Noncoherent Coded Modulation

Thesis by

Dan Raphaeli

In Partial Fulfillment of the requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

1994

(Submitted April 19, 1994)

Acknowledgments

I gratefully acknowledge the help and support of various people who have influenced the growth of this thesis.

First and foremost, I would like to thank my advisor Professor Robert J. McEliece for his support and encouragement through the years. I thank him for his advice on my research and his substantive comments. I enjoyed his course on error-correcting codes immensely, and was able to use some of the concepts I learned in my research.

Special thanks are due to the late Professor Edward C. Posner, my previous advisor, for his initial guidance in my research. I am particularly indebted to him for the flexibility he allowed me in defining my research. He introduced me to JPL (NASA), where I have had the opportunity to be in close contact with many professionals and work on challenging problems. Blessed be his memory.

It is a pleasure to acknowledge Dr. Dariush Divsalar from JPL with whom I have had stimulating discussions. He contributed to this thesis with specific comments and ideas as well as editorial comments. Also, I would like to thank Dr. Marvin K. Simon for various discussions and for his constructive comments in the course of my research. Both his and Dr. D. Divsalar's work inspired me in my research.

I thank Professor P.P. Vaidyanathan for his help in matrix theory. His extensive knowledge of signal processing has benefited me through his courses and working with him at JPL.

It has been a pleasure to be a student of Professor Yaser S. Abu-Mostafa, Professor David Rutledge and Professor Laif Swanson. I also thank Professor Amnon Yariv for introducing me to the physics of quantum electronics.

Thanks to Professor Joel Franklin, Professor Samuel Kotz and Professor Norman L. Johnson for bringing to my attention several references relevant to my work.

I would like to thank the members of my thesis committee, Professor Robert J. McEliece, Professor David Rutledge, Dr. Fabrizio Pollara, Dr. Dariush Divsalar and Dr. Marvin K. Simon.

From JPL, I thank Dr. Sami M. Hinedi, Dr. Ramin Sadr, Keyvan Farazian, Dr. Fabrizio Pollara and Arthur W. Kermode, people with whom it has been a pleasure to

work.

Special thanks to my loving wife, Ruth Raphaeli, for her help in typing, editing, organizing and proofreading and to John Cortese, Dr. Dimitrios Antsos and Per-Olov Pettersson, for proofreading.

I thank Linda Dosza, Bette Linn, Illiana Salazar and Paula Samazan for helping me a great deal with the paperwork required to fulfill my administrative responsibilities.

Financial support was provided by fellowships from the Schlumberger Foundation and the Charles Lee Powell Foundation. Their unconditional support allowed me to proceed in my own original direction. Their support is gratefully acknowledged.

I reserve the greatest gratitude to the very special woman in my life, my beautiful wife Ruth, and to my sweet angel Iyelette, my daughter, for their love, support and encouragement.

Thank you all. I hope this thesis reflects your best hopes for me.

Abstract

In this thesis, we are concerned with the transmission of data over noncoherent channels (the carrier phase is random). We consider a receiving system which does not attempt to estimate the carrier phase from the received data. Instead, the transmitter and receiver will be designed so that the data transmission is robust with respect to the unknown phase variations of the channel. For the transmitter, we propose new combined coding and modulation, specifically designed to match the noncoherent channel. For the receiver, efficient decoders to noncoherently decode the coded modulation are developed. As a result, we are able to show, both analytically and by computer simulations, that Noncoherent Coded Modulation (NCM) approaches the performance of coded coherent modulation. NCM achieves almost the same power efficiency without bandwidth expansion or an extensive increase in complexity.

We consider the problem of the Uniform Error Property (UEP) for a broad class of transmitters and receivers. A sufficient condition for a general linear code to satisfy the UEP is presented and a structure of a trellis-coded modulation that satisfies this condition is offered. We call these codes "linear noncoherent trellis-coded modulation" since they apply to noncoherent detection. The problem of noncoherently catastrophic codes, which can result in noncoherent detection of trellis codes, is discussed and a general solution which does not rely on differential encoding of the code output is offered.

