PERFORMANCE OF TURBO CODES IN RAYLEIGH FADING ENVIRONMENTS WITH AWGN

A thesis Submitted to the Graduate School of Natural and Applied Science of
Dokuz Eylül University
In Partial Fulfillment of the requirements for the Degree of Master of Science in
Electrical and Electronics Engineering

by
Ali ÖZÇELİK

109597

July, 2001
İZMİR

Ms.Sc. THESIS EXAMINATION RESULT FORM

We certify that we have read the thesis, entitled "Performance of Turbo Codes in Rayleigh Fading Environments with AWGN" completed by Ali Özçelik under supervision of Asst. Prof. Dr. Reyat YILMAZ and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

[illegible]

Asst. Prof. Dr. Reyat YILMAZ

(Supervisor)

[illegible]

Prof. Dr. Kemal ÖZMEHMET

(Committee Member)

[illegible]

Asst. Prof. Dr. Mustafa A. Altınkaya

(Committee Member)

Approved by the

Graduate School of Natural and Applied Sciences

[illegible]

Prof. Dr. Cahit Helvacı

(Director)

ACKNOWLEDGEMENTS

First, I would like express my gratitude to Asst. Prof. Dr. Reyat YILMAZ, my supervisor, for having given to me the chance to study in such an exciting field of research. I want to thank him also for his guidance and advices throughout this study.

I would particularly like to thank to my friends Müjdat AYBAR, Erdinç MUTLU, Mümtaz YILMAZ, and Serdar BAYIR for their patience, encouragement and continuous support.

Finally, I would like to thank to my family for their support without which I could not finalise this thesis.

ABSTRACT

Fading in wireless environments is one of the biggest obstacles to reliable communications. The basic goal of this thesis is to investigate the performance of a turbo coded BPSK system over Rayleigh Fading environments, based on iterative decoding, soft-in/soft-out decoding, channel interleaving, parallel concatenation of two recursive systematic convolutional codes and non-uniform random interleaving.

As turbo codes became one of the most powerful error correction techniques since their first introduction in 1993 by Claude Berrou, (Berrou et al, 1993) their advantages for reliable communications over fading environments should also be explored.

In this thesis BPSK signalling over correlated flat fading channels with channel interleaving and with both perfect and estimated channel state information are considered. It has been shown that turbo codes are also well suited as a channel coding scheme in such environments and they achieve reasonable BERs.

As a decoding technique mainly the MAP (Maximum *A Posteriori* Probability) or the soft output Viterbi algorithm (SOVA) is used. In the simulations, which were created with MATLAB program, the MAP algorithm is used to explore the resultant BERs for different considerations (for different number of iterations, with and without the usage of channel block interleaver and fading estimator in the case of Rayleigh environment).

It has been shown that channel interleaving is necessary on a correlated fading channel, and found that the performance of the system with channel interleaver is considerably better than without channel interleaving applied.

ÖZET

Kablosuz haberleşme ortamlarında sinyal zayıflaması ve sönümlenmesi, güvenilir haberleşmenin gerçekleşmesi için karşımıza çıkan en büyük engel olmaktadır. Bu tezin basit anlamda amacı; özyinelemeli deşifreleme, kanal serpiştiricili, iki adet tekrarlı sistematik konvolüsyonel kodun paralel yapısı esasına dayanan BPSK modülasyon kullanan turbo kodlama sistemlerinin Rayleigh sönümlü ortamlardaki performansının incelenmesidir.

1993 yılında Claude Shannon tarafından ilk kez ortaya atılmalarından bugüne, en güçlü hata düzeltme tekniklerinden birisi haline gelen turbo kodlama sistemlerinin performanslarının sönümlü ortamlarda da araştırılması gereklidir.

Bu tezde, BPSK modülasyonlu sinyallerin toplanır Gauss gürültülü ve Rayleigh sönümlü kanallarda, kanal serpiştirici, tam ve tahmin edilmiş kanal durum bilgisi kullanılarak simüle edilmelerine çalışılmıştır. Turbo kodlama sistemlerinin bu tür ortamlara da çok iyi uyum sağladığı ve en iyi bit hata oranı değerlerine ulaştığı gösterilmiştir.

Turbo deşifreleyici algoritmaları olarak çoğunlukla, MAP (Maximum *A Posteriori* Probability) veya SOVA (Soft Output Viterbi Algorithm) algoritmaları kullanılmaktadır. Tezde, MATLAB programlarıyla oluşturulan simülasyonlarda değişik durumlar için (değişik sayılarda yinelemeli deşifreleme, kanal blok serpiştiricinin ve kanal sönüm tahminleyicinin kullanıldığı ve kullanılmadığı gibi) deşifreleyicide MAP algoritması kullanarak ortaya çıkan sonuçlar incelenmiş ve karşılaştırma sonuçları yorumlanmıştır.

Sonuç olarak Rayleigh sönümlü kanallarda, kanal blok serpiştiricinin gerekli olduğu, sistem performanslarının karşılaştırma sonuçlarına dayanılarak gösterilmiştir

CONTENTS

CHAPTER THREE

TURBO ENCODING AND DECODING TECHNIQUES AND INTERLEAVER STRUCTURES

CHAPTER FOUR

SYSTEM MODEL

CHAPTER FIVE

THE SIMULATION RESULTS OF THE INTRODUCED SYSTEM

CHAPTER SIX

CONCLUSIONS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE
INTRODUCTION

The ever increasing demand for information exchange is a common characteristic of most areas of modern civilisation. The transfer of information must be done in a way that the quality of the received and the transmitted information should be as close as possible to each other.

A typical communication system is shown in Figure 1.1 as a block diagram. The information to be transmitted can be machine generated (*e.g.*, images, computer data) or human generated (*e.g.*, speech). Regardless of its source, the information should be translated into a set of signals optimised for the channel over which we want to transmit it. The first step is to eliminate the redundant part in order to maximise the information transmission rate. This is achieved by the source encoder block in Figure 1.1. In order to ensure the secrecy of the information that we want to send, an encryption method should be used. The data also has to be protected against perturbations which could lead to misinterpretation of the transmitted signal at the receiving end. This protection can be done through error control strategies : forward error correction (FEC), that is, using error correction codes that are able to correct errors at the receiving end, or automatic repeat request (ARQ) systems. The modulator block generates a signal that is suitable for the communication channel.

From coding theory, it is known that by increasing the codeword's length or the encoder memory , greater protection, or coding gain, can be achieved. At the same time the complexity of maximum likelihood decoding algorithms (MLDA) increases exponentially and the algorithms become difficult to implement. The increased error correction capability of long block or convolutional codes requires a very high computational effort. This led to research for new coding techniques which could replace the MLDA with simpler algorithms. These new strategies combine more powerful codes in a way that allows each code to be decoded separately with less complex decoders. Using this iterative technique, we can approach the performance of the MLDA.

We can include to that coding category; product block codes, concatenated codes, multilevel codes, and finally, turbo codes. All of these codes have the common feature of decoding in a sequential fashion, using one decoder at a time. This is a necessary condition for simpler decoding.

Since their first introduction in 1993 by Claude Berrou, (Berrou et al, 1993) turbo codes became the best coding technique among the others by showing the best BERs. The reason is that in the case of other iterative codes, no information is exchanged between the two decoders in the receiver structure. However with the introduction of turbo codes the information exchange between the two decoders became possible. The reason behind is that turbo codes encode the same information twice but in a different order. As a result, the more "scrambled" the information sequence, the more "uncorrelated" the information exchange is.

Several studies have shown that turbo codes may achieve good bit error performance over flat Rayleigh fading channels (Hagenauer & Robertson, 1994), (Jung, 1996), (Koorapaty et al, 1997), (Valenti & Woerner, 1997), (Hall & Wilson, 1998). Many of these studies assume that the fades are fully interleaved. In order for this assumption to be valid, a channel interleaver is required and it must have a depth greater than the ratio of the channel coherence time to the symbol duration. In case of slow fading the interleaver may not separate the fades good enough so the performance degrades.

The performance of turbo codes in the presence of non fully-interleaved flat Rayleigh fading channels has been studied in the proceeding "Design and performance of turbo codes on Rayleigh Fading channels by E. Hall. In this thesis it is shown that the performance of turbo codes degrades quickly as the product of the fading bandwidth and the symbol duration decreases. However in the above mentioned study, there was no additional channel interleaving used.

The subject of channel estimation was addressed in "The effects of channel characteristics on turbo code performance" by M. Jordan. This study was limited to the effect of noise variance estimation errors. However again the effect of fading amplitude estimation errors was not provided.

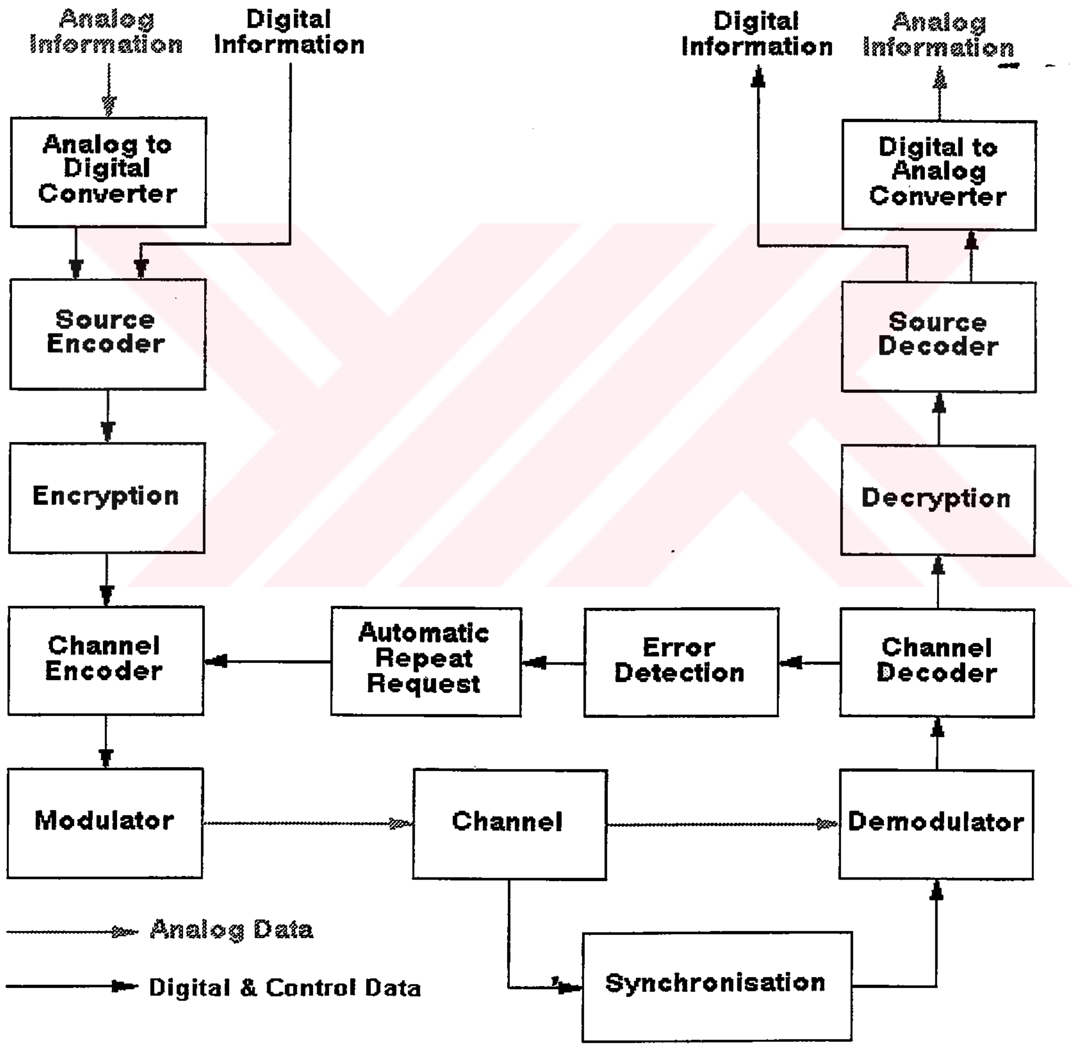


Figure 1.1 A Typical System Diagram of a Modern Communications System

The main objective of this study is to show that channel interleaving is necessary on a correlated fading channel and that performance degrades as the product of the fading bandwidth and the symbol duration decreases.

The chapters of this thesis are organised as follows. In this chapter besides the basics of digital communication systems; historical evaluation of the error control coding and Shannon's capacity bound for reliable communications are presented.

In the second chapter the basics of error control coding and especially the convolutional codes are presented. Besides the BPSK modulation technique used in the simulations is explained shortly.

Turbo coding and decoding techniques are discussed in chapter three. Turbo encoder and decoder structures, decoder algorithms, the basic performance parameters and interleaver structures are explained. In the fourth chapter the system model is presented. The blocks of the whole model are explained and their functions are given.

Simulation results and the performance curves of the system are given in chapter five. Also the effect of the channel block interleaver and de-interleaver and the effect of doppler frequency on the overall performance of the system is discussed. It is seen that the modifications for the Rayleigh fading channels are necessary and improve the performance of the system.

1.1 HISTORICAL PROGRESS

It is arguable that the information age had its beginnings in the late 1940's. Although depending upon much earlier mathematical and scientific work, three discoveries in particular were about to change the way the world communicated. Not only were the announcements of these discoveries almost simultaneous, they stemmed from a single commercial research laboratory.

Claude Shannon (1916-), a mathematician working at Bell Laboratories, is generally regarded as the father of Information Theory. Prior to Shannon, it was believed that in order to achieve higher reliability in communications over noisy channels, it was necessary to keep increasing the transmit power. In his paper "A Mathematical Theory of Communication", *Bell System Technical Journal* Vol. 27, pp 623-656, July, October 1948, Shannon introduced a fundamental measure of information – entropy, and coined the term "bit", referring to the amount of information in a single binary digit (Shannon, 1948). He went on to show that every noisy channel has associated with it an information capacity C bits/second, below which it is possible to encode messages (by representing them as long binary strings) such that they can be received and decoded with arbitrarily low probability of error. He defined the capacity bound for a band-limited additive white Gaussian channel (AWGN) as;

$$C = W \log_2(1 + E_s/N_0) \text{ bits per second} \qquad (1.1)$$

where W is the bandwidth of the channel in Hertz and E_s is the average signal energy in each two-dimensional signalling interval of duration T seconds. E_s/N_0 is called the signal to noise ratio (SNR) where $N_0/2$ is the double sided noise density. His proof of the capacity theorem however relied on a mathematical "trick". He showed for transmission rates below C, that the error probability averaged over all randomly selected codes can be made as small as desired. While this implies the existence of good codes, it left open the problem of designing such codes.

Shannon did however refer to a practical code in his paper. This code was found by his Bell Labs colleague, Richard Hamming (1915-1998) (Shannon, 1948). Hamming became interested in the idea of codes that could correct errors when he was working on an early "relay computer" at Bell Labs. Upon detection of an error, the machine would sound an alarm, and the operator would try to fix it. Unfortunately for Hamming, but fortunately for Error Control Coding, Hamming only had access to the computer on the weekends, when there was no operator to fix errors. He decided that he needed a way of enabling the computer to automatically correct errors. Hamming constructed a code which added three additional "parity" bits to four information bits. The parity bits are chosen based on the information bits in such a way that the resulting codeword must obey a set of linear equations. If any single error occurs (a bit is reversed), the codeword no longer satisfies the equations, and furthermore, it is possible to tell which position is in error. The Hamming code was the first result in the field that is now known as Coding Theory, the object of which is to find good codes that are also easy to implement.

When it comes to implementation, one cannot overlook the impact of the transistor on the information age. Dr. John Bardeen, Dr. William Shockley and Dr. Walter Brattain, all members of Bell Laboratories, first observed the transistor effect on December 15, 1947. The invention was announced June 30, 1948 (Shannon, 1948). Without the transistor and other solid state devices, computing and communications would have languished inside house sized boxes filled with vacuum tubes or relays.

Turning back to the Equation 1.1, by making little manipulation, we can find the limit on coding gain for our BPSK-modulated system that we will make use of in our simulation. Let η be the *spectral (or bandwidth) efficiency* of the modulation format. The spectral efficiency of a modulation format is the average number of information bits transmitted per two-dimensional signalling interval of duration T. If T is normalised and related in terms of inverse-bandwidth (assuming perfect Nyquist signalling : WT = 1), η is expressed in terms of bits per second per Hertz (b/s/Hz). It follows that $E_s/N_0 = \eta\ E_b/N_0$, where E_b is the average energy per information bit. Substituting into Equation 1.1 and rearranging, we obtain

$$E_b/N_0 > 2 \exp \eta - 1 / \eta \qquad (1.2)$$

For a given spectral efficiency, this expression sets the limit on SNR above which there exists coding schemes that will provide arbitrarily low bit error rates.

