Application of Turbo-Codes in Digital Communications

by

Atousa Haj Shir Mohammadi

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2001

Abstract

This thesis aims at providing results and insight towards the application of turbo-codes in digital communication systems, mainly in three parts.

The first part considers systems of combined turbo-code and modulation. This section follows the pragmatic approach of the first proposed such system. It is shown that by optimizing the labeling method and/or modifying the puncturing pattern, improvements of more than 0.5 dB in signal to noise ratio (SNR) are achieved at no extra cost of energy, complexity, or delay.

Conventional turbo-codes with binary signaling divide the bit energy equally among the transmitted turbo-encoder output bits. The second part of this thesis proposes a turbo-code scheme with unequal power allocation to the encoder output bits. It is shown, both theoretically and by simulation, that by optimizing the power allocated to the systematic and parity check bits, improvements of around 0.5 dB can be achieved over the conventional turbo-coding scheme.

The third part of this thesis tackles the question of "the sensitivity of the turbo-code performance towards the choice of the interleaver", which was brought up since the early studies of these codes. This is the first theoretical approach taken towards this subject. The variance of the bound is evaluated. It is proven that the ratio of the standard deviation over the mean of the bound is asymptotically constant (for large interleaver length, N), decreases with N, and increases with SNR. The distribution of the bound is also computationally developed. It is shown that as SNR increases, a very low percentage of the interleavers deviate quite significantly from the average bound but the majority of the random interleavers result in performances very close to the average.

The contributions of input words of different weights in the variance of performance bound are also evaluated. Results show that these contributions vary significantly with SNR and N. These observations are important when developing interleaver design algorithms.

Acknowledgments

I wish to thank my supervisors, Professor Amir K. Khandani and Professor Weihua Zhuang. Professor Khandani directed my attention and interest towards the subject of this thesis at the early stages of this research. His time in reviewing my thesis and his suggestions for improvement are appreciated. I am indebted to Professor Zhuang for making the completion of this work possible with her continuous attention, care and valuable discussions during these years. Her promptness, precision, and suggestions are greatly appreciated.

I also wish to thank my thesis committee members, Professor I. F. Blake, Professor A. Hasan, Professor K. Ponnambalam, and especially my external examiner, Professor A. Yongacoglu, for their thorough reading of this thesis and insightful questions and suggestions. I also thank Professor J. Mark for serving in my thesis defense session as a delegate.

Many thanks to the staff of the E&CE Department for their support. Special thanks to Wendy Boles for her positive and helpful attitude. I also wish to thank our computer system administrator, David Evans, for his always prompt and helpful response to my questions and problems.

My sincere thanks to Professor A. Vannelli and Professor S. Chaudhuri for listening to me and showing me the solution during the difficult time I encountered in the midst of my program, and thanks again to Professor Zhuang for making this solution possible.

This research was supported by funds from the Communication and Information Technology of Ontario (CITO) and the University of Waterloo.

Their financial support is greatly appreciated.

I was blessed with many true friendships during my time at Waterloo. I would like to thank Claudia, Radmila, Laleh, Alice & Mike, Fatereh & Masood, Nooshin & Kamran, Elham & Amir, Mehrnaz & Koorosh, families of Safavi-Naeini, Yousefi, and Teimoortagh. They made Waterloo a home away from home when I first arrived in Canada, and have continued to offer their genuine friendship and support throughout the years. Very special thanks to the best lunch partners ever, Mala and Janaki, for all their technical and emotional help, for all the fond memories we share, and for staying close even when we were miles away. Thanks also to my long time friend, Roya, for our long chats and for always being there for me.

My deepest gratitude to Mom and Dad. They were parents when I needed love and support, friends when I needed to talk, and great teachers when I needed to learn. I dedicate this thesis to you. Thank you Ima, for starting by being my little sister and growing to become my best friend. Your support during these years and your help during the last days of this work are very much appreciated.

At last, praise to the almighty God, whose help and guidance has always been present in my life and who gave me the ability and strength to complete this work.

To my parents,

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The history of error-correcting coding dates back to 1948 when Claude Shannon published his famous paper titled "A Mathematical Theory of Communications" [70]. Shannon showed that associated with any communication channel is a number C (measured in bits per second), called the capacity of the channel, which has the following significance. Whenever the transmission rate (in bits per second) in a communication system is less than C, it is possible to design error control codes that can provide arbitrarily high levels of reliability at the receiver output. The proof to this theorem was existential, however, Shannon did not tell us how to find these codes.

When the 50th anniversary of the birth of Information Theory was celebrated at the 1998 IEEE International Symposium of Information Theory in Boston, there was a great deal of reflection on the year 1993 as a critical year. This was the year in which Berrou, Glavieux and Thitimajshima presented their conference paper, "Near Shannon Limit Error-Correcting Coding and

Decoding: Turbo Codes" [16]. In their presentation, Berrou *et al* claimed that a combination of parallel concatenated convolutional codes and iterative decoding can provide reliable communication at a signal to noise ratio (SNR) that is within a few tenths of a dB from the Shannon limit.

Nearly fifty years of striving to achieve the promise of Shannon's noisy channel coding theorem had come to an end. The astonishing performance of these codes has resulted in a great deal of excitement, among the coding community. The following few lines, selected from a 1995 article by Divsalar and Pollara, is an example of such excitement [27]:

> Perhaps the most exciting and potentially important development in coding theory in recent years has been the dramatic announcement of "Turbo-codes" by Berrou *et al* in 1993. The announced performance of these codes was so good that the initial reaction of the coding establishment was deep skepticism, but recently researchers around the world have been able to reproduce those results.

This thesis addresses some theoretical and practical issues concerning turbo-codes for application in digital communication systems. To show the revolutionary impact of turbo-codes in coding for reliable communications, we continue this introduction with a brief history of the research in coding theory, formulation of Shannon's capacity and performance limit, and the coding gain achieved by turbo-codes, in Section 1.1. Following that, Section 1.2 outlines the research reported in the rest of this thesis and the motivation for each research subject.

1.1 Shannon's Theorem and Turbo-Codes

Soon after Shannon's fundamental theorem, much effort was devoted to finding explicit constructions for codes that would produce the arbitrarily small probability of error promised by the theorem, but progress was meager. In the 1960s, coding research began to settle down along two branches: the algebraic and the probabilistic directions [17].

The algebraic approach has been more successful in the area of block codes, where polynomial time hard decision algorithms for many block codes have emerged. The second approach attempted to understand the encoding and decoding from a probabilistic point of view and these attempts led to the notion of sequential decoding. This, in turn, led to the introduction of a class of non-block codes of indefinite length, which can be represented by a tree and can be decoded by algorithms searching the tree. The most useful of these codes are highly structured codes called *convolutional codes*, invented by Elias in 1954 [34]. It was not until 1967 that a much simpler algorithm, the Viterbi algorithm [36], was developed and replaced sequential decoding for convolutional codes.

The two avenues of coding research were brought together by combining a convolutional code with an algebraic block code and introducing the *concatenated codes* [35]. Until turbo-codes, this class of error-correcting codes was known to have the closest performance to Shannon's limit.

The effectiveness of an error correcting code is usually expressed in terms of its *coding gain*. Coding gain is the difference between the SNR per information bit (E_b/N_0) required to achieve a given bit error rate (BER) in a coded system and the E_b/N_0 required to achieve the same BER in an uncoded system. When power limitations are more restricted than bandwidth limitations, binary phase shift keying (BPSK) is the logical choice for modulation given its power efficiency [39]. An uncoded BPSK system requires an E_b/N_0 of approximately 9.6 dB to achieve a BER of 1×10^{-5}. The question of how much coding gain is potentially possible for a given communication system when used over an additive white Gaussian noise (AWGN) channel was definitively answered by Shannon. Shannon defined the *capacity* of an AWGN channel to be

$$C = W \log_2(1 + E_s/N_0) \tag{1.1}$$

where W is the bandwidth in Hertz, E_s is the average signal energy in each two-dimensional signaling interval of duration $T = 1/W$ seconds, and N_0 is the two-sided noise power spectral density. Thus, C is in bits/sec. In this formulation, the capacity corresponds to a two-dimensional communication channel. Let us introduce the spectral efficiency, η, defined as the average number of information bits transmitted per each signaling interval T. According to the Shannon's theorem, for reliable communication we should have

$$\eta < \frac{C}{W}. \tag{1.2}$$

On the other hand,

$$\frac{E_s}{N_0} = \eta \frac{E_b}{N_0}. \tag{1.3}$$

Combining Eqs. (1.1) and (1.3) and substituting C in non-equality (1.2) results in

$$\frac{E_b}{N_0} > \frac{2^\eta - 1}{\eta}. \tag{1.4}$$

According to this relationship, for a communication system of spectral efficiency 1 there exists a coding/modulation scheme for reliable transmission with an SNR of at least 0 dB. Thus, a maximum coding gain of 9.6 dB is possible for this spectral efficiency.

The NASA/ESA [1] deep space coding standard consists of a Reed-Solomon code in serial concatenation with a convolutional code [76]. When used with BPSK modulation, it provides a BER of 1×10^{-5} at an E_b/N_0 of approximately 2.2 dB, i.e., a coding gain of 7.4 dB, but still 2.2 dB from the theoretical limit.

Many efforts were made to close this gap during the 1980s and early 1990s, mostly resulting in serial concatenated systems. Additional gains of a few tenths of a dB were obtained at a great expense, both in complexity and delay. However, for almost 50 years after Shannon's paper was published, the 2 dB gap continued to separate the performance of the most advanced error control systems from the theoretical limit, until the advent of turbo-codes in 1993. For a rate 1/2 turbo-code and information block length $2^{16} = 65536$, a required E_b/N_0 of 0.7 dB was reported, that is only 0.7 dB from the capacity bound and 0.5 dB from the BPSK bound [24]. A comparison of the performance and complexity issues of the well known coding schemes with deep-space applications, including turbo-codes, can be found in [24].

The importance of coding gain can be emphasized from an economical standpoint by considering that the current value for Deep Space Network is \$80,000,000 per dB of gain [39].

[1]National Aeronautics and Space Administration/ European Space Agency

1.2 Outline

Turbo-coding consists of two key design innovations: parallel concatenated encoding and iterative decoding. The basic principles of turbo-coding and decoding are described in Chapter 2. For now, it suffices to mention that turbo-code is constructed by the parallel concatenation of two (or more) convolutional component codes that are connected through one (or more) interleaver(s). The component codes are terminated to the all zero state at the end of each block of input data. Thus, turbo-code is essentially equivalent to a block code of usually very large block length. The coded bits consist of the systematic bits (identical to the information bits) and the parity check bits corresponding to each component encoder.

Although it is theoretically possible to derive the optimal decoder for these codes, the complexity of optimal decoding, even for turbo-codes with relatively short block lengths, is large enough to make it practically impossible. Turbo-codes would have been impractical were it not for the very high performance suboptimal iterative decoding algorithm proposed for these codes. The decoder consists of a soft-input/soft-output (SISO) component decoder for each of the component encoders. These decoders take turn in operating on the received data, forming and exchanging estimates of the message block. The way these decoders operate on each other's "incompletely decoded" outputs is reminiscent of a turbo charged engine. The name "turbo-code" is earned for parallel concatenated codes due to their decoding algorithm not the codes themselves [39].

The basic principles of turbo-codes are described in Chapter 2. More emphasis is given to the encoder structure. The details of the SISO decoding algorithm, developed by Bahl, Cock, Jelinek, and Raviv (BCJR), are given in Appendix A. A survey of the relevant literature ends this chapter.

Turbo-codes were first proposed for binary modulation. The use of BPSK modulation is well justified for deep-space communication systems with code rates 1/2 or less, due to several reasons including the abundance of bandwidth in these channels. However, for applications such as communication over bandwidth limited channels, or spread spectrum systems, the combination of these codes with multi-level modulation schemes has been proposed and studied by researchers. In Chapter 3, we study the system of combined turbo-code and modulation proposed by Goff, Glavieux, and Berrou in [37], where the coded bits are simply grouped together and mapped to points from a signal constellation. For such system of combined coding and modulation, the labeling method, i.e., how each encoded bit is mapped to a labeling position of the signal constituent bits, can affect the performance of the overall scheme. Another concern is how correlated noise affects the coded bits mapped to the same signal, and its impact on the performance. The focus of this chapter is on the study of the above two effects.

Back to turbo-code with BPSK signaling, in Chapter 4, we consider the possibility of improving the code performance by allocating unequal energy levels to the systematic and the parity check bits. Conventional turbo-codes assign equal noise margins to the encoder output bits. This would be optimal if the encoded bits contributed equally to the distance of the low weight codewords. In this chapter we show that this contribution is different for the systematic and parity check bits, and that providing the encoded bits with unequal noise margins can improve the code performance.

The critical role that the interleaver plays in the high performance achieved by turbo-codes has been well known to the researchers since the introduction of turbo-codes and especially since the theoretical work of Benedetto and Montorsi in [14]. This resulted in an extensive amount of research focused on the design of good interleavers for turbo-codes, some of which are mentioned in the literature review in Chapter 2. The effect of the choice of the interleaver on the performance, however, has not been explicitly studied in these works. In Chapter 5, we find the variance of the turbo-code performance bound over all possible interleavers, as an attempt to tackle the question of the sensitivity of the turbo-code performance to the interleaver choice.

Finally, Chapter 6 concludes this thesis and suggests research directions as extensions of this work.

Chapter 2

Turbo-Codes: Basic Principles and Literature Review

This chapter gives a brief overview of the basic concepts of turbo-codes and some of the terms and notations used through out this thesis. Sections 2.1, 2.2, and 2.3 explain the basic principles of the encoder, decoder and trellis termination, respectively. A survey of the relevant literature follows in Section 2.4.

2.1 Turbo-Encoder

The turbo-encoder is composed of the parallel concatenation of two (or more) recursive systematic convolutional (RSC) codes, connected through one (or more) interleaver(s). Since RSC codes are one of the main components of a turbo-code, these codes are explained first and following that, in Section 2.1.2, the structure of the turbo-encoder is described.

2.1.1 Recursive Systematic Convolutional Codes

As their name says, RSC codes are convolutional codes with a recursive (feedback) structure. The word "systematic" reflects the fact that the input bits are directly included as part of the output codeword. To describe the structure of RSC codes, we first consider a rate $r = 1/2$, binary, linear, non-systematic, and feedback free convolutional encoder with constraint length K. Fig. 2.1(a) gives an example with two memory units or $K = 3$.

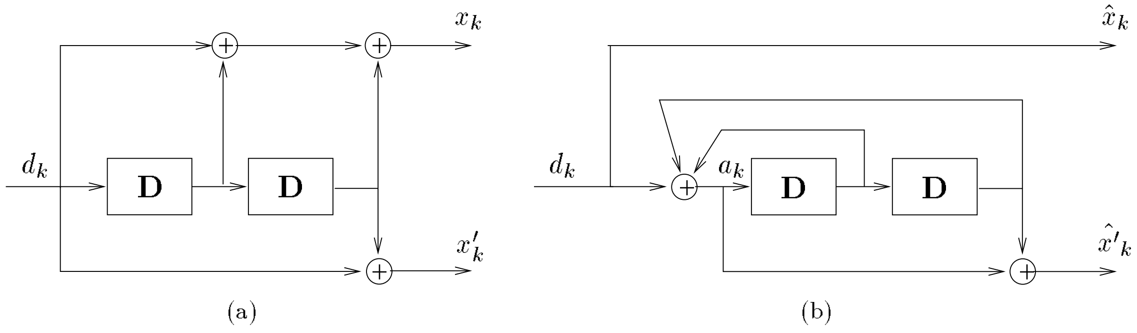


Figure 2.1: Examples of convolutional encoders of rate 1/2 and $K = 3$, (a) non systematic, feedback free, convolutional code, (b) recursive systematic convolutional code.

If we denote the input sequence by $\mathbf{d} = (d_1, d_2, \ldots)$ and the two output sequences by $\mathbf{x} = (x_1, x_2, \ldots)$ and $\mathbf{x}' = (x'_1, x'_2, \ldots)$, we have

$$x_k = \sum_{i=0}^{K-1} g_{1i}\, d_{k-i} \qquad g_{1i} = 0, 1$$

$$x'_k = \sum_{i=0}^{K-1} g_{2i}\, d_{k-i} \qquad g_{2i} = 0, 1 \tag{2.1}$$

where the summations are in modulo 2 arithmetic and g_{1i}'s and g_{2i}'s are determined by the structure of the code. Let us define the D-transform of a sequence $\mathbf{x}$ by

$$\mathbf{X}(D) = \sum_k x_k D^k, \tag{2.2}$$

then, Eqs. (2.1) can be written as

$$\begin{aligned} \mathbf{X}(D) &= \mathbf{G_1}(D)\mathbf{D}(D), \\ \mathbf{X}'(D) &= \mathbf{G_2}(D)\mathbf{D}(D), \end{aligned} \tag{2.3}$$

where the polynomials $\mathbf{G_1}(D) = \sum_{k=0}^{k=K-1} g_{1k}D^k$ and $\mathbf{G_2}(D) = \sum_{k=0}^{k=K-1} g_{2k}D^k$ are called the generator polynomials of the code.

A rate 1/2 RSC code is obtained from a non-systematic feedback free code by introducing the auxiliary variable $\mathbf{A}(D) = \mathbf{D}(D)/\mathbf{G_1}(D) = \sum_k a_k D^k$, and by defining the new output sequences by

$$\begin{aligned} \hat{\mathbf{X}}(D) &= \mathbf{D}(D), \\ \hat{\mathbf{X}}'(D) &= \mathbf{G_2}(D)\mathbf{A}(D) = \frac{\mathbf{G_2}(D)}{\mathbf{G_1}(D)}\mathbf{D}(D). \end{aligned} \tag{2.4}$$

$\mathbf{G_1}(D)$ and $\mathbf{G_2}(D)$ are referred to as the "backward" and "forward" generator polynomials, respectively. This gives rise to the encoder structure shown in Fig. 2.1(b). The output sequences $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, \ldots)$ and $\hat{\mathbf{x}'} = (\hat{x'}_1, \hat{x'}_2, \ldots)$ are called the *systematic* and the *parity check* sequences, respectively.

It can be shown that the two encoders in Figs. 2.1 (a) and (b) have the same trellis structure, in terms of the number of states, branches and the output labels. This can be verified as follows. First, both codes have the same constraint length, and thus, the same number of states. Now, assume that both codes are in the same state at time k and consider their transition to the same state at time $k+1$. To distinguish between the input bits of the non-recursive and recursive codes for this transition, let us denote them with d_k^{NR} and d_k^R, respectively. It can be easily seen that in order for both codes to transfer to the same state, we have,

$$a_k = d_k^{NR}. \tag{2.5}$$

As a consequence,

$$x'_k = \hat{x}'_k. \tag{2.6}$$

On the other hand, considering that the add operations are in modulo 2, a_k can be written as

$$a_k = x_k + d_k^{NR} + d_k^R. \tag{2.7}$$

Combining Eqs. (2.5) and (2.7) and considering that $\hat{x}_k = d_k^R$,

$$x_k = \hat{x}_k. \tag{2.8}$$

As a consequence, the two codes have the same free distance and probability of an error event. The only difference between these two codes is in their input-output relationships, which causes their bit error probabilities to be different. In particular, in the feedback-free realization of the encoder, two branches merging to the same state always have the same input labels, while in the recursive systematic realization of the encoder, two branches merging in the same state always have different input labels. Fig. 2.2 shows the trellis structure for the codes shown in Fig. 2.1. The three bits in the labels on each branch show the input and the two output bits corresponding to that transition, from left to right, respectively.

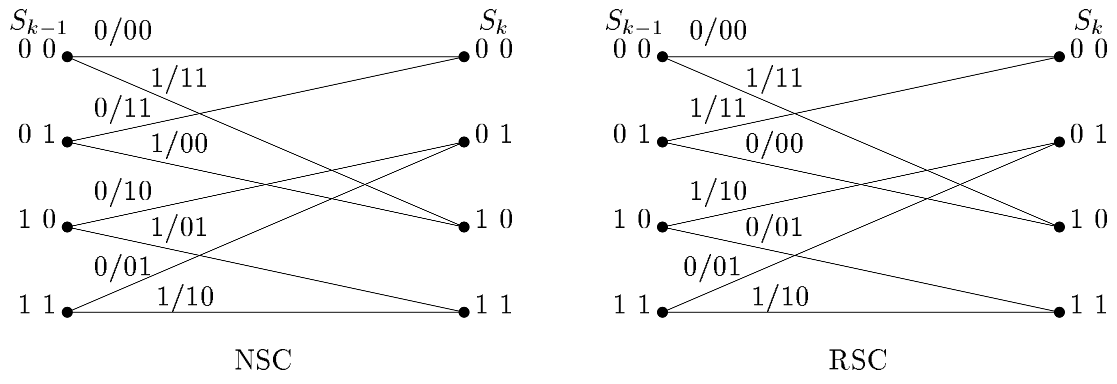


Figure 2.2: Trellis structure corresponding to the codes shown in Fig. 2.1, (a) non systematic, feedback free, convolutional code, (b) recursive systematic convolutional code.

An RSC code is determined by its generator polynomials and is usually denoted by (g_f, g_b), where g_f and g_b are the representations of the forward and backward generator polynomials in octal form, respectively. Thus the code shown in Fig. 2.1(b) is a (5,7) RSC code.

An important property of recursive convolutional codes is their infinite impulse response (IIR), i.e., a single "1" in the input data results in a non-terminating output sequence. As we will see later, this property is the main reason behind using RSC codes as the components of turbo-codes.

2.1.2 Turbo-Code: Parallel Concatenation of RSC Codes

The turbo-encoder consists of the parallel concatenation of two RSC codes, referred to as component codes, that are connected through an interleaver of length N. The first component encoder operates directly on the binary information bit sequence, $\mathbf{d} = (d_1, \ldots, d_N)$, producing the systematic and parity check sequences $\mathbf{x^s} = (x_1^s, \ldots, x_N^s)$ and $\mathbf{x^{1p}} = (x_1^{1p}, \ldots, x_N^{1p})$, respectively. The second component encoder operates on a re-ordered (interleaved) sequence of the information bits, $\mathbf{d^I} = (d_1^I, \ldots, d_N^I)$. The second encoder output often only consists of the parity information sequence, $\mathbf{x^{2p}} = (x_1^{2p}, \ldots, x_N^{2p})$, although in some applications, where low rate data transmission is not a concern, the second systematic information sequence is also transmitted [29]. Fig. 2.3 shows a schematic example of a rate $r = 1/3$ turbo-code, employing two identical component codes with generator polynomials (5,7).

The importance of employing a recursive convolutional code for constructing turbo-codes and the reason behind their high performance can now be explained based on the presence of the interleaver. If the component codes are not recursive, a single "1" in the input data stream will have a low weight codeword in the first code and result in a finite error event. This single bit will be moved to a new location in the block of the input data after passing through the interleaver, but will still result in a low weight second codeword. Thus, the overall codeword will have a low weight. On the other hand, as mentioned before, RSC codes are IIR codes, thus only input data of weight two or higher can result in finite length error events in these codes. If a low

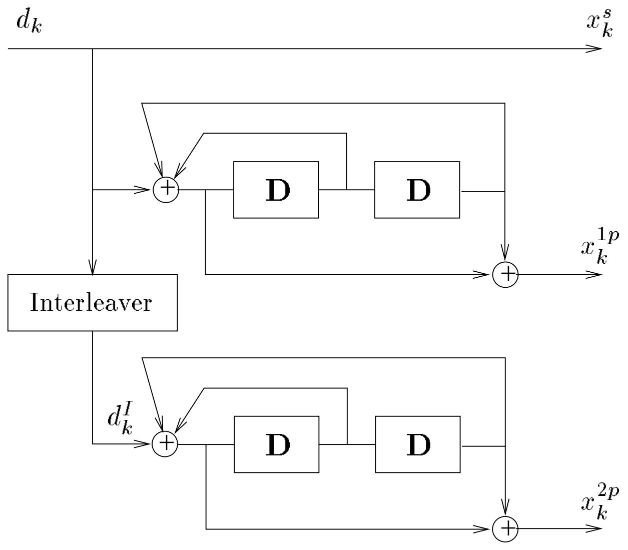


Figure 2.3: Turbo-encoder of rate 1/3 and identical RSC codes with generator polynomials (5,7).

weight input sequence has a pattern that results in a low weight codeword in the first code, it is very likely that the interleaver breaks this pattern such that the second codeword is of high weight. This phenomenon gets stronger as the interleaver length increases. The good distance properties and high performance of turbo-codes are explained in more theoretical detail in Chapter 4.

In general, the number of component codes can be more than two and they do not need to be identical. Also, to achieve higher global rates, it is possible to puncture the encoded bits. For instance, in order to increase the rate of the code shown in Fig. 2.3 from 1/3 to 1/2, the systematic and only one of the parity check bits are transmitted for each information bit, d_k. In other words, the transmitted sequence will be $(x_1^s, x_1^{1p}, x_2^s, x_2^{2p}, \ldots)$.

The receiver inserts the punctured bits, with the value zero into the received sequence.

