CAPACITY ESTIMATION AND CODE DESIGN PRINCIPLES FOR CONTINUOUS PHASE MODULATION (CPM)

A Thesis

by

ARAVIND GANESAN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2003

Major Subject: Electrical Engineering

CAPACITY ESTIMATION AND CODE DESIGN PRINCIPLES FOR CONTINUOUS PHASE MODULATION (CPM)

A Thesis

by

ARAVIND GANESAN

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Krishna R. Narayanan (Chair of Committee)	
Costas N. Georghiades (Member)	Shankar P. Bhattacharyya (Member)
Donald K. Friesen (Member)	Chanan Singh (Head of Department)

May 2003

Major Subject: Electrical Engineering

ABSTRACT

Capacity Estimation and Code Design Principles for Continuous Phase Modulation (CPM). (May 2003)

Aravind Ganesan, B.Tech., Indian Institute of Technology, Madras

Chair of Advisory Committee: Dr. Krishna R. Narayanan

Continuous Phase Modulation is a popular digital modulation scheme for systems which have tight spectral efficiency and Peak-to-Average ratio (PAR) constraints. In this thesis we propose a method of estimating the capacity for a Continuous Phase Modulation (CPM) system and also describe techniques for design of codes for this system. We note that the CPM modulator can be decomposed into a trellis code followed by a memoryless modulator. This decomposition enables us to perform iterative demodulation of the signal and improve the performance of the system. Thus we have the option of either performing iterative demodulation, where the channel decoder and the demodulator are invoked in an iterative fashion, or a non-iterative demodulation, where the demodulation is performed only once followed by the decoding of the message.

We highlight the recent results in the estimation of capacity for channels with memory and apply it to a CPM system. We estimate two different types of capacity of the CPM system over an Additive White Gaussian Noise (AWGN). The first capacity assumes that optimum demodulation and decoding is done, and the second one assumes that the demodulation is done only once. Having obtained the capacity of the system we try to approach this capacity by designing outer codes matched to the CPM system. We utilized LDPC codes, since they can be designed to perform very close to capacity limit of the system. The design complexity for LDPC codes

can be reduced by assuming that the input to the decoder is Gaussian distributed. We explore three different ways of approximating the CPM demodulator output to a Gaussian distribution and use it to design LDPC codes for a Bit Interleaved Coded Modulation (BICM) system. Finally we describe the design of Multi Level Codes (MLC) for CPM systems using the capacity matching rule.

ACKNOWLEDGMENTS

I would like to thank Dr. Krishna R. Narayanan, chair of my committee, for his guidance and encouragement. His insights were invaluable to my research and his time and effort is greatly appreciated.

Most importantly I would like to thank my parents, sister and brother in law for their love and encouragement.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

Continuous Phase Modulation (CPM) is a digital phase modulation technique which constrains the phase of the carrier to be continuous over signalling intervals. The phase pulse ($p(t)$)of CPM describes how the transition takes place between the signalling intervals. This constraint on the phase implies that the CPM is a modulation with memory. Due to the smooth phase transitions M-ary CPM has better spectral efficiency than other phase modulation schemes like M-PSK. Also since the peak to average power ratio of CPM is the best possible value (0dB) it is popular in applications which have stringent peak-to-average power ratio constraints. Apart from these benefits, CPM modulator can be decomposed into an outer encoder (Continuous Phase Encoder - CPE) and a memoryless modulator, this fact can be used to improve the performance of the system by performing iterative demodulation.

The decomposition of the CPM also allows us to explore different realizations of CPM easily. In the equivalent representation of CPM, the input information stream is passed to the CPE, whose output is then mapped to the transmitted signal by a memoryless modulator. The number of unique signals that can be transmitted during any signalling interval is determined by the characteristics of the phase pulse $p(t)$ and is independent of the nature of CPE. Thus by changing the nature of the CPE we can obtain different realizations of the CPM. Although the signal set of these different realizations of CPM are the same, the performance of these CPM systems is different due to differences in the overall mapping of the input sequence to the signal waveform.

Different communication systems can be compared using the Bit Error Rate

---

The journal model is *IEEE Transactions on Automatic Control*.

(BER) performance as the criteria. The BER performance of a practical communication system is augmented by using a channel code which introduces redundancy into the transmitted message and has the capability to correct certain kinds of errors. The redundancy in the output messages implies that there are some sequence of symbols that are not produced by the encoder. Systems with a channel code followed by a modulator are called constrained modulation systems.

An important quantity that represents the system limit is the capacity of a channel. Capacity of a channel, or a system viewed as an equivalent channel, is the maximum rate at which information can be reliably transmitted over the channel. The knowledge of capacity of a given channel helps us to compare the simulated performance of the designed system with the theoretical limit. This gives us an indication of whether any further improvement can be made to the system through better design.

The computation of the capacity of a system involves maximizing the mutual information between the input to a channel and its output over all possible input distributions. Due to the inherent memory introduced in the modulation (due to the continuous phase constraint) the CPM system can be viewed as a channel with memory. The capacity of a channel with memory and constrained inputs cannot be easily computed. We are unaware of any published results on the capacity for CPM systems. Currently the performance measure used is the cut-off rate [1], but with Turbo-codes and Low Density Parity Check (LDPC) codes performing better than the cut-off rate, we need a better measure of performance. Recently there have been some developments in the capacity estimation for Inter Symbol Interference (ISI) channel ([2],[3],[4]), which is a channel with memory. The computation of the capacity for CPM systems is based on these recent developments.

Since the CPM system can be decomposed into an outer encoder (CPE) and a

memoryless modulator, the system with an Error Correcting Code (ECC) and CPM can be viewed as a serially concatenated system. This motivates us to apply iterative techniques to the demodulation, where the decoder and the demodulator exchange extrinsic information between each other. The optimum detector for CPM is an Maximum A-posteriori Probability (MAP) algorithm like BCJR [5] algorithm. This further facilitates the soft information exchange between the demodulator and the outer code.

The constrained capacity of a system is the maximum information rate that can be transmitted over the channel in conjunction with a channel ECC. This capacity can be achieved by using a good channel code which performs well over the given channel. The rate of a code is defined as the ratio of the number of input symbols to the output symbols processed by the code. In order to achieve the capacity of the system the code needs to have its rate equal to the capacity and be able to perfectly decode the messages corrupted by the channel. In practice a code with a given rate will not perform the same with different channels, i.e. a code that decodes messages perfectly for a Phase Shift Keying (PSK) system over Additive White Gaussian (AWGN) channel might perform poorly for a CPM system over AWGN channel. The code hence not only needs to have the rate equal to the capacity of the system but also needs to be designed by taking the channel output characteristics into consideration.

Codes which are designed as described above are said to be matched to the channel. LDPC codes have gained popularity due to the availability of analytical tools which help us in predicting the performance of these codes over a large number of channels. For example Density Evolution (DE) technique developed by Richardson [6], allows us to analyze the performance of the belief propagation algorithm over a large number of channels. There have also been developments in algorithms which search for good codes for LDPC codes. These techniques have been used to design

codes for simple channels. The extension to channels like higher order CPM require new algorithms which take into account the characteristics of the modulation like dissimilar bit distribution.

Matched Bit Interleaved Coded Modulation (MBICM) and Multilevel coding (MLC) are two popular code design techniques which are used to design systems to approach the capacity of the system. The idea behind conventional MBICM is to design a single component binary code whose performance is optimum for the given channel. With MBICM the input bits are encoded using the designed code of appropriate rate, the output codeword is then interleaved at the bit level and the interleaved codewords are mapped to M-ary symbols by grouping appropriate number of bits together. We will propose new design procedure to optimally design codes using irregular LDPC codes. Thus with the knowledge of a channel's output distribution we can design capacity approaching codes by following certain design principles. The design procedure relies on representing the output distribution of the channel with equivalent channels. We will explore different ways of representing the channel and investigte the differences between the various methods.

Multilevel coding is a code design approach which jointly optimizes the coding and modulation to achieve significant coding gains. It is applied to higher order (typically 2^L) modulation schemes. In such a modulation scheme, each signal point is associated with a length L binary address. The input data is then split into L streams, and each stream is encoded by a component encoder with rate R_i for each $i = 1 \dots L$. Thus the channel can be broken down into L component channels. It has been shown in [7] that the the transmission over the channel can be separated in to parallel transmission of L binary bits over equivalent channels at each level i, provided that the bits at the lower levels $(0, 1, \dots, i-1)$ are known. The capacity design rule requires the code rates R_i to be equal to the capacity of the equivalent

channel at level i, which is equal to the capacity of the channel with the knowledge of the bits at levels $0, 1, \ldots, i-1$. Due to the complexity of estimating the capacities of the equivalent channels this design procedure has not been properly applied to M-ary CPM systems.

In this thesis we first estimate two different lower bounds on capacity for a M-ary CPM scheme based on some recent developments in the capacity estimation for channels with memory ([2],[3],[4] [8]). The two capacities we are interested in are the independent and identically distributed (i.i.d) capacity - C_{iid} and the non-iterative or one-shot demodulation capacity - C_{ni}. The i.i.d. capacity estimates the capacity assuming that the inputs to the modulator are independent and identically distributed from a finite set. This limit can be achieved in practice by Maximum Likelihood (ML) decoding. Due to the practical difficulties in performing Maximum Likelihood (ML) decoding/demodulation, we typically employ iterative demodulation/decoding techniques to achieve capacity. We will show that iterative technique can perform well by designing systems which perform very close to capacity. Since iterative demodulation requires substantially higher complexity, we need to know the capacity degradation due to non-iterative demodulation, i.e. when the demodulation is performed only once. We call the threshold of the system for which the demodulation is performed only once as the non-iterative capacity of the system.

Having obtained the capacity estimates for this system, we propose practical design techniques which try to achieve capacity. We will consider the design of LDPC codes that are matched to the channel using MBICM and MLC techniques. The design procedure involves the tracking of the message densities in the decoder which involves the convolution of densities. This is a computationally intensive procedure and the design process can be efficiently implemented by making the assumption that the message passed to the decoder is Gaussian distributed. The output of the CPM

demodulator for higher order modulation schemes is not Gaussian distributed. In this thesis we consider three different ways of approximating the demodulator output to a Gaussian distribution.

The organization of this thesis is as follows. In Chapter II gives an introduction to CPM and describes the system setup and the notation used in this thesis. We review the decomposition approach to CPM and illustrate how it can be exploited to perform iterative demodulation. In Chapter III we briefly review LDPC codes. We review the construction of these codes along with the encoding and decoding algorithm for these codes. We briefly explain how the codes can be designed. In Chapter IV we review the different definitions of capacity and introduce the mathematical expressions used to evaluate them. We then show how these quantities can be estimated. In Chapter V we describe the design techniques to obtain codes for iterative and non-iterative demodulation. We concentrate on MBICM and MC design techniques and design codes for BICM technique for iterative and non-iterative demodulation. We also present the simulation results for the performance of the BICM codes for iterative and non-iterative demodulation and the equivalent capacity estimate for MLC. Finally in Chapter VI we present the conclusion.

CHAPTER II

CONTINUOUS PHASE MODULATION

A. Continuous Phase Modulator as a Digital Modulator

Digital transmission of data offers offers a lot of advantages over analog forms of transmission like error protection, compression, etc. Since most of the physical media are not conducive to direct transmission of digital data, the transmission of digital data over such a medium, like a wireless channel or a copper cable, requires the use of a digital modulator. A digital modulator converts the discrete time digital data into an analog waveform which can be transmitted over the analog channel. The conversion of digital sequence into analog waveform is known as modulation and the reverse process is known as demodulation. The analog waveform over which the information is transmitted is called the carrier. The information is embedded in the carrier by making detectable changes to the characteristics of the pure waveform. The Different digital modulation schemes differ in the way the carrier is altered by the digital signal.

Digital phase modulation schemes use the phase of the carrier to carry the information. Phase Shift Keying (PSK) and Frequency Shift Keying (FSK) are two main types of phase modulation. In PSK the input sequence selects the phase of the signal at each signalling interval from a finite set of values. In FSK, the input sequence changes the frequency of the signal around the carrier frequency by discrete values. Since the instantaneous frequency of a signal is obtained by taking the derivative of the phase, FSK can also be viewed as a modulation scheme in which the phase of the signal is changed through a linear function. This function is referred to as the phase modulating function and an example is shown in Fig. 1.

CPM is a form of digital phase modulation where the phase of the carrier is kept continuous between signaling intervals. If the phase modulation function is a linear function of time, within the signalling interval, then the scheme is referred to as Continuous Phase Frequency Shift Keying (CPFSK). In general the phase modulation function need not be linear function of time. CPM modulator can be viewed as a finite state machine whose state is defined by the accumulated phase at the start of the signaling interval and whose output is dependent on the current state and the input symbol. The finite state machine model of CPM helps us view the modulation process as a path through a phase trellis as shown in Fig. 2. Due to the phase continuity of the carrier signal CPM has high spectral efficiency. The spectral efficiency can be improved by making use of a smooth phase modulating function.

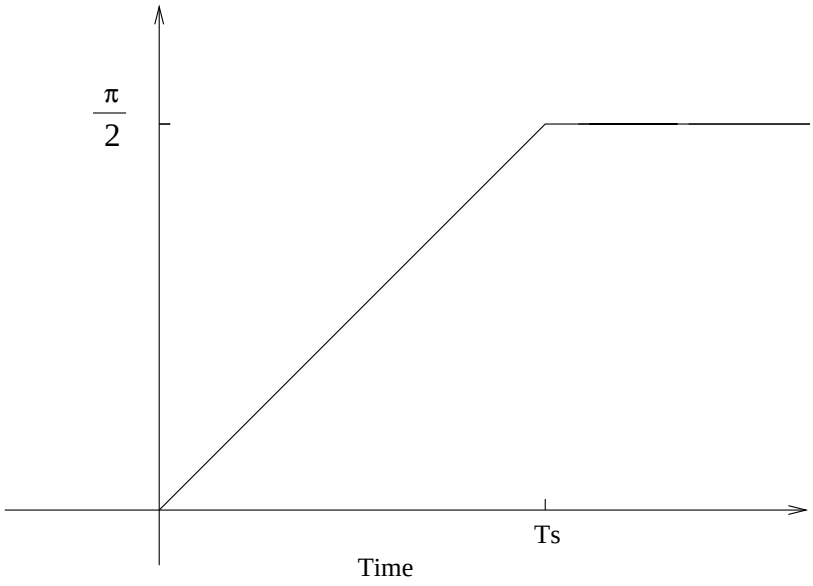

Fig. 1. Phase modulating function

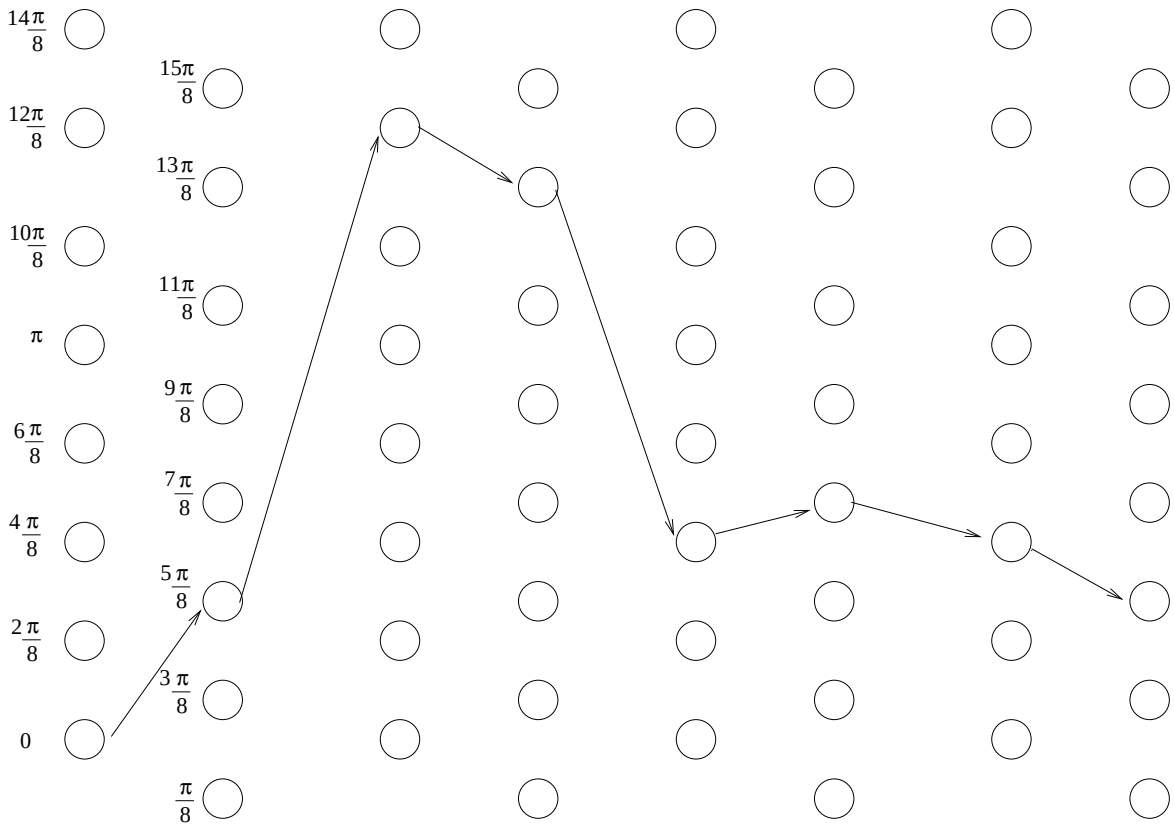

Fig. 2. Phase trellis

B. System Model

A general M-ary CPM system can be represented by a block diagram as shown in Fig. 3.

