|\mathbf{r})}{\sum_{\mathbf{s}_i \in \mathcal{S}_n^{(0)}} p(\mathbf{s}_i|\mathbf{r})} \\ &= \log \frac{\sum_{\mathbf{s}_i \in \mathcal{S}_n^{(1)}} p(\mathbf{r}|\mathbf{s}_i)\, p[s_i]}{\sum_{\mathbf{s}_i \in \mathcal{S}_n^{(0)}} p(\mathbf{r}|\mathbf{s}_i)\, p[s_i]} \end{aligned} \tag{2.15}$$

where $\mathcal{S}_n^{(1)}$ represents the set of symbols whose n^{th} bit position is a 1 and $p[s_i]$ is the a priori probability of symbol s_i.

2.3 Mutual Information

The channel capacity is defined as the maximum rate at which data could be reliably transmitted over a channel by Shannon [6] [7]. Using *mutual information*, the expression for the capacity can be found in terms of the signal-to-noise ratio (SNR). The channel capacity was found to be the mutual information between the channel input and output maximized over all input distributions.

The *mutual information* between the events $\{X = x\}$ and $\{Y = y\}$ is defined as [6][7]

$$I(X;Y) = \int\int p_{X,Y}(x,y) \log \frac{p_{X,Y}(x,y)}{p_X(x)p_Y(y)} dxdy, \tag{2.16}$$

where $p_{X,Y}(x,y)$ is the joint pdf and $p_X(x)$ and $p_Y(y)$ are the marginal pdfs of the random variables X and Y, respectively.

It is noted that (2.16) can be expressed in terms of the expectation

$$\begin{aligned} I(X;Y) &= E\left[i(x;y)\right] \\ &= \int\int p_{X,Y}(x,y) i(x;y) dxdy, \end{aligned} \tag{2.17}$$

where the *mutual information function* is defined as

$$i(x;y) = \log \frac{p_{X,Y}(x,y)}{p_X(x)p_Y(y)}, \tag{2.18}$$

Using the definition of conditional probability the mutual information function equation from (2.18) is obtained as:

$$i(x;y) = \log \frac{p_{Y|X}(y|x)}{p_Y(y)}, \tag{2.19}$$

where $p_{Y|X}(y|x)$ is the conditional pdf of Y given X.

From (2.19) we substitute the value of i(x;y)

$$\begin{aligned} I(X;Y) &= \int\int p_{X,Y}(x,y)\log\frac{p_{Y|X}(y|x)}{p_Y(y)}dxdy \\ &= \int\int p_{X,Y}(x,y)\left[\log p_{Y|X}(y|x) - \log p_Y(y)\right]dxdy \\ &= \int\int p_{X,Y}(x,y)\log p_{Y|X}(y|x)dxdy - \int\int p_{X,Y}(x,y)\log p_Y(y)dxdy \\ &= \int\int p_{X,Y}(x,y)\log p_{Y|X}(y|x)dxdy - \int p_Y(x,y)\log p_Y(y)dy \\ &= (-H(Y|X)) - (-H(Y)) \end{aligned} \tag{2.20}$$

where conditional entropy H(Y—X) and entropy H(Y) are

$$H(Y|X) = -\int\int p_{X,Y}(x,y)\log p_{Y|X}(y|x)dxdy \tag{2.21}$$

$$H(Y) = -\int p_Y(x,y)\log p_Y(y)dy \tag{2.22}$$

Therefore *Mutual Information* is given by

$$I(X;Y) = H(Y) - H(Y|X) \tag{2.23}$$

Fig 2.4 shows the mutual information versus $\mathcal{E}_s/N_0$ for various modulations.

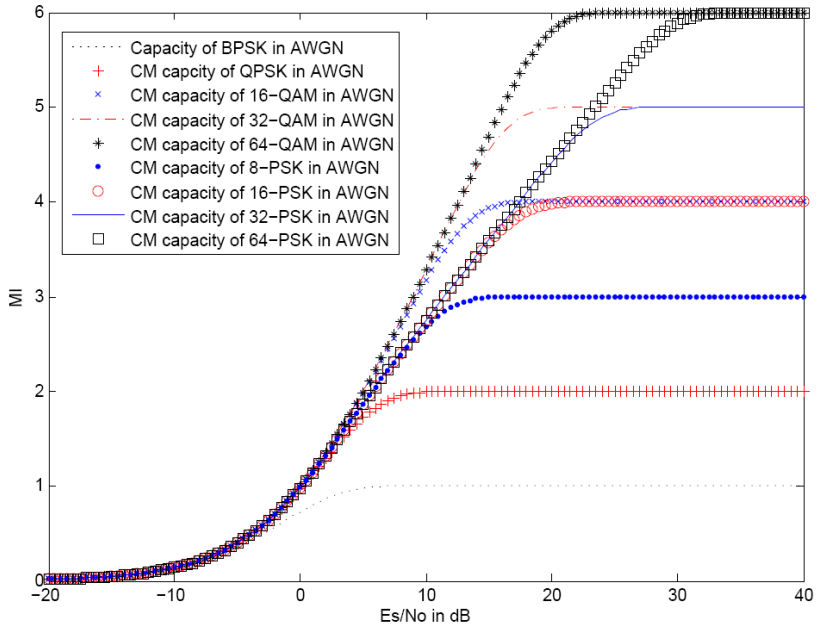

Figure 2.4: Mutual information Versus $\mathcal{E}_s/N_0$ for various modulations.