2.2 Turbo-Decoder

The output bits of the turbo-encoder are modulated and transmitted through the channel. Let $(y_1^s, y_1^{1p}, y_1^{2p}, y_2^s, y_2^{1p}, y_2^{2p}, \ldots, y_N^s, y_N^{1p}, y_N^{2p})$ denote the received sequence. Assuming an AWGN channel and BPSK modulation, we have

$$\begin{aligned} y_k^s &= 2x_k^s - 1 + n_k^1 \\ y_k^{1p} &= 2x_k^{1p} - 1 + n_k^2 \\ y_k^{2p} &= 2x_k^{2p} - 1 + n_k^3 \end{aligned} \tag{2.9}$$

where $\{n_k^1\}$, $\{n_k^2\}$, and $\{n_k^3\}$ are independent and identically distributed (i.i.d.) Gaussian random variables.

If maximum likelihood (ML) decoding were to be used for decoding the turbo-code, it would be necessary to obtain the so called *hyper-trellis* of the overall code [8]. The hyper-trellis consists of $2^{K_1-1} \times 2^{K_2-1}$ states, where K_1 and K_2 are the constraint lengths of the first and second component codes, respectively. Each step of the hyper-trellis corresponds to N steps of the individual trellises. The branch labels depend on the specific interleaver that is used. The complexity of this trellis grows very large even for relatively short interleaver lengths and makes the decoding procedure practically impossible (e.g., by employing the Viterbi algorithm). Thus, a sub-optimal decoding algorithm is required to break the decoding procedure into simpler steps.

Indeed, the availability of such sub-optimum decoding methods is one of the key points behind the significance of turbo-codes.

The turbo-decoder has an iterative structure and consists of two component decoders, which operate in a serial mode, as shown in Fig. 2.4. Each

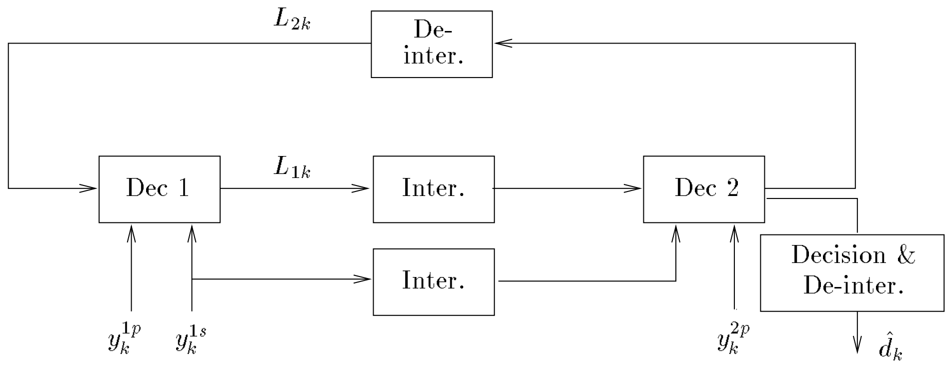


Figure 2.4: The structure of the turbo-decoder.

component decoder (Dec1 and Dec2) corresponds to one of the component codes and thus the complexity of the trellis of each decoder is the same as that of a single component code. Both decoders receive the systematic bits, and each decoder receives its corresponding parity check bits. The decoders employ a soft output decoding algorithm to obtain the logarithm of likelihood ratio (LLR) for each data bit, d_k, as follows,

$$\Lambda_i(d_k) = \log \frac{Pr(d_k = 1|\text{observation})}{Pr(d_k = 0|\text{observation})} \qquad k = 1, 2, \ldots, N \ , \tag{2.10}$$

where $i = 1, 2$ refers to Dec1 and Dec2, respectively, and $Pr(d_k = j|\text{observation})$, $j = 0, 1$, is the a-posteriori probability (APP) of d_k.

In order to explain the iterative procedure, let us start from Dec1. From each LLR calculated in this component decoder, the so called *extrinsic in-*

formation, denoted by L_{1k} in Fig. 2.4, is obtained. The extrinsic information is then interleaved through an interleaver with the same structure as the one used in the encoder, and is provided to Dec2. This information is regarded as the a-priori probability by the second decoder and is used in conjunction with the information it receives from the channel to in turn provide the first decoder with the extrinsic information, L_{2k}. The iterations continue and at each iteration the LLRs are updated to more reliable values. At the end of the last iteration a hard decision is made based on the sign of each LLR, as follows,

$$\hat{d}_k = \begin{cases} 0 & \text{if} \quad \Lambda_2(d_k) \leq 0 \\ 1 & \text{if} \quad \Lambda_2(d_k) > 0 \end{cases} \tag{2.11}$$

where $\hat{d}_k$ is the decoded value corresponding to d_k. For the first iteration, Dec1 assumes the value zero for $L_2(d_k)$, for all k.

One of the well known soft output decoding algorithms for convolutional codes, is the BCJR algorithm, otherwise known as the Bahl *et al* algorithm [1]. This algorithm minimizes the bit error probability and yields the APP for each decoded bit. For RSC-codes, the algorithm must be modified in order to take into account their recursive characteristic. The modified BCJR algorithm and the mathematical details of the decoding procedure can be found in Appendix A.

2.3 Trellis Termination

The decoding algorithm of turbo-code is such that each component decoder requires knowledge of the first and the last state of the trellis (refer to Appendix A). Therefore, both component codes are required to start and terminate at the all zero state. As a consequence, turbo-codes are equivalent to block codes.

Since the component encoders of a turbo-code are recursive, it does not suffice to set the last $K-1$ information bits to zero in order to drive the encoders to the all zero state. The necessary termination sequence depends on the state of the encoder after the Nth input bit. Note that, due to interleaving, the encoders are not in the same state at the end of the input sequence, and consequently, different termination sequences are required for terminating the encoders. The problem of trellis termination is not addressed in [16]. In [66], only the first encoder is terminated. This results in a slight modification in the way the decoder associated with the second encoder is initialized for the modified BCJR algorithm.

A more precise solution for the problem of trellis termination is given in [27]. This is shown in Fig. 2.5, where the switch is in position "A" for the first N clock cycles and in position "B" for the $K-1$ additional cycles. This method is used for both of the encoders and it is easy to verify that it will terminate the encoders simultaneously.

Other methods of trellis termination have also been developed, which are mentioned in Section 2.4.

2.4 Literature Review

During the relatively short time since the introduction of turbo-codes, a considerably large amount of research has been performed on almost every re-

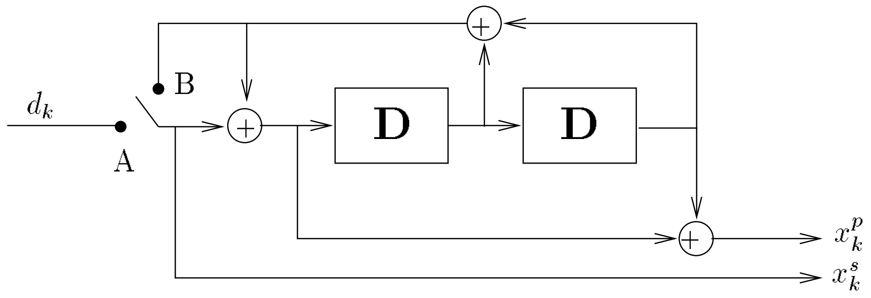


Figure 2.5: Trellis termination [27].

lated aspect of these codes. The enthusiasm raised among coding researchers towards investigating and better understanding of turbo-codes was so high that many closely related and similar research and publications, carried out independently, can often be found in this area.

Consequently, a thorough literature survey covering the voluminous research on different aspects related to these codes is beyond the scope of this work. Therefore, this section is intended to briefly review the main and substantial published research works in this area, as well as publications which contain rather large related bibliographies. Obviously, more emphasis is given to those works related to the research presented in this thesis, and the corresponding publications are explained in more detail.

Before we proceed with categorized literature review, we should cite the recently published books on turbo-codes, [39] and [75]. The earlier book covers the basic principles of turbo-codes and their operation but does not go into details on theories behind each principle. The more resent book however, covers different aspects of design and application of turbo-codes in more detail.

2.4.1 Basic Principles of Turbo-Codes

The first introduction of turbo-codes was in the 1993 short conference article of Berrou, Glavieux and Thitimajshima [16]. In this paper, the general structure of the encoder and the iterative decoding scheme are introduced and the decoding algorithm is briefly explained. Simulation results performed for a rate 1/2 turbo-code with component codes of constraint length 5, and block interleaver size 256×256 show an astonishingly high performance which is very close to the Shannon limit [16]. However, in this work, not much theoretical explanation is given on the performance and behavior of the code.

Later, Robertson, explained the behavior of these codes to some extent, by analyzing the interleaver and calculating approximations to the BER performance of the code for high signal to noise ratios [66]. He also showed how the iterative decoder can be formulated in a simpler fashion. The decoding algorithm presented in [66] is used throughout this research for simulation purposes and is explained in Appendix A.

Serious attempts towards the theoretical understanding and evaluation of the turbo-code performance appeared first in [13] and later in more detail in [14]. There, the concept of *uniform interleaver* and employing it for evaluating the union bound on the performance of turbo-codes is introduced. Their study evaluates the performance bound of a turbo-code with fixed component codes, averaged over all possible interleavers of the same length. The premise is that there exists at least one interleaver that performs as good as the average. The mathematical derivation of this bound is briefly explained in Chapter 4.

Similar work can be found in [28], where the possibility of applying the Gallager bound to improve the standard union bound is also mentioned. Based on this idea, in [32] the authors have derived a new upper bound for turbo-codes. This bound is tighter than the previous bound for a larger range of SNRs extending below the channel cutoff rate, which is the region where the standard union bound diverges from the actual performance.

In [63], the authors provide a semi-tutorial on the reasons behind the good performance of turbo-codes based on their distance spectrum. This work includes a rather large reference list of publications related to turbo-codes.

More recently, a theoretical study of turbo-codes based on their group properties, when considered as periodic linear systems, was published in [48]. The role of the interleaver in breaking the low weight sequences and the limitations that exist in this regard are presented in this work.

Other related works considering the distance properties of turbo-codes for improving their performance can be found in [40, 56, 31]. These works are based on the effect of unequal power allocation (UPA) to turbo-encoder output bits, on the performance of these codes. The discussions of Chapter 4 are based on this idea. It should be noted that the approach and the results presented in [31] are very close to our approach and results in [55], and the two works were carried out independently.

2.4.2 Turbo Decoding

Without the availability of the iterative algorithm for decoding of turbo-codes, these codes would perhaps have no practical value. The iterative decoding algorithms proposed in [16] and later in [66] employ the modified BCJR algorithm for decoding of each component code, as explained in Section 2.2 and Appendix A. Although these algorithms achieve the minimum bit error probability for the component codes, their realization is rather expensive due to their relatively high complexity and delay. To overcome this difficulty, sub-optimum component decoders have been introduced. In [42], a sub-optimum decoding scheme is proposed, which is based on a simplified version of the decoding method of [16]. This algorithm avoids exponential and logarithmic calculations of [16] and further replaces many of the multiplications by summations.

Another version of a simplified maximum a-posteriori (MAP) algorithm has been proposed in [12]. The algorithm works in a sliding window form, and can thus be used to decode the transmitted sequences continuously, without requiring trellis termination.

The a-priori soft output Viterbi algorithm (APRI-SOVA) is another soft-output decoding algorithm, proposed in [38], which has been employed by many researchers in their simulations because of the lower complexity of the algorithm.

In [30], turbo-codes using multiple (more than 2) component codes with a decoder that works in a parallel mode instead of the serial mode (which is the case for two component turbo-codes) are proposed.

Recently, a new turbo-decoding algorithm was proposed based on list decoding [61]. This algorithm almost has the same complexity as conventional turbo-decoding algorithms, but is shown to provide considerable reduction in the BER especially in the error floor region of these codes.

It should be noted that although it is intuitively and commonly expected that the iterative turbo-decoding algorithms converge to or very close to the optimal ML decision, in [53] it is shown that for blocks of length $N \geq 3$, the possibility exists that the turbo decoding algorithm is not optimal and the iterative procedure does not even converge. However, this may only happen for certain blocks of input data and such problem has not been reported to happen in practice.

2.4.3 Design of Turbo-Code

Among the research work carried out on the design of the turbo-encoder, only few investigate the design of the component codes, and the majority of the research is concentrated on the design of the interleaver.

A rather detailed work on the role of the component codes on the distance properties of turbo-codes, as well as some guidelines for optimal design of the component codes, has been published in [15]. In [26], the concept of the *effective free distance* of turbo-codes is defined and studied. This concept is then used to choose the corresponding component codes. The most recent results on this subject can be found in [11], where tables of the best RSC codes of various rates to be used in the construction of the turbo-encoder are presented.

The design of the interleaver has been studied from different aspects. When turbo-codes were first introduced, block interleavers were employed in their structure. In fact, as already mentioned, the simulation results corresponding to the astonishingly high performance of these codes, presented in [16], were obtained by employing a block interleaver.

Later, employing random interleavers, i.e., interleavers designed by generating random integers without replacement, was considered. Perhaps the most widely used class of random interleavers currently employed in turbo-codes is the so called *S-random* interleaver, introduced in [27]. This type of interleaver is designed as follows: each randomly selected integer is compared to S previously selected integers. If the current selection is equal to any of the S previous selections within a distance of $\pm S$, it will be rejected. This process is repeated until all N integers are selected, where N is the interleaver length. It has been observed that choosing $S < \sqrt{N/2}$ usually produces a solution in a reasonable time. The interleavers used for simulation purposes throughout this research are developed based on this algorithm.

Other interleaver design techniques have also been proposed, based on the distance properties of turbo-codes. In [46], an interleaver design technique is proposed which searches for a random interleaver resulting in the fewest output sequences with low weights corresponding to input weights of 2 or 3. The authors then use simulation results to show that for short frame transmission systems and BER of around 10^{-3}, a block interleaver outperforms the best such found pseudo-random interleaver, and the overall effect of the interleaver is not significant in this range [47]. In [63], however, it is shown that, for turbo-codes of large interleaver lengths, pseudo-random interleavers outperform block interleavers significantly, e.g., 2.7 dB at BER of 10^{-5}.

In [50], a deterministic interleaver design algorithm is proposed based on linear recursion to produce an initial interleaver which is subsequently optimized by pair-wise exchange of its elements. These optimized interleavers show more than 0.5 dB improvement over a randomly selected interleaver and about 0.2 dB improvement over an S-random interleaver for BERs of less than 10^{-5} and block length 576.

Most recently, a systematic approach for the design of the interleaver is proposed in [25]. The method is based on recursively minimizing a cost function to find an interleaver which best breaks a set of a-priori chosen error patterns. The weight distribution of a turbo-code employing the best such found interleaver of length 100 shows 0.5-0.9 dB improvement over a randomly selected interleaver of the same length.

Some other references which have considered issues relevant to the design of the interleaver are [6, 5, 30, 41, 49].

As can be seen, most of the above research are focused on search algorithms for good interleavers. Only very few conclusions, regarding the effect of different choices of interleavers on the performance, are implicitly stated in these works or can be drawn from their results. Some of these conclusions are even somewhat in contradiction with each other! For example, in [63] it is stated that *most* pseudo-random interleavers of large lengths result in the same multiplicity of the free distance codewords, and thus very close performances for high SNR. However, in [77], the authors state that at high SNR it is possible to design interleavers that will ensure that the turbo-code free distance is increased *significantly*. In [58] and [59], we find the variance of the performance bound over all possible interleavers, as a step towards giving more insight on the sensitivity of turbo-code performance to the choice of the interleaver. The discussions of Chapter 5 are based on this study.

2.4.4 Applications of Turbo-Codes

For the application of turbo-codes in practical communication systems, several proposals have been made on bandwidth efficient turbo-code systems, based on the idea of trellis-coded modulation [72]. The first such system and perhaps the least complicated approach is presented in [37], where the idea is to map the encoded bits of a standard turbo-code to a certain constellation. The discussions of Chapter 3 are based on this system.

Another approach is developed in [68], where two trellis codes are concatenated in parallel. In this case the interleaver operates on groups of bits. The decoding algorithm is a generalized version of the standard turbo-decoding algorithm of [66]. The performance of this system is comparable to the performance of the one presented in [37]. An earlier version of this work is published in [67].

In [10] and [9], the authors have taken quite a different approach and, although they suggest the performance of this scheme to be superior to those of the previous two schemes, it is shown that the difference is only about 0.1 dB [2].

Application of turbo-coded modulation systems is not limited to communication over bandwidth limited channels. For transmission over non-coherent channels, turbo-codes can be combined with orthogonal modulation. In [23] a system of combined turbo-code and orthogonal signaling is proposed, which consists of a rate $1/J$ turbo-code combined with a 2^J-ary orthogonal signaling. The proposed system is simulated over coherent and non-coherent AWGN channels with or without fading. Based on these results the author expects that orthogonally mapped turbo-codes will be significant enough to be considered as a replacement to the current coding schemes employed in the IS-95 code division multiple access (CDMA) system. In [57], we study the combination of a rate 1/3 turbo-code and M-ary orthogonal modulation, and the effect of the noise correlation on the performance of such a system.

So far, the only theoretical approach in the study of combined turbo-coded modulation systems is presented in [33], where the average union bound on the ML performance of such systems is evaluated, by assuming uniform interleaving just as its counterpart for turbo-coding and binary modulation [14]. This bound is based on introducing two extra interleavers at the output of the turbo-encoder, again in the form of uniform interleaving, to eliminate the correlation between the coded bits mapped to the same constellation point. Without these interleavers, the performance of the system depends not only on the weight distribution of the turbo-code, but also on the relative positions of the output bits. However, as shown in Chapter 3, it might be possible to eliminate the extra interleavers for certain modulation schemes, by optimizing the labeling method in signal mapping. Without the extra interleavers the approach of [33] will not be applicable.

Application of turbo-codes in CDMA mobile radio systems has also been considered by several researchers. Examples of early works on this subject can be found in [45] and [60]. In these examples, joint detection (JD) [51] in combination with coherent receiver antenna diversity (CRAD) [43] is assumed. The reported simulation results show gains of the order of 1 dB in the SNR, over conventional systems which employ convolutional codes.

In the IMT-2000 proposals for the third generation mobile radio systems, turbo-codes are proposed for data transmission rates higher than 14.4 kbps on both forward and reverse supplement channels (F-SCH, R-SCH) [74]. Other related works are presented in [21, 62].

Application of turbo-codes for unequal error protection (UEP), when the input data fall in different importance classes, have also been proposed [4, 20, 22]. The schemes proposed in [4, 20] achieve UEP by using ad hoc nonuniform puncturing and interleaving in the encoder. In [22], the authors elaborate the idea and propose a general method for adding UEP to turbo-codes. These schemes can be applied when combined source channel coding is performed.

Chapter 3

Combined Turbo-Code and Modulation for Personal Communications

Although turbo-codes were originally proposed for binary modulation, their combination with multi-level modulation schemes has also been proposed and studied by researchers in the area. The application for these systems can be divided into two main categories: (a) communication over bandwidth limited channels (e.g., [37, 67, 9]); (b) application in spread spectrum communication systems such as CDMA systems (e.g., [45, 60]).

The idea of turbo-coded modulation is inspired by the well known method of trellis coded modulation (TCM) [72]. Different approaches to turbo-coding combined with modulation have been suggested in literature, which are reviewed in Section 2.4.

The original and least complicated approach is presented in [37]. The idea is to map the encoded bits of a standard turbo-code (possibly after puncturing some of the parity bits) to a certain constellation. Soft demodulation and turbo-decoding are performed at separate stages as is explained in Section 3.1. Coding gains of up to 2.6 dB over 64-sate TCM are achieved for this system, employing a memory 4 turbo-code with interleaver size 4096.

In [68], another approach is developed where the authors concatenate two simple trellis codes in parallel, called turbo TCM (TTCM). In this case, the interleaver operates on groups of bits. The decoding algorithm is a generalized version of the turbo-decoding algorithm used for binary turbo-codes. The performance improvement of about 0.5 dB over that of [37] is due to the decoding algorithm, otherwise, the two performances are comparable [33].

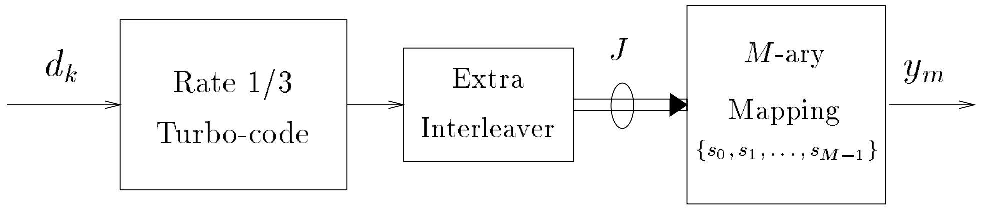
In [10], a different approach for parallel concatenated TCM (PCTCM) is adopted. In this case, the information block and the interleaver are divided into two parts. This allows each half of the information block to be punctured at one of the encoders. The parity bits, however, are generated by the complete block of information bits. At the output of the encoders, the systematic and parity check bits are mapped to an appropriate constellation. Decoding is done by using the MAP algorithm extended to symbols. The performance improvement over the previous two schemes is about 0.1 dB [2].

A closer look at the above three schemes shows that the proposed systems in [68] and [10] are special cases of [37]. That is, by restricting the interleaver structure and the puncturing scheme, one can obtain the other two from the original scheme. The difference in the performance is caused by the decoding algorithm and is not significant.

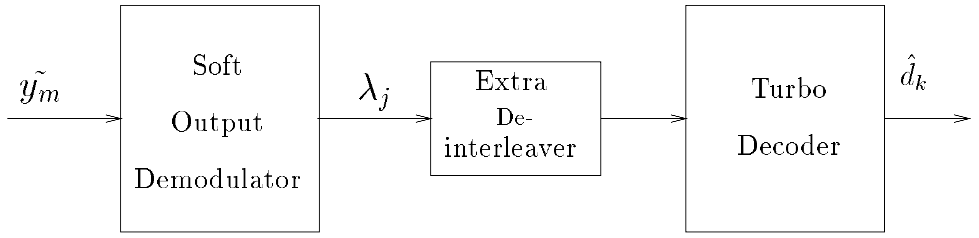
For this reason and also its simplicity and pragmatism, the system of combined turbo-code and modulation proposed in [37] is studied in this chapter. The focus is on the labeling and puncturing methods and the elimination of noise correlation among the coded bits mapped to the same signal point. These two issues are not considered in any of the above mentioned works, and as is shown in this chapter, they can affect the performance depending on the modulation employed. In Section 3.1, the general structure of a transmission system employing combined turbo-code and modulation is briefly described. Sections 3.2 and 3.3 study systems of combined turbo-code with 8-PSK and 16-QAM, respectively. Finally, Sections 3.4 concludes this chapter.

3.1 General Structure of a System of Combined Turbo-Code and M-ary Modulation

Fig. 3.1 shows the block diagram of a system of combined rate 1/3 turbo-code and modulation. After the appropriate puncturing is performed on the turbo-encoder output bits to adjust the code rate, each set of $J = \log_2 M$ coded bits is mapped to a signal point, $y_m \in \{s_0, s_1, \ldots, s_{M-1}\}$. In general, the labeling bits mapped to the same signal are affected by correlated noise. This obviously can affect the performance of the turbo-decoder, since the LLRs applied to the decoder are no longer independent, as would be the case if binary signaling were used for transmission. The extra interleaver in Fig. 3.1 is used to eliminate this correlation. However, as will be shown in



(a)



(b)

Figure 3.1: Combined turbo-code and M-ary modulation, (a) the transmitter, (b) the receiver.

the following sections, the effect of this correlation on the performance of the code can differ depending on the modulation scheme used.

The decoding is performed in two separate stages. The first stage consists of a symbol by symbol soft output demodulator. For each received signal, $\tilde{y}_m$, the demodulator calculates the LLR of each of the corresponding J labeling bits, denoted by $\lambda_j, j = 1, 2, \ldots, J$. For example, if the kth systematic bit is mapped to the jth labeling bit of the mth transmitted signal, y_m, we have

$$\lambda_k^{1s} = \lambda_j = \log \frac{Pr(x_k^{1s} = 1|\tilde{y}_m)}{Pr(x_k^{1s} = 0|\tilde{y}_m)}. \tag{3.1}$$

If $\mathcal{S}_i^{1s}$ is a partition of the signal constellation including the signals that correspond to $x_k^{1s} = i$, for $i = 0, 1$, then,

$$\lambda_k^{1s} = \log \frac{Pr(y_m \in \mathcal{S}_1^{1s}|\tilde{y}_m)}{Pr(y_m \in \mathcal{S}_0^{1s}|\tilde{y}_m)}. \tag{3.2}$$

Other LLRs are evaluated in a similar way. The output of the demodulator passes through the extra de-interleaver and is then passed to the turbo-decoder. The decoder interprets this information as channel information and the decoding algorithm is the same as in the case of BPSK signaling.

It should be noted that the decoding iterations can also include the demodulation (de-mapping), similar to the approach proposed in [19] for combined convolutional codes and modulation. Obviously, this adds to the complexity and delay of the receiver. Furthermore, it is shown in [19] that when Gray mapping is employed, including the demodulation in the iterative process, does not improve the performance. In Gray mapping, each two nearest neighbor constellation points differ only by a single bit in their labels, and this mapping is considered through out this section to follow the approach of [37].

3.2 Turbo-Code Combined with 8-PSK Modulation

In the coding system considered here, no puncturing is performed and thus, each set of the three encoder output bits is mapped to a signal point from the 8-PSK signal constellation, resulting in an spectral efficiency of 1 (bit/signal). Fig. 3.2 shows the 8-PSK signal constellation with Gray mapping. In this figure, b_1, b_2, and b_3 refer to the three bit positions in the signal labels, from right to left.