The input to the system is a binary sequence $\mathbf{X}_b = [X_{b\,0}, X_{b\,1}, \ldots, X_{b\,N_b-1}]$ of length N_b, which is mapped to an M-ary data sequence $\mathbf{X} = [X_0, X_1, .., X_{n_s-1}]$ of length N_s where $X_i \in [0, 1, \cdots, (M-1)]$. If $M = 2^A$, $N_b = N_s \times A$. The modulator operates at a symbol rate T and transmits the pass-band signal $s(t, \mathbf{X})$ given by

$$s(t, \mathbf{X}) = \sqrt{\frac{2E}{T}} \cos(2\pi f_o t + \varphi(t, \mathbf{X}) + \varphi_0), t \geq 0 \tag{2.1}$$

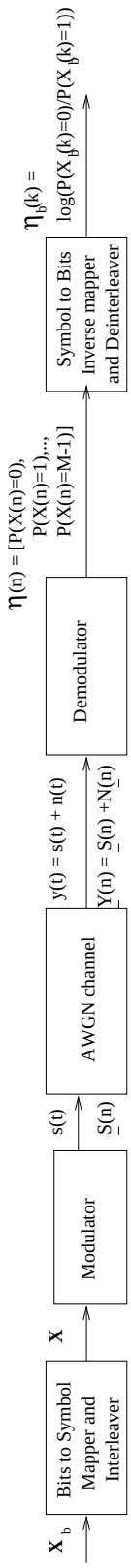

Fig. 3. System model

and

$$\varphi(t, \mathbf{X}) = 2\pi h \sum_{i=0}^{\infty} X_i f(t - iT)\,,\ t \geq 0 \tag{2.2}$$

h is the modulation index, φ_0 is the initial phase and $f(t)$ is the phase pulse of the system, which has the following properties :

$$f(t) = \begin{cases} 0 & t \leq 0 \\ \frac{1}{2} & t \geq LT \end{cases} \tag{2.3}$$

L is a positive integer which represents the duration of the phase pulse.

A CPFSK system is a full response CPM (i.e. L=1) with a linear phase change, i.e.

$$f(t) = \frac{t}{2T} \quad 0 \leq t < T$$

. The modulation index h is $1/M$. The phase notation used in this thesis is the true (physical) phase. With the above conditions, the system has M allowed states at each time instant (total of $2 \times M$ states of which only half are allowed at a given time). If the tilted phase representation ([9] , [10]) is used then the total number of states at any given time interval will be M. Each active state has M possible transitions during every signaling interval corresponding to M different input symbols. Each of these transitions is associated with a signal, and we denote the set of signals transmitted in even or odd signaling interval as $\mathbf{S}^E$ and $\mathbf{S}^O$ respectively and the union of these signals by $\mathbf{S}^T$ ($\mathbf{S}^T = \mathbf{S}^O \cup \mathbf{S}^E$). $\mathbf{S}^T$ has $2 \times M \times M$ signals. The signals in the odd and even time instances differ only by a constant phase offset.

The output of the channel represented in continuous time is $y(t) = s(t, \mathbf{X}) + n(t)$ where $n(t)$ is a white Gaussian process. Using Gram Schmidt Ortho-normalization, the system can also be represented using a vector notation, as described in section D. Using the vector representation the channel output is represented as $\underline{Y}_n = \underline{S}_n + \underline{Z}_n$ $1 < n < N_s$. The received signal is processed by the demodulator to produce the

symbol likelihoods $\eta(n) = [Prob(X_n = 0), Prob(X_n = 1), \ldots, Prob(X_n = M-1)]$ for each discrete time instant $n \in [1, 2, \ldots, N_s]$.

The M-ary CPM modulator works at the symbol level with its input symbol alphabet belonging to the set $= [0, 1, \ldots, (M-1)]$. As we are interested in concatenating CPM nodulator with an outer binary code we need to map the output bits of the outer code to M-ary symbols through a mapper which is a bijective function. Although the capacity of the system does not depend on the mapping strategy used, optimum decoding for most concatenated systems has very high complexity and hence we use iterative demodulation/decoding as an alternative. Because of this, the mapping of bits to symbols has an effect on the performance of iterative demodulation/decoding, and the mapper should be designed carefully.

To obtain a good mapping we look at the distance spectrum of the signal at each time interval. The mapper then tries to map bits to symbols in such a way that the signals which are closest to each other differ by the least number of bit positions. This ensures that the bit errors at the output of the demodulator are minimized. For example in the case of 8-ary CPFSK the signals corresponding to adjacent symbols (eg. [0,1] , [3,4]) have the smallest distance. One of the mapping that maps symbols, closest to each other, to bits which differ in only one position is given in table I.

C. Equivalent Representations

The continuity imposed on the phase of the signal means that CPM has memory inherently built into it. The fact that CPM exhibits memory was used in the alternative representation of the CPM modulator [11]. This representation allows us to view the CPM modulator as an outer encoder (Continuous Phase Encoder - CPE), which is a finite state machine, followed by a memoryless modulator. This allows us

Table I. Mapping for 8-PSK

Bits	Symbol
000	4
001	5
010	7
011	6
100	3
101	2
110	0
111	1

to study the encoding (CPE) operation independent of the signal mapping and allow us to form different realization of the CPM. For CPFSK the finite state machine can be realized as a convolutional encoder over the ring of integers modulo-M [11].

We investigate the properties of two realizations of the CPM - one with a recursive CPE and the other with a non-recursive CPE. The CPEs for the two realizations are shown in Fig. 4. The inputs to the CPE is an M-ary data sequence $\mathbf{X}$ the two outputs of the encoder are mapped by the memoryless modulator, which has the same signal set ($\mathbf{S}^T$) for all realizations of the CPE. The memoryless modulator maps the outputs of the CPE to to a signal $\mathbf{S}_i \in \mathbf{S}^T$. Equation (2.1) leads to the the CPM realization as shown in (4a) which has a recursive CPE which has been studied extensively ([12],[13]) in a serial concatenated coding structure. Although the non-recursive realization (4b) is well known it has not been widely used in concatenated schemes. We will study the two realizations in detail and analyze their performance in the context of a constrained system.

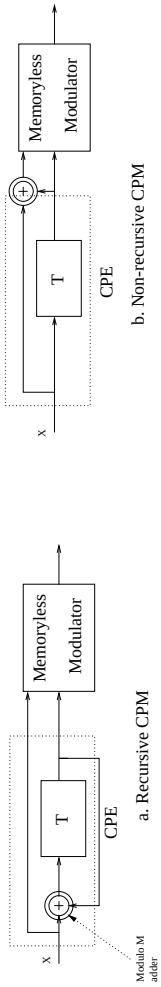

Fig. 4. Continuous phase encoders

D. Signal Space Representation

The demodulation of the CPM signal is usually done by extracting the sufficient statistics from the received signal. It can be shown that for the MLSE demodulator, for AWGN channels, the sufficient statistics are the output of the correlators which are matched to the ortho-normal basis functions of the input signal space. The ortho-normal basis functions of the CPM signal set for small alphabet size can be easily derived using the Gram-Schmidt ortho-normalization technique.

As discussed earlier, the CPM can be decomposed into a CPE and a memoryless modulator. The CPE introduces memory into the modulation and its output is then mapped to a signal $s(t) \in \mathbf{S}^T$ where $\mathbf{S}^T$ is the set of all possible signals that can be transmitted by the memoryless modulator. A Gram-Schmidt ortho-normalization of the signals $\mathbf{S}^T$ can be performed to yield d ortho-normal functions. The transmitted signal on each transition can then be projected on to the ortho-normal basis functions to obtain a vector representation of the signal. The transmission then can be easily modelled as a discrete-time vector channel (of dimension d) with the input signal represented as the projection, $\underline{S_i} = [S_{i,1}, S_{i,2}, \ldots, S_{i,d}]$, of the signal $S_i(t)$ on the ortho-normal basis functions. The AWGN channel can be modelled as the addition of i.i.d. Gaussian white noise of variance σ^2 to each component of the transmitted vector, where

$$\sigma^2 = \frac{1}{2\,\frac{E_s}{N_0}} = \frac{1}{2\,\frac{E_b}{N_0}\,R}$$

and

$$\frac{E_s}{N_0}$$

is the Signal to Noise Ratio (SNR).

E. Demodulation

The decomposition of the CPM into a finite state machine (CPE) and a memoryless modulator makes it clear that the optimum soft output demodulator for CPM system is a Maximum A-posteriori Probability (MAP) algorithm like the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [5].

The BCJR algorithm can be applied to the output of a Markov chain process corrupted by AWGN to produce *aposteriori* probabilities on the input symbols $(P(X_i), X_i \in \mathbf{X})$ and hence can be applied to CPM system. We detail the algorithm in Appendix A. The probabilities of the symbols need to be converted to bit probabilities and then passed to the channel decoder.

The BCJR algorithm can also make use of *apriori* probabilities of the symbols. This helps us in iterative demodulation, where the extrinsic information (for symbols or bits) from the channel decoder is converted to symbol probabilities and fed to the demodulator as a-priori information. This forms the basis of an iterative demodulation system.

CHAPTER III

LOW DENSITY PARITY CHECK CODES

Low Density Parity Check were first proposed by Gallager [14] and rediscovered recently [15],[16],[17]. LDPC codes are a class of block codes with special construction and decoding methods. Any codeword $\mathbf{x}$ of a linear code follows the property $\mathbf{Hx^T} = \mathbf{0^T}$, where $\mathbf{H}$ is the Parity Check Matrix (PCM). These codes are called Low Density Parity Check codes since the parity check matrix associated with these codes is filled with low fraction of non-zero values. The degree of the row or column is the number of non-zero entries in the given row or column of the PCM. LDPC codes are analyzed as an ensemble of codes with a given degree distribution ρ and λ which specify the check and variable node degree distribution. In this chapter we will describe the construction of these codes and explain their encoding and decoding operations.

A. Construction of LDPC Codes

In this section we will introduce the construction of LDPC codes. The performance of the LDPC codes is analyzed on an ensemble of codes with parity check matrices specified by the row and column degree profiles ($\rho(x)$ and $\lambda(x)$ respectively). The degree profile (node prospective) enumerates the fraction of nodes with different degrees. For example LDPC codes with row degree profile $\rho(x) = \sum_{i=1}^{\deg_\rho} \rho_i x^{i-1}$, has ρ_i fraction of degree i nodes. Once the degree profile has been optimized for a given channel, the encoder and decoders are designed from the degree profiles.

These codes can be easily analyzed through the bipartite representation introduced by Luby et al [18]. Any PCM can be represented as a bipartite graph. For example the PCM shown in Table II can be represented as in Fig. 5. The bipartite graph consists of two types of nodes connected through edges. The nodes on the

Table II. Example of a parity check matrix

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

left hand side of the graph represent the coded (variable) bits and the nodes on the right hand side of the graph represent the check conditions imposed by the PCM. The edge connecting a variable node and a check node indicates the participation of the variable node in the parity check represented by the check node. The degree of a node can be interpreted as the number of edges connected to it. The decoding of the LDPC codes can be achieved through a message passing algorithm, which iteratively exchanges messages between the variable and check nodes.

The performance of the codes can be analyzed through the density evolution algorithm, which predicts the convergence of the message passing algorithm under the assumption of a cycle free graph. The cycle free assumption is necessary to guarantee that the incoming messages to a node are independent. The cycle free nature of the graph requires infinite length code word length. In practice for finite length codeword, the performance of the code can be improved by designing graphs with large minimum cycle lengths. The graphs are usually designed by random construction. A simple constraint on the graph is to make sure that no two edges are connected to the same set of nodes. This constraint is sufficient when the code-word length is sufficiently large.

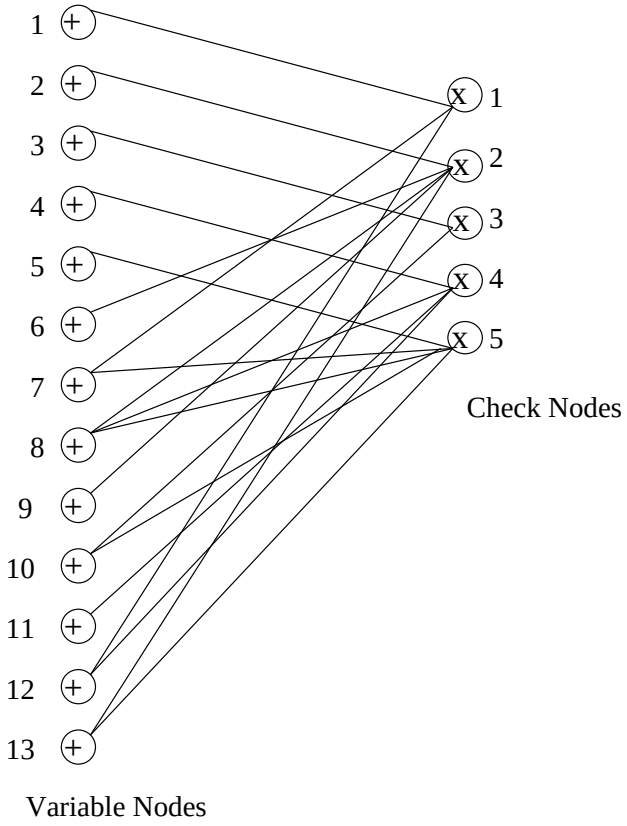

Fig. 5. Bipartite graph structure of the LDPC code

B. Encoding Operation

Since LDPC codes are block codes, the encoding operation is performed through the generator matrix ($\mathbf{G}$), i.e. for an (N, K) block code, the input information sequence of length K ($\mathbf{x}$) and the output sequence of length N ($\mathbf{c}$) are related by $\mathbf{c} = xG$.