High performance noncoherent detection is achieved using multiple symbol observations. Unlike previous approaches, a sliding window is used for the observations, with each observation covering several branches of the trellis, so that the observations are time-overlapped. We define a new type of noncoherent maximum likelihood sequence estimator, and analyze its performance over the Additive White Gaussian Noise (AWGN) channel by numerical calculation of the union bound. We perform a computerized search and present new high performance coded M-ary Phase Shift Keying (PSK) modulations for noncoherent detection and their performance. The new codes cover many useful rates and complexities and achieve higher performance than existing codes for noncoherent detection. We evaluate the performance of NCM in the presence of phase jitter in the channel. The method can also be used for multiple symbol demodulation of M-ary Differential PSK (MDPSK) and of Continuous Phase Modulation (CPM). We provide results for both, full and partial response CPM schemes as well as convolutionally coded CPM. The complexity and power efficiency of this new method is superior to all past schemes known to the author for noncoherent detection.

The optimal implementation of the decoder, using the Viterbi Algorithm (VA), is given. For L-symbols observation, it requires a number of states that grows exponentially with L. Three novel sub-optimal algorithms are presented, whose number of states is the same as the original code so their complexity has a relatively weak dependence on L. For practical values of L, these algorithms are substantially less complex than the optimal algorithm.

The first suboptimal algorithm to be described is called the Basic Decision Feedback Algorithm (BDFA). In this algorithm, the symbols from the decisions are fed back to be used in the subsequent decisions. This algorithm suffers from increased error event probability and from error propagation. However, by a small modification of the BDFA, we obtain another improved algorithm, which will be called Modified Decision Feedback Algorithm (MDFA).

To obtain close to optimal performance, the third algorithm, the Estimated Future Decision Feedback Algorithm (EFDFA) is offered. This sophisticated algorithm, which uses the BDFA as a basic building block, is based on a novel concept called "estimated future." Performance analysis and simulation results are given.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Digital communication is the dominant means of information transmission to large distances. The advent of Very Large Scale Integration (VLSI) technology has made it feasible to include coding into an increasing number of designs for reducing the transmitter power, the antenna size, or increasing the data rate. Our information is given in a digital form. A practical transmission medium (or a communication channel as it is commonly referred to) only permits the passage of continuous waveforms to match the channel. Hence, in any digital communication system, the input bit stream is converted to an analog waveform. As this waveform traverses the channel, it is distorted. Furthermore, noise and interfering signals are added. Sources of impairment include additive noise, phase shift, fading and distortion due to filtering. Timing error between transmitter and receiver can also be considered an impairment. An attempt is made at the receiver to decide which of the possible waveforms is the most likely to have been transmitted.

Let us briefly go over the various steps involved in the transfer of information in a digital communication system. Then, we will address the types of distortion and the approaches used to overcome them.

The block diagram of a typical digital communication system is shown in Figure 1.1. The output of the information source, which is assumed to be a bit stream, enters the encoder. The encoder output is fed to the modulator. In the modulator, the input bits are translated into analog waveforms. In a simple modulator, such as an M-ary Phase Shift Keying (MPSK) modulator, one waveform is selected out of a group of possible waveforms in each symbol period, according to the modulator input. For the MPSK modulator, the output is a sine wave of a certain frequency whose phase is selected by the input information out of M evenly spaced phases. Most transmission channels are located in a certain frequency band. The modulator makes use of a carrier signal input to generate waveforms, the energy of which is concentrated within the required spectral band. The modulator block may also include a frequency up-converter. In such systems, the modulator consists of two stages. In the first stage, a low frequency carrier is modulated and in the second stage the signal is up-converted to the RF frequency. The carrier input to the modulator is generated by a local oscillator.

