One of the strongest motives that lead men to art and science is escape from everyday life with its painful crudity and hopeless dreariness, from the fetters of one's own ever-shifting desires. A finely tempered nature longs to escape from the personal life into the world of objective perception and thought.

– Albert Einstein.

University of Alberta

Reliable Communications under Limited Knowledge of the Channel

by

Raman Yazdani

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Communications

Department of Electrical and Computer Engineering

Fall 2012
Edmonton, Alberta

To my lovely parents...

Abstract

For successful correction of errors in a digital communication system, information about the channel, such as timing, noise power, fading gain, etc., should normally be available at the receiver. To this end, most receivers use channel parameter estimation and timing recovery modules. However, these modules are costly in terms of overhead, occupied chip area, power dissipation, and are usually imperfect. Thus, the need for high-speed communication systems motivates finding other solutions.

In this thesis, we seek efficient coding solutions for reliable communication under limited channel and timing information. We address the problem by considering three important scenarios:

First, we consider Gaussian channels with unknown noise power at the receiver. By using low-density parity-check (LDPC) codes, we propose a robust decoding method which provides better performance compared to the existing methods.

Second, we consider decoding on wireless fading channels where the fading gain and/or noise power are unknown at the receiver. Most modern error-correcting codes require soft metrics, usually log-likelihood ratios (LLRs), to be calculated at the receiver. This calculation is cumbersome on fading channels especially when the fading gain is unknown. Thus, we first propose an LLR accuracy measure, propose an efficient approximate LLR calculation technique, and then show that the performance under approximate LLRs is extremely close to that of exact LLRs.

Third, we seek practical coding for channels with imperfect timing at the receiver. In vast majority of the coding schemes invented, perfect synchronization is assumed between the transmitter and receiver. In most communication systems, however, achieving perfect synchronization is not possible. This leads to random symbol insertions and deletions in the received signal and poses a great challenge for error correction since conventional error-correcting codes fail at these situations. In this thesis, we propose a practical coding strategy which allows recovering insertions, deletions, and substitution errors without sacrificing the transmission resources.

Acknowledgements

First, I would like to take this opportunity to express my sincere and deeply grateful thanks to my supervisor Dr. Masoud Ardakani. I am certain that this work would not be possible without his valuable support, advice, kindness, and encouragement. He spent uncountable hours of discussion with me even about the smallest details of my research. I surely cannot imagine a supervisor more caring and supportive than him. It is immeasurable how much I have benefited from my supervisor during my years at the University of Alberta; from conducting research, technical writing, presentation skills, to learning from his genial personality.

I would also like to thank my dear friends in our research group. In particular, I would like to thank Mahdi Ramezani, Moslem Noori, and Kaveh Mahdaviani. The discussions we had about our research and their valuable friendship and support have truly helped me improve the quality of this research.

My sincere thanks goes to my great friends here in Edmonton who made my graduate years enjoyable. In particular, I would like to thank Ali Hendi, Navid Paydavosi, Pirooz Chubak, Ali Hooshiar, Iman Khosravi Fard, and Mohammad Behnam. I would also like to extend my deep thanks to Hossein Sadeghi, Marjan Assai, Reza Hashemi, and Mehdi Ale Mohammad, whom without their continued support, encouragement, and friendship especially during my hard times, I would not have been able to finish this work. I have been extremely lucky to have these incredible people around me.

I would also like to thank the Informatics Circle of Research Excellence (iCORE) and Natural Sciences and Engineering Research Council (NSERC) of Canada, and University of Alberta who provided financial support during my PhD studies.

Last, but not the least, my eternal gratefulness goes to my parents for their everlasting support in all aspects of my life even though I left them to study thousands of miles away. No words can express how thankful I am for their immense love, care, and support.

Table of Contents

List of Tables

List of Figures

List of Symbols

Symbol	Description	First use
$\underline{\boldsymbol{X}}$	Random sequence of channel inputs	6
$\underline{\boldsymbol{Y}}$	Random sequence of channel outputs	6
$\mathcal{X}$	Channel input alphabet	6
N	Length of sequences, also codeword length	6
$p(E), P(E)$	Probability of the event E	6
$z, \boldsymbol{z}$	The additive white Gaussian noise	6
$\mathcal{CN}(\mu, \sigma^2)$	Complex Gaussian distribution with mean μ and variance σ^2	6
σ	Standard deviation of additive white Gaussian noise in each dimension	6
C	Channel capacity	8
$I(\boldsymbol{X};\boldsymbol{Y})$	Mutual information between random variables $\boldsymbol{X}$ and $\boldsymbol{Y}$	8
R	Code rate	10
K	Message length of a code	10
Σ	Symbol alphabet of a code	11
$\mathbf{G}$	Generator matrix of a linear block code	13
$\mathbf{H}$	Parity-check matrix of a linear block code	13
v	A variable node	15
c	A check node	15
E	Number of edges in the graph	15

List of Abbreviations

Abbreviation	Description	First use
AM	Amplitude modulation	43
AM/PM	Amplitude modulation/phase modulation	66
AWGN	Additive white Gaussian noise	57
BER	Bit error rate	33
BIAWGNC	Binary-input additive white Gaussian noise channel	26
BICM	Bit-interleaved coded modulation	23
BPSK	Binary phase-shift keying	35
BSC	Binary symmetric channel	101
CSI	Channel state information	36
GMI	Generalized mutual information	42
HIPERLAN	High performance radio local area network	36
HMM	Hidden Markov model	58
I/D	Insertion/deletion	4
i.i.d.	Independent and identically distributed	59
LDPC	Low-density parity-check	2
LLR	Log-likelihood ratio	3
LSE	Least squared error	24
MBISO	Memoryless binary-input symmetric-output	41

ML	Maximum likelihood	20
MMSE	Minimum-mean squared error	24
MS	Min-sum	20
pdf	Probability density function	35
PEG	Progressive edge growth	70
PSK	Phase-shift keying	60
QAM	Quadrature amplitude modulation	43
RLL	Run-length limited	88
SP	Sum-product	18
TCM	Trellis coded modulation	58
WER	Word error rate	70

Chapter 1

Introduction and Motivation

In a modern society, exchange of information in an efficient, reliable, and secure manner is of fundamental importance. Any communication is affected by noise, interference, and other imperfections leading to communication errors. Physical solutions for error-reduction are expensive and inefficient. Error-correction codes, on the other hand, at a very small overhead cost, allow for detection and correction of transmission errors. Therefore, error-correction codes are crucial parts of any data communication system and the quality of these systems is totally dependent on efficient error-correction coding.

Usually, for decoding of information and correction of errors, the receiver should have knowledge of the communication channel parameters. For example, parameters such as noise power, fading gain, timing and synchronization information, etc., should be known at the receiver. Without such information, perfect decoding is usually not possible. Therefore, most communication systems are equipped with channel estimation and timing recovery modules at the receiver side. Error-correction codes are also designed based on the assumption that the receiver has perfect channel knowledge and is perfectly synchronized with the transmitter. Channel estimation and timing techniques, however, are themselves subject to imperfections. They also increase the complexity of the system and can incur a great cost in terms of implementation complexities, required chip area, power dissipation, data overhead, and decoding latency. In modern high-throughput communication systems, the receiver may not be able to handle this extra complexity or overhead. Therefore, other solutions are needed.

In this dissertation, we seek solutions for reliable communication under limited knowledge of the channel parameters at the receiver. Various scenarios can be

considered as different communication channels have different parameters. Also, the lack of knowledge of each of these parameters at the receiver poses a different challenge in the design of the system. In this dissertation, we consider three of the most important scenarios.

1.1 Proposed research problems

The three scenarios considered in this dissertation are as follows:

1.1.1 Robust decoding on Gaussian channels with unknown noise power

In the first scenario, we consider wired or wireless communication channels which are modeled by the additive white Gaussian noise channel [5]. This can vary from a simple twisted-pair wired link to block fading wireless links. The invention of modern powerful error-correcting codes such as turbo codes [6] and low-density parity-check (LDPC) codes [7, 8] has enabled reliable and efficient communication on these channels. For successful decoding, however, usually the noise power should be known at the receiver. Knowing that the noise power is usually time-varying as it can also reflect the interference, this information may not be available at the receiver at all times. Thus, using LDPC codes and assuming that the noise power is unknown at the receiver, we propose a robust hybrid decoding method which provides better performance compared to the existing decoding methods. This scenario is explained in more detail in Chapter 3[1].

1.1.2 Practical decoding on wireless channels with unknown fading gain

Wireless communications has been a fast-growing part of the communication industry in the past decades and has enabled high-speed information exchange between devices located virtually anywhere in the world. There exists an ever-growing list of the existent and potential applications of wireless communications. Nevertheless, due to the nature of the channel, many big technical challenges are still faced by the system designer. The fading effect, which captures the time-varying nature of the channel and is usually represented by the channel instantaneous fading gain, poses one of the biggest challenges for reliable communication on wireless channels [10].

[1] Results of this chapter have been published in IEEE Communication Letters [9].

If the instantaneous fading gain is known at the receiver, its effect can be compensated. Nevertheless, as we stated, channel estimation techniques should be used to estimate the fading gain which are not always feasible. As a result, in many situations the receiver should deal with the receiver signal without knowledge of the channel instantaneous fading gain. Also, the noise power may be unknown at the receiver.

We pick this scenario as scenario 2 in this thesis. In particular, we consider uncorrelated flat-fading channels which are modeled by an instantaneous fading gain and additive noise. We assume that the instantaneous fading gain and/or the additive noise power are unknown at the receiver. Modern error-correcting codes such as LDPC codes require some soft metrics to be calculated at the receiver [11]. These metrics are usually in terms of log-likelihood ratios (LLRs). The calculation of LLRs is cumbersome and complex on fading channels especially when the fading gain is unknown. Thus, approximate LLRs should be considered. To this end, by proposing an LLR accuracy measure, we present an efficient approximate LLR calculation method. We will show that the performance of the system under our approximate LLRs is extremely close to that of the complex true LLRs. This scenario is explained in more detail in Chapter 4[2].

1.1.3 Practical coding for channels with imperfect timing at the receiver

Since the seminal work of Shannon [15], there have been huge advancements in coding and information theory. The fundamental limits and efficient coding solutions approaching these limits are now known for many communication channels.

In the vast majority of the coding schemes invented, it is assumed that the receiver is perfectly synchronized with the transmitter, i.e., the symbol arrival times are known at the receiver. In most communication systems, however, achieving perfect synchronization is not possible even with the existence of timing recovery systems. This problem is further intensified in modern communication systems which have more stringent synchronization constraints. Asynchrony is also a great problem in digital magnetic recording systems.

When perfect synchronization does not exist, random symbol insertions and deletions occur in the received sequence and even the length of the resulting sym-

[2]Results of this chapter have been published in IEEE Transactions on Communications [12, 13] and presented in 2010 IEEE International Conference on Telecommunications [14].

bol sequence at the receiver may not be equal to that of the transmitted symbol sequence. This phenomenon poses a great challenge for error correction. Since the positions of the inserted/deleted symbols are unknown at the receiver, even a single uncorrected insertion/deletion can result in a catastrophic burst of errors. Thus, conventional error-correcting codes fail at these situations.

Channels which suffer from random insertions and deletions (synchronization errors) are called insertion/deletion (I/D) channels [1] and these channels are usually used to model systems suffering from synchronization errors. For proper communication over these channels, error-correcting codes capable of dealing with insertions and deletions, i.e., *synchronization codes* are needed. Synchronization codes have a long history but their design and analysis have proved to be extremely challenging, hence few practical results exist in the literature [16, 17].

In this scenario, we consider communication channels with limited timing information at the receiver, i.e., I/D channels. We propose a coding strategy which allows recovering insertions, deletions, and substitution errors without sacrificing the transmission resources. This strategy is explained in more detail in Chapter 5[3].

1.2 Organization of the thesis

The rest of this thesis is organized as follows. Required background information is reviewed in Chapter 2. In particular, since most of the results of this thesis are shown through using LDPC codes, we provide preliminaries on LDPC codes structure, analysis, and design methods. Chapters 3, 4, and 5 provide literature survey, formal definition of the problems, and research results pertaining to the three stated scenarios, respectively. Finally, the thesis is concluded in Chapter 6 where some future research directions are also stated.

[3]The results of this chapter are under second round of reviews for publication in IEEE Transactions on Communications [18].

Chapter 2

Preliminaries and Background

In this chapter, we review some background information necessary for the proposed research scenarios. In particular, we first review the structure of a generic digital communication system. We then provide a probabilistic definition of communication channels and discuss their fundamental limits. Next, we briefly review the basics of error-correcting codes. In particular, since most of our results are shown through using LDPC codes, we review their structure, analysis, and design techniques. A brief discussion on modulation and coding comes next and finally a brief review on channel estimation and timing recovery techniques is provided at the end.

2.1 Overview of a communication system

A digital communication system consists of three main parts: the transmitter, channel, and the receiver. Fig. 2.1 depicts the block diagram of a generic communication system from an information theoretic point of view. The transmitter generally consists of a source encoder which compresses the source signal, an error-correction or channel encoder which adds redundancy to the signal for error correction at the receiver, and a modulator which maps the encoded signal to appropriate symbols for the channel. The communication channel is the physical or logical path through which information is sent and it ranges from copper wire pairs and optical fibers to free air, network links, and even storage devices.

The goal at the receiver is to reconstruct the source signal based on the received signal from the channel. The demodulator provides appropriate information for the error-correction decoder by using the channel output and considering the mapping used in the modulator. The error-correction decoder then uses this information to decode the signal. Based on the channel output, information about the channel and

symbol arrival times are provided for the demodulator and decoder by the channel estimation and timing recovery modules. Finally, the source signal is recovered by the source decoder.

Our focus in this dissertation is mainly on error-correction coding, modulator, demodulator, and the channel.

2.2 Communication channels

In information theory, we are interested in a probabilistic modeling of the communication channel. To this end, the channel is viewed as a system with probabilistic relation between its random input and output. Formally, the channel model is characterized by its input alphabet $\mathcal{X}$, its output alphabet $\mathcal{Y}$, and a set of conditional probability assignments $p_{\underline{\boldsymbol{Y}}|\underline{\boldsymbol{X}}}(\cdot|\cdot)$ between the random input and output sequences $\underline{\boldsymbol{X}}$ and $\underline{\boldsymbol{Y}}$. This concept is depicted in Fig. 2.2. The random input and output sequences could contain real or complex symbols. The alphabets could also be continuous or discrete.

In general, the channel output sequence could depend on multiple input symbols, giving rise to a channel with *memory*. However, if each output symbol depends only on the input symbol at the same time instant then the channel is called *memoryless*. On a memoryless channel we have:

$$p_{\underline{\boldsymbol{Y}}|\underline{\boldsymbol{X}}}(\underline{\boldsymbol{y}}|\underline{\boldsymbol{x}}) = \prod_{i=1}^{N} p_{\boldsymbol{Y}_i|\boldsymbol{X}_i}(\boldsymbol{y}_i|\boldsymbol{x}_i), \tag{2.1}$$

where $\underline{\boldsymbol{X}}$ and $\underline{\boldsymbol{Y}}$ are sequences of length N, corresponding to N channel uses. In this thesis, we deal with discrete-input continuous-output channels both memoryless and with memory.

An example of a memoryless discrete-input continuous-output channel is the additive white Gaussian noise channel which is of great theoretical and practical importance in communications [5]. It is defined as

$$\boldsymbol{y}_i = \boldsymbol{x}_i + \boldsymbol{z}_i, \tag{2.2}$$

where $\boldsymbol{x}_i$ comes from an M-ary discrete alphabet $\mathcal{X}$, $\boldsymbol{z}_i \sim \mathcal{CN}(0, 2\sigma^2)$ is the circularly symmetric additive Gaussian noise with variance σ^2 per dimension (real and complex), and $\boldsymbol{y}_i$ is the continuous output symbol at time instant i. In this case channel conditional probabilities are defined as $p_{\boldsymbol{Y}_i|\boldsymbol{X}_i}(\boldsymbol{y}_i|\boldsymbol{x}_i) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{|\boldsymbol{y}_i - \boldsymbol{x}_i|^2}{2\sigma^2}\right)$. The subscript i is usually dropped for convenience.

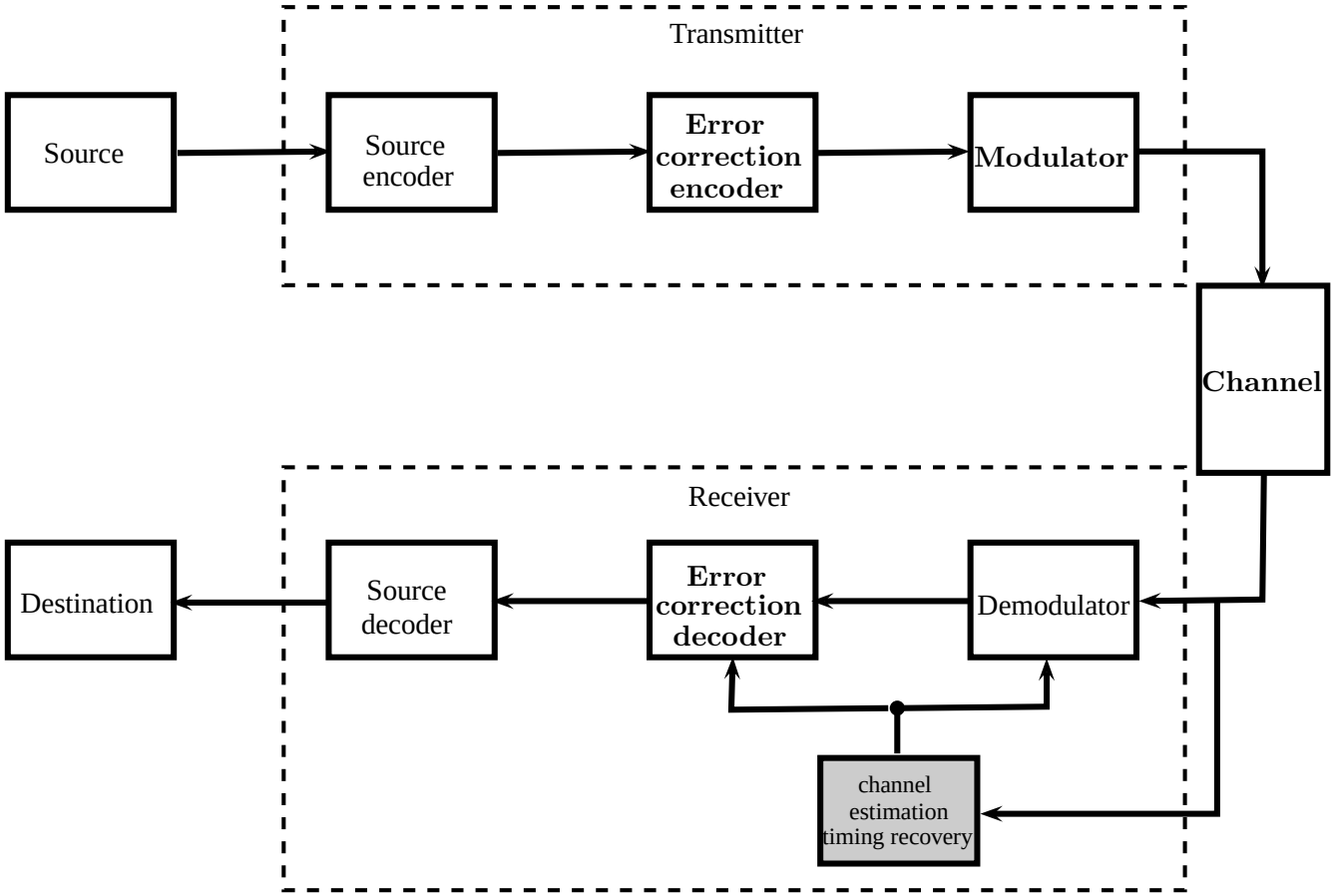


Figure 2.1: The block diagram of a generic communication system from an information theoretic point of view.

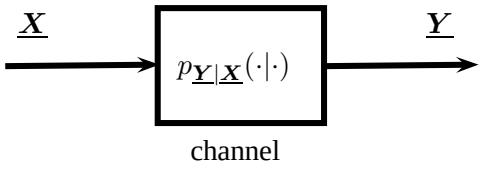


Figure 2.2: Probabilistic definition of a communication channel. Symbols in the input sequence $\underline{\boldsymbol{X}}$ are drawn from $\mathcal{X}$, symbols in the output sequence $\underline{\boldsymbol{Y}}$ from $\mathcal{Y}$, and $p_{\underline{\boldsymbol{Y}}|\underline{\boldsymbol{X}}}(\cdot|\cdot)$ denotes the conditional probabilities assigned between them.

In this thesis, apart from the additive white Gaussian noise channel described above, we also consider flat-fading and insertion/deletion channels. The definition and details are given in Chapters 4 and 5.

2.2.1 Channel capacity

Assuming the probabilistic model of the communication channel, in a seminal work [15], Shannon showed that there exists a fundamental maximum limit, called *capacity*, on the amount of information that can be reliably transmitted over any discrete memoryless channel.

Definition 2.1. The capacity of a discrete memoryless channel is defined as [15]

$$C = \sup_{p(\boldsymbol{X})} I(\boldsymbol{X};\boldsymbol{Y}), \tag{2.3}$$

where $I(\boldsymbol{X};\boldsymbol{Y})$ denotes the mutual information between the random variables $\boldsymbol{X}$ and $\boldsymbol{Y}$ and the maximization is done over the marginal distribution of $\boldsymbol{X}$.

The mutual information is a non-negative number measuring the mutual dependency of two random variables and is formally defined as

$$I(\boldsymbol{X};\boldsymbol{Y}) = \sum_{\boldsymbol{x}\in\mathcal{X}}\sum_{\boldsymbol{y}\in\mathcal{Y}} p(\boldsymbol{x},\boldsymbol{y}) \log_2\left(\frac{p(\boldsymbol{x},\boldsymbol{y})}{p(\boldsymbol{x})p(\boldsymbol{y})}\right), \tag{2.4}$$

where $p(\boldsymbol{x},\boldsymbol{y})$ is the joint probability distribution function of $\boldsymbol{x}$ and $\boldsymbol{y}$. The mutual information is zero if and only if $\boldsymbol{X}$ and $\boldsymbol{Y}$ are statistically independent.

The importance of channel capacity is due to Shannon's *noisy-channel coding theorem* [15]. This theorem states that for any $\epsilon > 0$ and any amount of information transmission rate $R_{\mathrm{t}} < C$ measured in bits per channel use, there exists an error-correcting encoding and decoding scheme which ensures that the probability of error is less than ϵ for sufficiently large code. Also if $R_{\mathrm{t}} > C$, probability of error is larger than a certain number no matter what encoding and decoding is used.

In other words, he showed that for a given channel and any amount of noise contamination on the channel, the communication error probability can be made arbitrary close to zero if the information transmission rate is below the capacity. Furthermore, he showed that reliable communication is not possible for rates beyond the capacity.

The noisy-channel coding theorem has been later generalized for other channels such as continuous, time-varying, and channels with memory [19, 20]. For information stable channels [19] the capacity is defined as

$$C = \lim_{N\to\infty} \frac{1}{N} \sup_{p(\underline{\boldsymbol{X}})} I(\underline{\boldsymbol{X}}; \underline{\boldsymbol{Y}}), \tag{2.5}$$

where $I(\underline{\boldsymbol{X}}; \underline{\boldsymbol{Y}})$ denotes the mutual information between the random sequences $\underline{\boldsymbol{X}}$ and $\underline{\boldsymbol{Y}}$ and the maximization is done over the distribution of $\underline{\boldsymbol{X}}$. The variable N denotes the number of channel uses and is also equal to the length of $\underline{\boldsymbol{X}}$. On most channels, the length of the output sequence $\underline{\boldsymbol{Y}}$ is also equal to N. However on insertion/deletion channels, as we will see later in Chapter 5, this is normally not true.

Although the definition of capacity exists for many channels, finding the channel capacity (evaluating (2.5)), is still an open problem for many channel models such as insertion/deletion channels [16].

We finish this section with another useful definition:

Definition 2.2 (Achievable transmission rate)**.** When $I(\underline{\boldsymbol{X}}; \underline{\boldsymbol{Y}})$ is evaluated under a certain input probability distribution $p(\underline{\boldsymbol{X}})$ or under a certain encoding and decoding rule for which the converse part of the noisy-channel coding theorem cannot be proved, the calculated rate is called an *achievable transmission rate*.

The channel capacity is the supremum of all achievable rates on the channel.

2.3 Error-correction coding

Shannon's theorem proves the existence of optimal error-correcting codes but does not show how such codes can be constructed which allow for practical encoding and decoding. The theory of error-correction coding deals with finding practical codes that can approach channel capacity.

The main idea in error-correcting coding is to provide protection against channel noise and other sources of errors by adding redundancy to the transmitted infor-

mation sequence. In an error-correcting code, the redundant symbols are computed based on the information symbols using a pre-determined algorithm which both the encoder and decoder know. A redundant symbol is usually formed from many information symbols and information symbols themselves may or may not appear in the final transmitted sequence. At the receiver, the decoder which knows the pre-determined algorithm decodes the information symbols from the contaminated received sequence without any retransmission.

The most basic form of an error-correcting code is a repetition code. In a binary $(N, 1)$-repetition code, each information bit is repeated N times and sent on the channel. For example, '0' is sent as '$\underbrace{0 \ldots 0}_{N}$' and '1' as '$\underbrace{1 \ldots 1}_{N}$'. The decoder can then do a majority vote and correct the errors. Though very simple, the repetition code is normally not a very efficient code as n channel uses are sacrificed for transmitting only one bit.

Definition 2.3 (Code rate)**.** The code rate R is a dimension-less quantity which measures the efficiency of a code (from the point of view of information transmission) and is defined as $R = K/N$ when each K information symbols are encoded into N symbols.

For example, for the binary $(N, 1)$-repetition code, $R = 1/N$. In other words, in each channel use, $1/N$ information bit is transmitted. Formally, error-correction coding deals with finding practical codes which can reliably work on a given channel with rates as close as possible to capacity.

Coding theory has a long history. The first class of efficient error-correcting codes were invented by Hamming in 1950 [21]. Hamming codes, though simple, provide achievable transmission rates very far from channel capacity [22]. Thus, they are not desirable for many applications. Since Hamming codes, the coding theory has been dealing with finding optimal codes that can approach capacity.

Block codes and convolutional codes

In general, error-correcting codes are classified into two categories: *block* codes and *convolutional* codes. Generally, convolutional codes are distinguished from block codes from the presence of *memory* in their encoding process.

Block codes are a very rich family of error-correcting codes which work with data in blocks. In particular, a block code of length N, with rate $R = K/N$,

Messages	Codewords
(0 0 0 0)	(0 0 0 0 0 0 0)
(0 0 0 1)	(1 0 1 0 0 0 1)
(0 0 1 0)	(1 1 1 0 0 1 0)
(0 0 1 1)	(0 1 0 0 0 1 1)
(0 1 0 0)	(0 1 1 0 1 0 0)
(0 1 0 1)	(1 1 0 0 1 0 1)
(0 1 1 0)	(1 0 0 0 1 1 0)
(0 1 1 1)	(0 0 1 0 1 1 1)
(1 0 0 0)	(1 1 0 1 0 0 0)
(1 0 0 1)	(0 1 1 1 0 0 1)
(1 0 1 0)	(0 0 1 1 0 1 0)
(1 0 1 1)	(1 0 0 1 0 1 1)
(1 1 0 0)	(1 0 1 1 1 0 0)
(1 1 0 1)	(0 0 0 1 1 0 1)
(1 1 1 0)	(0 1 0 1 1 1 0)
(1 1 1 1)	(1 1 1 1 1 1 1)

Table 2.1: The message sequences and their corresponding codewords for a block code with $K = 4$, $N = 7$, and $R = 4/7$.

and alphabet Σ, encodes a block of information symbols of length K (called a *message*) into a block of symbols of length N (called a *codeword*) where all symbols come from the alphabet Σ. The number of redundant symbols are then given by $N - K$. As a result, there are $|\Sigma|^K$ number of distinct messages which are mapped to $|\Sigma|^K$ distinct codewords, where $|\Sigma|$ denotes the size of the alphabet Σ. A block code is the collection of all the messages and their corresponding codewords. Since each message is encoded independently and is mapped to a distinct codeword, the encoding process is memoryless and can be implemented by a combinatorial logic circuit. When $|\Sigma| = 2$, the code is called a binary block code. An example of a binary block code with $K = 4$ and $N = 7$ is given in Table 2.1.

Given a noisy channel and a fixed code rate R, the main challenge in designing block codes is how to choose K and N and an appropriate mapping from the messages to the codewords which provide reliable error-correction performance on the channel. Increasing K and N usually improves the decoding performance but increases the encoding and decoding complexities. The literature pertaining to block codes is very rich and there exist various codes designed for many different channels. Some of the most famous block codes are Repetition, Hamming, Golay, Reed-Solomon, Hadamard, and LDPC codes [22].

As opposed to block codes, convolutional encoders work with serial data. The information sequence is serially fed into the encoder and the encoded sequence is produced. The encoder can still be seen as producing N-tuple coded symbols for each K-tuple information symbols. However, the N-tuple coded symbols not only depend on the corresponding K-tuple information symbols at the same time instance but also on m previous K-tuple information symbols. This means that the encoder has a memory order of m. This is the main difference between a convolutional code and a block code. Convolutional codes can be implemented using sequential logic circuits. Unlike block codes, the error-correction performance of convolutional codes can be improved by increasing m rather than increasing K and N.

Modern codes

Until early 1990's, no designed coding scheme was able to approach the channel capacity without a prohibitive encoding or decoding complexity. The state of the art was to concatenate a convolutional code with a block code (usually a Reed-Solomon code) which was still far from the channel capacity. In light of the discovery of belief propagation decoding which enabled low-complexity decoding for very large codes, modern error-correcting codes emerged. Two important classes of modern codes are turbo codes [6] and LDPC codes [7, 23]. The discovery of these codes enabled approaching the capacity on many channels with practical encoding and decoding complexity.

LDPC codes, due to their reliance on belief propagation decoding, are typically presented on a graph. They also posses desirable properties such as having a flexible structure and efficient analysis and design techniques and they outperform turbo codes when the block length is large. In this dissertation, most of our results are presented through using LDPC codes although some of our approaches are general. Thus, we will review LDPC codes in the next section.

2.4 Low-density parity-check codes

Here, we review the necessary background on LDPC codes. Since they are a subclass of linear block codes, we first briefly review linear block codes.

2.4.1 Linear block codes

Error-correcting codes are also classified into *linear* and *non-linear* codes. The error-correcting code is called linear if and only if any linear combination of the code's codewords is also a codeword. Otherwise, the code is called non-linear. Linear error-correcting codes benefit from desirable properties and structure which greatly reduce their encoding and decoding complexities. Their design and analysis are also facilitated by using the rich literature of linear analysis.

An important class of linear codes is the class of linear block codes, where the $|\Sigma|^K$ codewords form a K-dimensional subspace of the vector space $|\Sigma|^N$. In a linear block code, first K linearly independent vectors $(\underline{g}_0, \underline{g}_1, \ldots, \underline{g}_{K-1})$ are chosen from $|\Sigma|^N$. The codewords are then formed by a linear combination of these K vectors. The K linearly independent vectors are usually grouped together as the rows of a $K \times N$ matrix called the *generator* matrix $\mathbf{G}$. The codewords are then given by

$$\begin{aligned} \underline{v} &= \underline{u} \cdot \mathbf{G} \\ &= (u_0, u_1, \ldots, u_{K-1}) \cdot \begin{bmatrix} \underline{g}_0 \\ \underline{g}_1 \\ \vdots \\ \underline{g}_{K-1} \end{bmatrix} \\ &= u_0\underline{g}_0 + u_1\underline{g}_1 + \cdots + u_{K-1}\underline{g}_{K-1}, \end{aligned} \tag{2.6}$$

where $\underline{u} = (u_0, u_1, \ldots, u_{K-1})$ is the message vector and $\underline{v} = (v_0, v_1, \ldots, v_{N-1})$ is the corresponding codeword.

There is also another matrix associated with linear block codes called the *parity-check* matrix $\mathbf{H}$ which is mostly useful in their decoding. The parity-check matrix is an $(N-K) \times N$ matrix whose rows generate the null space of $\mathbf{G}$. Also a vector $\underline{v}$ of length N is a codeword if and only if $\underline{v} \cdot \mathbf{H}^T = 0$. The rows of the parity-check matrix also give $N-K$ even *parity-check* equations on the message symbols:

$$v_0 h_{i0} + v_1 h_{i1} + \cdots + v_{n-1} h_{i,(N-1)} = 0, \tag{2.7}$$

where $\underline{h}_i = (h_{i0}, h_{i1}, \ldots, h_{i,(N-1)})$ for $i = 0, 1, \ldots, N-K-1$ are the rows of $\mathbf{H}$. Parity-check equations are useful in decoding; if any of them is not satisfied, an error is detected.

As an example, consider the binary linear block code of Table 2.1. We have:

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

and

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}, \tag{2.9}$$

with parity-check equations

$$v_0 + v_3 + v_5 + v_6 = 0 \tag{2.10}$$

$$v_1 + v_3 + v_4 + v_5 = 0 \tag{2.11}$$

$$v_2 + v_4 + v_5 + v_6 = 0. \tag{2.12}$$

2.4.2 LDPC codes: graphical representation

An LDPC code [7] is a linear block code with a *sparse* parity-check matrix. More specifically, the sparsity of parity-check matrix $\mathbf{H}$ means that it has low density of 1's, i.e., the number of 1's in $\mathbf{H}$ grows linearly with the block length N. In this thesis, we are interested in binary LDPC codes. Thus, when we refer to LDPC codes, we mean binary LDPC codes unless otherwise stated.

As said, linear block codes are usually represented by their generator and parity-check matrices[1]. However, for the purpose of analysis, LDPC codes are usually represented graphically. This is because LDPC codes are usually decoded by iterative message-passing decoding algorithms and this graphical representation provides better insight in analyzing these algorithms. Before explaining the graphical representation we give some definitions.

A *bipartite graph* is a graph whose nodes can be divided into two disjoint sets such that the graph's edges are only allowed to connect two nodes from different sets. A *factor graph* [24], is a bipartite graph used to visualize the factorizations of a multi-variate function into local functions with less number of variables. In a factor graph, the nodes are divided into *variable* nodes which represent the variables of the functions and *check* nodes which represent the functions. Factor graphs are mainly used in conjunction with message-passing algorithms to efficiently compute

[1] All linear block codes can also be represented graphically.

the marginal distributions of a multivariate distribution function. Thus, they are also useful in analyzing linear block codes.

To see how a binary linear block code is represented by factor graphs, first recall that a linear block code can be seen as a set of even parity-check equations imposed on the codeword bits. Now, consider a bipartite graph $\mathcal{G}$ with N variable nodes and M check nodes. Here, the variable nodes represent the codeword bits and are binary variables. The check nodes represent the even parity-check equations imposed on the variable nodes. For the i-th check node we have

$$\bigoplus_{j:\mathsf{v}_j \in n(\mathsf{c}_i)} \mathsf{v}_j = 0,$$

where v_j is the j-th variable node, $n(\mathsf{c}_i)$ denotes the set of all variable nodes connected to the i-th check node c_i and $\oplus$ represents the modulo-2 sum.