1.2 ERROR CONTROL CODING TIMELINE

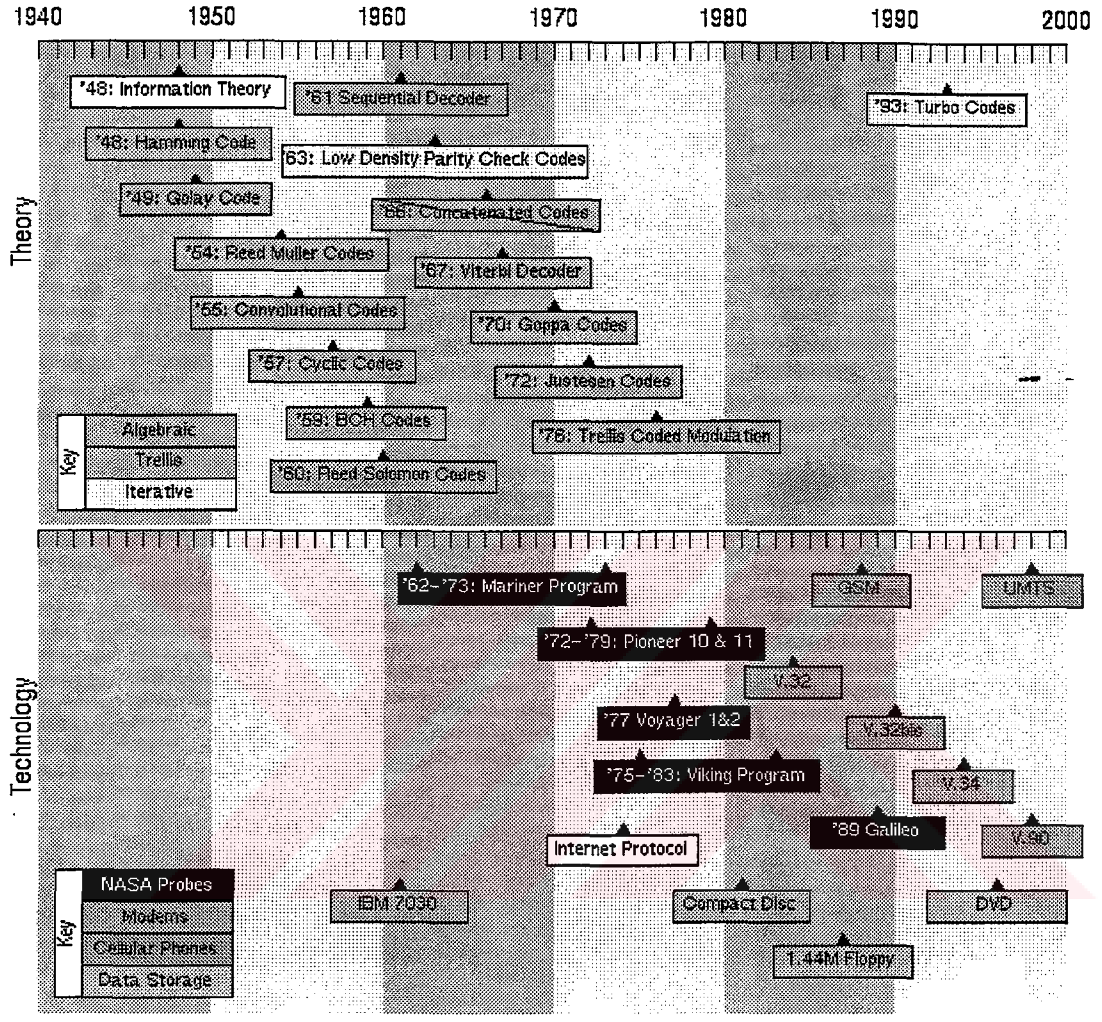


Figure 1.2 Error Control Coding Timeline (Grant, 1999)

As a summary we may say that the goal of error control coding is to design controlled ways of introducing redundancy such that the errors can be detected or corrected automatically using digital circuits. In the Figure 1.2 the timeline shows some of the important developments in the theory and applications of error control coding.

1.3 APPROACHING SHANNON'S CHANNEL CAPACITY

In Figure 1.3 we show some of the milestones towards reaching the Shannon's channel capacity. Shannon's coding theorem for the AWGN channel shows that points to the left of the black line on the figure are not achievable, regardless of the coding scheme used.

Along the x-axis is the bit energy to noise power spectral density ratio, in dB. This is a measure of how much noise is added to the signal. The y-axis carries the "spectral efficiency", measured in message bits/code bit. This is a measure of how much redundancy has been added by the code. It is called the spectral efficiency because it is roughly related to the amount of bandwidth expansion required if the transmission rate is kept constant.

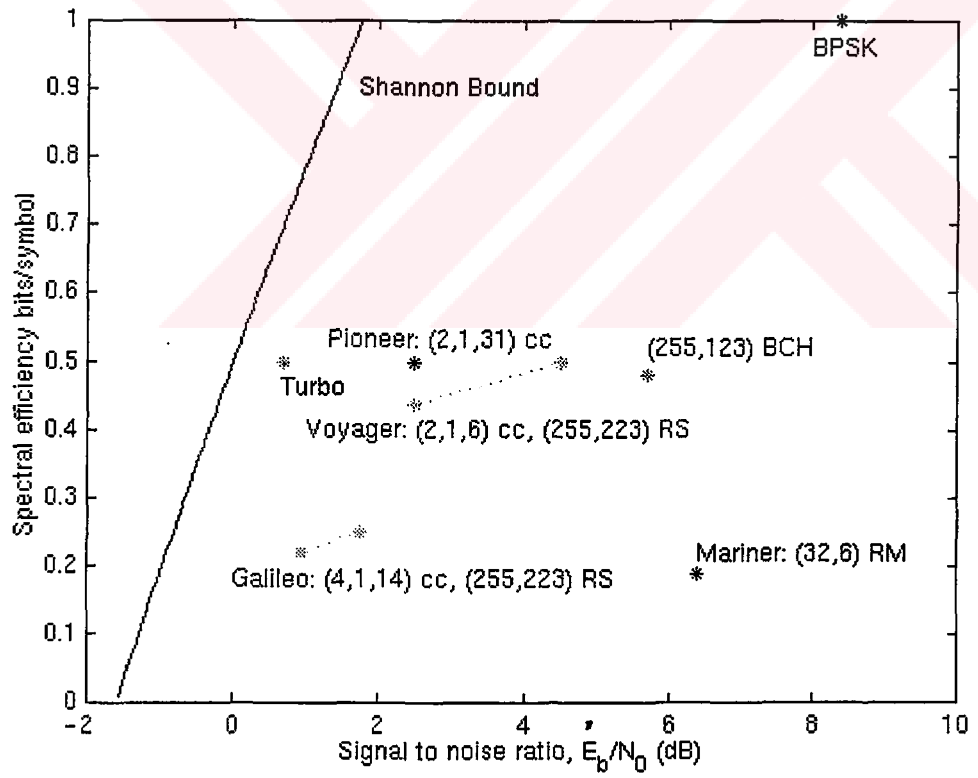


Figure 1.3 Approaching Shannon's Channel Capacity (Schlegel, 1997)

From the figure, one can see that uncoded BPSK is about 8dB away from the Shannon limit. On the other hand turbo codes come within fraction of dB!

In Figure 1.3, we also show some of the examples from Figure 1.2. Two points are shown for the Voyager and Galileo systems. The points to the right are using only the convolutional code. The leftmost points add an outer Reed-Solomon code for improved performance (but reducing the spectral efficiency).

CHAPTER TWO

ERROR CONTROL CODING TECHNIQUES

As we know, in the case of linear block codes, for every input message block of length *k*, we have an output codeword of length *n*. Block codes have no *memory,* that is each codeword depends only upon the corresponding message block. If we allow the output blocks to depend not only on the current input block, but on a finite number of previous input blocks, we have the class of codes known as *trellis codes*.

The error-correcting capability of the code is mainly due to the memory, which may be large. A generic block diagram of a *(n, k, m)* trellis encoder is shown in Figure 2.1. At each *frame time* an input frame of length *k* is input to the encoder. The encoder state is stored in a memory and depends upon the last *m* input frames. Combinational logic is used to form an output frame of length *n* from the current input frame and the encoder state. This block may then be mapped onto the modulation signal set (e.g. 8-PSK, 16-QAM) using a mapper function. We shall not be concerned with the joint design of the encoder and mapper function, which is the subject of trellis-coded modulation.

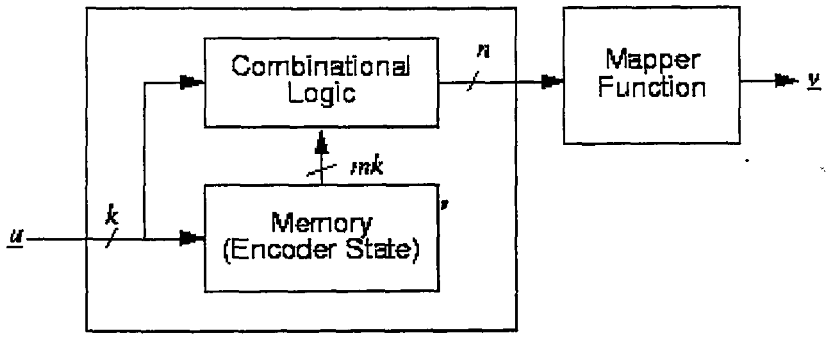


Figure 2.1 Generic Form of a Trellis Encoder

One of the most important distinctions between block codes and trellis codes is that block codes encode message blocks of finite length, producing finite length code-words. Trellis codes on the other hand encode message *sequences* of infinite length, producing infinite length codeword sequences. Trellis codes are so called because they can be described by a *trellis diagram*, which is really just a representation of the finite-state machine that is the encoder.

As an example the following figure shows the trellis diagram for a (2, 1, 2) trellis code. The memory of the code is the two bit state vector. Each state transition is labelled with the corresponding input and output frame. Notice that only certain state transitions are allowed. The encoder operation can be thought of as tracing a path through a trellis, consisting of copies of the code state diagram. Figure 2.3 shows the result of encoding the input sequence u = 11001 with the code defined by the trellis in Figure 2.2. Note that this is only a segment of the infinite length information and codeword sequences.

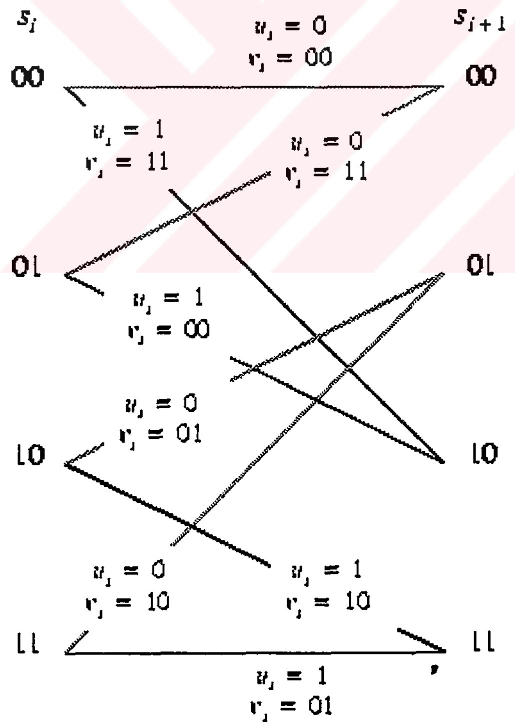


Figure 2.2 Trellis Diagram for a (2,1,2) trellis code

The class of convolutional codes is a sub-class of the trellis codes.

2.1 CONVOLUTIONAL CODES

Definition (Convolutional Code) : A (n, k, m) convolutional code is a mapping from the set of sequences of GF(q) onto itself, such that:

Tree Structure : For any M, if two input sequences agree in the first Mk elements, the corresponding output sequences agree in the first Mn elements.

Linearity : The mapping is linear. For a mapping G, scalars a, b and sequences $\underline{u}1$, $\underline{u}2$ we have $G(a\,\underline{u}1 + b\,\underline{u}2) = aG(\underline{u}1) + bG(\underline{u}2)$.

Time Invariant : If two input sequences are identical, apart from a time-shift by an integer number of frames, the codeword sequences are also identical apart from a time shift of the same number of frames.

Finite Memory : The mapping has finite memory, meaning that any input bit affects only a finite number of codeword bits.

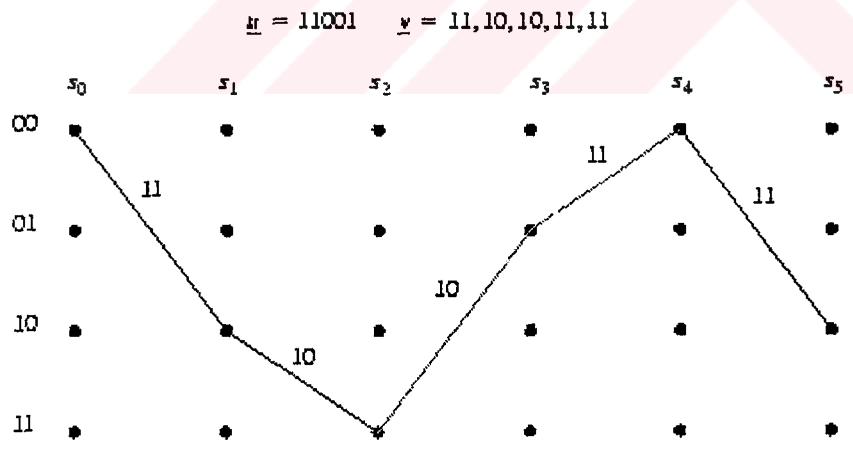


Figure 2.3 Codeword generation using the trellis diagram

Definition (Rate) : The rate of a trellis code is defined as

$$R = \frac{k}{n} \tag{2.1}$$

Definition (Systematic) : A convolutional code is called systematic if the input sequence appears unchanged in the output sequence.

2.1.1 Encoder Implementation

A convolutional encoder is a linear time-invariant system, and may therefore be described as a discrete-time finite-field convolution of the message sequence with a time-invariant generator sequence, which is the impulse-response of the encoder. The encoder for a convolutional code can always be implemented as a discrete-time feed-forward filter. Recall that the convolution of a sequence $\underline{u}$ by a length *m* impulse response $\underline{g}$ is given by

$$v_i = \sum_{l=0}^{m-1} u_{i-l}\, g_l \tag{2.2}$$

As an example, Figure 2.4 shows a feed-forward implementation of the (2; 1; 2) code from Figure 2.2. This code takes input frames of length 1 and produces output frames of length 2. Note that the bits in the output frame are produced in parallel, and must be multiplexed to form a serial data stream. We use the superscript (i) notation to signify the i-th position within a frame. The subscript refers to the frame number. The output sequences, or "streams"; v^0 and v^1 are interlaced to form the codeword sequence;

$$\underline{v} = v_0^0, v_0^1, v_1^0, v_1^1, v_2^0, v_2^1, \ldots. \tag{2.3}$$

If we drive the encoder with an impulse, $\underline{u}$ = 100000…., we get an impulse response of

$$g^{(0)} = 10100\ldots \tag{2.4}$$

for the first output, v^o, and

$$g^{(1)} = 11100... \tag{2.5}$$

for the second output, v^1. Note that once the encoder state is all-zero and the input consists of only zeros, we see only zeros at the output. The generator sequences can be read from the figure as the non-zero filter taps. The encoder outputs are given by the following modulo-2 discrete-time convolutions,

$$v^{(0)} = \underline{u} * (101) \tag{2.6}$$

$$v^{(1)} = \underline{u} * (111)$$

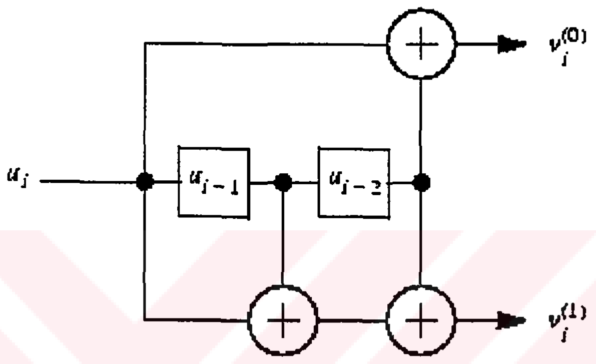


Figure 2.4 (2, 1, 2) convolutional encoder

Definition (Memory Order) : The memory order of a convolutional encoder is the length of its longest shift register,

$$m = \max_{i=0,1,....,k-1} m_i \tag{2.7}$$

When we refer to an *(n,k,m)* code, *m* is the memory order of the code. Each output frame will depend on the *m* most recent input frames (but not necessarily every bit of these frames).