The LDPC codes are designed through the PCM ($\mathbf{H}$), hence we need to derive the generator matrix from the PCM. Consider a LDPC code with input information sequence length of K and output code-word length of N, when its PCM expressed in the systematic form , i.e. as $\mathbf{H} = [I \mid P]$, the generator matrix has the form $\mathbf{G} = [-P^T \mid I]$, where $\mathbf{P}$ is a $N - K \times K$ matrix and $\mathbf{I}$ is an identity matrix of size $\mathbf{N} - K$.

The PCM for an LDPC code is randomly constructed; because of this, it is not necessarily in the systematic form. Hence we need to convert the PCM into the systematic form through Gaussian elimination. Since we are interested in fairly long code-word lengths, in general the Gaussian elimination is a computationally intensive operation. Since LDPC codes are linear codes, where in sum of any two code-words is also a code-word, the performance of the codes can be analyzed by considering the transmission of any one code word. The encoding operation can be avoided by assuming that the input to the encoder is an all zeros sequence, which is mapped to the all zeros sequence. This assumption can be extended to other systems concatenated to the encoder if those systems are also linear. In this thesis we are interested in concatenating the LDPC code with a CPM system, which is a non-linear system. We will need to take care of the non-linear nature of the CPM system by converting flipping the all zeros sequence to a random sequence and transmitting the random sequence through the CPM, and undoing this bit operation at the output of the demodulator.

C. Decoding Operation

For the general block codes, the decoding is performed by a combination of forming hard decisions, $\hat{\mathbf{x}}$, on the demodulator output, then computing the parity check bits $\mathbf{p}$ through $\mathbf{p} = \hat{x}H$ and finally obtaining the error bit positions using a lookup table. The complexity and the memory requirement for the decoder increases exponentially with increasing code-word length. Since the code-word length of a LDPC code is typically large, the decoding algorithm described above will prove to be computationally intensive and have very large memory requirements. Also since the decoder uses only the hard decisions, the performance can be improved by using an algorithm that can use the soft decisions provided by the demodulator.

The message passing algorithm makes use of the soft information of the coded bits. The message passing algorithm involves the exchange of processed information between the variable and check nodes. Each node receives extrinsic information from edges connected to it and then processes the information according to its type and then passes back the extrinsic information through the edges.

The operation performed at the edges depends on its type. In this thesis we will deal with binary LDPC codes with sum-product decoding. For binary LDPC decoding its convenient to represent the messages exchanged between the nodes as Log Likelihood Ratio (LLR), which is the logarithm of the ratio of probability of bit being equal to 1 (P_1) and probability of bit being equal to 0 (P_0).

The variable nodes represents the coded bit, and the edges connected to the node carry messages for the same bit. Since the messages incident on the node represent the same bit, the operation at the variable node is arithmetic addition of the messages. Since the messages exchanged through the edges have to be extrinsic messages, the sum of LLRs should not involve the incoming message on that edge. Therefore the

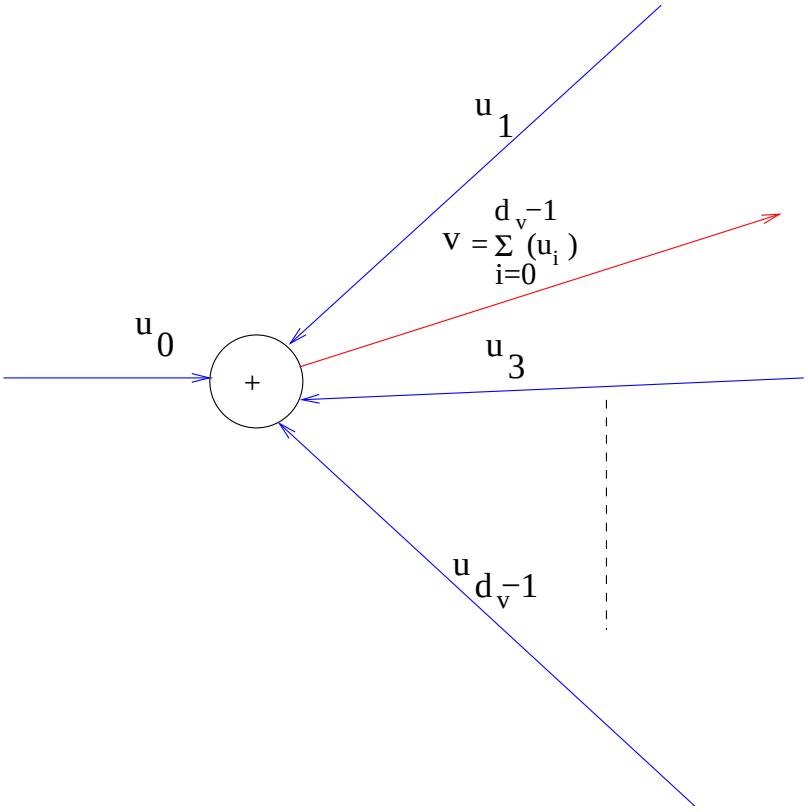

Fig. 6. Operation at the variable node

updated message for an edge connected to a variable node of degree d_v is

$$v = \sum_{i=0}^{d_v-1} u_i \tag{3.1}$$

where v is the output message, u_0 is the LLR for the bit obtained from the channel and u_i $i = 1 \ldots d_v - 1$ are the input messages from all the edges of the bipartite graph connected to the variable node except the edge on which the message is updated. Fig. 6 shows this operation graphically.

The check node represents the parity check constraint imposed by the PCM. The incident edges represent different coded bits and hence the messages cannot be simply added arithmetically. It can be shown that the output message u of a degree d_c check

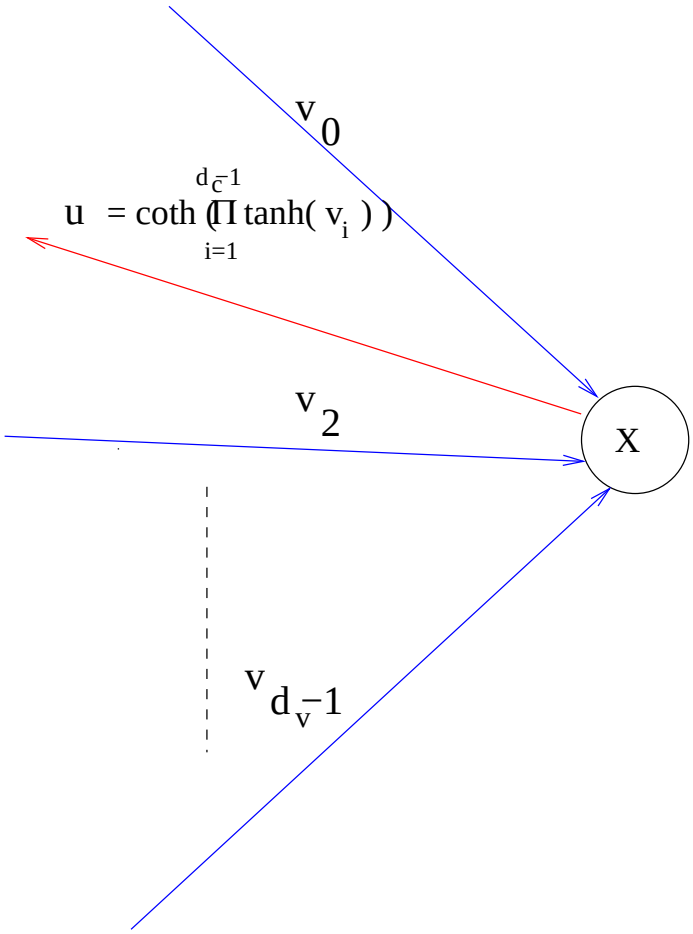

Fig. 7. Operation at the check node

node with incident messages v_i $i = 1 \ldots d_c$ is

$$u = \coth \prod_{i=0}^{d_c-1} \tanh v_i \tag{3.2}$$

where the product is carried over all the edges except the edge carrying the updated messages. Fig. 7 shows this operation graphically.

In sum-product decoding, at the start of decoding, the edges in the graph initially carry no messages. The channel provides the estimate for the coded bits, the decoding then starts with the processing at the variable nodes. The updated messages are then processed at the check nodes and the extrinsic message is sent back to the variable

nodes and the procedure repeated a fixed number of times. The iterations can be stopped if the hard decisions on the coded bits $\hat{\mathbf{x}}$ satisfy the condition $\hat{\mathbf{x}}H = 0$. If the decoded bits do not satisfy the above condition, then a decoder failure is indicated.

CHAPTER IV

CAPACITY ESTIMATION

A. Capacity Definition

The channel capacity is the maximum rate at which information can be transmitted with a single use of the channel with arbitrarily low probability of error. It is usually expressed as bits per second per hertz (bps/hz). The capacity of the channel is a useful measure as it tells us the highest rate at which information can be reliably transmitted. The information theoretic definition of capacity helps us calculate the channel capacity. For a memoryless channel with input X and output Y, the capacity is the maximum of the mutual information between X and Y $(I(X;Y))$, over all possible distributions of X. The channel capacity is completely specified by X, Y and the conditional distribution $f_Y(y \mid x)$. But in general, for arbitrary channels, the calculation of the capacity through the above equation is difficult, because the maximization of the mutual information has to be carried over all possible input distributions $(f_X(x))$. The channel capacity of some memoryless channels like Additive White Gaussian Channel (AWGN) with continuous input, Binary Symmetric Channel (BSC) which has discrete inputs and outputs, can be computed easily, since the properties of the channel make it easy to find the distribution that maximizes the mutual information.

For channels with memory the information theoretic definition of capacity is maximum of

$$\lim_{n\to\infty} \frac{1}{N} I(X_1^N; Y_1^N)$$

, over all possible distributions of the input sequence X_1^N, where the notation $_1^N$ denotes a partial sequence of X consisting of elements $1, \ldots, N$. The capacity of such

a system is hard to compute even by making further simplifying assumptions. In this section we will see how this expression can be estimated.

First, we provide a more formal definition of the capcity of a channel with memory. For a channel with M-ary inputs $\mathbf{X} = X_1^N = (X_1, X_2, \ldots, X_N)$ and outputs $\underline{\mathbf{Y}} = \underline{Y}_1^N = (\underline{Y}_1, \underline{Y}_2, \ldots, \underline{Y}_N)$ the capacity of a channel is then defined as

$$C = \lim_{n\to\infty} \frac{1}{N} \sup_{Pr(X_1^N = x_1^N)} I(X_1^N; \underline{Y}_1^N)$$

The maximization in the above expression needs to be performed over all possible distributions of the sequence X_1^N. As $N \to \infty$ this maximization procedure will be practically impossible to compute. Instead of this we will compute the information rate of the channel under an assumed distribution. We will assume that the input to the channel is usually independent and identically distributed (i.i.d.) and compute the information rate of the system under such an input distribution ($Pr(X_1^N) = \prod_{i=1}^{N} Pr(X_i)$). This information rate is referred to as i.i.d. capacity C_{iid}.

This capacity (C_{iid}) can be achieved by a system with a channel code with rate equal to the capacity of the channel and a jointly optimum detector. Such a system has very high complexity and we will show that iterative demodulation can come close to the capacity with reasonable complexity. To further limit the complexity of the system, we are also interested in the non-iterative capacity of the system with i.i.d. inputs. This is the capacity of the system assuming that the input to the channel is a sequence of M-ary i.i.d. symbols and demodulation is performed only once. This capacity will help us in quantifying the loss in performance by using a sub-optimum detector where the demodulation is performed only once.

Most of the channel codes are designed to correct signals which are uncorrelated. Since the output of the demodulator (which is a BCJR algorithm) is correlated we assume that correlation is removed by the interleaver at the input of the decoder.

Fig. 3 shows this interleaver being combined with the bits to symbol mapper. This interleaving is done either at the symbol level or at the bit level. In the former case the interleaver follows the bits to symbol mapper, where as in the latter case the interleaver precedes the bits to symbol mapping. The two different capacity estimates are referred to as symbol interleaved non-iterative capacity C_{ni}^{s} and bit-interleaved non-iterative capacity C_{ni}^{b}.

B. IID Capacity Calculation

The i.i.d. capacity of the CPM system is the maximum information rate supported by a given channel given that the inputs to the modulator are i.i.d. Since the channel is symmetric the input distribution that maximizes the rate is the uniform distribution. We need to calculate

$$C_{iid} = \sup_{Pr(\mathbf{X}_1^N = x_1^N) = \Pi_{i=1}^{N} Pr(\mathbf{X}_i = x_i)} I(X; \underline{Y})$$

where $I(X; \underline{Y})$ can be estimated as

$$I(X; \underline{Y}) = \lim_{N \to \infty} \frac{1}{N} I(\mathbf{X}_1^N; \underline{Y}_1^N)$$

where $\Pr(X_i = x_i) = \frac{1}{M}$. From the definition of mutual information we have $I(X; \underline{Y}) = h(\underline{Y}) - h(\underline{Y} \mid X)$. From our system model we know that $\underline{Y}_n = \underline{S}_n + \underline{Z}_n$, hence $h(X \mid \underline{Y})$ is nothing but $h(\underline{Z})$ as the modulator maps $\mathbf{X}$ uniquely to $\underline{\mathbf{S}}$. Therefore all we need to do is estimate is $h(\underline{Y}) = \frac{1}{N} h(\underline{Y}_1^N)$. For this we use the method followed in [2] where the authors have estimated the capacity of a binary input channel with memory. This method essentially uses the forward recursion of a BCJR algorithm to estimate the channel output entropy $h(\underline{Y})$. The algorithm can be applied to any channel that can be represented as a finite state machine with additive noise, such as

CPM modulator.

We know that $h(\underline{Y}_1^N)$ can be estimated as $-E[\log(Pr(Y_1^N))]$. Thus we can compute $Pr(\underline{Y}_1^N)$ for many realizations of the input X_1^N and channel $\underline{Z}_1^N$ and take the average value of $-[\log(Pr(Y_1^n))]$ as the estimate of $h(\underline{Y})$. But we know from Shannon-McMillan-Breimann theorem [19], that for a stationary ergodic finite state hidden Markov process χ that

$$\frac{1}{N}\log(Pr(\chi_1^N)) \rightarrow h(\chi)$$

with probability one. Thus with a long simulation we can estimate $h(\underline{\mathbf{Y}})$ with a single input realization $\mathbf{X}$ and channel realization $\underline{\mathbf{Z}}$.