For example, consider the sample graph of Fig. 2.3. The variable nodes are shown by circles and check nodes by squares. This graph has eight variable nodes and four check nodes. Since each check node represents an even parity-check constraint, there are four parity-check equations written as

$$\begin{aligned}
\mathsf{c}_1 : &\quad \mathsf{v}_2 \oplus \mathsf{v}_3 \oplus \mathsf{v}_4 \oplus \mathsf{v}_7 = 0 \\
\mathsf{c}_2 : &\quad \mathsf{v}_1 \oplus \mathsf{v}_4 \oplus \mathsf{v}_5 \oplus \mathsf{v}_6 \oplus \mathsf{v}_8 = 0 \\
\mathsf{c}_3 : &\quad \mathsf{v}_1 \oplus \mathsf{v}_2 \oplus \mathsf{v}_3 \oplus \mathsf{v}_5 \oplus \mathsf{v}_6 = 0 \\
\mathsf{c}_4 : &\quad \mathsf{v}_4 \oplus \mathsf{v}_5 \oplus \mathsf{v}_6 \oplus \mathsf{v}_7 \oplus \mathsf{v}_8 = 0.
\end{aligned}$$

Now, if we define the $M \times N$ matrix $\mathbf{H}$ as the adjacency matrix of $\mathcal{G}$, then the graph represents a linear code with length N and dimension $K \geq N - M$. The (i, j) entry in $\mathbf{H}$ will be 1 if and only if the i-th check node c_i is connected to the j-th variable node v_j. Since in a linear code with dimension K, the K parity-check equations should be linearly independent, the dimension of the code defined by $\mathcal{G}$ is only equal to $N - M$ if all the parity-check constrains defined by the check nodes are linearly independent. This gives rise to $\mathbf{H}$ being full rank.

All linear block codes have factor graph representations. If the factor graph is also sparse, i.e., the number of edges E grows linearly with the number of variable nodes, then the graph represents an LDPC code.

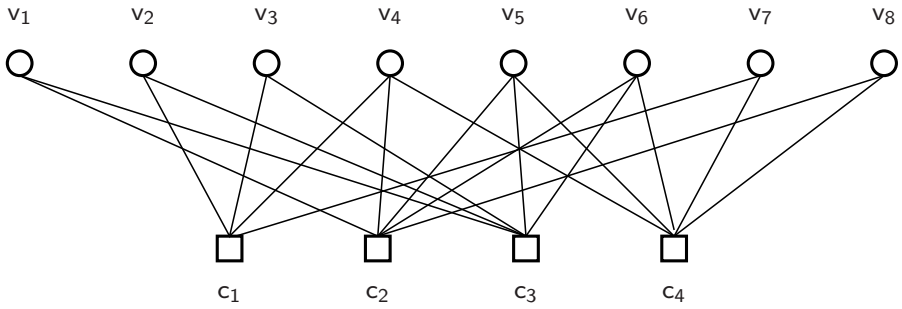


Figure 2.3: A factor graph representing an LDPC code with 8 variable nodes $\mathsf{v}_1, \mathsf{v}_2, \ldots, \mathsf{v}_8$ and 4 check nodes $\mathsf{c}_1, \ldots, \mathsf{c}_4$.

2.4.3 Regular and irregular LDPC codes

Based on their graph structure, LDPC codes are classified as *regular* or *irregular*. If all the variable nodes have the same fixed degree d_{v} and also all the check nodes have the same fixed degree d_{c}, then the LDPC code is called regular. Otherwise, it is called irregular. In a regular LDPC code we have

$$E = d_{\mathsf{v}} \cdot N = d_{\mathsf{c}} \cdot M. \tag{2.13}$$

Also, in the parity-check matrix associated with a regular LDPC code, there are d_{c} number of 1's in each row and d_{v} number of 1's in each column.

In an irregular LDPC code, since not all of the variable or check nodes have the same degree then the variable and check nodes's edges are defined by two edge degree distributions $\{\lambda_2, \lambda_3, \ldots, \lambda_{d_{\mathsf{v}}}\}$ and $\{\rho_2, \rho_3, \ldots, \rho_{d_{\mathsf{c}}}\}$, where λ_i (ρ_i) denotes the fraction of all edges connected to degree i variable (check) nodes. Also, d_{v} and d_{c} represent the maximum degree of variable and check nodes, respectively. For computational purposes, it is also common to represent the degree distributions by their polynomial generators $\lambda(x) = \sum_{i=2}^{d_{\mathsf{v}}} \lambda_i x^{i-1}$ and $\rho(x) = \sum_{i=2}^{d_{\mathsf{c}}} \rho_i x^{i-1}$ [25]. This allows us to write

$$N = E \sum_{i=2}^{d_{\mathsf{v}}} \frac{\lambda_i}{i} = E \int_0^1 \lambda(x) dx, \tag{2.14}$$

and

$$M = E \sum_{i=2}^{d_{\mathsf{c}}} \frac{\rho_i}{i} = E \int_0^1 \rho(x) dx. \tag{2.15}$$

Regular codes are simpler than irregular LDPC codes in terms of their structure. Nevertheless, it is shown in [25] that by using irregular graphs, the performance of

LDPC codes can be extremely improved. The degree distributions together with N define not only a single graph but an ensemble of graphs as the graphs (hence parity-check matrices) which can be constructed from these distributions are not unique. In this thesis, we denote an ensemble of LDPC codes with block length N and degree distributions $\lambda(x)$ and $\rho(x)$ by $\mathcal{C}^N(\lambda(x), \rho(x))$. The rate of this ensemble of codes is given by

$$R = \frac{K}{N} \geq \frac{N-M}{N} = 1 - \frac{\sum_{i=2}^{d_{\mathsf{c}}} \frac{\rho_i}{i}}{\sum_{i=2}^{d_{\mathsf{v}}} \frac{\lambda_i}{i}} = 1 - \frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx}, \tag{2.16}$$

where equality holds only if the corresponding parity-check equations are linearly independent. In large graphs, the fraction of dependent equations is very small. Thus, it is common to ignore these dependencies. An ensemble of regular LDPC codes is usually denoted by the pair $(d_{\mathsf{v}}, d_{\mathsf{c}})$.

As stated, the degree distributions do not uniquely define an LDPC code. Nevertheless, the concentration theorems of [11] show that the performance of all instances from code ensemble concentrates around an average performance for sufficiently large code lengths. This average performance corresponds to the performance of an infinite-length code with the same degree distributions. This means that the average performance of an ensemble of LDPC codes can be determined by its degree distributions. As a result, it is usually sufficient to represent LDPC codes by their degree distributions.

This section is concluded with an example. Consider the ensemble of irregular LDPC codes $\mathcal{C}^{10000}(\lambda(x), \rho(x))$ with

$$\lambda(x) = 0.4x^2 + 0.4x^5 + 0.2x^8$$
$$\rho(x) = x^8.$$

Then $E = 45000$ and $R = 0.5$. The degree distribution shows that 40% of edges (18000 edges) are connected to variable nodes of degree 3 (6000 nodes), another 40% of edges to variable nodes of degree 6 (3000 nodes) and 20% to variable nodes of degree 9 (1000 nodes). Also, all 5000 check nodes have degree 9.

2.4.4 Decoding

The decoding of LDPC codes is usually done by means of a class of iterative decoding algorithms called message-passing algorithms. These algorithms provide very good and efficient decoding performance by exploiting the graphical structure of the code.

In these algorithms, some messages in the form of probabilities or beliefs are passed between the variable and check nodes of the graph along the edges connecting them. The main advantage of these algorithm is their low complexity which scales linearly with the block length N and thus is constant per information bit.

It is usually more advantageous to work with log-likelihood ratios (LLRs) as the messages. LLRs make the message update rules simpler and also provide numerical stability which is useful for implementation.

There are numerous message-passing algorithms available in the literature. Two main message-passing algorithms used in the decoding of LDPC codes are the *sum-product* (also known as belief propagation) [24] and *min-sum* (also known as max-product) [26] algorithms.

The sum-product algorithm

Among other message-passing algorithms, the sum-product (SP) is the most powerful, and simultaneously the most complex algorithm for decoding LDPC codes. The formulation of SP depends on the type of messages being passed in each iteration. Here, we describe the SP algorithm assuming that messages are real valued LLRs.

At first, based on each channel output y, a channel LLR message (also called intrinsic message) is calculated for each corresponding variable node v. This channel LLR message is given by

$$m_{0_\mathsf{v}} = \log \frac{p_{X|Y}(0|y)}{p_{X|Y}(1|y)}, \tag{2.17}$$

where $x \in \{0, 1\}$ denotes the binary channel input. All variable nodes are then initialized by these intrinsic messages and pass these messages to their connecting check nodes c. These messages are denoted by $m^{(0)}_{\mathsf{v}\to\mathsf{c}} = m_{0_\mathsf{v}}$.

Next, the check nodes compute their outgoing messages based on the messages received from the variable nodes. In particular, $m^{(1)}_{\mathsf{c}\to\mathsf{v}}$ is calculated by each check node c and is sent to the connecting variable node v based on the messages previously received from all the neighboring variable nodes except v. This check node message update completes the first half of the first iteration. In the second half iteration, the variable nodes update their messages based on the received messages from check nodes in a similar manner and send them to the neighboring check nodes. This process then continues iteratively.

There are two iterative message update rules used in this iterative process. One for the variable nodes and one for the check nodes. The message update rule at

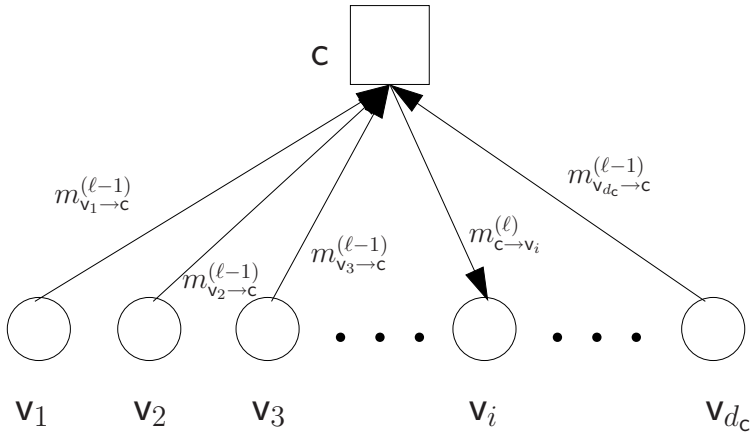


Figure 2.4: The message update process in a check node of degree $d_\mathsf{c} > 4$. The message from c to the variable node v_i is updated based on the messages received from the neighboring variable nodes in the previous iteration.

check node c is as follows [24, 26]:

$$m^{(\ell)}_{\mathsf{c}\to\mathsf{v}} = 2\tanh^{-1}\left(\prod_{\mathsf{v}_i\in n(\mathsf{c})-\{\mathsf{v}\}}\tanh\left(\frac{m^{(\ell-1)}_{\mathsf{v}_i\to\mathsf{c}}}{2}\right)\right), \tag{2.18}$$

where $m^{(\ell)}_{\mathsf{c}\to\mathsf{v}}$ denote the message sent from c to variable node v in the ℓ-th iteration, $n(\mathsf{c})$ is the set of all variable nodes connected to c, and $m^{(\ell-1)}_{\mathsf{v}_i\to\mathsf{c}}$ shows the message sent from variable node v_i to c in the previous iteration. This message updating procedure has been illustrated in Fig. 2.4.

The variable node update rule is given by [24, 26]

$$m^{(\ell)}_{\mathsf{v}\to\mathsf{c}} = m_{0_\mathsf{v}} + \sum_{\mathsf{c}_j\in n(\mathsf{v})-\{\mathsf{c}\}} m^{(\ell)}_{\mathsf{c}_j\to\mathsf{v}}, \tag{2.19}$$

where $n(\mathsf{v})$ denotes the set of all check nodes connected to v. This updating rule is depicted in Fig. 2.5 for a variable node with degree d_v. Notice that these messages are initialized at the start of the decoding by $m^{(0)}_{\mathsf{c}\to\mathsf{v}} = 0$. The messages passed between variable nodes and check nodes are called extrinsic messages.

At each iteration, a decision can be made on the variable node v based on the messages it has received. Since the messages are LLRs, the decision can be made based on the sign on the messages. This is done using the following rule:

$$V = \begin{cases} 0, & m_{0_\mathsf{v}} + \sum_{\mathsf{c}_j\in n(\mathsf{v})} m^{(\ell)}_{\mathsf{c}_j\to\mathsf{v}} > 0 \\ 1, & m_{0_\mathsf{v}} + \sum_{\mathsf{c}_j\in n(\mathsf{v})} m^{(\ell)}_{\mathsf{c}_j\to\mathsf{v}} < 0 \end{cases}, \tag{2.20}$$

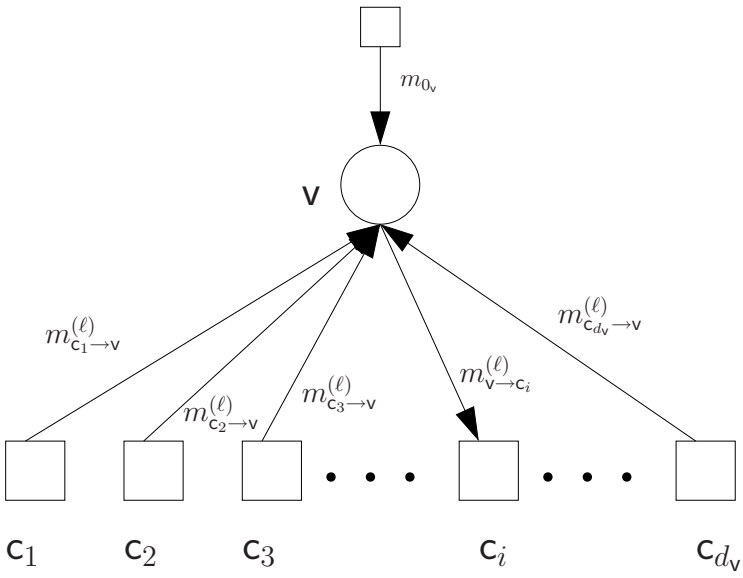


Figure 2.5: The message update process in a variable node of degree $d_\mathsf{v} > 4$. The message from v to the variable node c_i is updated based on the messages received from the neighboring check nodes in the previous half iteration and also the intrinsic messages from the channel m_{0_v}.

and if $m_{0_\mathsf{v}} + \sum_{\mathsf{c}_j \in n(\mathsf{v})} m^{(\ell)}_{\mathsf{c}_j\to\mathsf{v}} = 0$ either $V = 1$ or $V = 0$ can be declared randomly with equal probability. As can be seen from (2.20), a positive LLR votes in favor of $V = 0$ and a negative LLR votes in favor of $V = 1$.

It should be noted that SP is not as powerful as maximum likelihood (ML) decoding in general. This is because SP is only exact in calculating marginal distributions when the factor graph does not have any cycles, i.e., it is a tree. Noticing that avoiding cycles in the factor graphs of finite-length LDPC codes is impossible in practice, SP always gives a sub-optimum performance. Nevertheless, when the block lengths are large, the effect of cycles become negligible under large number of iterations. As a result, the performance of SP is very close to that of ML at large block lengths. SP is more practical in terms of the decoding complexity than ML. Thus, it is normally preferred over ML.

2.4.5 The min-sum algorithm

The min-sum (MS) algorithm is another widely used message-passing algorithm for decoding LDPC codes. It can be viewed as a simplified version of SP [26]. As

a result, its performance is not as good as that of SP. Nevertheless, it has some practical benefits such as its low complexity and the fact that on some channels its performance is independent of channel estimation errors. We will discuss this in more detail in the next chapter.

In MS, the variable node message update rule is similar to SP and is given by (2.19). The check node message update rule is given by [24, 26]

$$m_{\mathsf{c}\to\mathsf{v}}^{(\ell)} = \min_{\mathsf{v}_i\in n(\mathsf{c})-\{\mathsf{v}\}} \left|m_{\mathsf{v}_i\to\mathsf{c}}^{(\ell-1)}\right| \prod_{\mathsf{v}_i\in n(\mathsf{c})-\{\mathsf{v}\}} \operatorname{sign}\left(m_{\mathsf{v}_i\to\mathsf{c}}^{(\ell-1)}\right), \tag{2.21}$$

which can be seen as an approximation to (2.18).

2.4.6 Analysis methods

As stated in Sec. 2.4.4, the SP algorithm correctly calculates the marginal distributions only if the factor graph is cycle-free and avoiding cycles is impossible. In randomly constructed LDPC codes with large block lengths, the neighborhood of a fixed depth ℓ of a large fraction of the variable nodes is a tree. As a result, the SP algorithm correctly calculates the LLRs on these variable nodes for ℓ iterations. The fraction of variable nodes with cycles in their neighborhood of depth ℓ is also small. This means that when the block lengths are large, graph cycles have small effect in decoding performance. Thus for the asymptotic analysis of LDPC codes, it can be assumed that the factor graph is a tree.

Given the linear property of the code and the symmetry of the message update rules, the convergence behavior of the iterative decoding algorithm is independent of the transmitted codeword on symmetric-output channels. This greatly simplifies the analysis as it can be assumed that the all-zero codeword is transmitted.

Under output-symmetric channels, the LLR messages give sufficient statistics for the analyzing the decoding. As a result, it is beneficial to study the statistics of the LLR messages (extrinsic messages) for the purpose of analysis. This is done under the tree assumption for the code's graph and the all-zero codeword transmission assumption. The exact asymptotic analysis method technique for LDPC codes is called *density evolution* [11] which tracks the evolution of the pdf of the extrinsic messages in each iteration.

Although density evolution is exact, it is computationally complex and its exact formulation is not suitable for direct practical use. As a result, a discrete version of density evolution is used for numerical analysis. This technique is called *discrete*

density evolution [27] and is done by quantizing the LLR messages and tracking the probability mass functions (pmfs) instead of pdfs. This method is quite accurate if the quantization step sizes are small. Here, we omit the details of this method. Interested reader can refer to [27].

Decoding threshold

One important asymptotic property of LDPC codes is their decoding threshold.

Definition 2.4 (Decoding threshold)**.** The decoding threshold of an ensemble of infinite-length LDPC codes is defined as the worst channel condition for which the message error rate approaches zero as the number of decoding iterations approaches infinity [11].

In other words, for LDPC codes of asymptotic length, for channel conditions worse than the decoding threshold the decoding is unsuccessful while for channel conditions better than the decoding threshold the decoding is successful. The decoding threshold depends on the code's degree distributions, the decoding algorithm, and the channel type. Decoding threshold is normally found by density evolution.

As an example, on a binary-input additive white Gaussian noise channel, density evolution calculates a threshold of $\sigma = 0.8809$ (equivalent to signal-to-noise ratio (SNR)=1.1015 dB) for $(3, 6)$-regular LDPC codes under SP decoding. This means that decoding is successful when $\sigma \leq 0.8809$ (SNR$\geq$ 1.1015 dB) and unsuccessful when $\sigma > 0.8809$ (SNR$<$ 1.1015 dB).

2.4.7 Design

As stated in Sec. 2.4.3, the average performance of an ensemble of LDPC codes is specified by its degree distributions. Thus, modifying the degree distributions affects the average performance of the ensemble. This hints that good ensemble of codes with a desired performance can be found by optimizing the degree distributions. The optimization should normally be subjected to some constraints. In this process, different measures of performance and different constraints could be considered such as having the highest rate given a fixed decoding threshold [28], having the highest decoding threshold given a minimum code rate [8], or having the lowest decoding complexity given fixed decoding threshold and code rate [28, 29]. The degree distribution optimization process is normally done by numerical optimization methods.

2.5 Modulation and coding

As stated, error-correcting codes add redundant data to the information symbols to detect and correct the errors in the receiver. This advantage of error-correcting codes, however, comes at a price: increased bandwidth. If the transmission symbol rate is kept fixed, adding redundancy is equivalent to using more bandwidth for transmission. This is not desirable in many applications especially in band-limited communications.

In a landmark paper [30], Ungerboeck showed that bandwidth efficiency can be achieved by combining modulation and coding in a single entity called *coded modulation*. The idea in coded modulation was to first provide redundancy by increasing the number of available symbols in the modulation and then limit the signal transitions in a controlled manner using an error-correcting code. He showed that improved performance can be achieved at minimum cost to the required bandwidth. This was due to using an increased number of symbols in the modulation.

After the introduction of coded modulation by Ungerboeck, it was generally accepted that for improved performance and bandwidth efficiency, modulation and coding should be combined. However, in a seminal paper [31], Caire et al. showed that a performance very close to that of coded modulation can be achieved on fading and Gaussian channels by using a separate binary code, a bit-interleaver and non-binary modulation. Their method is called *bit-interleaved coded modulation* (BICM). The advantage of BICM is mostly due to using coding and modulation in separate entities. Thus, the binary error-correcting code can be designed separately by taking advantage of their rich literature and their modern coding methods. Design, analysis, and implementation of the BICM scheme are also much simpler than coded modulation. As a result, BICM provides a robust, bandwidth efficient, and easy to design solution which is desirable in many applications especially for communication on wireless fading channels.

In this thesis, we will use BICM for reliable and bandwidth efficient communication on wireless fading channels. The details are discussed in Chapter 4.

2.6 Channel estimation and timing recovery

As stated, modern error-correcting codes provide close to the Shannon limit performance on many channels and this great performance is due to the existence of

iterative message-passing algorithms. To gain benefit of the superior performance of these algorithms, channel parameters such as noise power or channel fading gain should be perfectly known at the receiver. Usually, this information is provided for the decoder by the channel estimation modules at the receiver[2]. Fig. 2.1 shows a communication system with a channel estimation module at the receiver.

Channel estimation techniques are usually classified as *data-aided* and *non data-aided*. In data-aided methods, a training or pilot sequence is periodically inserted into the transmitted blocks and sent to the receiver. The training sequence, known to the receiver, helps the receiver to track the channel parameters. Channel estimation is usually done by ML, minimum-mean squared error (MMSE), least squared error (LSE), or iterative methods. For example in ML methods, a likelihood function for the received signal is formed which is parameterized by the quantity which is desired to be estimated. An estimate of the desired parameter is then found by maximizing the likelihood function. Normally, channel estimation can also be jointly done with decoding using iterative message-passing algorithms. The main drawback of these data-aided methods, apart from the added complexity, is the increased overhead required for transmitting the training sequences. They are also imperfect, i.e., there is always an error in the estimation which degrades the decoding performance especially on time-varying channels.

In non data-aided methods, no training sequence is used and the receiver exploits the statistical properties of the channel and the transmitted signal to obtain information about the channel parameters. These methods are usually more bandwidth efficient compared to data-aided methods as they require less overhead. Nevertheless, they are usually imperfect and thus may not be desirable for many applications.

In Chapters 3 and 4, we provide solutions for dealing with imperfect or limited knowledge of the channel parameters at the receiver on Gaussian and wireless fading channels.

It is also critical to provide timing information for the decoder. Most of the error-correcting codes are designed assuming perfect synchronization between the receiver and the transmitter. In other words, the arrival time of the transmitted symbols should be perfectly known at the receiver. To this end, timing recovery modules are used at the receiver. Timing recovery techniques are usually classified as *deduc-*

[2]The performance improves if channel information is also provided for the transmitter through a feedback path. In this thesis, we only deal with cases where such feedback path does not exist.

tive and *inductive* [5]. The inductive methods which are more powerful are mostly similar in essence to the channel estimation methods and they are also classified as data-aided and non data-aided. Compared to other channel parameter estimation methods, however, the decoding performance is much more sensitive to the inaccuracy of timing estimates. As we will discuss in Chapter 5, imperfect synchronization leads to insertions and deletions of symbols in the received sequence. Even one undetected insertion/deletion then leads to a catastrophic burst of errors. Conventional error-correcting codes are not capable of dealing with such cases. Noticing that timing recovery becomes even more difficult in channel conditions where powerful error-correcting codes promise to work, motivates finding other solutions. We will introduce codes capable of dealing with imperfect synchronization in Chapter 5.

Chapter 3

Robust LDPC Decoding using Irregular Decoders

For decoding LDPC codes many different message-passing iterative decoding algorithms have been proposed in the literature. For example, *sum-product* (SP) decoding has been shown to be the most powerful in terms of error performance when the code's graph is cycle free [24]. Another widely used soft-decoding message-passing algorithm is the *min-sum* (MS) algorithm, which is an approximation to SP.

In [32], the idea of irregular decoding of LDPC codes is proposed where the decoding algorithm that one variable node (check node) runs remains the same for all iterations, but it may vary from one variable node (check node) to another in a single iteration. As an example of a simple irregular decoder see Fig. 3.1. It has been shown in [33] that irregular decoders can decrease the decoding complexity of LDPC codes and in [34], it has been shown that by using irregular decoders which combine SP and MS, the robustness of iterative decoders to channel estimation errors can be improved. This is motivated from two facts. (1) On the binary-input additive white Gaussian noise channel (BIAWGNC), MS decoder does not need to know the power of the additive noise while SP decoder is quite sensitive to noise power estimation errors. (2) With no channel estimation error, SP can significantly outperform MS. As a result, depending on the amount of channel estimation error, MS can outperform (or be outperformed by) SP. Thus, using an irregular decoder which combines MS and SP can potentially provide the best performance. However, no quantitative robustness measure is presented in [34] and no method is proposed to design the irregular decoder parameters.

In this chapter, we first present a robustness measure for irregular decoders which reflects a fundamental property of the decoder, i.e., its decoding threshold, in

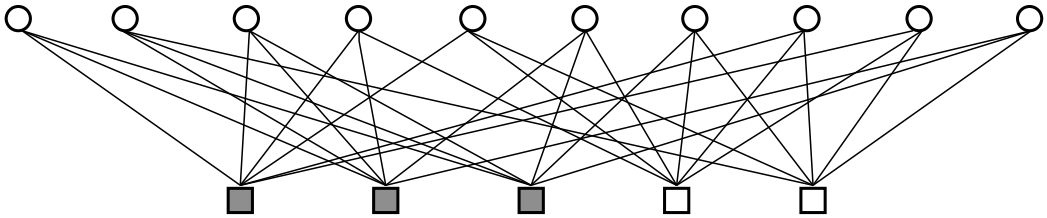


Figure 3.1: The concept of an irregular LDPC decoder. The highlighted check nodes perform SP and the rest of the check nodes perform MS.

the presence of channel estimation errors. This measure is similar to the measure used in [35, 36] to quantify the sensitivity of SP decoders to channel estimation errors. Next, for a given LDPC code, we propose a method to design irregular decoders that provide the best robustness to the channel estimation errors on the BIAWGNC. As a first step, for a given code, we find the irregular decoder with optimal mixture of SP and MS to obtain the best decoding threshold in the presence of channel estimation errors. Next, to achieve the widest possible convergence region we provide an iterative approach for a joint optimization of the irregular code's degree distributions and the irregular decoder. In particular, this approach combines the conventional irregular LDPC codes optimization methods with the irregular decoder design method presented in this chapter.

3.1 SP, MS, and channel estimation errors

Consider a BIAWGNC whose output y is defined by

$$y = x + z, \tag{3.1}$$

where $x \in \{-1, 1\}$ is the channel input and z is the zero-mean Gaussian additive noise with variance σ^2_{act}. We also represent the channel estimation error with α defined as

$$\alpha = \frac{\sigma^2_{\text{est}}}{\sigma^2_{\text{act}}}, \tag{3.2}$$

where σ^2_{est} is the decoder's estimate of the channel noise variance.

On this channel, assuming equiprobable inputs, the true channel LLRs are given by

$$l = \log \frac{P(x = +1|y)}{P(x = -1|y)} = \log \frac{P(y|x = +1)}{P(y|x = -1)} = \frac{2}{\sigma^2_{\text{act}}} y. \tag{3.3}$$

As described in Section 2.4.4, SP and MS use these channel LLRs as their intrinsic messages $m_{0_{\text{v}}}$. When there is channel estimation error, channel LLRs calculated at

the decoder are given by

$$\hat{l} = m_{0_v} = \frac{2}{\sigma^2_{\text{est}}} y = \frac{2}{\alpha \sigma^2_{\text{act}}} y, \tag{3.4}$$

which shows how channel estimation errors (quantified by α) affects the intrinsic messages m_{0_v}.

The variable node message update rules for SP and MS are similar and are given by (2.19). Noticing that the check node message update rule for MS is given by (2.21), it can be seen that the term $\frac{2}{\alpha \sigma^2_{\text{act}}}$ can be factored out from both variable and check node update rules. Since the final decision on the variable nodes is made based on the sign of the variable node messages as given in (2.20), α does not have any affect on the final decisions. Thus, MS is robust to channel estimation errors.

Notice that the same discussion does not apply to SP since $\frac{2}{\alpha \sigma^2_{\text{act}}}$ cannot be factored out from the check node message update rule given by (2.18). As a result, SP is vulnerable to channel estimation errors.

3.2 Robust irregular decoders

In this section, we define and design robust irregular decoders. As stated, the goal is to mix the advantages of SP and MS in an irregular decoder design to obtain the best performance in the presence of channel estimation errors.

Since the variable-node update rules are the same for both SP and MS, we define an ensemble of irregular LDPC code-decoder pairs by $(\lambda(x), \rho(x), \beta)$, where $\lambda(x)$ and $\rho(x)$ are the code's original degree distribution polynomials [8] and $0 \leq \beta \leq 1$ is the fraction of check nodes performing the MS algorithm.

Now, assume that due to channel estimation imperfections, α changes in the range $[\alpha_{\min}, \alpha_{\max}]$[1]. For a given code characterized by $(\lambda(x), \rho(x))$, our first goal is to find the value of β which provides the best tolerance of channel mismatch. This is equivalent to having the best decoding threshold measured in terms of the required $\frac{E_b}{N_0}$ for guaranteed convergence in the presence of channel mismatch. Here, the definition of decoding threshold is rather different than its conventional meaning since the effect of channel estimation error must be taken into account. To this end, first notice that for a given β and a specific value of α, the minimum $\frac{E_b}{N_0}$ required for successful convergence of the given code-decoder pair (here denoted by $\frac{E_b}{N_0}^*$) can

[1]Since α is usually a random variable in practice, this range can be more precisely defined as a confidence interval which contains α with high probability.

be found using density evolution [8]. Since α changes in the range $[\alpha_{\min}, \alpha_{\max}]$, the definition of the decoding threshold must be modified to

$$\left(\frac{E_b}{N_0}\right)^{\text{thr}} = \max_{\alpha_{\min} \leq \alpha \leq \alpha_{\max}} \left(\frac{E_b}{N_0}\right)^{*}. \tag{3.5}$$

This is the smallest value of $\frac{E_b}{N_0}$ required for guaranteed convergence of the code-decoder in the presence of channel estimation error. In fact, $\left(\frac{E_b}{N_0}\right)^{\text{thr}}$ acts as a robustness measure since it quantitatively reflects the ability of the code-decoder to withstand channel estimation errors.

It should be noted that for computing $\left(\frac{E_b}{N_0}\right)^{\text{thr}}$, it is only necessary to find $\left(\frac{E_b}{N_0}\right)^{*}$ for $\alpha = \alpha_{\min}$ and $\alpha = \alpha_{\max}$ and pick the larger one. In other words, the maximum decoding threshold occurs at the boundaries of α. This can be argued as follows. It is known that the MS algorithm, overestimates the amplitude of the LLR messages compared to the SP [37]. When $\alpha > 1$, since $|\frac{\hat{l}}{l}| = \frac{1}{\alpha} < 1$, the amplitude of the channel LLRs are underestimated by the receiver. As a result, the overestimation in the output of the check nodes running MS is partially compensated when these underestimated LLRs pass through them. Increasing α makes $|\frac{\hat{l}}{l}|$ larger. This means there exist a point $\alpha = \alpha_0$ that $\left(\frac{E_b}{N_0}\right)^{*}$ is a decreasing function of α when $\alpha \leq \alpha_0$ and is an increasing function of α when $\alpha > \alpha_0$ where α_0 is where the effect of channel LLR underestimation becomes stronger than the overestimation caused by MS. When $\alpha < 1$, since $|\frac{\hat{l}}{l}| = \frac{1}{\alpha} > 1$, i.e., we only have overestimation, the performance gets worse by decreasing α and as a result $\left(\frac{E_b}{N_0}\right)^{*}$ increases. Thus, over the whole range of α we have only one local minimum for $\left(\frac{E_b}{N_0}\right)^{*}$ and thus over any range of α the maximum value occurs at the boundaries of α. The significance of this observation is that in (3.5), one only needs to run density evolution twice and not for all values of α and we can write

$$\left(\frac{E_b}{N_0}\right)^{\text{thr}} = \max_{\alpha=\alpha_{\min}, \alpha=\alpha_{\max}} \left(\frac{E_b}{N_0}\right)^{*}. \tag{3.6}$$

To find the best tolerance of channel mismatch, we must solve the following optimization problem:

$$\beta_{\text{opt}} = \arg\min_{\beta} \left(\frac{E_b}{N_0}\right)^{\text{thr}}. \tag{3.7}$$

To solve (3.7), by using density evolution, we obtain the modified decoding threshold defined in (3.5) for a finite set of points in the range $0 \leq \beta \leq 1$. Next, we choose the β which gives the best decoding threshold and denote the resulting code-decoder pair by $(\lambda(x), \rho(x), \beta_{\text{opt}})$.

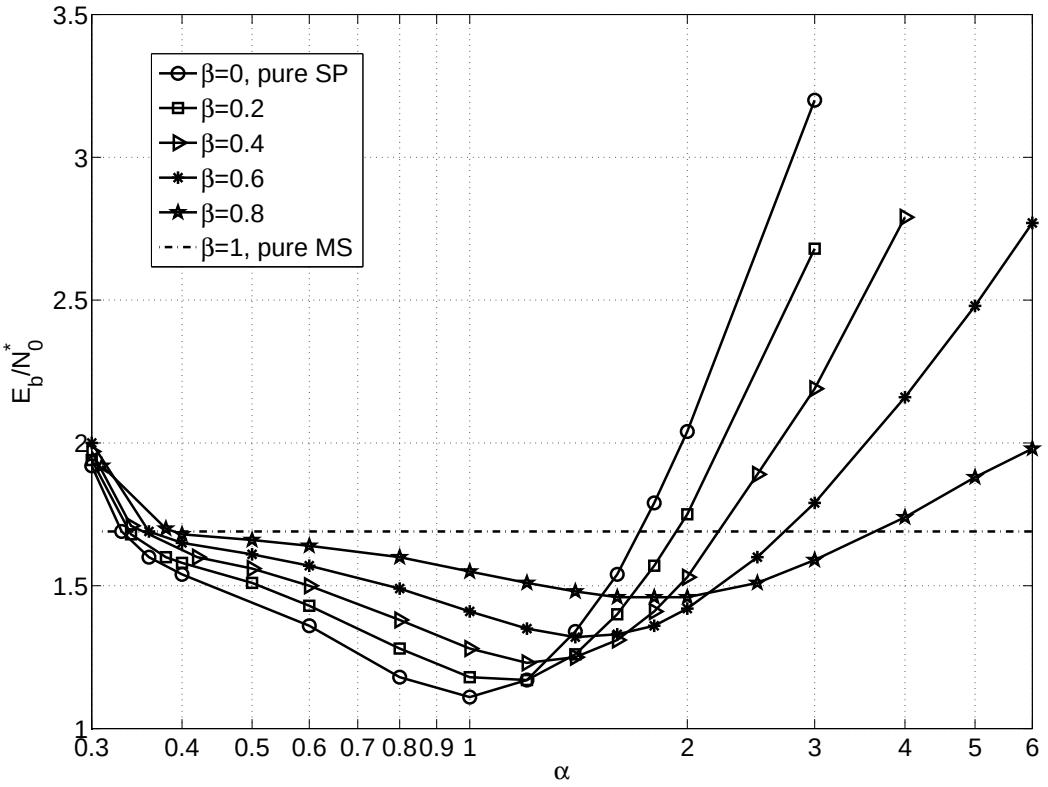


Figure 3.2: The decoding threshold of various (x^2, x^5, β) code-decoder pairs as a function of $\alpha = \sigma^2_{\text{est}}/\sigma^2_{\text{act}}$ and β.