Definition (Total Memory) : The total memory of a convolutional encoder is the total number of memory elements required for the implementation, namely

$$M = \sum_{i=0}^{k-1} m_i \tag{2.8}$$

We shall see that the complexity of decoding of a convolutional code is heavily dependent on its total memory. The number of states in the trellis diagram for a code over GF(q) with total memory M is

$$q^M$$

since each memory element can take on one of the *q* values.

Definition (Constraint Length) :The constraint length *v* of a convolutional code is defined as the maximum number of bits in a single output stream that can be affected by a single input bit, which is equal to

$$v = 1 + max\, m_i \qquad (2.9)$$

The expression given in the last definition for the constraint length is due to the fact that if the longest shift register is length *m* (say), than a particular bit input to this register can only affect the output at the time of input, followed by the next *m* frames times. After this, the bit has been discarded from the shift register.

2.1.2 Decoding

The decoder for a convolutional code consists of two parts, a codeword estimator, and an inversion circuit for recovering the transmitted message sequence from the codeword. This is shown in Figure 2.5. It is clear that in order for a convolutional code to be any good, the encoder must have a unique (pseudo) inverse. Note that for a systematic code, the encoder inverse always exists and is trivial, since the message bits may be read straight from the codeword sequence.

It is the codeword estimator, a non-linear function that is the most complex part of the decoder. Just as for block codes, the maximum likelihood decoder must find the codeword sequence that is closest to the received vector (minimum distance/maximum likelihood decoding).

However for truly infinite length sequences, we would have to wait forever before making a decision. In practice, the received sequence is truncated, with little effect on the probability of error (for long messages).

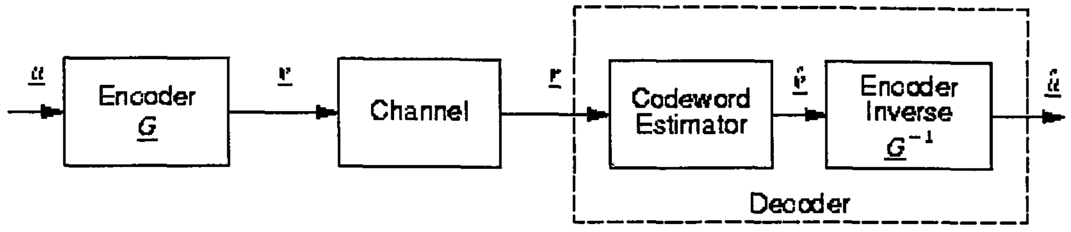


Figure 2.5 Convolutional Code : Encoder and Decoder (Barbulescu & Pietrobon, 1994).

For finite length blocks, it is common that the encoder is terminated in the all zero state, which assists the decoding process. This however introduces some rate loss (since for a memory order *m* encoder, we need to enter *m* frames of *k* zeros, which can convey no information. This is called *flushing* the encoder). For an L-bit message sequence, encoded with an *(n, k, m)* code, we must append *km* zeros to the message, and *nk* extra code bits are produced. The resulting code has rate

$$R_{effective} \frac{L+km}{nL/k+nm}. \tag{2.10}$$

Definition (Minimum Free Distance) The minimum free distance of a convolutional code is defined to be

$$d_{free} = min\{d(u,v)\}. \tag{2.11}$$

CHAPTER THREE

TURBO ENCODING & DECODING TECHNIQUES AND INTERLEAVER STRUCTURES

Historically, product codes represented the first attempt at achieving a higher error correction capability without the decoding complexity required by long codes. Unfortunately, the alternative decoding of the component codes did not produce the improvements in performance as expected, even though the error correction power of the product code increased as a whole. This was mostly due to the hard-in-hard-out (HIHO) decoding algorithms that were traditionally used.

Better results than for block codes were obtained with concatenated codes. The component codes are called the inner and the outer codes. First, the decoding is done for each inner code vector. The decoded bits are then decoded again according to the outer code used (Forney, 1966). Multilevel coding uses several error-correcting codes with different error correcting capabilities. The transmitted symbols are constructed by combining symbols of codewords of these codes. In the study of "Multilevel codes based on patitioning" by G., P. Pottie, a staged decoder for a multilevel partition code passes reliability information only in one direction, from the first decoding stage through intermediate stages to the last one.

The new class of turbo codes encodes ,the same information twice, but in a different order. The more "scrambled" the information sequence is for the second encoder, the more "uncorrelated" the information exchange is between the decoders. This is one of the key ideas that allows a continuous improvement in correction capability when the decoding process is iterated.

In the traditional approach, the demodulator block from Figure 1.1 makes a "hard" decision of the received symbol and passes it to the error control decoder block. This is equivalent to deciding which of two logical values - say 0 and 1 - was transmitted. No information was passed about how reliable the hard decision was. Better results are obtained when the quantised analogue received signal was passed directly to the decoder. This led to the development of "soft" input decoding algorithms. This was the best solution if only one code was used. For the same reason, in the case of combining more codes as explained above, new SISO algorithms were developed in order to pass more information from the output of one decoder to the input of the next decoder.

Soft output decision algorithms provide as an output a real number which is a measure of the probability of error in decoding a particular bit. This can also be interpreted as a measure of the reliability of the decoder's hard decision. This extra information is very important for the next stage in an iterative decoding process, as will be shown later. There are two important categories of soft output decision algorithms. The first category includes the maximum likelihood decoding algorithms which minimise the probability of symbol error, such as the maximum *a posteriori* probability (MAP) algorithm (Bahl et al, 1974), (Barbulescu & Pietrobon, 1994). The second category includes the maximum likelihood decoding algorithms which minimise the probability of word or sequence error, such as the Viterbi algorithm (Viterbi, 1967) or soft output Viterbi algorithm (SOVA) (Hagenauer, 1991).

Iterative decoding relies on a link between soft output decision algorithms, special encoding, and information transfer techniques. Since the early 1990's, these concepts have been combined to create more powerful decoders. The link between the three created a new powerful decoding technique and led to the appearance of turbo codes, which made possible communications very close to channel capacity.

The definition of turbo codes was given for the first time in (Berrou et al, 1993). They represent a particular class of parallel concatenation of two recursive systematic convolutional codes. Since then, the term "turbo codes" has been extended to different concatenation schemes as explained in the following section.

3.1 ENCODING

In Figure 3.1 we present a generic turbo encoder in two dimensions. The input sequence of the information bits is organised in blocks of length N. The first block of data will be encoded by the ENC^{-} block which is a rate half recursive systematic encoder. The same block of information bits is interleaved by the interleaver INT, and encoded by ENC I which is also a rate half systematic recursive encoder. The coded bit produced by the encoder is the output of each encoder block.

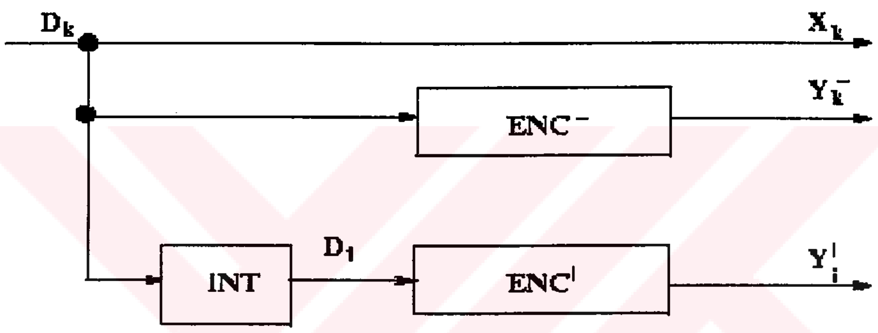


Figure 3.1 Generic rate 1/3 turbo encoder

Due to similarities with product codes, we can call the ENC^{-} block the encoder in the "horizontal" dimension and the ENC I block the encoder in the "vertical" dimension. The interleaver block, INT, rearranges the order of the information bits for input to the second encoder. The main purpose of the interleaver is to increase the minimum distance of the turbo code such that after correction in one dimension the remaining errors should become correctable error patterns in the second dimension.

Ignoring for the moment the delay for each block, we assume both encoders output data simultaneously. This is a rate 1/3 turbo code, the output of the turbo encoder being the triplet $(X_k, Y_k^{-}, Y_k^{|})$. This triplet is then modulated for transmission across the communication channel. Since the code is systematic, $D_k = X_k$ is the input data at time k. Y_k^{-} and $Y_k^{|}$ are the two parity bits at time k.

The two encoders do not have to be identical. In Figure 3.2 the two encoders are rate 1/2 systematic encoders with only the parity bit shown. The parity bits can be "punctured" as in Figure 3.2 where puncturing is implemented by a multiplexing switch in order to obtain higher coding rates.A rate 1/2 turbo code can be implemented by alternatively selecting the outputs of the two encoders in order to produce the following output sequence:

$$(X_1, Y_1^-, X_2, Y_2^|, X_3, Y_3^-, X_4, Y_4^| \dots) \qquad (3.1)$$

The symbol D represents a D flip-flop (the clock is not shown) and the symbol + represents an exclusive-OR gate.

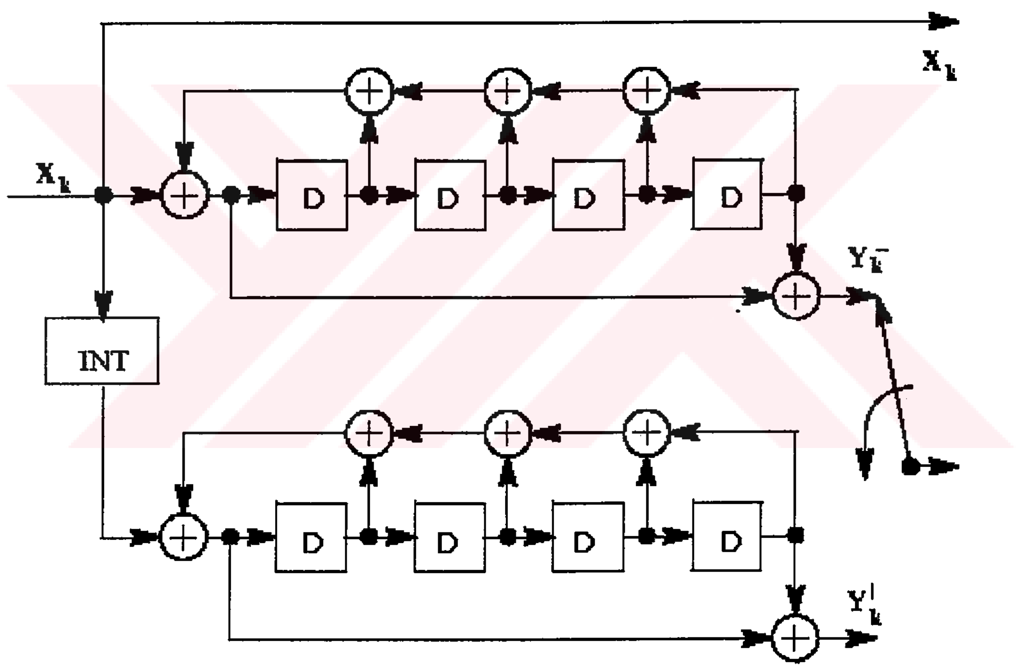


Figure 3.2 A rate half turbo code

Figure 3.2 shows a particular implementation of a two dimensional turbo code using recursive systematic codes (RSC). Lower coding rates can be achieved using either less puncturing or more interleaver and encoder blocks as shown in Figure 3.3 for an *n*-dimensional turbo code (Barbulescu & Pietrobon, 1994). The advantage of using more interleavers will become clear in the coming sections.

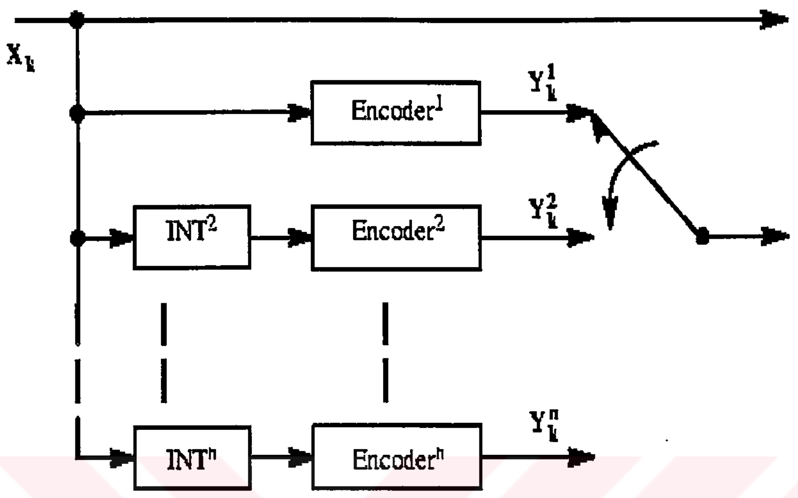


Figure 3.3 Low rate turbo encoder

The turbo code presented in the study of "Source-controlled channel decoding" by J. Hagenauer, the achieved BERs were less than 10 exp-5 within 0.7 dB of Shannon capacity. The basic turbo encoder has two constituent convolutional encoders separated by a random interleaver. This structure is called the parallel concatenated convolutional code (PCCC) structure since the same information stream is encoded twice, in parallel, using the straight and interleaved sequences of the information bits. Figure 3.1 also shows a PCCC structure.

An obvious alternative is to interleave the output of one encoder and re-encode it again. As shown in Figure 3.4, this is called the serial concatenated convolutional code (SCCC) structure. This structure has been proposed in several studies (Couleaud, 1995), (Benedetto & Montorsi, 1996), (Pollara et al, 1996).

Performance of serial and parallel concatenated convolutional schemes with iterative decoding techniques for different interleaver designs were investigated in some studies (Divsalar et al, 1996), (Gray, 1998). It was found that PCCC schemes perform better than SCCC schemes at low SNRs. However, increasing the SNR, SCCC schemes outperform PCCC schemes. The cross –over point depends on the interleaver size and interleaver design.

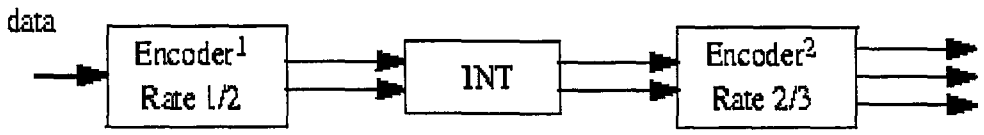


Figure 3.4 Serial Concatenated Convolutional Encoder

Any other possible combination of PCCC and SCCC can be used. One particular structure was labelled as a hybrid concatenated convolutional code (HCCC) structure (Divsalar & Pollara, 1997). This structure is shown in Figure 3.5 and may offer performance advantages over either PCCC or SCCC structures.

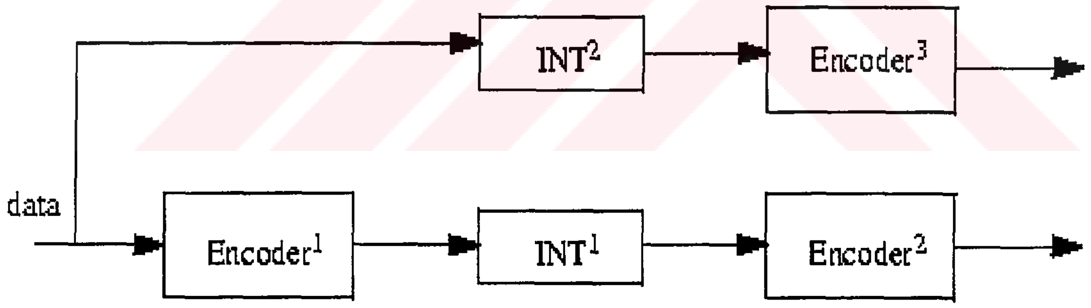


Figure 3.5 Hybrid Concatenated Convolutional Encoder

3.2 INTERLEAVING

The interleaver design is a key factor which determines the good performance of a turbo code. Some interleaver types used in turbo codes are presented in the following sections.

3.2.1 "Row-Column" Interleaver

The simplest interleaver is a memory in which data is written row-wise and read column-wise. This is called a "row-column" interleaver and belongs to the class of "block" interleavers. For example, data is written as shown in Table 3.1.

x_1	x_2	x_3
x_4	x_5	x_6
x_7	x_8	x_9
x_{10}	x_{11}	x_{12}
x_{13}	x_{14}	x_{15}
x_{16}	x_{17}	x_{18}
x_{19}	x_{20}	x_{21}

Table 3.1 Writing data row-wise in memory.

The interleaving process is done by reading data as shown in Table 3.2.

x_1	x_4	x_7	x_{10}	x_{13}	x_{16}	x_{19}	x_2	x_5	x_8	x_{11}	x_{14}	...	...	...