The transmission medium is the link required for the signal to reach the demodulator. In an RF wireless system, it includes a power amplifier, transmit antenna, space, receive antenna and receive amplifiers. Traditionally, the demodulator function is to re-derive the modulated symbols. The decoder tries to recover the information bits out of these symbols. In modern systems employing combined codes and modulation, the demodulator and the decoder are combined in one block. The demodulator, as well as the modulator, needs a locally generated carrier signal for proper operation. Any phase difference between these two oscillators can be viewed as a phase shift in the received signal. It is convenient to assume that the local oscillators have no phase difference, but the phase shift is caused by the channel instead. Another source of phase shift is the physical channel. For example, in a mobile environment, the path delay is variable, causing variations in the phase. The final block, the decoder, should provide the best estimate of the transmitted information from its noisy input.

There are two approaches to overcome the channel distortion. The first is to try to cancel the distortion and the other is to design a system that is robust with respect to that distortion. Because it is impossible to cancel the noise added to the signal, the second approach is used for additive noise. For the carrier phase shift, both approaches are feasible. In the first approach, the carrier phase is estimated from the received signal or given by side information, e.g., a pilot tone. Then the phase shift can be reversed at the receiver and thus canceled. When this approach is used, the transmission is called coherent. In this case we can redefine the channel excluding the carrier phase shift

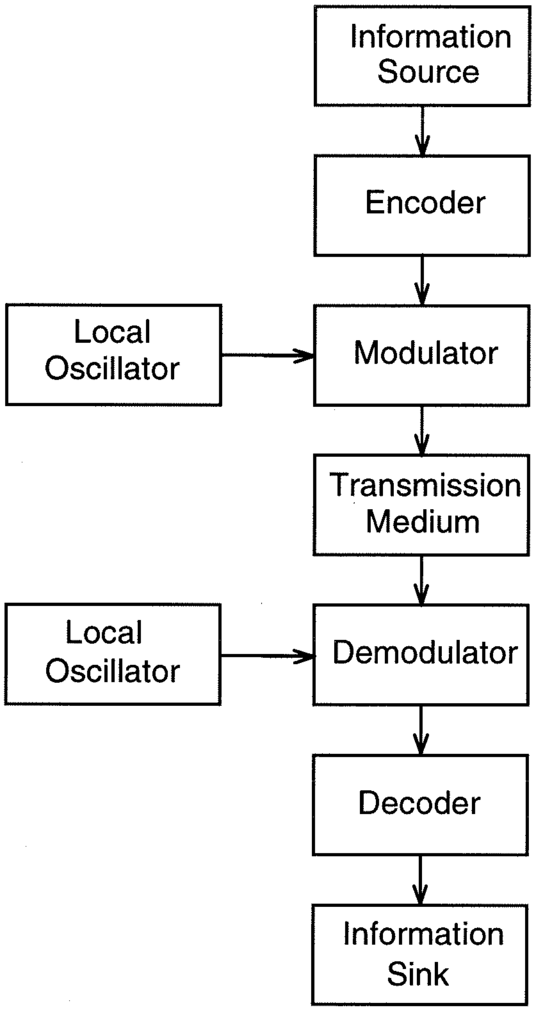


Figure 1.1: A digital communication system.

distortion, and the resulting channel is called a "coherent channel." The modulation used over a coherent channel is called "coherent modulation." The second approach assumes no knowledge about the carrier phase, and no attempt is made to estimate it. However, the waveforms used in the transmission are selected so that a phase shift cannot reduce the set of distances between them, and the receiver is implemented in such a way that it is insensitive to phase shifts. Hence, the system is robust to arbitrary and unknown phase shifts (but there is a limit to the phase variation rate). When this approach is used, the transmission is called "noncoherent."

An example of coherent modulation is Binary Phase Shift Keying (BPSK). In a typical system employing BPSK, a Costas loop is used to lock on to the input signal and extract the carrier phase[1]. This phase is used in the demodulator to detect the information bits. Frequency Shift Keying (FSK) is an example of a modulation that can be used in a noncoherent system. In this modulation, the information is conveyed by the frequency of the signal. The frequency of a signal can be measured without reference to its phase, so noncoherent reception can be implemented for this modulation. There are many advantages of noncoherent systems over coherent ones. We will explore and compare these two methods later in this thesis. The main drawback of noncoherent methods is a performance degradation relative to the coherent ones, when both are used over the Additive White Gaussian Noise (AWGN) channel.