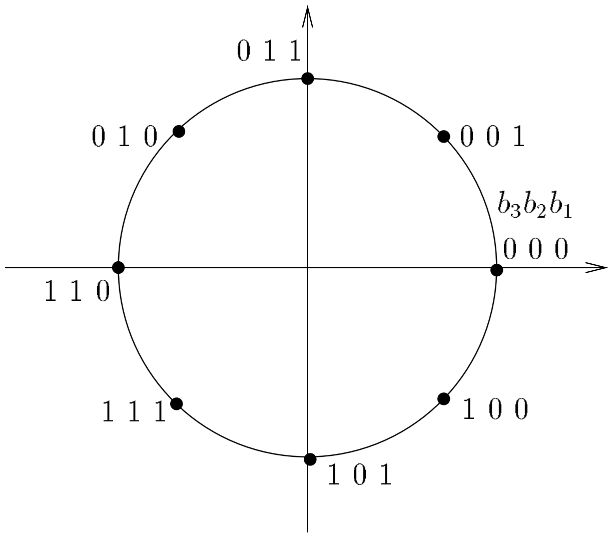


Figure 3.2: Signal mapping for an 8-PSK Gray code.

Referring to Fig. 3.2, it can be seen that position b_1 is protected less than the other two positions against the channel noise. Assuming that at the output of the demodulator a hard decision is made for each labeling bit, it can be shown that for asymptotically high SNR,

$$\begin{aligned} P_e(b_1) &= p, \\ P_e(b_3) = P_e(b_2) &= \tfrac{1}{2}p, \end{aligned} \tag{3.3}$$

where $P_e(b_i)$ is the probability that $b_i, i = 1, 2, 3,$ is detected with error, $p = \frac{1}{2}\text{erfc}(\sqrt{E_b/N_0}\sin\frac{\pi}{8})$, and E_b/N_0 is the SNR per information bit [64]. The effect of this unequal noise protection in a soft-output demodulator is that the LLRs of the lower protected bits have, on the average, smaller absolute values and thus are less reliable than those of the higher protected bits.

For this system, three different labeling methods can be achieved based on which encoder output bit is mapped to the labeling position with the least noise protection, i.e., position b_1:

- Labeling I: Position b_1 is assigned to the second parity check bit. In this case, because of the higher protection on the systematic bits, the extrinsic information, which is regarded as the a-priori probability by each component decoder, and the information about the first parity check sequence are more reliable than the information on the second parity bits. However, it is expected that in the second decoder, the high reliability of the systematic information and the soft decision made by the first decoder will compensate for the poor error protection on the second parity check bit.

- Labeling II: Position b_1 is assigned to the systematic bit. the systematic bits are protected less than the parity bits, thus resulting in less reliable systematic information than in the first mapping, for both encoders.

- Labeling III: Position b_1 is assigned to the first parity check bit. This is similar to Labeling I, only with the first and second parity bit positions switched. It is thus expected that this labeling will result in the same performance as Labeling I, if the number of decoding iterations is increased by one.

According to the results shown later in Chapter 4, lower protection over the systematic bits improves the turbo-code performance when BPSK modulation is employed. The same effect can be expected for multilevel modulations, only when extra interleaving is performed to eliminate the noise correlation among the coded bits. However, as will be discussed in the following, eliminating the noise correlation does not necessarily improve the overall performance.

For the present system, the systematic and first parity check bits corresponding to each information bit are affected by correlated noise. They are almost independent from the second parity check bit, due to the interleaving present in the structure of the turbo-code. When Gray mapping is used, the correlation between the systematic and first parity check bits is such that if one of these bits is detected with error, the probability that the other bit is also in error reduces. In fact,

$$\lim_{SNR \to \infty} P_e(b_i|b_j) = 0, \ i, j = 1, 2, 3, \ i \neq j, \tag{3.4}$$

where $P_e(b_i|b_j)$ is the conditional probability that the labeling bit b_i is detected with error given b_j is detected with error. This is because in PSK signaling with Gray mapping, each signal differs only in one labeling bit from its two nearest neighbor signals. This suggests that it might be to the advantage of the decoding performance if the noise correlation between the systematic and parity check bits is not eliminated in this system, although the turbo-decoding algorithm is suboptimal due to overlooking this correlation.

Simulations have been performed to compare the performance of different labeling methods as well as the effect of noise correlation for this system. The results are given in 3.2.1.

3.2.1 Simulation Results

Simulations are performed for a system employing a turbo-code with generator polynomials (5,7). Number of decoding iterations is equal to 4. For an AWGN channel, the kth received signal can be represented by:

$$\tilde{y}_k = y_k + n_k \tag{3.5}$$

where n_k's are the i.i.d., two dimensional, zero mean, Gaussian noise vectors with independent components and variance $N_0/2$ over each dimension.

Fig. 3.3 shows the results corresponding to interleaver lengths $N = 100$, and 380. The BER performances are shown for the system without extra interleaving and Labeling methods I and II, and for the system with extra interleaving. For the system with extra interleaving only the performance corresponding to Labeling II is shown. The reason is that, in this case, the correlation between the coded bits is ideally eliminated, and based on the results shown later in Chapter 4, providing less noise protection to the systematic bits than to the parity bits results in a better performance. Simulation results have also confirmed that labeling II results in the best performance

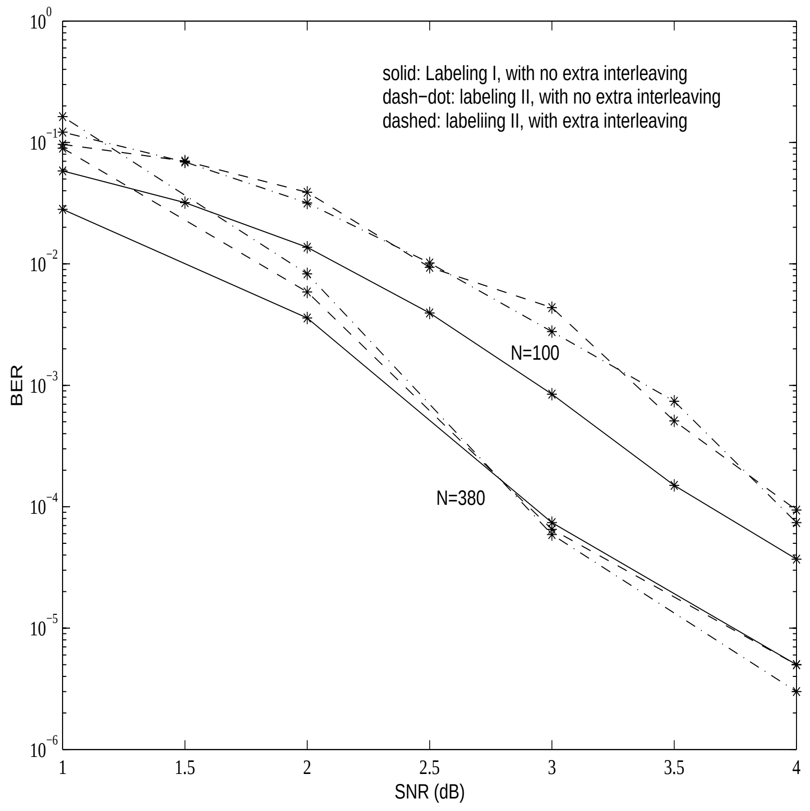


Figure 3.3: Performance of combined turbo-code and 8-PSK modulation for different labeling methods, with and without extra interleaving.

when the extra interleaver is employed. Results for Labeling III are not presented here but simulations show that this labeling method performs almost the same as Labeling I, subject to increasing the number of iterations by one.

As can be seen from the figure, Labeling I without extra interleaving results in the best performance for $N = 100$ for the range of SNRs shown here and for $N = 380$ for SNRs up to 3 dB. Above 3 dB for $N = 380$ Labeling II, again without extra interleaving, performs better. Extra interleaving does not improve the performance in either case and can be eliminated.

As the simulation results show, a saving of around 0.3-0.5 dB in energy can be obtained by optimizing the labeling method, and this is achieved at no extra cost in energy or complexity. Furthermore, elimination of the extra interleaver not only improves the BER performance, but also reduces the time delay and complexity of the system.

3.3 Turbo-Code Combined with 16-QAM Modulation

For this system, puncturing is performed to gain spectral efficiency of 2 (bits/signal). The Gray mapping can be achieved in several different ways. Here, we have adopted the mapping of [37] where each signal differs in only 2 bits with each of its diagonal neighbors, as shown in Fig. 3.4. For this mapping, it can be seen that the bit positions b_1 and b_3 are less protected than b_2 and b_4. In fact, assuming asymptotically high SNR and hard decision at the demodulator,

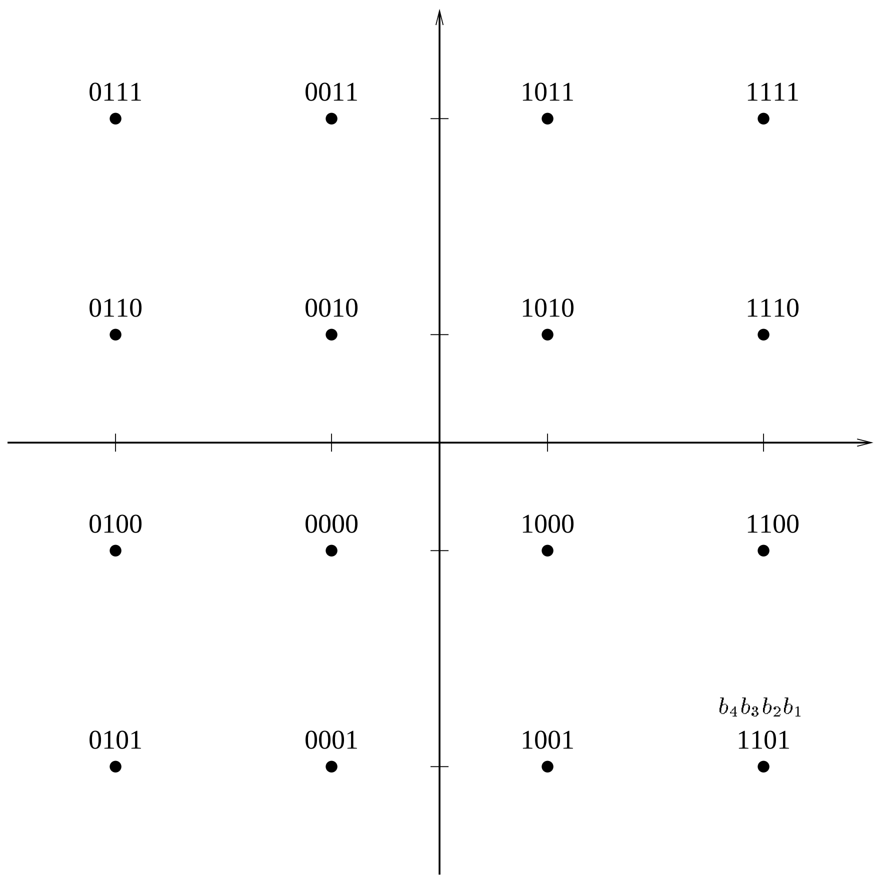


Figure 3.4: Signal mapping for a 16-QAM Gray code.

we would have:

$$P_e(b_2) \approx P_e(b_4) \approx P_e(b_1)/2 \approx P_e(b_3)/2. \tag{3.6}$$

This can be verified by comparing the number of nearest neighbors and nearest diagonal neighbors that differ in each of these four positions.

The choice of the labeling method depends on the puncturing pattern. Conventionally, to achieve spectral efficiencies higher than 1 (bit/signal) for combined turbo-code and modulation, the parity check bits are punctured and the systematic bits are all transmitted (e.g. [37], [33]). For example, for a spectral efficiency of 2 (bits/signal), every other parity bit is punctured and only one parity bit is transmitted per each information bit. It is easy to deduce that in order to compensate for the punctured parity check bits, the labeling method should assign the transmitted parity bits to the higher protected bits. Simulation and analytical results confirm this deduction [33].

Here, however, we propose a different puncturing approach based on the results developed in Chapter 4. There, it is shown that the performance of turbo-codes can be improved by assigning more energy to the parity check bits than to the systematic bits. This suggests that the performance can be improved if the desired spectral efficiency is achieved by puncturing a portion of the systematic bits as well, rather than only the parity check bits. Thus, we consider puncturing half the systematic and half the second parity check bits and transmitting all the first parity check bits. However, it should be noted that the pattern according to which the systematic bits are punctured is critical and depends on the RSC codes.

For example, consider the (5,7) turbo-code studied here. For the first

RSC code, assume that every other systematic bit is punctured and thus unknown to the decoder, and that all the parity bits are transmitted. If an error occurs at some point of the decoded block, the error can propagate through the rest of the block even if all the transmitted bits from that point on are correctly received. Fig. 3.5 illustrates this phenomenon.

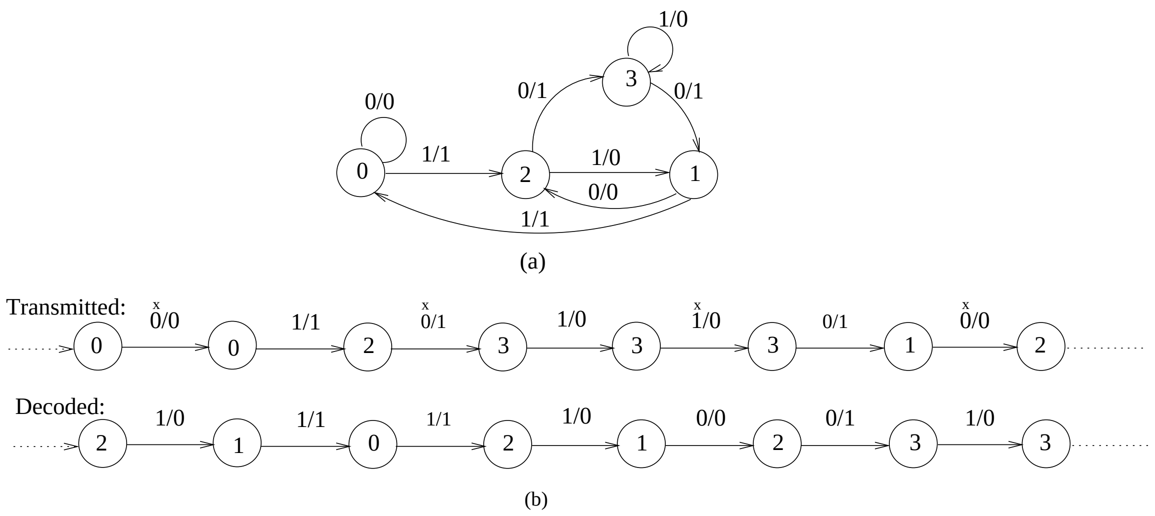


Figure 3.5: Error propagation in a (5,7) RSC code with punctured systematic bits.

Fig. 3.5 (a) shows the error state diagram of the code. Fig. 3.5 (b), shows the state transitions of the transmitted code word and a possible decoded word. On each transition i/j, i and j indicate the systematic and the parity check bits of that transition, and the systematic bits marked with a small “x” on top of them are the punctured bits. Assume that due to an error the decoder is at state 2 at some point in the decoding procedure, where as the transmitted codeword is at state 0. From this point on, as can be seen from the code state diagram, all the punctured bits can be decoded with error, even though all the transmitted systematic and parity check bits are received with no error.

For this reason, here we employ a different puncturing pattern for the systematic bits. In this pattern, the systematic bits are divided into pairs and every other pair is punctured, i.e., the transmitted systematic bits follow the pattern, $x_1^s, x_2^s, x_5^s, x_6^s, x_9^s x_{10}^s, \ldots$. In order to prevent more than two consecutive punctured systematic bits at the second decoder after interleaving is performed, we restrict the structure of the interleaver such that the punctured and un-punctured bits are interleaved among themselves. In mapping the codeword bits to the signal constellation, it is reasonable to assign the bits belonging to the punctured streams to the labeling positions that are higher protected, i.e., b_2 and b_4.

Several other mapping techniques were also examined, some of which are shown in the simulation results (Section 3.3.1) for comparison. As in the case of 8-PSK modulation, the use of extra interleaving for elimination of noise correlation seems unnecessary and even degrades the performance. Simulation results confirm this statement as shown in the following section.

3.3.1 Simulation Results

Simulations are performed for a turbo-code with (5,7) RSC component codes. The number of iterations in this case is equal to 10. Several different puncturing and labeling methods with and without extra interleaving were examined, three of which are illustrated in Fig. 3.6.

The methods shown here are as follows:

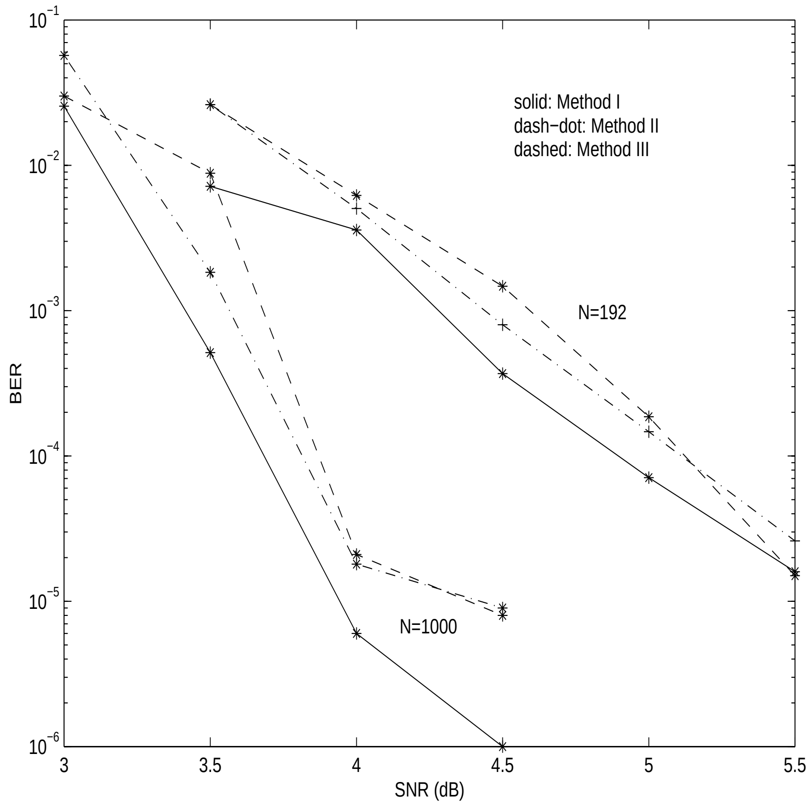


Figure 3.6: Performance of combined turbo-code and 16-QAM modulation for different puncturing and labeling methods, with and without extra interleaving.

- Method I: This method employs the puncturing pattern proposed in Section 3.3, i.e., puncturing every other pair of the systematic and parity check bits. The punctured parity check bits correspond to information bits different from those with punctured systematic bits. In this way, for every information bit in addition to the first parity bit, either the systematic or the second parity bit is transmitted. The labeling method assigns the higher protected labeling positions to the punctured streams, i.e., the systematic and second parity check bits. As mentioned before, the interleaver is randomly chosen but with the restriction that the punctured and un-punctured bits are interleaved separately.

- Method II: This method follows the conventional method of puncturing every other parity check bit and transmitting all the systematic bits. Again, labeling is performed such that the punctured streams, i.e., the first and second parity check bits are assigned to higher protected positions. No extra interleaving is performed in this case.

- Method III: This is the method that results in the best performance among the labeling methods used in [33]. This method uses the conventional puncturing, and uses extra interleavers before signal mapping to eliminate the noise correlation.

As can be seen from the simulation results, the proposed method, Method I, results in the best performance among all the methods. The improvement is in the range of 0.3 dB to more than 0.5 dB over the best results indicated in literature (Fig. 11 of [33]). The simulation results also re-confirm our

Table 3.1: Coding gains for different combined coding and 16-QAM modulation methods.

System	Proposed System	Turbo Modulation [33]	TTCM [68]	PCTCM [10]
Parameters	$N = 1000$ $M = 4$	$N = 1000$ $M = 4$	$N = 15000$ $M = 8$	$N = 16384$ $M = 16$
Coding Gain (dB)	2.1	1.6	1.0	3.2

proposition that, when Gray mapping is used, the noise correlation among the received coded bits is to our benefit and should not be eliminated.

Finally, for the sake of comparison, Table 3.1 shows the coding gains obtained over 64-state TCM at BER=10^{-5} [65], for several other proposed systems employing 16-QAM modulation with spectral efficiency of 2 (bits/signal). The complexity of these systems is similar to or higher than (larger number of states or larger interleaver lengths) the system proposed here.

3.4 Conclusion

In this chapter, systems of combined turbo-code and modulation are studied. For a coding scheme which employs 8-PSK modulation, it is shown that the unequal error protection imposed on the labeling bits should be taken into account when mapping the encoder output bits to the signal points. With

Gray mapping, the best performance is achieved when the labeling position with the lower error protection is assigned to the second parity check bit and the other two labeling positions are assigned to the systematic and the first parity check bits. This can achieve up to 0.5 dB SNR improvement without any additional complexity or delay.

The second system is the combination of turbo-code with 16-QAM signaling. For this system puncturing is performed to achieve a spectral efficiency of 2 (bits/signal). A new puncturing method is proposed which, together with appropriate interleaver and labeling method, results in more than 0.5 dB improvement over the best results shown in literature for the same system.

Conventionally, extra interleaving is performed after the turbo-encoder and before signal mapping in order to eliminate the noise correlation in the received coded bits. For both of the above systems, it is shown that using extra interleaver(s) either degrades the performance or results in marginal performance improvement for SNRs of practical interest. Thus, the extra interleaver has been eliminated in the proposed schemes and as a result the delay and complexity of the systems have been reduced.

Chapter 4

Unequal Power Allocation to the Turbo-Encoder Output Bits

Conventional turbo-codes assign an equal noise margin to the encoder output bits when BPSK modulation is employed. In this chapter, we study the effect of UPA to the encoder output bits on the performance of turbo-codes. This means providing the two groups of turbo-encoder output bits, namely the systematic and the parity check bits, with different noise margins in order to achieve a performance improvement. This problem was first addressed in [40], for very low signal to noise ratios. There, it is suggested that more power should be allocated to the systematic bits. Here, however, it is shown that as the interleaver length grows and SNR increases more power should be allocated to the parity check bits for better performance.

In order to explain the code behavior when UPA is applied, the theoretical performance evaluation of turbo-codes is first reviewed in Section 4.1. Section 4.2 studies the effect of UPA on the distance properties and the BER performance. Simulation results are shown in Section 4.3. Section 4.4 gives some final remarks on: i) the problem reformulation for applying this method when turbo-code is employed in a CDMA system; ii) comparison of the results of this chapter with those of Chapter 3; and iii) conclusion.

4.1 Average Turbo-Code Performance Bound

The performance of a turbo-code depends on the component codes and the specific interleaver employed in the structure of the code. The presence of the interleaver, however, makes the theoretical analysis of the performance practically impossible, even for relatively short interleaver lengths. The idea to evaluate the *average performance bound* of a turbo-code over all interleavers was first proposed in [13]. This average performance bound is independent of the specific interleaver employed and depends only on the component codes and the interleaver length. The idea is based on introducing the so called "uniform interleaver (UI)". A UI of length N is defined as a probabilistic device, which maps a given input of weight i to any one of its $\binom{N}{i}$ distinct permutations with equal probability $1/\binom{N}{i}$. In [13] the union bound over the ML performance of a hypothetical turbo-code employing a UI is evaluated and it is shown that this bound is the average performance bound of a turbo-code over all possible interleavers of length N.

In the following, the evaluation of this bound is briefly reviewed. Note that, because of the trellis termination at the end of each block of N input bits, turbo-code is considered as the parallel concatenation of two component block codes, denoted as C_1 and C_2 in the following.

The input-redundancy weight enumerating function (IRWEF), or WEF for short, of a systematic block code, C, of length N is defined as:

$$A^C(W,Z) \triangleq \sum_{i,j} A_{i,j} W^i Z^j, \tag{4.1}$$

where $A_{i,j}$ denotes the number of codewords originated by an input word of Hamming weight i whose parity check bits have Hamming weight j, and W and Z are dummy variables representing the systematic and parity bits, respectively. The overall weight of the codeword is then equal to $i+j$. The conditional weight enumerating function (CWEF), given the input weight i, can be obtained from the WEF as follows

$$A_i^C(Z) = \sum_j A_{i,j} Z^j. \tag{4.2}$$

This function enumerates the parity check bits of the codewords corresponding to the input words of weight i. The IRWEF can be obtained from the CWEF by the following relationship:

$$A^C(W,Z) = \sum_i W^i A_i^C(Z). \tag{4.3}$$

If the CWEFs of the component codes, $A_i^{C_1}(Z)$ and $A_i^{C_2}(Z)$, are known, the CWEF of the turbo-code employing the UI of length N can be evaluated as

$$A_i(Z) = \frac{A_i^{C_1}(Z) A_i^{C_2}(Z)}{\binom{N}{i}}. \tag{4.4}$$

The IRWEF of this code can now be found using Eq. (4.3) and represented in the same form as in Eq. 4.1,

$$A(W,Z) = \sum_{i,j} A_{i,j} W^i Z^j. \tag{4.5}$$

This function depends only on the component codes and the interleaver length of the turbo-code and is independent of the specific interleaver used. It can be shown that in Eq. (4.5), $A_{i,j}$ is the expected value of the number of codewords in the turbo-code with Hamming weights i and j in their systematic and parity check bits, respectively, over all possible interleavers of length N. For this reason, this function is called the average weight enumerating function (AWEF).