Table 3.2 Reading data column-wise from memory

3.2.2 Helical Interleaver

A "helical" interleaver writes data row-wise as in Table 3.1 but reads data diagonal-wise as shown in Table 3.3.

x_{19}	x_{17}	x_{15}	x_{10}	x_8	x_6	x_1	x_{20}	x_{18}	x_{13}	x_{11}	x_9	...	...	...

Table 3.3 Reading data diagonal-wise from memory.

3.2.3 “Odd-even” Interleaver

In order to understand this type of interleaver, first we assume that we have a random sequence of binary data input to a rate one half systematic encoder and we store the odd-positioned bits, as shown in Table 3.4.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
y_1	–	y_3	–	y_5	–	y_7	–	y_9	–	y_{11}	–	y_{13}	–	y_{15}

Table 3.4 Odd-positioned coded bits

If we were now to interleave the same sequence of binary data in a pseudo-random order, encode it and store the even-positioned coded bits, the result would be as in Table 3.5.

x_a	x_b	x_c	x_d	x_e	x_f	x_g	x_h	x_i	x_j	x_k	x_l	x_m	x_n	x_o
–	z_b	–	z_d	–	z_f	–	z_h	–	z_j	–	z_l	–	z_n	–

Table 3.5 Even-positioned bits

The data which is actually sent through the channel is shown in Table 3.6; the original sequence of information bits x_i, i = 1,... 15 as in Table 3.4 and a multiplexed sequence of the odd- and even-positioned coded bits from Tables 3.4 and 3.5.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
y_1	z_b	y_3	z_d	y_5	z_f	y_7	z_h	y_9	z_j	y_{11}	z_l	y_{13}	z_n	y_{15}

Table 3.6 The output for a pseodo-random interleaver

In Table 3.4 all the odd-positioned information bits have their own coded bit. Due to the pseudo-random way of interleaving, some of the coded bits stored in Table 3.5 can be for even-positioned information bits and some for odd-positioned information bits. This means that some of the information bits will have two coded bits associated with them and others will have no coded bit associated with them. Thus, the coding power is not uniformly distributed across all the bits. So for errors which affect information bits not associated with any coded bit the decoder will perform worse in both dimensions.

An example of an "odd-even" type of interleaver is a block interleaver with an odd number of rows and an odd number of columns as in Table 3.7, in which we store row-wise the sequence of random data.

x_1	x_2	x_3	x_4	x_5
x_6	x_7	x_8	x_9	x_{10}
x_{11}	x_{12}	x_{13}	x_{14}	x_{15}

Table 3.7 3x5 Block Interleaver

We produce the coded bits and store only the odd-positioned coded bits as in Table 3.4. Now we read column-wise, encode and store the even-positioned coded bits as in Table 3.8.

x_1	x_6	x_{11}	x_2	x_7	x_{12}	x_3	x_8	x_{13}	x_4	x_9	x_{14}	x_5	x_{10}	x_{15}
–	z_6	–	z_2	–	z_{12}	–	z_8	–	z_4	–	z_{14}	–	z_{10}	–

Table 3.8 Even positioned coded bits

In Table 3.8 all the even-positioned information bits have their own coded bit present and in Table 3.4 all the odd information bits have their own coded bit present as well. When we multiplex the coded bits from both Table 3.4 and Table 3.8 we produce the coded sequence as in Table 3.9. This means that each of the information bits will have its own associate coded bit associated with it. Thus the coding power is now uniformly distributed.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
y_1	z_6	y_3	z_2	y_5	z_{12}	y_7	z_8	y_9	z_4	y_{11}	z_{14}	y_{13}	z_{10}	y_{15}

Table 3.9 Information bits and multiplexed coded bits for an "odd-even" interleaver

3.2.4 "Simile" Interleaver

In Section 3.2.3 we introduced an "odd-even" type of interleaver where each information bit is associated with one and only one coded bit. In this way the correction capability of the code is uniformly distributed over all information bits. We now impose another restriction on the interleaver design: after encoding both sequences of information bits, (the original and the interleaved one), the state of both encoders of the turbo code are to be the same. This allows only one "tail" to be appended to the information bits, which drives both encoders to the same zero state. This is why we called it a "simile" type of interleaver (Barbulescu & Pietrobon, 1995).

The idea behind the simile interleaver is that the whole block of N information bits can be divided in $\nu + 1$ sequences, where ν is the memory length of the code. For $\nu = 2$, we get:

Sequence 0 = $\{d_k \mid k \bmod (\nu + 1) = 0\}$

Sequence 1 = $\{d_k \mid k \bmod (\nu + 1) = 1\}$

Sequence 2 = $\{d_k \mid k \bmod (\nu + 1) = 2\}$

For example, consider the particular four state encoder shown in Figure 3.6.

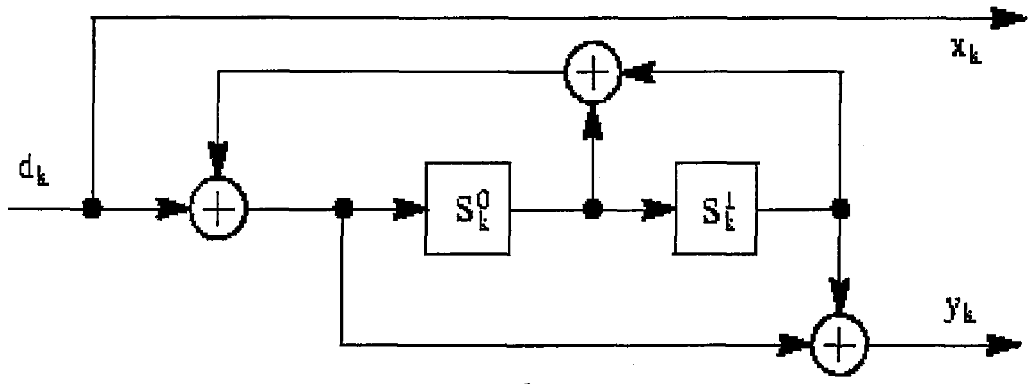


Figure 3.6 A four state systematic convolutional encoder

For a given N, the final state of the encoder represented by the state of the two D flip-flops will be a combination of the above sequences as shown in Table 3.10.

N mod (v + 1)	S^0_N	S^1_N
0	Sequence1 + Sequence2	Sequence0 + Sequence1
1	Sequence2 + Sequence0	Sequence1 + Sequence2
2	Sequence0 + Sequence1	Sequence2 + Sequence0

Table 3.10 Final encoder state for $v = 2$ for N information bits

The important conclusion is that from the point of view of the final encoder state, the order of the individual bits in each sequence does not matter, as long as they belong to the same sequence. The simile interleaver has to perform the interleaving of the bits within each particular sequence in order to drive the encoder to the same state as that which occurs without interleaving. Since both encoders end in the same state, we need only one tail to drive both encoders to state zero at the same time.

The above interleaver types can be combined in a single interleaver. An example of a "simile odd-even" block helical interleaver that can be used with the four state RSC encoder given in Figure 3.6 is shown in Table 3.11.

x_1	x_2	x_3	x_4	x_5	x_6
x_7	x_8	x_9	x_{10}	x_{11}	x_{12}
x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}
x_{19}	x_{20}	x_{21}	x_{22}	x_{23}	x_{24}
x_{25}	x_{26}	x_{27}	x_{28}	x_{29}	x_{30}

Table 3.11 Simile odd-even block helical interleaver

Part of the interleaved sequence is shown in Table 3.12.

	x_{25}	x_{20}	x_{15}	x_{10}	x_5	x_{30}	x_{19}	x_{14}	x_9	x_4	x_{29}	x_{24}	x_{13}	x_8	x_3	x_{28}	x_{23}	x_{18}	x_7	...
j mod 3	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0	1	...

Table 3.12 The output of a simile odd-even block helical interleaver

3.2.5 Pseudo-random interleavers

These pseudo-random interleavers are defined by a pseudo-random number generator or a look-up table where all integers from 1 to N (the block size to be interleaved) can be generated. This approach can lead to good or bad interleavers, especially for small interleaver sizes. The only criterion for choosing between them is based on computer simulations; there seam to be no analytical criteria.

3.2.6 "Uniform" interleavers

Performance bounds for turbo codes were given in [23] using a conceptual uniform interleaver which is the average of all possible interleavers. Consider a sequence made of **w** ones and **k-w** zeros. A uniform interleaver of length k is a probabilistic device which maps this sequence to all distinct (k,w) permutations with equal probability 1 / (k, w).

This techniques allows the analysis of a turbo code as if it were made of two independent elementary codes, due to the uniform distribution produced by the interleaver. This method gives lower bounds on the error probability which are quite accurate at high values of SNRs. However, the bounds are significantly worse from the performance which can be achieved by turbo codes at very low SNR values. Despite this, the concept of uniform interleaving is a key to understanding turbo code performance.

3.2.7 The best interleaver

Interleavers are designed for specific system considerations. There is no universal formula that can be applied.

In Figure 3.7 the extrinsic information $\mathbf{E_k}$ (which will be described in the decoding section) associated with the information bit $\mathbf{d_k}$ at time **k** is produced by the

MAP algorithm using a forward and backward recursion applied to the whole received sequence.

In can be shown that, 95% of the value of the $\mathbf{E_k}$ is determined by the previous Nm and the following Nm received symbols, where Nm is ten times the constraint length of the code. The larger the interleaver size, the more "uncorrelated" the 2Nm symbols of the straight sequence of data are, compared with the 2Nm symbols of the interleaved data. This process allows for two "independent" criteria to estimate the soft value of the same bit and then the estimates are passed as extra information from one decoder to another.

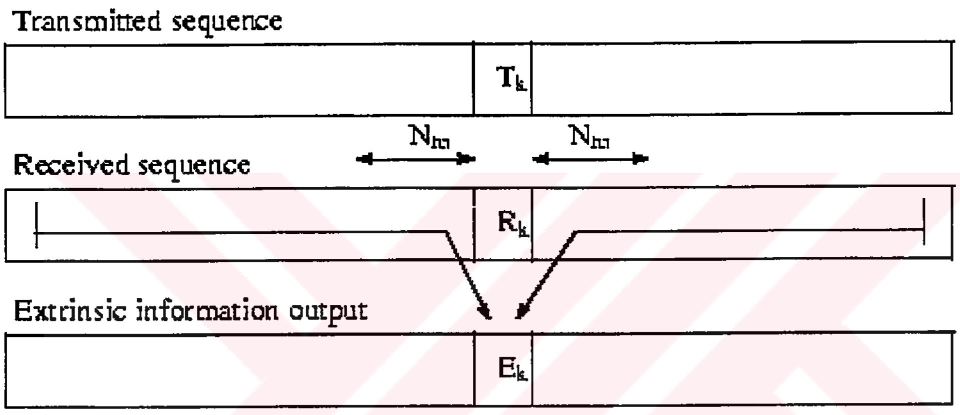


Figure 3.7 Extrinsic information for turbo codes

Once the above condition is fulfilled, to obtain a further improvement in the performance of turbo codes, a second limiting factor has to be considered: the degree of "randomness" with which the interleaver and the deinterleaver spread the bursts of errors from one decoder output to the next decoder input. From this point of view the ideal interleaver is a random interleaver.

From the above discussed types of interleaver designs it is clear that the role of the interleaver is to allow the decoders to make uncorrelated estimates of the soft values of the same information bit. The less "correlated" the two estimates are, the better the convergence of the iterative decoding algorithm.

3.3 DECODING

The optimum decoding of turbo codes is the maximum likelihood decoding algorithm applied to the turbo code trellis structure. However, due to the interleaver embedded in the turbo encoder, the turbo code trellis will have an extremely large number of states. This fact makes the maximum likelihood decoding process almost impossible to implement in practice for large interleaver sizes.

A more practical approach is an iterative decoding approach where the maximum likelihood decoding algorithm is applied to the elementary convolutional/block codes of the turbo code.

This iterative process raises the question of how close its performance is to the performance of the optimum decoding process. It was shown that turbo codes are near-optimal codes (Dolinar et al, 1998). Simulation results based on iterative decoding approach within 0.7 dB of the sphere-packing lower bound for various code rates and information block size from 100 to 10,000 bits. Therefore, this iterative technique is a very efficient way to decode turbo codes and to achieve performance close to the theoretical limits.

The second question related to the iterative process is how to pass the information from one decoder to the other. We will answer this while discussing the turbo decoder structures in the following sections. However, it is worth mentioning here that due to the iterative process, some parameters can grow to infinity unless special care is taken in the decoding algorithm. Limitations imposed on these parameters can have negative effects when very low BERs are simulated. Another cause for the limitation in BER performance is a poor interleaver design. Due to highly correlated sequences, the BER decreases to a certain level from where there is no further improvement in the decoding process. This is called the "error floor" effect of the turbo code BER curve.

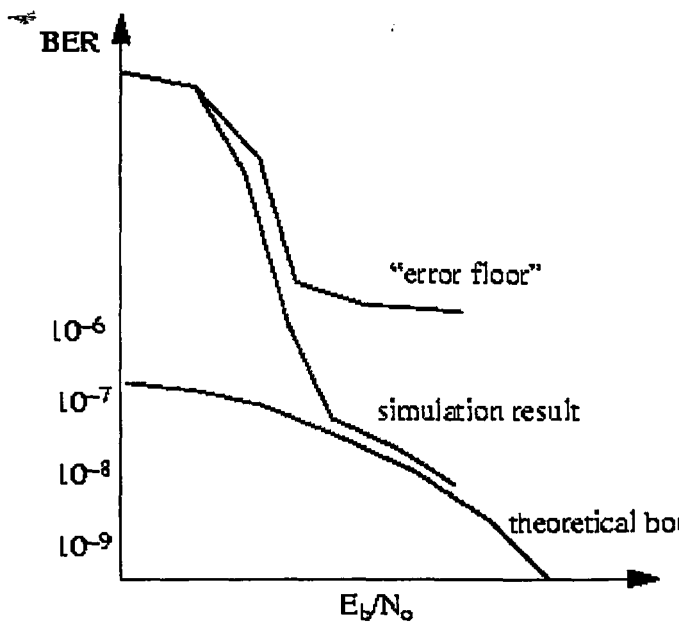


Figure 3.8 : Typical BER curve for turbo codes [Hall & Wilson, 1998).

Figure 3.8. shows three curves. The "error floor" curve is typical of bad interleaver design or truncations in the decoding algorithm. The simulation result curve also presents a change in slope but at very low BERs, when the theoretical bound for the turbo code is reached.

In the original paper on turbo codes, Berrou proposed an iterative decoding scheme based on an algorithm originally presented by Bahl in 1974 (Frey & Kscchischang, 1998). The Bahl decoding algorithm (it is also called as the BCJR algorithm in the memory of its discoverers) differs from the Viterbi algorithm in the sense that it produces soft outputs. While the Viterbi algorithm outputs either a 0 or 1 for each estimated bit, the Bahl algorithm outputs a continuous value that weights the confidence or log-likelihood of each bit estimate. While the goal of the Viterbi decoder is to minimise the code word error by finding a maximum likelihood (ML) estimate of the transmitted code word, the Bahl algorithm attempts to minimise bit error by estimating the a posteriori probabilities (APP) of the individual bits of the code word.

When used in an iterative fashion, we may call the Bahl algorithm as a MAP algorithm. Rather then the details of the Viterbi algorithm, we will explain first the idea of soft output algorithms and then the basics of the Bahl algorithm before getting into the decoder structure in the next section.

3.3.1 SISO ("Soft Input Soft Output") Algorithms

A SISO algorithm is basically defined as one that accepts *a priori* information at its input and produces *a posteriori* information at its output. Suppose the probability of a bit being a 1 is p(1) = 0.4 and that of 0 is p(0) = 0.6. Now, these probabilities are called soft-values. If, using these probabilities, we decide that the bit is 0 (since p(0) > p(1)), then we are essentially making a hard-decision of the information bit. The information is no longer "soft". Another representation of the soft-value of a bit is its actual decimal value (say, between 0 and 1). Suppose the bit received from the channel is 0.623, we can intuitively say that the bit is more likely to the 1 than 0.

From the above examples, we can intuitively feel that using the soft-values provide more information to the Channel Decoder, than a hard-decision bit. Following the same principle, we can also intuitively feel that algorithms using soft-values have a better performance than those which use hard-values (with an added complexity, of course), since they use the side-information generated by the channel.

Consider the simple case of binary (0's and 1's) transmission. Now, in the case of Hard Input (HI) Channel Decoder, the Demodulator will do a "hard decision" on the data it received. (i.e.) it decides whether the bit it received is a 1 or a 0 and sends only the demodulated bits to the Channel Decoder. The algorithm used in the Channel Decoder is called a Hard Input (HI) algorithm. In such algorithms there is a loss of information because the "side-information" or the "soft-values" generated by the channel is ignored. If soft-values, instead of hard-ones were used in the Channel Decoder, there will be an improvement in the BER performance, due to the additional information.