Example 3.1. Consider the ensemble of (x^2, x^5)-regular LDPC codes on the BI-AWGNC. The decoding threshold of various (x^2, x^5, β) code-decoders is depicted versus α in Fig. 3.2. For each β, the region above the corresponding curve is where the decoder succeeds to converge to an error rate of 10^{-6} assuming a maximum log-likelihood ratio (LLR) value of 25. The decoding threshold is 1.70 dB under pure MS decoding (i.e., $(x^2, x^5, 1)$) and is 1.11 dB under pure SP decoding (i.e., $(x^2, x^5, 0)$) with perfect noise variance estimation ($\alpha = 1$). It is clear from the figure that in some regions, an irregular decoder is more robust to channel estimation errors and has a wider convergence region. For example, when $\frac{E_\text{b}}{N_0} = 1.20$ dB and $\alpha = 1.25$, $(x^2, x^5, 0.2)$ succeeds to converge to small error rates while both pure MS and pure SP decoders fail to converge. □

Example 3.2. Now consider that $\alpha \in [0.5, 2]$ which reflects ± 3 dB error in estimation of the variance of the additive noise. As can be seen from Fig. 3.2, a pure min-sum decoder is guaranteed to converge for $\frac{E_\text{b}}{N_0} \geq 1.70$ dB and a pure sum-product decoder is guaranteed to converge for $\frac{E_\text{b}}{N_0} \geq 2.04$ dB. However, by solving

(3.7), we get $\beta_{\text{opt}} = 0.39$ (see Fig. 3.3) whose convergence region is $\frac{E_b}{N_0} \geq 1.55$ dB. □

It is worth mentioning that for lower rate codes, a larger improvement (e.g., around 0.3 dB for (x^2, x^3)-regular LDPC ensemble) can be obtained by using irregular decoders. We used the (x^2, x^5) code since most of its performance results exist in the literature.

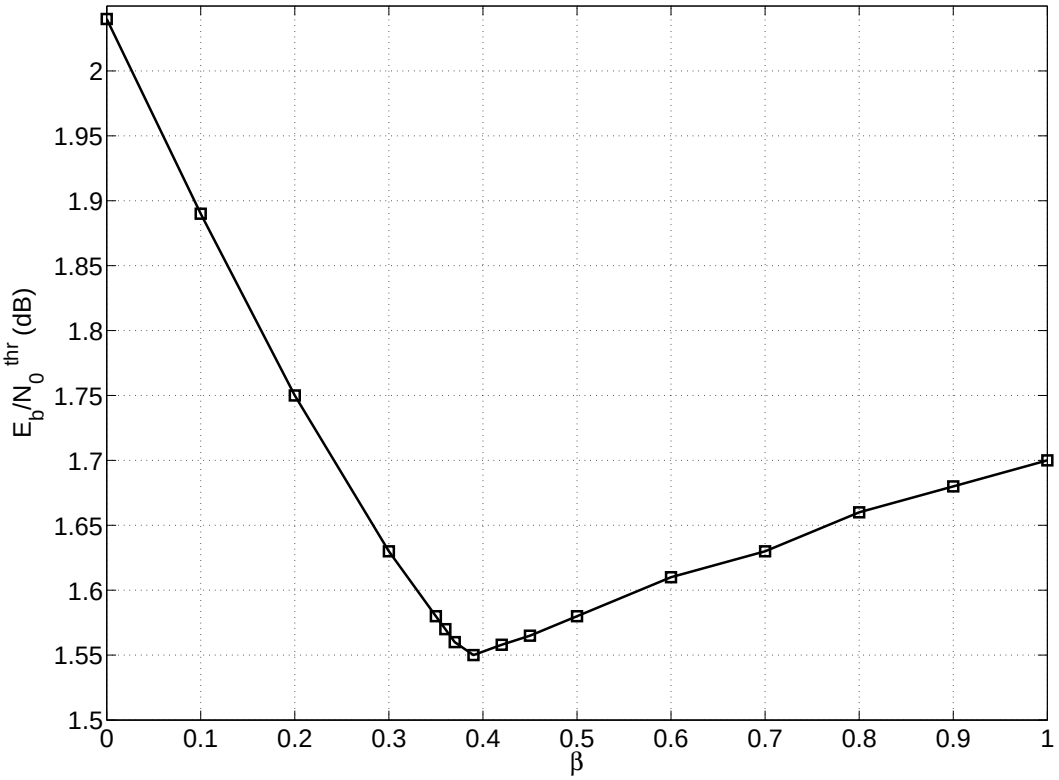


Figure 3.3: The decoding threshold of various (x^2, x^5, β) code-decoder pairs (defined in Eq. (3.5)) as a function of β when $\alpha \in [0.5, 2]$.

3.3 Irregular code-decoder design

The irregular decoder design method presented in the previous section provides the widest convergence region and the best robustness for a fixed given LDPC code. It should be noted that the convergence region of an LDPC code-decoder pair in the presence of channel mismatch (obtained by solving (3.7)) depends on the code's degree distributions ($\lambda(x)$ and $\rho(x)$). As a result, a joint optimization of the irregular code's degree distributions and the decoder seems reasonable. More specifically, we are interested in the joint optimization of $(\lambda(x), \rho(x), \beta)$ for a given code rate R, and α range which gives the minimum $\frac{E_b}{N_0}^{\text{thr}}$ on the BIAWGNC.

Since the search space for this optimization is very large, we present an iterative approach, described below, to solve this optimization. Although this iterative approach may not provide the globally optimum solution, it converges to a near optimum solution very fast (as discussed later and shown by numerical results).

1. Fix $\alpha_{\min}$, $\alpha_{\max}$, and R in all the steps and assume an initial $\lambda(x)$ and $\rho(x)$. Let $\ell = 1$ be the iteration number, $\beta_{\text{opt}}(0) = 0$, and δ be a small positive constant.

2. For the fixed code specified by $(\lambda(x), \rho(x))$, solve (3.7) and find $\beta_{\text{opt}}(\ell)$ and its corresponding threshold and denote it by $\frac{E_b}{N_0}^{\text{thr}^*}(\ell)$.

3. If $|\frac{E_b}{N_0}^{\text{thr}^*}(\ell) - \frac{E_b}{N_0}^{\text{thr}^*}(\ell - 1)| < \delta$ stop, otherwise go to the next step.

4. Let $\beta = \beta_{\text{opt}}(\ell)$. Optimize $\lambda(x)$ and $\rho(x)$ by minimizing the code's decoding threshold (in terms of the required $\frac{E_b}{N_0}$) while guaranteeing convergence of the code-decoder $(\lambda(x), \rho(x), \beta)$ for $\alpha = \alpha_{\min}$ and $\alpha = \alpha_{\max}$.

5. Let $\ell := \ell + 1$ and go to step 2.

Step 4 of the algorithm can be done by the numerical optimization techniques used in [8,27,36]. To guarantee the convergence of the code-decoder pair for $\alpha = \alpha_{\min}$ and $\alpha = \alpha_{\max}$, we can add their convergence conditions to the optimization constraints.

In each step of the algorithm, by optimizing β or $(\lambda(x), \rho(x))$, the minimum decoding threshold is found. Since there exists a fundamental limit, i.e., the Shannon limit, on the code's decoding threshold, this iterative approach is guaranteed to converge to a solution. It is worth mentioning that with proper initialization, this approach usually converges very fast in a few iterations (normally less than 4).

Example 3.3. For $R = 0.50$ and $\alpha \in [0.5, 2]$, using 9-bit message quantization, maximum LLR value of 30, maximum variable-node degree of 15, and the procedure outlined above, an irregular code-decoder pair $(\lambda(x), \rho(x), \beta_{\text{opt}})$ has been optimized. The optimized parameters are

$$\begin{aligned} \lambda(x) &= 0.2219x + 0.3035x^2 + 0.0345x^3 + 0.0006x^{13} \\ &\quad + 0.4398x^{14} \\ \rho(x) &= x^7, \end{aligned} \tag{3.8}$$

and $\beta_{\text{opt}} = 0.18$. Using the designed code-decoder $(\lambda(x), \rho(x), 0.18)$ it is possible to guarantee convergence for $\frac{E_b}{N_0} \geq 1.075$ dB. We compare the designed code with another code with the same rate optimized for the best performance under SP without having channel mismatch [8]. The degree distribution of the code is as follows:

$$\begin{aligned} \lambda(x) &= 0.2382x + 0.2100x^2 + 0.0349x^3 + 0.1202x^4 \\ &\quad + 0.0159x^6 + 0.0045x^{13} + 0.3763x^{14} \\ \rho(x) &= x^7. \end{aligned} \tag{3.9}$$

Although this code has been optimized for SP, in the presence of ± 3 dB mismatch ($\alpha \in [0.5, 2]$) and without irregular decoding, the best decoding threshold for this code is achieved with a pure MS decoder compared to a pure SP decoder. A pure MS decoder gives a convergence region of $\frac{E_b}{N_0} \geq 1.478$ dB. It can be seen that by using the irregular code-decoder optimization approach it is possible to obtain about 0.4 dB threshold improvement in the presence of ± 3 dB channel mismatch.

Codes of length 10^4 have been constructed randomly using (3.8) and (3.9) which we call code 1 and code 2, respectively. The simulated bit error rate (BER) curves for both codes have been compared in Fig. 3.4. Code 1 is decoded with an irregular decoder with $\beta = 0.18$ and the worst case corresponding to $\alpha = 0.5$ has been plotted. Code 1 is also decoded by a pure SP decoder and the worst case ($\alpha = 2$) has been plotted. Since the values of α corresponding to the worst cases are different, the slopes of the two curves are also different. Code 2 is decoded with a pure MS decoder. The BER curves also confirm the achieved performance improvement. □

3.4 Conclusion

We proposed a measure of robustness to channel estimation errors for irregular decoders and proposed a method to design the most robust irregular decoder for a pre-selected LDPC code. We then provided an iterative procedure to optimize the irregular code-decoder pair jointly. Significant performance improvement can be seen by using the presented approach compared to conventional irregular code design methods.

As we stated in Chapter 1, our solution can be of interest in the cases that perfect channel estimation is not available at the decoder or when the channel noise power changes in time.

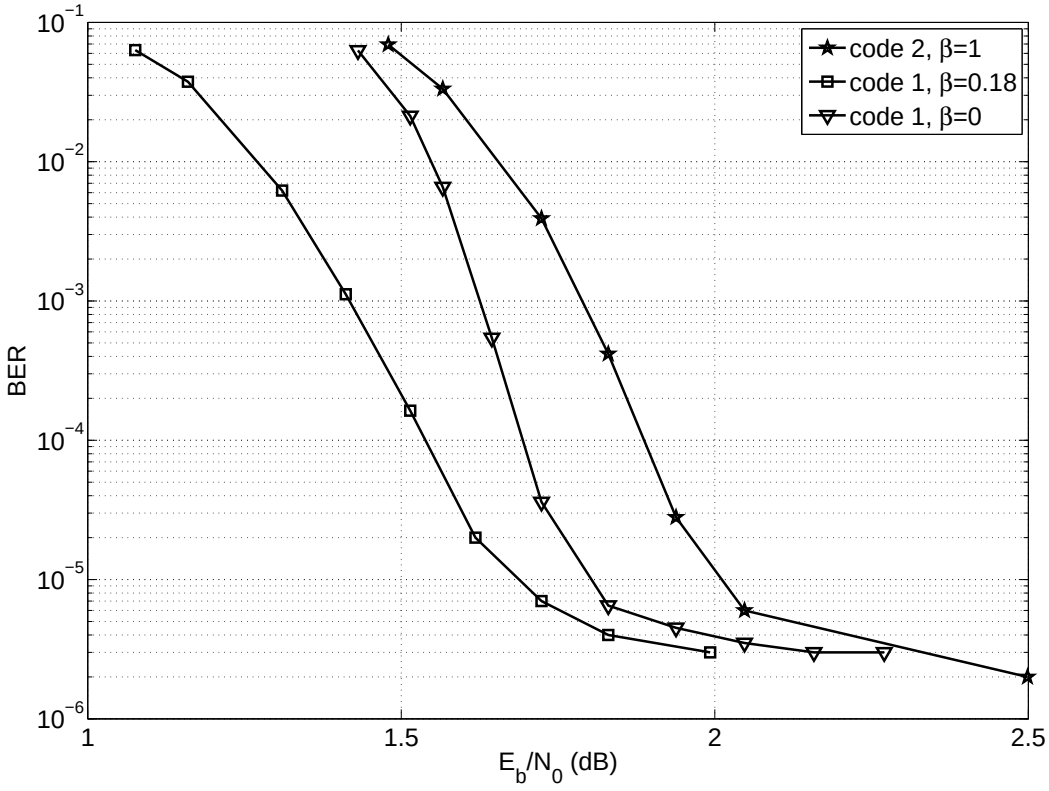


Figure 3.4: Comparison of the BER performance of code 1 decoded by an irregular decoder with $\beta = 0.18$ and a pure SP decoder and code 2 decoded by a pure MS decoder.

Chapter 4

LLR Approximation for Iterative Decoding on Wireless Channels

It is well known that soft-decision decoding algorithms outperform hard-decision decoding algorithms. In hard-decision decoding, decisions on the transmitted sequence are made without considering how reliable the channel output is. In soft-decision decoding, however, reliability metrics are calculated at the receiver based on the channel output. The decoder uses these reliability measures to gain knowledge of the transmitted codewords. The superiority of soft decoding comes at the expense of their higher complexity.

LLRs have been shown to be very efficient metrics for soft decoding of many powerful codes such as the convolutional codes [22], turbo codes [6], and LDPC codes [7]. LLRs offer practical advantages such as numerical stability and simplification of many decoding algorithms. Moreover, due to some properties of the probability density function (pdf) of the LLRs, such as symmetry and invariance [38], LLRs are used as convenient tools for the performance analysis of binary linear codes [8, 38, 39]. Nevertheless, on many communication channels, even for binary modulations, i.e., binary phase-shift keying (BPSK), channel LLRs are complicated functions of the channel output [40]. This fact greatly increases the complexity of the LLR calculation modules in the decoder causing decoding delays and power dissipation. In high speed wireless transmissions, the decoder may not be able to handle this complexity. Thus, for an efficient implementation of the decoder, approximate LLRs should be considered.

Approximate LLRs have been previously used in the literature [40–43]. Piecewise

linear LLRs have been suggested in [42] for soft Viterbi decoding of convolutional codes in the HIPERLAN/2 standard [44]. The presented method uses the log-sum approximation which is only accurate at high signal-to-noise ratio (SNR). Moreover, it assumes that perfect channel state information (CSI) is available at the receiver. In [45], linear LLRs have been used for BPSK modulation on uncorrelated fading channels without CSI and an empirical measure of LLR accuracy has been introduced. Using that measure, linear LLR approximating functions have been designed with almost no performance gap to that of true LLR calculation. The proposed measure, however, is heuristic and is not analytically justified. Also, it is only applicable to symmetric channels and BPSK. With non-binary modulations, used in most practical systems, a linear approximation of bit LLRs is not always a good choice. Moreover, the equivalent bit-channels are asymmetric.

In this chapter, we seek approximate LLRs for both binary and non-binary signalling over uncorrelated fading channels without CSI at the receiver. We first analytically justify the measure introduced in [45] as an LLR accuracy measure. Other measures of accuracy can also be considered. We discuss some of them and show that approximate LLRs designed under the proposed measure outperforms those of other measures. Next, to be able to apply the LLR approximation method to non-binary modulations and asymmetric channels, we generalize the proposed measure to asymmetric channels. Since the true LLR calculating functions are no longer linear under non-binary modulations, we consider non-linear LLR approximating functions. We finally optimize the parameters of the LLR approximating functions using the generalized LLR accuracy measure and evaluate the proposed solution in terms of bit error rate (BER) and the maximum achievable rates on the channel. To compute LLRs at bit levels independently, we consider bit-interleaved coded modulation (BICM) introduced in Sec. 2.5 [31].

While our approaches are general, to demonstrate our methods, we focus on piecewise linear approximations. Such approximations are easy to implement and we observe that, when the parameters are optimized, they perform very close to true LLRs. We prove that the optimization of these piecewise linear approximations is convex. Due to the close-to-capacity performance of LDPC codes on many channels [27,41,46], we employ LDPC-coded BICM [47] to show that even with the proposed approximation, close-to-capacity performance is obtained.

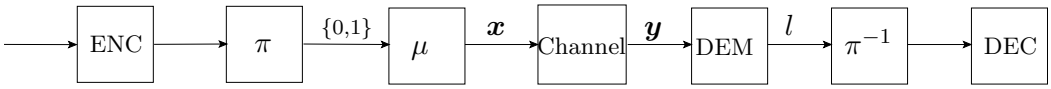


Figure 4.1: The block diagram of the BICM scheme. ENC and DEC represent the binary encoder and decoder, π and π^{-1} are the bit interleaver and de-interleaver, μ is the symbol mapping in the modulation, and DEM is the demodulator which calculates the required LLRs.

4.1 Problem definition

Consider a flat-fading environment where the received signal is expressed as

$$\boldsymbol{y} = r \cdot \boldsymbol{x} + \boldsymbol{z}, \tag{4.1}$$

where $\boldsymbol{x}$ is the complex transmitted signal chosen from the signal set $\mathcal{X} \subseteq \mathbb{C}$ of size $|\mathcal{X}| = 2^m$, $r \geq 0$ is the channel fading gain with arbitrary pdf $f_R(r)$ which changes independently from one channel use to another, and $\boldsymbol{z}$ is the additive noise which is a complex zero-mean white Gaussian random variable with variance $2\sigma^2$, i.e., $\boldsymbol{z} \sim \mathcal{CN}(0, 2\sigma^2)$. Notice that if the modulation is real (e.g., BPSK), then the transmitted signal x, additive noise z, and the received signal y are real variables and $z \sim \mathcal{N}(0, \sigma^2)$. Also notice that this channel model is equivalent to knowing and compensating for the channel phase shifts at the receiver. Thus when we say no CSI is available at the receiver, we mean the receiver does not know the amplitude of the fading gain.

4.1.1 LLRs for equivalent bit-channels

Using the BICM scheme [31], the information sequence is first encoded by a binary code. Next, the coded sequence is bit interleaved and is broken into m-bit sequences which are then Gray labeled onto signals in $\mathcal{X}$ and transmitted on the channel. Fig. 4.1 shows the block diagram of the BICM scheme. Assuming ideal interleaving, the system can be seen equivalently as m parallel independent and memoryless binary-input *bit-channels* (see Fig. 4.2). At the receiver, based on $\boldsymbol{y}$, LLRs are computed for each bit-channel independently from other bits. These LLRs are then de-interleaved and passed to the decoder.

When the channel fading gain r is known at the receiver for each channel use, the true LLR for the ith bit-channel, assuming uniform input distribution, is calculated

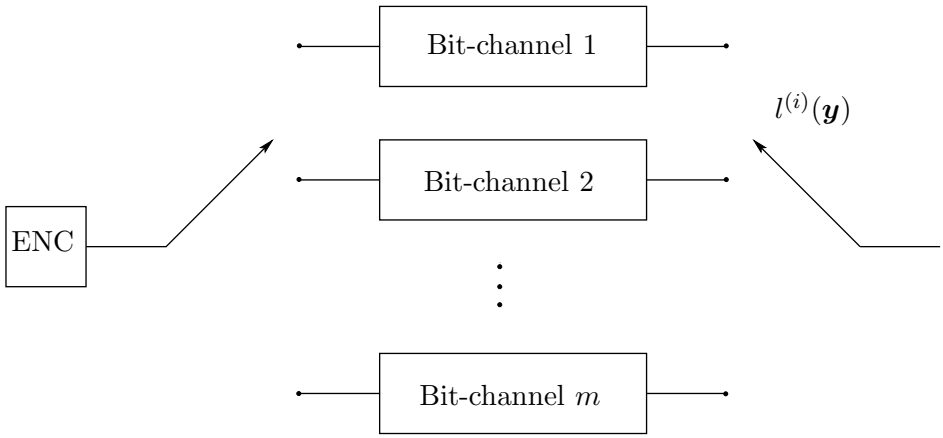


Figure 4.2: Equivalent channel model for BICM. The switch models ideal interleaving and each bit-channel refers to one position in the label of the signals in $\mathcal{X}$. Also, $l^{(i)}(\boldsymbol{y})$ refers to the LLR of the bit-channel i.

as

$$l^{(i)} = \log \frac{P(\boldsymbol{y}|b^{(i)}(\boldsymbol{x}) = 0, r)}{P(\boldsymbol{y}|b^{(i)}(\boldsymbol{x}) = 1, r)} = \log \frac{\sum_{\boldsymbol{x}\in\mathcal{X}_0^{(i)}} f_{\boldsymbol{Y}|\boldsymbol{X},R}(\boldsymbol{y}|\boldsymbol{x}, r)}{\sum_{\boldsymbol{x}\in\mathcal{X}_1^{(i)}} f_{\boldsymbol{Y}|\boldsymbol{X},R}(\boldsymbol{y}|\boldsymbol{x}, r)} = g_r^{(i)}(\boldsymbol{y}), \qquad (4.2)$$

where $i \in \{1, \ldots, m\}$, $b^{(i)}(\boldsymbol{x})$ is the ith bit of the label of $\boldsymbol{x}$, $\mathcal{X}_w^{(i)}$ is the subset of signals in $\mathcal{X}$ where $b^{(i)}(\boldsymbol{x}) = w$, and the conditional pdfs are given by $f_{\boldsymbol{Y}|\boldsymbol{X},R}(\boldsymbol{y}|\boldsymbol{x}, r) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{|\boldsymbol{y}-r\boldsymbol{x}|^2}{2\sigma^2}\right)$ or $\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y-rx)^2}{2\sigma^2}\right)$ when the signal set is real. Also, $g_r^{(i)}(\boldsymbol{y})$ represents $l^{(i)}$ as a function of $\boldsymbol{y}$ when r is known. When r is not known at the receiver, the true LLR is calculated as

$$l^{(i)} = \log \frac{P(\boldsymbol{y}|b^{(i)}(\boldsymbol{x}) = 0)}{P(\boldsymbol{y}|b^{(i)}(\boldsymbol{x}) = 1)} = \log \frac{\sum_{\boldsymbol{x}\in\mathcal{X}_0^{(i)}} f_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y}|\boldsymbol{x})}{\sum_{\boldsymbol{x}\in\mathcal{X}_1^{(i)}} f_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y}|\boldsymbol{x})} = g^{(i)}(\boldsymbol{y}), \qquad (4.3)$$

where $f_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y}|\boldsymbol{x}) = \int_0^\infty \frac{1}{2\pi\sigma^2} \exp\left(-\frac{|\boldsymbol{y}-r\boldsymbol{x}|^2}{2\sigma^2}\right) f_R(r)dr$, and $g^{(i)}(\boldsymbol{y})$ represents $l^{(i)}$ as a function of $\boldsymbol{y}$.

As can be seen from (4.2) and (4.3), calculation of $g_r^{(i)}(\boldsymbol{y})$ and $g^{(i)}(\boldsymbol{y})$ is complicated. In particular, the calculation of $g^{(i)}(\boldsymbol{y})$ involves evaluating sums of integrations which are usually not available in closed forms. Moreover, the number of terms in each sum grows exponentially with the number of bits. This makes the LLR calculation complex and as a result approximate LLRs ($\hat{g}_r^{(i)}(\boldsymbol{y})$ and $\hat{g}^{(i)}(\boldsymbol{y})$) are of practical interest. One approximation which is useful at high SNR is obtained by the log-sum approximation: $\log \sum_k u_k \approx \max_k \log u_k$. This approximation is good

when the sum is dominated by a single large term. Thus,

$$\hat{g}_r^{(i)}(\boldsymbol{y}) = \log \frac{\max_{\boldsymbol{x}\in\mathcal{X}_0^{(i)}} f_{\boldsymbol{Y}|\boldsymbol{X},R}(\boldsymbol{y}|\boldsymbol{x},r)}{\max_{\boldsymbol{x}\in\mathcal{X}_1^{(i)}} f_{\boldsymbol{Y}|\boldsymbol{X},R}(\boldsymbol{y}|\boldsymbol{x},r)}, \tag{4.4}$$

and

$$\hat{g}^{(i)}(\boldsymbol{y}) = \log \frac{\max_{\boldsymbol{x}\in\mathcal{X}_0^{(i)}} f_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y}|\boldsymbol{x})}{\max_{\boldsymbol{x}\in\mathcal{X}_1^{(i)}} f_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y}|\boldsymbol{x})} = \log \frac{\max_{\boldsymbol{x}\in\mathcal{X}_0^{(i)}} \int_0^\infty f_{\boldsymbol{Y}|\boldsymbol{X},R}(\boldsymbol{y}|\boldsymbol{x},r) f_R(r) dr}{\max_{\boldsymbol{x}\in\mathcal{X}_1^{(i)}} \int_0^\infty f_{\boldsymbol{Y}|\boldsymbol{X},R}(\boldsymbol{y}|\boldsymbol{x},r) f_R(r) dr}.$$

The log-sum approximation is particularly useful when CSI is available at the receiver, in which (4.4) leads to piecewise linear LLRs which can be efficiently implemented [42]. However, with no CSI, the log-sum approximation no longer leads to piecewise linear functions and still involves complicated integrations[1]. The focus of our work is on cases where CSI is not available. Nonetheless, we seek approximate LLRs which are piecewise linear functions of $\boldsymbol{y}$.

Now, consider general LLR approximation functions parameterized by sets of parameters $\mathcal{A}_i$

$$\hat{l}^{(i)} = \hat{g}_{\mathcal{A}_i}^{(i)}(\boldsymbol{y}), \quad \text{for} \quad i = 1, \ldots, m. \tag{4.5}$$

This function maps the complex received signal $\boldsymbol{y}$ to real-valued approximate LLR for the ith bit-channel. Clearly, it is desired to choose $\mathcal{A}_i$ such that accurate LLR estimates are obtained. To this end, we need an LLR accuracy measure.

4.2 Measures of LLR accuracy

To find good LLR approximating functions different methods can be applied. The most trivial method is the direct exhaustive search using Monte Carlo simulation. In the case of LDPC codes, instead of Monte Carlo simulation, one may use density evolution [27]. Although these approaches are exact, they are too complex to be practical even for a simple linear approximating function with only one design variable. Therefore, indirect methods based on different measures of accuracy of LLRs will be investigated.

One measure of accuracy can be the minimum-mean squared error (MMSE) defined as

$$\mathcal{A}_i^{\text{MMSE}} = \arg\min_{\mathcal{A}_i} E\left[|l^{(i)} - \hat{l}^{(i)}|^2\right],$$

for $i = 1, \ldots, m$. Alternatively, from the analysis of iterative decoders, we know that the pdf of channel LLRs has a fundamental effect on the performance of the

[1] In fact in this case, the log-sum approximation only replaces sums with comparisons.

decoder [8,11]. Thus, we can look for measures which make the pdf or the probability mass function (pmf) of the approximate LLRs close to the pdf or the pmf of the true LLRs. One way of doing this is to consider the relative entropy or the Kullback-Leibler distance [20] between the pmfs of the true LLRs and approximate LLRs. The relative entropy between two pmfs $p_L(l)$ and $q_L(l)$ is defined as

$$D(p_L \| q_L) = \sum_{l \in \mathfrak{L}} p_L(l) \log \frac{p_L(l)}{q_L(l)},$$

where $\mathfrak{L}$ is the alphabet set of the random variable L. The relative entropy is non-negative and is zero if and only if $p_L(l) = q_L(l)$ for all l. Thus, the relative entropy can be used as a measure of accuracy of the approximate LLRs. Therefore, we define

$$\mathcal{A}_i^{\mathrm{D}} = \arg\min_{\mathcal{A}_i} D(p_L \| p_{\hat{L}}),$$

where $p_{\hat{L}}$ is the pmf of the approximate LLRs parameterized by $\mathcal{A}_i$ as in (4.5).

The main drawback of the MMSE and relative entropy methods is that they require the pdf of true LLRs. Since the motivation for using approximate LLRs was to avoid the complexity of finding true LLRs, a measure which does not rely on true LLRs is preferred. In the next section, we use the method we proposed in [45] and justify it as a measure. This measure gives very close to optimal BER results, yet does not require the pdf of true LLRs.

4.2.1 LLR accuracy measure for symmetric binary-input channels

Symmetric LLR approximation

First assume that the modulation is BPSK, i.e., $\mathcal{X} = \{-1, 1\}$. Thus, there is only one bit-channel. Also, assume that the channel is output-symmetric, i.e., $P(y|x = +1) = P(-y|x = -1)$. From the definition of LLR and the symmetry of the channel we have

$$g(-y) = \log \frac{P(x = +1| - y)}{P(x = -1| - y)} = \log \frac{P(x = -1|y)}{P(x = +1|y)} = -g(y).$$

In order to keep the symmetry properties of the channel, we are most interested in an approximate LLR function that satisfies an odd symmetry, i.e.,

$$\hat{g}_{\mathcal{A}}(-y) = -\hat{g}_{\mathcal{A}}(y) \quad \text{for} \quad \forall y. \tag{4.6}$$

Definition 4.1 (Symmetric LLR approximation)**.** Any LLR approximation which satisfies (4.6) is a symmetric LLR approximation.

Accuracy measure

Assuming equally likely inputs, the capacity of a memoryless binary-input symmetric output (MBISO) channel can be given via the pdf $f_L(l)$ of the channel LLR by [48,49]

$$C = 1 - E_L[\log_2(1+e^{-L})] = 1 - \int_{-\infty}^{\infty} \log_2(1+e^{-l}) f_L(l) dl. \tag{4.7}$$

The above equation is only valid for MBISO channels where the LLR pdf satisfies the consistency condition (i.e., $f_L(-l) = e^{-l} f_L(l)$) [8, 39]. This equation, however, cannot be used with approximate LLRs, since they do not necessarily satisfy the consistency condition.

We prove that if instead of the pdf of true LLRs, the pdf of some symmetric LLR approximation is used in (4.7), the value of C is reduced. By defining $f_{\hat{L}}(l)$ as the pdf of approximate LLRs, and

$$\hat{C} = 1 - E_{\hat{L}}[\log_2(1+e^{-\hat{L}})] = 1 - \int_{-\infty}^{\infty} \log_2(1+e^{-l}) f_{\hat{L}}(l) dl, \tag{4.8}$$

we have the following theorem.

Theorem 4.1. *The maximum of $\hat{C}$ in (4.8) is equal to C which is achieved by true LLRs (no symmetric LLR approximation can result in $\hat{C} > C$).*

The proof of this theorem is given in Appendix A. It is also shown that as the approximate LLRs become less accurate, $\Delta C = C - \hat{C}$ gets larger. Thus, $\hat{C}$ acts as a measure of the accuracy of the approximate LLRs and the goal is to maximize $\hat{C}$.

4.2.2 LLR accuracy measure for asymmetric binary-input channels

As previously stated, the equivalent bit-channels of the BICM scheme are normally asymmetric. As a result, the LLR accuracy measure of (4.8) cannot be applied. Here, we generalize this measure to asymmetric channels. To this end, we consider the pdfs of the ith bit-channel LLR conditioned on the transmitted bit $b \in \{0, 1\}$, defined as $f^b_{L^{(i)}}(l) = E_{\boldsymbol{x} \in \mathcal{X}_b^{(i)}}[f_{L^{(i)}|\boldsymbol{X}}(l|\boldsymbol{x})]$. Using these LLR pdfs, it is possible to calculate the capacity of each asymmetric bit-channel and thus the BICM scheme. By capacity, we mean the mutual information between the input and output of each bit-channel when its input b is equally likely 0 or 1. The capacity of the ith bit-channel is given by [50]

$$C^{(i)} = 1 - \frac{1}{2}\int \log_2(1+e^{-l}) f^0_{L^{(i)}}(l) dl - \frac{1}{2}\int \log_2(1+e^{l}) f^1_{L^{(i)}}(l) dl. \tag{4.9}$$

Thus, the capacity of the BICM is found by $C = \sum_{i=1}^{m} C^{(i)}$.

When instead of the true LLRs, approximate LLRs $\hat{l}^{(i)}$ are used, their pdf is given by $f^{b}_{\hat{L}^{(i)}}(l) = E_{\boldsymbol{x}\in\mathcal{X}_b^{(i)}}[f_{\hat{L}^{(i)}|\boldsymbol{X}}(l|\boldsymbol{x})]$. Similar to (4.8), by inserting these approximate LLR pdfs in (4.9) we get

$$\hat{C} = \sum_{i=1}^{m} \hat{C}^{(i)} = \sum_{i=1}^{m} \left(1 - \frac{1}{2}\int \log_2(1+e^{-l}) f^{0}_{\hat{L}^{(i)}}(l)dl \right. \\ \left. - \frac{1}{2}\int \log_2(1+e^{l}) f^{1}_{\hat{L}^{(i)}}(l)dl \right). \tag{4.10}$$

The following theorem, similar to Theorem 4.1, shows that $\hat{C}$ can be used as an LLR accuracy measure for the BICM scheme.

Theorem 4.2. *The maximum of $\hat{C}$ in (4.10) is equal to C which is achieved by true LLRs (no LLR approximation can result in $\hat{C} > C$).*

The proof is given in Appendix B. Notice that for a symmetric channel (i.e., $f^{1}_{L^{(i)}}(l) = e^{-l} f^{0}_{L^{(i)}}(l)$), (4.10) reduces to the (4.8). Again using similar arguments as the previous theorem, it can be shown that as the approximate LLRs become less accurate, $\Delta C = C - \hat{C}$ gets larger. Thus, $\hat{C}$ is a measure of the accuracy of the approximate bit LLRs. In other words, good approximate LLRs can be found by maximizing $\hat{C}$.