The AWEF can be used in conjunction with the union bound to compute an upper bound on the average bit error probability for ML decoding of the code. For an AWGN channel the average union bound is found as

$$P_b \le \sum_{i,j} \frac{i}{N} A_{i,j} W^i Z^j \bigg|_{W=Z=e^{-E_b/3N_0}} \tag{4.6}$$

where P_b is the average bit error rate and E_b/N_0 is the SNR per information bit and thus is multiplied by the factor 1/3 to incorporate the rate of the turbo-code. The factor $\frac{i}{N}$ is multiplied by each term in order to incorporate the measure of BER. A tighter upper bound can be found using the following formula [13]

$$P_b \le \frac{1}{2} \sum_{i,j} \frac{i}{N} \mathrm{erfc}\left(\sqrt{\frac{E_b}{3N_0}(i+j)} \right). \tag{4.7}$$

The performance obtained by the uniform interleaver is achievable by at least one deterministic interleaver [13]. In [14], the average performance bounds based on the above discussion have been compared with simulation results for turbo-codes of different interleaver lengths. The results show that the average bound is very close to those obtainable in practice by employing randomly chosen interleavers.

4.2 Unequal Power Allocation: Theoretical Analysis

For any linear block code used over an AWGN channel, the minimum weight codewords (for the asymptotic case) or the first few lowest weight codewords (for lower SNRs) are dominant in determining the code performance. For a turbo-code, each codeword consists of two groups, the systematic and the parity check bits. The role of these two different groups is not necessarily the same in the weight distribution of the dominant codewords. If one group has a higher contribution in the weight of the dominant codewords, by allocating more power to this group (and less power to the other group), the distance properties, and consequently, the performance of the code can be improved.

To study the performance of turbo-code under the effect of UPA, we consider the average weight distribution and performance bound of these codes. Consider a turbo-code with two identical RSC codes and interleaver length N. If the systematic and parity check bits are allocated equal power in transmission through the channel, a 1 in the systematic part of a codeword will have the same effect in the distance of that codeword, as a 1 in the parity part. Thus, both W and Z in Eq. (4.5) can be replaced by the same variable, e.g. w, to result in

$$A(w) = \sum_{i,j} A_{i,j} w^{i+j}. \tag{4.8}$$

Now, suppose that the bit-energy E_b is divided unequally between the systematic and the parity check bits, such that the energy assigned to each systematic bit is equal to $E_s = x\frac{E_b}{3}$ and the energies given to the first and second parity check bits are equal to $E_p = (\frac{3-x}{2})\frac{E_b}{3}$, where $x \in [0,3]$ (x=1 results in equal power allocation (EPA)) [1]. For calculating the bound in this case, W and Z in Eq. (4.5) should be replaced with w^x and $w^{(3-x)/2}$, respectively. A codeword of the original form $W^i Z^j$ is now equivalent to a codeword of distance $ix + j(\frac{3-x}{2})$. This will result in the following weight distribution as a function of x

$$\begin{aligned} A(w,x) &= \sum_{i,j} A_{i,j} w^{ix} w^{j(\frac{3-x}{2})} \\ &= \sum_{m} D_m w^m, \end{aligned} \tag{4.9}$$

where in the second equality D_m is defined as

$$D_m \triangleq \sum_{i,j:ix+j(\frac{3-x}{2})=m} A_{i,j}. \tag{4.10}$$

It should be noted that, in Eqs. (4.9) and (4.10), m can take non-integer values for $x \neq 1$.

To analyze the behavior of turbo-codes when UPA is applied and to compare this behavior for codes with long and short interleaver lengths, we

[1]The exact bit energy is equal to $\frac{M+N}{N}E_b$, where M is the number of required terminating bits. However, this does not affect our discussion.

consider two turbo-codes with (5,7) RSC component codes and interleaver lengths 760 and 20 as our examples. The weight distribution and performance bound of these codes are studied in Sections 4.2.1 and 4.2.2, respectively.

Before proceeding further, it should be noted that although in this discussion the energies assigned to the parity check bits of both component codes are considered to be equal, the role of the two parity check bits is not necessarily the same in determining the performance of the code. However, when a uniform interleaver is considered these two bits have the exact same role in the AWEF and the effect of allocating unequal power to these bits cannot be shown by this function.

4.2.1 Average Weight Distribution

As explained in Section 2.1.2, due to the presence of the interleaver, and the recursiveness of the component codes, if a low weight input word results in a low weight codeword in the first component code, it is likely that it will result in a higher weight codeword in the second component code after going through the interleaver. Thus, it is expected that, in a turbo-code, the contribution of the parity check bits be higher than that of the systematic bits in codewords with low weight, on average. The multiplicities of the first few lowest weight codewords in the AWEFs of the two turbo-codes, chosen as examples, are shown in Table 4.1. As can be seen from the table, for both codes, the expected value of the number of codewords in which the contribution of the systematic bits is higher ($j < 2i$, in a codeword of the form $W^i Z^j$) is less than those in which the contribution of the parity bits is

Table 4.1: Multiplicities of the codewords of the form W^iZ^j, with $i+j \leq 12$, in the AWEFs corresponding to (5,7) turbo-codes with $N=20$ and $N=760$.

Codeword	$N=20$	$N=760$	Codeword	$N=20$	$N=760$
W^3Z^4	0.34	0.01	W^6Z^4	0.01	0.00
W^3Z^5	0.00	0.01	W^2Z^9	0.00	3.99
W^2Z^6	0.23	0.00	W^3Z^8	6.13	0.20
W^4Z^4	0.08	0.00	W^4Z^7	0.00	0.07
W^3Z^6	1.96	0.05	W^5Z^6	0.46	0.01
W^4Z^5	0.00	0.01	W^2Z^{10}	4.81	5.98
W^5Z^4	0.03	0.00	W^3Z^9	0.00	2.30
W^2Z^8	2.58	2.00	W^4Z^8	4.96	0.15
W^3Z^7	0.00	0.10	W^5Z^7	0.00	0.05
W^4Z^6	0.95	0.02	W^6Z^6	0.34	0.00
W^5Z^5	0.00	0.01			

higher ($j > 2i$). This effect is much stronger for $N = 760$ as compared to $N = 20$. This agrees with the fact that an increase in the interleaver length reduces the probability that a low weight input block of data results in low weight outputs in both component codes, simultaneously [14].

Now, consider the effect of the UPA on the weight distribution of these codes. Fig. 4.1 shows the weight distribution of the first few codewords in AWEFs corresponding to three different levels of UPA for $N = 760$ and $N = 20$. The horizontal axis (weight) shows the total Hamming weight of the codewords, and the vertical axis (freq.) corresponds to the average number of codewords at each weight. In the diagrams corresponding to the same interleaver length, the total number of codewords is the same. For $N = 760$, it is easy to see that the distribution of the codewords gets nearer to the origin for $x = 1.4$ and further from the origin for $x = 0.6$ (with respect to the distribution corresponding to EPA). This suggests that the distance properties of this code improve as x decreases and degrade as x increases. For $N = 20$, however, in both levels of UPA the distribution spreads out in both directions and it is hard to make a comparison of the distance properties for different levels of UPA at this point.

4.2.2 Average Performance Bound

The average performance bound for different levels of UPA is obtained according to the following formula

$$P_b(x) \leq \frac{1}{2} \sum_{m=m_{\min}(x)}^{m_{\max}(x)} \hat{D}_m \operatorname{erfc}(\sqrt{m \frac{E_b}{3N_0}}), \tag{4.11}$$

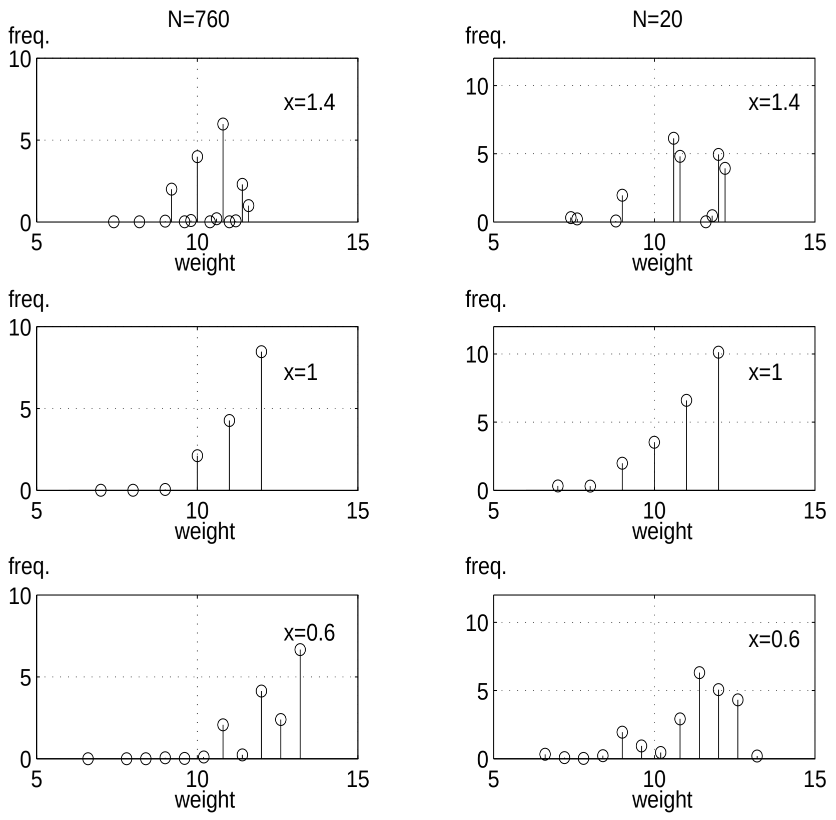


Figure 4.1: Weight distributions corresponding to different levels of UPA applied to (5,7) turbo-codes with $N = 760$ and $N = 20$.

where $P_b(x)$ is the probability of bit error and is a function of x, and $\hat{D}_m$ is defined as

$$\hat{D}_m \triangleq \sum_{i,j:ix+j(\frac{3-x}{2})=m} \frac{i}{N} A_{i,j}. \tag{4.12}$$

In Eq. (4.11), $m_{\min}(x)$ is the minimum value of m corresponding to x. In choosing $m_{\max}(x)$ as the truncation weight of the summation, the following criteria are considered in order to make a fair comparison of the bounds corresponding to different levels of UPA:

Criterion 1:

$m_{\max}(x)$ is chosen such that the value of $\sum_{m=m_{\ min}}^{m_{\max}(x)} D_m$ is almost constant for different values of x. This criterion results in roughly the same number of code words for different values of x in evaluating the uniform performance bound. However, since the distribution of the weights varies with respect to the value of x, this results in different values for the maximum weight codewords that are incorporated in evaluating the bound, and this may not necessarily make a fair comparison. Thus a second criterion is considered as well;

Criterion 2:

Here, $m_{\max}(x)$ is chosen to be equal for all values of x, so that the union bound is evaluated by incorporating codewords of up to the same weight, but not necessarily the same number of codewords.

The average of the bounds evaluated based on the above criteria gives a close approximation of the actual behavior of the turbo-code as will be seen in the next section. The results for $N = 760$ with $E_b/N_0 = 2$ dB and for $N = 20$ with $E_b/N_0 = 3$ dB are shown in Table 4.2. In obtaining these results, $m_{\max}(1)$ is chosen such that the bound is close enough to the corresponding simulation result.

Table 4.2: Average performance bounds for (5,7) turbo-codes with $N = 760$ and $N = 20$, for different UPA levels.

	$N = 760, E_b/N_0 = 2$ (dB)		$N = 20, E_b/N_0 = 3$ (dB)	
x	Criterion 1	Criterion 2	Criterion 1	Criterion 2
1.4	3.3×10^{-5}	3.9×10^{-5}	7.1×10^{-4}	7.2×10^{-4}
1.2	2.3×10^{-5}	2.3×10^{-5}	6.4×10^{-4}	6.1×10^{-4}
1	1.8×10^{-5}	1.8×10^{-5}	5.8×10^{-4}	5.8×10^{-4}
0.8	1.5×10^{-5}	9.9×10^{-6}	5.6×10^{-4}	5.8×10^{-4}
0.6	1.3×10^{-5}	9.2×10^{-6}	1.1×10^{-3}	1.1×10^{-3}
0.4	1.0×10^{-5}	5.6×10^{-6}	6.9×10^{-4}	6.7×10^{-4}
0.2	8.8×10^{-6}	4.1×10^{-6}	8.6×10^{-4}	9.4×10^{-4}
0	5.1×10^{-6}	4.9×10^{-6}	1.3×10^{-3}	1.8×10^{-3}

For $N = 760$, the WEFs corresponding to every branch in the hyper-trellis of the turbo-code have been approximated by the WEF corresponding to the branch which connects the all-zero states in both component codes, as in [14]. Also, in order to limit the amount of computations in developing the AWEF, the transfer function corresponding to the component codes has to be truncated to codewords of Hamming weights less than a threshold.

Since codewords of large weights do not affect and are not incorporated in developing the average bound, this truncation does not affect the bound for $x = 1$. However, for $x \neq 1$, there might exist codewords which should have been encountered in the summation because of their low effective weight as a result of the value of x, but were excluded from the truncated transfer function due to their high original weight. For this reason, the threshold should be appropriately selected to reduce this side effect. Here, we have enumerated codewords of Hamming weights less than or equal to 75 in the component transfer function. This way, only the bounds corresponding to values of $x < 0.2$ will be affected by this truncation. Thus, in Table 4.2, the average bounds shown for $x = 0$ are under estimated. For $N = 20$, no approximation has been used and the component wise transfer function has been evaluated completely.

As can be seen from Table 4.2, for $N = 760$, performance improvement is achieved by reducing the protection of the systematic bits. The two criteria show slightly different behavior for $x < 0.4$. For $N = 20$, the best performance is achieved by EPA, or very close to EPA. Fig. 4.2 shows the average of the union bounds obtained according to these two criteria. As can be seen, for $N = 760$, an improvement of about 0.5 in the $\log_{10}(\text{BER})$ can be achieved for $x = 0.2$. Referring to the figure, this corresponds to more than 0.5 dB improvement in the signal to noise ratio over the case of equal power allocation. For $N = 20$ however, the three lowest bounds, corresponding to $x = 0.8, 1, 1.2$, are almost indistinguishable and the bound for $x = 0.4$ shows higher BERs.

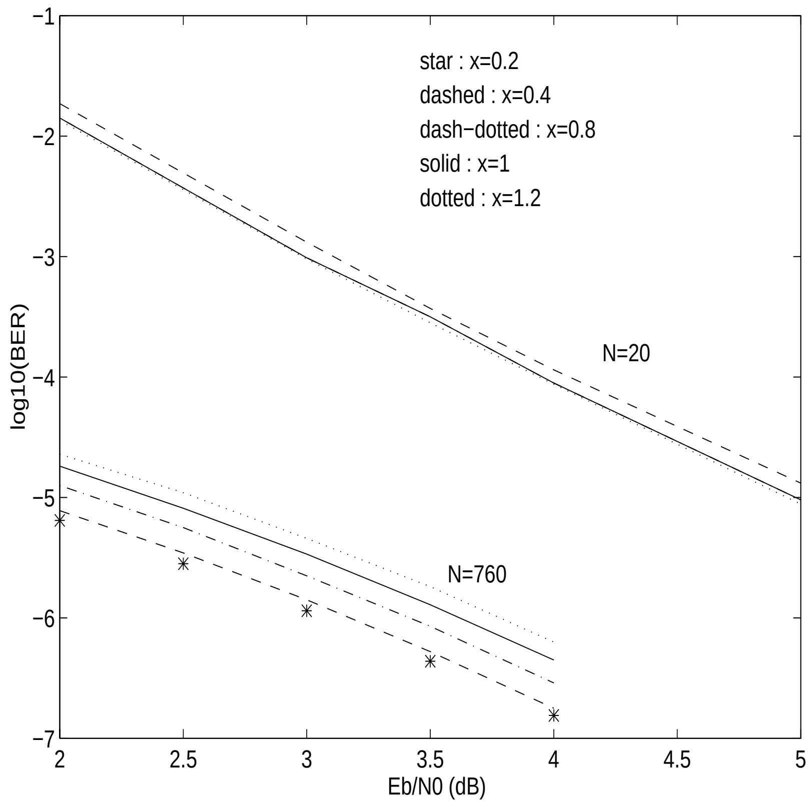


Figure 4.2: Average performance bounds of (5,7) turbo-codes with interleaver lengths 20 and 760 for different levels of UPA.

4.3 Simulation Results and Discussion

Simulations are performed for signaling over AWGN channels. The decoding is performed using the modified BCJR algorithm, where the Gaussian distributions of each received bit is based on its assigned noise margin. Fig. 4.3 shows the BER performance for different values of x and several interleaver lengths. As can be seen, the effect of UPA is negligible for a short interleaver length ($N = 20$). For larger interleaver lengths, better performance is achieved when the systematic bits are protected less than the parity check bits and this effect gets stronger as the block length increases.

It is also observed that as x decreases to very low values, there is a sudden rise in the BER of the simulation results. This degradation is due to the sub-optimum nature of the turbo-decoding algorithm. When the error protection over the systematic bits becomes too low, the extrinsic information passed to the second component decoder is very unreliable, and consequently, the iterative decoding procedure does not converge to (or close to) the ML performance.

In order to observe the effect of UPA for SNRs below the error floor of the performance curves, simulation results for this region are shown in Fig. 4.4. As can be seen, allocating more power to the parity bits results in better performance after the waterfall region, and also for $x = 0.2$ the curve is much more steep in the waterfall region.

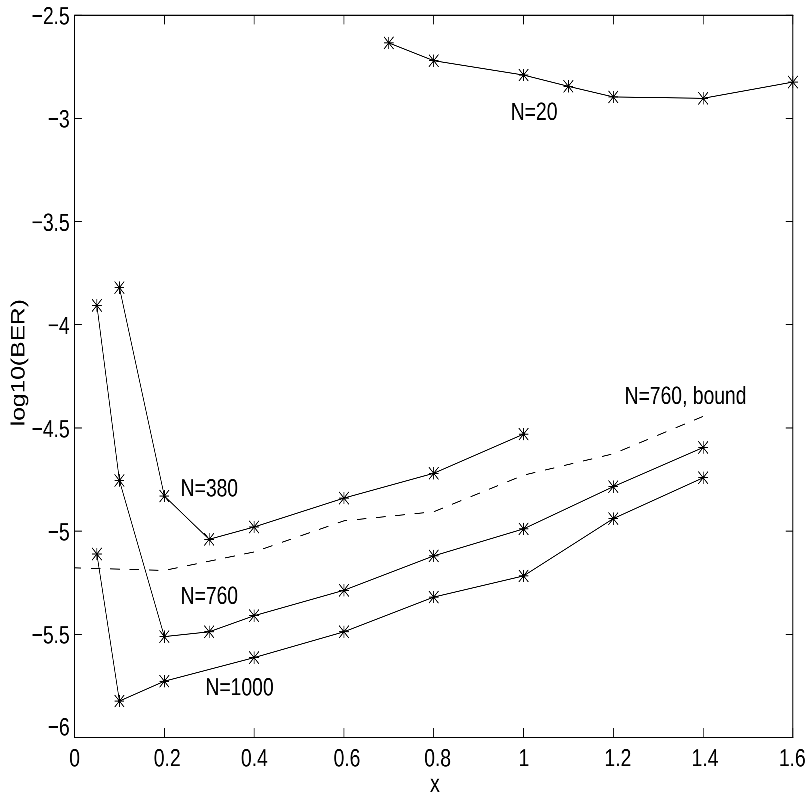


Figure 4.3: Simulation results corresponding to (5,7) turbo-codes with $N = 380, 760$, and 1000 at $E_b/N_0 = 2$ dB, and $N = 20$ at $E_b/N_0 = 3$ dB.

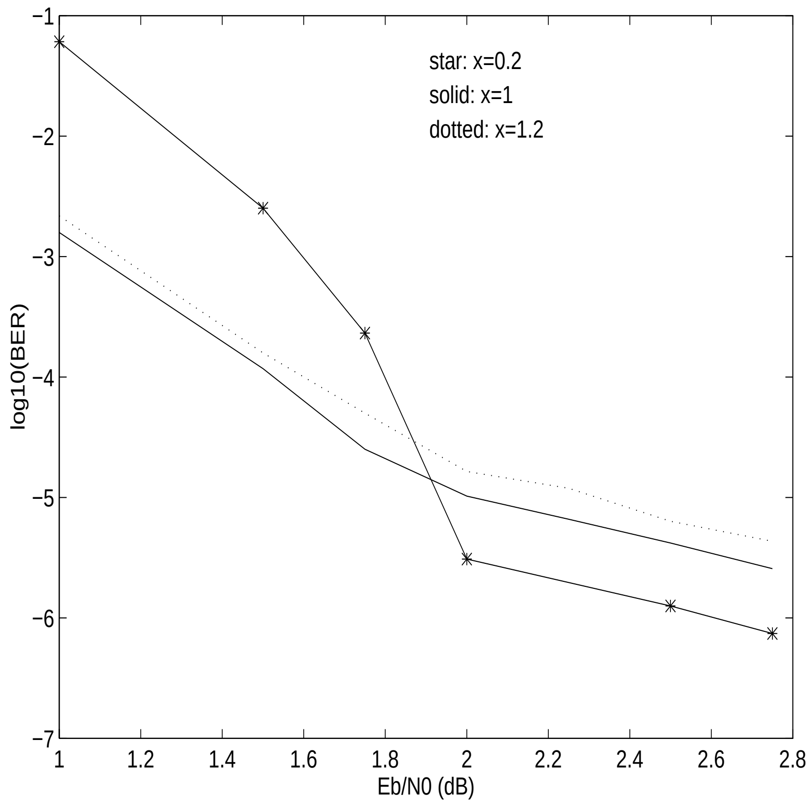


Figure 4.4: Simulation results corresponding to different UPA levels for (5,7) turbo-code with $N = 760$, in the waterfall region.

4.4 Final Remarks

4.4.1 Application to CDMA Systems [56]

In this section we show an alternative formulation of the proposed system based on the application of turbo-codes for signaling over a CDMA channel. A usual practice in CDMA systems for matching of rate is based on repeating the encoder output bits. This feature provides us with a practical method of implementing the UPA with only a negligible increase in the complexity.

To achieve UPA, we consider a CDMA system in which the systematic and the parity bits are repeated k_s and k_p times, respectively. The effective energies assigned to the systematic and parity bits in this case are then equal to $E_s = \frac{k_s E_b}{k_s+2k_p}$ and $E_p = \frac{k_p E_b}{k_s+2k_p}$, respectively. With this definition Eq. (4.9) can now be written as

$$A(W,Z) = \sum_{i,j} A_{i,j}\; w^{i\frac{k_s}{k_s+2k_p}}\; w^{j\frac{k_p}{k_s+2k_p}}. \tag{4.13}$$

It can be easily verified that this coding scheme is equivalent to a turbo-code employing a UPA of level $x = 3k_s/(k_s+2k_p)$.

4.4.2 Comparison with Combined Turbo-Code and Modulation

In Chapter 3, where combined turbo-code and modulation is considered, it is shown that the best performance of the system, when 8-PSK modulation is employed, can be achieved by assigning the two labeling positions with higher noise protection to the systematic and the first parity check bits, and the labeling position with less protection to the second parity check bits. This may seem to be in contradiction with the results developed in this chapter, at first glance. However, it should be noticed that the comparison between the two systems is not a fair comparison to begin with. Note that the performance of a system of combined turbo-code with multilevel modulation is determined by the Euclidean distance between the codewords, not the Hamming distance which is case when BPSK modulation is employed. The Euclidean weight distribution in a turbo-coded modulation system depends not only on the weights of the systematic and parity parts of the codewords, but also on the relative positions of these bits in each codeword, because every three output bits are grouped together and mapped to one signal. However, as stated in Chapter 3, when extra interleaving is used to eliminate the correlation between the bits mapped to the same signal, the results are consistent with the results developed in this chapter.

4.4.3 Conclusion

In this chapter, the effect of applying UPA to the turbo-encoder output bits is studied. It is shown that the roles of the two groups of turbo-encoder output bits, the systematic and the parity check bits, are not the same in determining the code performance and, when binary modulation is employed, the code performance can be improved by allocating unequal power to these bits. For turbo-codes of very short interleaver lengths, the protection over the systematic information should be more than the parity information, although the performance improvement is negligible. Studying the AWEF suggests that as the interleaver length gets larger, the contribution of the parity check bits in the distance of the low weight codewords increases. Thus, for large interleaver lengths, higher protection of these bits will improve the code performance. The theoretical bounds that are obtained for different levels of UPA agree with the simulation results for a wide range of UPA. Improvements of about 0.5 in the $\log_{10}(\mathrm{BER})$ can be achieved by selecting the proper level of UPA over EPA. This is approximately equivalent to 0.5 dB improvement in the SNR, in the range of the BERs of interest for voice transmission $(10^{-3.5}, 10^{-2.5})$. Simulation results also show that UPA is most beneficial when turbo-code is operating at the beginning of the error floor region. The proposed coding scheme can be applied to CDMA systems by unequally repeated transmission of the coded bits.