The same argument applies to the output of the decoder. The decoder can output either soft-values (probabilites or quantized decimal values). The decoder that accepts soft-inputs from the demodulator and outputs soft-values is called a Soft-Input, Soft-Output or a SISO decoder and the corresponding algorithm used in the decoder is a SISO algorithm.

The *a-priori* information is the soft inputs to the channel decoder and *the a-posteriori* information is the soft-output of the channel decoder.

The most widely used soft-values are the log-likelihood ratios (LLR's). These are represented as follows

$$\wedge_i = \ln \left(\frac{\mathrm{P}\ [m_i = 1 \mid \hat{y}]}{\mathrm{P}\ [m_i = 0 \mid \hat{y}]}\right) \tag{3.2}$$

If the LLR of a bit is positive, it implies that the bit is most likely to be a 1 and if it is negative, the bit is most likely to be a zero. You can also make a hard-decision on these ratios such that

$$b(i) = \begin{cases} 1, \text{ if } \wedge_i > 0 \\ 0, \text{ if } \wedge_i < 0 \end{cases} \tag{3.3}$$

3.3.2 MAP (“Maximum A Posteriori Probability”) Algorithm

The followings are the key terminology used in the explanation of the MAP algorithm (Barbulescu & Pietrobon, 1994):

-s_i represents the state of the trellis at the time i. It can be any of the 2^m possible states. (m = Constraint Length -1).

-L represents the state of the decoder

-y and "ŷ" represent the received observation vectors. It is the output of the demodulator. It consists of the sequence y(0), y(1) , y(2) .. y(L-1), where L is the length of the Convolutional Coded block and the values of y are the soft-outputs of the previous block.

-z_i is the a-priori information to the decoder. This is the output of the previous channel decoder block. It is in the form of a LLR. These LLR's must be converted into proper values before processing.

-m_i is the message bit associated with the state transition s_i to s_{i+1}.

-x_i is the output bit associated with the state transition s_i to s_{i+1}.

-P[x] represents the probability of x and P[y/x] is the probability of y conditioned on x.

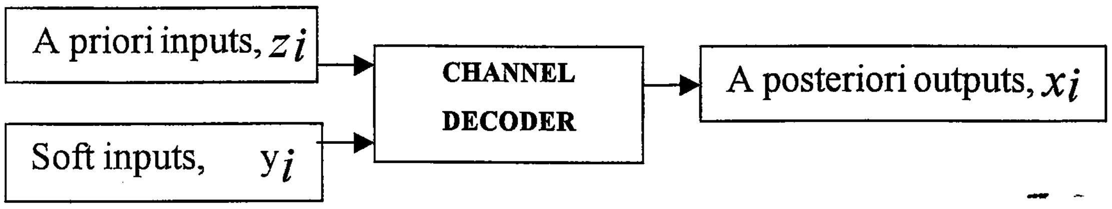


Figure 3.9 : Channel Decoder Structure [27].

The MAP algorithm proceeds as follows :

First of all, the MAP algorithm finds the probability $P[s_i \to s_{i+1} | y]$ of each valid state transition (in the Trellis diagram) given the noisy observation vector y. From, the definition of conditional probabilities;

$$P[s_i \to s_{i+1}, \hat{y}] = P[s_i \to s_{i+1} | \hat{y}]\, P[\hat{y}] \tag{3.4}$$

which implies that,

$$P[s_i \to s_{i+1} | \hat{y}] = \frac{P[s_i \to s_{i+1}, \hat{y}]}{P[\hat{y}]} \tag{3.5}$$

As the state machine that represents a Convolutional Code is a Markov Process and these Markov Processes have their own properties, we may partition the numerator as follows :

$$P[s_i \to s_{i+1}, \hat{y}] = \alpha(s_i)\, \gamma(s_i \to s_{i+1})\, \beta(s_{i+1}) \tag{3.6}$$

Here the term "alpha" represents the probability that the current state is s_i given the noisy observation vector y(0),y(1) ... y(i-1). This is because, the current state doesn't depend on the future values of the observation vector. The final state s_{i+1} is not at all considered here. Then we may write :

$$\alpha(s_i) = P[s_i, (y_0, \ldots y_{L-1})] \tag{3.7}$$

On the other hand;

$$\gamma(s_i \rightarrow s_{i+1}) = P[s_{i+1}, y_i \mid s_i] \tag{3.8}$$

represents the probability of the state transition from s_i to s_{i+1}, given the current state is s(i). The phrase in block letters is important, as it establishes the difference between "gamma" and the actual P[$s_i \rightarrow s_{i+1} \mid \mathbf{y}$]. "Gamma" is also called the branch metric associated with the state transition $s_i \rightarrow s_{i+1}$.

$$\beta(s_i) = P[(y_{i+1}, \ldots y_{L-1}) \mid s_{i+1}] \tag{3.9}$$

Finally the third term in our numerator represents the probability that the final state is s_{i+1}. The starting state s_i is not considered here, at all.

The branch metric can also be represented as follows by applying again the Markov principles :

$$\gamma(s_i \rightarrow s_{i+1}) = P[s_{i+1} \mid s_i]\, P[y_i \mid s_i \rightarrow s_{i+1}] = P[m_i]\, P[y_i \mid x_i] \tag{3.10}$$

where m_i and x_i are the message and the output (respectively) associated with the state transitions $s_i \rightarrow s_{i+1}$. For the above equation to be true, the states s_i and s_{i+1} should be connected in the trellis. Otherwise the transition probability of such a transition is (obviously) zero.

The term P[m_i] in the above equation, represent the *a-priori* information to the decoder and can be calculated as follows :

$$P[m_i] = \begin{cases} \dfrac{e^{zi}}{1+e^{zi}} & \text{for } m_i = 1, \\ \dfrac{e^{zi}}{1+e^{zi}} & \text{for } m_i = 0, \end{cases} \tag{3.11}$$

where, z_i ,as explained earlier, is the *a priori* information to the decoder. This is in the form of Log Likelihood ratios and these *a priori* information are converted into probabilities using the above formula.

The second term in the same equation, P[y_i |x_i], is easily calculated as follows :

$$P[y_i|x_i] = \frac{1}{\sqrt{\Pi N_0/E_s}} \exp\left(\frac{-E_s}{N_0}\sum_{q=0}^{n-1}[y_i^q - a_i^{(q)}(2x_i^{(q)}-1)]^2\right). \tag{3.12}$$

It is a function of the modulation and the channel characteristics. For a flat-fading channel the following expression of P[y_i |x_i] holds.

One can recognise that this is the Euclidean distance equation between two signal vectors in the n-dimensional signal space.

Turning back to the calculation of the parameters α, β, and γ, we may write the following forward and backward recursive equations by looking at the probability equations as follows :

$$P\,[s_i, (y_0, \ldots y_{L-1})] = \sum P[s_{i-1}, (y_0, \ldots y_{L-1})] \,.\, P[s_i, y_{i-1} \mid s_{i-1}] \tag{3.13}$$

$$\alpha\,(s_i) = \sum \alpha\,(s_{i-1})\,\gamma\,(s_{i-1} \rightarrow s_i) \tag{3.14}$$

In these equations the summations are taken over the set of states s_{i-1} , that are connected to the state s_i . Also the probability β (s_i) can be calculated as

$$\beta(s_i) = \sum \beta(s_{i+1})\, \gamma(s_i \rightarrow s_{i+1}) \tag{3.15}$$

where the summation is taken over the states s_{i+1} , that are connected to the state s_i .

Once the *a posteriori* probability of each possible state transition, P[$s_i \rightarrow s_{i+1}$| y] is calculated, the message bit probabilities at each time instant “i“ are calculated as

$$P[\,m_i = 1 \mid \hat{y}\,] = \sum_{S_1} P[\,s_i \rightarrow s_{i+1}, \hat{y}\,] \tag{3.16}$$

and,

$$P[\,m_i = 0 \mid \hat{y}\,] = \sum_{S_0} P[\,s_i \rightarrow s_{i+1}, \hat{y}\,] \tag{3.17}$$

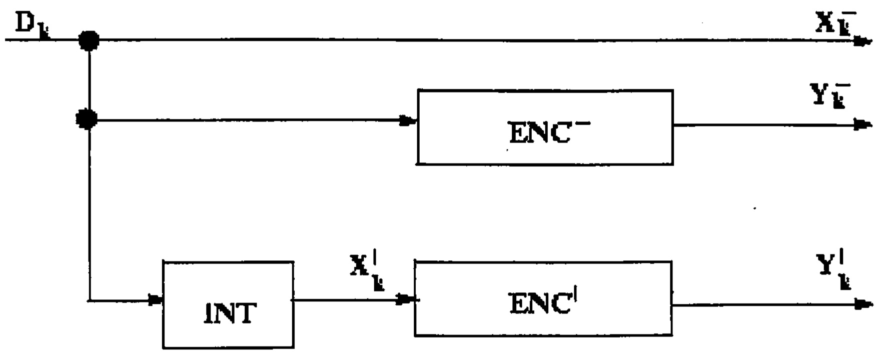
where S_1 represents the set of all state transitions for which the input message bit is “1” and S_0 represents all the state transitions for which the input message bit is “0”. The LLR now becomes:

$$LLR_i = \ln\left(\frac{\sum_{S_1} P[\,s_i \rightarrow s_{i+1}, \hat{y}\,]}{\sum_{S_0} P[\,s_i \rightarrow s_{i+1}, \hat{y}\,]}\right. \tag{3.18}$$

3.3.3 Turbo Decoder Structure

In this section we will examine the implementation of the MAP algorithm to the turbo decoder structure having iterative decoding process.

Consider the generic turbo encoder in Figure 3.10. The outputs of the turbo encoder are the information sequence D_k ,renamed as X_k^- together with the corresponding parity sequence Y_k^- produced by the encoder block ENC$^-$ and the parity sequence Y_k' produced by the encoder block ENC' at time k. These sequences are modulated and sent through the channel. The interleaved data sequence X_k' is not sent because it can be regenerated at the receiver by interleaving the received sequence corresponding to X_k^-. Upper case is used for the binary turbo encoder



outputs. Lower case is used for the noisy received signals at the turbo decoder.

Figure 3.10 : Generic Turbo Encoder [27].

We can shortly explain a single iteration within the decoder by considering Figure 3.11. In this figure the decoder DEC$^-$ provides a soft output which is a measure of the reliability of each decoded bit. From this reliability information, the extrinsic information is produced, which does not depend on the current inputs to the decoder. This extrinsic information, after interleaving, is passed on to DEC' which uses this information to decode the interleaved bit sequence. From the soft outputs of DEC', the new extrinsic information is fed back to the input of DEC$^-$ and so on.

If an error occurs at the output of the first decoder due to a very noisy input, it might very well be corrected by the second decoder.

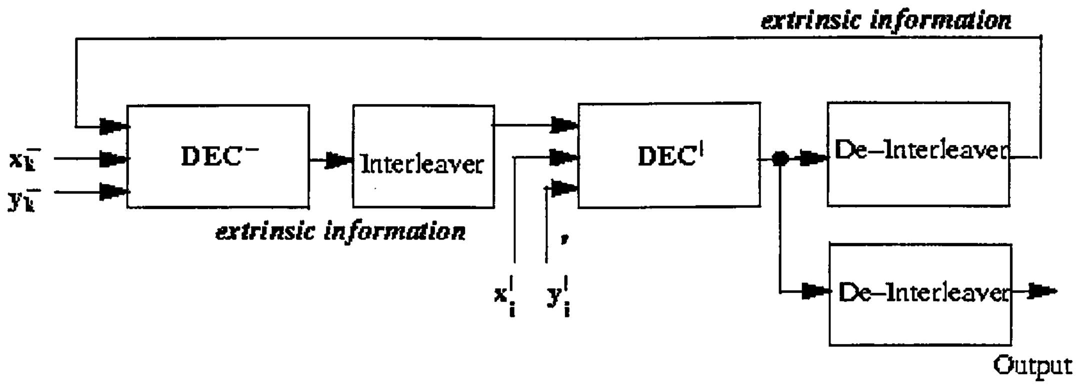


Figure 3.11 : Generic Turbo Decoder

Besides the MAP algorithm, there are also some other algorithms which are also suitable for the turbo decoders. The first one is the modifed version of the Viterbi algorithm, namely the Soft Output Viterbi Algorithm (SOVA). This algorithm requires less computational complexity, however especially in the case of low SNR performs worser than the MAP algorithm.

The performance of a turbo coding scheme improves as the number of decoder iterations is increased. However, the coding gain from one iteration to another, decreases with the number of iterations. Each iteration involves two decoding stages. Therefore, the overall complexity of a turbo decoder depends on how efficient the decoding algorithm is implemented.

In order to reduce the computational complexity of the turbo decoder, an early detection method was investigated (Franz & Anderson, 1998). Based on a confidence criterion, some information symbols, state variables and parity symbols are detected early during decoding. This early detection allows sections to be removed from the constituent trellises, resulting in spliced trellises that can be decoded faster and with a lower computational effort.

A reduced-search MAP algorithm was also tried where a reduction in computational complexity is achieved based on a threshold scheme (Barbulescu & Pietrobon, 1994). This scheme uses only the state metrics that are above a certain threshold in computing the bit estimates. However, the variable computational effort required might present difficulties for hardware implementation.

As the principles of iterative decoding are given (Barbulescu, 1996), (Sklar, 1997), (Comatlas, 1995), the detailed implementation of an iterative rate 1/2 decoder is described in Figure 3.12. This represents only one iteration of the decoding process. The whole block can be repeated for the number of desired iterations (less than 10 usually).

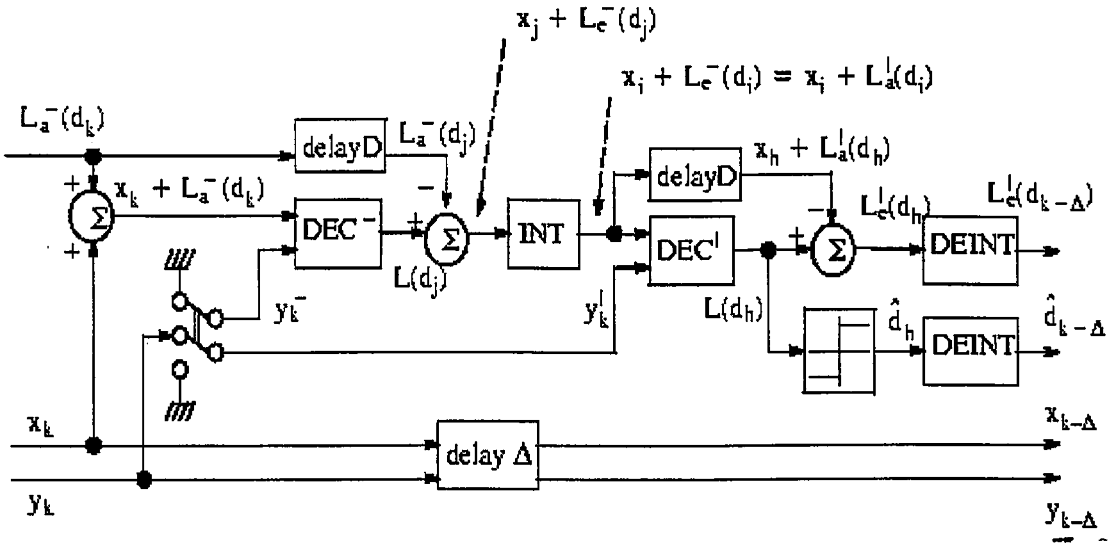


Figure 3.12 : Rate ½ Turbo decoder

For an AWGN channel x_k and y_k represent the received noisy values of X_k and Y_k, respectively, which are scaled by the reliability value of the channel, $L_c = 4\ E_s / N_0$ where $4\ E_s / N_0$ is the SNR. For example, if the binary value X_k is transmitted, the x_k value used by the decoder is :

$$x_k = 4\ E_s / N_0\ (1 - 2\ X_k + n_k) \tag{3.19}$$

where n_k is the noise sample at time k.

A turbo decoder starts by first decoding the data transmitted by X_k and Y_k^-. The input to the first decoder must then be x_k and y_k^-, the punctured version of y_k.

Figure 3.12 shows the inputs to the first decoder (DEC⁻) as $x_k + L_a^-(d_k)$ and y_k^-. The input $L_a^-(d_k)$ represents the a priori information about the transmitted information bit d_k . In the first decoding stage, we know nothing about d_k , and so $L_a^-(d_k)$ is set to zero. A positive value of $L_a^-(d_k)$ would indicate that $d_k = 0$ was likely to have been sent, while a negative value of it would mean that $d_k = 1$ was likely to have been sent.