While $\hat{C}$ does not represent the capacity under the approximate LLR calculation, a close connection exists between (4.10) and the generalized mutual information (GMI) of BICM [51]. The GMI is proved to be an achievable rate under mismatched decoding and the random coding regime [52]. The GMI of the BICM is given by [51]

$$\begin{aligned} I_{\text{gmi}} &= \sup_{s>0} I_{\text{gmi}}(s) \\ &= \sup_{s>0} \sum_{i=1}^{m} E\left[\log \frac{q^{(i)}(B^{(i)}, \boldsymbol{Y})^s}{\frac{1}{2}\left(q^{(i)}(0,\boldsymbol{Y})^s + q^{(i)}(1,\boldsymbol{Y})^s\right)} \right], \end{aligned}$$

where $B^{(i)} \in \{0,1\}$ denotes the ith bit and $q^{(i)}(B^{(i)}, \boldsymbol{Y})$ denotes the decoding metric for the ith bit. Also, the expectation is taken with respect to the joint distribution of $B^{(i)}$ and $\boldsymbol{Y}$. Noticing that the approximate bit LLR can be written in terms of mismatched decoding metrics as $\hat{l}^{(i)} = \log \frac{q^{(i)}(b^{(i)}=0,\boldsymbol{y})}{q^{(i)}(b^{(i)}=1,\boldsymbol{y})}$, it can be shown that $I_{\text{gmi}}(s=1)$ is equal to $\hat{C}$ in (4.10). In other words, $\hat{C}$ represents an achievable rate of the BICM operating under the approximate LLRs $\hat{l}^{(i)}$. As a result, maximizing $\hat{C}$ is also meaningful with regard to increasing the achievable transmission rate.

4.3 Finding approximate LLRs

The procedure of finding good approximate LLRs using $\hat{C}$ as the accuracy measure is as follows. Since bit-channels are independent, we maximize each $\hat{C}^{(i)}$ individually. Thus, for each bit-channel i, assuming a class of approximating functions $\hat{g}^{(i)}_{\mathcal{A}_i}(\boldsymbol{y})$, we find:

$$\mathcal{A}_i^{\text{opt}} = \arg\max_{\substack{\mathcal{A}_i \\ \text{s.t. } \boldsymbol{\Phi}_i(\mathcal{A}_i)=0}} \hat{C}^{(i)}, \tag{4.11}$$

where $\boldsymbol{\Phi}_i(\mathcal{A}_i) = 0$ denotes some constraints imposed on $\mathcal{A}_i$ (e.g., to preserve continuity).

4.3.1 Choosing LLR approximating functions

It is evident from (4.2) and (4.3) that true LLR functions depend on the signal set $\mathcal{X}$ and the labeling. Thus, choosing appropriate class of approximating functions also depends on $\mathcal{X}$ and the labeling. Here, we consider non-binary amplitude modulation (AM) and rectangular quadrature AM (QAM) with Gray labeling[2]. Our optimization approach, however, is general.

Moreover, we put our focus on piecewise linear approximate LLRs. Clearly, the optimization problem (4.11) can be solved for any other approximating function. Using piecewise linear approximations has benefits such as simplicity of demodulator and (as will be shown) convexity of the optimization problem. Numerical results verify that the obtained performance is also very close to true LLRs.

By viewing a complex variable as a two-dimensional vector denoted by $\boldsymbol{y} = (\text{Re}\{\boldsymbol{y}\}, \text{Im}\{\boldsymbol{y}\})$, a piecewise linear function of a complex variable is defined as follows. First, the complex domain $\mathbb{C}$ is divided into a finite number of regions $\mathbb{C}_1, \mathbb{C}_2, \ldots, \mathbb{C}_N$ by a finite number of one-dimensional boundaries. Then the function is represented by $f(\boldsymbol{y}) = \langle \boldsymbol{\alpha}_k, \boldsymbol{y} \rangle + \beta_k$ for any $\boldsymbol{y} \in \mathbb{C}_k$, where $\langle , \rangle$ denotes the inner product of two-dimensional vectors, i.e., $\langle \boldsymbol{\alpha}_k, \boldsymbol{y} \rangle = \text{Re}\{\boldsymbol{a}_k\}\text{Re}\{\boldsymbol{y}\} + \text{Im}\{\boldsymbol{\alpha}_k\}\text{Im}\{\boldsymbol{y}\}$. Thus,

$$\hat{l}^{(i)} = \hat{g}^{(i)}_{\mathcal{A}_i}(\boldsymbol{y}) = \sum_{k=1}^{N^{(i)}} \left(\langle \boldsymbol{\alpha}_k^{(i)}, \boldsymbol{y} \rangle + \beta_k^{(i)} \right) \mathbf{1}_{(\boldsymbol{y} \in \mathbb{C}_k^{(i)})}, \tag{4.12}$$

[2]The Gray labeling is chosen since it leads to some symmetries in the LLR calculating functions reducing the complexity of LLR calculation [14]. It also plays a key role in BICM by improving both the capacity and BER [31].

where $N^{(i)}$ is the number of segments of the piecewise linear function, $\mathcal{A}_i$ is the set of all $\boldsymbol{\alpha}_k^{(i)}$, $\beta_k^{(i)}$, and $\mathbf{1}_{(\cdot)}$ is the indicator function. The parameters are chosen to preserve continuity over $\boldsymbol{y}$.

The following theorem, proved in Appendix C, states that optimizing a piecewise linear approximating function according to (4.11) is a convex optimization problem.

Theorem 4.3. *Assuming that approximate bit LLRs are calculated by (4.12) for $i = 1, \ldots, m$, and assuming fixed $\mathbb{C}_k^{(i)}$ for $k = 1, \ldots, N^{(i)}$, $\hat{C}^{(i)}$ is a concave function of $\boldsymbol{\alpha}_k^{(i)}$ and $\beta_k^{(i)}$ for all k.*

Using (4.12), (4.11) can be numerically solved as follows. For a given SNR, and assuming fixed $\mathbb{C}_k^{(i)}$'s, $\hat{C}^{(i)}$ can be computed by first computing $f^0_{\hat{L}^{(i)}}(l)$ and $f^1_{\hat{L}^{(i)}}(l)$ for given $\boldsymbol{\alpha}_k^{(i)}$'s and $\beta_k^{(i)}$'s and inserting them in (4.10). Since $\hat{C}^{(i)}$ is a concave function of $\boldsymbol{\alpha}_k^{(i)}$'s and $\beta_k^{(i)}$'s and the constraints are linear, maximizing $\hat{C}^{(i)}$ can be done efficiently using numerical optimization techniques. The proper number of regions $N^{(i)}$ is selected based on the affordable complexity and the curve of true LLRs. Optimizing the regions can be done through search. As will be seen in the next section, usually the size of the parameter sets is small and symmetry further reduces the number of unknown parameters. It is also worth mentioning that this optimization is performed off-line thus the complexity limitations may not be crucial.

4.4 Numerical results and examples

Now, we describe the proposed method through examples of binary, non-binary, real, and complex signal constellations. We use LDPC codes decoded by the sum-product algorithm.

Example 4.1. Consider half-rate $\mathcal{C}^{\infty}(x^2, x^5)$ LDPC codes with BPSK modulation on an uncorrelated normalized Rayleigh fading channel ($f_R(r) = 2re^{-r^2}$). Considering the shape of true LLRs and since there is only one bit, we propose a linear LLR approximation of the form

$$\hat{l} = \hat{g}_{\mathcal{A}}(y) = \alpha \cdot y, \tag{4.13}$$

Here, we calculate different values for α and compare their corresponding decoding threshold. The results are given in Table 4.1 where α_{exh} corresponds to the exhaustive search method using density evolution, $\alpha_{\text{r}} = 2E[r]/\sigma^2$ is the method of [40, 41] which is given by the MMSE of r, and $\alpha_{\hat{C}}$ is calculated based on (4.11).

	$\alpha_{\mathrm{r}} = 4.513$	$\alpha_{\hat{C}} = 2.957$	$\alpha_{\mathrm{exh}} = 2.957$
σ^*	0.6266	0.6449	0.6449
$\frac{E_\mathrm{b}}{N_0}^*$ (dB)	4.06	3.81	3.81

Table 4.1: Comparison between the achieved threshold for $\mathcal{C}^{\infty}(x^2, x^5)$ LDPC codes using different linear LLR calculations.

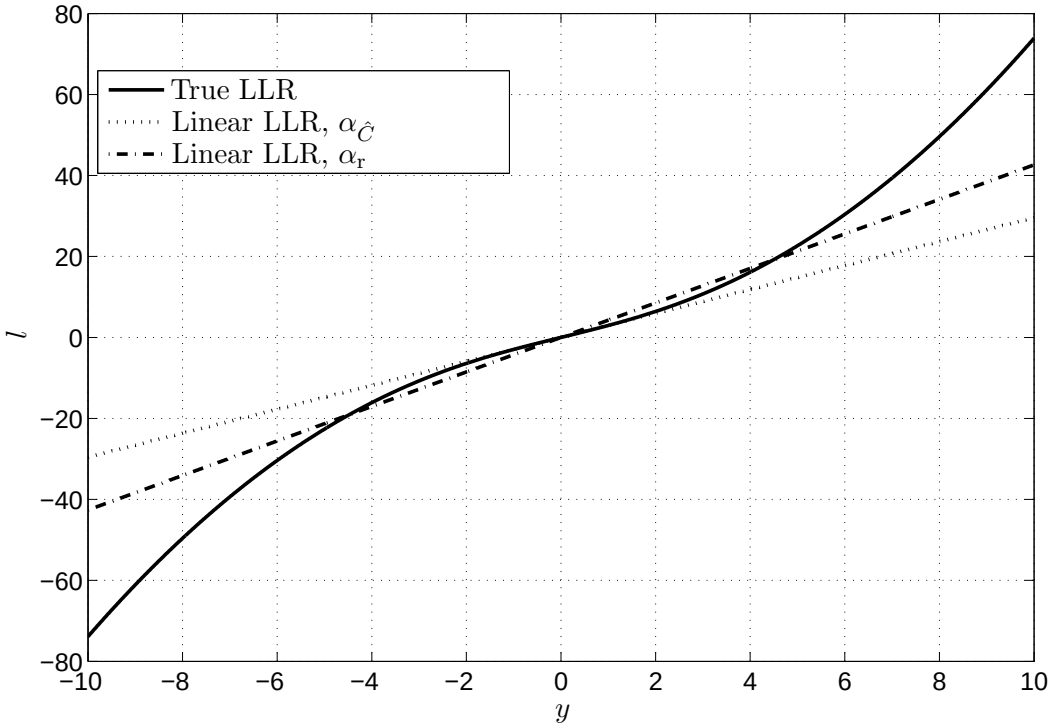


Figure 4.3: Comparison of exact LLRs and linear approximations for a normalized Rayleigh fading channel with $\sigma = 0.6449$ and no CSI available at the receiver. It is seen that the linear approximation with $\alpha_{\hat{C}}$ gives a nearly perfect approximation when $|y|$ is small.

It can be seen that the best achieved threshold is 3.81 dB given by $\alpha = 2.957$. Interestingly, the optimum value of α which is calculated using exhaustive search and density evolution is equal to the $\alpha_{\hat{C}}$ given by our method.

In Fig. 4.3, the exact LLR values obtained from (4.3) are compared to the linear approximation (4.13) with $\alpha_{\hat{C}}$ and linear approximation with α_{r}.

To show that our LLR approximation also improves the BER of the code, a randomly constructed $\mathcal{C}^{104}(x^2, x^5)$ LDPC code is simulated on an uncorrelated normalized Rayleigh fading channel. Fig. 4.4 shows the BER of the code with and without CSI at the receiver. When CSI is not available and σ is known, three cases have been plotted. One under linear LLR approximation with $\alpha = \alpha_{\mathrm{r}}$, one with $\alpha = \alpha_{\hat{C}}$, and one under true LLR calculation. The figure shows considerable BER

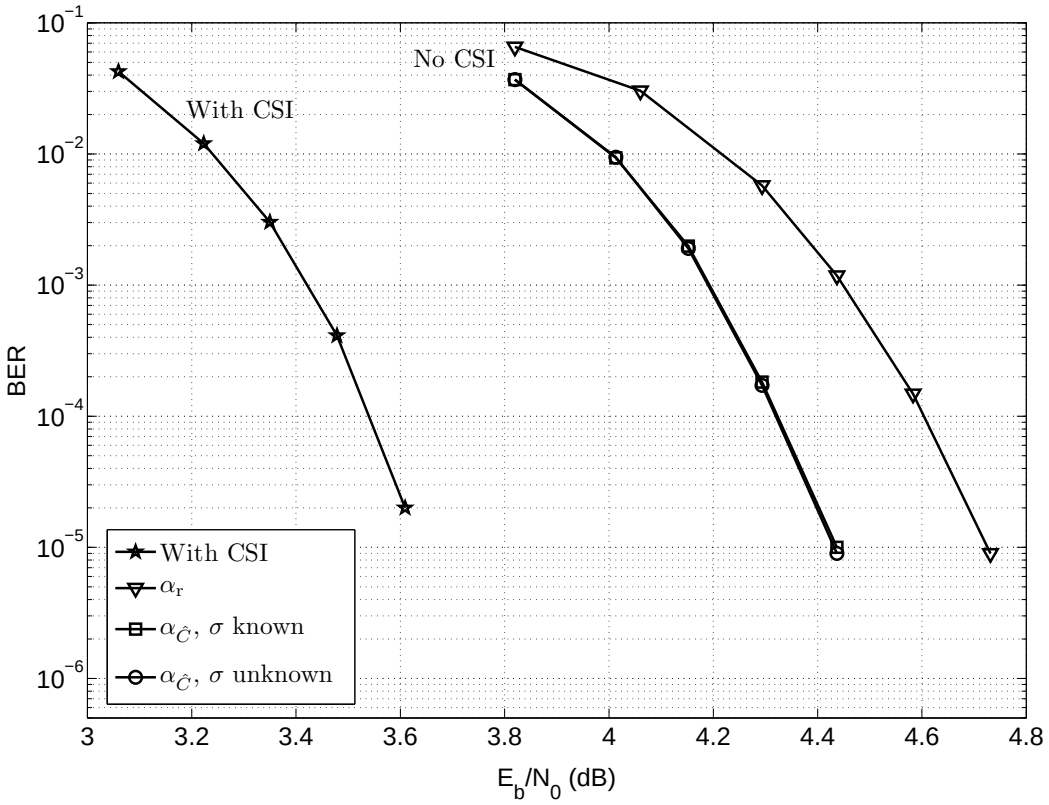


Figure 4.4: Comparison of BER for a $\mathcal{C}^{10^4}(x^2, x^5)$ LDPC code in different cases on a normalized Rayleigh fading channel. The performance under $\alpha_{\hat{C}}$ remains almost the same regardless of whether σ is known or not and is extremely close to the performance of true LLR calculation.

improvement under $\alpha_{\hat{C}}$ comparing to that of $\alpha_{\rm r}$. This improvement is about 0.3 dB at BER of 10^{-5}. Furthermore, MCLA shows a minor extra gap (less than 0.02 dB) compared to true LLR calculation. It is worth mentioning that BER improvement increases when higher rate codes are used, e.g., about 0.75 dB improvement when $\mathcal{C}^{10^4}(x^2, x^{15})$ LDPC of rate 0.75 is used. It should be further noted that the linear approximation with $\alpha_{\hat{C}}$ also outperforms those of $\alpha^{\rm MMSE}$ and $\alpha^{\rm D}$. These measures, however, are not reported in here because they defy the purpose of avoiding true LLR calculation. □

Example 4.2. Now consider 8-AM constellation with Gray labeling shown in Fig. 4.5 on the normalized Rician fading channel. On this channel,

$$f_R(r) = 2r(K+1)e^{-(K+(K+1)r^2)}I_0(2r\sqrt{K(K+1)}),$$

where K is the Rician K-factor, and $I_0(\cdot)$ is the zero-order modified Bessel function of the first kind. The Rician fading model is considered here because by changing

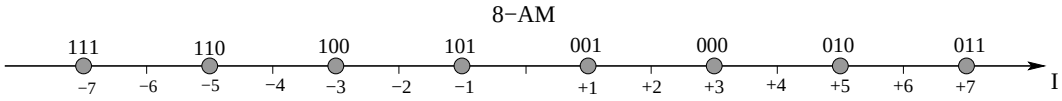


Figure 4.5: The 8-AM constellation points with Gray mapping.

K it can easily be transformed to Rayleigh fading ($K = 0$) and Gaussian models ($K \to \infty$).

Using (4.2) and (4.3), true LLRs are calculated for $i = 1, 2, 3$, and they have been plotted versus y for extreme values of K in Fig. 4.5. Considering the general model of (4.12), and the symmetry in the true LLR functions, we propose the following piecewise linear LLR approximations

$$\hat{l}^{(1)} = \hat{g}^{(1)}_{\mathcal{A}_1}(y) = \alpha^{(1)}_1 y, \tag{4.14}$$

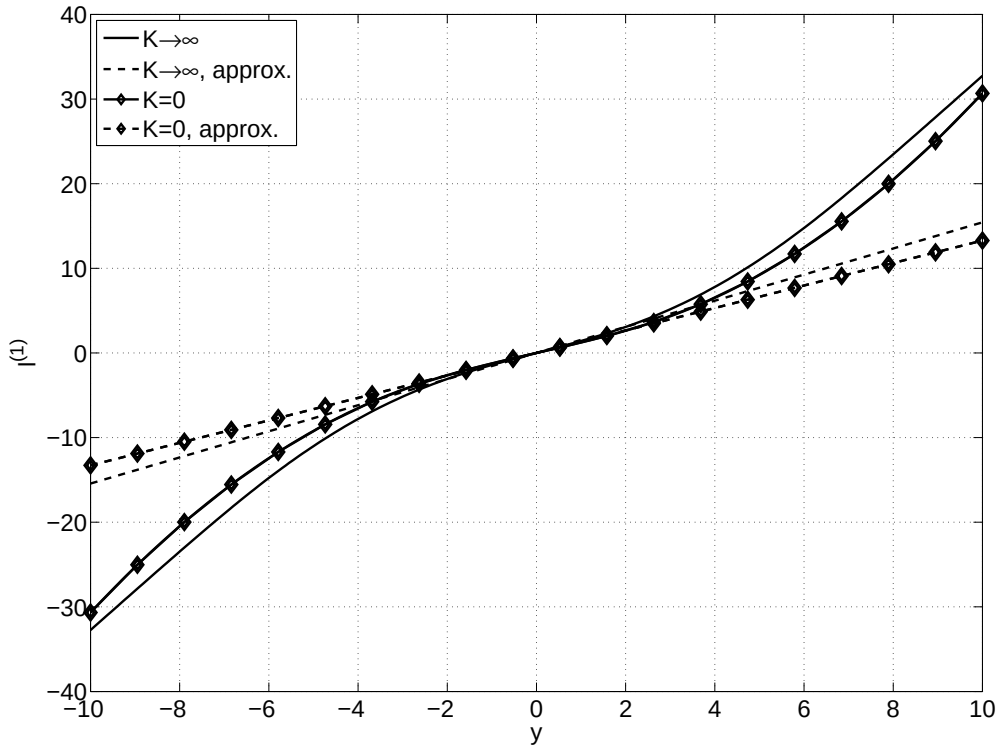
$$\begin{aligned}\hat{l}^{(2)} = \hat{g}^{(2)}_{\mathcal{A}_2}(y) &= (\alpha^{(2)}_1 y + \beta^{(2)}_1)\mathbf{1}_{(y\le 0)} + (\alpha^{(2)}_2 y + \beta^{(2)}_2)\mathbf{1}_{(0<y)} \\ &= -\alpha^{(2)}_1 |y| + \beta^{(2)}_1,\end{aligned} \tag{4.15}$$

$$\begin{aligned}\hat{l}^{(3)} = \hat{g}^{(3)}_{\mathcal{A}_3}(y) &= (\alpha^{(3)}_1 y + \beta^{(3)}_1)\mathbf{1}_{(y\le \gamma^{(3)}_1)} + (\alpha^{(3)}_2 y + \beta^{(3)}_2)\mathbf{1}_{(\gamma^{(3)}_2 < y \le 0)} \\ &\quad + (\alpha^{(3)}_3 y + \beta^{(3)}_3)\mathbf{1}_{(0<y\le \gamma^{(3)}_3)} + (\alpha^{(3)}_4 y + \beta^{(3)}_4)\mathbf{1}_{(\gamma^{(3)}_4 < y)},\end{aligned} \tag{4.16}$$

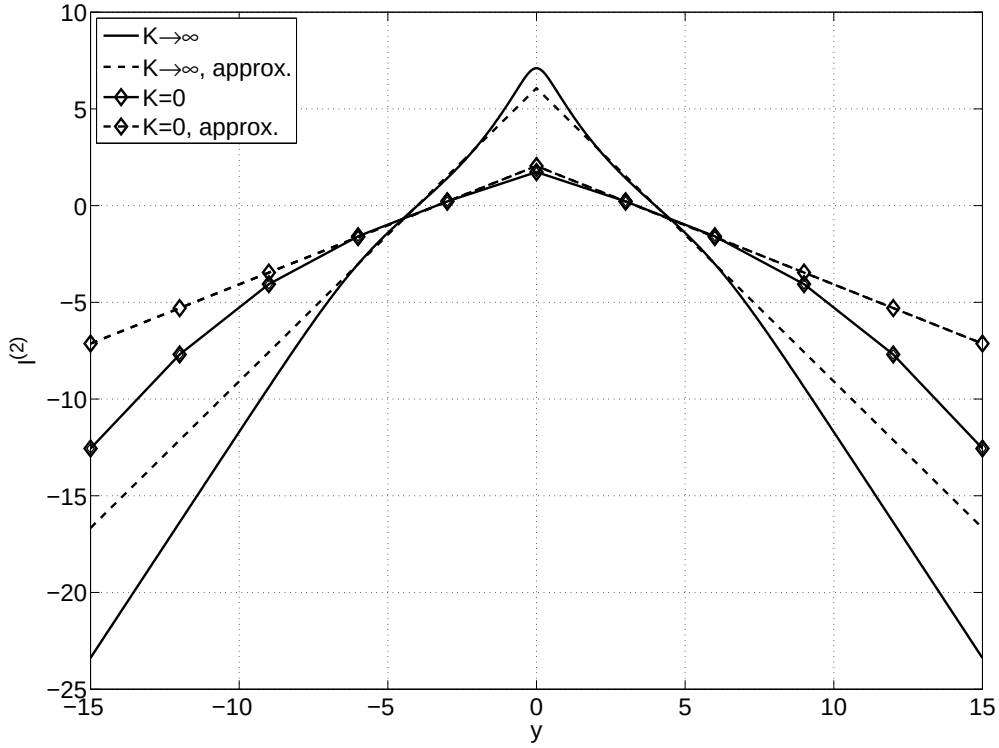
where due to the symmetry of the LLRs, we have assumed in (4.15) that $\alpha^{(2)}_2 = -\alpha^{(2)}_1$ and $\beta^{(2)}_2 = \beta^{(2)}_1$. Also, in (4.16), we have $\alpha^{(3)}_1 = -\alpha^{(3)}_4$, $\alpha^{(3)}_2 = -\alpha^{(3)}_3$, $\beta^{(3)}_1 = \beta^{(3)}_4$, $\beta^{(3)}_2 = \beta^{(3)}_3$, and $\gamma^{(3)}_1 = \gamma^{(3)}_2 = -\gamma^{(3)}_3 = -\gamma^{(3)}_4$. Thus, $\mathcal{A}_1 = \{\alpha^{(1)}_1\}$, $\mathcal{A}_2 = \{\alpha^{(2)}_1, \beta^{(2)}_1\}$, and $\mathcal{A}_3 = \{\alpha^{(3)}_1, \alpha^{(3)}_2, \beta^{(3)}_1, \beta^{(3)}_2, \gamma^{(3)}_1\}$. It is evident that symmetry reduces the number of unknown parameters. We also impose $\Phi_3(\mathcal{A}_3) = \gamma^{(3)}_1(\alpha^{(3)}_1 - \alpha^{(3)}_2) + \beta^{(3)}_1 - \beta^{(3)}_2 = 0$ to preserve continuity in (4.16).

For a given SNR, we optimize the parameter sets $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}_3$, by solving (4.11). Numerical results confirm that $\hat{C}_{\max} = \sum_{i=1}^{m} \max_{\mathcal{A}_i} \hat{C}^{(i)}$ is always very close to the capacity of BICM employing true LLRs, i.e., C. For example, for $K = 0$, $\hat{C}_{\max} = 0.851$ and $C = 0.855$ bits per channel use at SNR= 5.00 dB, and $\hat{C}_{\max} = 1.544$ and $C = 1.553$ bits per channel use at SNR= 30.00 dB. When K increases, $\Delta C = C - \hat{C}_{\max}$ becomes even smaller. Fig. 4.6 compares C and $\hat{C}_{\max}$ for extreme values of K.

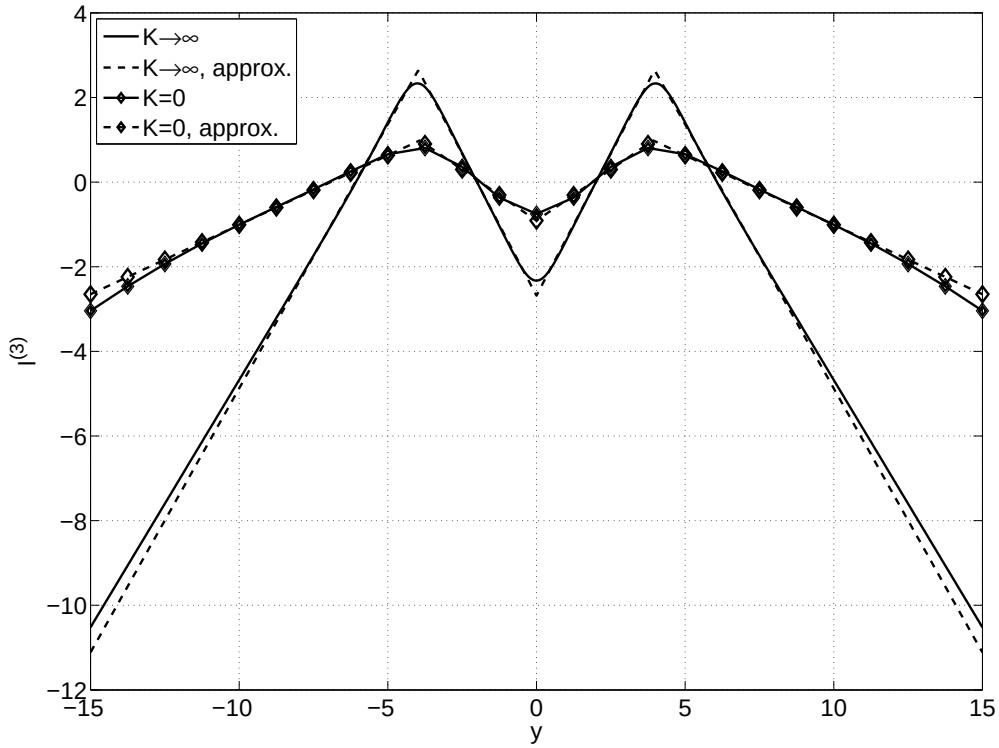
To evaluate the decoding performance of the LDPC-coded BICM system under optimized piecewise linear approximations, we compare the decoding threshold of LDPC codes and their BER under approximate and true LLRs. The decoding threshold can be found by density evolution [8, 27] and by using the technique of



(a) $i = 1$



(b) $i = 2$



(c) $i = 3$

Figure 4.5: True bit LLR values $l^{(i)}$ $(i = 1, 2, 3)$ as functions of the channel output y for the 8-AM at SNR$= 7.88$ dB. Also, the optimized piecewise linear LLR approximations are depicted.

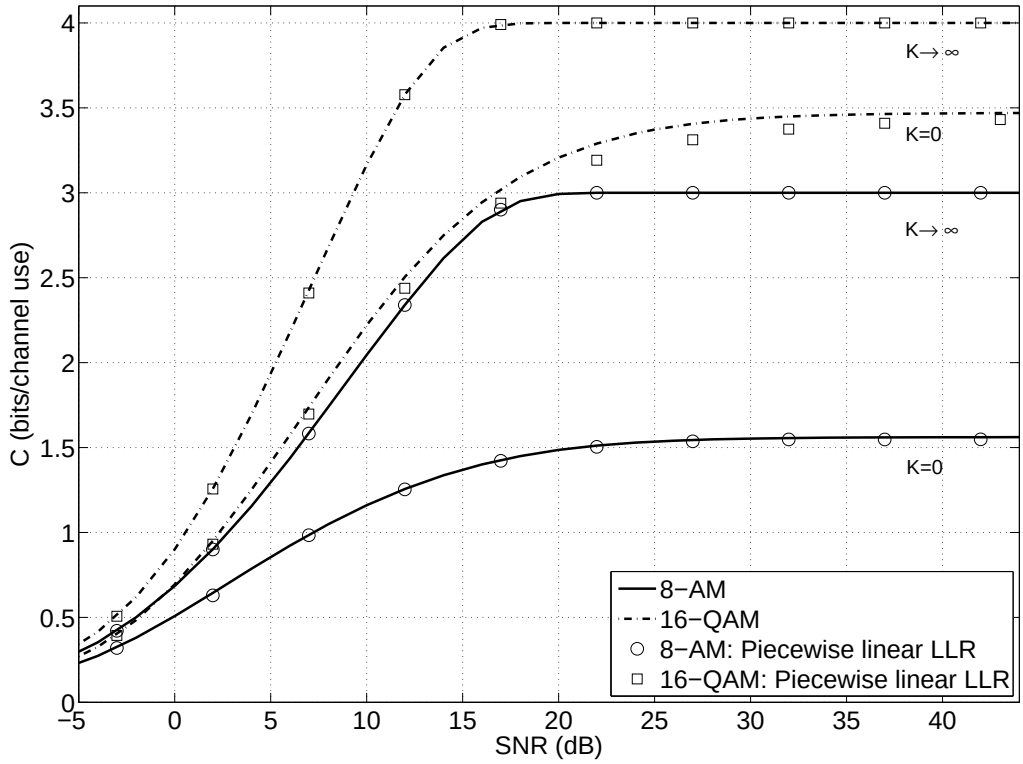


Figure 4.6: Comparison between the capacity of BICM C and $\hat{C}_{\max}$ under optimized piecewise linear LLR approximation. It is seen that $\Delta C = C - \hat{C}_{\max}$ is always very small. As K increases, $\Delta C = C - \hat{C}_{\max}$ becomes smaller.

8-AM			
SNR	bit 1	bit 2	bit 3
7.88 dB	$\alpha_1^{(1)} = 1.328$	$\alpha_1^{(2)} = 0.612$ $\beta_1^{(2)} = 2.046$	$\alpha_1^{(3)} = 0.328$ $\beta_1^{(3)} = 2.273$ $\alpha_2^{(3)} = -0.482$ $\beta_2^{(3)} = -0.909$ $\gamma_1^{(3)} = -3.928$
21.02 dB	$\alpha_1^{(1)} = 8.538$	$\alpha_1^{(2)} = 0.825$ $\beta_1^{(2)} = 3.098$	$\alpha_1^{(3)} = 0.384$ $\beta_1^{(3)} = 2.513$ $\alpha_2^{(3)} = -1.528$ $\beta_2^{(3)} = -2.357$ $\gamma_1^{(3)} = -2.547$

16-QAM		
SNR	bit 1	bit 2
5.02 dB	$\alpha_1^{(1)} = 1.262$	$\boldsymbol{\alpha}_1^{(2)} = (0.868, -0.200)$ $\beta_1^{(2)} = -1.257$

Table 4.2: Optimized piecewise linear LLR parameters at different SNRs for 8-AM and 16-QAM when $K = 0$.

i.i.d. channel adapters [47] which provides the required symmetry conditions.

As an example, consider $(3, 4)$-regular LDPC codes on the normalized Rayleigh channel (equivalent to $K = 0$) with 8-AM signalling of Fig. 4.5. By using the approximating functions of (4.14)–(4.16), we find the decoding threshold of the code under optimized LLR parameters reported in Table 4.2. The decoding threshold given by density evolution is 7.88 dB while under true LLR calculation of (4.3), the decoding threshold is 7.85 dB showing only a 0.03 dB performance gap.

To see how the piecewise linear LLR calculation affects the BER performance, we simulate a given LDPC code on the normalized Rayleigh fading channel. In Fig. 4.8, the performance of a randomly constructed $(3, 4)$-regular LDPC code of length 15000 is depicted in two cases: once decoded using true LLRs of (4.3), and once with the piecewise linear approximation of (4.14)–(4.16) and the optimized parameters reported in Table 4.2. It should be noted that the parameters are optimized once at the decoding threshold and are kept fixed at the receiver for other SNRs[3]. It is seen that the performance of the optimized approximate LLRs is almost identical to that of the more complex true LLRs although parameters are only optimized at the decoding threshold.

[3]This suggests that the performance loss is negligible even when the noise variance σ^2 is also unknown at the receiver. Our method can be easily extended to cases where noise power is also unknown. Interested reader can refer to [12, 14] for details.

Also, to show that optimizing $\hat{C}$ is meaningful in terms of the maximum transmission rate achievable by the piecewise linear LLRs, we optimize the degree distributions of LDPC codes under our approximate LLRs. At SNR = 21.02 dB, the capacity of BICM under true LLRs is $C = 1.500$ in the absence of CSI when $K = 0$. Since $m = 3$, then the maximum binary code rate achievable on this channel is 0.500. At this SNR, solving (4.11) gives $\hat{C}_{\max} = 1.493$ and the parameters reported in Table 4.2. Now, by using the designed piecewise linear approximation, assuming a fixed check node degree of 8 and maximum variable node degree of 30, an irregular LDPC code is optimized. The optimization process is done by linear programming together with density evolution [53,54]. The variable node degree distribution of the optimized code is $\lambda(x) = 0.250x + 0.217x^2 + 0.221x^6 + 0.048x^7 + 0.119x^{22} + 0.145x^{29}$, and the code rate is $R = 0.490$. Thus, the proposed approximate LLRs can achieve rates very close to the capacity of BICM under true LLRs. □

Example 4.3. Now consider a 16-QAM constellation with Gray labeling as depicted in Fig. 4.7. Using the general piecewise linear model of (4.12), due to the symmetry and the similarity of the bit LLR functions, we propose the following LLR approximations:

$$\hat{l}^{(1)} = \hat{g}^{(1)}_{\mathcal{A}_1}(\boldsymbol{y}) = \alpha^{(1)}_1 \mathrm{Re}\{\boldsymbol{y}\}, \tag{4.17}$$

$$\begin{aligned}\hat{l}^{(2)} &= \hat{g}^{(2)}_{\mathcal{A}_2}(\boldsymbol{y}) = \sum_{k=1}^{4} \left(\langle \boldsymbol{\alpha}^{(2)}_k, \boldsymbol{y} \rangle + \beta^{(2)}_k \right) \mathbf{1}_{(\boldsymbol{y} \in \mathbb{C}^{(2)}_k)} \\ &= \mathrm{Re}\{\boldsymbol{\alpha}^{(2)}_1\} |\mathrm{Re}\{\boldsymbol{y}\}| + \mathrm{Im}\{\boldsymbol{\alpha}^{(2)}_1\} |\mathrm{Im}\{\boldsymbol{y}\}| + \beta^{(2)}_1, \end{aligned} \tag{4.18}$$

where $\mathbb{C}^{(2)}_1, \ldots, \mathbb{C}^{(2)}_4$ are the four quadrants of the complex plane. It should be noted that bit-channel LLR calculations are similar for bit 1 and 3, and for bit 2 and 4 except that the real and imaginary parts of $\boldsymbol{y}$ are swapped, i.e., $\hat{l}^{(3)} = \alpha^{(1)}_1 \mathrm{Im}\{\boldsymbol{y}\}$ and $\hat{l}^{(4)} = \mathrm{Re}\{\boldsymbol{\alpha}^{(2)}_1\} |\mathrm{Im}\{\boldsymbol{y}\}| + \mathrm{Im}\{\boldsymbol{\alpha}^{(2)}_1\} |\mathrm{Re}\{\boldsymbol{y}\}| + \beta^{(2)}_1$. Thus, it is enough to optimize $\mathcal{A}_1 = \{\alpha^{(1)}_1\}$ and $\mathcal{A}_2 = \{\boldsymbol{\alpha}^{(2)}_1, \beta^{(2)}_1\}$.