To estimate $h(\underline{Y}_1^N)$ we make use of the forward recursion of the BCJR algorithm. We note from Appendix (A) that for a system which can be represented as a finite state machine with Θ^M states, defining $\alpha_n(m)$ as $Pr(\Theta_n = m, \underline{Y}_1^n)$ and $\gamma_n(x, m', m)$ as $P(\Theta_n = m, \underline{Y}_n, X_n = x \mid \Theta_{n-1} = m')$, we can obtain the recursion (normalized form)

$$\alpha'_n(m) = \sum_{m,m'} \gamma_n(m', m) \times \alpha'_{n-1}(m') \times \lambda_n \text{ for } 1 \leq n \leq N$$

where

$$\lambda(n) = \frac{1}{\sum_{m,m'} \gamma_n(m', m) \times \alpha'_{n-1}(m')}$$

The initialization for this recursion sets the $\alpha' s$ for the starting state to a known value. In our analysis we always start from the state zero and hence

$$\alpha_0(i) = \begin{cases} 0 & \forall\, i \neq 0 \\ 1 & \text{for } i = 0 \end{cases}$$

We can now estimate $Pr(\underline{\mathbf{Y}})$ as the sum of the un-normalized values of α_N for each

state. i.e.

$$\begin{aligned} \log(Pr(\underline{\mathbf{Y}})) &= \log(\sum_{m=1}^{\Theta_m} \alpha_n(m)) \\ &= \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \log(\lambda_i) \end{aligned} \tag{4.1}$$

Thus by choosing a sufficiently large value of N we can obtain a good estimate of $h(\underline{Y})$. Having obtained $h(\underline{Y})$ the capacity can now be calculated as $C_{iid} = h(\underline{Y}) - h(\underline{Z})$. If in our system $\underline{Z}$ is an d-dimensional AWGN vector, then $h(\underline{Z}) = d\ \log(2\pi e\sigma^2)$ and hence

$$C_{iid} = \lim_{n\to\infty} \frac{1}{N} \sum_{i=1}^{N} \log(\lambda_i) - d\ \log(2\pi e\sigma^2)$$

.

C. Symbol and Bit Interleaved Non-iterative Demodulation Capacity

The capacity estimate derived in the section B corresponds to a system which uses a jointly optimal receiver. This is typically achieved in practice by a system which iterates between the outer code and the demodulator. Iterations between the demodulator and the outer decoder leads to higher computational complexity. To reduce the computational complexity we may decide to employ a suboptimal system in which the BCJR algorithm for the demodulator is invoked only once. The capacity estimate of such a system would help us in quantifying the loss in performance in using a suboptimal system. We define the capacity of this system as the non-iterative demodulation capacity - C_{ni}.

We note that although the output of the demodulator is correlated due to the BCJR algorithm, the presence of an ideal interleaver removes this correlation. The interleaving is done either at the symbol or bit level, leading to the two capacities - symbol interleaved non-iterative capacity (C_{ni}^{s}) and bit interleaved non-iterative

capacity C_{ni}^{b} respectively. The equivalent channel for the symbol interleaved system has inputs $\mathbf{X} = (X_1, X_2, \ldots, X_N)$ and outputs $\underline{\eta} = (\underline{\eta}_1, \underline{\eta}_2, \ldots, \underline{\eta}_N)$, where $\underline{\eta}$ is the vector of log likelihoods of the M-ary symbols.

The capacity of the equivalent channel is defined as

$$C_{ni}^{s} = \sup_{Pr(\mathbf{X_1^n}=\mathbf{x_1^n})=\mathbf{\Pi_{i=1}^{n}}\mathbf{Pr}(\mathbf{X_i}=\mathbf{x_i})} I(X;\underline{\eta}) \tag{4.2}$$

We will compute $I(X;\underline{\eta})$ as $I(X;\underline{\eta}) = H(X) - H(X \mid \underline{\eta})$. Since the channel is symmetric with respect to the input symbols, the input i.i.d., $Pr(\mathbf{X_i} = \mathbf{x_i})$, that maximizes the capacity is uniform distribution with

$$Pr(\mathbf{X_i} = \mathbf{x_i}) = \frac{\mathbf{1}}{\mathbf{M}} \; \forall \; \mathbf{i}$$

. Therefore $H(X)$ is equal to A where $M = 2^A$. $H(X \mid \underline{\eta})$ is computed as

$$\int_{\underline{\eta} \in \Re^M} f(\underline{\eta}) \; H(X \mid \underline{\eta})$$

, where we need to estimate $\Pr \underline{\eta}$ the distribution of $\underline{\eta}$. This can be done by observing the output of the demodulator $\underline{\eta}_i$ for $i = 1 \cdots N$. This is equivalent to defining C_{ni}^{s} of the system as

$$C_{ni}^{s} = \lim_{N \to \infty} \frac{1}{N} \sup_{Pr(\mathbf{X_1^N}=\mathbf{x_1^N})=\mathbf{\Pi_{i=1}^{N}}\mathbf{Pr}(\mathbf{X_i}=\mathbf{x_i})} \sum_{i=1}^{n} I(X_i;\underline{\eta}_i)$$

where we calculate the value of $H(X \mid \underline{\eta})$ at each value of $\underline{\eta}$ and form the expectation, instead of first estimating the p.d.f. of $\underline{\eta}$ and then take the expectation of $H(X_n;\underline{\eta}_n)$. Both these methods converge to $H(X \mid \underline{\eta})$ as $N \to \infty$. To compute $H(X \mid \underline{\eta})$ we note that $\underline{Y}_1^N$ is a sufficient statistic for $\underline{\eta}_i$ and hence $I(X_n;\underline{\eta}_n) = I(X_n;\underline{Y}_1^N)$ and $H(X_n \mid \underline{\eta}_n) = H(X_n \mid \underline{Y}_1^N)$. All we need to do now is estimate

$$\lim_{N \to \infty} \frac{1}{N} \sum_{n=1}^{N} H(X_n \mid \underline{Y}_1^N) \tag{4.3}$$

We know that the BCJR algorithm computes $Pr(X_i \mid \underline{Y}_1^N)$ and hence we can estimate (4.3) by computing

$$H(X_n \mid Y_1^N) = \sum_{m=0}^{M-1} -Pr(X_n = m \mid Y_1^N)\ \log_2(Pr(X_i = m \mid Y_1^N))$$

for each n and take the average as $N \to \infty$. Having computed $H(X)$ and $H(X \mid \underline{\eta})$ can form the estimate of C_{ni}^s as $H(X) - H(X \mid \underline{\eta})$.

The bit interleaved non-iterative capacity calculation assumes the presence of a bit level interleaver between the data sequence $\underline{X}_b$ and the modulator. C_{ni}^b can be calculated by applying the above definition to a system with output $[\eta_b^1, \eta_b^2, \ldots, \eta_b^A]$, which are LLRs of the bits $[X^1, X^2, \ldots, X^A]$ obtained from the symbol likelihood. C_{ni}^b is defined as

$$C_{ni}^b = \sup_{Pr(\mathbf{X_1^n}=\mathbf{x_1^n})=\mathbf{\Pi_{i=1}^n Pr(X_i=x_i)}} \sum_{k=1}^{A} I(X; \eta_b^k) \tag{4.4}$$

where $I(X; \eta_b^k)$ is computed as $H(X) - H(X \mid \eta_b^k)$ and

$$H(X \mid \eta_b^k) = E[(Pr(X^k = 0)\log(Pr(X^k = 0)) + Pr(X^k = 0)\log(Pr(X^k = 0)))]$$

D. Capacity Results

The capacity calculation for full response binary, 4-ary and 8-ary CPFSK systems, with the modulation index = 1/2, 1/4 and 1/8 respectively, was done using the methods described above. The capacity estimates are plotted in Figs. 8 to 11. These capacity plots can be used to compare the performance of a system with the theoretical limit. For example, in a system that uses 8-ary CPFSK with modulation index 1/8 and has binary decoder, we can see from the graph that the per bit Signal to Noise Ration(SNR) E_b/N_0 required for a rate equal to 2 bps/hz is around 3.2dB for the ideal demodulator. On the other hand a system that uses the non-iterative demodulation, which is sub-optimum, requires about 3.7dB which means that the loss due to the

sub-optimum detection is around 0.5dB.