Chapter 3

Convolutional Coding

3.1 Encoder

Convolutional codes were first mentioned by Elias in 1955 [8]. Convolutional codes are linear. Using a shift register bank a binary convolutional encoder takes a stream of information bits and converts it into a stream of transmitted bits. Input is given to shift registers as the information bits and the output encoded bits are obtained by modulo-2 addition of the input information bits and the contents of the shift registers. Each input bit influences the output during its own interval as well as the next two input bit intervals. A convolutional code is defined by the number of stages in the shift register, the number of modulo-2 adders and the connection between the shift register and the modulo-2 adders. The contents of the shift register defines the state of the encoder which is completely determined by the previous two information bit inputs. The code rate r for a convolutional code is defined as

$$r = \frac{k}{n} \tag{3.1}$$

where k is the number of input information bits and n is the number of output encoded bits at one time interval. The constraint length K for a convolutional code is defined as

$$K = m + 1 \tag{3.2}$$

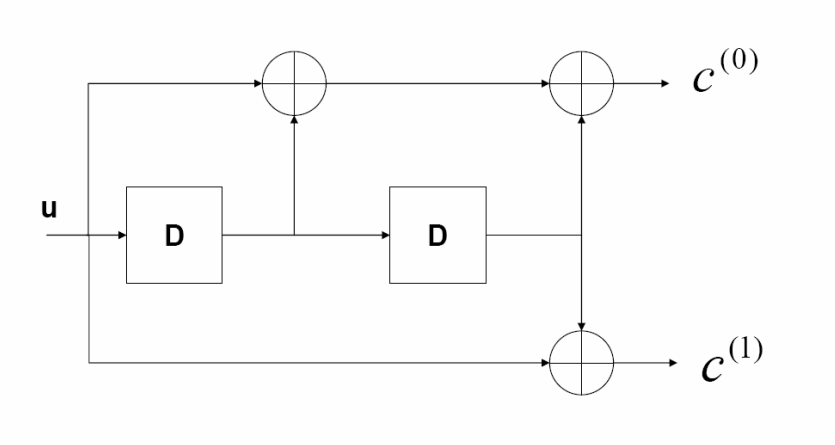

Figure 3.1: Block diagram of a [7,5] NSC convolutional code.

where m is the maximum number of stages (memory size) in any shift register. The shift registers store the state information of the convolutional encoder and the constraint length relates the number of bits upon which the output depends. The performance of a convolutional code improves as the constraint length increases. Constraint length is the measure of the memory within the encoder. Depending on whether the information sequence appears directly within the code sequence, convolutional codes can be *systematic* or *nonsystematic*. One of the code generator polynomials must be equal to 1 so that one of the n outputs is simply the current information bit for a convolutional code to be systematic. Fig. 3.1 shows the block diagram for NSC [7, 5].

A binary convolutional code is a finite state machine with 2^m states. A state diagram shows the relation between the encoder states by specifying the inputs required to move from one state to another and the corresponding outputs that are produced for any given set of state transitions. Fig. 3.2 shows the state diagram for a NSC code [7, 5]. Each node denotes a possible encoder state in the diagram. For each node there are two branches entering and

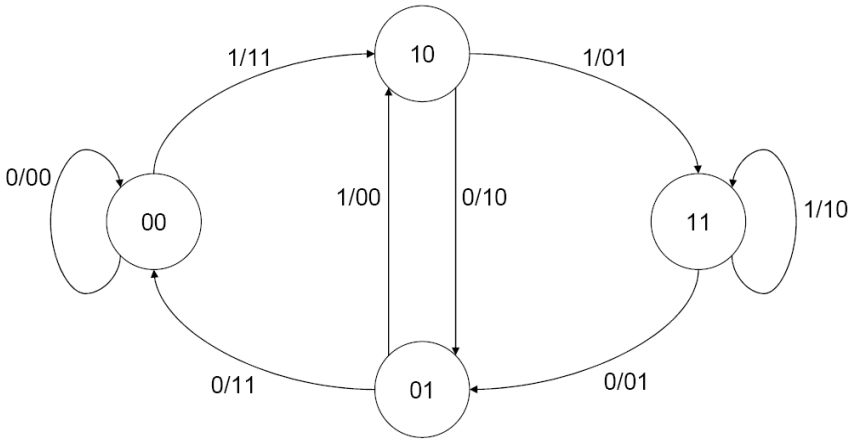

Figure 3.2: State diagram of a [7,5] NSC convolutional code.

leaving it. The branches are labeled (u/p_1p_2), where u is an input bit and p_1, p_2 are the encoded parity bits.

A trellis is an expansion of the state diagram. It shows how the state transitions evolve over time for different possible input sequences and each distinct sequence of input bits corresponds to a unique path through the trellis. Fig. 3.3 shows the trellis diagram for the NSC [7,5] code.

3.2 Decoder

Convolutional codes are conveniently decoded on a trellis. Two popular approaches to decoding based on trellis are maximum likelihood (ML) decoding using the Viterbi algorithm and log-MAP(BCJR) algorithm. The Viterbi algorithm is used to minimize the codeword error rate and log-MAP algorithm is used to minimize the bit error rate. Fig. 3.4 shows the the input/output relationship for a soft-input soft-output (SISO) decoder.

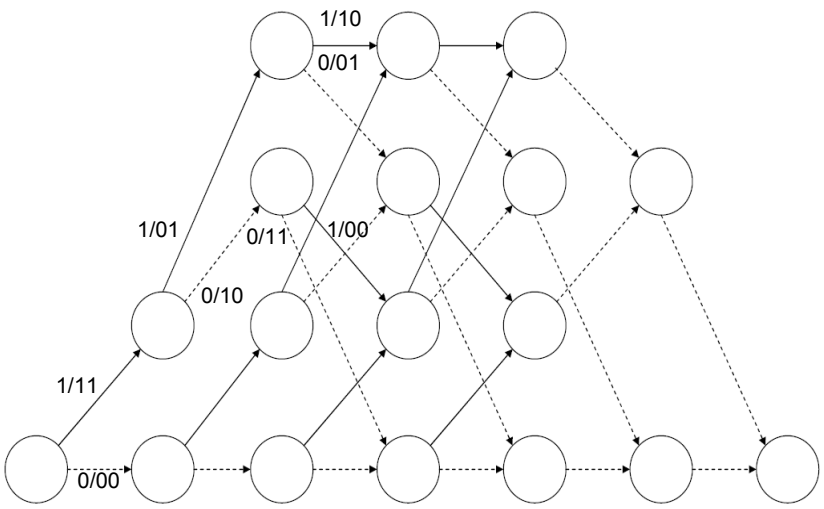

Figure 3.3: Trellis diagram for a [7,5] NSC convolutional code.

Figure 3.4: SISO decoder.

3.2.1 Viterbi/Hard Decoding

A computationally-efficient method for performing maximum likelihood decoding of convolutional codes is to use the Viterbi algorithm. If the encoder can be represented as a finite state machine, then the code can be decoded using the Viterbi algorithm. The goal of the Viterbi algorithm is to minimize the probability of making a codeword error.