Again numerical results suggest that the gap between $\hat{C}_{\max}$ and the true BICM capacity C is always small. For example, when $K = 0$, we have $\hat{C}_{\max} = 1.074$ and $C = 1.097$ bits per channel use at SNR= 3.00 dB. Fig. 4.6 depicts the comparison of C and $\hat{C}_{\max}$ at various SNRs for extreme values of K.

For $(3, 4)$-regular LDPC codes, density evolution gives a decoding threshold of 5.02 dB under approximate LLRs with the optimized parameters of Table 4.2. Under true LLR calculation, the decoding threshold is 4.83 dB. As a result, approximate

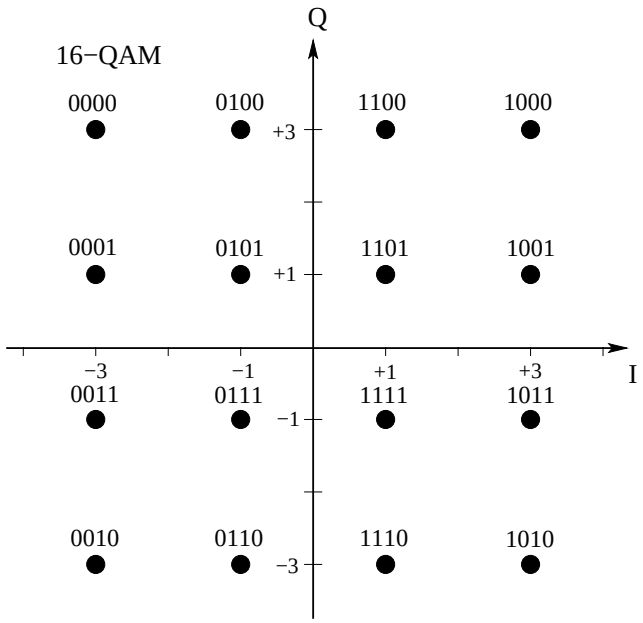


Figure 4.7: The 16-QAM constellation points with Gray mapping.

LLRs show about 0.19 dB performance gap to true LLRs. The BER comparison is depicted in Fig. 4.8. It is worth mentioning that this gap can be further reduced by proposing piecewise linear LLRs with more segments. □

4.5 Conclusion

LLR computation is an important issue when soft iterative decoding is used. LLR computation is generally complicated especially for equivalent bit-channels of a non-binary modulation and on fading channels, when the channel gain is unknown. We discussed designing good approximate LLR calculating functions by first justifying the previously proposed LLR accuracy measure for MBISO channels. Next, noticing that the equivalent bit channels were asymmetric, in order to find good approximate LLRs, we generalized the proposed LLR accuracy measure to memoryless binary-input asymmetric-output channels. This accuracy measure can be used to optimize the parameters of any approximating function. We used our accuracy measure to optimize piecewise linear LLR approximations. By using LDPC-coded BICM, we showed that the performance loss under the optimized piecewise linear approximation was very small. We also showed that under approximate LLRs, asymptotic irregular LDPC codes having rates very close to the capacity of BICM under true LLRs could be obtained. Our solution can also be applied to other coding schemes

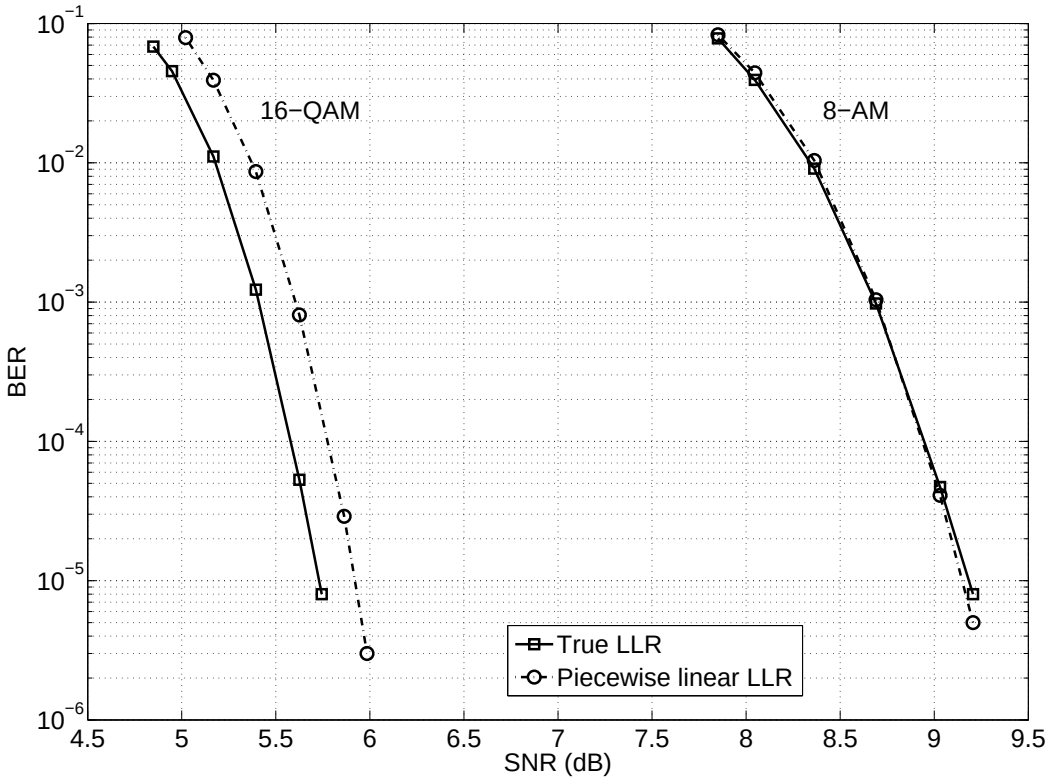


Figure 4.8: Comparison between the BER of a randomly constructed $(3, 4)$-regular LDPC code of length 15000 decoded by true and approximate LLRs on the Rayleigh fading channel ($K = 0$). The approximate LLR parameters are reported in Table 4.2.

which use LLRs such as the convolutional and turbo codes.

Chapter 5

Practical Coding for Channels with Imperfect Timing at the Receiver

5.1 Introduction

Since the seminal work of Shannon [15], there have been huge advancements in coding and information theory. The fundamental limits and efficient coding solutions approaching these limits are now known for many communication channels. However, in the vast majority of coding schemes invented, it is assumed that the receiver is perfectly synchronized with the transmitter, i.e., the symbol arrival times are known at the receiver. In most communication systems, however, achieving perfect synchronization is not possible even with the existence of timing recovery systems.

When perfect synchronization does not exist, random symbol insertions and deletions (synchronization errors) occur in the received sequence. This phenomenon poses a great challenge for error correction. Since the positions of the inserted and deleted symbols are unknown at the receiver, even a single uncorrected insertion/deletion can result in a catastrophic burst of errors. Thus, conventional error-correcting codes fail at these situations.

Error-correcting codes designed for dealing with such insertion/deletion (I/D) channels are called *synchronization codes*. Synchronization codes have a long history but their design and analysis have proven to be extremely challenging, hence few practical results exist in the literature. Moreover, standard approaches do not lead to finding the optimal codebooks or tight bounds on the capacity of I/D channels and finding their capacity is still an open problem [16].

The first synchronization code was proposed by Sellers in 1962 [55]. He inserted *marker* sequences in the transmitted bitstream to achieve synchronization. Long markers allowed the decoder to correct multiple insertion or deletion errors but greatly increased the overhead. In 1966, using number-theoretic techniques, Levenshtein constructed binary codes capable of correcting a single insertion or deletion assuming that the codeword boundaries were known at the decoder [56]. Most subsequent work were inspired by the number-theoretic methods used by Levenshtein, e.g., see [57–60]. Unfortunately, these constructions either cannot be generalized to correct multiple synchronization errors without a significant loss in rate, do not scale well for large block lengths, or lack practical and efficient encoding or decoding algorithms.

Some authors also generalized these number-theoretic methods to non-binary alphabets and constructed non-binary synchronization codes [4,61–64]. Following [64], *perfect* deletion-correcting codes were studied and constructed using combinatorial approaches [65–67]. Most of these codes, however, are constructed using ad hoc techniques and no practical encoding and decoding algorithm is provided. Non-binary low-density parity-check (LDPC) codes decoded by a verification-based decoding algorithm are designed for deletion channels in [68]. Unfortunately, the decoding complexity of this construction is also far from being practical.

There are also some works in the literature on codes that can detect synchronization misalignments. While these codes are able to regain synchronization, they are not able to correct any insertion or deletion errors. The first type of these codes are *comma-free* codes introduced in [69]. Some of the proposed comma-free codes, e.g., see [70–72], can also correct substitution errors. However, none of these codes are able to recover insertion and deletion errors.

In a more recent work [73], the authors proposed a class of *prefix synchronized* codes—which are themselves a subclass of comma-free codes—able to correct a single insertion or deletion assuming the decoder knows the beginning of the received sequence.

The drawback of all the above-mentioned synchronization codes is that they only work under very stringent synchronization and noise restrictions such as working only on deletion channels, or a single synchronization error per block. Coding methods proposed for error-correction on the I/D channels working under more general conditions are usually based on concatenated coding schemes with two layers

of codes, i.e., an inner and an outer code [1, 2, 74–76]. The inner code identifies the positions of the synchronization errors and the outer code is responsible for correcting the insertions, deletions, and substitution errors as well as misidentified synchronization errors.

In the seminal work of Davey and MacKay [1], a practical concatenated coding method is presented for error-correction on general binary I/D channels. They have called their inner code, a *watermark* code. The main idea is to provide a carrier signal or watermark for the outer code. The synchronization errors are inferred by the outer code via identifying discontinuities in the carrier signal. In more detail, the data is first encoded by the outer code, i.e., an LDPC code. Then, the encoded data is transformed into a sparse string, added to the watermark string (which is a pseudo-random binary sequence known to both transmitter and receiver), and sent on the channel. The inner decoder which receives a distorted and noisy version of the watermark, provides soft information for the outer decoder by finding the correct alignment of the received string with the watermark. Next, this soft information is fed to the outer decoder and the original data is decoded.

One of the advantages of watermark codes is that the decoder does not need to know the block boundaries of the received sequence. However, due to the use of a sparsifier, rate loss is significant. The watermark is substituted by fixed and pseudo-random markers in [2] and is shown that it allows better rates but is only able to outperform the watermark codes at low synchronization error rates. Also, it has recently been shown that the performance of both marker codes and watermark codes can be improved by using symbol-level decoding instead of bit-level decoding [3, 77].

In this chapter, we consider the problem of devising an efficient coding method for reliable communication over non-binary I/D channels. On these channels, synchronization errors occur at the symbol level, i.e., symbols are randomly inserted in and deleted from the received sequence. We also assume that all symbols are corrupted by additive white Gaussian noise (AWGN)[1]. The use of this channel model is motivated by the fact that at the receiver the received continuous waveform is first sampled at certain time instances to produce the discrete symbol sequence required by the decoder. If the symbol arrival times are not perfectly known at the receiver,

[1]It should be noted that a binary I/D channel model with AWGN has been the subject of study in the literature (e.g., see [78])

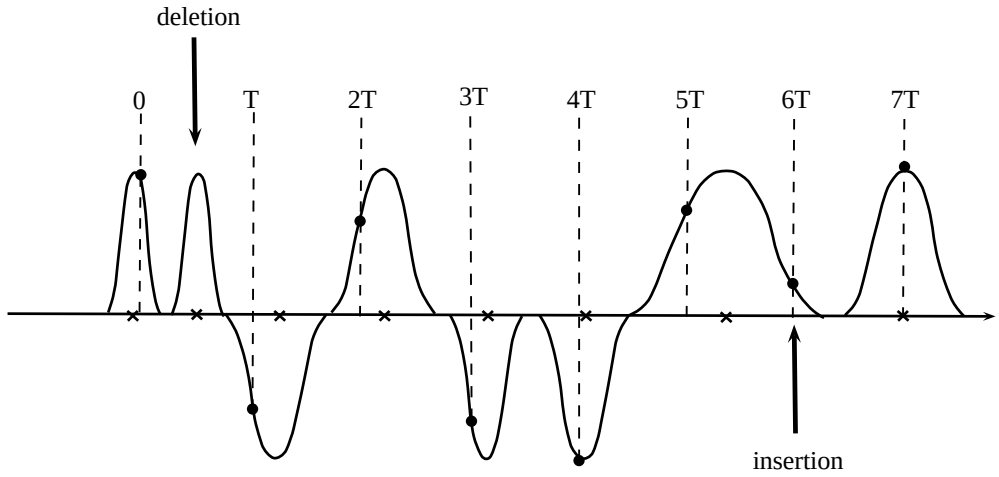


Figure 5.1: A continuous waveform is sampled at the receiver. Due to timing mismatch, there will be symbol deletions and insertions in the sampled sequence.

i.e., there is timing mismatch, some of the transmitted symbols are not sampled at all (symbol deletions) or sampled multiple times (symbol insertions) [79]. As a result, this channel model can be used to represent non-binary communications over the AWGN channel suffering from timing mismatch. This concept is depicted in Fig. 5.1. Most communication systems use non-binary signalling, where synchronization errors can result in insertion/deletion at the symbol level.

For the proposed channel model, we utilize the inherent redundancy that can be achieved in non-binary symbol sets by first expanding the symbol set and then allocating part of the bits associated with each symbol to watermark symbols. As a result, not all the available bits in the signal constellation are used for the transmission of information bits. In its simplest form, our solution can be viewed as a communication system using two different signal sets. The system switches between these two signal sets according to a binary watermark sequence. Since the watermark sequence is known both at the transmitter and the receiver, probabilistic decoding can be used to infer the insertions and deletions that occurred and to remove the effect of additive noise. In particular, the system is modeled by a hidden Markov model (HMM) [80] and the forward-backward algorithm [81] is used for decoding.

Our proposed scheme resembles trellis coded modulation (TCM) [30]. The main idea in both methods is to add redundancy by expanding the symbol set and limit the symbol transitions in a controlled manner. The proposed method is also closely related to the watermark codes of [1]. In both methods, decoding is done by the

aid of a watermark sequence which both the transmitter and receiver agree on. The difference is that the extra degree of freedom in non-binary sets allows us to separate information from the watermark.

Our proposed solution leads to significant system ability to detect and correct synchronization errors. For example, a rate 1/4 binary outer code is capable of correcting about $2,900$ insertion/deletion errors per block of $10,012$ symbols even when block boundaries are unknown at the receiver.

This chapter is organized as follows. In Sections 5.2 and 5.3, we state our proposed approach and describe the system model. Section 5.4 demonstrates the capabilities of the proposed solution by providing numerical results and discussions. Section 5.5 describes ways to increase the achievable information rates on the channel and Section 5.6 analyzes the system in terms of complexity, and practical considerations. Finally, Section 5.7 concludes the chapter.

5.2 Channel model and the proposed approach

Throughout this chapter, scalar quantities are shown by lower case symbols, complex quantities by boldface letters, and vectors by underlined symbols.

5.2.1 Channel model

The channel model we consider in this chapter is a non-binary I/D channel with AWGN where insertions and deletions occur at the symbol level. Similar to [1], it is assumed that the symbols from the input sequence $\underline{\boldsymbol{x}}$ first enter a queue before being transmitted. Then at each channel use, either a random symbol is inserted in the symbol sequence $\underline{\boldsymbol{x}}'$ with probability p_{i}, the next queued symbol is deleted with probability p_{d}, or the next queued symbol is transmitted (put as the next symbol in $\underline{\boldsymbol{x}}'$) with probability $p_{\mathrm{t}} = 1 - p_{\mathrm{d}} - p_{\mathrm{i}}$. For computational purposes, we assume that the maximum number of insertions which can occur at each channel use is I. The resulting symbol sequence $\underline{\boldsymbol{x}}'$ is finally affected by an independent and identically distributed (i.i.d.) sequence of AWGN $\underline{\boldsymbol{z}}$ where $\boldsymbol{z} \sim \mathcal{CN}(0, 2\sigma^2)$ and $\underline{\boldsymbol{y}} = \underline{\boldsymbol{x}}' + \underline{\boldsymbol{z}}$ is received at the receiver side[2]. The flow chart depicting the behavior of the channel is shown in Fig. 5.2

[2]Throughout this chapter it is assumed that σ^2 is perfectly known at the receiver. Also note that this general channel model can be easily transformed to insertion only, deletion only, or AWGN channel only, or other hybrid models by adjusting the corresponding probability values.

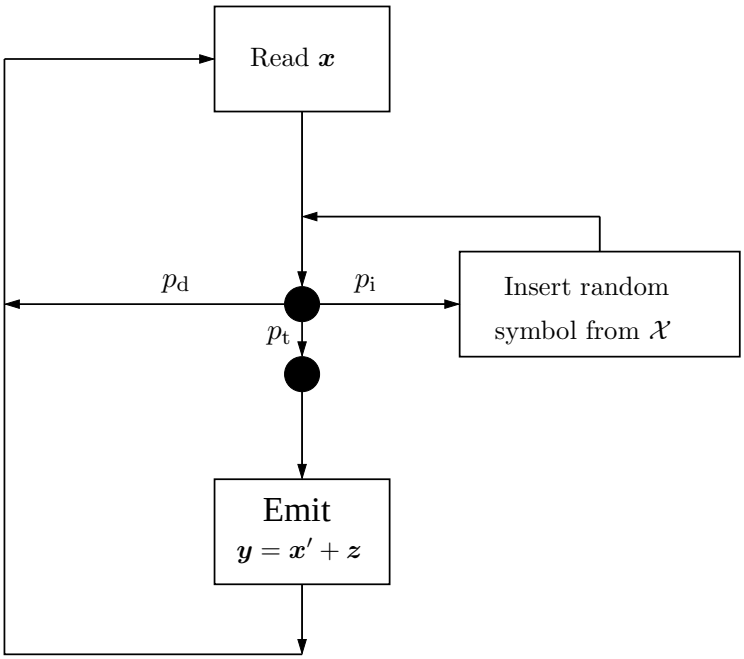


Figure 5.2: The flow chart showing our channel model. The channel insertion probability is p_{i}, deletion probability is p_{d}, and transmission probability is p_{t}. Also, $\boldsymbol{z}$ is the additive white Gaussian noise.

Note that in this chapter, to show the capabilities of the proposed method, we consider totally random and independent symbol insertions/deletions. When symbol insertions result from imperfect synchronization, insertions or deletions tend to be correlated. When this correlation information is used at the receiver, it can lead to easier identification of insertions/deletions compared to random independent insertions/deletions which we consider here. For example, when the insertions are in terms of symbol duplications, detecting these duplications will be easier in the receiver compared to random symbol insertions where all symbols are equally likely to be inserted.

5.2.2 Proposed approach

Now, consider a communication system working on this channel by employing an M-ary signalling (e.g., M-ary phase-shift keying (PSK)). In other words, the memoryless modulator maps each m (where $M = 2^m$) bits to a signal from a signal set of size M by a one-to-one binary labeling map and sends it on the channel. We call this the *base* system. Motivated by the idea of watermark codes [1], we are interested in embedding a watermark in the transmitted sequence. The watermark, being known at the receiver, allows the decoder to deduce the insertions and deletions and to

recover the transmitted sequence.

The watermark can be embedded in the transmitted sequence in many ways. One way of doing this is to add the watermark to the information sequence and treat the information sequence as additive noise at the receiver. This is a direct extension of the binary watermark codes of [1] to non-binary signalling. In particular, the additive watermark $\underline{w}$ can be defined as a sequence of M-ary symbols drawn from the base system constellation. The binary information sequence is first passed through a sparsifier; every k bits of the information sequence is converted to an n-tuple of M-ary symbols. The rate of the sparsifier is then given by $r_{\mathrm{s}} = k/n$ where $0 < r_{\mathrm{s}} < m$ and $M = 2^m$. The average density of the sparsifier f is defined as the average Hamming distance of the n-tuples divided by n. The mapping used in the sparsifier is chosen as to minimize f.

By defining addition as shifting over the constellation symbols, the watermark sequence could be added to the sparsified messages (denote it by $\underline{s}$) and $\underline{w} \oplus \underline{s}$ be sent over the channel. At the receiver, similar to [1], an inner decoder which knows the watermark sequence, uses the received sequence to deduce the insertions/deletions and provides soft information for an outer code.

The main drawback of this method is that the decoder is not able to distinguish between additive noise and the information symbols. This is because the information is embedded into the watermark by adding $\underline{s}$ to $\underline{w}$. Sequence $\underline{s}$ contains both zeros and non-zero symbols. Non-zero symbols shift the watermark symbols over the constellation, similar to what additive noise does. This greatly degrades the performance of the decoder. To improve the decoding performance, $\underline{s}$ should contain as many zeros as possible, i.e., be as sparse as possible, which is equivalent to having a small f. A small f is achieved by decreasing r_{s} which in turn decreases the achievable rates on the channel directly. Also, notice that even in the absence of additive noise, the decoder is still fooled by the shifts occurred over $\underline{w}$ and thus misidentifies some of the insertions/deletions.

To aid error recovery at the receiver, we are interested in an embedding method which makes the watermark as distinguishable as possible from the information sequence. This necessitates using some *extra* resources (other than those used to transmit the information sequence) for transmitting the watermark sequence. These extra resources can be provided by enlarging the signal set. The extra available bits per transmission can then be used to transmit the watermark. After embedding the

watermark, we refer to the system as the *watermarked* system.

In this chapter, we are mostly interested in binary watermark sequences. As a result, to accommodate the watermark bits in each symbol, we expand the signal set size M by the factor of 2, giving rise to a $2M$-ary signalling scheme. For example, if the base system uses 4-PSK, in the watermarked system we use 8-PSK modulation. To provide fair comparison, we put the symbol rate, information bits per symbol (denoted by r_c), and average energy of the signal constellation of the watermarked system equal to those of the base system. As a result, the spectral efficiency and the total transmitted power of the watermarked system are equal to those of the base system. In other words, no bit rate, bandwidth, or power is sacrificed as a result of embedding the watermark.

Notice that $r_c = m$ where $M = 2^m$ for the base system and also the watermarked system when a binary watermark sequence is used for each transmitted symbol. This is because in an M-ary base system all the m available bits are dedicated to information bits. Also, in each symbol of the $2M$-ary watermarked system (with $m+1$ available bits) m bits are assigned to information bits. Later, we will see that sometimes it is more efficient to use non-binary watermark sequences or to assign less than one bit per symbol on average to the watermark giving rise to $0 < r_c < m+1$. These cases will be investigated in Section 5.5.

Expanding the signal set while fixing the average energy of the constellation leads to reduction in the minimum distance of the constellation. Nevertheless, we show that by using the mapping described in Section 5.3.1, the minimum distance $d_{\min}$ between symbols corresponding to the same watermark value does not necessarily reduce. In fact in some cases, e.g., in PSK modulation, the minimum distance does not change compared to the base system. Thus, the noise immunity[3] of the system does not change after adding the watermark.

5.3 System model

The proposed system model is shown in Fig. 5.3. First, the binary information sequence $\underline{b}$ is encoded by the outer code producing the coded binary sequence $\underline{d}$ which is then broken into m-bit subsequences. The modulator then combines the binary

[3]Here, the noise immunity is measured in the absence of synchronization errors under the assumption of minimum distance decoding. As a result, the minimum distance between the signal constellation points can be used as the noise immunity measure.

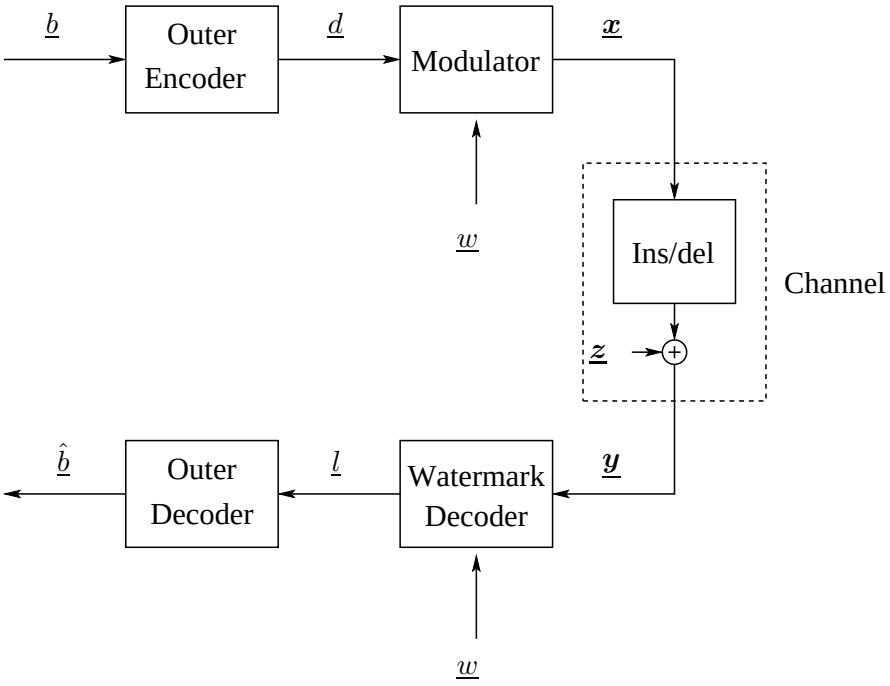


Figure 5.3: The proposed system model.

watermark w and the m-bit subsequences by a one-to-one mapping $\mu : \{0,1\}^{m+1} \to \mathcal{X}$ where $\mathcal{X}$ is the signal set of size $|\mathcal{X}| = 2M = 2^{m+1}$. Then $\underline{\boldsymbol{x}}$ is sent over the channel. The received sequence $\underline{\boldsymbol{y}}$ is first decoded by the watermark decoder which provides soft information for the outer decoder in terms of log-likelihood ratios (LLRs). The LLR sequence $\underline{l}$ is then utilized to decode the information sequence $\hat{\underline{b}}$.

5.3.1 Modulator

The modulator plays a key role in the proposed system. It allows embedding encoded data and watermark bits while ensuring a good minimum distance. The most important part of designing the modulator is to choose an appropriate mapping μ. By viewing μ as $\{0,1\} \times \{0,1\}^m \to \mathcal{X}$, we first divide $\mathcal{X}$ into two disjoint subsets $\mathcal{X}^0$ and $\mathcal{X}^1$ each having M signal points corresponding to watermark bit $w = 0$ and $w = 1$, respectively. Thus, in the label of each signal point, one bit (can be any of the $m+1$ bits) is dedicated to the watermark bit and the other m bits correspond to the m-bit subsequences of $\underline{d}$. Formally we have

$$\mathcal{X}^w = \{\boldsymbol{x} | \boldsymbol{x} \in \mathcal{X}; \ell_{\mathrm{w}}(\boldsymbol{x}) = w\}, \quad \text{for} \quad w = 0, 1,$$

where $\ell_{\mathrm{w}}(\boldsymbol{x})$ denotes the value of the bit in the label of $\boldsymbol{x}$ dedicated to the watermark. We also define $\ell^j(\boldsymbol{x})$ for $j = 1, 2, \ldots, m$ as the j-th non-watermark bit of the label

of $\boldsymbol{x}$.

Now the question is how to choose the labeling. To maximize the noise immunity of the system, and since the watermark sequence is known at the receiver, we maximize the minimum distance between the signal points in each of $\mathcal{X}^0$ and $\mathcal{X}^1$. To do this, first we do a one-level set partitioning [30], i.e., we divide $\mathcal{X}$ into two subsets with the largest minimum distance between the points in each subset. These subsets are named $\mathcal{X}^0$ and $\mathcal{X}^1$ and the watermark bit of the label is assigned accordingly. Next, by a Gray mapping [31] of the signals in each of $\mathcal{X}^0$ and $\mathcal{X}^1$, the non-watermark bits of the label are assigned. This process is illustrated for two different signal constellations in Figs. 5.4 and 5.5. The Gray mapping ensures the least bit error rate in each subset [31].

The minimum distance of the constellation is now defined as

$$d_{\min} = \min_{w} \min_{\{\boldsymbol{x}_i, \boldsymbol{x}_j\} \subset \mathcal{X}^w, \boldsymbol{x}_i \neq \boldsymbol{x}_j} ||\boldsymbol{x}_i - \boldsymbol{x}_j||.$$

Notice that by assuming signal constellations of fixed energy, going from M-PSK in the base system to $2M$-PSK in the watermarked system does not change $d_{\min}$ (see Fig. 5.4). For the QAM, as illustrated in Fig. 5.5, $d_{\min}$ does change because of energy adjustments but always stays very close to that of the original constellation. For example in Fig. 5.5, $d_{\min}$ is reduced by only 2.4%.

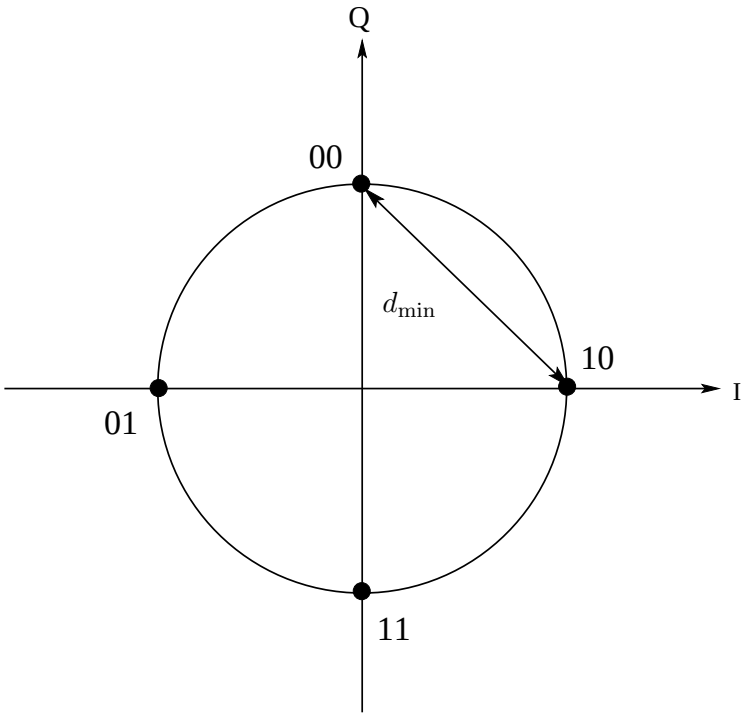
A definition which proves useful in the next sections is

$$\mathcal{X}_j(w_i, d_{i,j}) = \left\{ \boldsymbol{x} | \boldsymbol{x} \in \mathcal{X}; \ell_{\mathrm{w}}(\boldsymbol{x}) = w_i, \ell^j(\boldsymbol{x}) = d_{i,j} \right\},$$

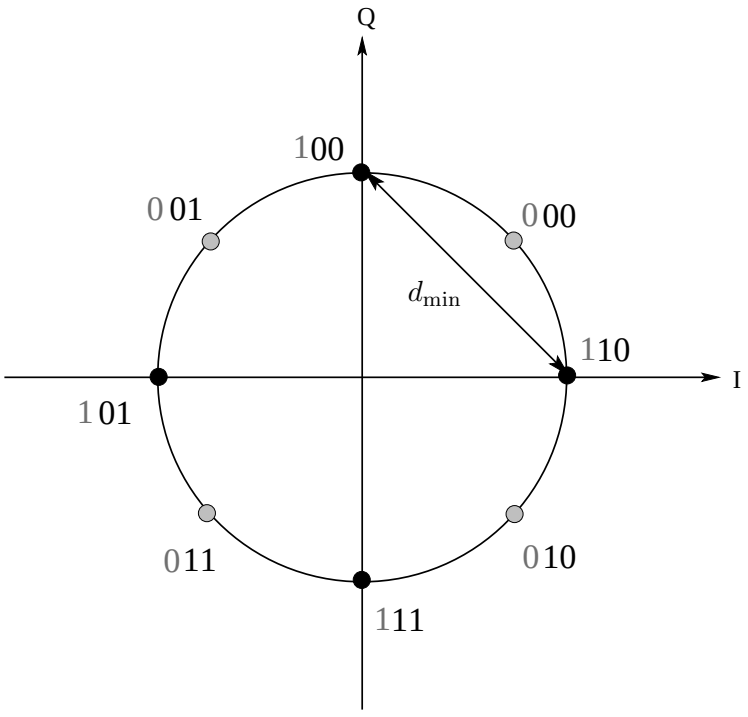
where i denotes the index of both the watermark bit and the m-bit subsequences of $\underline{d}$ and $d_{i,j} = d_{(i-1)m+j}$ denotes the j-th bit of the i-th subsequence. Considering a watermark sequence of length N and an encoded data sequence of length mN, then $i = 1, 2, \ldots, N$. Thus, $\mathcal{X}_j(u, v)$ refers to the subset of $\mathcal{X}$ where the watermark bit w_i is equal to u and the j-th data bit in the i-th subsequence, i.e., $d_{i,j}$, is equal to v. The size of this subset is $M/2$.

5.3.2 Watermark decoder

The goal of the watermark decoder is to produce LLRs for the outer decoder given $\underline{w}$ and the received sequence $\underline{\boldsymbol{y}}$. As in [1], by ignoring the correlations in $\underline{d}$, we can use an HMM to model the received sequence and then use the forward-backward algorithm [80] to calculate posterior probabilities or LLRs for the outer decoder.

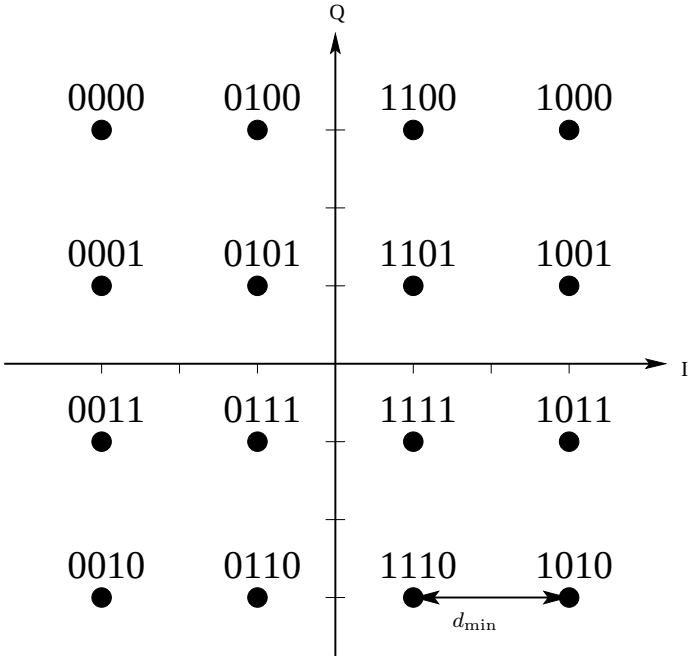


(a) Base system

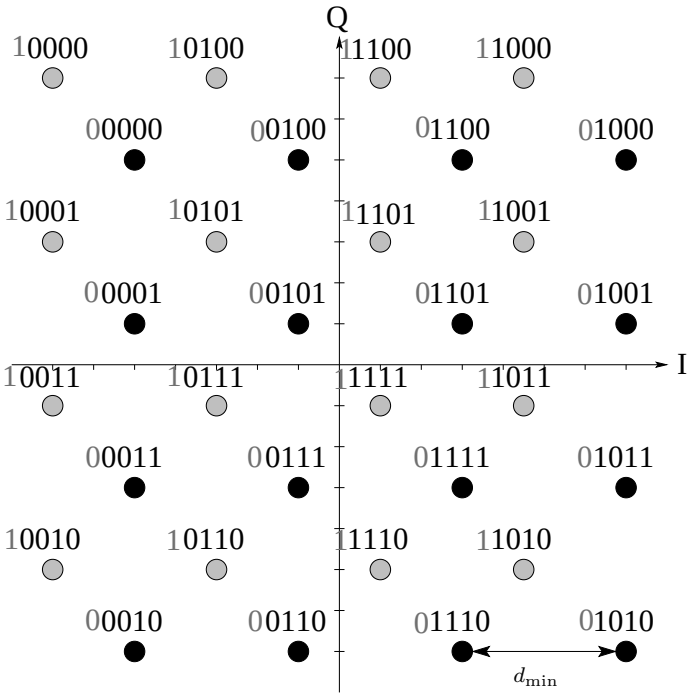


(b) Watermarked system

Figure 5.4: Signal constellations and their labeling for the base system (4-PSK) and the watermarked system (8-PSK). The leftmost bit in the label of the watermarked system corresponds to the watermark bit. For both constellation $d_{\min} = \sqrt{2}$.