It can be shown that the output of DEC^{-} is equal to

$$L(d_j) = x_j + L_a^{-}(d_j) + L_e^{-}(d_j) \tag{3.20}$$

where the subscript j = k-D, D is the delay of the decoder and $L_e^{-}(d_j)$ is the extrinsic information of d_j. We can consider $L_e^{-}(d_j)$ being a correction term produced by the decoder to correct any errors on the channel.

In Figure 3.12 we see that $L_a^{-}(d_j)$ is subtracted from $L(d_j)$ to give $x_j + L_e^{-}(d_j)$. Since $L_a^{-}(d_j) = 0$ for the first iteration, this has no effect on later decodings, however as we will see this a very important computation.

We now come to a very important block of a turbo decoder, the interleaver. A normal decoder will try to find the closest code sequence to the received noisy sequence that minimises some probability of error criteria. A good code has its code sequences separated by a large number of bits. Thus, it takes a large amount of noise in a short period of time to cause a decoder to make an error. Therefore, a decoder will produce its errors in bursts, instead of randomly as in an AWGN channel. The interleaver is used to randomise the burst errors from DEC^{-}. The larger the interleaver, the better that the error bursts can be de-correlated.

After interleaving, the data becomes $x_j + L_e^{-}(d_i)$ which now correspond to the data that is input to DEC'.

The extrinsic information from the first decoder can be thought of *a priori* information for the second decoder. So we let $L_e^{-}(d_i) = L_a'(d_i)$ and feed in $x_i + L_a'(d_i)$ and y_k', the punctured version of y_k corresponding to Y_k', in DEC' is

$$L(d_h) = x_h + L_a'(d_h) + L_e'(d_h) \tag{3.21}$$

where $h = i - D$ and $L_e'(d_h)$ is the new extrinsic information provided by DEC'. We can make a hard decision on $L(d_h)$ and output the decoded data.

We can now use the new information from DEC' and feed it into the next DEC$^-$. To do this we want to obtain the a priori information from $L(d_h)$. We see from Figure 2.12 that we subtract $x_h + L_a'(d_h)$ from $L(d_h)$ to leave only $L_e'(d_h)$. We then deinterleave $L_e'(d_h)$ to form $L_e'(d_{k-\Delta}) = L_a^-(d_{k-\Delta})$ where Δ is the total delay of DEC$^-$, DEC', INT and DEINT. $L_a^-(d_{k-\Delta})$ then becomes the *a priori* information for the next DEC$^-$ and the decoding process starts over again.

It should be noted that we do not include $L_a'(d_h) = L_e^-(d_{i-D})$ in $L_a^-(d_{k-\Delta})$. This is because after deinterleaving, $L_e^-(d_{i-D})$ becomes $L_e^-(d_{k-\Delta})$. Thus any burst errors in $L_e^-(d_{k-\Delta})$ would now reappear which we should avoid feeding into DEC$^-$. However any error bursts in $L_e'(d_h)$ will be de-correlated by DEINT. This is also the reason why we subtract $L_a^-(d_j)$ from $L(d_j)$ (Since INT will re-correlate the error bursts in $L_a^-(d_j)$).

The above process is iterated many times until eventually all the errors are corrected, or there remains an error pattern that can not be corrected despite the interleaving and deinterleaving processes.

CHAPTER FOUR

SYSTEM MODEL

In many wireless environments, the channel is modelled as time varying due to the effect known as fading. Fading results when the physical nature of the channel causes dispersion of the transmitted signal. This fading depends on the nature of the transmitted signal with respect to the characteristics of the channel. Typical sources of channel dispersion include multipath propagation, atmospheric absorption or ionospheric scintillation. In most realistic situations, dispersion occurs both in frequency and in time.

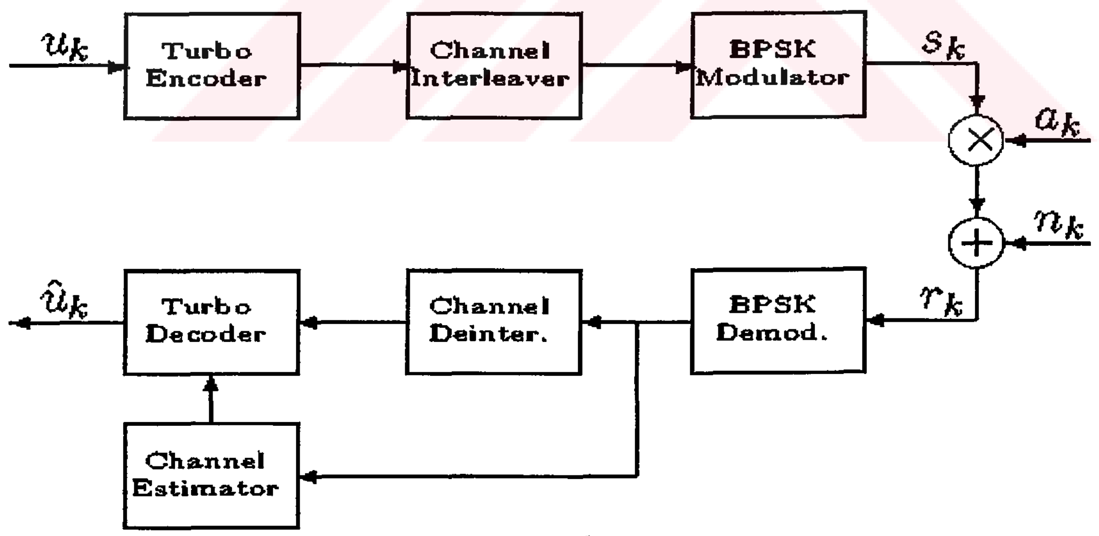


Figure 4.1 : System model block diagram

Time dispersion due to multipath causes the transmitted signal undergo either a flat or frequency selective fading. On the other hand, Doppler spread leads to fast or slow fading which is the dispersion in frequency.

The distribution of the instantaneous gain of flat fading channels is important for designing radio links and the most common amplitude distribution is the Rayleigh distribution.

As shown in Figure 4.1, in our system model the channel is Rayleigh flat fading and assumed to have slow fading which is the most common case in wireless communications.

In the following sections we will examine shortly the properties of Rayleigh slow fading channels and the simulation of the channel parameters namely by making use of the Jakes' isotropic scattering model.

4.1 RAYLEIGH SLOW FADING CHANNELS

In channels which are only dispersive in frequency, the channel acts to amplitude and phase modulate the transmitted signal by a channel function, a(t). In this way, the channel selectively alters certain time segments of the transmitted waveform. From this description, channels dispersive in frequency are often referred to as time-selective. However, a more common term is frequency-flat or simply flat-fading channels. The term flat-fading comes from the fact that all frequencies of the transmitted signal are modulated by the same function (Kennedy, 1969).

The fading function or channel gain is described by a probability density function and a frequency dispersion measure, B, often referred to as the Doppler spread (or bandwidth). If the Doppler spread is small compared to the reciprocal of the symbol rate, T_s^{-1} , the fading process is considered slow. For slow fading processes, the channel gain can be assumed constant over the symbol duration. Throughout this work, slow fading will be assumed.

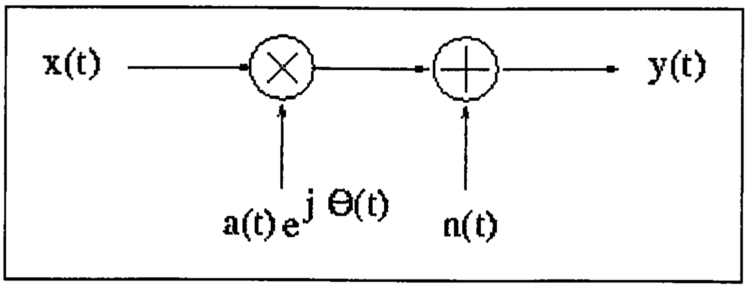


Figure 4.2 : Slow, flat-fading model with additive noise

Figure 4.2 shows the model for the fading process. In this model, BPSK signaling is assumed with coherent detection which implies that θ(t) is known at the receiver. Furthermore, aside from the channel gain, additive noise (n(t)) is present in the system. This additive noise will be assumed to be white Gaussian noise (AWGN) with zero mean and power spectral density, $N_0/2$ (W/Hz). The presence of the AWGN following the fading process is due to the fact that its primary source is the front-end electronics of the receiver (Schwartz et al, 1966).

In the literature, there are many models for the channel gain a(t) based on empirical and analytical study of real channels. However, in this work, the Rayleigh fading model will be used. The Rayleigh model represents a worst case model for the design engineer while offering analytical convenience. The Rayleigh model arises from the combination of many point scatterer contributions at the receiver. Each of these scatterers has only a small fraction of the overall received energy. The Rayleigh fading model can be justified using the Central Limit theorem hypothesis (Wilson, 1996).

Under the above mentioned assumptions, appropriate demodulation and sampling give our system model the discrete representation :

$$y_k = a_k \cdot x_k + n_k \tag{4.1}$$

In this representation, x_k is a BPSK symbol amplitude having values of $\pm\sqrt{E_S}$, where E_S is the received energy per symbol, and n_k is an AWGN component with zero mean and variance $N_0/2$. The channel gain a is described by a probability density function and an autocorrelation function. As previously mentioned, the channel gain is Rayleigh distributed with probability density function.

$$p_a(a) = 2a\,e^{-a^2}, a \geq 0. \tag{4.2}$$

The channel gain is scaled to have a mean-square value of $E_a[a^2] = 1$ indicating that the expected received power will be E_S .

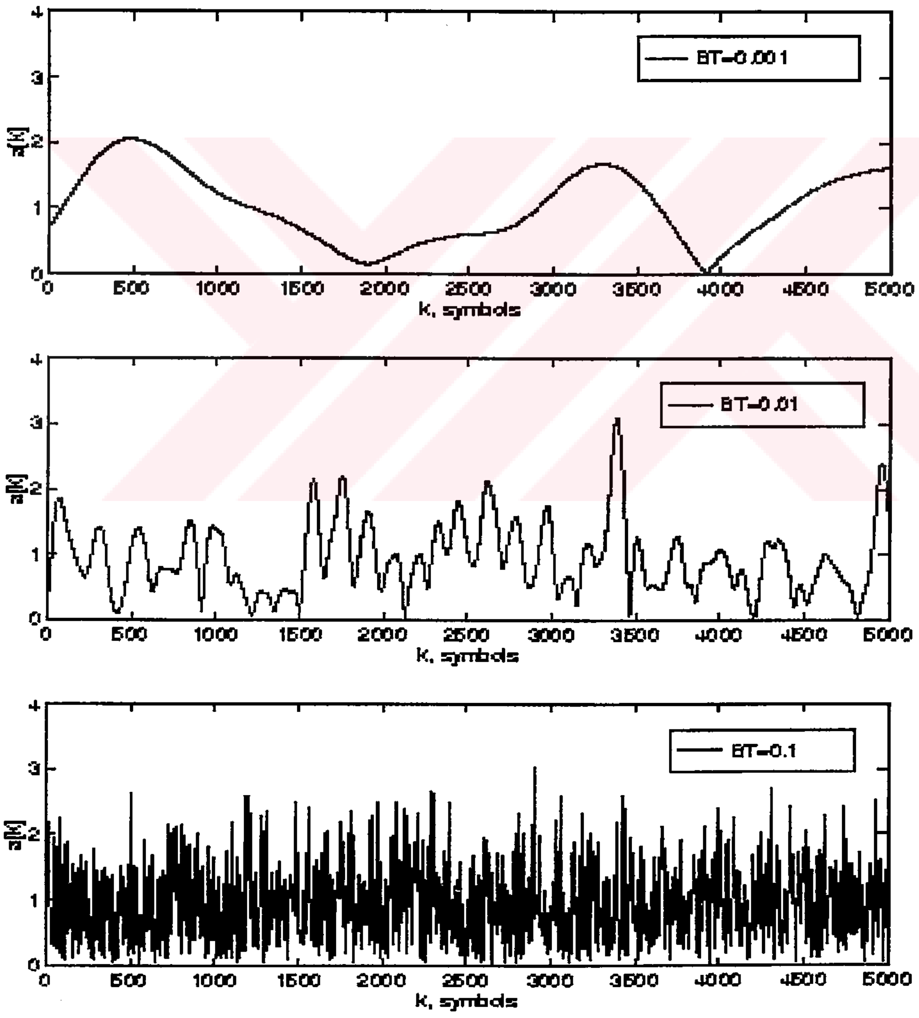


Figure 4.3 : Fading Trajectories

The auto-correlation function, $\rho(\tau)$, for a(t) reflects the time correlation between channel gains in realistic flat-fading channels. This function is usually characterized by a low-pass spectrum with one-sided Doppler bandwidth, B. In the discrete model, sampling of the fading process is done at the symbol rate, T_s^{-1}, of the link.

Therefore, flat-fading channels are often characterized by a normalized spectrum based on the Doppler bandwidth, symbol duration product, BT_s. For example, a cellular link sending 10ksymbols/sec through a fading channel with a Doppler bandwidth of 10Hz would have a Doppler bandwidth, symbol duration product of 0.001. In Figure 4.3, sample functions of fading processes with differing values of $BT_s = BT$ are shown. These trajectories are based on a fourth-order Butterworth approximation to the ideal low-pass spectrum having a normalised bandwidth of BT_s. Note that our slow-fading assumption requires that BT_s be much less than 1.

4.2 CHANNEL SIMULATOR

The simulation of the Rayleigh fading channel can best be done by making use of the Jakes' model (Jakes, 1974). The block diagram can be seen in Figure 4.4.

In this model, there are N_c low-frequency oscillators with frequencies equal to the Doppler Shifts $W_m \cos(2\pi n/N)$, $n = 1,2,\dots N_c$, plus one with frequency W_m , where $N = 2*(2*N_c + 1)$. These oscillators are used to generate signals frequency-shifted from a carrier frequency W_c. The amplitudes of all components are made equal to unity except for the one with frequency W_m , which is set to $1 / \sqrt{2}$. The phases b_n are chosen appropriately so that the probability distribution of the resultant phase will be as close as possible to uniform distribution, $1 / 2\pi$.

The various frequency components are scaled by amplifiers with gains set equal to 2 cos b_n or 2 sin b_n. The outputs of the individual oscillators, with the appropriate gain factors, are first summed to produce in-phase (x_c) and quadrature (x_s) bands, which are then multiplied by in-phase and quadrature carrier components, respectively, and then summed to produce the final composite output signal y(t)..

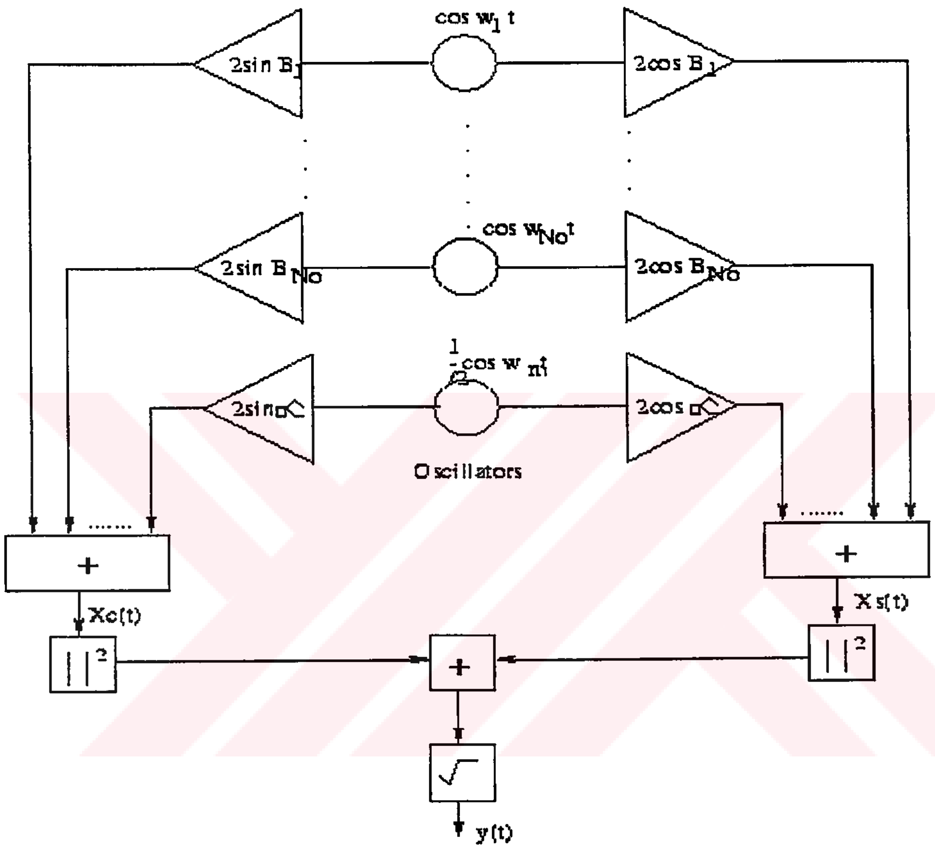


Figure 4.4 Block Diagram of a Rayleigh Fading Generator

$$y(t) = x_c(t) \cos W_c t + x_s(t) \sin W_c t \qquad (4.3)$$

where $x_c(t)$ and $x_s(t)$ are approximately Gaussian random processes. The signal envelope of y(t) follows a Rayleigh distribution. Also y(t) is a narrow-band signal centered on a carrier frequency W_c, having Rayleigh fading characteristics, and with autocorrelation function approximately equal to J_0 (W_m t).