The trellis state at time instant k is denoted by $\{S_k = s'\}$ for $s' = 0, 1, ..., 2^m - 1$ and $0 \leq k \leq T$. The state transition corresponding to an input u_k is denoted by $\{S_k = s'\} \longrightarrow \{S_{k+1} = s\}$. The branch metric $\gamma_{k+1}(s', s)$ is defined as the weight associated with a branch whose starting and terminal states are $\{S_k = s'\}$ and $\{S_{k+1} = s\}$ respectively. The path metric $\alpha_{s',k}$ is stored in trellis state $\{S_k = s'\}$.

Assuming that the encoder begins and ends in the all-zeros state, the Viterbi algorithm is as follows

1. Each state is labeled with a path metric $\alpha_{s',k}$ for $s' = 0, 1, ..., 2^m - 1$ and $0 \leq k \leq T$. At $k = 0$, initialize

$$\alpha_{s',0} = \begin{cases} 0 & k = 0 \\ \infty & k \neq 0 \end{cases}$$

2. Begin with time index $k = 1$.

3. Begin with state index $s' = 0$.

4. Compute the branch metric $\gamma_{k+1}(s', s)$ for every branch in the trellis

$$\gamma_{k+1}(s', s) \quad = \quad \sum_{i=0}^{n-1} \parallel r_{kn+i} - b_{kn+i}(s' \longrightarrow s) \parallel^2 \tag{3.3}$$

   where $b_{kn+i}(s' \longrightarrow s)$ for i = 0,1,...n-1 are the modulated code bits associated with the state transition $\{S_k = s'\} \longrightarrow \{S_{k+1} = s\}$

5. Label each state with a partial path metric. The partial path metric associated with a particular trellis state is equal to the sum of all the branch metrics along the path leading into that state.

6. For each state s there are two branches entering into it. At each state perform the add-compare-select(ACS) operation and pick the branch which has the minimum value. The surviving branch is the branch corresponding to the state transition $\{S_k = s'\} \longrightarrow \{S_{k+1} = s\}$ and the path metric is now

$$\alpha_{s,k} = \min\{(\alpha_{s',k-1} + \gamma_{k-1 \longrightarrow k}), (\alpha_{s'+1,k-1} + \gamma_{k-1 \longrightarrow k})\} \tag{3.4}$$

   The branch which has the larger metric is deleted. The algorithm needs to keep a track of the surviving state sequence in addition to storing the surviving path's metric.

7. Increment the states s'. If $s' < 2^m$, return to step 5. Otherwise proceed to step 8.

8. Increment time $k = k + 1$. If $k = T$ the end of the trellis is reached so proceed to step 9. Otherwise return to step 3.

9. The ML path leads into the state $S_{Nu} = 0$. The Viterbi algorithm traces the ML path by sweeping right to left across the trellis using the stored surviving state sequence. The input bits associated with each branch (state transition) of the ML path is the ML estimate of data bits.

3.2.2 Soft-output Decoding

The goal of the log-MAP algorithm is to minimize the bit error rate and provide soft-outputs, which can be fed back to the demodulator as a priori information. It is a forward-backward algorithm. Fig. 3.4 shows the SISO decoder. Log likelihood ratios are sent as soft inputs to the SISO decoder. SISO decoder produces a posteriori information as the soft output. We get the extrinsic information by subtracting the log likelihood ratios from the a posteriori information. We can execute this algorithm in time-domain as multiplication becomes addition. In log-domain, addition is not straight-forward. Let us consider the Jacobian logarithm, to illustrate how addition is performed in the log-domain. Given X, Y and $x = \ln X$, $y = \ln Y$ and $X = e^x$, $Y = e^y$. Multiplication can be written as

$$XY = Z \tag{3.5}$$

$$\begin{aligned} z &= \ln(Z) \\ &= \ln(XY) \\ &= \ln e^{x+y} \end{aligned} \tag{3.6}$$

$$z = x + y \tag{3.7}$$

Addition can be written as

$$\mathrm{X} + \mathrm{Y} = \mathrm{Z} \tag{3.8}$$

$$\begin{aligned} z &= \ln Z \\ &= \ln e^{x} + e^{y} \\ &= \max(x, y) + \ln(1 + e^{-|y-x|}) \end{aligned} \tag{3.9}$$

where $\ln(e^x + e^y)$ is the *Jacobian logarithm*. The final equation is

$$\max *(x, y) = max(x, y) + \ln(1 + e^{-|y-x|}) \tag{3.10}$$

Log likelihoods are sent as soft inputs to the SISO decoder. The log likelihood ratio is given by

$$\Lambda_i = \ln \frac{p(r_i|c_i = 1)}{p(r_i|c_i = 0)} \tag{3.11}$$

The log-MAP algorithm proceeds as follows.

1. Forward Recursion

   (a) Each state is labeled with a path metric $\alpha_{s',k}$ for $s' = 0, 1, ..., 2^m - 1$ and $0 \leq k \leq T$. In forward sweep at $k = 0$, initialize

$$\alpha_{s',0} = \begin{cases} 0 & k = 0 \\ \infty & k \neq 0 \end{cases}$$

(b) Begin with time index $k = 1$.

(c) Begin with state index $s' = 0$.

(d) Compute the branch metric $\gamma_{k+1}(s', s)$ for every branch in the trellis

$$\gamma_{k+1}(s', s) = \frac{2}{\sigma^2} \sum_{i=0}^{n-1} \| r_{kn+i} - b_{kn+i}(s' \longrightarrow s) \|^2 \tag{3.12}$$

where $b_{kn+i}(s' \longrightarrow s)$ for i = 0,1,...n-1 are the code bits associated with the state transition $\{S_k = s'\} \longrightarrow \{S_{k+1} = s\}$ and σ^2 is the noise variance.

(e) For each state s there are two branches entering into it. This branch corresponding to the state transition $\{S_k = s'\} \longrightarrow \{S_{k+1} = s\}$ is now

$$\alpha_{s,k} = \max * \{(\alpha_{s',k-1} + \gamma_{k-1 \longrightarrow k}), (\alpha_{s'+1,k-1} + \gamma_{k-1 \longrightarrow k})\} \tag{3.13}$$

Note that the operation given above serves the role of the ACS operation in the Viterbi algorithm.