Chapter 5

Variance of the Turbo-Code Performance Bound

As mentioned before, the interleaver plays a key role in the pseudo-random nature and consequently the high performance of turbo-codes, by reordering the input block of data given to the second encoder. Thus, the study and design of the interleaver has become an attractive subject for many researchers in this area and many publications can be found on the design of the interleaver, some of which are mentioned in the literature review.

Although these works implicitly suggest some conclusions regarding the effect of different choices of interleavers on the performance of turbo-codes, they are mainly focused on either search algorithms for the best (or at least *good*) interleaver(s) or explaining the behavior of these codes in general. So far, the only statistical study of the turbo-code performance with respect to interleavers, considers the upper bound on the ML performance of the turbo-code, averaged over all possible interleavers (e.g. [14]).

If higher order statistic averages of the turbo-code performance with respect to the interleaver are known, it will be possible to have a more accurate estimate of the distribution of the performance bound with respect to the interleaver.

As a first step, in this chapter, we study the effect of the interleaver on turbo-codes by looking at the variance of the turbo-code performance bound with respect to all possible interleavers of the same length. This study is intended to tackle the question brought up in [14]: "For a given interleaver length, how sensitive is the performance to the interleaver choice?" and to give more insight regarding what performance to expect from a turbo-code with fixed component codes and interleaver length. The results of this study are also expected to provide an estimate of how well a particular interleaver performs among the range of all possible interleavers and help to evaluate the effectiveness of an interleaver search algorithm.

The same approach in evaluating the variance of the bound can be used to evaluate the contribution of codewords of different forms in the variance of the bound and the correlation between the number of codewords of different forms. These results can then be used to generate algorithms for interleaver design, or be considered in existing algorithms. This idea is briefly presented later in this chapter.

This chapter is organized as follows. In Section 5.1, the mathematical formulations for developing the second moments of the WEF and union performance bound for a turbo-code are derived. Asymptotic analysis of the derived formulas for large interleaver length is presented in Section 5.2. Section 5.3 shows some numerical and simulation results for the non-asymptotic and asymptotic cases, and explains the approach in deriving those results. Section 5.4, extends the results of previous sections and proposes general guidelines for using these results in interleaver design algorithms. Finally, Section 5.5 concludes the chapter.

5.1 Variance of the Performance Bound

The concept of uniform interleaver as a tool to evaluate the average performance bound for turbo-codes was explained in Chapter 4. A similar concept is used here to evaluate the second order moment of the performance bound. To do this, however, some new variables and notations need to be introduced. In order to keep the consistency of the notations in the mathematical formulas of this chapter, we start by repeating the main formulas of the AWEF and average performance bound derived in Section 4.1, incorporating these new notations.

As mentioned in Chapter 4, the WEF of a turbo-code employing a UI of length N is equivalent to the average weight enumerating function, or in other words, the expected value of the WEF of the turbo-code over all possible interleavers of this length. Thus for a rate 1/3 turbo-code of interleaver length N, the AWEF can be written as

$$\mathbf{E}[A] = \sum_{i,j_1,j_2} \mathbf{E}[X_{i,j_1,j_2}] W^i Z_1^{j_1} Z_2^{j_2} \tag{5.1}$$

where A is the random WEF with respect to the interleaver, X_{i,j_1,j_2} is the random variable representing the number of codewords with Hamming weights

i, j_1, and j_2 in the systematic, first RSC parity check, and second RSC parity check bits, respectively, and $\mathbf{E}[\cdot]$ is the expectation operation over all interleavers of length N.

The average union bound can then be written as

$$\mathbf{E}[B] = \sum_{i,j_1,j_2} \frac{i}{N} \mathbf{E}[X_{i,j_1,j_2}] Q(i, j_1, j_2). \tag{5.2}$$

where B is the performance bound corresponding to a random interleaver and $Q(i, j_1, j_2) = \frac{1}{2}\mathrm{erfc}(\sqrt{\frac{E_b}{3N_0}(i + j_1 + j_2)})$. To evaluate the second moment of the performance bound,

$$\mathbf{E}[B^2] = \sum_{i,j_1,j_2} \sum_{i',j_1',j_2'} \frac{ii'}{N^2} \mathbf{E}[X_{i,j_1,j_2} X_{i',j_1',j_2'}]\, Q(i, j_1, j_2)\, Q(i', j_1', j_2'), \tag{5.3}$$

we start with the mean square of the WEF (MSWEF), which is equal to

$$\mathbf{E}[A^2] = \sum_{i,j_1,j_2} \sum_{i',j_1',j_2'} \mathbf{E}[X_{i,j_1,j_2} X_{i',j_1',j_2'}]\, W^i Z_1^{j_1} Z_2^{j_2} W'^{i'} Z_1'^{j_1'} Z_2'^{j_2'}. \tag{5.4}$$

Based on the definition of UI, coding with a turbo-code that employs this interleaver is equivalent to coding with a turbo-code which randomly chooses an interleaver for each block of input data independently from those of the previous blocks [14]. Thus, the random variables X_{i,j_1,j_2} and $X_{i',j_1',j_2'}$ are independent unless $(i, j_1) = (i', j_1')$. This, however, is not true when using a practical interleaver. For a practical turbo-code, if the value of one random variable (e.g., X_{i,j_1,j_2}) is fixed, some restriction is imposed on the structure of the interleaver and thus on the value of other random variables (e.g., $X_{i',j_1',j_2'}$). The assumption of the independence, however, does not affect the validity of Eq. (5.1), since only the first moment of these variables is evaluated in that case. On the other hand, to evaluate the higher order statistics of the WEF, the correlation between the above random variables has to be taken into account. Thus, in evaluating the MSWEF, we assume that every probabilistic experiment consists of choosing any one of the $N!$ possible interleavers with equal probability, and fixing it for the rest of that experiment.

For simplicity in notations, in the following, X_{i,j_1,j_2} and $X_{i',j_1',j_2'}$ are replaced with X and X', respectively. In order to find $\mathbf{E}[XX']$, the following definitions are made: Let $S_{x,y}$ be the set of all input words of weight x which result in codewords of parity weight y from the first component code, and $S^I_{x,y}$ be the set of all input words of weight x which result in codewords of weight y from the second component code. Also let:

i) s_g, $1 \leq g \leq G$, be the elements of S_{i,j_1},

ii) $s'_{g'}$, $1 \leq g' \leq G'$, be the elements of $S_{i',j_1'}$,

iii) s^I_k, $1 \leq k \leq K$, be the elements of S^I_{i,j_2},

iv) $s'^I_{k'}$, $1 \leq k' \leq K'$, be the elements of $S^I_{i',j_2'}$.

Letting

$$X_g = \begin{cases} 1, & \text{if } s_g \text{ is re-ordered to an element of } S^I_{i,j_2} \text{ by the interleaver} \\ 0, & \text{otherwise} \end{cases}$$

and

$$X'_{g'} = \begin{cases} 1, & \text{if } s'_{g'} \text{ is re-ordered to an element of } S^I_{i',j_2'} \text{ by the interleaver} \\ 0, & \text{otherwise} \end{cases}$$

we have

$$
\begin{aligned}
X &= X_1 + X_2 + \ldots + X_G \qquad (5.5)\\
X' &= X_1' + X_2' + \ldots + X_{G'}'
\end{aligned}
$$

and consequently,

$$\mathbf{E}[XX'] = \mathbf{E}[\sum_{g=1}^{G} X_g \sum_{g'=1}^{G'} X_{g'}'] = \sum_{g=1}^{G}\sum_{g'=1}^{G'} \mathbf{E}[X_g X_{g'}']. \qquad (5.6)$$

Since the random variables X_g and $X_{g'}'$ only take values of 0 or 1, we have

$$
\begin{aligned}
\mathbf{E}[X_g X_{g'}'] &= Pr(X_g = 1 \text{ and } X_{g'}' = 1)\\
&= Pr[\bigcup_{k=1}^{K}\bigcup_{k'=1}^{K'} (s_g \rightarrow s_k^I \bigcap s_{g'}' \rightarrow s_{k'}'^I)]\\
&= \sum_{k=1}^{K}\sum_{k'=1}^{K'} Pr(s_g \rightarrow s_k^I \bigcap s_{g'}' \rightarrow s_{k'}'^I) \qquad (5.7)
\end{aligned}
$$

where "$\rightarrow$" indicates the re-ordering process using the interleaver, $Pr(s_g \rightarrow s_k^I \bigcap s_{g'}' \rightarrow s_{k'}'^I)$ is the probability that the randomly chosen interleaver re-orders the input words s_g and $s_{g'}'$ into s_k^I and $s_{k'}'^I$, respectively, and the last equality results from the fact that the events $(s_g \rightarrow s_k^I \bigcap s_{g'}' \rightarrow s_{k'}'^I)$ are disjoint for different values of k or k'. Substituting (5.7) in (5.6), we get

$$\mathbf{E}[XX'] = \sum_{g=1}^{G}\sum_{g'=1}^{G'}\sum_{k=1}^{K}\sum_{k'=1}^{K'} Pr(s_g \rightarrow s_k^I \bigcap s_{g'}' \rightarrow s_{k'}'^I). \qquad (5.8)$$

The following theorem evaluates the probability involved in (5.8).

Theorem 1: If s_g and $s_{g'}'$ have r 1's in common positions, and s_k^I and $s_{k'}'^I$ have r^I 1's in common positions, where $0 \leq r, r^I \leq \min(i, i')$, then

$$Pr(s_g \rightarrow s_k^I \bigcap s_{g'}' \rightarrow s_{k'}'^I) = \begin{cases} \frac{r!(i-r)!(i'-r)![N-(i+i'-r)]!}{N!} & \text{if} \quad r = r^I \\ 0 & \text{if} \quad r \neq r^I. \end{cases} \qquad (5.9)$$

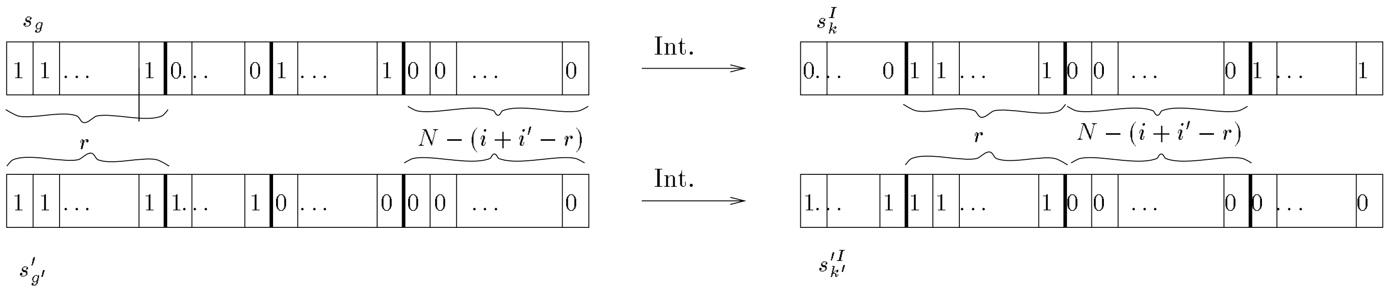


Figure 5.1: An example of two input words and their interleaved versions with r commonly positioned 1's.

Proof

Since s_g and $s'_{g'}$ have r commonly positioned 1's, their interleaved versions should have the same number of commonly positioned 1's, as well. Thus, there exists no interleaver that realizes this event when $r \neq r^I$, i.e.,

$$Pr(s_g \to s_k^I \bigcap s'_{g'} \to s'^I_{k'}) = 0, \quad \text{if } r \neq r^I.$$

Fig. 5.1 shows a schematic example of s_g and $s'_{g'}$, interleaved to s_k^I and $s'^I_{k'}$, respectively, where $r = r^I$.

In this case the interleaver has the following properties:

- mapping the r positions of the common 1's in s_g and $s'_{g'}$ to the r positions of the common 1's in s_k^I and $s'^I_{k'}$;
- mapping the $(i-r)$ remaining 1's in s_g to the positions of the remaining $(i-r)$ 1's in s_k^I;
- mapping the $(i'-r)$ remaining 1's in $s'_{g'}$ to the positions of the remaining $(i'-r)$ 1's in $s'^I_{k'}$;

- finally, mapping the $[N-(i+i'-r)]$ positions of the common 0's in s_g and $s'_{g'}$ to the positions corresponding to the common 0's in s^I_k and $s'^I_{k'}$.

Within each of the above bit groups, the permutation of the bits does not matter. Thus, the number of interleavers satisfying the above conditions is

$$r!\,(i-r)!\,(i'-r)!\,[N-(i+i'-r)]!$$

and consequently, the probability that one of these interleavers is selected when randomly picked from the ensemble of all $N!$ interleavers is

$$\frac{r!(i-r)!(i'-r)![N-(i+i'-r)]!}{N!}$$

and the proof is complete. □

We denote the non-zero part of the probability in Eq. (5.9) with $\rho_r(i,i')$. Note that in (5.8), for each pair of input words $(s_g, s'_{g'})$ with r commonly positioned 1's, only those pairs of input words $(s^I_k, s'^I_{k'})$ which also have r commonly positioned 1's result in the non-zero terms $\rho_r(i,i')$. Let $q_r(i,j_1,i',j'_1)$ and $q_r(i,j_2,i',j'_2)$ denote the number of input word pairs $(s_g, s'_{g'})$ and $(s^I_k, s'^I_{k'})$ respectively, where each pair has r commonly positioned 1's. Eq. (5.8) can now be written as

$$\mathbf{E}[XX'] = \sum_{r=0}^{\min(i,i')} q_r(i,j_1,i',j'_1)q_r(i,j_2,i',j'_2)\rho_r(i,i'). \tag{5.10}$$

And finally, substituting Eq. (5.10) in Eq. (5.3) results in

$$\mathbf{E}[B^2] = \sum_{i,j_1,j_2}\sum_{i',j'_1,j'_2}\sum_{r=0}^{\min(i,i')} \frac{ii'}{N^2}\, q_r(i,j_1,i',j'_1)q_r(i,j_2,i',j'_2)\rho_r(i,i')Q(i,j_1,j_2)\,Q(i',j'_1,j'_2) \tag{5.11}$$

The variance of the bound can then be evaluated as

$$\mathbf{var}[B] = \mathbf{E}[B^2] - \mathbf{E}^2[B]. \tag{5.12}$$

5.2 Asymptotic Behavior

In this section we study the MSWEF and $\mathbf{E}[B^2]$ for a large interleaver length. The function $q_r(i, j, i', j')$, defined in Section 5.1, depends on N as well as the component codes. In the following, this function will be further analyzed in order to be able to represent the MSWEF and $\mathbf{E}[B^2]$ in the form of polynomials in N. Once this is achieved, it can be said that, the behavior of these functions for asymptotically large N is dominated by the terms with the highest power of N.

A codeword of the form $W^i Z^j$ in each component code, is constructed by the concatenation of n, $1 \leq n \leq \lfloor i/2 \rfloor$, error events with possible zeros in between consecutive error events. Suppose two input words, s and s', consisting of n and n' error events, respectively, have r commonly positioned 1's. Furthermore, assume that these r common bits are contained in m and m' error events of s and s', respectively, where $\min(1, r) \leq m \leq \min(r, n)$ and $\min(1, r) \leq m' \leq \min(r, n')$. We call these error events the *tied up* events. The reason for choosing this name is that each of these error events is bound to be placed in certain position(s) relative to one or more tied up events of the other codeword, such that the two codewords have r commonly positioned 1's in their corresponding input words. The remaining $(n - m)$ and $(n' - m')$ error events (called *loose* events) can be placed anywhere along the block of length N as long as they do not cause extra overlapping 1's between the two input words. Fig. 5.2 illustrates the above definitions, where the horizontal lines represent the all-zero path and the diverged paths represent the error events.

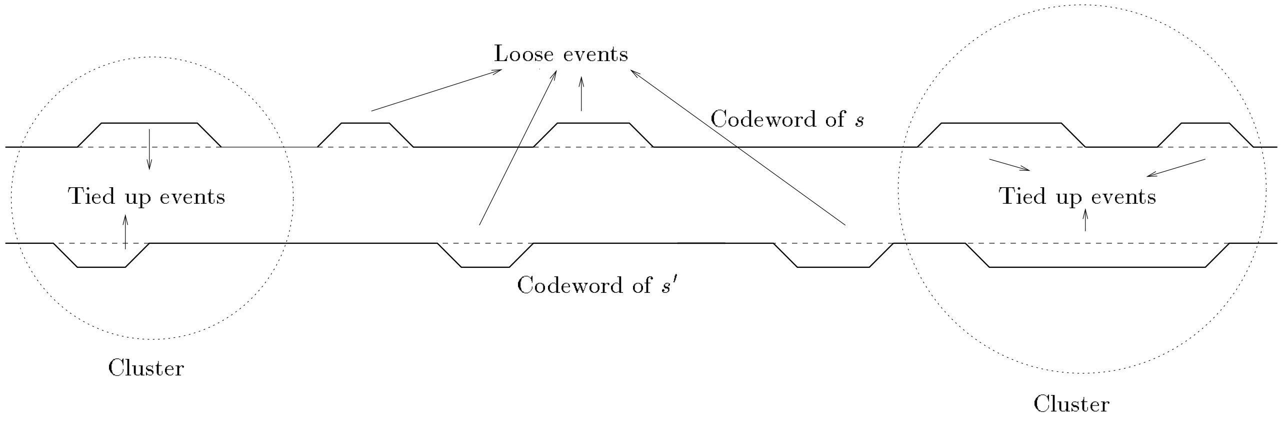


Figure 5.2: Schematic illustration of the relative positions of two codewords resulting from input words s and s' with $n = 5, n' = 4, m = 3, m' = 2$, and $c = 2$.

The tied up events form c "cluster" of events, $\min(1, r) \leq c \leq \min(m, m')$. These c clusters plus the $(n-m)+(n'-m')$ loose error events can be placed in different positions along the block of length N by adding zeros between them or placing them adjacent to each other. This can be done in $\binom{N-L_t+1}{n+n'-m-m'+c}$ ways, where L_t is the summation of the lengths of the clusters and loose events in both codewords. The length of the cluster is defined as the distance between the point where the first error event in the cluster starts, up to the point where the last error event in that cluster ends. In the following formulas for asymptotic behavior, L_t is eliminated with respect to N in order to reduce the complexity of computations.

Now, $q_r(i,j,i',j')$ can be written as

$$q_r(i,j,i',j') = \sum_n \sum_{n'} \sum_m \sum_{m'} \sum_c \hat{q}_r(i,j,i',j',n,n',m,m',c) \binom{N}{n+n'-m-m'+c} \tag{5.13}$$

where $\hat{q}_r(i,j,i',j',n,n',m,m',c)$ is the number of codeword pairs of the form $(W^iZ^j, W'^{i'}Z'^{j'})$ with n and n' error events; these error events have r commonly positioned 1's, resulting in m and m' tied up events which form c clusters. Note that $\hat{q}_r$ depends only on the component codes and not on N. As a result, Eq. (5.10) can be re-written as

$$\mathbf{E}[XX'] = \sum_r \sum_{e_1 \in \mathcal{E}_1} \sum_{e_2 \in \mathcal{E}_2} C_{e_1}\, f_{e_1}(N)\, C_{e_2}\, f_{e_2}(N) \rho_r(i,i') \tag{5.14}$$

where $\mathcal{E}_1$ and $\mathcal{E}_2$ are the sets of the possible 5-tuples $(n_1, n_1', m_1, m_1', c_1)$ and $(n_2, n_2', m_2, m_2', c_2)$, respectively, and

$$C_{e_1} = \hat{q}_r(i, j_1, i', j_1', n_1, n_1', m_1, m_1', c_1) \tag{5.15}$$

$$f_{e_1}(N) = \binom{N}{n_1 + n_1' - m_1 - m_1' + c_1} \tag{5.16}$$

$$C_{e_2} = \hat{q}_r(i, j_2, i', j_2', n_2, n_2', m_2, m_2', c_2) \tag{5.17}$$

$$f_{e_2}(N) = \binom{N}{n_2 + n_2' - m_2 - m_2' + c_2}. \tag{5.18}$$

Substituting Eq. (5.14) in Eq. (5.4) and using the approximation $\binom{a}{b} \approx a^b/b!$ for $a \gg b$, we have

$$\mathbf{E}[A^2] \approx \sum_{t \in \mathcal{T}} F_t\, N^{G_t}\, W^i Z_1^{j_1} Z_2^{j_2} W'^{i'} Z_1'^{j_1'} Z_2'^{j_2'} \tag{5.19}$$

where $\mathcal{T}$ represents the set of possible 17-tuples of all the variables involved in the summation, $G_t = n_1+n'_1+n_2+n'_2-m_1-m'_1-m_2-m'_2-i-i'+c_1+c_2+r$, and F_t is not a function of N. The following theorem finds the conditions under which G_t takes its maximum value.

Theorem 2: The maximum value of G_t is equal to 0 and is achieved if and only if the following conditions hold

$$\begin{gathered} i = 2k, \\ n_1 = n_2 = i/2 = k, \\ i' = 2k', \\ n'_1 = n'_2 = i'/2 = k', \\ r = 2l, \\ m_1 = m_2 = m'_1 = m'_2 = c_1 = c_2 = r/2 = l, \end{gathered} \tag{5.20}$$

where $k, k' = 1, 2, 3, \ldots$, and $l = 0, 1, 2, \ldots, \min(k, k')$.

Proof:

It can be easily seen that, in G_t, maximizing the terms corresponding to subscript 1 and those corresponding to 2 can be performed independently, i.e.,

$$(G_t)_{\max} = (n_1+n'_1-m_1-m'_1+c_1)_{\max}+(n_2+n'_2-m_2-m'_2+c_2)_{\max}-i-i'+r.$$

Thus, in this proof, we only find $M_1 = (n_1 + n'_1 - m_1 - m'_1 + c_1)_{\max}$ as the maximization corresponding to the second RSC code follows with exactly the same analogy.

Suppose m_1 of the n_1 error events of the input word s are tied up with m'_1 of the n'_1 error events of the input word s', and the tied up events result in c_1 clusters. Since s and s' have r commonly positioned 1's, together they contain $i + i' - r$ positions containing a 1. Define a "bunch" of events to be either a cluster or a loose event. The number of bunches is $b = c_1 + (n_1 - m_1) + (n_1' - m_1')$. Since for recursive convolutional codes each error event has a Hamming weight of at least two in the systematic part, and each bunch contains at least one error event,

$$c_1 + n_1 - m_1 + n_1' - m_1' \leq \frac{i + i' - r}{2}. \tag{5.21}$$

The equality holds if and only if each loose event and each cluster contains exactly two positions containing a 1. This condition is met by codewords satisfying the following conditions:

- Each codeword is constructed of error events with input weight 2.
- Each error event is either a loose event or exactly matches and is tied up with an error event of the other codeword.

Similarly,

$$c_2 + n_2 - m_2 + n_2' - m_2' \leq \frac{i + i' - r}{2} \tag{5.22}$$

with equality holding under the same conditions. As a result,

$$(G_t)_{\max} = 0. \tag{5.23}$$

$\square$

In addition, Appendix B gives an alternative proof of Theorem 2, which is mathematically more detailed and elaborates on the contribution of codeword pairs (with different systematic weights and relative positions) to the variance of the bound.

Keeping only the terms with the largest power of N and defining a new function $\tilde{q}(k, j, k', j', l) \triangleq \hat{q}_{2l}(2k, j, 2k', j', k, k', l, l, l)$ result in the following formula for the asymptotic MSWEF

$$\mathbf{E}[A^2] = \sum_{(k,k',j_1,j_2 j_1',j_2',l)} \frac{(2l)!(2k-2l)!(2k'-2l)!}{[(k+k'-l)!]^2}$$
$$\cdot \tilde{q}(k, j_1, k', j_1', l)\, \tilde{q}(k, j_2, k', j_2', l)\, W^i Z_1^{j_1} Z_2^{j_2} W'^{i'} Z_1'^{j_1'} Z_2'^{j_2'}. \tag{5.24}$$

The asymptotic formula for the mean square of the bound is then

$$\mathbf{E}[B^2] = \sum_{(k,k',j_1,j_2 j_1',j_2',l)} \frac{(2k)(2k')}{N^2} \frac{(2l)!(2k-2l)!(2k'-2l)!}{[(k+k'-l)!]^2}$$
$$\cdot \tilde{q}(k, j_1, k', j_1', l)\, \tilde{q}(k, j_2, k', j_2', l)\, Q(i, j_1, j_2)\, Q(i', j_1', j_2'). \tag{5.25}$$

As can be seen from Eq. (5.25), $\mathbf{E}[B^2]$ is proportional to N^{-2}. On the other hand, $\mathbf{E}[B]$ is proportional to N^{-1} for asymptotically large N [15]. Thus, the variance of the performance bound is also proportional to N^{-2}. As a result, we see that the coefficient of variation ($\mathbf{cv}[B] = \mathbf{stdv}[B]/\mathbf{E}[B]$) of the performance bound, i.e., the ratio of the standard deviation to the mean, does not change with N as N approaches infinity.

5.3 Numerical Results and Discussion

To reduce the operational complexity, the error functions are replaced with their exponential upper bounds, in calculating the numerical results of this section.