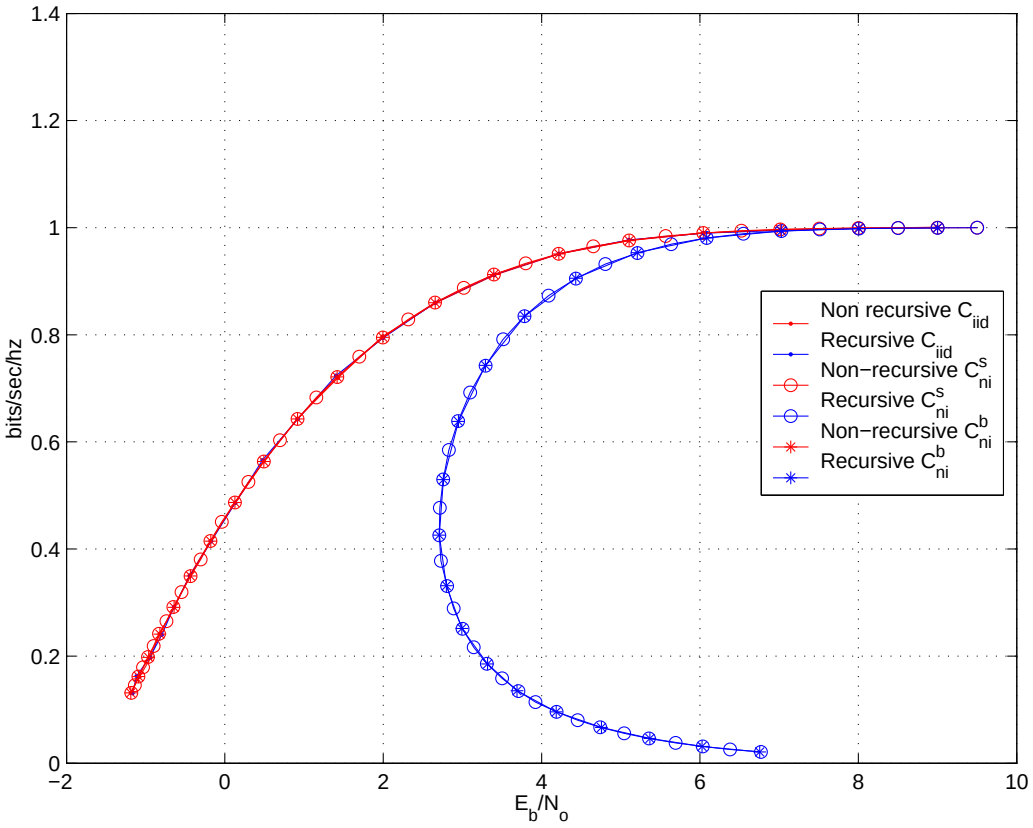

Fig. 8. Capacity estimate for binary CPFSK

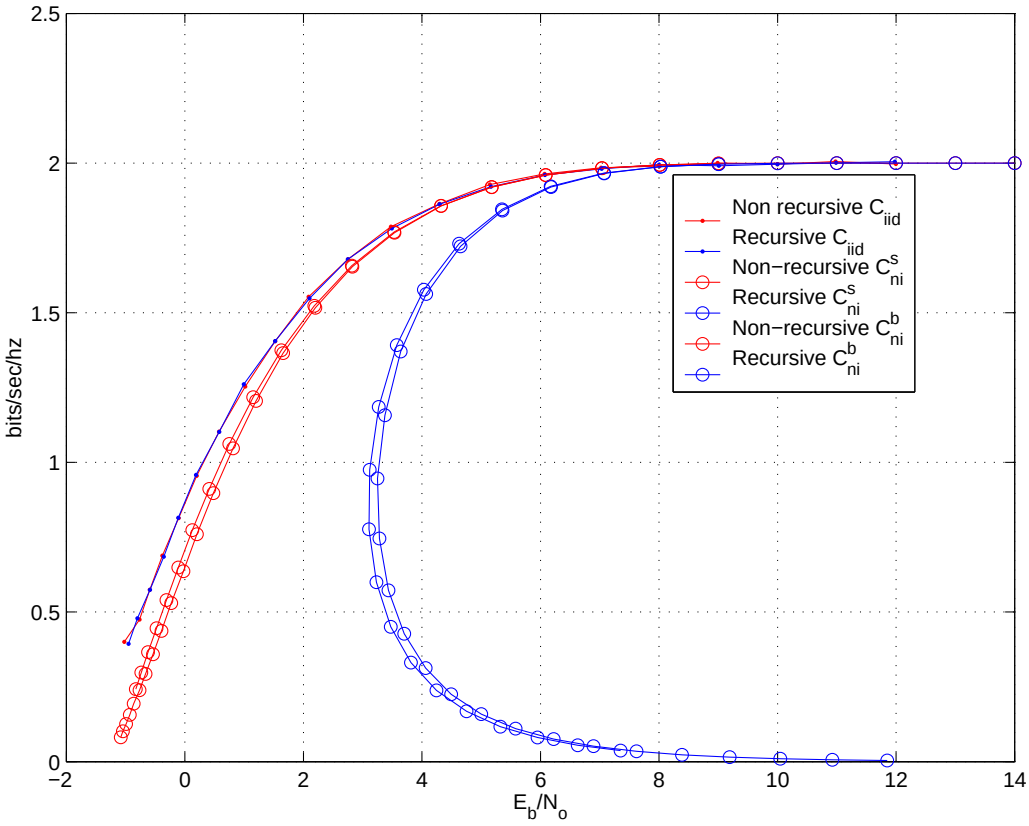

Fig. 9. Capacity estimate for 4-ary CPFSK

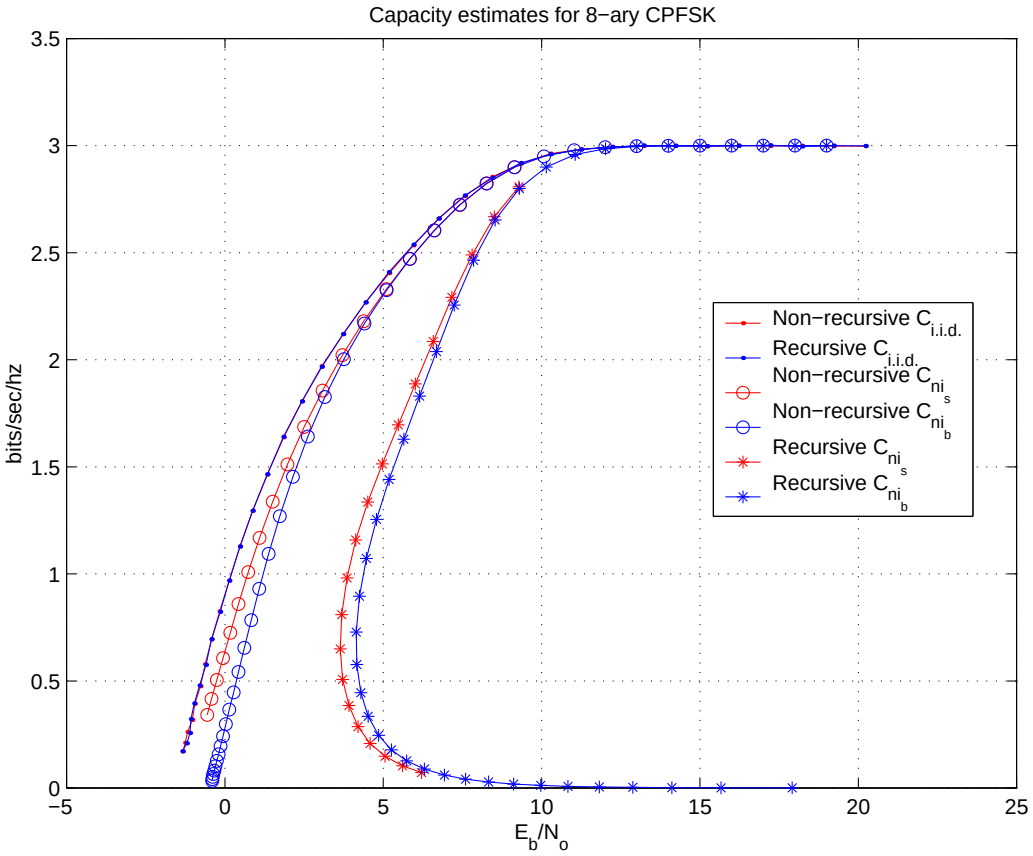

Fig. 10. Capacity estimate for 8-ary CPFSK

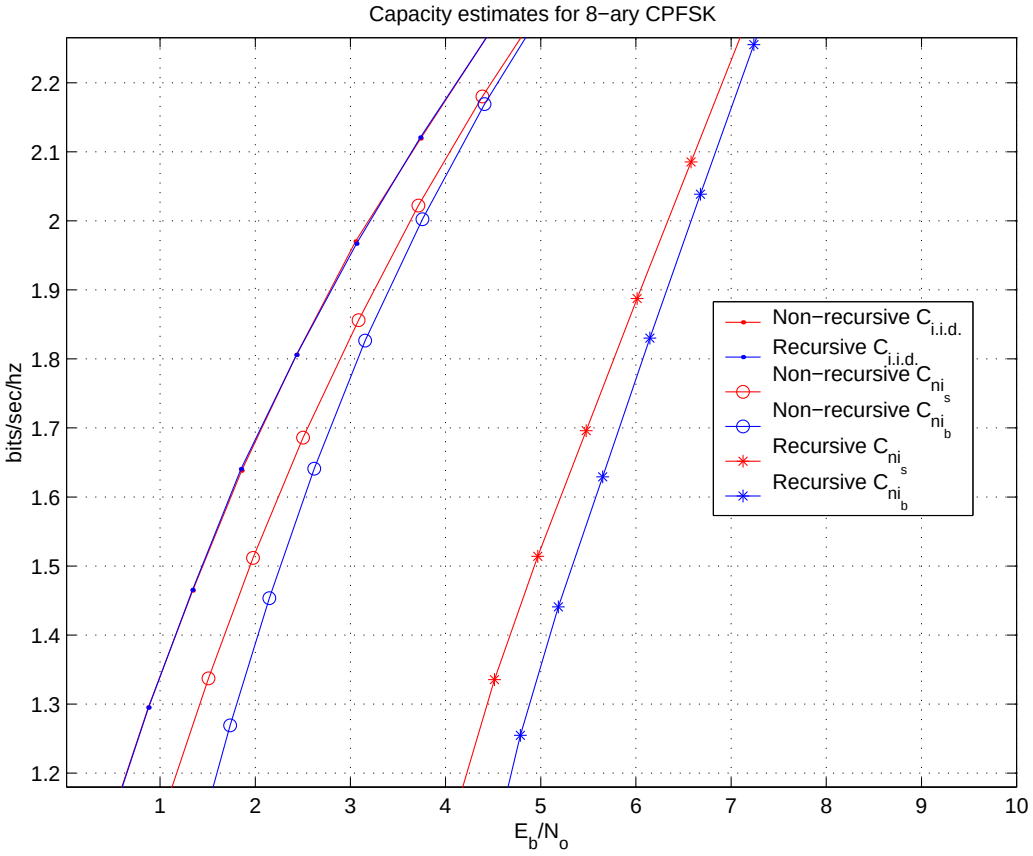

Fig. 11. Capacity estimate for 8-ary CPFSK (Zoomed)

CHAPTER V

CODE DESIGN

Having obtained the estimates for the capacity of CPM system we will now focus on designing channel codes which will approach the capacity limit. We will discuss two code design principles that were investigated. The first design principle is Matched Bit Interleaved Coded Modulation(MBICM) which can be either applied to a system which employs iterative or non-iterative demodulation. The second is Multilevel Coding (MLC) which can be applied to higher order modulation schemes and requires iteration between the channel code and the demodulator.

The idea behind conventional MBICM is to use a single component binary code which is designed so that the performance is optimum for the given channel. With MBICM the input bits are encoded using the code of appropriate rate. The output codeword is then interleaved at the bit level and mapped to M-ary symbols by grouping appropriate number of bits together. MBICM has been used to design binary ECC for channels like BPSK and binary CPFSK [20] where the output of the demodulator can be modelled easily. In case of higher order modulation, like 8-ary CPFSK, the properties of the bits to symbol mapper causes the output distribution of the different levels of the bits vary at different levels. For example, if we use the mapping given in Table I, the output bit Log Likelihood Ratio (LLR) of the three levels are distributed as shown in Fig. 12. This nature of the bit distributions (probability distribution function) needs to be incorporated when designing good codes. In this thesis we investigate different ways of representing the distributions and incorporating them in the code design procedure. Thus with the knowledge of a channel's output distribution we can design capacity approaching codes by following certain design principles.

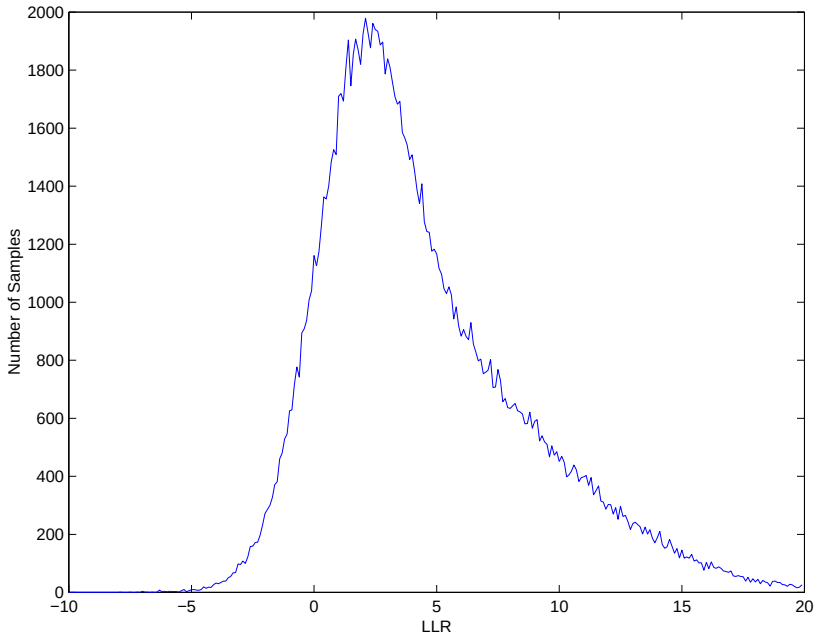

Fig. 12. Distribution of LLR output of the demodulator

Multilevel coding is one of the code design approach which jointly optimizes the coding and modulation to achieve significant coding gains. It is applied to higher order (typically 2^L) modulation schemes. In such a modulation scheme, each signal point is associated with a length L binary address. The input data is then split into L streams, and each stream is encoded by a component encoder with rate R_i for each $i = 1 \ldots L$. Thus the channel can be broken down into L component channels. It has been shown in [7] that the the transmission over the channel can be separated in to parallel transmission of L binary bits over equivalent channels at each level i provided that the bits at the lower levels $(0, 1, \ldots, i-1)$ are known. The capacity design rule requires the code rates R_i to be equal to the capacity of the equivalent channel at level i, which is equal to the capacity of the channel with the knowledge of the bits at levels $0, 1, \ldots, i-1$. So far there has been no way to estimate the equivalent capacity at the different levels for CPM systems. We will show how these capacities can be

estimated for these systems.

A. Matched Bit Interleaved Coded Modulation (MBICM)

Matched Bit Interleaved Coded Modulation involves designing a single binary encoder and decoder which is matched to the channel. The term matched refers to the fact that the design of the code incorporates the nature of the distribution of the channel outputs in its optimization procedure. Due to the availability of design algorithms for LDPC codes we will focus on adapting these algorithms for CPM concatenated system. Fig. 13 shows the block diagram for a MBICM system. The bits from the binary source are encoded by the binary encoder. The outputs are mapped to M-ary symbols and modulated and sent over the channel. The demodulator produces reliabilities on the symbols which are converted to bit reliabilities which are processed by the decoder. In non-iterative demodulation the demodulation is performed only once, where as in iterative demodulation the demodulation is repeated. If the decoder is an iterative decoder the demodulation is repeated after a fixed number of decoder iterations. The dotted line shows the feedback of the extrinsic information to the demodulator from the decoder.

1. MBICM code design with LDPC codes

LDPC codes are block codes which have a randomly generated Parity-Check Matrix (PCM) discovered by Gallager [14] , [21]. In this thesis we will design LDPC codes based on differential evolution algorithm used in [22]. The differential evolution arrives at an optimized degree profile by evaluating the performance of different profiles through density evolution developed by Richardson and Urbanke [23]. This algorithm iteratively computes the densities of the messages and can be used to compute the

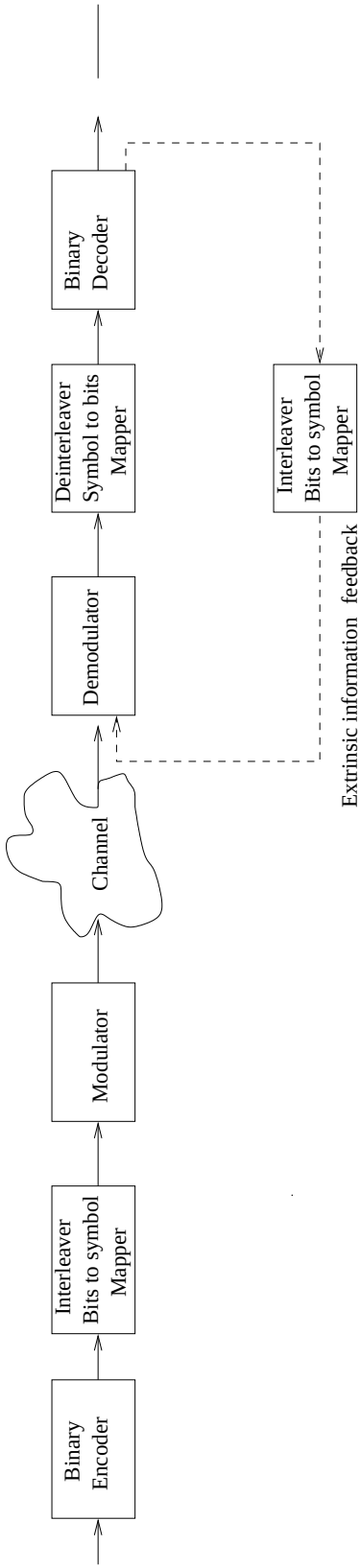

Fig. 13. MBICM system model

threshold. Differential evolution generates random degree profiles subject to certain constraints like rate , maximum left and right degrees. These random profiles performance can be analyzed using the density evolution algorithm. The best degree profiles are mutated to obtain other profiles, whose performance is then evaluated through the density evolution algorithm. This procedure is repeated a number of times and the profile with the lowest bit error rate is used for simulation.

2. Density evolution with Gaussian approximation

Density evolution based analysis of LDPC codes with Gaussian approximation can be used to compute the threshold of a given LDPC code ensemble. The Gaussian approximation for density evolution assumes that the messages (LLR) passed along the edges are Gaussian distributed with certain mean and whose variance is twice the mean. This approximation makes it easier to analyze the evolution of the density since the distribution of the messages is now described simply by the means of the mixture Gaussian. We will briefly explain how the density evolution can be applied to our system.

Consider an LDPC code with the edge degree distribution $\lambda(x)$ and $\rho(x)$ and maximum left and right node degrees D_l and D_r respectively. Let the random variables u and v represent the output messages from the check node and variable node respectively. Density evolution with Gaussian approximation tracks the means of messages passed from degree i variable nodes ($m^q_{u,i}$) and degree j check nodes ($m^q_{v,j}$) for each stage of iteration q. Since the operations performed at the variable and check nodes are known we can compute the mean of the output extrinsic information given the input mean. We approximate the channel output LLRs using an equivalent mixture of Gaussian distributions which will be discussed in section 3. Thus the

demodulator output can be expressed as

$$\sum_{i=1}^{N_g} f_i \, \mathcal{N}(\mu_i, 2\mu_i)$$

. The mean of the output messages of the degree i variable nodes are represented by $m^q_{v,i,k}$ $k = 1 \ldots N_g$, where k denotes the index of the Gaussian mixture component from which the node receives its input message from the channel. Then

$$m^q_{v,i,k} = \mu_k + (i-1)m^{q-1}_u \tag{5.1}$$

Thus v is distributed as $\sum_{k=1}^{N_g} \sum_{i=1} D_l \lambda_i \; f_i \; \mathcal{N}(\mu^q_{v,i,k}, 2\mu^q_{v,i,k})$. For v distributed as $\mathcal{N}(x, 2x)$ we define $\phi(x) = 1 - E(\tanh(v/2))$ hence

$$E(\tanh(\frac{u^q}{2})) = 1 - \sum_{k=1}^{N_g} \sum_{i=2}^{D_l} \lambda_i \; f_i \; \phi(m_{v,i,k}) \tag{5.2}$$

Since for a degree j node, $E(\tanh(\frac{u^q}{2})) = E(\tanh(\frac{v^q}{2}))^{j-1}$, we have for the q^{th} iteration

$$m^q_{u,j} = \phi^{-1}[1 - (1 - \sum_{k=1}^{N_g} \sum_{i=2}^{D_l} \lambda_i \; f_i \; \phi(m_{v,i,k}))^{j-1}] \tag{5.3}$$

Thus the expected value of $m^q_{u,j}$ is given by $m^q_u = \sum_{j=2}^{D_r} \rho_j \mu^q_{u,j}$. The density evolution equation can then be written as

$$m^q_u = \sum_{j=2}^{D_r} \rho_j \; \phi^{-1}[1 - (1 - \sum_{i=2}^{D_l} \sum_{k=1}^{N_g} \lambda_i \; f_k \; \phi(\mu_k + (i-1)m^{q-1}_u))^{j-1}] \tag{5.4}$$

After fixing the operating SNR, number of iterations (Q) and a value of mean (μ_{max}) above which the code is assumed to have converged, we can iterate the equation (5.4) up to Q times and if the value of the mean exceeds (μ_{max}) we can conclude that the given profile is a good profile.

The above algorithm can be used to check the convergence of the decoder when

there is no iteration between the decoder and the demodulator. In case of iterative demodulation, the decoder and the demodulator exchange extrinsic information between them. The performance of the iterative demodulator can be evaluated by modifying the above algorithm slightly. Instead of letting the decoder iterate on its own, the decoder is allowed to iterate N_l times. The extrinsic information from the decoder at the end of N_l iteration is passed to the demodulator and the state of the demodulator is stored in memory. The decoder works on the coded binary inputs to the demodulators and produces reliabilities on these bits. Since the demodulator works at the symbol level, the extrinsic information on the bits has to be converted to symbol likelihood through the inverse mapping function. These symbol likelihoods are then used as $a - priori$ probabilities of the symbols in the demodulator. The demodulation is repeated with the channel signal output and the symbol $a - priori$ information. A new set of equivalent means are then estimated from the new output and the density evolution process is continued from the saved state.

3. Representing modulator output distribution

Due to the higher order CPM modulation and the mapping strategy, the distribution of the LLR at the output of the demodulator is no longer Gaussian. The density evolution with Gaussian approximation on the other hand expects its input to be described in terms of Gaussian distributions. In this thesis we investigate three different means of representing the arbitrarily distributed output as a sum of Gaussian densities.