(f) Increment the states s'. If $s' < 2^m$, return to step 1(e). Otherwise proceed to step 1(g).

(g) Increment time $k = k + 1$. If $k = T$ the end of the trellis is reached so proceed to step 2. Otherwise return to step 1(c).

2. Backward Recursion

(a) Each state is labeled with a path metric $\alpha_{s',k}$ for $s' = 0, 1, ..., (2^m) - 1$ and $0 \leq k \leq L$. If the time index $k = T$ the trellis is terminated so initialize as follows

$$\beta_{s',L} = \begin{cases} 0 & k = 0 \\ \infty & k \neq 0 \end{cases}$$

(b) Begin with time index $k = T - 1$.

(c) Begin with state index $s' = 0$.

(d) For each state s there are two branches entering into it. At each state perform the add-compare-select(ACS) operation and pick the branch which has the minimum value. The surviving branch is the branch corresponding to the state transition $\{S_k = s'\} \longrightarrow \{S_{k+1} = s\}$ and the path metric is now

$$\beta_{s,k} = \max * \{(\beta_{s',k+1} + \gamma_{k \longrightarrow k+1}), (\beta_{s'+1,k+1} + \gamma_{k \longrightarrow k+1})\} \tag{3.14}$$

The branch which has the larger metric is deleted. The algorithm needs to keep a track of the surviving state sequence in addition to storing the surviving path's metric.

(e) Increment the states s'. If $s' < 2^m$, return to step 2(d). Otherwise proceed to step 2(f).

(f) Decrement time $k = k - 1$. If $k = 0$ the end of the trellis is reached so proceed to step 2. Otherwise return to step 2(c).

3. Determining a posteriori probabilities in LLR form. Consider a branch connecting $\alpha_{s,k}$ and $\beta s', k+1$ the likelihood of the branch is

$$\Lambda_k = \alpha_{s,k} + \gamma_{k \longrightarrow k+1} + \beta_{s',k+1} \tag{3.15}$$

The a posteriori output in LLR form is

$$\begin{aligned} L(c_k) &= \max_{k:c=1} * \Lambda_k - \max_{k:c=0} * \Lambda_k \\ &= \log p(c = 1|\underline{r}) - \log p(c = 0|\underline{r}) \\ &= \log \frac{p(c_k = 1|\underline{r})}{p(c_k = 0|\underline{r})} \end{aligned} \tag{3.16}$$

The extrinsic information is obtained as follows.

$$L_e(c_k) = L(c_k) - \Lambda_k \tag{3.17}$$

Chapter 4

EXIT Charts

Even though BICM is a convenient way to design, implement and provide high diversity in fading channels, due to the data-processing inequality its capacity is lower than that of coded-modulation (CM). However, by iteratively exchanging bit-wise extrinsic information between the demodulator and the decoder, we can mitigate the performance loss due to BICM and approach the CM capacity. Such a process is called Bit Interleaved Coded Modulation with Iterative Decoding (BICM-ID), a term coined by Li and Ritcey in [3]. To begin the iterations, the demodulator receives the noisy signal and generates bit LLRs which are interleaved and fed as a priori information to the decoder. To generate LLRs for the code bits the decoder uses this information. By subtracting the *a priori* information produced by the decoder from the LLRs we can obtain the extrinsic information of the demodulator. After deinterleaving the extrinsic information becomes a priori information for the demodulator.

A powerful tool for visualizing the process of iterative decoding and predicting the convergence threshold was developed by ten Brink in [4] and is known as Extrinsic Information Transfer (EXIT) charts. An EXIT chart consists of two curves. One is the detector mutual information transfer characteristic and the other is decoder mutual information transfer characteristic. A plot of the mutual information of the extrinsic information at the output of the decoder/ demodulator as a function of the mutual information of the *a priori* input is the mutual information transfer characteristic. The a priori information (as an LLR) is Gaussian distributed with a variance (σ^2) equal to twice the mean is a key assumption used when generating the transfer characteristics. We can observe that there is a one-to-one cor-

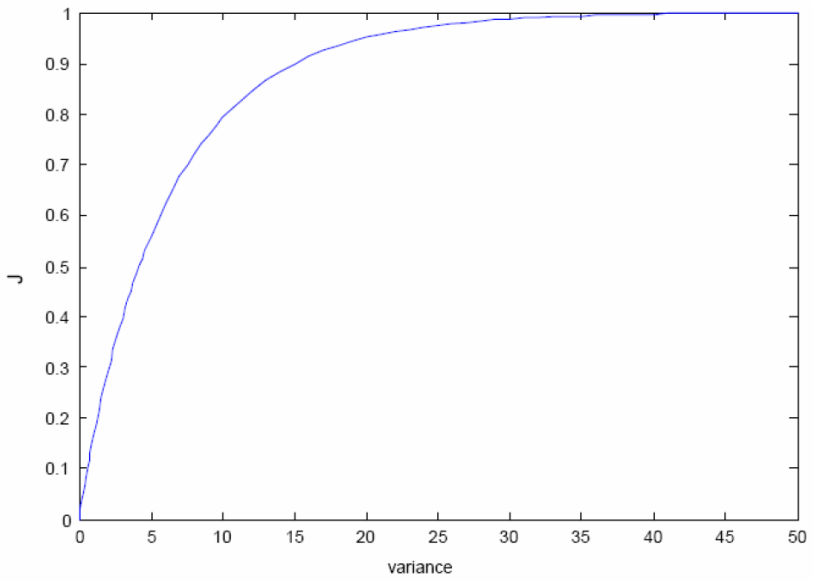

Figure 4.1: Mutual information of Gaussian distributed a priori information as a function of the variance.

respondence between the variance σ^2 and the mutual information of the a priori information given [9] by

$$J(\sigma) = 1 - \frac{1}{\sqrt{2\pi\sigma}} \int_{-\infty}^{\infty} \exp\left(-\frac{(x-\sigma^2/2)^2}{2\sigma^2}\right) \log_2(1+\exp(-x))dx \tag{4.1}$$

Fig. shows $J(\sigma)$ as a function of the variance σ^2.