(a) Base system



(b) Watermarked system

Figure 5.5: Signal constellations and their labeling for the base system (16-QAM) and the watermarked system (32-AM/PM). The leftmost bit in the label of the watermarked system corresponds to the watermark bit. Both constellation have unit average energy. Thus, $d_{\min} = 2/\sqrt{10} = 0.633$ for the base system and $d_{\min} = 4/\sqrt{42} = 0.617$ for the watermarked system.

Notice that due to the nature of the channel which introduces insertions and deletions, there will be a synchronization *drift* between $\underline{\boldsymbol{x}}$ and $\underline{\boldsymbol{y}}$. The synchronization drift at position i, i.e., t_i is defined as the (number of insertions) $-$ (number of deletions) occurred in the signal stream until the ith symbol, i.e., $\boldsymbol{x}_i$, is ready for transmission[4]. The drifts $\{t_i\}_{i=1}^{N}$, form the hidden states of the HMM. Each state t_i takes values from

$$\mathbf{T} = \{\dots, -2, -1, 0, 1, 2, \dots\}. \tag{5.1}$$

Thus, t_i performs a random walk on $\mathbf{T}$ whose mean and variance depend on p_{i} and p_{d}. To reduce the decoding complexity, as in [1], we limit the drift to $|t_i| \leq t_{\max}$ where $t_{\max}$ is usually chosen large enough such that it accommodates all likely drifts with high probability. For example, when $p_{\mathrm{i}} = p_{\mathrm{d}}$, $t_{\max}$ is chosen several times larger than $\sqrt{Np_{\mathrm{d}}/(1-p_{\mathrm{d}})}$ which represents the standard deviation of the drifts over a block of size N.

To further characterize the HMM [80], we need the state transition probabilities, i.e., $P_{ab} = P(t_{i+1} = b | t_i = a)$. Each symbol $\boldsymbol{x}_i$ entering the channel can produce any number of symbols between 0 and $I+1$ at the channel output. As a result, if $t_i = a$, then $t_{i+1} \in \{a-1, \dots, a+I\}$. Notice that the transition from $t_i = a$ to $t_{i+1} = b$ can occur in two ways. One is when $\boldsymbol{x}_i$ is deleted by the channel and $(b-a+1)$ symbols are inserted by the channel. The other one is when $\boldsymbol{x}_i$ is transmitted and $(b-a)$ symbols are inserted by the channel. In either case, $(b-a+1)$ symbols are produced at the channel output. As a result, the state transition probabilities are given by

$$P_{ab} = \begin{cases} p_{\mathrm{d}} & b = a-1 \\ \alpha_I p_{\mathrm{i}} p_{\mathrm{d}} + p_{\mathrm{t}} & b = a \\ \alpha_I (p_{\mathrm{i}}^{b-a+1} p_{\mathrm{d}} + p_{\mathrm{i}}^{b-a} p_{\mathrm{t}}) & a < b < a+I \\ \alpha_I p_{\mathrm{i}}^{I} p_{\mathrm{t}} & b = a+I \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

where $\alpha_I = 1/(1-p_{\mathrm{i}}^{I})$ is a constant normalizing the effects of maximum insertion length I and ensuring that the sum of probabilities is 1.

We also need to calculate the conditional probability of producing the observation sequence $\widetilde{\underline{\boldsymbol{y}}} = \{\boldsymbol{y}_{i+a}, \dots, \boldsymbol{y}_{i+b}\}$ given the transition from $t_i = a$ to $t_{i+1} = b$. As

[4]This means that if $\boldsymbol{x}_{i-1}$ is not deleted by the channel it is received as $\boldsymbol{y}_{i-1+t_i}$.

stated, this transition can occur in two ways. Thus,

$$\begin{aligned} Q^i_{ab}(\underline{\widetilde{\boldsymbol{y}}}) &= P(\underline{\widetilde{\boldsymbol{y}}}|t_i = a, t_{i+1} = b, w_i, \mathcal{H}) \\ &= \left(\alpha_I p_{\mathrm{i}}^{b-a+1} p_{\mathrm{d}} \prod_{k=i+a}^{i+b} \gamma_k + \alpha_I p_{\mathrm{i}}^{b-a} p_{\mathrm{t}} \beta_{i+b} \prod_{k=i+a}^{i+b-1} \gamma_k \right) / P_{ab}, \end{aligned} \tag{5.3}$$

where $\mathcal{H}$ denotes set of parameters of the HMM, i.e., $\mathcal{H} = \{[P_{ab}], \mathbf{T}\}$, γ_k is the probability of receiving $\boldsymbol{y}_k$ given that $\boldsymbol{y}_k$ is an inserted symbol, and β_{i+b} is the probability of receiving $\boldsymbol{y}_{i+b}$ assuming that it is the result of transmitting $\boldsymbol{x}_i \in \mathcal{X}^{w_i}$. Formally, we have

$$\gamma_k = \frac{1}{2M} \sum_{\boldsymbol{x} \in \mathcal{X}} \frac{1}{2\pi\sigma^2} \exp\left(-\frac{|\boldsymbol{y}_k - \boldsymbol{x}|^2}{2\sigma^2} \right), \tag{5.4}$$

and

$$\beta_{i+b} = P(\boldsymbol{y}_{i+b}|t_i = a, t_{i+1} = b, w_i, \mathcal{H}) = \frac{1}{M} \sum_{\boldsymbol{x} \in \mathcal{X}^{w_i}} \frac{1}{2\pi\sigma^2} \exp\left(-\frac{|\boldsymbol{y}_{i+b} - \boldsymbol{x}|^2}{2\sigma^2} \right). \tag{5.5}$$

Now that the HMM is defined, we use the forward-backward algorithm to calculate LLRs. By ignoring the correlations between the bits of $\underline{d}$ and assuming $P(d_{i,j} = 0) = P(d_{i,j} = 1)$, the bit by bit LLR is calculated as

$$l_{i,j} = \log \frac{P(d_{i,j} = 0|\underline{\boldsymbol{y}}, \underline{w}, \mathcal{H})}{P(d_{i,j} = 1|\underline{\boldsymbol{y}}, \underline{w}, \mathcal{H})} \tag{5.6}$$

$$= \log \frac{P(\underline{\boldsymbol{y}}|d_{i,j} = 0, \underline{w}, \mathcal{H})}{P(\underline{\boldsymbol{y}}|d_{i,j} = 1, \underline{w}, \mathcal{H})} = \log \frac{\sum_{\boldsymbol{x}_i \in \mathcal{X}_j(w_i,0)} P(\underline{\boldsymbol{y}}|\boldsymbol{x}_i, \underline{w}, \mathcal{H})}{\sum_{\boldsymbol{x}_i \in \mathcal{X}_j(w_i,1)} P(\underline{\boldsymbol{y}}|\boldsymbol{x}_i, \underline{w}, \mathcal{H})}. \tag{5.7}$$

By using the forward-backward algorithm, the posterior probabilities are found by [1, 80]

$$P(\underline{\boldsymbol{y}}|\boldsymbol{x}_i, \underline{w}, \mathcal{H}) = \sum_{a,b} F_i(a) \acute{Q}^i_{ab}(\underline{\widetilde{\boldsymbol{y}}}|\boldsymbol{x}_i) B_{i+1}(b) \tag{5.8}$$

where the forward quantity is defined as

$$F_i(a) = P(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{i-1+a}, t_i = a|\underline{w}, \mathcal{H}), \tag{5.9}$$

the backward quantity as

$$B_i(b) = P(\boldsymbol{y}_{i+b}, \ldots |t_i = b, \underline{w}, \mathcal{H}), \tag{5.10}$$

and

$$\begin{aligned} \acute{Q}^i_{ab}(\underline{\widetilde{\boldsymbol{y}}}|\boldsymbol{x}_i) &= P(\underline{\widetilde{\boldsymbol{y}}}, t_{i+1} = b|t_i = a, \boldsymbol{x}_i, \mathcal{H}) \\ &= \alpha_I p_{\mathrm{i}}^{b-a+1} p_{\mathrm{d}} \prod_{k=i+a}^{i+b} \gamma_k + \alpha_I p_{\mathrm{i}}^{b-a} p_{\mathrm{t}} \acute{\beta}_{i+b} \prod_{k=i+a}^{i+b-1} \gamma_k, \end{aligned} \tag{5.11}$$

where

$$\acute{\beta}_{i+b} = P(\boldsymbol{y}_{i+b}|t_i = a, t_{i+1} = b, \boldsymbol{x}_i, \mathcal{H}) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{|\boldsymbol{y}_{i+b} - \boldsymbol{x}_i|^2}{2\sigma^2}\right).$$

The forward and backward quantities are recursively computed by the forward pass

$$F_i(a) = \sum_{c\in\{a-I,\ldots,a+1\}} F_{i-1}(c) P_{ca} Q_{ca}^{i-1}(\boldsymbol{y}_{i-1+c}, \ldots, \boldsymbol{y}_{i-1+a}), \tag{5.12}$$

and the backward pass

$$B_i(b) = \sum_{c\in\{b-1,\ldots,b+I\}} P_{bc} Q_{bc}^{i}(\boldsymbol{y}_{i+b}, \ldots, \boldsymbol{y}_{i+c}) B_{i+1}(c). \tag{5.13}$$

If the block boundaries are not known at the decoder, we can use the sliding window decoding technique used in [1]. Assuming a continuous stream of transmitted blocks and received symbols, the forward-backward algorithm is used to infer the block boundaries. Then the decoding window is anchored at the most likely start of the next block and next block is decoded. Most of the results of this chapter are shown using this sliding window decoding technique. We will briefly explain the methodology in Section 5.4.1. For the first transmitted block, we assume that the initial drift is zero. Thus, we use

$$F_1(a) = \begin{cases} 1 & a = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{5.14}$$

It is also possible to insert some markers which specify the block boundaries into the transmitted sequence. By dedicating all the $m + 1$ bits in the symbols at the boundaries to markers, they can be easily detected at the receiver. In this case, the block boundaries can be inferred by detecting the markers. Notice that as the block length becomes larger, recognizing the block boundaries requires less overhead and becomes more efficient. To see this, first define the marker rate as the number of marker symbols in each block divided by N. Given a fixed marker rate, the number of marker symbols is increased as N grows. Increasing the number of marker symbols leads to a better boundary detection at the decoder for a fixed p_{d} and p_{i} because the probability of misidentifying larger block of markers is decreased. Thus, for large block lengths one can assume that the block boundaries are known at the decoder and use the following initial conditions for the backward pass of each block:

$$B_N(b) = P_{bc} Q_{bc}^{N}(\boldsymbol{y}_{N+b}, \ldots) \tag{5.15}$$

where $c = t_{N+1}$ is the final drift at the end of the block.

In the next stage of decoding, the LLRs, calculated by inserting (5.8) in (5.6), are passed to the outer decoder.

5.3.3 Outer code

The outer code can almost be any binary error correcting code. Due to the exemplary performance of LDPC codes on many communication channels and their flexible structure, we choose LDPC codes in this chapter.

At the transmitter, a binary LDPC code of rate R is used to encode the binary information sequence $\underline{b}$ of length mNR producing the binary coded sequence $\underline{d}$ of length mN. At the receiver, the mN bit LLRs of (5.6) are used to recover $\underline{b}$.

5.4 Error rates and fundamental limits

In this section, we demonstrate the capabilities of the proposed solution through examples and discussions. In particular, we evaluate the watermarked system by providing bit and word error rates (BER and WER), maximum achievable transmission rates, and comparing them with two benchmark systems. We demonstrate our results using the following two examples.

Example 1: Consider a base system with 4-PSK modulation depicted in Fig. 5.4a which gives rise to a watermarked system with 8-PSK modulation. The labeling μ is chosen based on the method described in Section 5.3.1. The constellation and labeling are depicted in Fig. 5.4b.

Example 2: In this example, we consider a 16-QAM base system and a 32-QAM watermarked system. Notice that different constellations can be considered for 32-ary modulation. We consider a 32-AM/PM constellation whose $d_{\min}$ is very close to the base system (they differ by only 2.4%). The constellations and their labelings are depicted in Fig. 5.5.

5.4.1 Error rates

First, consider Example 1. We use a $(3, 6)$-regular LDPC code ($R = 0.5$) of length $20,024$ constructed by the progressive edge growth (PEG) algorithm [82] as the outer code. The LDPC code is decoded by the sum-product algorithm [8] allowing a maximum of 400 iterations. The watermark sequence $\underline{w}$ is chosen to be a pseudo-random binary sequence. Since $m = 2$, the block length is $N = 10,012$. The

maximum insertion length is chosen as $I = 5$, the channel insertion and deletion probabilities are assumed equal, i.e., $p_{\rm i} = p_{\rm d} = p_{\rm id}$, and $t_{\max} = 5\sqrt{Np_{\rm id}/(1-p_{\rm id})}$.

A continuous stream of blocks of $\underline{b}$, $\underline{d}$, and $\underline{\boldsymbol{x}}$ is generated and sent over the channel. A continuous sequence of $\underline{\boldsymbol{y}}$ is then received at the decoder. We assume that block boundaries are not known at the receiver. Thus, $\underline{\boldsymbol{y}}$ is decoded by the forward-backward algorithm using a sliding window decoding technique [1]. For the first block, we assume that the receiver knows the starting position, i.e, we use (5.14) to initialize the forward pass. For subsequent blocks, the watermark decoder is responsible to infer the boundaries and calculate LLRs for the outer code. This is done by first running the forward pass several multiples of $t_{\max}$ (here, six) beyond the expected position of the block boundary and initializing the backward pass from these last calculated forward quantities. Then the most likely drift at the end of each block is found as $\hat{t}_{N+1} = \arg\max_a F_{N+1}(a)B_{N+1}(a)$ and is used to slide the decoding window to the most likely start of the next block.

Occasionally, the watermark decoder makes errors in identifying the block boundaries. If these errors accumulate, synchronization is lost and successive blocks fail to be successfully decoded. To protect against such gross synchronization loss, we use the re-synchronization technique of [1] whose details are omitted in the interest of space.

We simulate the system under different SNRs and insertion/deletion probabilities. The BER and WER of the system are plotted in Fig. 5.6 versus different values of $p_{\rm id}$ under fixed SNRs. For example, at SNR$=10$ dB, the system is able to recover on average $1,400$ symbol insertions/deletions per block of $10,012$ symbols with an average BER less than 10^{-5}. This increases to recovering about $1,920$ insertions/deletions at SNR$=20$ dB. Fig. 5.7 shows the performance of the system versus SNR under fixed $p_{\rm id}$.

We also simulate the system under a $(3,4)$-regular LDPC code ($R = 0.25$) of length $20,024$ with the same parameters. The system is now capable of recovering on average $2,700$ insertions/deletions per block of $10,012$ symbols at SNR$=20$ dB with an average BER$< 10^{-5}$. This increases to $2,900$ insertions/deletions using an optimized irregular LDPC code of the same rate and length with degree distributions reported in Table 5.1 (Code 1). We will briefly describe the LDPC optimization method in the next section.

We are not aware of any practical coding method in the literature that can

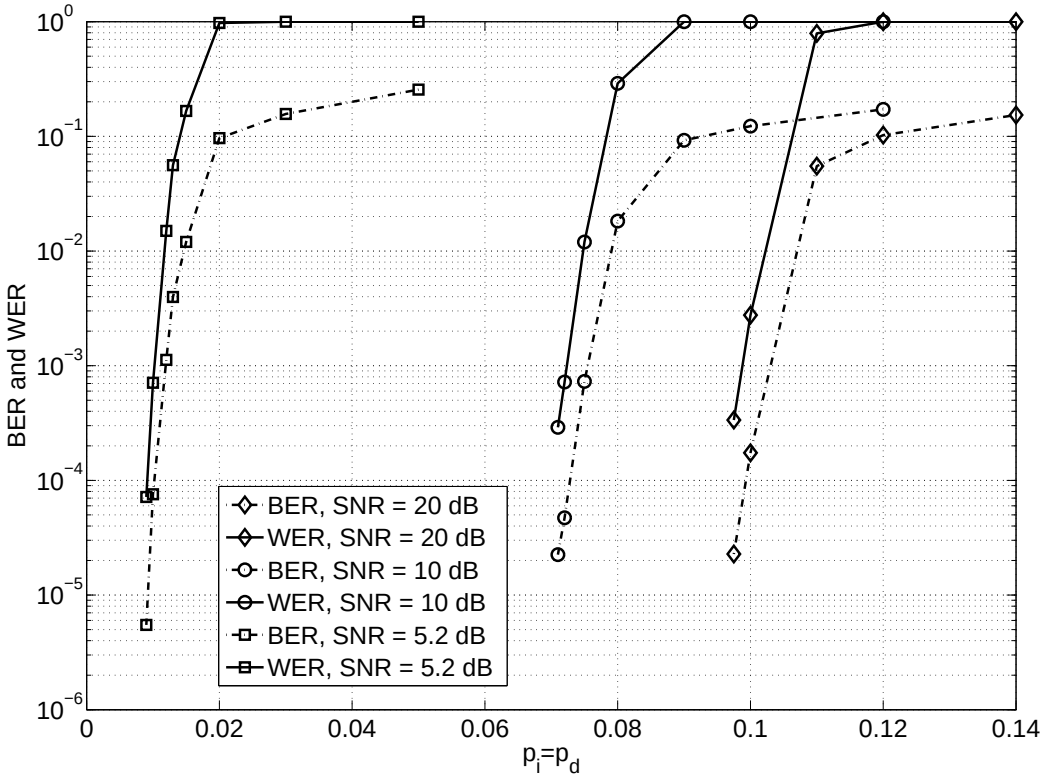


Figure 5.6: BER and WER of the 8-PSK watermarked system employing a (3,6)-regular LDPC code of length $20,024$ versus p_{id} at fixed SNRs.

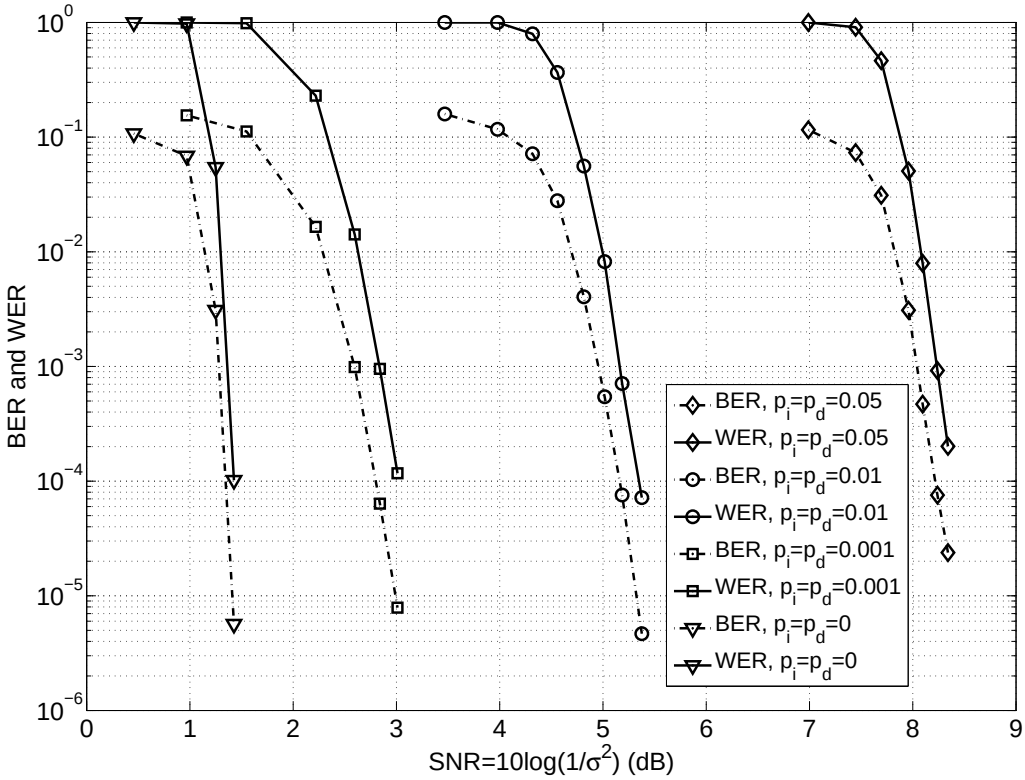


Figure 5.7: BER and WER of the 8-PSK watermarked system employing a (3,6)-regular LDPC code of length $20,024$ versus SNR at fixed values of p_{id}.

be directly and fairly compared to our proposed system. However, we provide comparisons with the best results of [1–3, 77]. It is worth mentioning that this comparison is not completely fair as the I/D channel considered in these works is binary whereas in our case is non-binary. All in all, we believe that this comparison provides insight into what can be achieved by exploiting the extra degrees of freedom provided by non-binary signalling. This comparison is depicted in Figs. 5.8, 5.9, and 5.10. To make the comparison as fair as possible, we have adjusted the block size and the rate of our 8-PSK watermarked system according to the parameters of codes considered in the comparison. It is evident from these figures that a significant improvement in the error correction performance is achieved by using non-binary signalling. There is also a significant performance improvement compared to the method of [77] which considers marker codes with iterative exchange of information between the inner and outer decoders. Marker codes concatenated with optimized irregular LDPC outer codes with overall rates around 0.4 and block length $5,000$ have been reported in [77] which can reliably work under $p_{\mathrm{id}} < 0.04$. As Fig. 5.9

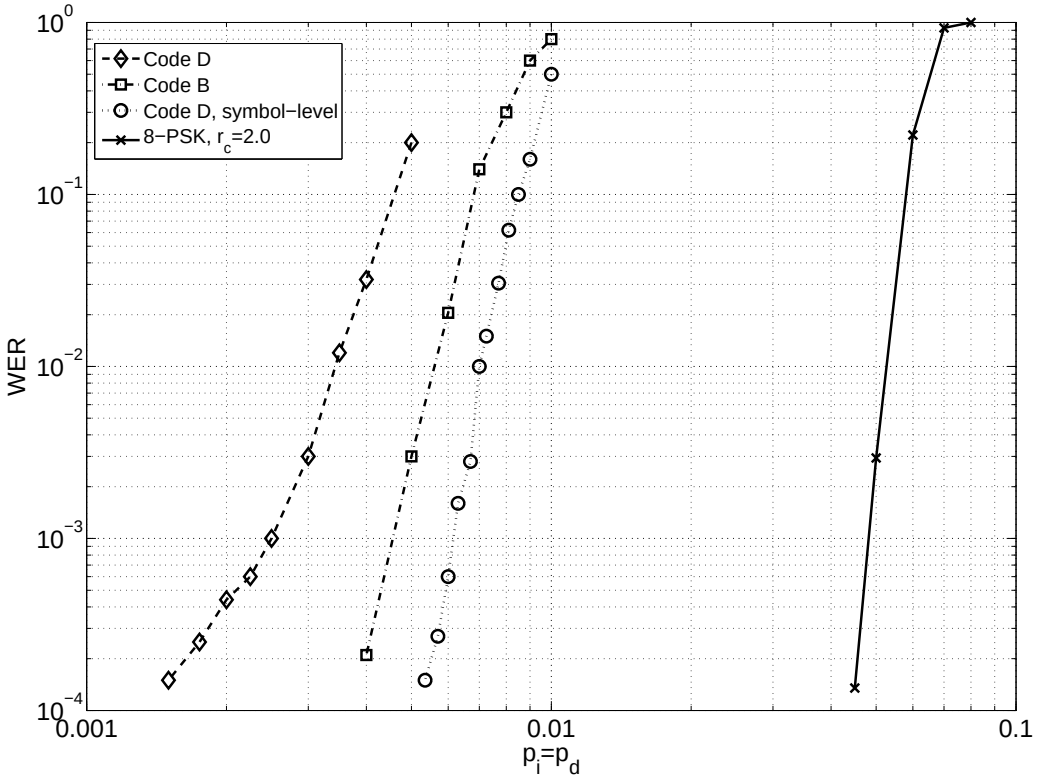


Figure 5.8: WER comparison of the 8-PSK watermarked system with the best results of [1–3] at $R = 0.71$. Code D is a binary watermark code from [1] with overall rate 0.71, overall block length $4,995$, and outer LDPC code defined over $\mathrm{GF}(16)$. Code B is a binary marker code from [2] with overall rates 0.71, overall block length $4,995$ with a binary LDPC code as outer code. Codes D is also decoded by the symbol-level decoding method of [3]. All these codes are decoded on the binary I/D channel with no substitution errors or additive noise. For the non-binary I/D channel with 8-PSK signalling in Example 1, we have done sliding window decoding at SNR$= 20$ dB for an LDPC code with variable node degree 3, rate 0.71, and block lengths $4,996$.

shows, a regular half-rate code with block length $4,002$ can do much better in our case even without iterative exchange of information.

5.4.2 Achievable information rates

To obtain the capacity of the I/D channel, one is interested to calculate [19]

$$C = \lim_{N\to\infty} \frac{1}{N} \sup_{p(\underline{\boldsymbol{x}})} I(\underline{\boldsymbol{x}}; \underline{\boldsymbol{y}}), \tag{5.16}$$

where $\underline{\boldsymbol{x}}$ is the channel input sequence of length N, $\underline{\boldsymbol{y}}$ is the received sequence of random length, and $p(\underline{\boldsymbol{x}})$ denotes the joint distribution of the input sequence. Unfortunately, due to the presence of insertions/deletions, finding (5.16) or its bounds has proven to be extremely challenging and the capacity is unknown. No single

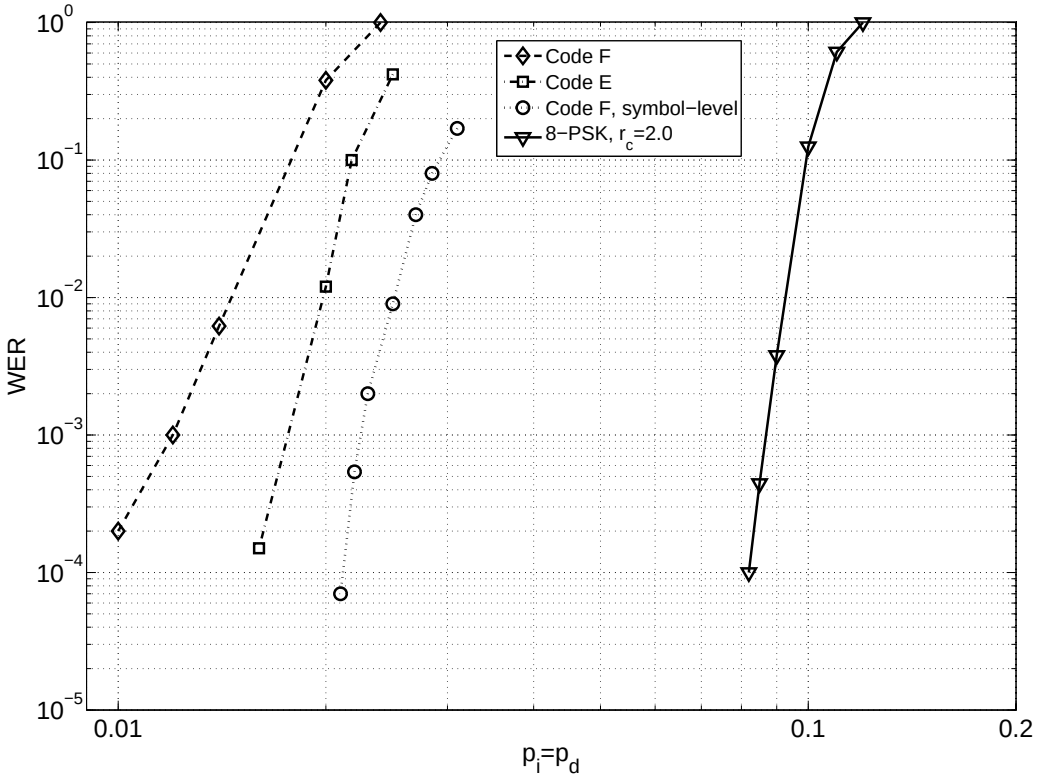


Figure 5.9: WER comparison of the 8-PSK watermarked system with the best results of [1–3] at $R = 0.50$. Code F is a binary watermark code from [1] with overall rate 0.50, overall block lengths $4,002$, and outer LDPC code defined over $\mathrm{GF}(16)$. Code E is a binary marker code from [2] with overall rate 0.50 and overall block length $4,000$ with a binary LDPC code as outer code. Codes F is also decoded by the symbol-level decoding method of [3]. All these codes are decoded on the binary I/D channel with no substitution errors or additive noise. For the non-binary I/D channel with 8-PSK signalling in Example 1, we have done sliding window decoding at SNR$=20$ dB for an LDPC codes with variable node degree 3, rate 0.50, and block length $4,002$.

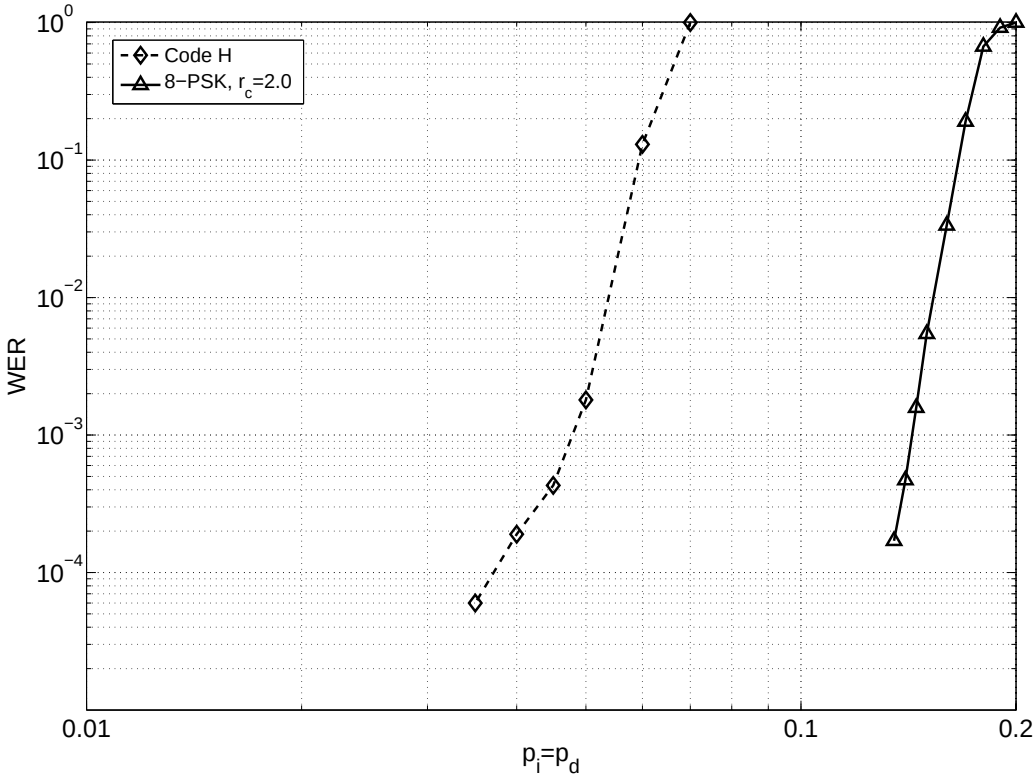


Figure 5.10: WER comparison of the 8-PSK watermarked system with the best results of [1] at $R = 3/14$. Code H is a binary watermark code from [1] with overall rate $3/14$, overall block length $4,662$, and outer LDPC code defined over $\mathrm{GF}(8)$. This code is decoded on the binary I/D channel with no substitution errors or additive noise. For the non-binary I/D channel with 8-PSK signalling in Example 1, we have done sliding window decoding at SNR$= 20$ dB for an LDPC code with variable node degree 3, rate $3/14$, and block length $4,662$.

letter characterization of the mutual information also exists. Most of the results in the literature focus on sub-optimal decoding or more constrained channel models (such as deletion-only channel) and provide bounds on the capacity [83–86]. Most of these bounds, however, are driven for binary I/D channels and binary synchronization codes and either cannot be extended to non-binary I/D channel or become computationally intensive such as the bounds in [87].

A trellis-based approach is developed in [78] to obtain achievable information rates for binary I/D channels with AWGN and inter-symbol interference under i.i.d. inputs (uniform $p(\underline{\boldsymbol{x}})$). This approach which mainly uses the forward pass of the forward-backward algorithm, can be extended to i.i.d. non-binary inputs and thus our channel model. We will use this method to find lower bounds on the capacity of the channel, i.e., $C_{\text{i.u.d.}}$, and compare the achievable rates of our watermarked system with this lower bound. There also exist bounds on the performance of q-ary synchronization codes [4] which we will use in the comparisons.

To obtain the achievable rates of our watermarked system, we calculate the maximum average per-symbol mutual information. In particular, we obtain an estimate of the average mutual information between $\underline{\boldsymbol{y}}$ and $\underline{\boldsymbol{x}}$ given $\underline{w}$. Assuming that $\underline{\boldsymbol{x}}$ is a sequence of i.i.d. symbols, the average per-symbol mutual information is given by $\frac{1}{N}\sum_{i=1}^{N} I(\boldsymbol{x}_i;\underline{\boldsymbol{y}}|\underline{w})$ where

$$\begin{aligned} I(\boldsymbol{x}_i;\underline{\boldsymbol{y}}|\underline{w}) &= H(\boldsymbol{x}_i|w_i) - H(\boldsymbol{x}_i|\underline{\boldsymbol{y}},w_i) \\ &= r_{\text{c}} - E_{\underline{\boldsymbol{y}}}\Bigg[-\sum_{\boldsymbol{x}_i\in\mathcal{X}^{w_i}} \Bigg(\frac{P(\underline{\boldsymbol{y}}|\boldsymbol{x}_i,w_i)}{\sum_{\boldsymbol{x}_i\in\mathcal{X}^{w_i}} P(\underline{\boldsymbol{y}}|\boldsymbol{x}_i,w_i)} \\ &\qquad \times \log_2\left(\frac{P(\underline{\boldsymbol{y}}|\boldsymbol{x}_i,w_i)}{\sum_{\boldsymbol{x}_i\in\mathcal{X}^{w_i}} P(\underline{\boldsymbol{y}}|\boldsymbol{x}_i,w_i)}\right)\Bigg)\Bigg], \end{aligned} \tag{5.17}$$

and the conditional probabilities are given by the watermark decoder. While it is not possible to do an exact calculation of the expectation, it is possible to calculate it numerically by Monte Carlo simulation. Then, (5.17) can be used to find an estimate of the achievable rates of the watermarked system and a lower bound on the capacity of the channel. It should be noted that under large block lengths N, the variance of (5.17) under Monte Carlo simulation is usually very small. Thus, it converges to the average very fast. Here, our results are averaged over 100 blocks.