The output code word is a sequence of ones and zeros which is dependant on the input information bits and the generator matrix of the LDPC code. The demodulator produces reliabilities on the input bits as LLR , which is the logarithm of the ratio of p_0 and p_1 where p_i is the probability of the bit being equal to $i,\ i = 0, 1$. Normalized

LLR is the LLR multiplied by a_i where $a_0 = 1, a_1 = -1$. Since the LDPC code is a linear code the performance of the code can be estimated by assuming that the code word is a sequence of all zeros. This property helps us to avoid generating code words from random information bit sequence. The high computational complexity of the encoding operation can be avoided by assuming that the input bit sequence is always an all zero sequence, which due to the linearity is encoded to the all zeros code word. But since the CPM modulator-demodulator system is a non-linear system in our simulation we need to transmit random bits to correctly model system performance. This normalization procedure allows us to use random coding bits to be transmitted through the modulator and converts the reliabilities on these random bits to reliabilities if the all zeros sequence was transmitted.

We need to represent the distribution of this normalized LLR using a mixture of N_g normal distributions - $\sum_{i=1}^{N_g} \mathcal{N}(\mu_i, 2\mu_i)$. We have designed codes with two different types of approximations for an M-ary modulation scheme in which $M = 2^A$. In the first method we represent the distribution by A Gaussian distributions whose means are equal to the mean of the LLRs at the A bit levels. In the second method we match the distribution by finding f_i and μ_i such that such that $\sum_{i=1}^{N_g} f_i \, \mathcal{N}(\mu_i, 2\mu_i)$ matches the output distribution of the demodulator. In the third method we compute the capacity of the channel looking at L levels of the bit, and then approximate each level with a Gaussian channel with mean LLR μ_i which has the same capacity as that channel.

a. Matching with sample means

This is the simplest form of approximating the distribution of the messages passed from the demodulator. The output of the demodulator at each level is shown in Fig. 14. It is clear that the output distribution is not Gaussian distributed. In this method

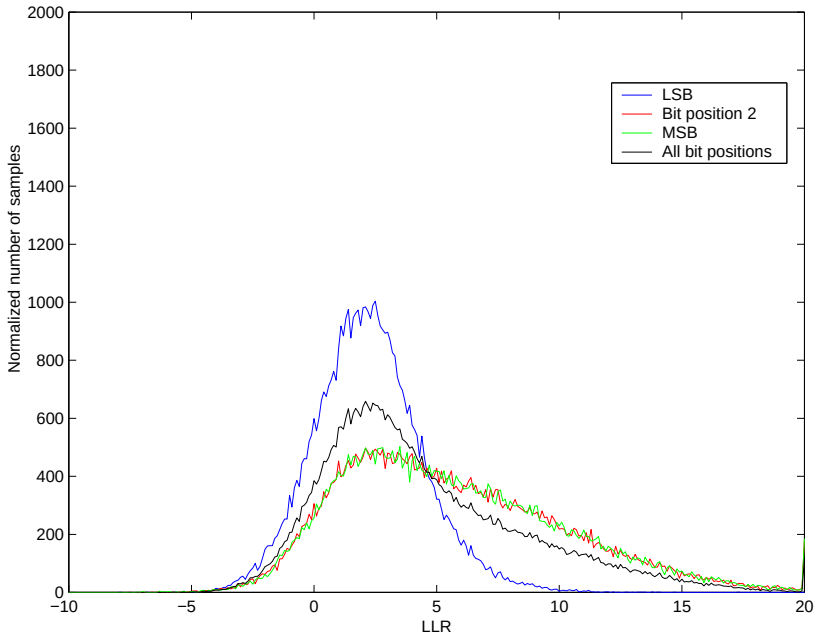

Fig. 14. Distribution of LLR output of demodulator at different bit positions

we choose one Gaussian distribution for each level of the M-ary modulator and the mean of the normalized LLRs is taken as the mean of the Gaussian distribution at each level. Thus in case of 8-ary CPFSK we will have three means μ_i where

$$\mu_i = \frac{\sum_{k=1}^{N} llr_k^i}{N}$$

and $f_i = 1/3, i = 1 \ldots 3$, where llr_k^i is the LLR for the k^th bit at the i^{th} level.

We will compare the matching of distributions using sample means and mutual information in section c. The figure on page 47 shows a sample distribution of the demodulator and the approximated Gaussian distribution and the matching through mutual information for the different levels of the 8-ary CPM system. As seen from the figure, the distribution represented by the sample means has smaller fraction of LLR values near the cross-over region (close to the x = 0 axis) than the actual distribution. The decoder performance is largely dictated by values which are less than zero. This

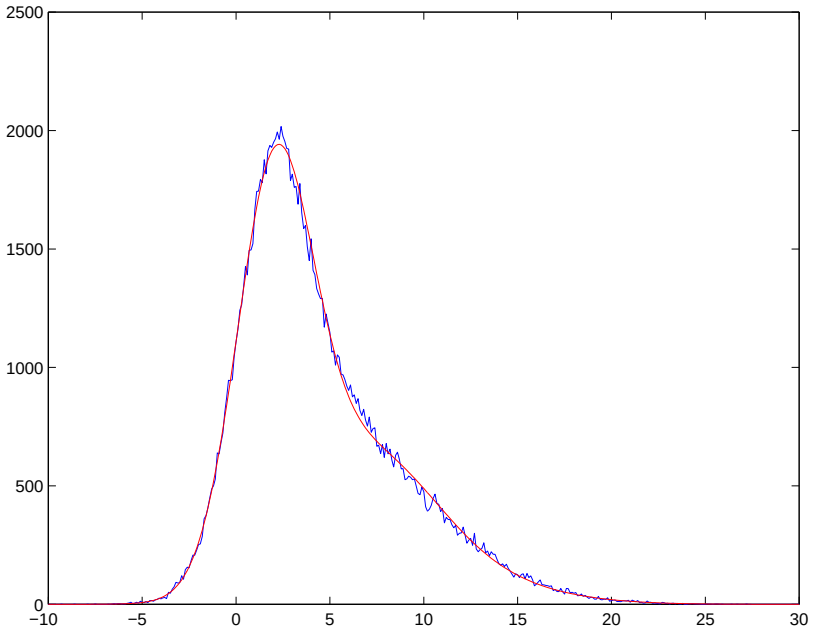

Fig. 15. Matching of distribution using mixture Gaussian distributions with 5 components

means that the estimated distribution will provide a slightly optimistic estimate of the threshold, when compared to the actual distribution.

b. Matching through mixture density

In this method we will approximate the composite output of the demodulator using a mixture of Gaussian densities. The LLRs at the output of the demodulator are no longer separated into L levels, and the combined distribution is processed by an Expectation Maximization (EM) algorithm [24] which produces N_g pairs of (f_i, μ_i) such that $\sum_{i=1}^{N_g} f_i \times N(\mu_i, 2\mu_i)$ matches the output of the demodulator. Where the number of Gaussian components in the distribution is equal to N_g. For our system a value of 5 was sufficient to closely match the distribution. Fig. 15 and 16 show the matching of the distribution using 5 and 20 components respectively. As seen from the figure, both mixture distributions match the output of the demodulator.

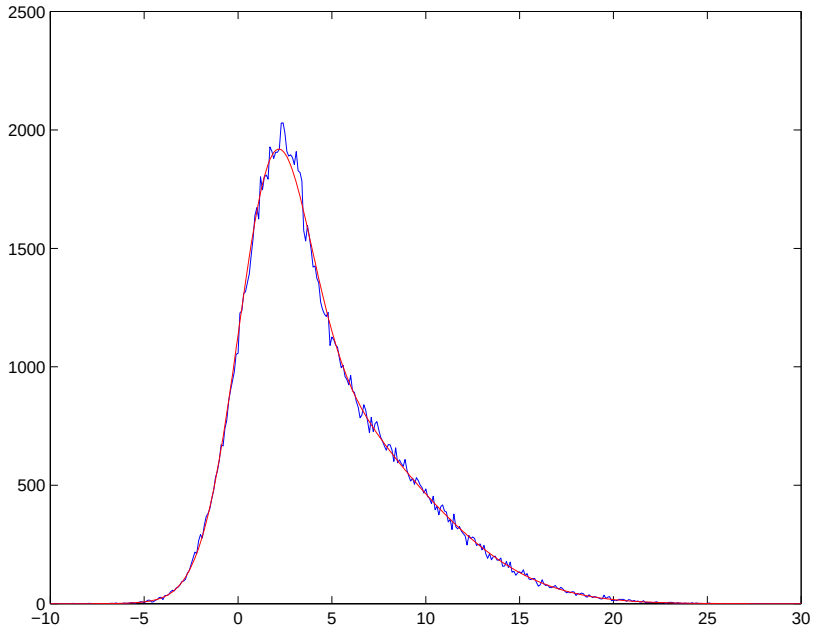

Fig. 16. Matching of distribution using mixture Gaussian distributions with 20 components

c. Matching through mutual information

The last method investigated is the matching of the distribution through mutual information. In this method we replace each level with an equivalent AWGN channel whose capacity is equal to the mutual information at that level. The idea behind this method is to replace the channel at each level with an equivalent channel instead of matching the actual distribution of the channel. We expect this method to be more accurate than the first method. Fig. 17 to 19 shows the distribution of the LLRs at the 3 levels for 8-ary CPFSK against the distribution of the equivalent channels and the matching through sample means. As seen from the figure, the distribution represented by the mutual information matches the distribution of LLR values near the cross over region (close to the x = 0 axis). We expect that the estimated distribution will provide a a more accurate estimate of the threshold, when

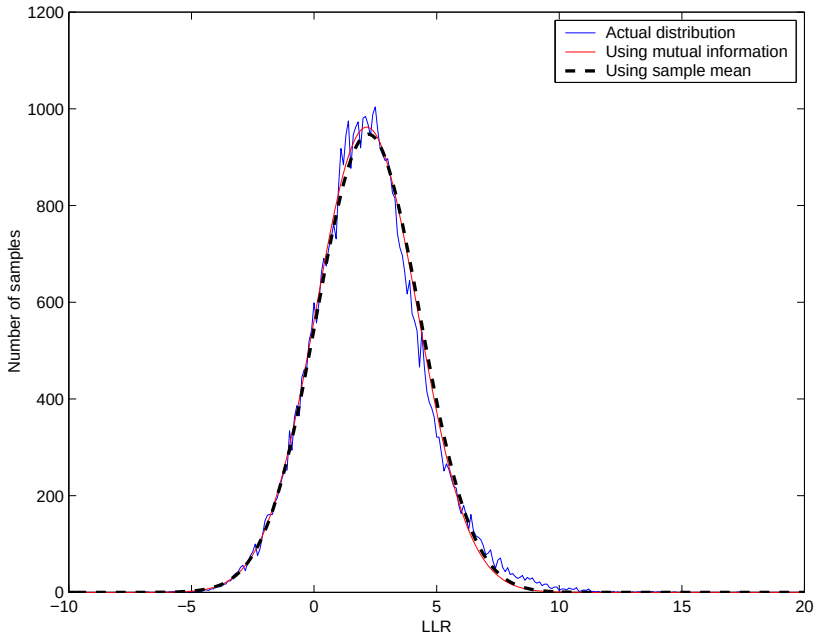

Fig. 17. Matching of distribution using sample mean and mutual information for Least Significant Bit (LSB)

compared to the distribution matched through the sample means.

4. Optimized degree profiles

We obtain some optimized degree profile for iterative and non-iterative demodulation using the three methods outlined in this section and the differential evolution algorithm. The means of the equivalent Gaussian distribution are passed to the differential evolution algorithm and the best profile is selected as discussed in section 2.

First we present the profiles obtained for non-iterative demodulation. The optimization was performed for 8-ary CPFSK with the mapping described in section B. For the optimization the maximum number of iterations was fixed at 300 and the maximum right degree was 15 and the maximum left degree was 25. For the simu-

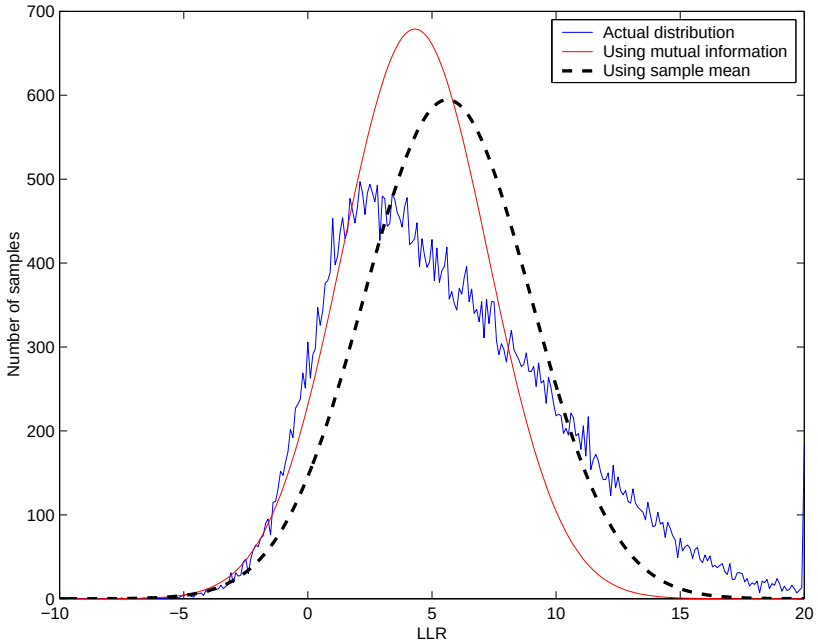

Fig. 18. Matching of distribution using sample mean and mutual information for bit position 2

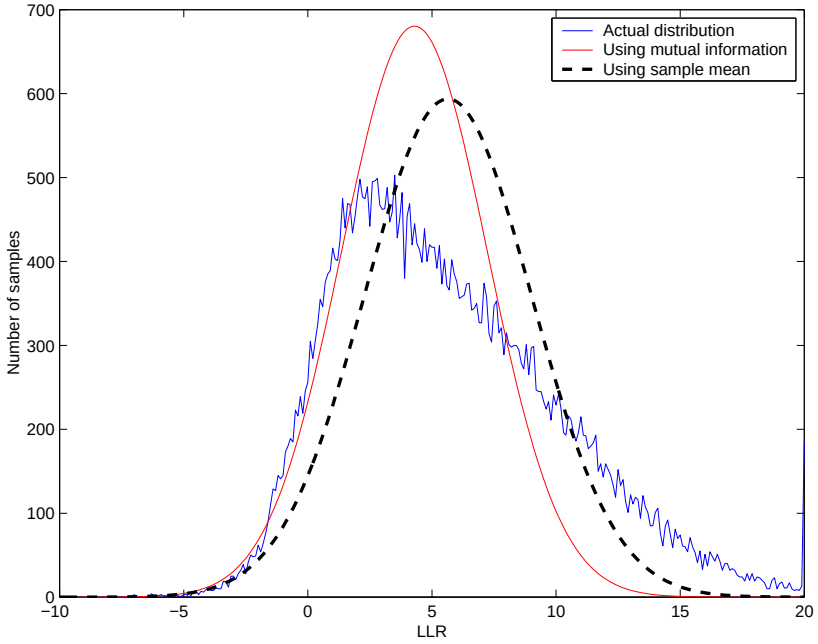

Fig. 19. Matching of distribution using sample mean and mutual information for Most Significant Bit (MSB)

lation the length of the LDPC code was fixed at 100000 bits, since the modulation scheme maps 3 bits to each symbol, the number of symbols transmitted was kept at 33334. The target code rate was 2/3. The minimum value of E_b/N_0 required for reliable transmission can be found to be 3.7dB from the capacity curve for non-iterative capacity plot for 8-ary CPFSK system as shown in Fig. 10.

Table III shows the degree profile obtained from the optimization program that uses the sample means of the LLR as the means of the equivalent Gaussian distribution. Fig. 20 shows the BER curve obtained through simulation. The threshold for this code was found to be 4.3dB by using density evolution algorithm for the code with the given profile.