4.1 Demodulator Transfer Characteristics

In order to plot the demodulator transfer characteristic, a long sequence of bits b are generated and mapped to symbols. The symbol sequence is modulated and then transmitted through the desired channel with a certain $\mathcal{E}_s/N_0$. Next, Gaussian distributed *a priori* LLRs i.e $\bar{I} \sim \mathcal{N}((2b-1)\sigma^2/2, \sigma^2)$ with mutual information I_I are generated such that the noise variance σ^2 satisfies (4.1). The extrinsic information $\bar{\lambda}$ is obtained by subtracting $\bar{I}$ from the

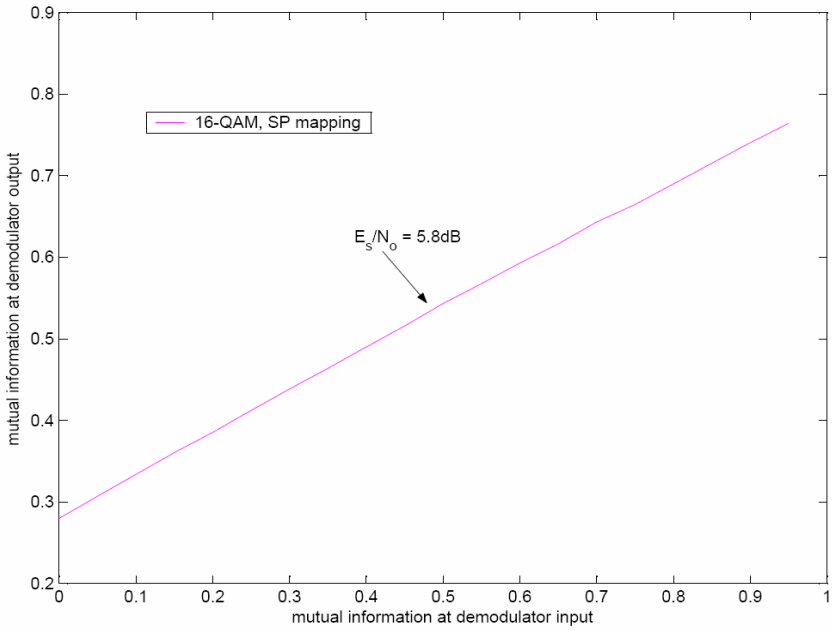

Figure 4.2: Demodulator mutual information transfer characteristics for 16-QAM.

LLRs λ,

$$\bar{\lambda} = \lambda - \bar{I} \tag{4.2}$$

The mutual information at the output of the demodulator is

$$I_\lambda = 1 - \frac{1}{\log(2)\log_2(M)} \sum_{i=1}^{\log_2 M} E[\max * (0, \bar{\lambda}_i(-1)^{b_i})] \tag{4.3}$$

The mutual information at the output of the demodulator is a function of the I_I and $\mathcal{E}_s/N_0$. I_λ is found using Monte-Carlo integration. Fig. 4.2 shows demodulator transfer characteristics for 16-QAM with SP mapping in AWGN at $\mathcal{E}_s/N_0 = 5.8dB$. When $I_I = 0$ the value of I_λ is the BICM capacity.

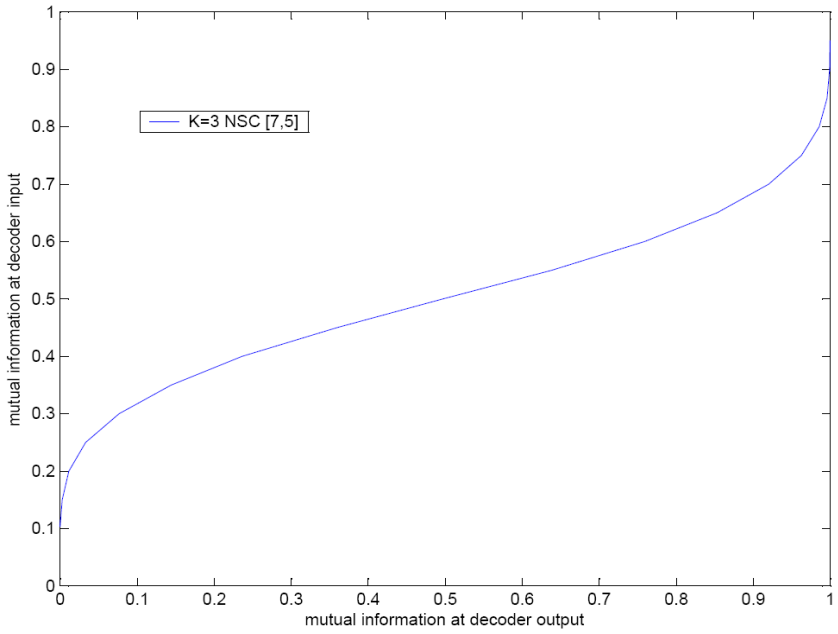

Figure 4.3: Decoder mutual information transfer characteristics for rate 1/2, non-recursive codes with constraint length K=3.

4.2 Decoder Transfer Characteristics

The mutual information at the output of the channel decoder I_I is a function of the mutual information of the APP input $\bar{\lambda}$ to the decoder i.e I_λ. A long sequence of random code bits (b) are generated, to plot the decoder transfer characteristic. The a priori input to the decoder is Gaussian distributed such that $\bar{\lambda} \sim \mathcal{N}((2b-1)\sigma^2/2, \sigma^2)$. The decoder produces extrinsic information $\bar{I}$ from which we can calculate I_I using (4.3). Fig. 4.3 shows the decoder transfer characteristics for rate = 1/2 NSC convolutional codes with generators [7,5] and constraint length K = 3. The decoder transfer characteristic does not directly depend on the signal-to-noise ratio of the channel as long as the a priori's generated by the demodulator are Gaussian. Whereas a new demodulator characteristic must be generated for each desired value of $\mathcal{E}_s/N_0$, only one decoder characteristic must be generated since it is independent of the $\mathcal{E}_s/N_0$.