By making one of the offset oscillator outputs dominant in amplitude, the fading can also be changed from Rayleigh to Ricean.

4.3 SYSTEM MODEL BLOCK DIAGRAM

4.3.1 General Process

The system model which is shown in Figure 4.1, is based on the model used in the Matthew Valenti's study regarding the variable latency of turbo codes (Valenti, 1997). Information bits u_k are passed through the turbo encoder, interleaved, modulated using a Binary Phase Shift Keying (BPSK) modulator, and sent over a flat fading channel.

The flat fading channel multiplies each symbol s_k at the output of the modulator by a fading amplitude a_k . A sample n_k from a white Gaussian noise process with double-sided power spectral density (PSD) of $N_0 / 2 = \sigma_n^2$ is added to the faded symbol and passed to a BPSK demodulator. The soft output from the BPSK demodulator is deinterleved and passed to the turbo decoding algorithm which produces estimates $\hat{u}_k$ of the data.

The turbo decoding algorithm uses the Log-MAP algorithm (Robertson et al, 1997). In addition, the BPSK demodulator outputs are sent to a channel estimator, which provides the turbo decoding algorithm with estimates of the fading amplitudes.

The process that generates the fading amplitudes is a critical factor in the system model. It was suggested that if the system has sufficient interleaving, then the fading amplitudes are statistically independent (Jung, 1996). In our system model we make no such assumption and model fading using a correlated fading process.

The fading amplitude a_k is the magnitude of a complex gain and may be represented as :

$$a_k = \left| (\alpha + x_k) + j(y_k) \right| \tag{4.4}$$

where α represents the amplitude of the specular component, and x_k and y_k are samples of zero mean stationary Gaussian random processes each with variance σ_f^2 (Kaplan & Shamai, 1994).

Since the two jointly Gaussian processes are independent, $E[x_k\, y_k] = 0$. However, each process is self-correlated, and the particular autocorrelation function used is an important consideration. We use the following autocorrelation, which is the consequence of the Jack's model

$$R(k) = \frac{1}{\pi B T_S} J_0(2\pi B T_S k) \tag{4.5}$$

where $J_0(.)$ is the zero-order Bessel function of the first kind.

4.3.2 Channel Interleaving for Correlated Rayleigh Fading

The channel interleaver is required because turbo encoding may not be able to cope with the errors induced by the fading channel. Correlated fading channels tend to produce burst errors, while turbo codes are more effective with uncorrelated errors.

Therefore the channel interleaver scrambles the order of the symbols at the transmitter in order to make the channel appear uncorrelated at the input to the decoder. Note that the channel interleaver is different from the nonuniform interleaver required within the turbo encoder. To distinguish these two interleavers, we refer to the interleaver within the turbo encoder as the coding interleaver and the interleaver between the encoder and modulator as the channel interleaver

4.3.3 Channel Estimator

Aside from the interleaver, the knowledge of the fading amplitudes is necessary to improve the decoding capabilities of the receiver. Because of the correlation in the channel, a reasonable estimate of the fading amplitudes can be obtained by passing the absolute value of the received signal r_k through an FIR filter. The estimate of the fading amplitude $\hat{a}_k$ is thus :

$$\hat{a}_k = \sum_{n=0}^{N} h_n \left| r_{k-n} \right| \tag{4.6}$$

where h_n , n = 0,1,...N is the impulse response of the FIR filter, which has order N.

CHAPTER FIVE

THE SIMULATION RESULTS OF THE INTRODUCED SYSTEM

Based on the system model block diagram in Figure 4.4, the performance of the turbo coded system was evaluated for both AWGN and Rayleigh channels. In the case of Rayleigh fading channel additional block interleavers were implemented and their effect was examined for different moving mobile speeds and hence for different Doppler frequencies.

In all simulations information bits are grouped into frames of 1024 bits and passed through a rate ½ (puncturing chosen), constraint length of three turbo encoders with the generator matrix $(5/7)_8$ and random coding interleaver. The code bits are interleaved using a 32 by 64 block channel interleaver, and then passed through Binary Phase Shift Keying (BPSK) modulator and sent to the channel. The turbo decoders make use of the Log-MAP algorithm with eight iterations.

In order to obtain reliable data, enough trails had been carried out for each of the different situations. The usual criterion was chosen as to count 32 frame errors to get reliable estimates of the BERs.

While studying the effect of different Doppler frequencies and hence different fade rates, slow fading was assumed, and two normalised fade rates are considered : $f_dT_s = 0.003$ and $f_dT_s = 0.01$. The slower rate corresponds to a typical digital cellular system operating at 900MHz with 19.2 kbaud symbol rate and a relative mobile velocity of approximately 70 km/h, while the faster code rate corresponds to a PCS system operating at 1.9 GHz with 9.6 kbaud symbol rate and approximate velocity of 56 km/h mobile speed.

5.1 THE EFFECT OF RAYLEIGH FADING ON THE PERFORMANCE OF THE TURBO CODES

In Figure 5.1 and in Figure 5.2 we have the results of our system for pure AWGN and for Rayleigh fading with AWGN channel types respectively. It is obvious that the performance is much more worse in the case of Rayleigh fading even for lower noise levels.

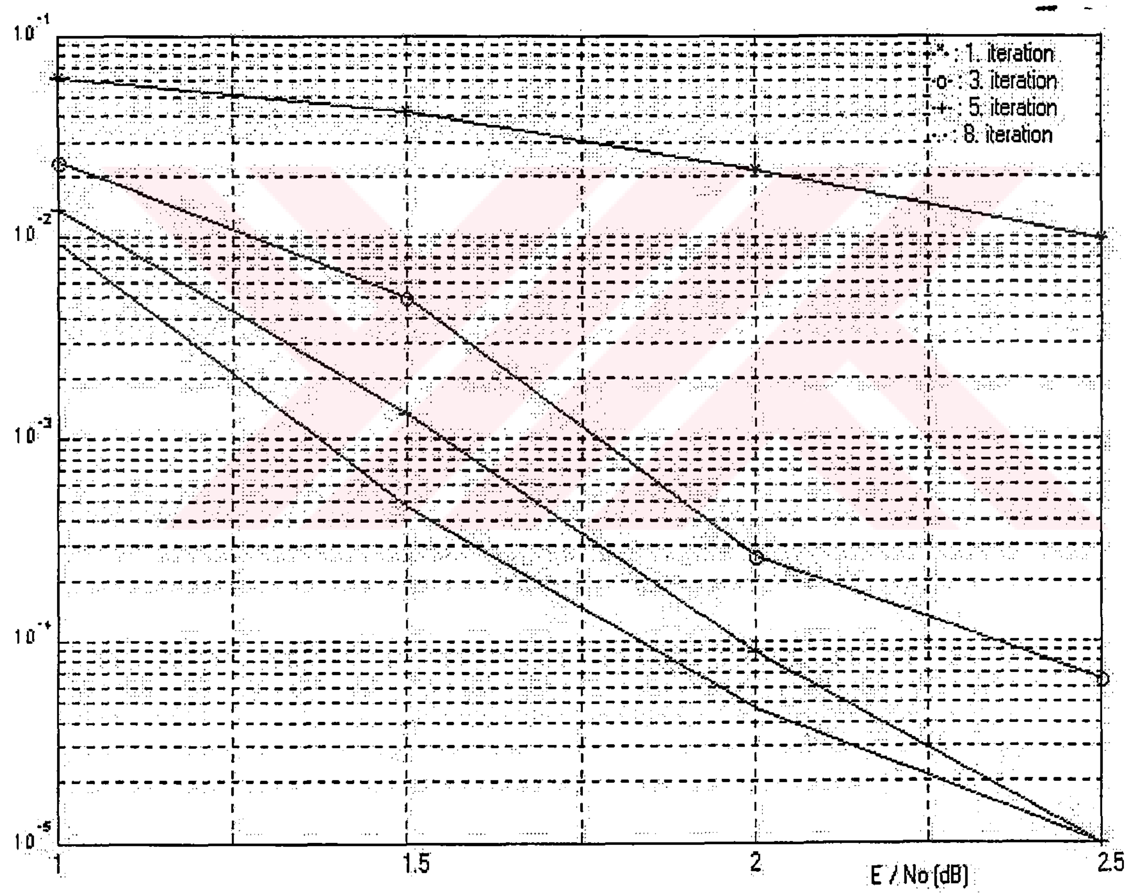


Figure 5.1 Performance curve for the pure AWGN channel.

In Figure 5.1 and also in the other performance curves of this chapters the BER curves were prepared based on the number of iterations. This also show us the BER performance improvement of turbo codes from one iteration to another (the four graphics in the figures are for the 1.,3.,5., and 8. Iterations).

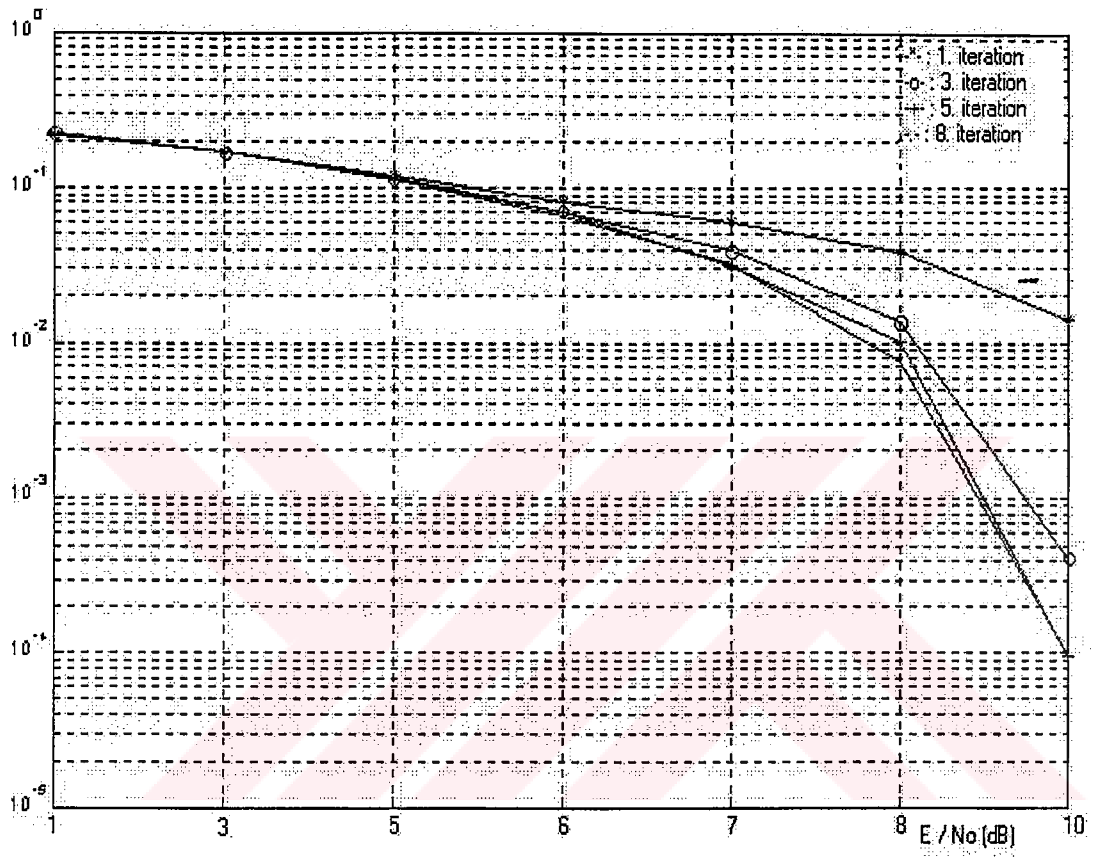


Figure 5.2 Performance curve for the Rayleigh Fading Channel with AWGN (f_dT_s = 0.01).

As it can be seen from the Figure 5.2 the effect of Rayleigh fading is so high that the iterative decoding shows its effect after a certain signal to noise ratio.

5.2 THE EFFECT OF ADDITIONAL INTERLEAVING ON THE PERFORMANCE OF TURBO CODES

As explained in section 4.3.2, the performance gets better when additional channel interleaving is used. In Figure 5.3 under the same conditions, we get much more lower BERs when compared with Figure 5.2.

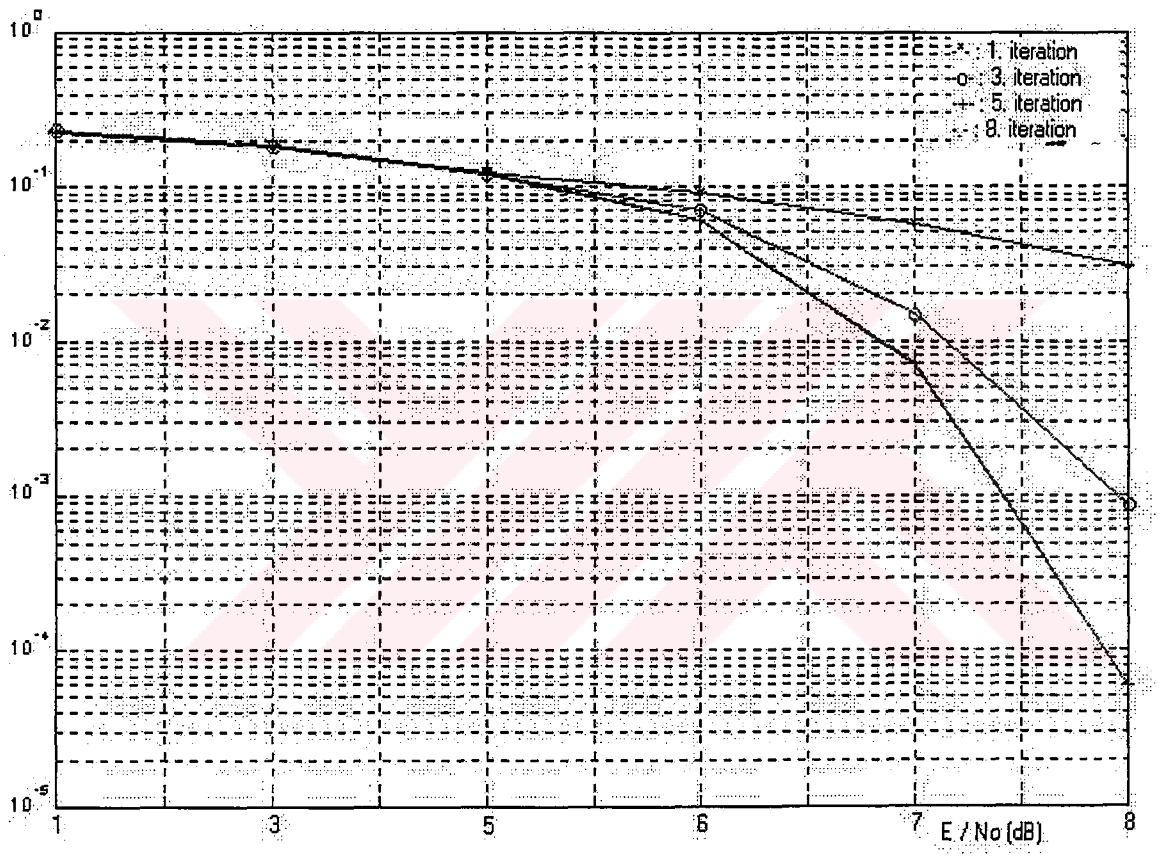


Figure 5.3 Performance curve for the Rayleigh Fading Channel with Channel Interleaving ($f_dT_s = 0.01$).

During the simulations it was also observed that with channel interleaving, the performance improves as f_dT_s decreases, which corresponds to a faster moving mobile or in another words more relative velocity between the transmitter and the receiver. In Figure 5.4 we have the performance curve of our system under same conditions but with a different fading bandwidth and symbol duration.