5.3.1 Non-Asymptotic Case

For this case, $\mathbf{E}[B^2]$ is evaluated according to (5.11). We consider a super error state diagram constructed by the combination of the component code error digram with itself [1]. The super state diagram is constructed as follows:

- The state $[S, S']$ corresponds to states S and S' of the RSC code;
- The transition labels from state $[S_1, S_1']$ to $[S_2, S_2']$ are in the form of $W^i Z^j W'^{i'} Z'^{j'} R^r L$, where i and j correspond to the systematic and parity check bits of the transition from state S_1 to S_2 in the component code and are evaluated to 0 or 1 accordingly, i' and j' correspond to the transition from S_1' to S_2' and are evaluated in the same way, r is equal to 1 if both i and i' are equal to 1, and is equal to 0 otherwise, and L represents the length of the codeword and obviously has power 1 in all transition labels;
- The state diagram starts from the state $[0, 0_s]$ and ends at the state $[0, 0_f]$, where the subscripts are added to distinguish between the start-

[1]If different component codes are employed, we need to construct two super state diagrams, each corresponding to one component code.

ing and the finishing states. There is no transition to the state $[0, 0_s]$ and no transition from the state $[0, 0_f]$.

Fig. 5.3 shows the trellis diagram and part of the super state diagram corresponding to a (5,7) RSC code, where the brackets in denoting the states are omitted.

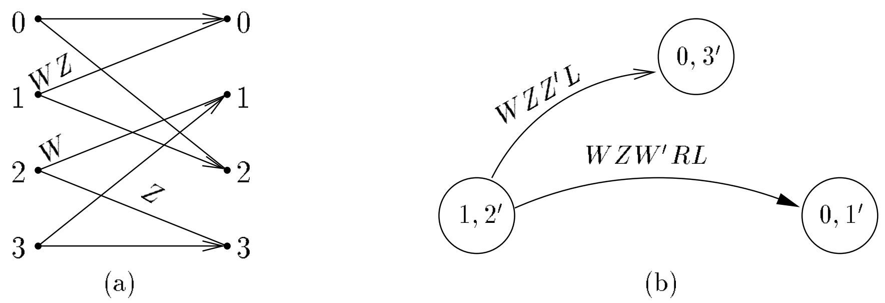


Figure 5.3: (a) Trellis section of a (5,7) RSC code, (b) Part of the super error state diagram.

The transfer function of this state diagram enumerates the error events of the super trellis corresponding to the state diagram. Each error event is characterized by the number of 1's in the systematic and parity check bits of each codeword, the number of their overlapping 1's in the systematic part, and the length of the event from the point where at least one of the codes diverges from the all zero path to the point where both codes re-merge to the all zero path. The transfer function is then used to find $q_r(i, j, i', j')$ for different i, j, i', j', and r, with an approach analogous to that of finding the conditional weight enumerating function explained in [14].

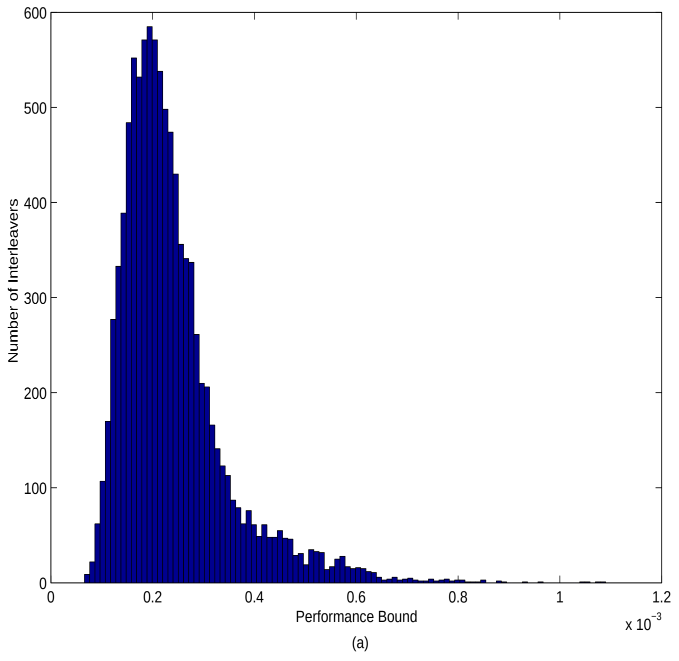
Table 5.1 shows the coefficient of variation of the bound for rate 1/3 turbo-codes with identical (5,7) RSC component codes and interleaver lengths $N = 100$ and $N = 1000$.

Table 5.1: Non-asymptotic results corresponding to turbo-codes with (5,7) RSC codes.

E_b/N_0	$N = 100$		$N = 1000$	
(dB)	$\mathbf{cv}[B]$	$\log_{10} \mathbf{E}[B]$	$\mathbf{cv}[B]$	$\log_{10} \mathbf{E}[B]$
2	0.45	-2.60	0.44	-4.23
3	0.61	-3.65	0.55	-4.96
4	1.02	-4.50	0.78	-5.17
5	1.36	-5.43	1.10	-6.28

In order to obtain an estimation of the distribution of the performance bound with respect to different interleavers, the union upper bound has been calculated for the same codes over a number of randomly selected interleavers. Figs. 5.4 and 5.5 show the corresponding results.

In these histograms, the x-axis represents the performance bound and the y-axis represents the number of interleavers which result in that performance. In calculating these results only input words of weights up to 6 (for $N = 100$) and 4 (for $N = 1000$), resulting in codewords with total weights up to 30, are considered. These limitations are the cause for the simulation results to differ from the theoretical results of Table 5.1. As can be seen from Figs. 5.4 and 5.5 and Table 5.1, the coefficient of variation increases with SNR. This is



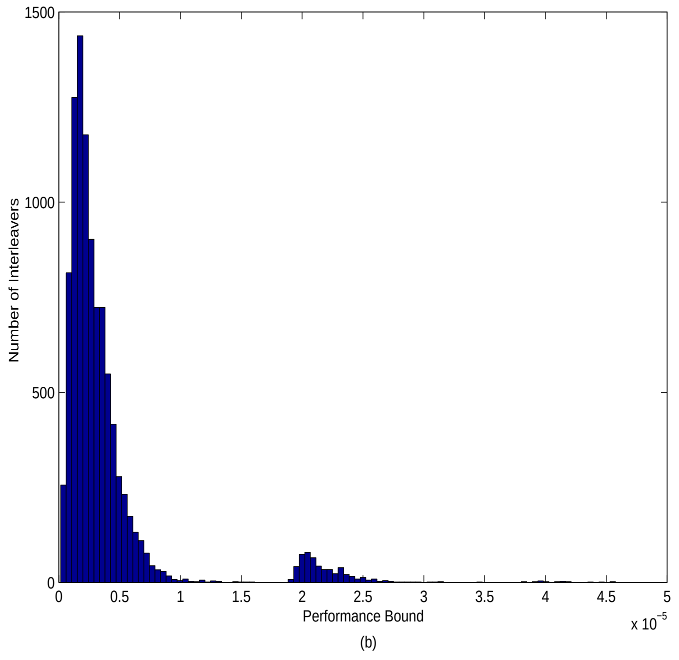
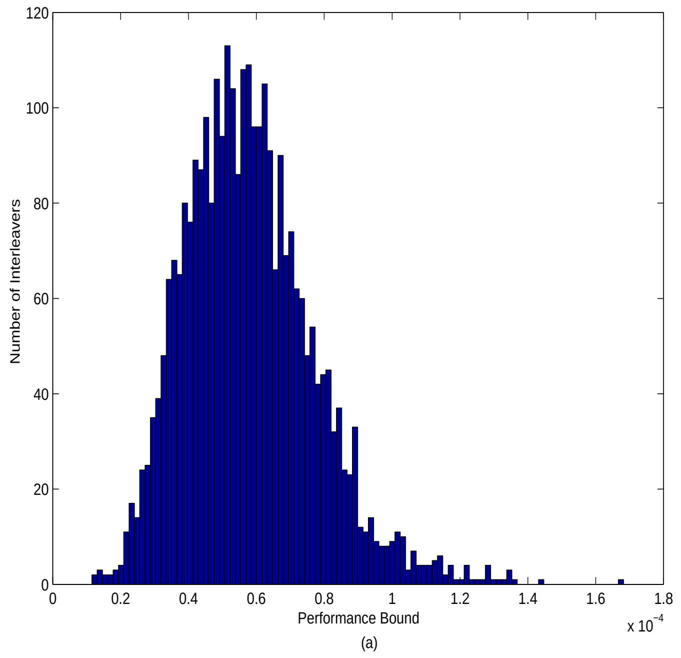


Figure 5.4: Distribution of the performance bound with respect to interleavers of length $N = 100$ for (5,7) turbo-code, (a) SNR=3 dB, $\mathbf{cv}[B]= 0.45$; (b) SNR=5 dB, $\mathbf{cv}[B]= 1.27$



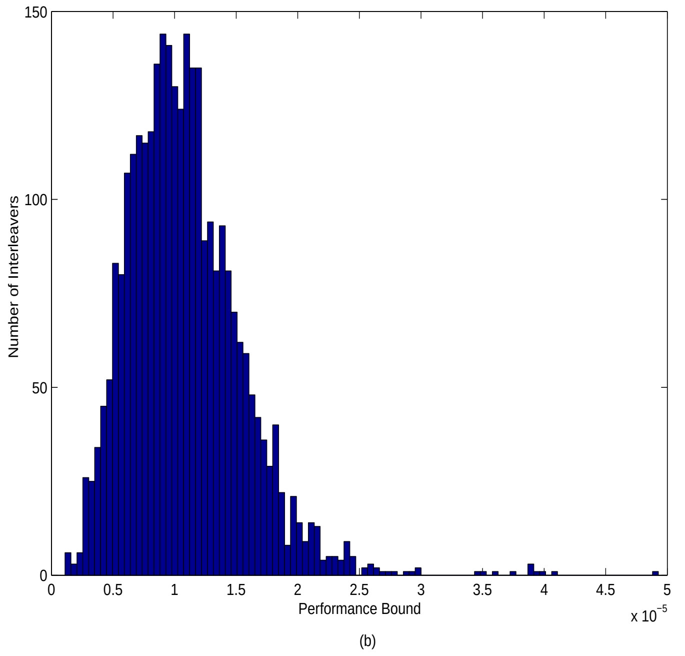
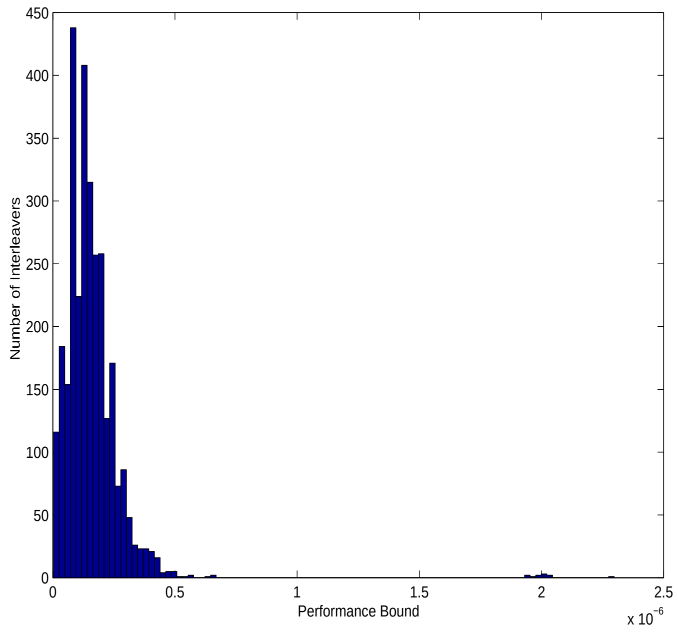


Figure 5.5: Distribution of the performance bound with respect to interleavers of length $N = 1000$ for (5,7) turbo-code. (a) SNR=2 (dB), $\mathbf{cv}[B]$=0.33, (b) SNR=3 (dB), $\mathbf{cv}[B]$=0.43, (Cont.)



(c)

Figure 5.5: (Cont.) Distribution of the performance bound with respect to interleavers of length $N = 1000$ for (5,7) turbo-code, (c) SNR=5 dB, $\mathbf{cv}[B]= 0.90$.

due to the very low percentage of the interleavers which result in higher BER bounds. The majority of the interleavers, however, have performances close to each other. For example, for $N = 100$ and SNR=5 dB, less than 7% of the randomly chosen interleavers result in error performance bounds higher than 10^{-5}; and for $N = 1000$ and SNR=5 dB, only 0.37% of the interleavers result in bounds higher than 0.7×10^{-6}.

5.3.2 Asymptotic Case

In this case, only codeword pairs which satisfy the conditions stated in Theorem 2 are enumerated. For this reason, in the super state diagram only those paths corresponding to weight 2 in the systematic part of the codewords are taken into account and the error events are either completely overlapping ($r = 2$) or have no overlapping bits. Table 5.2 shows the results of the asymptotic analysis.

Table 5.2: Asymptotic results corresponding to turbo-codes with (5,7) and (7,5) component codes.

E_b/N_0 (dB)	$\mathbf{cv}[B]$	
	(7,5)	(5,7)
2	0.11	0.33
3	0.21	0.41
4	0.27	0.48
5	0.34	0.55

The asymptotic results for the (5,7) turbo-code follow the trend of Table 5.1; the coefficient of variation increases with SNR but remains lower than the corresponding values for finite interleaver lengths.

In order to compare turbo-codes with primitive and those with non-primitive feedback polynomials, the asymptotic results corresponding to the (7,5) turbo-code are shown in Table 5.2 as well. As can be seen from the table, the performance bound of the non-primitive feedback polynomial turbo-code has a smaller coefficient of variation.

5.4 Contribution of Different Error Patterns in Turbo-Code Performance

Knowledge of the variance of the turbo-code performance bound with respect to the interleaver gives an estimation of the degree by which the performance can vary according to the choice of the interleaver. On the other hand, when designing or searching for an interleaver, it is not practically possible to optimize the interleaver by considering more than a few number of codewords of different weights. For example, the complexity of the algorithm proposed in [25] is of $O(N^3)$ when error patterns consisting of a single error event are considered in the cost function. The complexity grows to $O(N^4)$, if only one error pattern consisting of double error events is included, and to $O(N^5)$, for triple error patterns and so on. Thus it is important to identify the most significant codewords to be considered in the interleaver design procedure.

The second order mean of the bound can be written as follows:

$$\begin{aligned}\mathbf{E}[B^2] &= \sum_{i,j_1,j_2} \frac{i^2}{N^2}\mathbf{E}[X^2]Q^2(i,j_1,j_2) \\ &+ \sum_{i,j_1,j_2} \sum_{(i',j_1',j_2')\neq(i,j_1,j_2)} \\ &\quad \frac{ii'}{N^2}\mathbf{E}[XX']Q(i,j_1,j_2)Q(i',j_1',j_2'). \end{aligned} \tag{5.26}$$

Thus,

$$\begin{aligned}\mathbf{var}(B) &= \mathbf{E}[B^2] - (\mathbf{E}[B])^2 \\ &= \sum_{i,j_1,j_2} \frac{i^2}{N^2}\mathbf{var}[X^2]Q^2(i,j_1,j_2) \\ &+ \sum_{i,j_1,j_2} \sum_{(i',j_1',j_2')\neq(i,j_1,j_2)} \\ &\quad \frac{ii'}{N^2}\mathbf{cov}[XX']Q(i,j_1,j_2)Q(i',j_1',j_2') \end{aligned} \tag{5.27}$$

The idea proposed here is to obtain the contribution of the variances of each codeword of the form $W^iZ_1^{j_1}Z_2^{j_2}$ to the total variance and identify those codewords which have the most significant contribution to the variance. These are the codewords that lead to the changes in the performance bound as a result of changing the interleaver. Furthermore, the correlation coefficient between each pair of codewords can also be evaluated:

$$\mathbf{corr}(X,X') = \frac{\mathbf{cov}(X,X')}{\sqrt{\mathbf{var}(X)\mathbf{var}(X')}}. \tag{5.28}$$

The correlation coefficient provides a measure of the dependency and the nature of this dependency between the two codewords. If $\mathbf{corr}(X,X')$ is close to 1, it can be concluded that, by changing the interleaver, the values of X and X' change in the same direction. As a result even if both X and X' have high contributions in the variance, it suffices to consider only one of them (preferably the one which results in lower computational complexity) in the design procedure. Of course, this group of codewords should be given the appropriate weight to compensate for the eliminated group of codewords. On the other hand, if the correlation coefficient is close to -1, it can be concluded that the values of X and X' change in opposite directions and thus there is a tradeoff in minimizing one or the other. Consequently, both codeword groups should be considered in the design procedure if their contributions to the variance are significant. Finally, when the correlation coefficient between two codewords is close to zero, which indicates lower dependency between the two, again both codewords should be considered for interleaver design purposes.

Here, the same experimental approach of Section 5.3 is used to evaluate the contribution of codewords with different input weights in the mean and variance of the performance bound. Tables 5.3 and 5.4 show the results corresponding to $N = 100$ and $N = 1000$, respectively. These results show the degree by which the choice of the interleaver affects the contribution of codewords with certain input (systematic) weight in the total performance bound. For example, it can be seen from Table 5.3 that codewords with input weights 2 and 3 both contribute significantly in the average performance bound, however, the contribution of codewords with input weight 3 is significantly higher in the variance.

For $N = 1000$ and moderate SNR (Table 5.4, SNR=3 dB), the contribu-

Table 5.3: Contribution of codewords of different input weights in the mean and variance of the performance bound for $N = 100$, SNR=3 dB and SNR= 5 dB.

input weight	contribution to mean	contribution to variance	contribution to mean	contribution to variance
2	0.42	0.13	0.37	0.02
3	0.40	0.72	0.59	0.95
4	0.08	0.02	0.03	0.001
5	0.05	0.00	0.00	0.001
6	0.04	0.00	0.00	0.00

Table 5.4: Contribution of codewords of different input weights in the mean and variance of the performance bound for $N = 1000$, SNR=3 dB and SNR=5 dB.

	SNR=3 dB		SNR=5 dB	
input weight	contribution to mean	contribution to variance	contribution to mean	contribution to variance
2	0.83	0.75	0.87	0.28
3	0.09	0.17	0.12	0.73
4	0.08	0.005	0.0013	0.00

tion of codewords with input weight 2 is higher both in the mean and the variance. However, for SNR=5, the contribution of codewords with input weight 2 drops significantly in the variance, although even slightly increased in the mean. This shows that codewords with a high contribution to the average of the performance bound do not always contribute highly in the variation of the performance. In all cases, input words of weights 4 and higher have a very insignificant contribution in both the mean and the variance.

The distributions of the performance bound components corresponding to codewords of input weights 2 and 3, are shown in Figs. 5.6 for $N = 100$ and SNR=5 dB and Fig. 5.7, for $N = 1000$ and SNR=2 dB. The distributions of the total bounds corresponding to these figures are shown in Figs. 5.4 (b) and 5.5 (a), respectively. As can be seen from Fig. 5.6, the distribution of the component of the bound corresponding to codewords of input weight 2 is quite concentrated around the mean and does not have a large standard deviation. The distribution of the component of the bound corresponding to codewords of input weight 3 mainly follows the shape of the distribution of the total bound (refer to Fig. 5.4 (b)). On the other hand, it can be seen from Fig. 5.7 that for $N = 1000$ and SNR=2 dB, the component of the bound that is more spread out and resembles the shape of the distribution of the total bound (refer to Fig. 5.5 (a)) corresponds to codewords with input weight 2. These results are also expected from Tables 5.3 and 5.4.

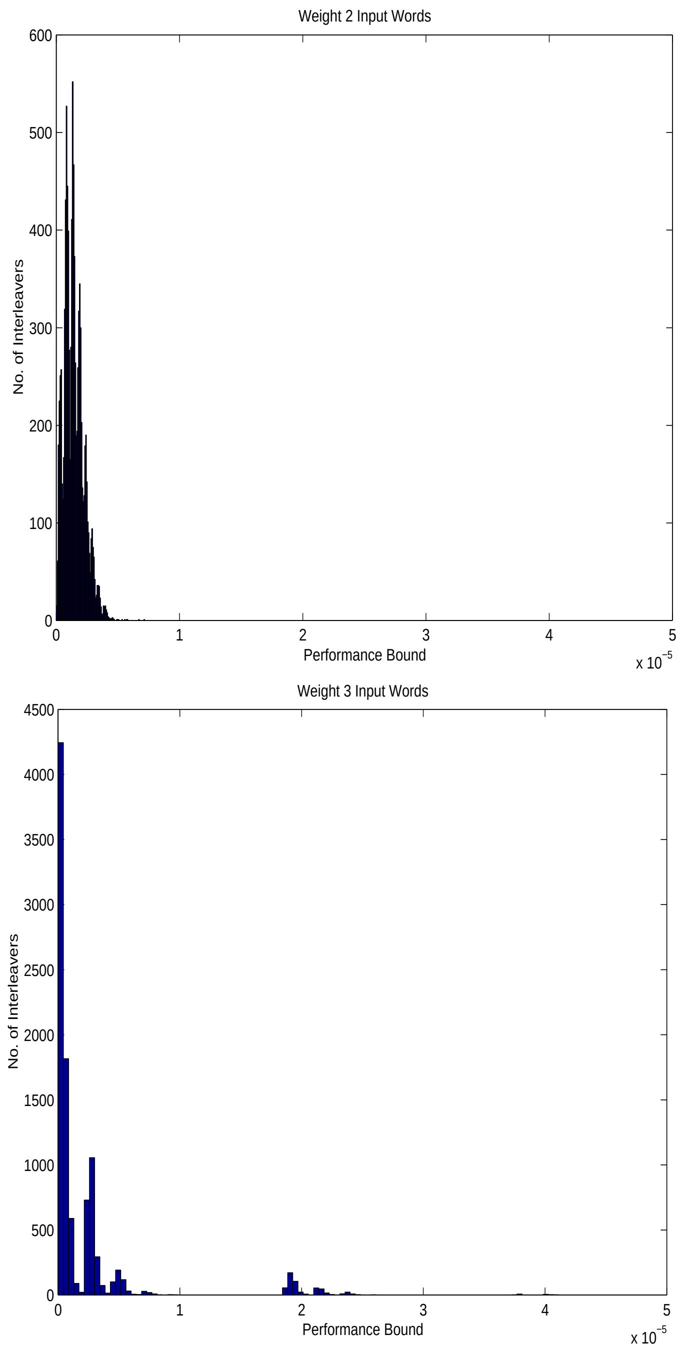
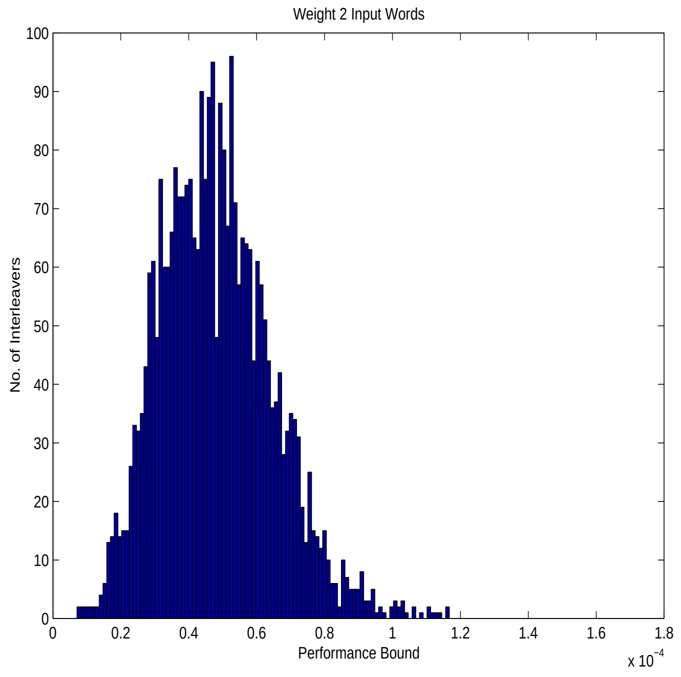


Figure 5.6: Distribution of the (a) weight 2 component and (b) weight 3 component of the performance bound, with respect to random interleavers of length $N = 100$ for (5,7) turbo-code with SNR= 5 dB.



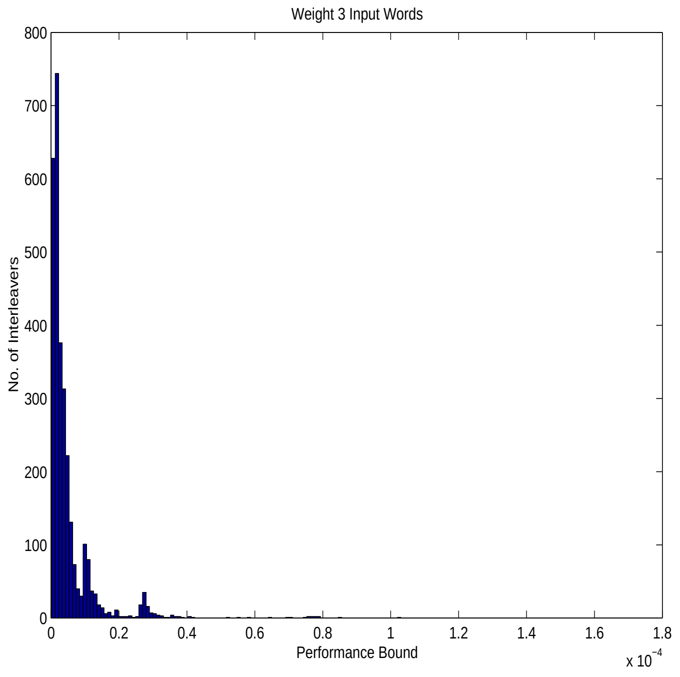


Figure 5.7: Distribution of the (a) weight 2 component and (b) weight 3 component, of the performance bound, with respect to random interleavers of length $N = 1000$ for (5,7) turbo-code with SNR=2 dB.