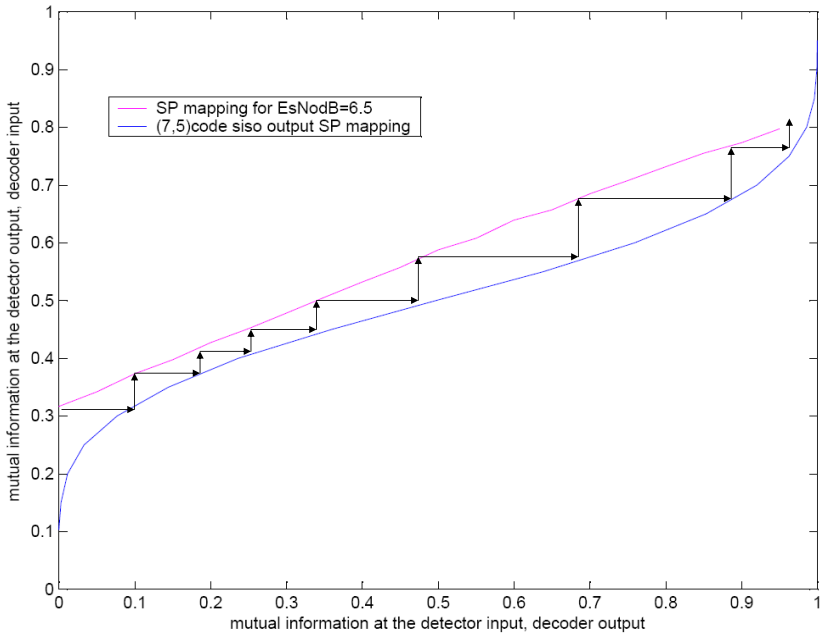

Figure 4.4: Extrinsic information transfer chart for $\mathcal{E}_s/N_0$=6.5dB .

4.3 Threshold Estimation

Since the extrinsic information from the detector/decoder becomes the APP input to the decoder/demodulator after deinterleaving/interleaving an EXIT chart is obtained by plotting the detector and decoder transfer characteristics on the same plot. The minimum $\mathcal{E}_s/N_0$ value required to raise the demodulator curve high enough to open a tunnel between the decoder and the demodulator transfer trajectories is the convergence threshold.

Fig. 4.4 shows EXIT curves for 16-QAM, SP mapping and rate 1/2, K=3 [7, 5] NSC code in an AWGN channel for $\mathcal{E}_s/N_0 = 6.5$dB. We can observe from Fig. 4.4 that a tunnel exists between the two transfer curves for $\mathcal{E}_s/N_0 = 6.5$dB. The two transfer curves come closer as the SNR is lowered. The SNR threshold is the smallest SNR for which the two curves do not touch, i.e, for which a tunnel exits. The EXIT chart for SNR threshold i.e, $\mathcal{E}_s/N_0 = 5.8$dB is shown in Fig 4.5.

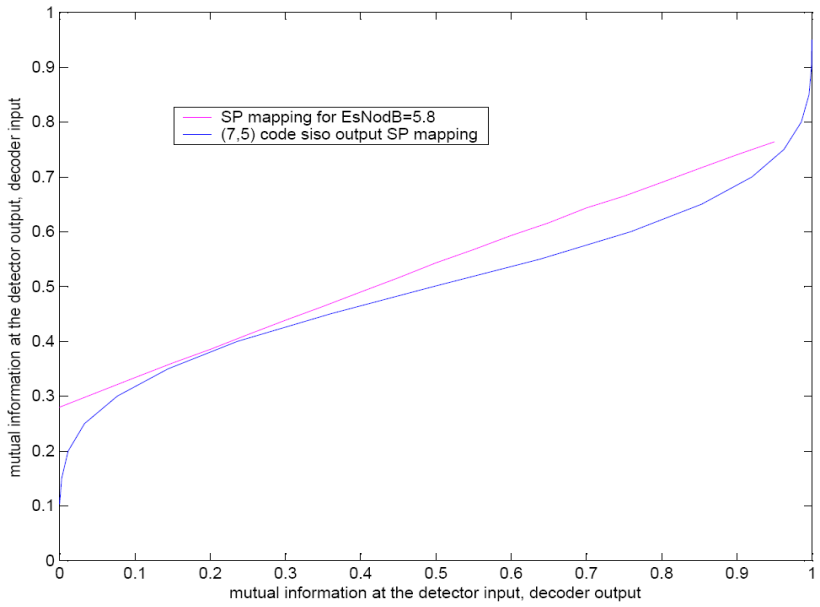

Figure 4.5: Extrinsic information transfer chart for $\mathcal{E}_s/N_0$=5.8dB.

Chapter 5

Results and Conclusion

5.1 Results

In the last three chapters, results were given for 16-QAM with SP mapping. In this section, results are given for a different mapping, namely the MSEW mapping. As before, a NSC code with generators [7,5] is used along with log-MAP decoding. The message size at the input of the encoder is 100,000 bits and the iterative BICM-ID receiver was set to run 40 iterations.

Fig. 5.1 shows the graph plotted between mutual information at the demodulator input verses the mutual information at the demodulator output for 16-QAM with MSEW mapping in AWGN channel at $\mathcal{E}_s/N_0 = 6.6dB$.

Fig. 5.2 shows the graph plotted between mutual information at the decoder output verses the mutual information at the decoder input for rate 1/2, NSC [7,5] convolutional code with constraint length K=3.

Since the extrinsic information from the detector/decoder becomes the APP input to the decoder/demodulator after deinterleaving/interleaving an EXIT chart is obtained by plotting the detector and decoder transfer characteristics on the same plot. Fig. 5.3 shows the EXIT curves for 16-QAM and rate 1/2, K = 3 [7, 5] NSC code in an AWGN channel for $\mathcal{E}_s/N_0 = 7$dB. As the SNR is lowered, the two transfer curves come closer and the smallest SNR for which the two curves do not touch, that is, for which a tunnel exits, is the SNR threshold. The EXIT chart for SNR threshold i.e, $\mathcal{E}_s/N_0 = 6.6$dB is shown in Fig. 5.4.