Using (5.17) and assuming known block boundaries, the achievable information rates of the watermarked system of Example 1 is plotted versus SNR in Fig. 5.11. We also compare the achievable rates with those of the two benchmark systems. One

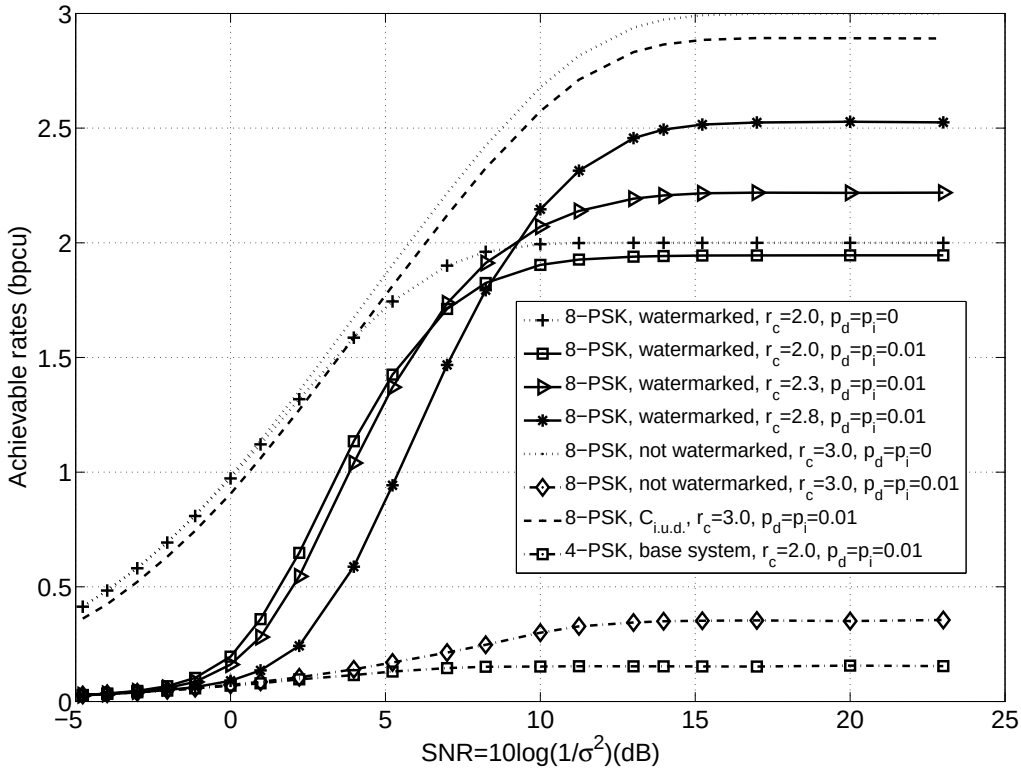


Figure 5.11: Maximum achievable information rates (bits per channel use) versus SNR under different modulations assuming a 4-PSK base system ($r_{\rm c} = 2.0$). The 8-PSK watermarked system mentioned in Section 5.3.1 has $r_{\rm c} = 2.0$. The maximum achievable rates given by (5.17) under 8-PSK modulation with no watermark ($r_{\rm c} = 3.0$), and under two 8-PSK watermarked system with partial watermarking ($r_{\rm c} = 2.3$ and 2.8) mentioned in Section 5.5 are plotted for comparison. Also, $C_{\rm i.u.d.}$ is plotted for the 8-PSK modulation.

is the base system (4-PSK) which has the same $d_{\min}$, and another one is the system which has the same number of modulation points (8-PSK) as in the watermarked system but is not watermarked, i.e., all the $m+1$ bits are dedicated to information bits. Both of these systems use Gray mapping and are decoded by the forward-backward algorithm described in Section 5.3.2 with the exception that there is no watermark. The number of symbols per block is kept fixed at $10,012$ for all three systems so that the average number of symbols corrupted by insertions/deletions remains the same. Notice that $r_{\rm c} = 2.0$ for the base and the watermarked system and $r_{\rm c} = 3.0$ for the 8-PSK system with no watermark.

The dashed curves in Fig. 5.11 correspond to the maximum achievable information rates of the three systems when $p_{\rm id} = 0$. In this case, it is clear that the watermarked and the base system achieve the same rates but the 8-PSK system with no watermark achieves higher rates. At $p_{\rm id} = 0.01$, however, the watermarked

system performs much better than the two benchmark systems in terms of the maximum achievable rates (by using (5.17)) on the channel. This is of course not very surprising since no watermark is used in the benchmark systems and their only source of protection against insertions/deletions comes from the fact that they are decoded by the forward-backward algorithm.

The figure also depicts $C_{\text{i.u.d.}}$ under 8-PSK signalling at $p_{\text{id}} = 0.01$. Comparing this curve with the results given by (5.17) shows how far the achievable rates of our watermarked system are from the maximum achievable rates on the channel using the same constellation under i.i.d. inputs. Although this gap is not small, we are not aware of any results in the literature that can approach $C_{\text{i.u.d.}}$. This gap can be made smaller by the method we show in Section 5.5.

We also provide a comparison with [1] in terms of comparing the achievable rates of these two systems as viewed by the outer code. In particular, we calculate the average $I(d_{i,j}; l_{i,j})$ which is a number between 0 and 1 for both systems. This is done by Monte Carlo simulation, using the LLRs produced by the watermark decoder, i.e., using (5.6). This comparison is depicted in Fig. 5.12. The achievable rates seen by the outer code from [1] are compared to those of the watermarked 8-PSK system at SNR$=20$ dB. The rates of [1] are given for three binary watermark codes with sparsifier rates of 3/7, 4/6, and 4/5 and are calculated assuming no substitution error on the channel which is similar to the high SNR case on our channel. As depicted, the achievable rates of the proposed watermarked 8-PSK system are much higher than those of [1].

Given the success of LDPC codes on many channels, we expect that the information rates of Fig. 5.12 can be approached with carefully designed irregular LDPC codes of large block lengths. To demonstrate this, we have optimized the degree distributions of irregular LDPC codes of rates 0.25, 0.50, and 0.75, and constructed codes of length $20,024$. The optimization process is done by the conventional numerical LDPC optimization methods in the literature (e.g., see [53]). These optimization techniques usually use the pdf of the LLRs. On most channels, this LLR pdf can be calculated analytically. However, this cannot be done in our case due to nature of the channel. Thus, Monte Carlo simulation is used to find estimates of the LLR pdf. Given the channel parameters, this is done by simulating a large number of channel realizations, calculating LLRs using (5.6), and finally computing the average LLR pdf (probability mass function to be more precise). Next, the rate of the code is

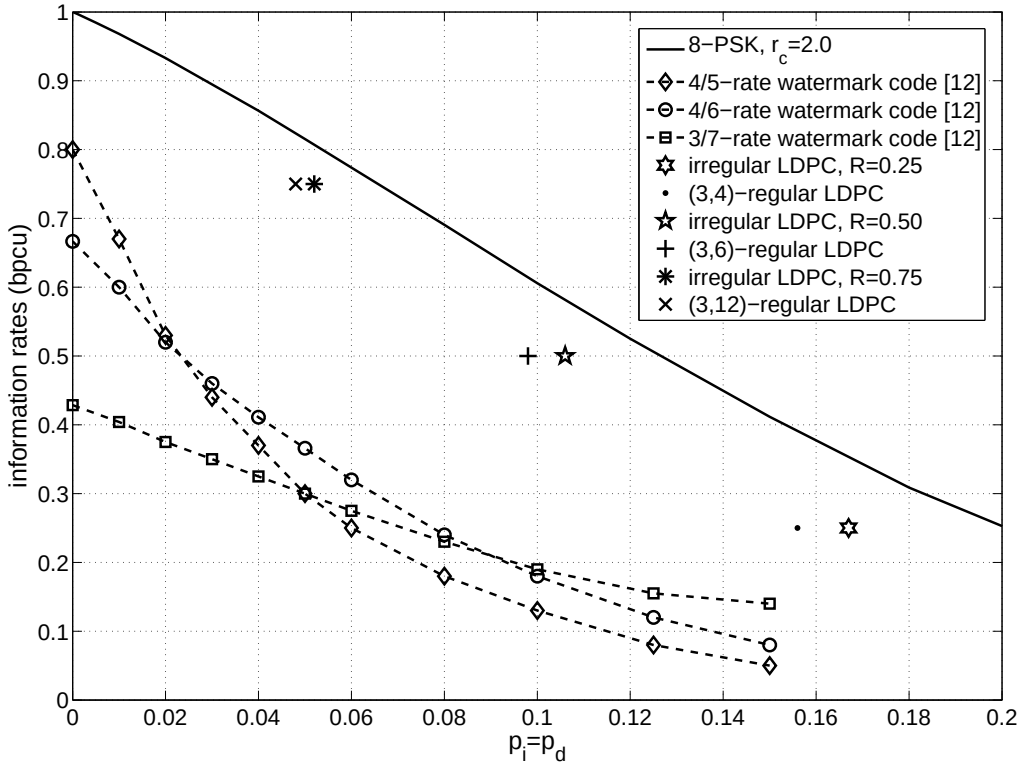


Figure 5.12: Maximum achievable information rates as viewed by the outer code $I(d_{i,j}; l_{i,j})$ for the 8-PSK watermarked system at high SNR are compared with the obtained rates of [1]. The rates of [1] are calculated assuming no substitution errors. Also, the maximum p_{id} that the 8-PSK watermarked system can tolerate with BER less than 10^{-5} is indicated for the three optimized irregular LDPC codes of rates 0.25, 0.50, and 0.75 and three regular LDPC codes.

	Code 1	Code 2	Code 3
λ_2	0.2793	0.1920	0.2562
λ_3	0.2648	0.2480	0.3334
λ_4		0.0064	0.0010
λ_5			0.0022
λ_6	0.0173	0.0171	0.3621
λ_7	0.0575	0.0709	0.0434
λ_8	0.0938	0.1223	0.0017
λ_9	0.0279	0.0278	
λ_{10}	0.0528	0.0302	
λ_{18}		0.0413	
λ_{26}	0.0494	0.0302	
λ_{27}	0.0126	0.0137	
λ_{28}	0.0179	0.0199	
λ_{29}	0.0303	0.0358	
λ_{30}	0.0964	0.1507	
ρ_5	1.0000		
ρ_9		1.0000	
ρ_{13}			1.0000
Rate	0.25	0.50	0.75

Table 5.1: Variable and check node degree distributions for the optimized irregular LDPC codes; All results achieved assuming maximum variable node degree of 30

maximized by optimizing its degree distributions using the computed LLR pdf. The optimized degree distributions are given in Table. 5.1. After optimizing the degree distributions, the parity-check matrices of the codes are constructed by the PEG algorithm [82]. Finally, the codes are simulated on the channel at high SNR by assuming known block boundaries with the rest of parameters being the same as in Example 1. Fig. 5.12 shows the maximum p_{id} under which the constructed irregular LDPC codes perform with BER less than 10^{-5}. It is seen that these practically achievable rates are not far from the maximum achievable rates given by $I(d_{i,j}; l_{i,j})$. Also depicted in Fig. 5.12 are the results for three regular LDPC codes of the same length, i.e., $(3, 4)$-regular, $(3, 6)$-regular, and $(3, 12)$-regular LDPC codes with rates 0.25, 0.50, and 0.75, respectively.

Now, consider Example 2. Using the same method, the maximum achievable information rates of the watermarked system (by using (5.17)) are compared to those of the two benchmark systems (Gray labeled 16-QAM in Fig. 5.5a and quasi Gray 32-QAM) and $C_{\text{i.u.d.}}$ in Fig. 5.13. The block size is again kept fixed at $10,012$ symbols. The same discussion as in Example 1 applies to this case as well.

We also compare the maximum achievable information rates with the bounds

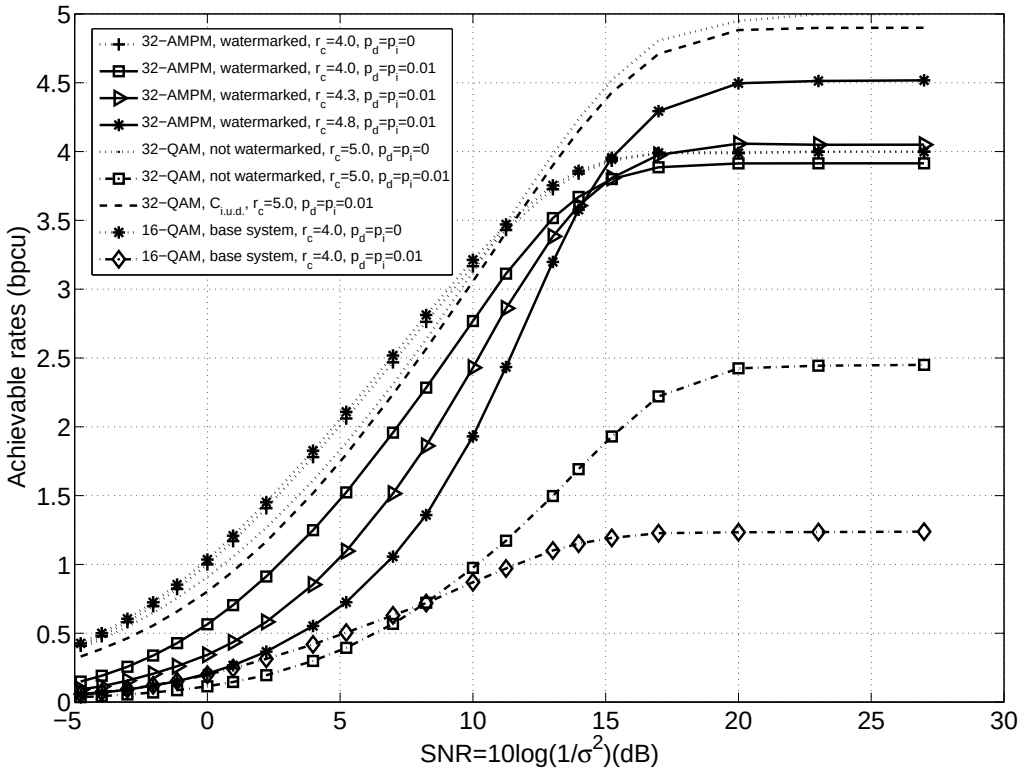


Figure 5.13: Maximum achievable information rates (bits per channel use) versus SNR under different modulations assuming a 16-QAM base system ($r_{\mathrm{c}} = 4.0$). The 32-AM/PM watermarked system mentioned in Section 5.3.1 has $r_{\mathrm{c}} = 4.0$. The maximum achievable rates given by (5.17) under quasi-Gray 32-QAM modulation with no watermark ($r_{\mathrm{c}} = 5.0$), and under two 32-AM/PM watermarked system with partial watermarking ($r_{\mathrm{c}} = 4.3$ and 4.8) mentioned in Section 5.5 are plotted for comparison. Also, $C_{\mathrm{i.u.d.}}$ is plotted for the 32-QAM modulation.

available for q-ary synchronization codes. We use the asymptotic bounds for q-ary codes of [4] to compare with our scheme. These bounds are achieved by considering the Levenshtein distance [56] between q-ary codewords and enumerating the maximum size of codes capable of correcting insertions/deletions with *zero* error probabilities. Since these bounds do not consider substitution errors or additive noise, we compare them with the achievable rates of our scheme in the high SNR region.

Figs. 5.14 and 5.15 compare the upper and lower bounds of q-ary codes for $q = 8, 32$, with the achievable information rates of our 8-PSK and 32-AM/PM schemes using (5.17), $C_{\text{i.u.d.}}$. Fig. 5.14 also includes the practical rates achievable by the optimized irregular LDPC codes of Fig. 5.12. Notice that $C_{\text{i.u.d.}}$ and our achievable information rates in some regions exceed the upper bound of [4]. This is due to the fact that the upper bounds are computed assuming zero error probabilities whereas $C_{\text{i.u.d.}}$ and our achievable rates given by (5.17) are computed assuming asymptotically vanishing error probabilities and thus computed in different scenarios. As seen on the figure, the achievable rates of our watermarked system is below $C_{\text{i.u.d.}}$ and in some regions below the q-ary upper bound. Nevertheless, no code exists in the literature which can approach these limits. At small p_{id}, the q-ary codes can theoretically achieve higher information rates than our scheme. This suggests that it is not efficient to dedicate one whole bit to the watermark when the number of synchronization errors is small. We will show in the next section how the information rates can be increased.

5.5 Increasing the achievable information rates

In this section, we show how the maximum achievable rates of the watermarked system can be improved.

We defined r_{c} to be the average number of bits assigned to the information bits (more precisely coded bits) per each transmitted symbol. Until now, we considered cases where one bit was assigned to the binary watermark in each of the transmitted symbols. For example, for the 4-PSK base system and the 8-PSK watermarked system discussed in Example 1, $r_{\text{c}} = 2.0$ bits. It is also possible to embed watermark bits into only some of the symbols but not all of them. For 8-PSK, this means $2.0 < r_{\text{c}} < 3.0$. We use Gray mapping for those symbols which are not watermarked (see Fig. 5.16a). Also, we scatter those symbols which contain watermark uniformly

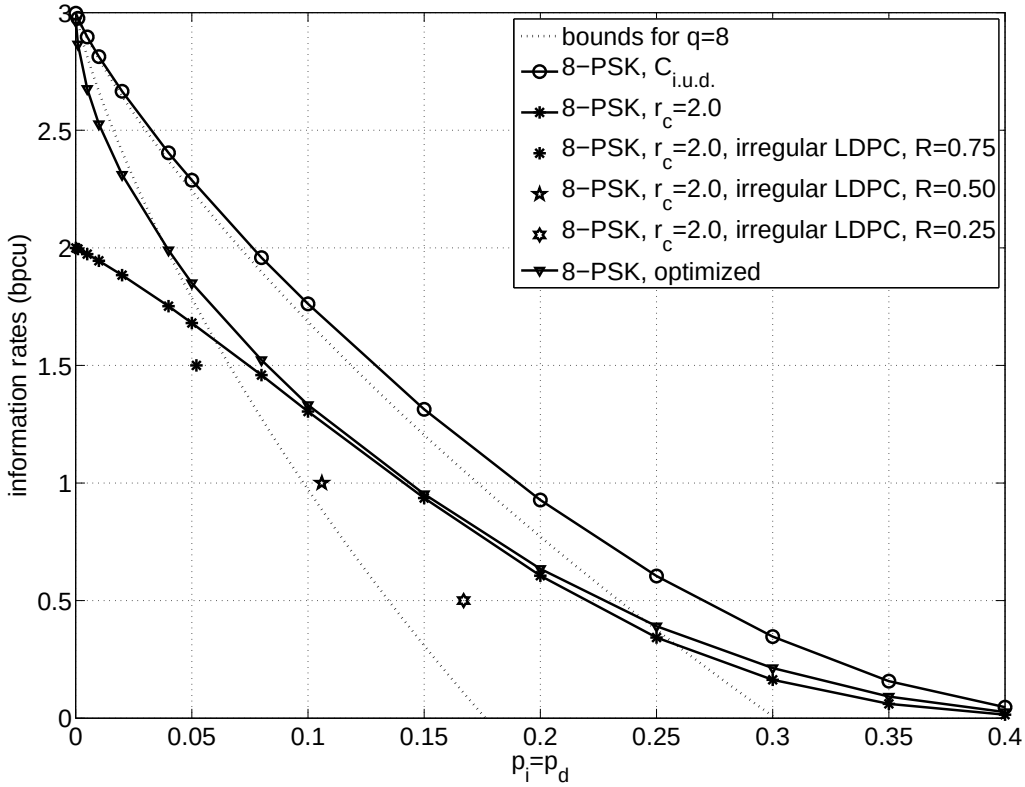


Figure 5.14: Maximum achievable information rates of the 8-PSK watermarked system at high SNR are compared to the upper and lower bounds of 8-ary insertion/deletion correcting codes [4], the achievable rates $C_{\mathrm{i.u.d.}}$, and practical rates achievable by the optimized irregular LDPC codes of Fig. 5.12. Achievable rates are plotted for the 8-PSK watermarked system in two cases. First, using binary watermark and assigning one bit to the watermark in each symbol ($r_{\mathrm{c}} = 2.0$). Second, the achievable rates are maximized by optimizing q_{w} and r_{c} in each point.

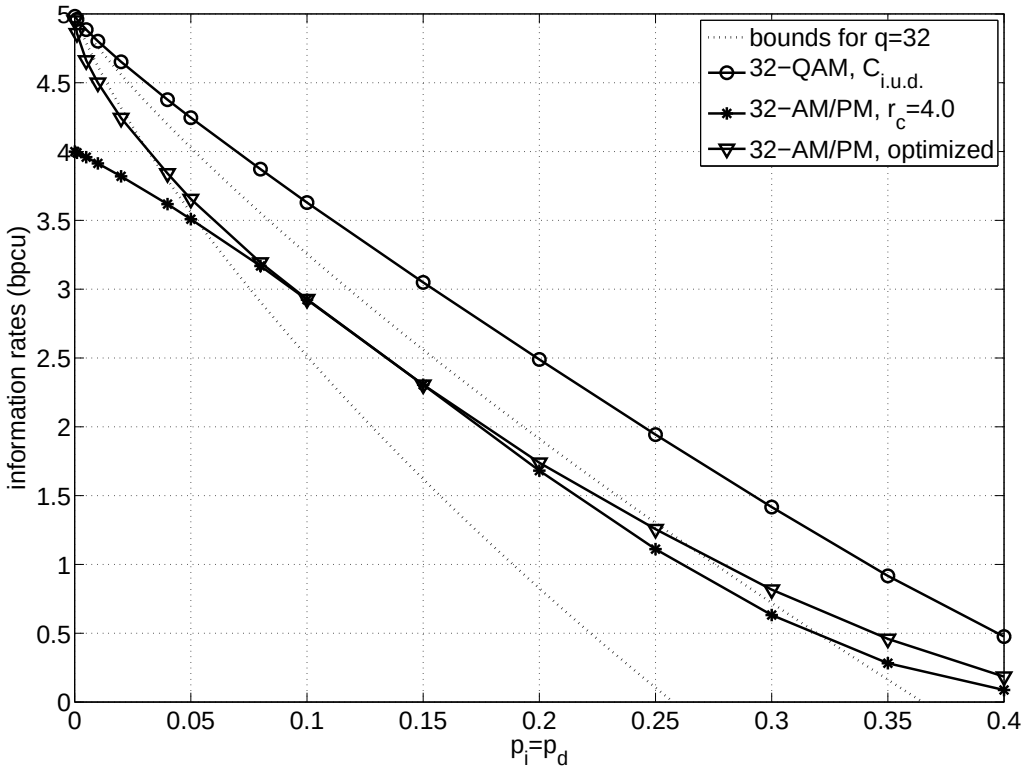


Figure 5.15: Maximum achievable information rates of the 32-AM/PM watermarked system at high SNR are compared to the upper and lower bounds of 32-ary insertion/deletion correcting codes [4] and the achievable rates $C_{\mathrm{i.u.d.}}$. These rates are plotted for the 32-AM/PM watermarked system in two cases. First, using binary watermark and assigning one bit to the watermark in each symbol ($r_{\mathrm{c}} = 4.0$). Second, the achievable rates are maximized by optimizing q_{w} and r_{c} in each point.

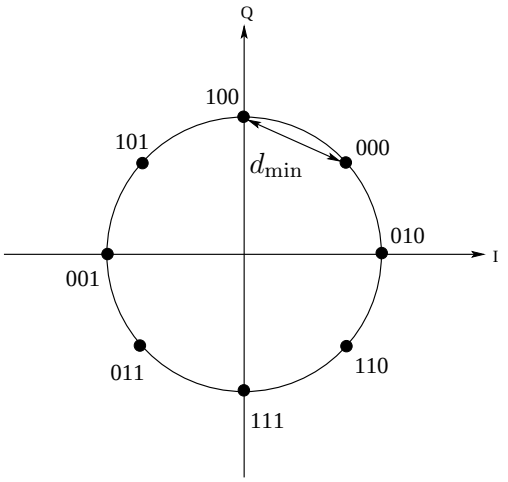
in the transmitted block.

It is evident that there is a trade-off between r_c and the system ability to recover synchronization errors. Increasing r_c potentially lets more information to pass through the channel but at the same time increases the system vulnerability to synchronization errors since less bits are assigned to the watermark. As a result, for a fixed signal set, there exists an optimum r_c for each p_{id} and SNR which provides the largest transmission rate on the channel.

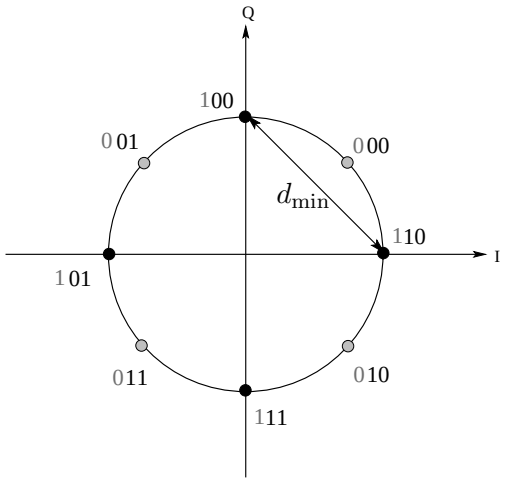
For example consider the system of Example 1. At high SNR and $p_{id} = 0.01$, the maximum achievable rate is 1.945 bits per channel use when $r_c = 2.0$ bits which increases to 2.528 bits per channel use when $r_c = 2.8$ bits. This implies that dedicating one bit in every symbol to the watermark, i.e., $r_c = 2.0$ is wasteful in this case. A better protection is provided against synchronization errors by assigning one bit to the watermark in only 20% of the symbols. As examples, Fig. 5.11 also shows the maximum achievable rates of the 8-PSK system under $r_c = 2.3$ and $r_c = 2.8$. For $p_{id} = 0.01$, when SNR<6.44 dB the system with $r_c = 2.0$ achieves higher rates compared to those of systems with $r_c = 2.3$ and $r_c = 2.8$. When $6.44<$SNR<9.31, $r_c = 2.3$ provides the largest rates and when SNR>9.31 dB, $r_c = 2.8$ provides the largest rates compared to the other two systems. Fig. 5.13 depicts the comparison for the 16-QAM and 32-AM/PM systems.

Until now, we have only considered binary watermark sequences. When the number of synchronization errors is large, a binary watermark may not be very helpful in localizing these errors. Increasing the alphabet size of the watermark q_w increases the system ability to combat synchronization errors. Increasing q_w, however decreases r_c, i.e., less number of bits are available in each symbol for information bits. As a result, there is a trade-off between the q_w, r_c, and the maximum achievable information rate on the channel. Fig. 5.16 illustrates the constellations and their labeling for the 8-PSK watermarked system with both $q_w = 2$ and $q_w = 4$.

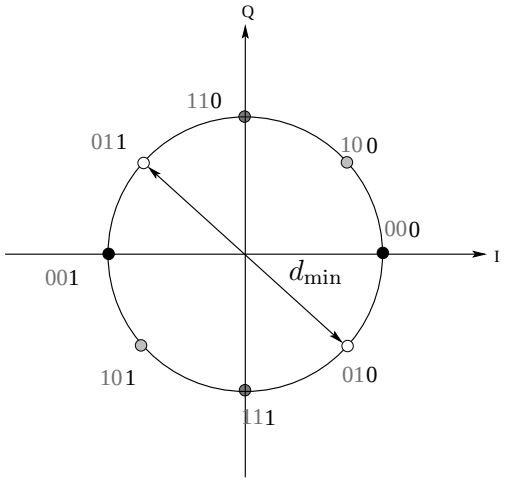
Figs. 5.14 and 5.15 depict the maximum achievable rates that 8-PSK and 32-AM/PM watermarked systems can achieve by finding the optimum r_c and q_w at each point. It is evident that the maximum achievable rates can be increased significantly by this strategy. This is especially beneficial when p_{id} is small where the achievable rates gets closer to the lower bound on q-ary codes.



(a) Gray labeled constellation for symbols without watermark



(b) Constellation for symbols with binary watermark, $q_{\mathrm{w}} = 2$



(c) Constellation for symbols with 4-ary watermark, $q_{\mathrm{w}} = 4$

Figure 5.16: Signal constellations and their labeling for the 8-PSK watermarked system and the technique of Sec. 5.5.

5.6 Complexity and the watermark sequence

5.6.1 Decoding complexity

The complexity of the forward-backward algorithm determines the complexity of watermark decoding. This complexity scales as $O\left((1+2t_{\max})IMN\right)$, where $1+2t_{\max}$ is the number of states in the HMM and N is equal to the number of symbols on which the forward-backward algorithm is performed. It should be noted that it is possible to reduce this complexity using arguments similar to those of [1].

5.6.2 Watermark sequence

The watermark sequences used in this chapter are pseudo-random sequences. Our experiments confirm that these sequences perform well under different insertion and deletion rates. Periodic sequences with small periods are usually not good choices. First, they are vulnerable to successive insertios/deletions. As a simple example, if the watermark is a periodic sequence with period 4, then the decoder cannot detect 4 successive deletions. Furthermore, certain patterns of insertions/deletions can fool the decoder such that it fails to detect them. Randomness lowers the probability of false detecting or missing insertions/deletions by the decoder.

Among other factors which affect the decoding performance is the number of successive identical symbols (runs) in the watermark. One advantage of having runs is that it provides the ability to detect successive deletions. The larger the run-length, the larger successive deletions that can be detected. Nevertheless, larger runs lead to a worse localization of insertions/deletions as the decoder is not able to detect where exactly insertions/deletions have occurred. This gives rise to less reliable LLRs in the vicinity of insertions/deletions. Thus, there should be a balance between large and small runs in the watermark sequence. Pseudo-random sequences usually have this property.

Among candidates for the watermark are the run-length limited (RLL) sequences. RLL sequences with small maximum run-lengths (e.g., around 3 or 4) are good choices particularly when p_{id} is very small since the probability of having successive insertions/deletions is small. The performance gain over pseudo-random watermarks is only notable at small p_{id} where high-rate outer codes are used.

The presence of additive noise can also affect the choice of watermark. All in all, it is possible that sequences with structure could offer better performance than

the above-mentioned sequences. This can be the subject of further investigation.

5.7 Conclusion

In this chapter, we proposed a concatenated coding scheme for reliable communication over non-binary I/D channels where symbols were randomly deleted from or inserted in the received sequence and all symbols were corrupted by additive white Gaussian noise. First, we provided redundancy by expanding the symbol set while maintaining almost the same minimum distance. Then, we allocated part of the bits associated with each symbol to watermark bits. The watermark sequence, known at the receiver, was then used by the forward-backward algorithm to provide soft information for the outer code. Finally, the received sequence was decoded by the outer code.

We evaluated the performance of the watermarked system and through numerical examples we showed that significant amount of insertions and deletions could be corrected by the proposed method. The maximum information rates achievable by this method on the I/D channel were provided and compared with existing results and the available bounds on q-ary synchronization codes. Practical codes were also designed that could approach these information rates.

Chapter 6

Conclusion

In this chapter, we summarize our contributions and results and conclude this dissertation. Some new questions and possible future research directions are also discussed.

6.1 Summary of contributions

In light of the discovery of modern error-correcting codes and their efficient soft-decision decoding algorithms in the past two decades, close to the Shannon limit communication has been promised on many channels. Nevertheless, there are still numerous practical challenges facing the communication system designer. To gain benefit of the effectiveness of modern soft-decision decoding algorithms, information about the channel parameters should be perfectly known at the receiver. The performance of these algorithms is greatly degraded when channel parameters estimations are not perfect at the receiver. Noticing that these estimations are normally imperfect and they are also disadvantageous in terms of overhead, complexity, and power dissipation, other solutions should be sought. In this dissertation, we considered the problem of reliable communication under limited knowledge of the channel parameters at the receiver. Since each communication channel demands a different treatment and also the knowledge of each of channel parameters poses different challenges to the design of the system, we divided the problem into three important scenarios.

6.1.1 Robust decoding on Gaussian channels with unknown noise power

Many wired or wireless communication channels can be modeled by the additive white Gaussian noise channel. It has been shown that modern error-correcting codes such as turbo codes and LDPC codes work very close to the Shannon limit on this channel. This is mainly due to the existence of efficient and powerful soft-decoding algorithms such as iterative message-passing algorithms. For successful decoding, normally the noise power should be perfectly known at the decoder. Usually, this information is not available at the receiver because it is time-varying and the estimations are not perfect.

Considering LDPC codes and assuming an imperfect estimation of the noise power at the receiver, we proposed a hybrid decoding algorithm combining the sum-product and min-sum message-passing algorithms. The decoder was called an irregular decoder. We showed that a robust performance can be achieved under imperfect estimation of the noise power by using the proposed decoding method. Since the parameters of the designed irregular decoder depended on the LDPC code used on the channel, we also designed irregular code-decoder pairs with optimum performance in the presence of channel estimation errors.

6.1.2 Practical decoding on wireless channels with unknown fading gain

In the second scenario, we focused on wireless channels which were statistically modeled by a fading gain and additive white Gaussian noise. We stated that the instantaneous fading gain which captures the time-varying nature of the channel was hard to be perfectly tracked at the receiver. As a result, the decoder should usually deal with cases where the knowledge of the channel fading gain is not available.

For efficient decoding of modern error-correcting codes such as LDPC codes, some soft metrics should be calculated at the receiver which are usually in the form of LLRs. LLR computation is generally complicated on fading channels when the channel fading gain is not perfectly known. This problem is further intensified when non-binary modulations are used. In this scenario, we considered flat-fading channels where the absolute value of the fading gain and/or the additive noise power were unknown at the receiver.

We proposed an approximate LLR calculation technique by first justifying the

previously proposed LLR accuracy measure for binary-input symmetric output channels. In the next step, we generalized the accuracy measure to binary-input asymmetric output channels. Using BICM, we then applied the LLR accuracy measure to non-binary signalling and proposed an optimization technique to optimize the parameters of a general LLR approximating function. Optimized piecewise linear LLR approximating functions were then presented and through using LDPC-coded BICM it was shown that performance loss under the approximate LLRs was very small. Furthermore, we showed that the capacity of BICM under true LLRs can be approached by optimizing irregular LDPC codes under the proposed approximate LLRs. Although we demonstrated our results through using LDPC codes, our approaches are more general and can also be applied to other coding schemes which work with LLRs.