Table IV shows the degree profile obtained from the optimization program that uses the matching of LLR distribution using mixture Gaussian densities. Fig. 21 shows the BER curve obtained through simulation. The threshold for this code was found to be 4.1dB by using density evolution algorithm for the code with the given profile.

Table V shows the degree profile obtained from the optimization program that matches the distribution of the LLR through mutual information of the LLR distributions at each bit position to equivalent Gaussian distributions. Fig. 22 shows the BER curve obtained through simulation. The threshold for this code was found to be 4.1dB by using density evolution algorithm for the code with the given profile.

For the optimization for iterative demodulation, the maximum number of iterations was fixed at 300 and the maximum right degree was 15 and the maximum left degree was 25, after every 10 LDPC iterations, the extrinsic means were passed to the demodulator and new equivalent means used for further density evolution. For the simulation the length of the LDPC code was fixed at 100000 bits, since the modulation scheme maps 3 bits to each symbol, the number of symbols transmitted was kept

Table III. Profile for non-iterative demodulation using matching sample means

x	λ_x	y	ρ_y
3	0.348424	14	0.338365
4	0.024834	15	0.661635
5	0.152702		
6	0.131878		
7	0.056686		
8	0.027028		
10	0.030159		
11	0.017783		
12	0.018564		
17	0.014683		
18	0.016285		
21	0.010515		
25	0.150450		

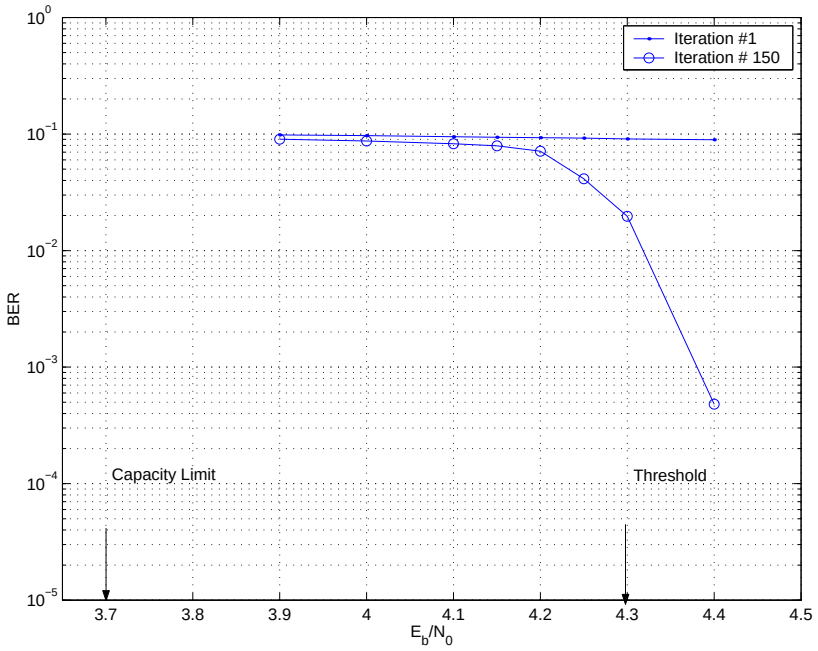

Fig. 20. Performance of non-iterative demodulation, code designed using matching through sample means, codeword length = 100000, maximum number of iterations = 150

Table IV. Profile for non-iterative demodulation using matching distribution using Gaussian mixture

x	λ_x	y	ρ_y
2	0.168565	14	0.523416
3	0.129567	15	0.476584
4	0.094297		
5	0.071966		
6	0.015794		
7	0.059866		
8	0.053016		
9	0.030805		
10	0.015902		
14	0.016801		
17	0.011078		
25	0.332341		

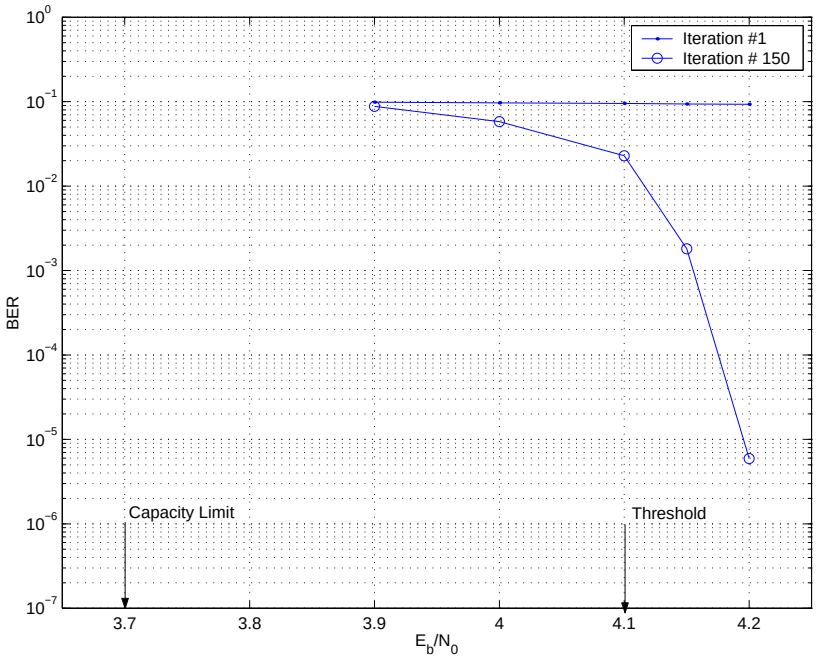

Fig. 21. Performance of non-iterative demodulation, code designed using matching through Gaussian mixture, codeword length = 100000, maximum number of iterations = 150

Table V. Profile for non-iterative demodulation, code designed using matching through mixture distribution

x	λ_x	y	ρ_y
2	0.175104	14	0.955328
3	0.159987	15	0.044672
4	0.085880		
5	0.032303		
6	0.028370		
7	0.058345		
8	0.025296		
9	0.023208		
10	0.019155		
11	0.045910		
12	0.021402		
13	0.013459		
14	0.015011		
25	0.296571		

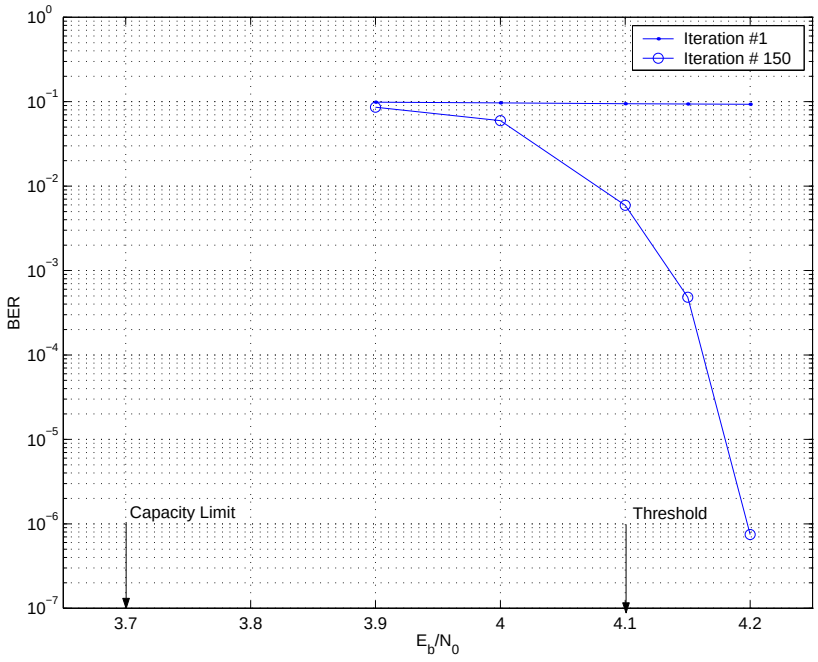

Fig. 22. Performance of non-iterative demodulation, code designed using matching through mutual information, codeword length = 100000, maximum number of iterations = 150

Table VI. Profile for iterative demodulation, code designed using matching through mutual information

x	λ_x	y	ρ_y
2	0.201292	14	0.752598
3	0.149273	15	0.247402
4	0.025528		
5	0.017135		
6	0.051927		
7	0.060836		
8	0.050813		
9	0.026362		
11	0.032148		
12	0.010875		
14	0.014388		
16	0.016431		
18	0.011393		
25	0.331600		

at 33334. The target code rate was 2/3. The minimum value of E_b/N_0 required for reliable transmission can be found to be 3.2dB from the capacity curve for iterative capacity plot for 8-ary CPFSK system as shown in Fig. 10.

Table VI shows the degree profile obtained from the optimization program. Fig. 23 shows the BER curve obtained through simulation. The threshold for this code was found to be 3.7dB by using density evolution with Gaussian approximation algorithm for the code with the given profile.

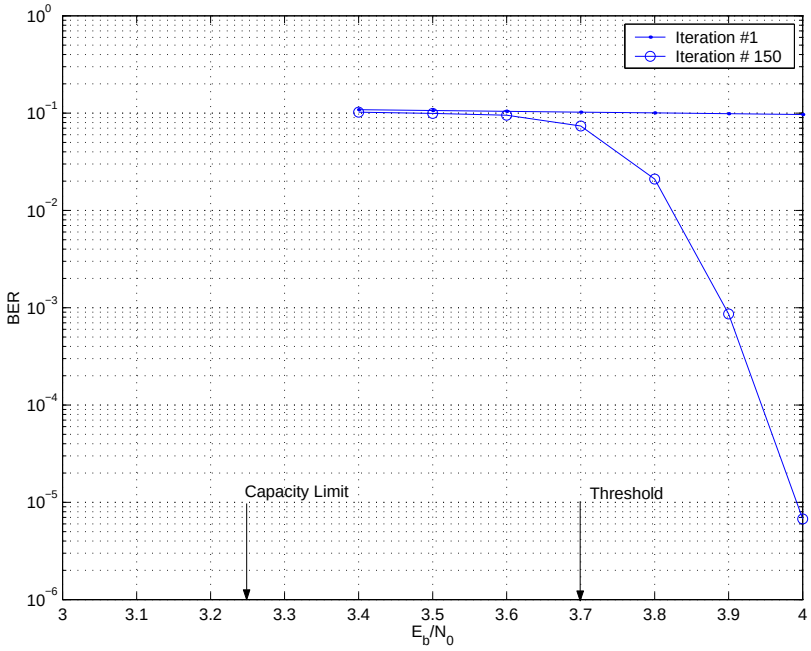

Fig. 23. Performance of iterative demodulation, code designed with matching through mutual information, codeword length = 100000, maximum number of iterations = 150, demodulation iterated every 10 LDPC iteration

B. Multilevel Coding

Multilevel coding (MLC) [25] is a code design principle which tries to jointly optimize the channel coding and the modulation so as to provide better performance. There are many ways to design MLC codes. We will discuss the MLC design for a 2^A-ary code using capacity design rule. This involves protecting the A bit positions independently using codes with carefully chosen rate for each bit position.

1. System model

Fig. 24 shows the block diagram for a MLC system. The decoding technique usually employed is Multi Stage Decoding (MSD) where each bit position j is decoded with decision fed from the $1, \ldots, j-1$ bit positions into the demodulator. In our analysis we have assumed that hard decisions are fed back to the demodulator. The demodulator effectively sees a better channel at higher levels assuming that the decisions fed in are right. It can be shown that the performance of a MLC with MSD is optimum if the rates of the codes at each level are chosen to be equal to the capacity of the equivalent channel at each level (C_j) [25]. Thus the design of a MLC then reduces to estimating the capacity of each level with feedback from the lower levels, and then designing codes with rates equal to the estimated capacity.

Since the demodulation of the CPM is done using a BCJR algorithm using the trellis as shown in Fig. 25, we can estimate the capacities C_j of each level by modifying the trellis using the the bits fed to the demodulator. This means that, since the MSD in MLC assumes that the information fed from the lower levels is correct, the transitions, in the trellis, for which the input bits are not equal to the feedback bits are deleted from the trellis as shown in Fig. 26. The capacity can then be estimated by following a procedure similar to the estimation of IID non-iterative capacity with

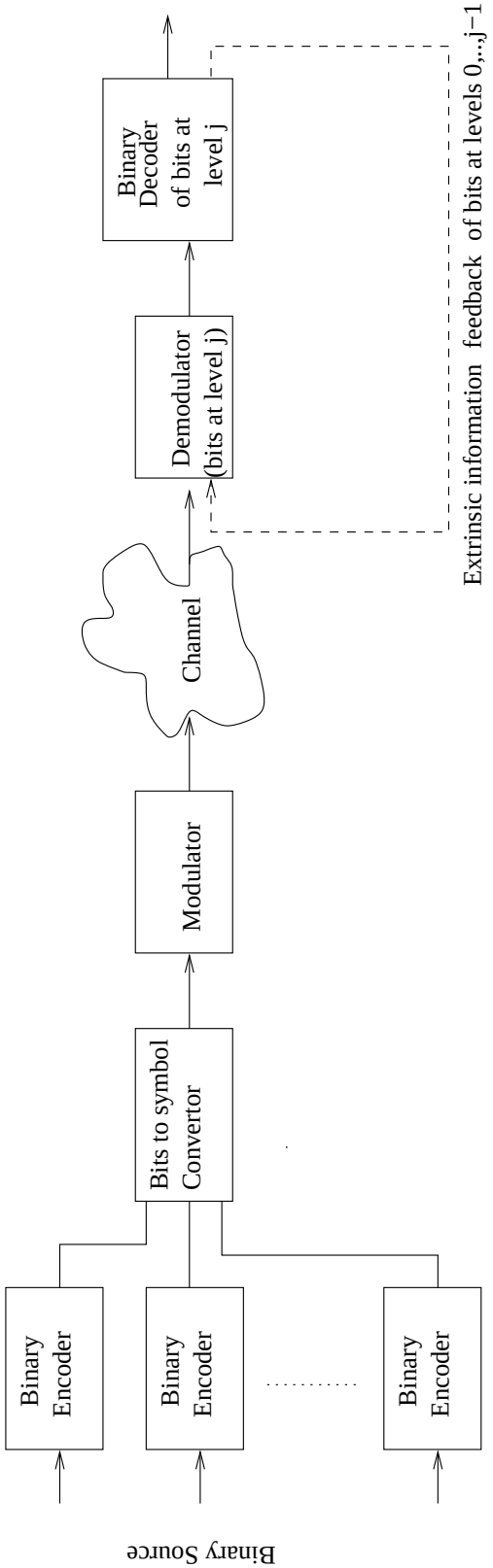

Fig. 24. Multi Level Coding (MLC) system

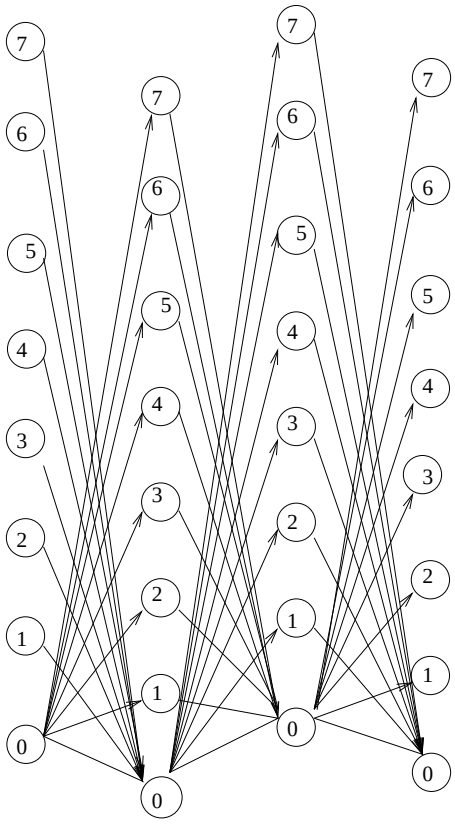

Fig. 25. Trellis for demodulation of full response 8-ary CPFSK

the modified trellis.