5.5 Conclusion

In this chapter, the variance of the turbo-code performance bound over all possible interleavers is evaluated. It is shown that the coefficient of variation of the bound is asymptotically constant with the interleaver length. Furthermore, this ratio is relatively small for lower SNR values and increases as the SNR increases. Study of the analytical results and the distribution of the performance bound over a sample of randomly chosen interleavers shows that: (a) as the interleaver length increases, $\mathbf{cv}[B]$ decreases; (b) as SNR increases, the distributions get more concentrated around the average performance bound and only a small percentage of interleavers result in high BERs, which cause $\mathbf{cv}[B]$ to increase. These results support the statement made in [63], where for a turbo-code of length 65536 it is stated that *most* pseudo-random interleavers result in the same multiplicity of the free distance codewords. In addition, it can be observed that the performance of those interleavers which are not close to the performance of the majority of the random interleavers, in fact, deviate quite significantly from the average bound.

Asymptotic results for large interleaver lengths are also evaluated. The results agree with the non-asymptotic results in terms of the trend of the coefficient of variation when SNR and interleaver length change. The asymptotic results also show that turbo-codes with non-primitive feedback polynomials have smaller standard deviations. This may suggest that the choice of the interleaver has a stronger effect on the performance of turbo-codes with primitive than that of those with non-primitive feedback component codes.

Results of this work can also be extended to evaluate the contribution of different error patterns in the variance of the bound, and thus to provide guidelines in identifying the most significant codewords in the design of turbo-code interleaver. The idea is based on considering the codewords which contribute highly in the variance of the performance. The correlation coefficient between different error patterns can also be evaluated among these codewords. This can be used to identify the codewords that should be considered simultaneously or those that can be considered independently in the interleaver design procedure as stated in Section 5.4.

Interleaver design algorithms which consider optimizing the interleaver based on a selection of error patterns (e.g. [25]) choose patterns that contribute highly in the mean of the performance bound. However, numerical results obtained here, show that codewords that contribute highly in the variance of the performance bound are not necessarily the same as codewords with high contribution in the mean. This suggests that better interleavers can be designed by making that selection based on the contribution of error patterns in the variance of the bound. Moreover, their contribution levels change according to the operating SNR, again suggesting that an interleaver which is good at low SNR might not be good at high SNR.

Chapter 6

Conclusion

This thesis aims at providing results and insight towards the application of turbo-codes in digital communication systems, mainly in three subjects: coding combined with modulation; performance improvement by employing UPA; and sensitivity of the code performance towards the choice of the interleaver.

Turbo-codes can be combined with bandwidth efficient modulation, to create a coding scheme applicable for communication over bandwidth limited channels. Chapter 3 studies these applications by focusing on the puncturing and signal mapping methods and the presence of noise correlation in these systems.

When mapping the coded bits to the signals of an M-ary constellation, the labeling method employed can affect the performance of the coding scheme. This is due to the fact that, in general, the bits mapped to the same signal are protected unequally from the channel noise, depending on their position in the labeling bits. Here we have studied examples of turbo-code combined with 8-PSK and 16-QAM modulations and Gray mapping, to show how the labeling and puncturing methods can affect the performance of the coding scheme. The labeling methods are optimized for both systems and a new puncturing method is proposed for the system employing 16-QAM modulation. Improvements of more than 0.5 dB in SNR are achieved over the best performances so far indicated in literature.

For any multi-dimensional modulation scheme, the labeling bits of the same signal are affected by correlated noise. In [37], this correlation is eliminated by interleaving the encoded bits prior to signal mapping. However, as shown in Chapter 3 the impact of this correlation is different according to the modulation scheme and may even be unnecessary. It is shown that, for the coding schemes employing 8-PSK or 16-QAM modulation, extra interleaving for eliminating the noise correlation results in no or negligible performance improvement for BERs of practical interest. This result is important especially when long delays are not tolerable.

Theoretical analysis of combined turbo-code and modulation is suggested for future research in order to provide further insight towards the design of more powerful coding schemes of this type. The presence of the interleaver in the structure of the code imposes a challenging problem in this regard. Even the assumption of uniform interleaving does not solve this problem, unlike when BPSK signaling is used. When a group of encoded bits are mapped to a signal, the performance of the coding scheme depends not only on the weight distribution of the turbo-code, but also on the relative positions of the output bits. The only theoretical approach in this subject so far considers extra interleavers in the form of uniform interleavers, which eliminate the correlation between the coded bits mapped to the same signal [33]. This, in fact, reduces the problem to a problem very similar to performance evaluation for turbo-codes with BPSK signaling. However, as shown in Chapter 3, extra interleaving of the encoded bits is not always justified.

In Chapter 4, a coding scheme based on providing the turbo-encoder output bits with unequal noise margins is proposed. It is shown that the contributions of the systematic and parity check bits in the distance properties of turbo-codes are not the same. As interleaver length gets larger, the contribution of the parity check bits in the distance of the low weight codewords increases. Thus, for large interleaver lengths, higher protection of these bits improves the code performance. Theoretical analysis and simulation results show improvements of about 0.5 in the $\log_{10}(\text{BER})$, equivalent to 0.5 dB in the SNR, when proper level of UPA is used compared to the conventional EPA case. It is also shown how the proposed coding scheme can be applied to CDMA systems by unequally repeated transmission of the coded bits. The results obtained here can also be applied when multi-level modulation is used in conjunction with turbo-coding and UPA is automatically imposed on the coded bits. This, however, is true only when noise correlation has been fully eliminated by means of extra interleaving as mentioned in Section 4.4.2.

In Chapter 5, the variance of the turbo-code performance bound over all possible interleavers of the same length is evaluated. This study is a first step towards answering the question of "the sensitivity of the turbo-code performance towards the choice of the interleaver", brought up in [14].

The variance of the bound is evaluated for both asymptotic (large N) and non-asymptotic cases. It is proven that the ratio of the standard deviation over the mean is asymptotically constant (for large N), and it is shown that this ratio decreases with N in approaching the constant asymptotic value. Theoretical results also show that this ratio increases with SNR.

In order to provide an estimate of the shape of the performance distribution, the performance bound is computationally evaluated for a large number of interleavers. These results agree with the theoretical results. These distributions also show that the increase in the ratio of the standard deviation over the mean for higher SNRs is due to the very low percentage of the interleavers which deviate quite significantly from the mean. The majority of the randomly chosen interleavers, however, result in performances very close to the average performance.

Due to high computational complexity, the distributions of the performance bounds developed here have the following limitations: the codewords taken into account in developing the bounds are limited in their input and overall weight and the number of interleavers for which the performance bound is evaluated is obviously limited. To obtain a better approximation of this distribution, theoretical evaluation of higher order moments of the performance bound with respect to interleavers can be considered as an extension of this work.

The approach in evaluating the variance is extended to evaluate the contribution of input words of different weights in the variance of the turbo-code performance. General guidelines are provided for identifying the most significant codewords in the design of turbo-code interleaver. Numerical results show that these codewords are not necessarily the same as codewords with high contribution on the mean of the performance bound. For example, for $N = 1000$ and SNR=5 dB, the contribution of codewords with input weight 2 is higher than those with input weight 3 in the mean of the performance bound, but drops significantly below that of code words with input weight 3 in the variance.

The above research and the obtained results provide the grounds for future research in developing interleaver design algorithms which focus on the most significant error patterns in optimizing the interleaver. For example, in [25] experimental results are used to identify error patterns to be considered in the cost function of the interleaver design algorithm. The selection is made based on the contribution of the error patterns in the BER but does not consider how effective the choice of the interleaver is on "changing" the contribution of these error patterns. The results shown in this thesis can be applied to the above algorithm in search for the most effective error patterns in optimizing the structure of the interleaver.

Appendix A

Turbo-Decoding Algorithm

A.1 Modified BCJR Algorithm [16]

The BCJR algorithm considers the general problem of estimating the a-posteriori probabilities of the states and transitions of a Markov source, observed through a discrete memoryless noisy channel. This algorithm is optimum in the sense that it minimizes the "bit" error probability, as opposed to the Viterbi algorithm which minimizes the probability of the "word" error, for convolutional codes. In the following, we describe the modified version of this algorithm which is used for the soft decoding of RSC-codes.

Consider an RSC-code with constraint length K, i.e., $K-1$ memory units. At time k, the encoder state is denoted by S_k, where $S_k \in \{0, 1, \ldots, 2^{K-1}-1\}$. Suppose that the input sequence $\mathbf{d}$ consists of N independent bits, d_k, $k = 1, \ldots, N$. Also, suppose that the initial state S_0 and the final state S_N

are both equal to the all zero state, i.e.,

$$S_0 = S_N = \mathbf{0} \tag{A.1}$$

Let us consider the decoder corresponding to the first RSC-code. The received sequence for this decoder consists of the systematic and the first parity check bit. This is denoted by $R_1^N = \{R_1, \ldots, R_N\}$, where $R_k = (y_k^s, y_k^{1p})$ is given in Eqs. (2.9). Defining the joint probability,

$$\mu_k^i(m) = Pr(d_k = i, S_k = m | R_1^N), \tag{A.2}$$

the APP of the data bit d_k is then equal to,

$$Pr(d_k = i | R_1^N) = \sum_m \mu_k^i(m) \qquad i = 0, 1. \tag{A.3}$$

From Eqs. (2.10) and (A.3), the LLR associated with the k'th bit can be written as,

$$\Lambda(d_k) = \log \frac{\sum_m \mu_k^1(m)}{\sum_m \mu_k^0(m)}. \tag{A.4}$$

Using the Bayes rule, the joint probability $\mu_k^i(m)$ can be rewritten as,

$$\mu_k^i(m) = \frac{Pr(d_k = i, S_k = m, R_1^k, R_{k+1}^N)}{Pr(R_1^k, R_{k+1}^N)}, \tag{A.5}$$

thus,

$$\mu_k^i(m) = \frac{Pr(d_k = i, S_k = m, R_1^k)}{Pr(R_1^k)} \times \frac{Pr(R_{k+1}^N | d_k = i, S_k = m, R_1^k)}{Pr(R_{k+1}^N | R_1^k)}. \tag{A.6}$$

To facilitate the problem formulation, the functions $\alpha_k^i(m)$, $\beta_k(m)$, and $\gamma_i(R_k, m', m)$ are defined as,

$$\alpha_k^i(m) = \frac{Pr(d_k = i, S_k = m, R_1^k)}{Pr(R_1^k)} = Pr(d_k = i, S_k = m | R_1^k), \tag{A.7}$$

$$\beta_k(m) = \frac{Pr(R_{k+1}^N | S_k = m)}{Pr(R_{k+1}^N | R_1^k)}, \tag{A.8}$$

and,

$$\gamma_i(R_k, m', m) = Pr(d_k = i, R_k, S_k = m | S_{k-1} = m'). \tag{A.9}$$

Taking into account that if state S_k is known, the events after time k are not influenced by the observation R_1^k and the bit d_k, it can be readily seen that,

$$\mu_k^i(m) = \alpha_k^i(m)\beta_k(m). \tag{A.10}$$

The probabilities $\alpha_k^i(m)$ and $\beta_k(m)$ can be recursively calculated from the probability $\gamma_i(R_k, m', m)$ in the following way,

$$\alpha_k^i(m) = \frac{\sum_{m'} \sum_{j=0}^{1} \gamma_i(R_k, m', m)\alpha_{k-1}^j(m')}{\sum_m \sum_{m'} \sum_{i=0}^{1} \sum_{j=0}^{1} \gamma_i(R_k, m', m)\alpha_{k-1}^j(m')}, \tag{A.11}$$

and

$$\beta_k(m) = \frac{\sum_{m'} \sum_{i=0}^{1} \gamma_i(R_{k+1}, m, m')\beta_{k+1}(m')}{\sum_m \sum_m' \sum_{i=0}^{1} \sum_{j=0}^{1} \gamma_i(R_{k+1}, m', m)\alpha_k^j(m')}. \tag{A.12}$$

The probability $\gamma_i(R_k, m', m)$ can be determined from the transition probabilities of the Gaussian memoryless channel and the transition probabilities of the encoder trellis. From Eq. (A.9), we obtain,

$$\begin{aligned} &\gamma_i(R_k, m', m) = p(R_k | d_k = i, S_k = m, S_{k-1} = m') \times \\ &Pr(d_k = i | S_k = m, S_{k-1} = m') \times Pr(S_k = m | S_{k-1} = m'), \end{aligned} \tag{A.13}$$

where p is a probability density function. Given $(d_k = i, S_k = m, S_{k-1} = m')$, y_k^s and y_k^{1p} are two uncorrelated Gaussian random variables. Thus, we obtain,

$$\begin{aligned} &Pr(R_k | d_k = i, S_k = m, S_{k-1} = m') = \\ &Pr(y_k^s | d_k = i, S_k = m, S_{k-1} = m')\; Pr(y_k^{1p} | d_k = i, S_k = m, S_{k-1} = m'). \end{aligned} \tag{A.14}$$

Since the convolutional encoder is a deterministic machine, $Pr(d_k = i|S_k = m, S_{k-1} = m')$ is either 0 or 1. In Eq. (A.13), the state transition probabilities, namely, $Pr(S_k = m|S_{k-1} = m')$, are defined by the encoder input statistics. We assume that, the input data bits are equi-probable, i.e., $Pr(d_k = 0) = Pr(d_k = 1) = 1/2$. If no further information is available, the decoder assumes $Pr(S_k = m|S_{k-1} = m') = 1/2$. However, as we will see later, in the procedure of iterative feedback decoding these probabilities are determined from the feedback information provided by the other decoder.

In summary, the modified BCJR algorithm receives the noisy version of the encoded bits, and obtains the LLR for each information bit. The algorithm can be summarized in the following steps:

Step 0: Initialization

$$\begin{aligned} &\alpha_0^i(0) = 1 \quad \alpha_0^i(m) = 0 \quad \forall m \neq 0, \;\; i = 0, 1 \\ &\beta_N(0) = 1 \quad \beta_N(m) = 0 \quad \forall m \neq 0 \end{aligned} \tag{A.15}$$

Step 1: For each observation R_k, the probabilities $\alpha_k^i(m)$ and $\gamma_i(R_k, m', m)$ are computed using Eqs. (A.11) and (A.13), respectively.

step 2: When the sequence R_1^N has been completely received, probabilities $\beta_k(m)$ are computed using Eq. (A.12).

step 3: Finally, the LLR associated with each bit d_k is computed from Eqs. (A.4) and (A.10).

A.2 Turbo-Decoding

Fig. A.1 shows the structure of the turbo-decoder.

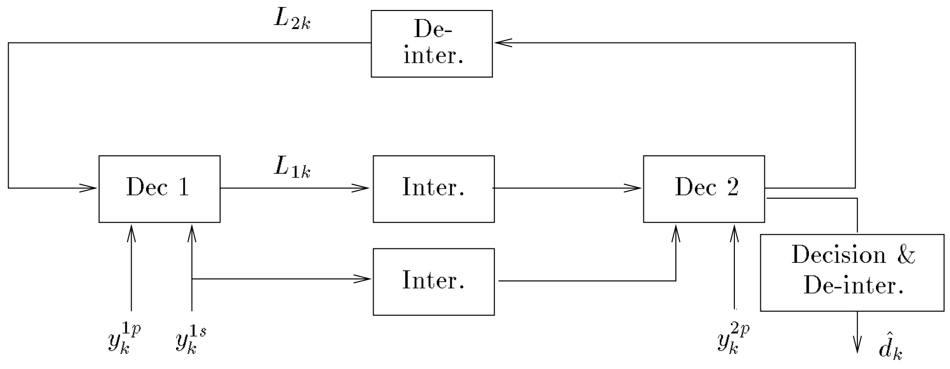


Figure A.1: The structure of the turbo-decoder.

The decoder consists of two component decoders, which operate in a serial mode. Both decoders receive the systematic bit, and each decoder receives its corresponding parity check bit. Each component decoder uses the modified BCJR algorithm to calculate the LLR for each bit. An information called the *extrinsic information* is then obtained from each LLR and is provided to the other decoder. These are denoted by L_{1k} and L_{2k} in Fig. A.1, and their calculation method will be discussed later in this section. This information is regarded as the a-priori probability by the decoders and is used to determine the state transition probabilities as follows [66] (consider the first decoder),

$$Pr(S_k = m|S_{k-1} = m') = \begin{cases} \frac{e^{L_{2k}}}{1+e^{L_{2k}}} & \text{if} \quad m' \xrightarrow{d_k=1} m \\ \frac{1}{1+e^{L_{2k}}} & \text{if} \quad m' \xrightarrow{d_k=0} m \end{cases} . \qquad \text{(A.16)}$$

If state m' is not connected to state m, then $\pi(S_k = m|S_{k-1} = m')$ is assigned arbitrarily. Substituting Eqs. (A.10) and (A.11) in (A.4), we obtain,

$$\Lambda(d_k) = \log \frac{\sum_m \sum_{m'} \gamma_1(R_k, m', m)\alpha_{k-1}(m')\beta_k(m)}{\sum_m \sum_{m'} \gamma_0(R_k, m', m)\alpha_{k-1}(m')\beta_k(m)}, \tag{A.17}$$

where $\alpha_{k-1}(m') = \alpha^0_{k-1}(m') + \alpha^1_{k-1}(m')$. It should be noted that $\alpha_k(m)$ can be calculated by a single recursive equation and there is no need to splitting it up to $\alpha^0_k(m)$ and $\alpha^1_k(m)$.

Since the encoder is systematic, i.e., $x^s_k = d_k$, the transition probability, $Pr(y^s_k|d_k = i, S_k = m, S_{k-1} = m')$ in Eq. (A.14) is independent of the state values S_k and S_{k-1}. Therefore, this probability can be factorized in the numerator and denominator of Eq. (A.17). Using this fact and substituting Eq. (A.16) in (A.17), it can be shown that,

$$\Lambda(d_k) = \log \frac{\sum_m \sum_{m'} \gamma'_1(y^{1p}_k, m', m)\alpha_{k-1}(m')\beta_k(m)}{\sum_m \sum_{m'} \gamma'_0(y^{1p}_k, m', m)\alpha_{k-1}(m')\beta_k(m)} + L_2(d_k) + \frac{2y^s_k}{\sigma^2}, \tag{A.18}$$

where $\gamma'_i(y^{1p}_k, m', m) = p(y^{1p}_k|d_k = i, S_k = m, S_{k-1} = m').Pr(d_k = i|S_k = m, S_{k-1} = m')$. The second component in Eq. (A.18) is the extrinsic information which was provided by the second decoder. The last component in Eq. (A.18) is the LLR of the systematic bit, i.e.,

$$\log \frac{p(y^s_k|d_k = 1)}{p(y^s_k|d_k = 0)} = \frac{2y^s_k}{\sigma^2}, \tag{A.19}$$

where σ^2 is the variance of the white Gaussian noise. The first component in Eq. (A.18) is the extrinsic information of the first decoder, L_{1k}, which will now be regarded as the a-priori information by the second decoder.

At the end of the final iteration, a hard decision is made based on the sign of $\Lambda(d_k)$, i.e.,

$$\begin{cases} \hat{d}_k = 1 & \text{if} \quad \Lambda(d_k) > 0 \\ \hat{d}_k = 0 & \text{if} \quad \Lambda(d_k) \leq 0, \end{cases} \tag{A.20}$$

and the decoding is completed.

For the sake of completeness, in the following we give a brief description of the original turbo-decoding algorithm [16], which is slightly different from the above algorithm.

In this decoding scheme, the state transition probabilities, $\pi(S_k = m|S_{k-1} = m')$'s, are considered to be equal to 1/2 in each iteration. However, the extrinsic information is regarded as one of the observations (independent of the systematic and parity check information), i.e., the k'th observation vector for the first decoder is, $R_k = (y_k^s, y_k^{1p}, L_{2k})$. Here, L_{2k} is treated as a white Gaussian process with variance σ_L^2. Therefore,

$$p(L_{2k}|d_k = i) \;\propto\; \exp\{\frac{1}{\sigma_L^2} L_{2k}(2i-1)\}. \tag{A.21}$$

In this algorithm, the variance of the extrinsic information must be estimated at each iteration by the turbo-decoder. This is the main drawback of this approach with respect to the algorithm explained earlier. Using Eqs. (A.17) and (A.21), we obtain,

$$\Lambda(d_k) = \log \frac{\sum_m \sum_{m'} \gamma_1'(R_k, m', m)\alpha_{k-1}(m')\beta_k(m)}{\sum_m \sum_{m'} \gamma_0'(R_k, m', m)\alpha_{k-1}(m')\beta_k(m)} + \frac{2L_{2k}}{\sigma_L^2} + \frac{2y_k^s}{\sigma^2}. \tag{A.22}$$

In [16], it is shown that in the first iteration, the extrinsic information is poor about the input bit, d_k. Furthermore, the Gaussian hypothesis made for the distribution of the extrinsic information is not satisfied. Nevertheless, as the iteration number increases, the variance of extrinsic information decreases and its distribution merges to a Gaussian distribution with a mean value of $(2d_k - 1)$.

It has been shown that the performance of the earlier mentioned turbo-decoding a algorithm is better than the original one [44]. Further more, it has a simpler structure because it does not require estimating the variance of the extrinsic information.

Appendix B

Alternate Proof of Theorem 2, Chapter 5

In this alternative proof, we elaborate on the contribution of codeword pairs (with different systematic weights and relative positions) to the variance of the bound.

Suppose m_1 of the n_1 error events of the code word $W^i Z^{j_1}$ are tied up with m'_1 of the n'_1 error events of the code word $W^{i'} Z'^{j'_1}$, and the tied up events result in c_1 clusters. Also, suppose that the above tied up error events contain $(r + \hat{r}_1)$ and $(r + \hat{r}'_1)$ 1's, respectively. Thus, the remaining $(n_1 - m_1)$ and $(n'_1 - m'_1)$ loose events contain $(i - r - \hat{r}_1)$ and $(i' - r - \hat{r}'_1)$ 1's, respectively.

Since for recursive convolutional codes each error event has a Hamming weight of at least two in the systematic part, we have

$$(n_1 - m_1) \quad \leq \quad \lfloor \frac{i - r - \hat{r}_1}{2} \rfloor$$

$$(n_1' - m_1') \leq \lfloor \frac{i' - r - \hat{r}_1'}{2} \rfloor \tag{B.1}$$

Also,

$$\begin{aligned} (m_1) &\leq \lfloor \frac{r + \hat{r}_1}{2} \rfloor \\ (m_1') &\leq \lfloor \frac{r + \hat{r}_1'}{2} \rfloor \end{aligned} \tag{B.2}$$

and

$$c_1 \leq \min(m_1, m_1') \leq \min(\lfloor \frac{r + \hat{r}_1}{2} \rfloor, \lfloor \frac{r + \hat{r}_1'}{2} \rfloor). \tag{B.3}$$

where $\lfloor x \rfloor$ is the largest integer less than or equal to x. Define

$$\hat{M}_1 = (\lfloor \frac{i - r - \hat{r}_1}{2} \rfloor + \lfloor \frac{i' - r - \hat{r}_1'}{2} \rfloor + \min(\lfloor \frac{r + \hat{r}_1}{2} \rfloor, \lfloor \frac{r + \hat{r}_1'}{2} \rfloor))_{\max} \tag{B.4}$$

which is an upper bound on M_1. In the following, we consider the eight different cases corresponding to i, i' and r being even or odd integers. $\hat{M}_1$ is evaluated for each case, and finally it is shown that there exist conditions under which M_1 takes the maximum value found for $\hat{M}_1$.

$$\text{Case 1:} \begin{cases} i & \text{even} \\ i' & \text{even} \\ r & \text{even} \end{cases}$$

From (B.4), we have

$$\begin{aligned} \hat{M}_1 &= \left(\frac{i - r}{2} - \lfloor \frac{\hat{r}_1 + 1}{2} \rfloor + \frac{i' - r}{2} - \lfloor \frac{\hat{r}_1' + 1}{2} \rfloor + \min(\frac{r}{2} + \lfloor \frac{\hat{r}_1}{2} \rfloor, \frac{r}{2} + \lfloor \frac{\hat{r}_1'}{2} \rfloor) \right)_{\max} \\ &= \frac{i}{2} + \frac{i'}{2} - \frac{r}{2} + \left(-\lfloor \frac{\hat{r}_1 + 1}{2} \rfloor - \lfloor \frac{\hat{r}_1' + 1}{2} \rfloor + \min(\lfloor \frac{\hat{r}_1}{2} \rfloor, \lfloor \frac{\hat{r}_1'}{2} \rfloor) \right)_{\max} . \end{aligned} \tag{B.5}$$

Assuming that $\min(\lfloor \frac{\hat{r}_1}{2} \rfloor, \lfloor \frac{\hat{r}_1'}{2} \rfloor) = \lfloor \frac{\hat{r}_1}{2} \rfloor$, we have

$$M_1 = \frac{i+i'-r}{2} + \left(\lfloor\frac{\hat{r}_1}{2}\rfloor - \lfloor\frac{\hat{r}_1+1}{2}\rfloor - \lfloor\frac{\hat{r}'_1+1}{2}\rfloor\right)_{\max}. \tag{B.6}$$

It can be easily seen that,

$$\lfloor\frac{\hat{r}_1}{2}\rfloor - \lfloor\frac{\hat{r}_1+1}{2}\rfloor = \begin{cases} 0, & \text{iff } \hat{r}_1 \text{ even} \\ -1, & \text{iff } \hat{r}_1 \text{ odd} \end{cases} \tag{B.7}$$

and

$$-\lfloor\frac{\hat{r}'_1+1}{2}\rfloor \leq 0, \quad \text{``="} \text{ iff } \hat{r}'_1 = 0. \tag{B.8}$$

Thus,

$$\hat{M}_1 = \frac{i+i'-r}{2} \tag{B.9}$$

if and only if $(\hat{r}_1, \hat{r}'_1) = (0, 0)$.