6.1.3 Practical coding for channels with imperfect timing

In the third scenario, we considered coding on channels with imperfect synchronization between the transmitter and receiver. This was the most challenging scenario since most of the error-correcting codes in the literature are designed assuming perfect synchronization. Imperfect timing information at the receiver leads to symbols being inserted in or deleted from the received sequence. Since the positions of insertions/deletions are not known at the receiver, even an uncorrected insertion/deletion can lead to a catastrophic burst of errors. As a result, conventional error-correcting codes fail at these situations. We showed that channels with imperfect synchronization can be modeled by insertion/deletion channels with additive noise. Although insertion/deletion channels have a long history, their analysis and finding suitable error-correcting codes for them have proven to be extremely challenging. Finding the capacity of these channels is also an open problem.

For this scenario, we considered non-binary insertion/deletion channels where symbols were randomly inserted/deleted and all received symbols were corrupted by additive white Gaussian noise. For this channel model, we proposed codes capable of correcting insertions, deletions and the effect of noise without sacrificing the transmission resources. For this purpose, we first proposed to provide redundancy by expanding the modulation signal set. We then used these redundancies to embed a watermark sequence known both to the transmitter and receiver. Since the watermark sequence was separated from the information sequence by using the presented

symbol mapping, the receiver was able to efficiently deduce insertions/deletions. We presented a forward-backward algorithm for decoding at the receiver and evaluated the performance of the system in terms of bit and block error rates and maximum achievable rates on the channel. It was shown that significant amount of insertions/deletions could be corrected using this method.

We also compared the achievable rates of our watermarked system with the available bounds in the literature and designed practical codes capable of approaching these achievable rates. Furthermore, we showed how the achievable rates can be increased by using partial watermarking and/or non-binary watermark sequences.

6.2 Possible future research

6.2.1 Estimating the insertion/deletion channel parameters

For the insertion/deletion channel discussed in Chapter 5, we characterized a hidden Markov model (HMM) for the watermark decoder. Notice that by running the forward-backward algorithm, quantities other than LLRs can also be found. Among the most interesting quantities which can be found are the parameters of the HMM which include insertion and deletion probabilities p_{i} and p_{d}. As a result, by running the forward-backward algorithm the parameters of the insertion/deletion channel can be estimated. This is particularly useful for cases where these parameters are not perfectly known at the receiver or they change in time. The estimated parameters can be then used by the forward-backward algorithm to compute refined LLRs. It is also possible to use training symbols at the transmitter such that the receiver is able to find accurate estimates of the channel parameters.

6.2.2 More realistic channel models for imperfect timing

The channel model described in Chapter 5 and other existing channel models which represent lack of timing information are not always realistic. The model we used in this dissertation considers random symbol insertions and deletions. However, when the insertions and deletions are resulted from imperfect timing, they are highly correlated. For example, it cannot be imagined that a 0 is randomly inserted between two 1's. Also, when one symbol is deleted it becomes less likely for the next symbol to be deleted too.

Another issue which should be considered is the additive noise resulted from imperfect timing. When no perfect timing information exists at the receiver, the

samples are not taken at the best time instances and this leads to extra additive noise. The existing channel models do not consider this effect. Thus, more realistic channel models representing imperfect timing can be studied by considering the existing correlations between symbol insertions and deletions and the extra additive noise resulted from imperfect sampling instances.

After obtaining realistic channel models, the approaches presented in Chapter 5 can be modified to suit the new models. Usually this can be done by modifying the structure of the HMM.

6.2.3 Improved codes for correlated insertions/deletions

Having obtained more realistic channel models for imperfect timing which captures the correlations between insertions/deletions, it might be possible to find better codes for dealing with these correlated insertions/deletions. It is worth mentioning that the performance of the watermarked system of Chapter 5 already improves under correlated insertions/deletions. Nevertheless, it is possible to find better codes or better watermark sequences for these cases.

6.2.4 Watermark decoding as a timing recovery technique

After finding more realistic channel models and improved watermark codes, it is possible to use the watermark decoding scheme as a timing recovery technique. This is because the proposed watermark decoding system can deal with synchronization errors and has the potential to provide timing information at the receiver. Thus, watermark decoding can be used on a realistic system working with continuous-time signals to obtain sampling instances, do the sampling, and find estimates of the transmitted symbols.

Bibliography

[1] M. C. Davey and D. J. C. Mackay, "Reliable communication over channels with insertions, deletions, and substitutions," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 687–698, Feb. 2001.

[2] E. A. Ratzer, "Marker codes for channels with insertions and deletions," *Annals of telecommunications*, vol. 60, no. 1-2, pp. 29–44, 2005.

[3] J. Briffa, H. Schaathun, and S. Wesemeyer, "An improved decoding algorithm for the Davey-MacKay construction," in *Proc. of IEEE Intl. Conf. on Commun.*, May 2010, pp. 1–5.

[4] V. Levenshtein, "Bounds for deletion/insertion correcting codes," in *Proc. 2002 IEEE Intl. Symp. on Inf. Theory*, 2002, p. 370.

[5] J. Barry, E. A. Lee, and D. G. Messerschmitt, *Digital Communications*, 3rd ed. Kluwer Academic Publishers, 2004.

[6] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes (1)," in *Proc. 1993 IEEE Int. Conf. Commun.*, vol. 2, Geneva, Switzerland, May 1993, pp. 1064–1070.

[7] R. G. Gallager, "Low-Density Parity-Check codes," Ph.D. dissertation, M.I.T press, Cambridge, MA, 1963.

[8] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.

[9] R. Yazdani and M. Ardakani, "Robust LDPC decoding using irregular decoders," *IEEE Commun. Lett.*, vol. 12, no. 12, pp. 888–890, Dec. 2008.

[10] A. Goldsmith, *Wireless Communications*. Cambridge University Press, 2005.

[11] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[12] R. Yazdani and M. Ardakani, "Linear LLR approximation for iterative decoding on wireless channels," *IEEE Trans. Commun.*, vol. 57, no. 11, pp. 3278–3287, Nov. 2009.

[13] ——, "Efficient LLR calculation for non-binary modulations over fading channels," *IEEE Trans. Commun.*, vol. 59, no. 5, May 2011.

[14] ——, "Piecewise linear LLR approximation for non-binary modulations over Gaussian channels with unknown noise variance," in *Proc. 2010 IEEE Int. Conf. on Telecommun. (ICT)*, Doha, Qatar, Apr. 2010, pp. 1–7.

[15] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423 and 623–656, Jul. and Oct. 1948.

[16] M. Mitzenmacher, "A survey of results for deletion channels and related synchronization channels," *Probability Surveys*, vol. 6, pp. 1–33, 2009.

[17] H. Mercier, V. Bhargava, and V. Tarokh, "A survey of error-correcting codes for channels with symbol synchronization errors," *IEEE Communications Surveys Tutorials*, vol. 12, no. 1, pp. 87–96, 2010.

[18] R. Yazdani and M. Ardakani, "Reliable communication over non-binary insertion/deletion channels," *IEEE Trans. Commun.*, Jul. 2012, accepted for publication.

[19] R. L. Dobrushin, "Shannon's theorems for channels with synchronization errors," *Probl. Inf. Transm.*, vol. 3, no. 4, pp. 11–26, 1967.

[20] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* Wiley-Interscience, 2006.

[21] R. W. Hamming, "Error Detecting and Error Correcting Codes," *Bell System Technical Journal*, vol. 26, no. 2, pp. 147–160, 1950.

[22] S. Lin and D. J. Costello, *Error Control Coding, Second Edition.* Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2004.

[23] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of Low Density Parity Check codes," *Electronics Lett.*, vol. 32, no. 18, pp. 1645–1646, August 1996.

[24] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, pp. 498–519, Feb. 2001.

[25] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 585–598, Feb. 2001.

[26] N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation, Linköping University, Sweden, 1996.

[27] S.-Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the design of low-density perity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, no. 2, pp. 58–60, Feb. 2001.

[28] M. Ardakani, T. H. Chan, and F. R. Kschischang, "EXIT-chart properties of the highest-rate LDPC code with desired convergence behavior," *IEEE Commun. Lett.*, vol. 9, no. 1, pp. 52–54, Jan. 2005.

[29] W. Yu, M. Ardakani, B. Smith, and F. Kschischang, "Complexity-optimized low-density parity-check codes for Gallager decoding algorithm B," in *Proc. IEEE Int. Symp. on Inf. Theory*, Sep. 2005, pp. 1488–1492.

[30] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Trans. Inf. Theory*, vol. 28, no. 1, pp. 55–67, Jan. 1982.

[31] G. Caire, G. Taricco, and E. Biglieri, "Bit-interleaved coded modulation," *IEEE Trans. Inf. Theory*, vol. 44, no. 3, pp. 927–946, May 1998.

[32] P. Zarrinkhat, A. H. Banihashemi, and H. Xiao, "Time-invariant and switch-type hybrid iterative decoding of low-density parity-check codes," *Ann. Télécommun. / Ann. Telecommun.*, no. 1-2, pp. 99–127, Jan.-Feb. 2005.

[33] M. Ardakani, P. Zarrinkhat, and R. Yazdani, "Complexity-optimized irregular decoders," in *Proc. 2006 IEEE Int. Symp. on Inf. Theory*, Seattle, USA, 2006, pp. 2393–2397.

[34] P. Zarrinkhat and M. Ardakani, "On the robustness of iterative decoders," in *23rd Biennial Symp. on Commun.*, Queen's Univ., Kingston, Canada, May-Jun. 2006, pp. 240 – 243.

[35] H. Saeedi and A. H. Banihashemi, "Performance of belief propagation for decoding LDPC codes in the presence of channel estimation error," *IEEE Trans. Commun.*, vol. 55, pp. 83–89, Jan. 2007.

[36] ——, "Design of irregular LDPC codes for BIAWGN channels with SNR mismatch," *IEEE Trans. Commun.*, vol. 57, pp. 6–11, Jan. 2009.

[37] J. Chen and M. Fossorier, "Near optimum universal belief propagation based decoding of low-density parity-check codes," *IEEE Trans. Commun.*, vol. 50, pp. 406–414, Mar. 2002.

[38] A. Abedi and A. K. Khandani, "Invariance properties of binary linear codes over a memoryless channel with discrete input," *IEEE Trans. Inf. Theory*, vol. 53, no. 3, pp. 1215–1218, Mar. 2007.

[39] P. A. Hoeher, I. Land, and U. Sorger, "Log-likelihood values and Monte Carlo simulation: some fundamental results," in *Proc. Int. Symp. on Turbo Codes and Related Topics*, Brest, France, Sep. 2000.

[40] J. Hagenauer, "Viterbi decoding of convolutional codes for fading- and burst-channels," in *Proc. Zurich seminar on digital communications*, 1980.

[41] J. Hou, P. H. Siegel, and L. B. Milstein, "Performance analysis and code optimization of low-density parity-check codes on Rayleigh fading channels," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 5, pp. 924–934, May 2001.

[42] F. Tosato and P. Bisaglia, "Simplified soft-output demapper for binary interleaved COFDM with application to HIPERLAN/2," in *Proc. of the 2002 IEEE Int. Conf. on Commun. (ICC'02)*, 2002, pp. 664–668.

[43] J. K. Kwon, S. Park, and D. K. Sung, "Log-likelihood ratio (LLR) conversion schemes in orthogonal code hopping multiplexing," *IEEE Commun. Lett.*, vol. 7, no. 3, pp. 104–106, Mar. 2003.

[44] ETSI TS 101 475, "Broadband Radio Access Networks (BRAN); HIPERLAN Type 2; Physical (PHY) layer, v1.2.2," 2001.

[45] R. Yazdani, "Analysis and decoding of LDPC codes under some practical considerations," Master's thesis, University of Alberta, Edmonton, Canada, 2007.

[46] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes," *IEEE Trans. Inf. Theory*, vol. 47, pp. 569–584, 2001.

[47] J. Hou, P. H. Siegel, L. B. Milstein, and H. D. Pfister, "Capacity-approaching bandwidth-efficient coded modulation schemes based on low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 49, no. 9, pp. 2141–2155, Sep. 2003.

[48] O. Etesami and A. Shokrollahi, "Raptor codes on binary memoryless symmetric channels," *IEEE Trans. Inf. Theory*, vol. 52, no. 5, pp. 2033–2051, 2006.

[49] T. J. Richardson and R. Urbanke, *Modern Coding Theory.* Cambridge University Press, 2008.

[50] A. Sanaei and M. Ardakani, "LDPC code design considerations for non-uniform channels," *IEEE Trans. Commun.*, vol. 58, no. 1, pp. 101–109, Jan. 2010.

[51] A. Martinez, A. Guillen i Fabregas, G. Caire, and F. Willems, "Bit-interleaved coded modulation revisited: a mismatched decoding perspective," *IEEE Trans. Inf. Theory*, vol. 55, no. 6, pp. 2756–2765, Jun. 2009.

[52] G. Kaplan and S. Shamai (Shitz), "Information rates of compound channels with application to antipodal signaling in a fading environment," *AEU Int. J. Electron. Commun.*, vol. 47, no. 4, pp. 228–239, 1993.

[53] 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. Inf. Theory*, vol. 47, no. 2, pp. 657–670, 2001.

[54] M. Ardakani and F. R. Kschischang, "A more accurate one-dimensional analysis and design of irregular LDPC codes," *IEEE Trans. Commun.*, vol. 52, no. 12, pp. 2106–2114, 2004.

[55] F. F. Sellers Jr., "Bit loss and gain correction code," *IRE Trans. Inf. Theory*, vol. IT-8, pp. 35–38, Jan. 1962.

[56] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady*, vol. 10, no. 8, pp. 707–710, Feb. 1966.

[57] G. M. Tenengolts, "Class of codes correcting bit loss and errors in the preceding bit," *Automation and Remote Control*, vol. 37, no. 5, pp. 797–802, May 1976.

[58] A. S. J. Helberg, "Coding for the correction of synchronization errors," Ph.D. dissertation, Fac. Eng., Rand Afrikaans Univ., Nov. 1993.

[59] K. Saowapa, H. Kaneko, and E. Fujiwara, "Systematic binary deletion/insertion error-correcting codes capable of correcting random bit errors," *IEICE Trans. Fundamentals of Electronics, Communications and Computer Science*, vol. E83-A, no. 12, pp. 2699–2705, 2000.

[60] A. Helberg and H. Ferreira, "On multiple insertion/deletion correcting codes," *IEEE Trans. Inf. Theory*, vol. 48, no. 1, pp. 305 –308, Jan. 2002.

[61] L. Calabi and W. E. Hartnett, "A family of codes for the correction of substitution and synchronization errors," *IEEE Trans. Inf. Theory*, vol. 15, no. 1, pp. 102–106, Jan. 1969.

[62] E. Tanaka and T. Kasai, "Synchronization and substitution error-correcting codes for the levenshtein metric," *IEEE Trans. Inf. Theory*, vol. 22, no. 2, pp. 156–162, Mar. 1976.

[63] G. M. Tenengolts, "Nonbinary codes correcting single deletion or insertion," *IEEE Trans. Inf. Theory*, vol. 30, no. 5, pp. 766–769, Sep. 1984.

[64] V. I. Levenshtein, "On perfect codes in deletion and insertion metric," *Discrete Mathematics and Applications*, vol. 2, no. 3, pp. 241–258, 1992.

[65] J. Yin, "A combinatorial construction for perfect deletion-correcting codes," *Designs, Codes and Cryptography*, vol. 23, pp. 99–110, 2001.

[66] A. Klein, "On perfect deletion-correcting codes," *J. Combinatorial Designs*, vol. 12, no. 1, 2004.

[67] J. Wang and J. Yin, "Constructions for 5-deletion-correcding codes of length 7," *IEEE Trans. Inf. Theory*, vol. 52, no. 8, pp. 3676–3685, 2006.

[68] M. Mitzenmacher, "Polynomial time low-density parity-check codes with rates very close to the capacity of the q-ary random deletion channel for large q," *IEEE Trans. Inf. Theory*, vol. 52, pp. 2552–2557, 2006.

[69] S. Golomb, B. Gordon, and L. R.Welch, "Comma-free codes," *Can. J. Math.*, vol. 10, no. 2, pp. 202–209, 1958.

[70] J. J. Stiffler, "Comma-free error-correcting codes," *IEEE Trans. Inf. Theory*, vol. IT-11, pp. 107–111, Jan. 1965.

[71] S. E. Tavares and M. Fukada, "Matrix approach to synchronization recovery for binary cyclic codes," *IEEE Trans. Inf. Theory*, vol. IT-15, pp. 93–101, Jan. 1969.

[72] T. R. Hatcher, "On a family of error-correcting and synchronizable codes," *IEEE Trans. Inf. Theory*, vol. IT-15, pp. 620–624, Sep. 1969.

[73] H. Morita, A. J. van Wijngaarden, and A. J. H. Vinck, "Prefix-synchronized codes capable of correcting single insertion/deletion errors," in *Proc. IEEE Int. Symp. on Infor. Theory*, Ulm, Germany, Jun. 1997, p. 409.

[74] L. J. Schulman and D. Zuckerman, "Asymptotically good codes correcting insertions, deletions, and transpositions," *IEEE Trans. Inf. Theory*, vol. 45, pp. 2552–2557, Nov. 1999.

[75] J. Chen, M. Mitzenmacher, C. Ng, and N. Varnica, "Concatenated codes for deletion channels," in *Proc. 2003 IEEE Int. Symp. on Inf. Theory*, 2003, pp. 218–218.

[76] V. Buttigieg and J. Briffa, "Codebook and marker sequence design for synchronization-correcting codes," in *Proc. of IEEE Int. Symp. on Inf. Theory*, Aug. 2011, pp. 1579–1583.

[77] F. Wang, D. Fertonani, and T. Duman, "Symbol-level synchronization and LDPC code design for insertion/deletion channels," *IEEE Trans. Commun.*, vol. 59, no. 5, pp. 1287–1297, May 2011.

[78] J. Hu, T. Duman, M. Erden, and A. Kavcic, "Achievable information rates for channels with insertions, deletions, and intersymbol interference with i.i.d. inputs," *IEEE Trans. Commun.*, vol. 58, no. 4, pp. 1102–1111, Apr. 2010.

[79] J. Barry, A. Kavcic, S. LcLaughlin, A. Nayak, and W. Zeng, "Iterative timing recovery," *IEEE Signal Process. Mag.*, vol. 21, no. 1, pp. 89–102, jan. 2004.

[80] L. Rabiner and B. Juang, "An introduction to hidden Markov models," *IEEE ASSP Magazine*, vol. 3, no. 1, pp. 4–16, Jan. 1986.

[81] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate (corresp.)," *IEEE Trans. Inf. Theory*, vol. 20, no. 2, pp. 284–287, Mar. 1974.

[82] X.-Y. Hu, E. Eleftheriou, and D. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Trans. Inf. Theory*, vol. 51, no. 1, pp. 386–398, Jan. 2005.

[83] R. G. Gallager, "Sequential decoding for binary channels with noise and synchronization errors," Lincoln lab group report, Tech. Rep. 2502, 1961.

[84] K. S. Zigangirov, "Sequential decoding for a binary channel with drop-outs and insertions," *Probl. Pered. Inform.*, vol. 5, no. 2, pp. 23–30, 1969.

[85] S. Diggavi and M. Grossglauser, "Information transmission over a finite buffer channels," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 1226–1237, Mar. 2006.

[86] E. Drinea and M. Mitzenmacher, "On lower bounds for the capacity of deletion channels," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4648–4657, Oct. 2007.

[87] D. Fertonani, T. Duman, and M. Erden, "Bounds on the capacity of channels with insertions, deletions and substitutions," *IEEE Trans. Commun.*, vol. 59, no. 1, pp. 2–6, 2011.

Appendix A

Proof of Theorem 4.1

Here, we prove that (4.8) is maximized with true LLRs and no symmetric LLR approximation can result in a $\hat{C} > C$. We first focus on the binary symmetric channel. Extension to other channels is discussed afterwards.

A.1 Discrete symmetric-output channels

A.1.1 Binary symmetric channel

Consider a binary symmetric channel (BSC). We have $x \in \{0,1\}$, $y \in \{0,1\}$ and $p(y=0|x=1) = p(y=1|x=0) = p$. Therefore, assuming that $x = 0$, the true LLR is

$$l = g(y) = (1-2y) \cdot \log \frac{1-p}{p}.$$

Thus, the LLRs have the following pdf

$$f_{\text{BSC}}(l) = (1-p) \cdot \delta\left(l - \log(\frac{1-p}{p})\right) + p \cdot \delta\left(l + \log(\frac{1-p}{p})\right)$$

where $\delta(l)$ is the Dirac delta function.

One can view $g(y)$ as a function which maps $y = 0$ to $\log \frac{1-p}{p}$ and $y = 1$ to $-\log \frac{1-p}{p}$. Now, consider a symmetric LLR approximation $\hat{g}(y)$, such that $\hat{g}(y)$ maps $y = 0$ to $\hat{l} = a_0$ and $y = 1$ to $\hat{l} = -a_0$. The approximated LLRs have the following pdf

$$\hat{f}_{\text{BSC}}(\hat{l}) = (1-p) \cdot \delta\left(\hat{l} - a_0\right) + p \cdot \delta\left(\hat{l} + a_0\right). \tag{A.1}$$

From (4.8) and (A.1)

$$\hat{C}_{\text{BSC}} = 1 - p \cdot \log_2(1 + e^{a_0}) - (1-p) \cdot \log_2(1 + e^{-a_0}).$$

By taking the first and the second derivatives of $\hat{C}_{\text{BSC}}$ with respect to a_0, it can be seen that $a_0 = \log \frac{1-p}{p}$ results in the maximum $\hat{C}_{\text{BSC}}$. Interestingly, this value of a_0

corresponds to true LLR calculation. Therefore,

$$\max\{\hat{C}_{\text{BSC}}\} = C_{\text{BSC}} = 1 + p\log_2(p) + (1-p)\log_2(1-p).$$

The second derivative of $\hat{C}_{\text{BSC}}$ with respect to a_0 is always negative. Therefore, starting at $a_0 = \log\frac{1-p}{p}$ (i.e., true LLRs), where $\Delta C_{\text{BSC}} = C_{\text{BSC}} - \hat{C}_{\text{BSC}} = 0$, if a_0 changes (in one direction), the larger $|a_0 - \log\frac{1-p}{p}|$ the larger ΔC_{BSC}. In other words, as the approximated LLRs become less accurate, the value of ΔC_{BSC} increases. This shows that the $\hat{C}_{\text{BSC}}$ can be used as a measure of accuracy of the approximated LLRs.

A.1.2 Other channels

The above discussion was limited to the BSC. Here we consider a symmetric channel whose input alphabet is binary and whose output alphabet is non-binary and discrete. Thus we have $x \in \{0,1\}$, $y \in \{\pm y_i, 1 \le i \le M\}$, $\forall i, 1 \le i \le M$. Let us define $p(y_i|x=1) = p(-y_i|x=0) = p_i$, $p(-y_i|x=1) = p(y_i|x=0) = p'_i$, and $q_i = p_i + p'_i$. Clearly $\sum_{i=1}^{M} q_i = 1$.

When a channel output $y = y_i$ is observed, the true LLR value is

$$l_i = \log\frac{p'_i}{p_i}. \tag{A.2}$$

To obtain (A.2), we have used $p(x=0) = p(x=1)$. Now, defining $\theta_i = \frac{p_i}{q_i}$, we have $\frac{p'_i}{q_i} = 1 - \theta_i$ and thus

$$g(y_i) = l_i = \log\frac{1-\theta_i}{\theta_i}.$$

The LLR distribution under the all-zero codeword assumption is therefore

$$f_L(l) = \sum_{i=1}^{M} q_i \left[(1-\theta_i)\cdot\delta\left(l - \log\frac{1-\theta_i}{\theta_i}\right) + \theta_i\cdot\delta\left(l + \log\frac{1-\theta_i}{\theta_i}\right)\right],$$

Also, considering $\hat{g}(y_i) = a_i$, the pdf of the approximated LLRs is

$$f_{\hat{L}}(\hat{l}) = \sum_{i=1}^{M} q_i \left((1-\theta_i)\cdot\delta(\hat{l} - a_i) + \theta_i\cdot\delta(\hat{l} + a_i)\right),$$

The discrete channel, therefore, can be viewed as M parallel BSCs. Each BSC is selected with a probability of q_i and has a crossover probability of θ_i. Assuming a capacity of C_i for each BSC and a $\hat{C}_i$ under a symmetric LLR approximation, from linearity of C and $\hat{C}$ in f_L and $f_{\hat{L}}$ respectively, it is evident that

$$C = \sum_i q_i C_i$$

and similarly

$$\hat{C} = \sum_i q_i \hat{C}_i.$$

Since all q_i are positive and because each $\hat{C}_i$ achieves its unique maximum when $a_i = \log \frac{1-\theta_i}{\theta_i}$, it becomes clear that the *unique* maximum of $\hat{C}$ under symmetric LLR approximation is achieved by true LLRs.

Now, consider an M dimensional space, whose coordinates are $a_1, a_2, \cdots, a_M$. True LLRs show one point of this space, where $\forall i, 1 \leq i \leq M$, $a_i = \log \frac{1-\theta_i}{\theta_i}$. Let us call this point P. Any line which passes through P is given by

$$\frac{a_1 - \log \frac{1-\theta_1}{\theta_1}}{\gamma_1} = \frac{a_2 - \log \frac{1-\theta_2}{\theta_2}}{\gamma_2} = \cdots = \frac{a_M - \log \frac{1-\theta_M}{\theta_M}}{\gamma_M},$$

where γ_i, $\forall i$, $1 \leq i \leq M$ shows the arbitrary vector representing the direction of the line. Starting from P and moving along any line which passes through P, getting farther from P increases $|a_i - \log \frac{1-\theta_i}{\theta_i}|$ for all i. As we showed for the BSC, $\Delta C_i = C_i - \hat{C}_i$ increases with increasing $|a_i - \log \frac{1-\theta_i}{\theta_i}|$. Thus, $\Delta C = C - \hat{C} = \sum_i q_i \Delta C_i$ also increases. Notice that on any such line a direct comparison between the accuracy of LLRs is possible. That is, as we get farther from P, the LLRs become less accurate. Consistently, $\Delta C = C - \hat{C}$ also increases. Thus, using $\hat{C}$ as a measure of accuracy of LLRs sounds reasonable.

A.2 Continuous symmetric-output channels

Now consider a continuous-output MBISO channel defined by its conditional pdf $f_{Y|X}(y|x)$ where $f_{Y|X}(-y|x = -1) = f_{Y|X}(y|x = +1) = f'_Y(y)$ and $f_{Y|X}(y|x = -1) = f_{Y|X}(-y|x = +1) = f_Y(y)$. Similar to the discrete channel case, we define $q_Y(y) = f_Y(y) + f'_Y(y)$ where $\int_0^\infty q_Y(y)\mathrm{d}y = 1$. When the channel output y is observed the true LLR value is

$$l = g(y) = \log \frac{f'_Y(y)}{f_Y(y)} = \log \frac{1 - \theta_Y(y)}{\theta_Y(y)},$$

where $\theta_Y(y) = \frac{f_Y(y)}{q_Y(y)}$. Thus, the true LLR pdf $f_L(l)$ is equal to

$$f_L(l) = \int_0^\infty q_Y(y) \Bigg[(1 - \theta_Y(y)) \delta \left(l - \log \frac{1 - \theta_Y(y)}{\theta_Y(y)} \right) + \theta_Y(y) \delta \left(l + \log \frac{1 - \theta_Y(y)}{\theta_Y(y)} \right) \Bigg] dy.$$

Now considering the approximate LLR calculation of $\hat{l} = \hat{g}(y)$ and the fact that probability of receiving y is independent of the LLR calculation, the pdf of the approximate LLR is

$$f_{\hat{L}}(\hat{l}) = \int_0^\infty q_Y(y)\Big[(1-\theta_Y(y))\delta\left(\hat{l}-\hat{g}(y)\right) + \theta_Y(y)\delta\left(\hat{l}+\hat{g}(y)\right)\Big]\mathrm{d}y. \tag{A.3}$$

By applying (A.3) in (4.8), we get

$$\begin{aligned}\hat{C} =& 1 - \int_{-\infty}^{\infty} \log_2(1+e^{-\hat{l}}) \int_0^\infty q_Y(y) \\ & \times \Big[(1-\theta_Y(y))\delta(\hat{l}-\hat{g}(y)) + \theta_Y(y)\delta(\hat{l}+\hat{g}(y))\Big]\, dy d\hat{l} \\ =& 1 - \int_0^\infty q_Y(y)\Big[(1-\theta_Y(y))\log_2(1+e^{-\hat{g}(y)}) \\ & + \theta_Y(y)\log_2(1+e^{\hat{g}(y)})\Big]dy, \end{aligned} \tag{A.4}$$

where we have exchanged the order of integrals and used the delta function sifting property. Denoting $c_Y(y) = 1-(1-\theta_Y(y))\log_2(1+e^{-\hat{g}(y)}) - \theta_Y(y)\log_2(1+e^{\hat{g}(y)})$, we write (A.4) as

$$\hat{C} = \int_0^\infty q_Y(y)c_Y(y)\mathrm{d}y.$$

For each value of y, as we stated, $c_Y(y)$ is maximized when $\hat{g}(y) = \log\frac{1-\theta_Y(y)}{\theta_Y(y)}$ which is equivalent to true LLR calculation. Since $q_Y(y)$ is positive, it is clear that the maximum of $\hat{C}$ is achieved by true LLRs.

Appendix B

Proof of Theorem 4.2

Consider an arbitrary discrete binary-input memoryless channel whose output alphabet is non-binary. The channel input is $x \in \{0,1\}$, and its output is $y \in \{y_j | 1 \leq j \leq M\}$. Let us define $P(y_j|x=0) = p_j$ and $P(y_j|x=1) = q_j$ where $\sum_{j=1}^{M} p_j = \sum_{j=1}^{M} q_j = 1$. The true LLR value, when $y = y_j$ is observed at the channel output and the binary inputs are equiprobable, is

$$l_j = g(y_j) = \log \frac{p_j}{q_j}. \tag{B.1}$$

Thus, the true LLR pdf when $x = 0$ is sent is given by $f_L^0(l) = \sum_{j=1}^{M} p_j \delta\left(l - \log \frac{p_j}{q_j}\right)$ and by $f_L^1(l) = \sum_{j=1}^{M} q_j \delta\left(l - \log \frac{p_j}{q_j}\right)$ when $x = 1$ is sent over the channel, where $\delta(\cdot)$ denotes the Dirac delta function.

Now, assuming that approximate LLR is calculated by $\hat{l}_j = \hat{g}(y_j) = a_j$ when y_j is observed at the channel output, the conditional pdfs of $\hat{l}$ are:

$$f_{\hat{L}}^0(\hat{l}) = \sum_{j=1}^{M} p_j \delta\left(\hat{l} - a_j\right), \tag{B.2}$$

$$f_{\hat{L}}^1(\hat{l}) = \sum_{j=1}^{M} q_j \delta\left(\hat{l} - a_j\right). \tag{B.3}$$

Inserting (B.2) and (B.3) in (4.9) gives

$$\hat{C}^{(i)} = 1 - \frac{1}{2} \sum_{j=1}^{M} \left(p_j \log_2(1 + e^{-a_j}) + q_j \log_2(1 + e^{a_j})\right). \tag{B.4}$$

Taking $\frac{\partial \hat{C}^{(i)}}{\partial a_j}$ reveals that $a_j = \log \frac{p_j}{q_j}$ maximizes $\hat{C}^{(i)}$ for all $1 \leq j \leq M$ since $\frac{\partial^2 \hat{C}^{(i)}}{\partial a_j^2} < 0$ for all $1 \leq j \leq M$ and $\frac{\partial^2 \hat{C}^{(i)}}{\partial a_j \partial a_k} = 0$ for all $1 \leq j \leq M$ and $1 \leq k \leq M$ and $j \neq k$. These values of a_j's are equal to the true LLR of (B.1). Thus, the maximizing point is only given by true LLRs. Noticing that these results are valid

for each equivalent bit-channel i of the BICM and since $\max \hat{C} = \sum_{i=1}^{m} \max_{\mathcal{A}_i} \hat{C}^{(i)}$ in (4.10), the theorem is proved.

Appendix C

Proof of Theorem 4.3

Denote $\hat{L}_b^{(i)} = E_{\boldsymbol{x}\in\mathcal{X}_b^{(i)}}[\hat{L}^{(i)}|\boldsymbol{x}]$ and $\boldsymbol{Y}_b^{(i)} = E_{\boldsymbol{x}\in\mathcal{X}_b^{(i)}}[\boldsymbol{Y}|\boldsymbol{x}]$ for $b \in \{0,1\}$. Then $\hat{C}^{(i)}$ can be written as

$$\begin{aligned}\hat{C}^{(i)} &= 1 - \frac{1}{2}E_{\hat{L}_0^{(i)}}\left[\log_2(1+e^{-\hat{L}_0^{(i)}})\right] - \frac{1}{2}E_{\hat{L}_1^{(i)}}\left[\log_2(1+e^{\hat{L}_1^{(i)}})\right] \\ &= 1 - \frac{1}{2}E_{\boldsymbol{Y}_0^{(i)}}\left[\log_2\left(1+e^{-\hat{g}_{\mathcal{A}_i}^{(i)}(\boldsymbol{Y}_0^{(i)})}\right)\right] - \frac{1}{2}E_{\boldsymbol{Y}_1^{(i)}}\left[\log_2\left(1+e^{\hat{g}_{\mathcal{A}_i}^{(i)}(\boldsymbol{Y}_1^{(i)})}\right)\right].\end{aligned}$$

By using (4.12) and with some abuse of notation we write

$$\hat{C}^{(i)} = 1 - \frac{1}{2}\sum_{b=0}^{1}\sum_{k=1}^{N^{(i)}} E_{(\boldsymbol{Y}_b\in\mathbb{C}_k^{(i)})}\left[\log_2\left(1+e^{(-1)^{b+1}(\langle\boldsymbol{\alpha}_k^{(i)},\boldsymbol{Y}_b\rangle+\beta_k^{(i)})}\right)\right].$$

It is clear that $\hat{g}$ is an affine function of $\boldsymbol{\alpha}_k^{(i)}$ and $\beta_k^{(i)}$ for each realization of $\boldsymbol{Y}_b$ inside $\mathbb{C}_k^{(i)}$. Noticing that the function $\log_2(1+\exp(\cdot))$ is convex and twice differentiable, it can be deduced that $\log_2(1+\exp((-1)^{b+1}(\langle\boldsymbol{\alpha}_k^{(i)},\boldsymbol{Y}_b\rangle+\beta_k^{(i)})))$ is also a convex function of $\boldsymbol{\alpha}_k^{(i)}$ and $\beta_k^{(i)}$. The convexity is also preserved under expectation. Thus, $E_{(\boldsymbol{Y}_b\in\mathbb{C}_k^{(i)})}[\log_2(1+\exp((-1)^{b+1}(\langle\boldsymbol{\alpha}_k^{(i)},\boldsymbol{Y}_b\rangle+\beta_k^{(i)})))]$ is also convex which makes $\hat{C}^{(i)}$ concave with respect to $\boldsymbol{\alpha}_k^{(i)}$ and $\beta_k^{(i)}$ for all $k = 1,\ldots,N^{(i)}$.