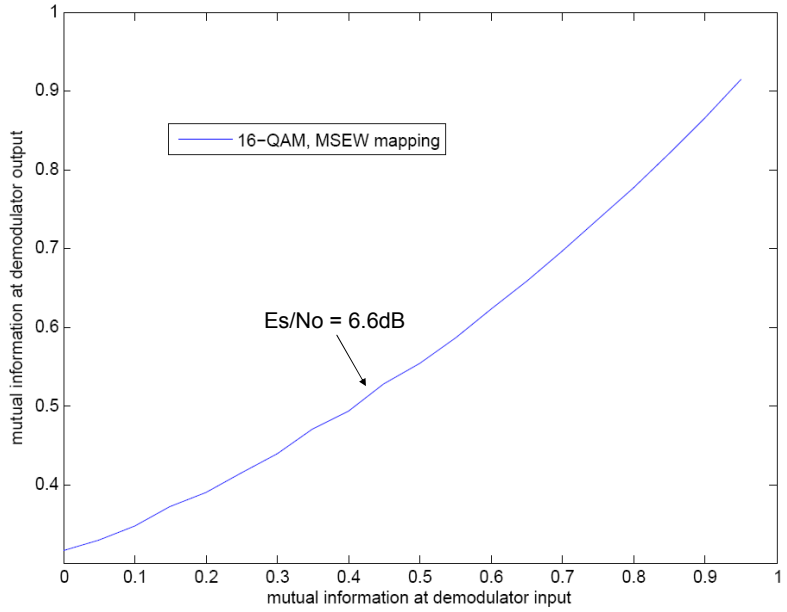

Figure 5.1: Demodulator mutual information transfer characteristics for 16-QAM, MSEW.

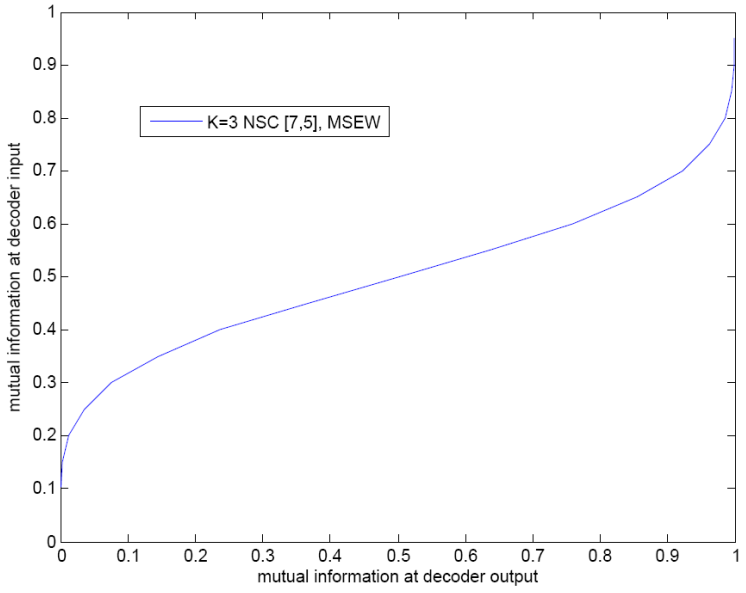

Figure 5.2: Decoder mutual information transfer characteristics for rate 1/2, non-recursive codes with constraint length K=3 for MSEW mapping.

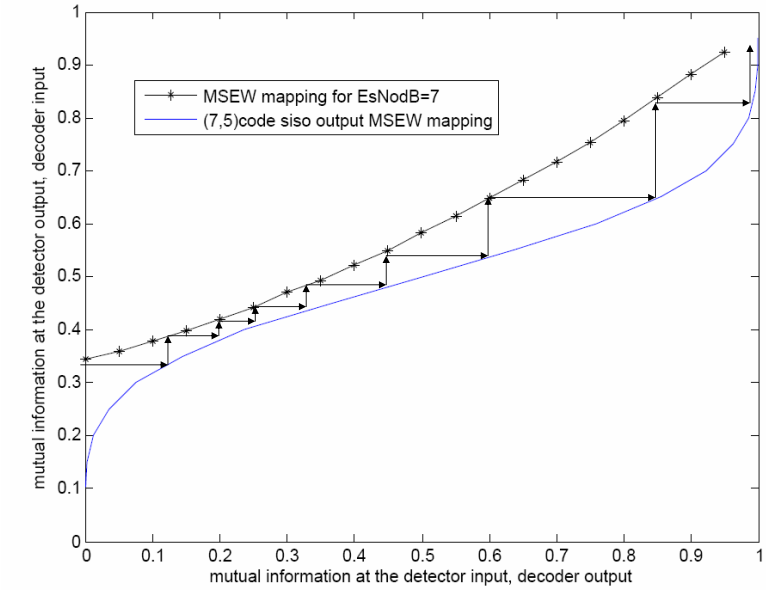

Figure 5.3: Extrinsic information transfer chart for $\mathcal{E}_s/N_0$=7dB for 16-QAM, MSEW .

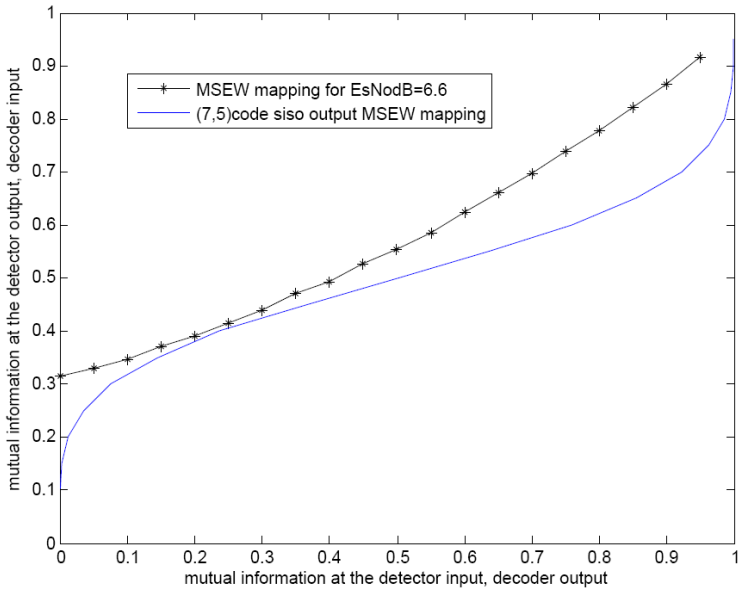

Figure 5.4: Extrinsic information transfer chart for $\mathcal{E}_s/N_0$=6.6dB for 16-QAM, MSEW.