It can be easily verified that assuming $\min(\lfloor\frac{\hat{r}_1}{2}\rfloor, \lfloor\frac{\hat{r}'_1}{2}\rfloor) = \lfloor\frac{\hat{r}'_1}{2}\rfloor$ results in the same maximum value for $\hat{M}_1$ under the same condition.

$$\text{Case 2: } \begin{cases} i & \text{even} \\ i' & \text{even} \\ r & \text{odd} \end{cases}$$

Assuming that $\min(\lfloor\frac{\hat{r}_1+1}{2}\rfloor, \lfloor\frac{\hat{r}'_1+1}{2}\rfloor) = \lfloor\frac{\hat{r}_1+1}{2}\rfloor$, we have

$$\hat{M}_1 = \frac{i+i'-r}{2} - \frac{3}{2} + \left(-\lfloor\frac{\hat{r}_1}{2}\rfloor - \lfloor\frac{\hat{r}'_1}{2}\rfloor + \lfloor\frac{\hat{r}_1+1}{2}\rfloor\right)_{\max}. \tag{B.10}$$

Also,

$$\lfloor\frac{\hat{r}_1+1}{2}\rfloor - \lfloor\frac{\hat{r}_1}{2}\rfloor = \begin{cases} 0, & \text{iff } \hat{r}_1 \text{ even} \\ 1, & \text{iff } \hat{r}_1 \text{ odd} \end{cases} \tag{B.11}$$

and

$$-\lfloor\frac{\hat{r}'_1}{2}\rfloor \leq 0, \quad \text{``="} \text{ iff } \hat{r}'_1 = 0, 1. \tag{B.12}$$

It can be easily shown that the maximum value in this case is

$$\hat{M}_1 = \frac{i+i'-r}{2} - \frac{1}{2} \tag{B.13}$$

and that it is achieved if and only if $(\hat{r}_1, \hat{r}'_1) = (1,1)$.

Again, assuming $\min(\lfloor\frac{\hat{r}_1+1}{2}\rfloor, \lfloor\frac{\hat{r}'_1+1}{2}\rfloor) = \lfloor\frac{\hat{r}'_1+1}{2}\rfloor$ does not affect the result.

$$\text{Case 3:} \begin{cases} i & \text{even} \\ i' & \text{odd} \\ r & \text{even} \end{cases}$$

In this case,

$$\hat{M}_1 = \frac{i+i'-r}{2} - \frac{1}{2} + \left(-\lfloor\frac{\hat{r}_1+1}{2}\rfloor - \lfloor\frac{\hat{r}'_1}{2}\rfloor + \min(\lfloor\frac{\hat{r}_1}{2}\rfloor, \lfloor\frac{\hat{r}'_1}{2}\rfloor)\right)_{\max} . \tag{B.14}$$

With a similar approach to the previous cases, it can be shown that

$$\hat{M}_1 = \frac{i+i'-r}{2} - \frac{1}{2} \tag{B.15}$$

if and only if $(\hat{r}_1, \hat{r}'_1) = (0,0)$, $(0,1)$.

$$\text{Case 4:} \begin{cases} i & \text{even} \\ i' & \text{odd} \\ r & \text{odd} \end{cases}$$

In this case,

$$\hat{M}_1 = \frac{i+i'-r}{2} - 1 + \left(-\lfloor\frac{\hat{r}_1}{2}\rfloor - \lfloor\frac{\hat{r}'_1+1}{2}\rfloor + \min(\lfloor\frac{\hat{r}_1+1}{2}\rfloor, \lfloor\frac{\hat{r}'_1+1}{2}\rfloor)\right)_{\max} \tag{B.16}$$

and it can be shown that

$$\hat{M}_1 = \frac{i+i'-r}{2} - 1 \tag{B.17}$$

if and only if $(\hat{r}_1, \hat{r}'_1) = (0,0)$, $(1,0)$, $(1,1)$, $(1,2)$.

Case 5: $\begin{cases} i & \text{odd} \\ i' & \text{even} \\ r & \text{even} \end{cases}$

This case is similar to Case 3 with only the conditions on i and i' switched. Thus,

$$\hat{M}_1 = \frac{i+i'-r}{2} - \frac{1}{2} \tag{B.18}$$

if and only if $(\hat{r}_1, \hat{r}'_1) = (0,0)$, $(1,0)$.

Case 6: $\begin{cases} i & \text{odd} \\ i' & \text{even} \\ r & \text{odd} \end{cases}$

This case is similar to Case 4 with only the conditions on i and i' switched. Thus,

$$\hat{M}_1 = \frac{i+i'-r}{2} - 1 \tag{B.19}$$

if and only if $(\hat{r}_1, \hat{r}'_1) = (0,0)$, $(0,1)$, $(1,1)$, $(2,1)$.

Case 7: $\begin{cases} i & \text{odd} \\ i' & \text{odd} \\ r & \text{even} \end{cases}$

In this case,

$$\hat{M}_1 = \frac{i+i'-r}{2} - 1 + \left(-\lfloor\frac{\hat{r}_1}{2}\rfloor - \lfloor\frac{\hat{r}'_1}{2}\rfloor + \min(\lfloor\frac{\hat{r}_1}{2}\rfloor, \lfloor\frac{\hat{r}'_1}{2}\rfloor)\right)_{\max} \tag{B.20}$$

and it can be shown that,

$$\hat{M}_1 = \frac{i+i'-r}{2} - 1 \tag{B.21}$$

if and only if $(\hat{r}_1, \hat{r}'_1) = (0,0)$, $(0,1)$, $(1,0)$, $(1,1)$.

Case 8: $\begin{cases} i & \text{odd} \\ i' & \text{odd} \\ r & \text{odd} \end{cases}$

We have

$$\hat{M}_1 = \frac{i+i'-r}{2} - \frac{1}{2} + \left(-\lfloor\frac{\hat{r}_1+1}{2}\rfloor - \lfloor\frac{\hat{r}'_1+1}{2}\rfloor + \min(\lfloor\frac{\hat{r}_1+1}{2}\rfloor, \lfloor\frac{\hat{r}'_1+1}{2}\rfloor)\right)_{\max} \tag{B.22}$$

and it can be shown that,

$$\hat{M}_1 = \frac{i+i'-r}{2} - \frac{1}{2} \tag{B.23}$$

if and only if $(\hat{r}_1, \hat{r}'_1) = (0,0)$.

As can be seen, Case 1 with the condition $(\hat{r}_1, \hat{r}'_1) = (0,0)$ results in the maximum value for $\hat{M}_1$. In other words, $\hat{M}_1 = \frac{i+i'-r}{2}$, if and only if the input words s_g and s'_h are such that their resulting codewords satisfy the following conditions:

- Each codeword is constructed of error events with input weight 2.
- Each error event is either a loose event or exactly matches and is tied up with an error event of the other codeword.

Since for every recursive convolutional code, there exists an input word of Hamming weight two which results in a finite response, the above conditions are achievable and under these conditions,

$$(M_1)_{\max} = \frac{i+i'-r}{2}. \tag{B.24}$$

Similarly,

$$(M_2)_{\max} = \frac{i + i' - r}{2}. \tag{B.25}$$

As a result,

$$(G_t)_{\max} = 0. \tag{B.26}$$

□

Bibliography

[1] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, “Optimal decoding of linear codes for minimizing symbol error rate,” *IEEE Transactions on Information Theory,* vol. 20, pp. 284-287, March 1974.

[2] S. A. Barbulescu, W. Farrell, P. Gray, and M. Rice, “Bandwidth efficient Turbo coding for high speed mobile satellite communications,” *Proceedings of the International Symposium on Turbo Codes*, pp. 119-126, Brest, France, Sept. 1997.

[3] A. S. Barbulescu and S. S. Pietrobon, “Terminating the trellis of Turbo-codes in the same state,” *Electronics Letters*, vol.31, no.1, pp. 22-23, Jan. 1995.

[4] A. S. Barbulescu and S. S. Pietrobon, “Rate compatible Turbo codes,” *Electronics Letters*, vol. 31, pp. 535-536, Mar. 1995.

[5] A. S. Barbulescu and S. S. Pietrobon, “Interleaver design for three dimensional Turbo codes,” *Proceedings of the 1995 IEEE International Symposium on Information Theory*, ISIT’95, p. 37, Whistler, B.C., Canada, Sept. 1995.

[6] A. S. Barbulescu and S. S. Pietrobon, “Interleaver design for Turbo-codes,” *Electronics Letters*, vol.30, no.25, pp. 2107-2108, Dec. 1994.

[7] W. J. Blackert, E. K. Hall, and S. G. Wilson, “Turbo code termination and interleaver condition,” *Electronics Letters*, vol.31, no.24, pp. 2082-2084, Nov. 1995.

[8] S. Benedetto, E. Biglieri, G. Caire, G. Montorsi, and G. Taricco, “Study on advanced coding techniques applied to the transmission of digital speech and video Signals”, Dipartimento di Elettronica Politecnico di Torino, Feb. 1995.

[9] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “Bandwidth efficient parallel concatenated coding schemes,” *Electronics Letters*, vol. 31, no. 24, pp. 2067-2069, Nov. 1995.

[10] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “Parallel concatenated trellis-coded modulation,” *Proceedings of IEEE International Conference on Communications,* ICC’96, pp. 974-978, Dallas, Texas, June 1996.

[11] S. Benedetto, R. Garello, G. Montorsi, “A search for good convolutional codes to be used in construction of Turbo-codes,” *IEEE Transactions on Communications*, pp. 1101-1105,vol. 46, no. 9, Sept. 1998.

[12] S. Benedetto, G. Montorsi, D. Divsalar and F. Pollara, “Soft-output decoding algorithms in iterative decoding of Turbo codes,” JPL TDA Progress Report 42-124, Feb. 15, 1996.

[13] S. Benedetto and G. Montorsi, "Performance evaluation of parallel concatenated codes," *Proceedings of the 1995 IEEE International Conference on Communications*, ICC'95, pp. 663-667, Seattle, June 1995.

[14] S. Benedetto and G. Montorsi, "Unveiling Turbo codes: some results on parallel concatenated coding schemes," *IEEE Transactions on Information Theory*, vol. 42, no. 2, pp. 409-428, Mar. 1996.

[15] S. Benedetto and G. Montorsi, "Design of parallel concatenated convolutional codes," *IEEE Transactions on Communications*, vol. 44, no. 5, pp. 591-600, May 1996.

[16] C. Berrou, A. Glavieux and P. Thitimajshima "Near Shannon limit error-correcting coding and decoding: Turbo-codes", *Proceedings of IEEE International Conference on Communications,* ICC'93, pp. 1064-1070, Geneva, Switzerland, May 1993.

[17] R. E. Blahut, *Theory and Practice of Error Control Codes.* NewYork, Addison-Wesley Publishing Company Inc., c 1983.

[18] J. Blanz, A. Klein, M. Naßan, A. Steil, "Performance of a cellular hybrid C/TDMA mobile radio system applying joint detection and coherent receiver antenna diversity," *IEEE Journal on selected Areas in Communications*, vol. 12, no. 4, pp.568-579, May 1994.

[19] S. Brink, J. Speidel, R. Yan, "Iterative demapping and decoding for multilevel modulation", it Proceedings of GLOBECOM'98, pp. 579-584, Sydney, Australia, Nov. 1998.

[20] F. Burkert, G. Caire, J. Hagenauer, T. Hindelang, and G. Lechner, "Turbo coding with unequal error protection applied to GSM speech coding," *Proceedings of GLOBECOM'96*, pp. 2044-2049, London, U.K., Nov. 1996.

[21] A. G. Burr, G. P. White, "Combined performance of joint detection with FEC coding for CDMA mobile communication systems," IEE Colloquium on UMTS Terminals and Software Radio, IEE, pp. 111-118, London, UK, 1999.

[22] G. Caire and E. Biglieri, "Parallel concatenated codes with unequal error protection," *IEEE Transactions on Communications*, vol. 46, no. 5, pp. 565-567, May 1998.

[23] N. Chayat, "Turbo codes for incoherent M-ary orthogonal signaling," *Proceedings of the . 19th Convention of Electrical and Electronics Engineers in Israel*, pp. 471-474, Jerusalem, Israel, Nov. 1996.

[24] D. J. Costello, J. Hagenauer, H. Imai, and S. B. Wicker, "Applications of error-control coding," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2531-2560, Oct. 1998.

[25] F. Daneshgaran and M. Mondin, "Design of interleavers for Turbo codes: iterative interleaver growth algorithms of polynomial com-

plexity," *IEEE Transactions on Information Theory,* , vol. 45, no. 6, pp. 1084-1859, Sept. 1999.

[26] D. Divsalar and R. J. McEliece, "Effective free distance of Turbo-codes," *Electronics Letters*, vol. 32, no. 5, pp.445-446, Feb. 1996.

[27] D. Divsalar and F. Pollara, "Turbo codes for PCS applications," *Proceedings of IEEE International Conference on Communications*, ICC'95, pp.54-59, Seattle, Washington, June 1995.

[28] D. Divsalar, S. Dolinar, R. J. McEliece, and F. Pollara, "Transfer function bounds on the performance of Turbo-codes," Jet Propulsion Lab., Pasadena, CA, TDA Progress Report 42-122, Aug. 1995.

[29] D. Divsalar and F. Pollara, " Turbo codes for deep-space communications," Jet Propulsion Lab., Pasadena, CA, TDA Progress Report 42-120, Feb. 1995.

[30] D. Divsalar and F. Pollara, "Multiple Turbo codes for deep-space communications," Jet Propulsion Lab., Pasadena, CA, TDA Progress Report 42-124, May 1995.

[31] T. M. Duman and M. Salehi, "On optimal power allocation for Turbo codes," *Proceedings of the 1997 International Symposium on Information Theory*, ISIT'97, p. 104, Ulm, Germany, June 97.

[32] T. M. Duman and M. Salehi, "New Performance Bounds for Turbo codes," *IEEE Transactions on Communications*, vol. 46, no. 6, pp. 717-724, June 1998.

[33] T. M. Duman and M. Salehi, "Performance bounds for Turbo-coded modulation systems," *IEEE Transactions on Communications*, vol. 47, no. 4, pp. 511-521, Apr. 1999.

[34] P. Elias, "Error-free coding," *IRE Transactions on Information Theory*, vol. 4, pp. 29-37, 1954.

[35] G. D. Forney, *Concatenated Codes.* PhD thesis, MIT, 1966.

[36] G. D. Forney, "Convolutional codes II: Maximum-likelihood decoding," *Information and Control*, vol. 25, no. 3, p.222-266, July 1974.

[37] S. Goff, A. Glavieux and C. Berrou, "Turbo-codes and high spectral efficiency modulation," *Proceedings of IEEE International Conference on Communications*, ICC'94, pp. 645-649, New Orleans, Louisiana, May 1994.

[38] J. Hagenauer, P. Robertson, L. Papke, "Iterative ("Turbo") decoding of systematic convolutional codes with MAP and SOVA algorithms," *Proceedings of the ITG conference "Source and Channel Coding"*, pp. 21-29, Munich, 1994.

[39] C. Heegard, S. B. Wicker, *Turbo coding*, Kluwer Academic Publishers, Boston, c1999.

[40] J. Hokfelt and T. Maseng, "Optimizing the energy of different bit streams of Turbo codes," Proceedings of Turbo Coding Seminar, pp. 59-63, Lund, Sweden, Aug. 1996.

[41] J. Hokfelt, O. Edfors, and T. Maseng, "A turbo code interleaver design criterion based on the performance of iterative decoding," *IEEE Communications Letters*, Vol. 5, No. 2, February 2001.

[42] P. Jung, "Novel low complexity decoder for Turbo-codes," *Electronics Letters*, vol. 31, no.2, pp. 86-87, Jan. 1995.

[43] P. Jung and J. Blanz, "Joint detection with coherent receiver antenna diversity in CDMA mobile radio systems," *IEEE Transactions on Vehicular Technology*, Bd. 44, pp. 76-88, 1995.

[44] P. Jung and M. M. Naßhan, "Comprehensive comparison of Turbo code decoders," *Proceedings of IEEE GLOBECOM 1995*, pp. 624-628, Singapore, Nov. 1995.

[45] P. Jung, M. Naßan, and J. Blanz, "Application of Turbo-codes to a CDMA mobile radio system using joint detection and antenna diversity," *Proceedings of the 44th IEEE Vehicular Technology Conference*, pp. 770-774, VTC'94, Stockholm, 1994.

[46] P. Jung and M. Naßhan, "Performance evaluation of Turbo-codes for short frame transmission systems," *Electronics Letters*, vol.30, no.2, pp. 111-113, Jan. 1994.

[47] P. Jung and M. Naßhan, "Dependence of the error performance of Turbo-codes on the interleaver structure in short frame transmission systems," *Electronics Letters,*, vol.30, no.4, pp. 287-288, Feb. 1994.

[48] A. K. Khandani, "Group structure of Turbo-codes," *Electronics Letters*, vol. 34, no. 2, pp. 168-169, Jan. 1998.

[49] A. K. Khandani, "Design of Turbo-code interleaver using Hungarian method," *Electronics Letters*, vol. 34, no. 1, pp. 63-65, Jan. 1999.

[50] A. K. Khandani, *Dynamic generation of good Turbo-code interleavers,* Technical Report UW-E&CE99-02, Dept. of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada, 1999.

[51] A. Klein, G. K. Kaleh, P. W. Baier, "Equalizers for multi-user detection in code division multiple access mobile radio systems," *Proceedings of IEEE 44th Vehicular Technology Conference*, VTC'94, pp. 762-766, Stockholm, 1994.

[52] F. Ling and D. D. Falconer, "Combined orthogonal/convolutional coding for a digital CDMA system," *Proceedings of IEEE Vehicular Technology Conference*, VTC'92, pp.63-66.

[53] R. J. McEliece, E. R. Rodemich, and J. Cheng, "The Turbo decision algorithm," *Proceedings of the 33rd Allerton Conference on Communication, Control, and Computing*, pp. 366-379, Urbana-Champaign, IL, Oct. 1995.

[54] A. H. S. Mohammadi and A. K. Khandani, "Effect of the labeling of the constellation points in a combined Turbo-code and modulation," *Proceedings of the 1996 Conference on Information Science and Systems*, CISS'96, pp. 67-69, Princeton, NJ, Mar. 1996.

[55] A. H. S. Mohammadi and A. K. Khandani, "Unequal Error Protection on the Turbo-Encoder Output Bits," *Proceedings of the 1997 IEEE International Conference on Communications*, ICC'97, pp. 730-734, Montreal, Quebec, June 1997.

[56] A. H. S. Mohammadi and A. K. Khandani, "Unequal power allocation to the Turbo-encoder output bits with application to CDMA systems," *IEEE Transactions on Communications*, vol. 47, no. 11, pp. 1609-1611, Nov. 1999.

[57] A. H. S. Mohammadi and Weihua Zhuang, "Combined Turbo-Code and Modulation for CDMA Wireless Communications", *Proceedings of the IEEE Vehicular Technology Conference*, VTC'98, pp. 1920-1924, Ottawa, Ont., May 1998.

[58] A. H. S. Mohammadi and W. Zhuang, "Variance of the Turbo-code performance bound over the interleavers," In *Proceedings of the 49th IEEE Vehicular Technology Conference*, VTC'99, Houston, Texas, May 1999.

[59] A. H. S. Mohammadi and W. Zhuang, "Variance of the Turbo-Code Performance Bound Over the Interleavers," Submitted to *IEEE Transactions on Information Theory*, June 1999, 28 pages, Revised.

[60] M. Naßan and P.Jung, "New results on the application of antenna diversity and Turbo-codes in a JD-CDMA mobile radio system," *Proceedings of the 5th International Symposium on Personal, Indoor and*

Mobile Radio Communications, PIMRC'94, pp. 524-528, Netherlands, 1994.

[61] K.R. Narayanan and G. L. Stuber, "List decoding of Turbo-codes," *IEEE Transactions on Communications,* vol. 46, no. 6, pp.754-762, June 1998.

[62] K. Pehkonen and P. Komulainen, "A super orthogonal Turbo-code for CDMA applications," *Proceedings of the IEEE 4th International Symposium on Spread Spectrum Techniques and Applications*, ISSSTA'96, pp. 580-584, Mainz, Germany, Sept. 1996.

[63] L. C. Perez, J. Seghers, and D. J. Costello Jr., "A distance spectrum interpretation of Turbo codes," *IEEE Transactions on Information Theory,* vol. 42, no. 6, pp. 1698-1709, Nov. 1996.

[64] J. G. Proakis, *Digital Communications.* 2nd ed. New York, McGraw-Hill Inc., 1989.

[65] R. Pyndiah, A. Picart, A. Glavieux, "Performance of block Turbo coded 16-QAM and 64-QAM modulation," *Proceedings of IEEE GLOBECOM'95,* pp. 1039-1043, Singapore, Nov. 1995.

[66] P. Robertson "Illuminating the structure of code and decoder of parallel concatenated recursive systematic (Turbo) codes", *Proceedings of IEEE GLOBECOM'94,* pp. 1298-1303, San Francisco, Nov. 1994.

[67] P. Robertson and T. Woerz, “Coded modulation scheme employing Turbo codes,” *Electronics Letters*, vol. 31, no. 18, pp. 1546-1547, Aug. 1995.

[68] P. Robertson and T. Woerz, “Novel bandwidth efficient coding scheme employing Turbo codes,” *Proceedings of IEEE International Conference on Communications,* ICC’96, pp. 962-967, Dallas, Texas, June 1996.

[69] A. Salmasi and K. S. Gilhousen, “On the system design aspects of code division multiple access (CDMA) applied to digital cellular and personal communication networks,” *Proceedings of IEEE Vehicular Technology Conference*, VTC’91, pp. 57-62, St. Luise, MO., May 1991.

[70] C. E. Shannon, “A mathematical theory of communication,” *Bell System Technical Journal* vol. 27, pp. 279-423, Oct. 1948.

[71] Y. V. Svirid, “Weight distributions and bounds for Turbo-codes,” *European Transactions on Telecommunications*, pp. 543-555, vol. 6, no. 5, Sept.-Oct. 1995.

[72] G. Ungerboek, “Channel coding with multilevel phase signaling,” *IEEE Transactions on Information Theory,* vol. IT-25, pp. 55-67, 1982.

[73] A. J. Viterbi, *CDMA Principles of Spread Spectrum Communication*, Addison-Wesley Pub. Co., c1995.

[74] J. Vogt, K. Koors, A. Finger, G. Fettweis, "Comparison of different turbo decoder realizations for IMT-2000", *Proceedings of GLOBE-COM'99*, Rio de Janeireo, Brazil, Dec. 1999.

[75] B. Vucetic, J. Yuan, *Turbo codes : principles and applications* , Kluwer Academic Publishers, Boston, c2000.

[76] S. B. Wicker, *Error Control Systems for Digital Communications and Storage.* Englewood Clifs: Prentice Hall, 1995.

[77] J. Yuan, B. Vuceti, and W. Feng, "Combined Turbo-codes and interleaver design", *Proceedings of IEEE International Symposium of Information Theory,* ISIT'98, p. 176, Cambridge, MA, Aug. 1998.

Glossary

APP	A-Posteriori Probability
AWEF	Average Weight Enumerating Function
AWGN	Additive White Gaussian Noise
BCJR	Bahl, Cock, Jelinek, and Raviv (algorithm)
BER	Bit Error Rate
BPSK	Binary Phase Shift Keying
CDMA	Code Division Multiple Access
CWEF	Conditional Weight Enumerating Function
EPA	Equal Power Allocation
IIR	Infinite Impulse Response
IRWEF	Input-Redundancy Weight Enumerating Function
LLR	Logarithm of Likelihood Ratio
MAP	Maximum A-Posteriori
ML	Maximum Likelihood
MSWEF	Mean Square of the Weight Enumerating Function
PCTCM	Parallel Concatenated Trellis Coded Modulation
QAM	Quadrature Amplitude Modulation
RSC	Recursive Systematic Convolutional

SISO	Soft Input Soft Output
SNR	Signal to Noise Ratio
TCM	Trellis Coded Modulation
TTCM	Turbo Trellis Coded Modulation
UEP	Unequal Error Protection
UI	Uniform Interleaver
UPA	Unequal Power Allocation
WEF	Weight Enumerating Function