2. Equivalent capacity estimation

We will now formally derive the expressions required for the estimation of the capacity of the CPM system. Consider a $M(= 2^A)$-ary CPM system which will be concatenated with a MLC to obtain an overall rate of r bit/sec/hz. This capacity can be achieved with a signal to noise ratio (E_b/N_0) which can be found from the IID capacity curve for the constrained M-ary modulation. Fixing the operating SNR equal to this value, we need to estimate the rate of the A binary codes, one for each level. As mentioned earlier the rate of a code at any level is equal to the capacity of the equivalent channel

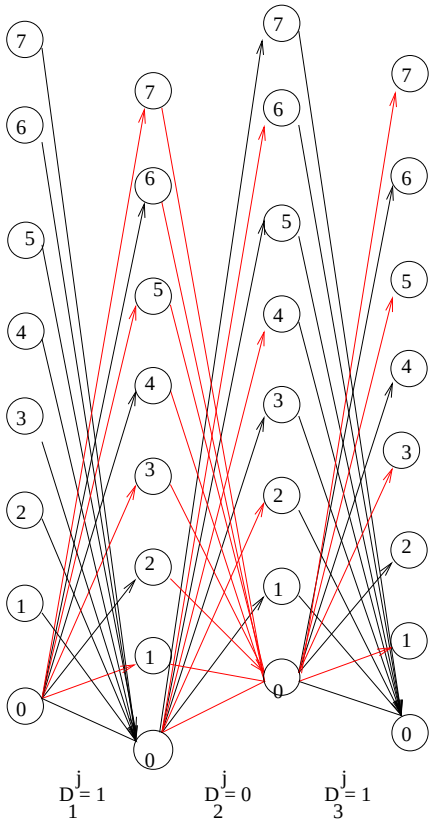

Fig. 26. Modified trellis for demodulation of full response 8-ary CPFSK to compute equivalent capacities

with information fed from the lower levels. Thus the rate of the code at each level l can then be found as

$$r_l = I(Y; X_{l-1} \mid X_0, X_1, \ldots, X_{l-2}) = h(y \mid X_0, X_1, \ldots, X_{l-2}) \\ -h(y \mid X_0, X_1, \ldots, X_{l-1}) \tag{5.5}$$

The information rates are calculated by using techniques described in section B. The estimation of $P(Y)$ now needs to be performed using the modified trellis. For example at $E_b/N_0 = 3.25$dB the rates for the three levels of 8-ary CPM evaluated to $0.4826, 0.7554$ and 0.7595 for the levels $1, 2$ and 3 respectively. These individual rates sum up to 1.9975 which is close to the i.i.d. capacity as expected.

CHAPTER VI

CONCLUSION

The knowledge of the information capacity rates supported by the system helps us to compare the system performance to the theoretical limits. Since the CPM system is a channel with memory, the computation of the capacity for these systems is not trivial. In this thesis we have applied some recently developed techniques in capacity estimation, of channels with memory, to the CPM system. The capacities computed in this thesis assume that the inputs to the CPM system are i.i.d. sequences. The first capacity estimate was for systems that are detected using optimum detectors. Since the optimum detectors for a CPM system concatenated with a outer error correction code has large complexity, we propose the use of iterative demodulation technique to approach this capacity. Another capacity that is of practical interest is the non-iterative capacity, which applies to systems that demodulate the signal only once.

The system modeled in this thesis consists of an outer binary ECC concatenated with a CPM modulator. Since the CPM systems with higher modulation order ($4-ary, 8-ary$) require its input to be a symbol of corresponding order, we require the use of a mapper that maps the bits to symbols. Although the performance of the optimum detector for CPM is independent of the mapping strategy, it is not the same case with the sub-optimum detector. We therefore compute two different non-iterative capacities, one assuming that the interleaving is done at the symbol level, the other assuming that the interleaving is performed at the bit level. Also, making use of the decomposition approach to CPM systems, we obtain two realizations of the CPM, one that has recursive nature and the other that has non-recursive nature. We then calculate the different capacities of the two different realizations of CPM.

The capacities of the system indicate the theoretical limits of performance for the system under consideration. Having computed the capacities we try to approach them by designing codes that are matched to the system. The two design principles explored in this thesis are Matched Bit Interleaved Coded Modulation (MBICM) and Multilevel Coding (MLC). For MBICM we concentrate on designing binary LDPC for iterative and non-iterative demodulation schemes. The design algorithm used in this thesis is based on the Differential Evolution algorithm. For MLC we illustrate how the rates for the different levels can be estimated by modifying the procedure to estimate the capacity of the system.

The MBICM technique uses a single binary encoder that has been designed to operate well for the given modulation scheme. The output of the encoder is mapped to symbols of appropriate modulation order and modulated. The resulting signal is sent over the channel and channel output is demodulated at the receiver. The symbol likelihood estimates are converted to bit Log Likelihood Ratios (LLR) and sent to the decoder. The design of the encoder requires the distribution of the LLR to be represented within the code design algorithm. The computational complexity of the design algorithm can be decreased dramatically, if we assume that the LLR is Gaussian distributed. In this thesis we considered three different ways of representing the LLR distribution. The output LLR of the CPFSK demodulator is not Gaussian distributed and hence we explore different ways of equivalently representing the LLR distribution in order to reduce the complexity of design process. The first method approximates the actual densities of the different bit levels to a Gaussian distribution whose mean is equal to the sample mean of the LLRs at the respective bit levels. The second procedure approximated the LLR distribution of all the bits taken together with a mixture Gaussian density. The last method approximated the distribution of the different LLR of different bit position with Gaussian distribution that had

the same mutual information as the distribution at a given level. By looking at the actual distributions and the approximated distributions we expected the codes designed though the first method to be more optimistic than those designed through the other methods. This is because the means estimated using the first method make the distribution of the LLR appear better than they actually are. This conclusion was confirmed by simulating the codes designed using the above-mentioned methods.

In case of MLC, we need to separate design encoders for the different bit positions in the modulation. Detection is performed through Multi Stage Decoding, where the signal is demodulated and the bits that belong to the lowest level are decoded. The decoded bits are then fed back to the demodulator and the demodulation is repeated with the assumption that the decoded bits do not have any errors. This is repeated till all levels are decoded. The different bit levels can be viewed as equivalent channels which have knowledge of the bits at lower levels. The capacity design rule requires the rate of the encoders at different levels be equal to the capacity of the equivalent channel at that level. In order to estimate the capacity at different levels, we modified the capacity estimation algorithm to make use of the feedback of the bit values at the lower levels. The equivalent capacities can then be used to design LDPC codes of appropriate rates and used as component encoders for a MLC system.

REFERENCES

[1] J.B.Anderson, T. Aulin and C.E. Sundberg, *Digital Phase Modulation*, New York: Plenum, 1986.

[2] D. Arnold and H.-A Loeliger, "On the information rate of binary-input channels with memory," in *Proc. IEEE ICC*, 2001, vol. 9, pp. 2692 –2695.

[3] A. Kavcic, X. Ma and M. Mitzenmacher, "Binary intersymbol interference channels: Gallager codes, density evolution and code performance bounds," in *ISIT*, Washington, DC, June 2001, pp. 24–29.

[4] H.D. Pfister, J.B. Soriaga and P.H. Siegel, "On the achievable information rates of finite state ISI channels," in *GLOBECOM*, 2001, vol. 5, pp. 2992–2996.

[5] L.R. Bhal, J. Cocke, F. Jelinek and J. Raviv, "Optimum decoding of linear codes for minimizing symbol error rate," *IEEE Trans. on Inform. Theory*, vol. 20, no. 284-287, Sept. 1974.

[6] S-Y. Chung, T. J. Richardson and R. L. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. Inform. Theory*, vol. 47, pp. 657–670, Feb. 2001.

[7] H. Imai and S. Hirakawa, "A new multilevel coding method using error correction codes," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 371–377, May 1977.

[8] V. Sharma and S. K. Singh, "Entropy and channel capacity in the regenerative setup with applications to markov channels," in *ISIT*, Washington, DC, June 2001, p. 283.

[9] F. Amoroso and J.A. Kivett, "Simplified MSK signaling technique," *IEEE Trans. Inform. Theory*, vol. COM-25, pp. 433–441, April 1977.

[10] F. Morales-Moreno and S. Pasupathy, "Convolutional codes and MSK modulation: A combined optimization," in *Proceeding of 12th Biennial Symposium Communications*, Queen's University, Kingston, ON, Canada, 1984, pp. c.1.4–c.1.7.

[11] B.E. Rimoldi, "A decomposition approach to CPM," *IEEE Trans. Inform. Theory*, vol. 34, pp. 260–270, March 1998.

[12] K.R. Narayanan and G.L. Stüber, "A serial concatenation approach to iterative demodulation and decoding," *IEEE Trans. Commun.*, vol. 47, pp. 956–961, July 1999.

[13] V.F. Szeto and S.Pasupathy, "Iterative decoding of serially concatenated convolutional codes and MSK," *IEEE Communications Letters*, vol. 3, pp. 272–274, Sept. 1999.

[14] R. G. Gallager, "Low density parity-check codes," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 21–28, Jan 1962.

[15] M. Sipser and D. Spielman, "Expander codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 1710–1722, Nov. 1996.

[16] D. J. C. MacKay, "Good error correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–431, Mar. 1999.

[17] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low-density parity-check codes," *Electron. Lett.*, vol. 32, pp. 1645–1646, Aug. 1996.

[18] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi and D. A. Spielman, "Analysis of low density codes and improved designs using irregular graphs," *Proceedings of the 30th annual ACM Symposium on Theory of Computing (STOC), 1998, pp. 249–258*

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

[20] R. Narayanaswami, "Coded modulation with low density parity check codes," M.S. thesis, Texas A&M University, 2001.

[21] R. G. Gallager, *Low Density Parity-Check Codes*, Cambridge, MA: MIT Press, 1963.

[22] T. J. Richardson, M. A. Shokrollahi and R. L. Urbanke, "Design of capacity approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 619–637, Feb. 2001.

[23] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599–618, Feb. 2001.

[24] K.R. Narayanan, X. Wang and G. Yue, "LDPC code design for turbo equalization," in *Information Theory Workshop*, 2000, vol. 3, pp. 57– 60.

[25] R. F. H. Fischer U. Waschmann and J. B. Huber, "Multilvel codes: Theoretical concepts and practical design rules," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1361–1391, July 1999.

APPENDIX A

BCJR ALGORITHM

Consider a general system with Θ^M states and input sequence to the encoder $\mathbf{X} = (X_1, X_2, \ldots, X_K)$, X_i belongs to a set with finite alphabets (M in case of M-ary CPM). Let the output of the modulator be modeled as a sequence of vectors denoted by $\underline{S}_1^N = \underline{\mathbf{S}} = (\underline{S_1}, \underline{S_2}, \ldots, \underline{S_N})$. The channel output sequence is denoted by $\underline{Y}_1^N = Y = (\underline{Y}_1, \underline{Y}_2, \ldots, \underline{Y}_N)$. The input symbol at discrete time t (X_t) causes the state of the system to change from Θ_{t-1} to Θ_t for which the encoder output is $\underline{S}_t$ and the channel output is $\underline{Y}_t$. The algorithm defines the following quantities: $\gamma_t(x, m', m) = \Pr(\Theta_t = m, \underline{Y}_t, X_t = x \mid \Theta_{t-1} = m')$, $\alpha_t(m) = \Pr(\Theta_t = m, \underline{Y}_1^t)$ and $\beta_t(m) = \Pr(\underline{Y}_{t+1}^N \mid \Theta_t = m)$.

We will first see how γ_t's can be computed

$$\gamma_t(x, m', m) = \Pr(\Theta_t = m, \underline{Y}_t, X_t = x \mid \Theta_{t-1} = m') \tag{A.1}$$

$$= \Pr(X_t = x \mid \underline{Y}_t, \Theta_t = m, \Theta_{t-1} = m') \ \Pr(\underline{Y}_t \mid \Theta_t = m, \Theta_{t-1} = m') \ \Pr(\Theta_t = m \mid \Theta_{t-1} = m') \tag{A.2}$$

$$= \Pr(X_t = x \mid \Theta_t = m, \Theta_{t-1} = m') \ \Pr(\underline{Y}_t \mid \Theta_t = m, \Theta_{t-1} = m') \ \Pr(\Theta_t = m \mid \Theta_{t-1} = m') \tag{A.3}$$

The first term is 1 or 0 depending on the input bit corresponding to the transition from state m' to m. The second term is the probability with which $\underline{Y}_t$ is received given that the transition from m' to m occurred. This is equivalent to the probability of receiving $\underline{Y}_t$ if $\underline{S}_t$ was transmitted on the channel, i.e. $\Pr(\underline{Y}_t \mid \underline{S}_t)$. The third term

is the *apriori* probability of the transition m' to m. The computation of γ mainly involves the computation of the second term. For an AWGN channel with variance σ^2 the second term is given by

$$\Pr(Y_t \mid X_t) = \Pr(Y_t \mid \underline{S}_t) = \frac{1}{(\sqrt{2\pi}\sigma)^n} \exp - \frac{\|\underline{Y}_t - \underline{S}_t\|^2}{2\sigma^2} \tag{A.4}$$

where $\underline{Y}_t$ and $\underline{S}_t$ are the received and transmitted signals in vector space notation and $\|.\|$ is the *norm* of the vector. Let $\gamma_t(m', m) = \sum_x \gamma_t(x, m', m)$.

The efficient computation of α and β is done by the following recursions (in normalized form) :

$$\alpha'_t(m) = \frac{\sum_{m'} \alpha'_{t-1}(m')\, \gamma_t(m', m)}{\sum_m \sum_{m'} \alpha'_{t-1}(m')\, \gamma_t(m', m)} \tag{A.5}$$

$$\beta'_t(m) = \frac{\sum_{m'} \beta'_{t+1}(m')\, \gamma_{t+1}(m, m')}{\sum_m \sum_{m'} \alpha'_t(m')\, \gamma_{t+1}(m', m)} \tag{A.6}$$

where $\sum_{m'}$ is summation over all $m' \in \Theta^M$. The values of $\alpha_0(m)$ and $\beta_N(m)$ are initialized depending on the initial state of the encoder and the termination of the trellis.

Once γ, α', β' have been computed the likelihoods for symbols can be computed by

$$L(X_t = x) = \log \left(\frac{\sum_{m,m'} \alpha'_{t-1}(m')\gamma_t(x, m', m)\beta'_t(m)}{\sum_x \sum_{m,m'} \alpha_{t-1}(m')\, \gamma_t(x, m', m)\, \beta_t(m)}\right) \tag{A.7}$$

VITA

Aravind Ganesan was born in Tamil Nadu, India. He received his Bachelor of Technology degree in Electrical Engineering from Indian Institute of Technology at Madras, India in May 2000. He has been a graduate research assistant in the Department of Electrical Engineering at Texas A&M University from September 2001 to April 2002. From May 2002 to December 2002 he was an intern at Nokia Research Center, Irving, Texas. His permanent address is 7 Prakasam Street, Apt 1-C, T. Nagar, Chennai, Tamil Nadu, India 600017. He can be also be reached through e-mail at aravind_g@yahoo.com.

The typist for this thesis was Aravind Ganesan.