The Figure 5.3 and the Figure 5.4 depict that the performance loss due to correlation in the fading channel can be significant even when channel interleaving is used.

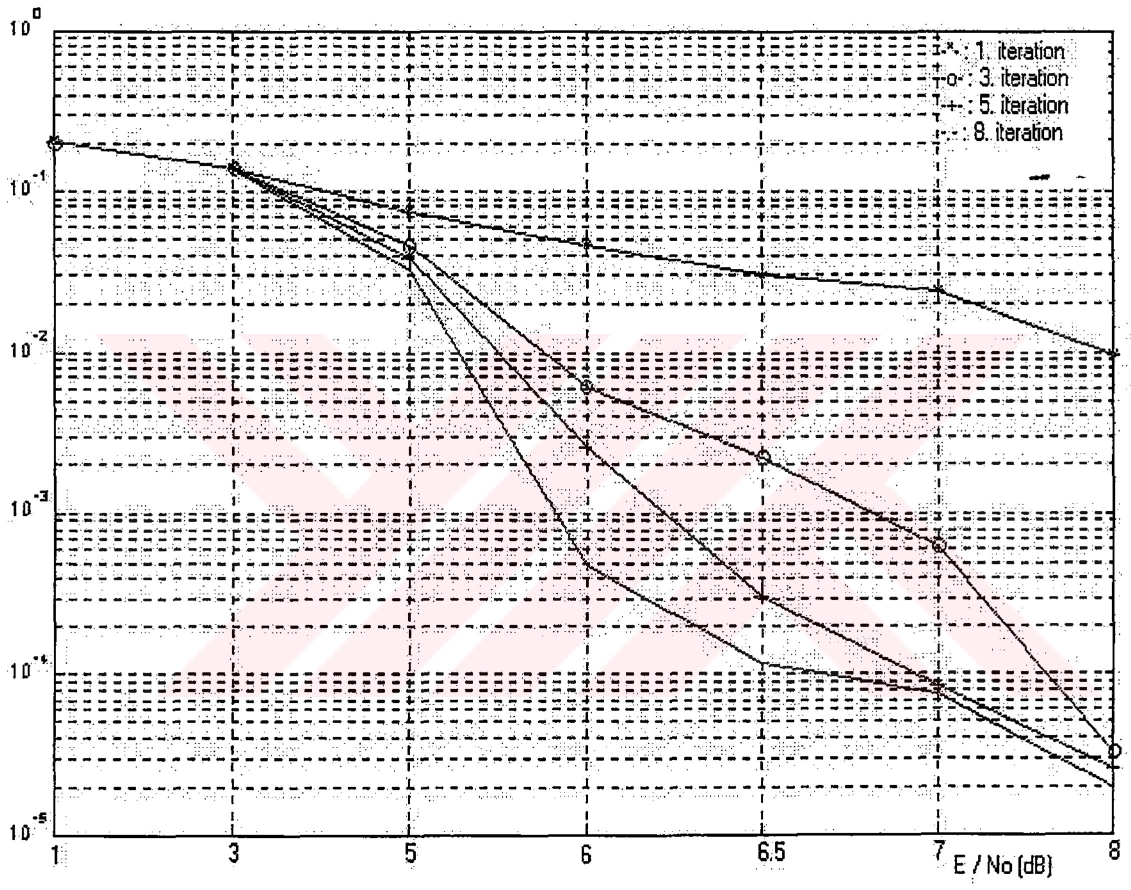


Figure 5.4 Performance curve for the Rayleigh Fading Channel with Channel Interleaving ($f_d T_s = 0.003$).

5.3 PERFORMANCE COMPARISON OF TRELLIS AND TURBO CODES OVER RAYLEIGH FADING CHANNELS

In Figures 5.5 and Figure 5.6, the performance of trellis codes over the frequency non-selective slowly Rayleigh fading channels are shown for different modulation types and for different channel bandwidth efficiencies.

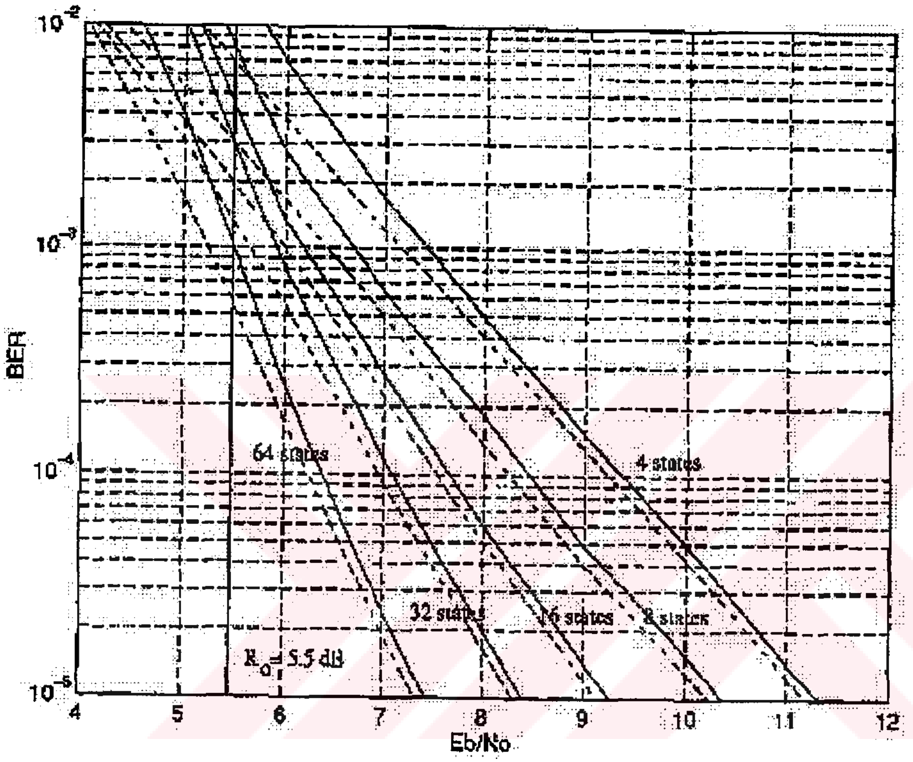


Figure 5.5 BER performance of 1 bit/sec/Hz. Trellis coded QPSK modulation.

As can be seen from Figure 5.5, the BER values of 10e-5 is reached with a 32 state encoder structure at an E/N$_o$ lower than the simulation results of turbo codes presented in the previous figures. However this 32 state encoder structure requires much more complex implementation schemes.

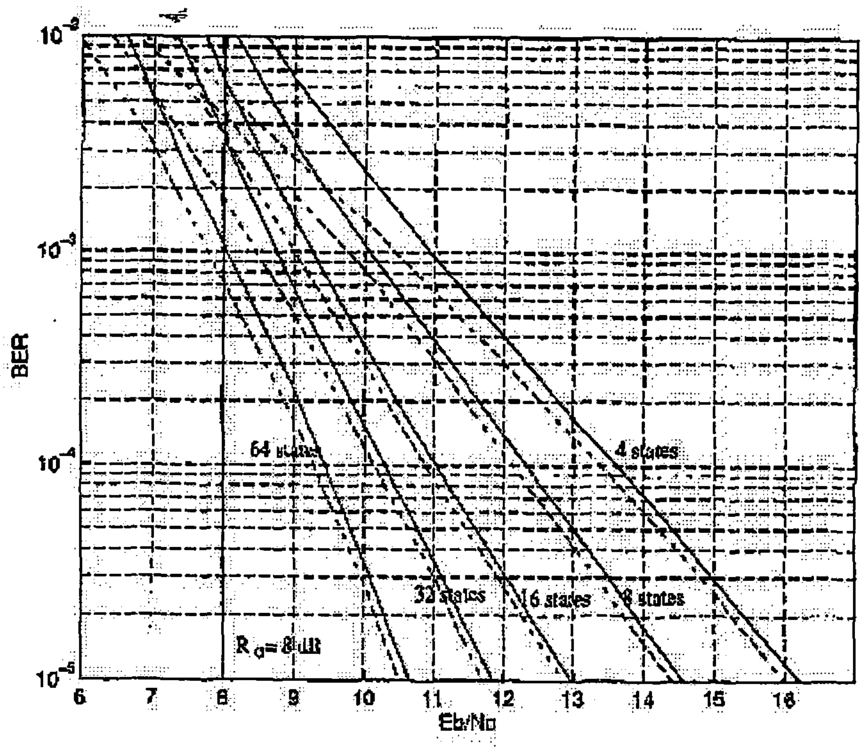


Figure 5.6 BER performance of 2 bits/sec/Hz. trellis coded 16-QAM.

CHAPTER SIX
CONCLUSIONS

The presence of fading in wireless environments increases the error probability and motivates the use of good channel codes. Channel codes help to mitigate this adverse effect on error performance by adding redundancy and memory to the transmission. In his study Claude Berrou introduced a new and exciting channel coding technique commonly referred to as turbo codes (Berrou et al, 1993). This new coding scheme was shown with sufficient block length to be capable of near-capacity performance on the AWGN channel. In this work, we investigated the performance of a typical turbo code transmitted over various channels using the Log-MAP decoding algorithm.

Wireless networks can experience a variety of fading conditions based on the physical nature of the channel. In this work, we have considered the frequency-flat, slow-fading Rayleigh channel where additive noise also exists. The channel models incorporate the concepts of correlated fading, channel interleaving, and the effect of different Doppler frequencies.

We showed that channel interleaving can be used to mitigate the detrimental effects of correlation in the channel. In our scheme, we have considered only interleavers with sizes matched to that of the codeword. This will confine the added latency to the transmitter. However, if latency is not a constraint, the use of a sufficiently large block interleaver should allow performance to approach that of the fully-interleaved channel.

It has been observed that the performance loss due to the correlation in the fading channel can be significant even when the channel interleaving is used. Thus we recommend that the effect of channel correlation and interleaving be taken into account when investigating the performance of turbo codes operating over flat fading channels.

1. Future Work

Beside the channel block interleaver, the effect of scrambling the systematic information before the construction of the turbo codeword may also be studied. This scheme will add no latency to the system, unlike that of our proposed channel block interleaver.

In practice the fading amplitudes of the channel are not known by the decoding algorithm and hence they should be estimated by the receiver. Because of the low signal to noise ratios typical of turbo coded systems, it is difficult to obtain perfect estimates of the fading amplitudes. Beside the simple low pass FIR filter proposed as an estimator in our study, the estimator structures can be optimised by making use of dynamic filters such as Kalman filters.

In addition to the different types of estimator filters, the pilot symbols can also be used at the receiver to obtain an estimate of the channel response so that coherent detection can be performed.

The usage of feedback path from the decoder to the estimator should also be explored. In that case, after each iteration of turbo decoding, the channel estimates will be refined using the information fed back from the decoder. In that way the channel will be re-estimated prior to the next decoder iteration after the re-insertion of the pilot symbols following the first estimated symbol sequence.

The receiver can also try to estimate the channel response in an iterative fashion. That is the decoder gives a decoded output and tries to reconstruct the transmitted sequence. Comparing this with the received sequence, the decoder tries to obtain a pdf for the fading amplitudes. The improved pdf is then used to better decode the data and so on. There are also various other equalising type techniques that should be explored.

REFERENCES

Berrou, C., Glavieux, A. & Thitimasjshima, P. (1993, May). Near Shannon limit error correcting coding and decoding : Turbo-codes. Proceedings of the IEEE International Conference on Communications, 1064-1070, Geneva, Switzerland.

Hagenauer, J., & Robertson, P., (1994, April). Iterative (turbo) decoding of systematic convolutional codes with the MAP and SOVA algorithms. Procedings of the ITG Conference, 21-29.

Jung, P., (1996, April). Comparison of turbo-code decoders applied to short frame transmission systems. IEEE Journal on Selected Areas in Communications, 14, 530-537.

Koorapaty, H., Wang, Y.E., & Balachandran, K., (1997, November). Performance of turbo codes with short frame sizes. Procedings of the IEEE Vehicle Technology Conference, 329-333.

Valenti, M.C., & Woerner, b.d., (1997, September). Variable latency turbo codes for wireless multimedia applications. Proceedings of International Symposium on Turbo Codes and Related Topics, 216-219.

Hall, E.K., & Wilson, S.G., (1996). Design and performance analysis of turbo codes on Rayleigh Fading channels. Procedings of CISS.

Hall, E.K., & Wilson, S.G., (1998). Design and analysis of turbo codes on Rayleigh Fading channels. IEEE Journal of Selected Areas of Communications, 16.

Jordan, M., & Nichols, R., (1996). The effects of channel characteristics on turbo code performance. Procedings of MILCOM, 17-21.

Forney, G.D. Jr., (1966). Concatenated Codes. MIT Press, Cambridge, Mass.

Pottie, G.J., & Taylor, D., P., (1977, September). Multilevel codes based on partitioning, IEEE Transactions on Information Theory, 23, 87-98.

Bahl, L., Cocke, J., Jelinek, F. & Raviv, J., (1974, March). Optimal decoding of linear codes for minimising symbol error rate, IEEE Transactions on Information Theory, 20, 284-287.

Pietrobon, S.,S. & Barbulescu, S.,A (1994, November). A simplification of the modified Bahl decoding algorithm for systematic convolutional codes, ISITA, NSW, 1073-1077.

Viterbi, A., J., (1967, April). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Transactions on Information Theory, 13, 260-269.

Hagenauer, J., (1991, September). Source-controlled channel decoding. IEEE Transactions on Communications, 43, 2449-2457.

Shannon, C.,E., (1948). A mathematical theory of communication. Bell Syst. Tech. J., 379-423 (pt. I); 623-656 (pt. II).

Couleaud, J., Y., (1995, September). High gain coding schemes for space communications. ENSICA Final Year Report, University of South Australia.

Benedetto, S. & Montorsi, G. (1996, June). Iterative decoding of serially concatenated convolutional codes. IEE Electronic Letter, 32, 1186-1188.

Benedetto, S., Montorsi, G., Divsalar, D. & Pollara, F. (1996, August). Serial concatenation of interleaved codes: Performance Analysis, Design, and Iterative Decoding. JPL TDA Progress Report,42-126.

Gray, P., (1998, August). Serially concatenated trellis coded modulation. PhD Dissertation, Univ. of South Australia.

Divsalar, D. & Pollara, F., (1997, August). Hybrid concatenated codes and iterative decoding. JPL TDA Progress Report, 42-130.

Divsalar, D. & Pollara, F., (1997, September). Serial and hybrid concatenated codes with applications. Proceedings of the International Symposium on Turbo Codes and Related Topics, 80-87, Brest, France.

Barbulescu, S., A., & Pietrobon, S.S., (1995, January). Terminating the trellis of turbo codes in the same state. Electronic Letters, 31, 22-23.

Benedetto, S. & Montorsi, G., (1996, March). Unveiling turbo codes: some results on parallel concatenated coding schemes. IEEE Transactions on Information Theory, 42, 409-428.

Dolinar S., Divsalar D. & Pollara F., (1998, May). Code performance as a function of block size. TDA Progress Report, 42-133.

Frey, B., J. & Kscchischang, F., R., (1998, February). Early Detection and Trellis Splicing : Reduced-Complexity Iterative Decoding. IEEE Journal on Selected Areas in Communications, 16, 153-159.

Franz, B. & Anderson, J., B., (1998, February). Concatenated Decoding with a Reduced-Search BCJR Algorithm. IEEE Journal on Selected Areas in Communications, 16, 186-195.

Pietrobon, S., S. & Barbulescu, S., A., (1994, November). A simplification of the modified Bahl decoding Algorithm for systematic convolutional codes. International Symposium on Information Theory and its Applications, 1073-1077, Sydney, Australia.

Barbulescu, S. A., (1996, February). Iterative decoding of turbo codes and other concatenated codes. Ph.D. Dissertation, Univ. of South Australia.

Sklar B., (1997, December). A Primer on Turbo Codes. IEEE Communications Magazine, 35, 94-102.

Comatlas, (1995 May). CAS 5093 40 Mbit/s turbo code codec.

Kennedy, R., (1969). Fading in Dispersive Communication Channels. Wiley Interscience.

Schwartz, M., Bennett, W., & Stein, S., (1966). Communication Systems and Techniques. McGrow-Hill Book Company.

Wilson, S., (1996). Digital Modulation and Coding. Prentice Hall.

Jakes., W.C., (1974). Mobile Microwave Communication. John Wiley & Sons.

Robertson, P., Hoeher, P. & Villebrun, E., (1997, April). Optimal and sub-optimal maximum a posteriori algorithms suitable for turbo decoding. European Transactions on Telecommunications; 8, 119-125.

Kaplan, G. & Shamai, S., (1994, November). Achievable performance over the correlated Rician channel. IEEE Transactions on Telecommunications, 42, 2967-2978.