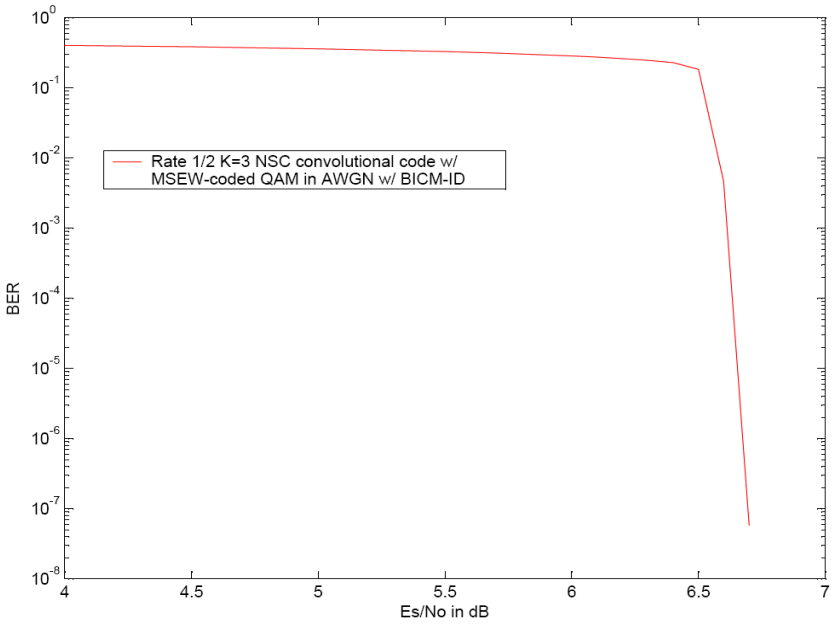

Figure 5.5: BER vs. $\mathcal{E}_s/N_0$ with iterative decoding for 16QAM,MSEW mapping in AWGN channel, R = 1/2 K = 3 NSC convolutional code.

Fig. 5.5 shows a typical BER curve for a system that uses a particular convolutional code [7,5], 16-QAM modulation with MSEW mapping. The threshold SNR value for which the BER drops sharply is determined analytically by using EXIT charts. The threshold SNR value is $\mathcal{E}_s/N_0 = 6.6$dB

When compared to the results obtained for MSEW and SP mappings, SP mapping has a lower threshold. SP mapping has better performance compared to MSEW mapping. MSEW mapping has a better error floor than SP mapping when it uses a larger interleaver.

5.2 Conclusion

BICM systems are commonly used in modern wireless systems. BICM-ID systems improve performance by iteratively exchanging information between demodulator and decoder. The convergence properties of iterative receivers in general, and BICM-ID in particular, may be investigated through the use of extrinsic information transfer (EXIT) charts. The

EXIT chart is a simple, yet powerful tool for visualizing the exchange of mutual information in iterative systems. This report has presented a detailed report on Extrinsic information transfer charts. The error floor of BICM-ID systems is characterized by a waterfall region and an error floor. The location of the waterfall region on the BER curve can be accurately predicted using EXIT charts.

That is, Given a BICM-ID system with a particular convolutional code and a particular modulation (including mapping), the threshold SNR value for which the BER drops sharply is determined analytically by using EXIT charts.

References

[1] E. Zehavi, "8-PSK trellis codes for a Rayleigh channel," *IEEE Trans. Commun.*, vol. 40, pp. 873–884, May. 1992.

[2] Caire, G. Taricco, and E. Biglieri, "Bit-interleaved coded modulation," *IEEE Trans. Inform. Theory*, vol. 44, pp. 927–946, May. 1998.

[3] X. Li and J. A. Ritcey, "Bit-interleaved coded modulation with iterative decoding," *IEEE Commun. Letters*, vol. 1, pp. 169–171, Nov. 1997.

[4] S. ten Brink, "Convergence of iterative decoding," *Electronics Letters*, vol. 35, pp. 1117–1118, Jun. 1999.

[5] J. Tan and G.L. Stuber, "Analysis and design of interleaver mappings for iteratively decoded BICM," *Proc. IEEE Int. Conf. on Commun. (ICC), Geneva, Switzerland*, pp. 1403–1407, May. 2002.

[6] R. G. Gallager, *Information Theory and Reliable Communication*, Wiley, 1968.

[7] J.G. Proakis, *Digital Communications*, 4th ed., New York, NY: McGraw-Hill, 2001.

[8] P. Elias, "Error-free coding," *IEEE Trans. Inform. Theory*, vol. 4, pp. 29–37, sept. 1954.

[9] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, pp. 1727–1737, Jun. 2001.

[10] Matthew C. Valenti, "Communication theory," Course Notes, WVU, SPRING 2007.

[11] Matthew C. Valenti, "Coding theory," Course Notes, WVU, SPRING 2008.

[12] Rohit Iyer Seshadri, "Capacity-based parameter optimization of bandwidth constrained cpm," Ph.d. dissertation, West Virginia University, 2007.

[13] David Scott Buckingham, "Information-outage analysis of finite-length codes," Master's thesis, West Virginia University, 2008.

[14] Tarik K Ghanim, "Analysis of hybrid-ARQ based relaying protocols under modulation constraints," Master's thesis, West Virginia University, 2006.

[15] Matthew C. Valenti, "Mutual information as a tool for the design, analysis and testing of modern communication systems," seminar presented to Hughes Networking Systems, June 2007.

[16] J. Hagenauer, "The EXIT chart - introduction to the extrinsic information transfer in iterative processing," submitted to EUSIPCO, 2004.

[17] S.Lin and D.J.Costello Jr., *Error Control Coding*, 2nd ed., Upper Saddle River, NJ: Prentice-Hall, Inc., 2004.

[18] F. Schreckenbach, N. Gortz, J. Hagenauer, and G. Bauch, "Optimization of symbol mappings for bit-interleaved coded modulation with iterative decoding ," *IEEE Commun. Letters*, vol. 7, pp. 593–595, Dec. 2003.

[19] A.J.Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, vol. 13, pp. 260–269, Apr. 1967.

[20] G. D. Forney, "Convolutional codes I: Algebraic structure," *IEEE Trans. Inform. Theory*, vol. 16, pp. 720–738, Jan. 1970.