In this thesis, we try to minimize the performance gap between coherent and noncoherent communication systems. A new noncoherent communication system, called Noncoherent Coded Modulation (NCM), is introduced as an alternative to coherent coded modulation. NCM approaches the power efficiency of coherent coded modulation without bandwidth expansion or an extensive increase in complexity compared to coherent. The same method is applied also to Continuous Phase Modulation (CPM). The complexity and power efficiency of this new scheme is superior to all past schemes known to the author for noncoherent detection. Unlike previous approaches, we do not require the use of differential encoding when using PSK in a noncoherent system. The problem of the Uniform Error Property (UEP) for multidimensional coded modulation, decoded by the generalized decoder, is analyzed. UEP means that the decoder error probability does not depend on which codeword is transmitted. A code with this property is convenient to analyze and to design. Ungerboeck said: "Good codes should exhibit regular structure." [1]. Forney [2] has shown that most good known trellis codes have an even more regular geometrical structure than might have been expected. Consider a space with some kind of metric defined over it and a one to one mapping between codewords and points in this space. We can heuristically argue that it is more probable that there exist high performance codes when the points of the space are distributed in such a way that the set of distances of any point from its neighbors are

[1]There is a residual ambiguity that also needs to be resolved.

the same for every point. This kind of distribution tends to be "more uniform" and thus the minimum distance is maximized.

A new family of combined coding and modulation for noncoherent decoding, that exhibits the UEP, is introduced, and high performance codes are found. The new codes cover many useful rates and complexities and achieve higher performance than existing codes for noncoherent detection. Unlike previous approaches, we do not require the use of differential encoding when using PSK in a noncoherent system. The optimal implementation of the decoder by the Viterbi Algorithm (VA) requires a large number of states. In order to overcome this increase in complexity, a number of reduced complexity algorithms are derived and analyzed.

If the performance and complexity of channel-robust systems are found to be close to those of coherent methods, then these new systems may be attractive alternatives to both coherent and noncoherent systems employed in various applications.

1.1 Outline of the Thesis

In Chapter 2, we try to form some unifying representation for digital communication systems. Working with this framework should enable us to treat all the different communication systems as one entity. The given representation will be used throughout the thesis. We describe the notion of noncoherent channels and their mathematical representation. We present and discuss the traditional noncoherent modulations and compare their performance with the performance of the coherent techniques.

In Chapter 3, we consider the general problem of UEP for multidimensional coded modulation, transmitted over the AWGN channel and received by a correlator-based receiver. These receivers are used in a broad class of communication systems, either coded or uncoded, either coherent or noncoherent. A sufficient condition for a general linear code to satisfy UEP is presented and a structure of trellis coded modulation that satisfies this condition is offered. We call these codes "linear noncoherent trellis coded modulation" since they apply to noncoherent detection. The main application of the theorems presented in this chapter is the design of coded modulations which are to be decoded by the emerging class of noncoherent decoders using multiple symbol observations. The scheme presented in this thesis is one of them.

In Chapter 4, we find the property of codes which allows them to be decoded noncoherently. For codes that noncoherent decoding is not possible, we present a possible correction to the encoder which allows the code to be decoded noncoherently.

A simple way to increase the performance of uncoded noncoherent orthogonal waveform modulation is suggested in Chapter 5. The idea is to overlap the transmitted symbols in time using an appropriate set of symbols. The best upper bound of the possible performance improvement in terms of $\frac{E_b}{N_0}$, when transmitting over the AWGN channel, is proven to be $\frac{M}{M-1}$. An example of a set of orthogonal symbols which can be overlapped to achieve this bound is the Walsh functions. The theory is a generalization of the relation which exists between binary FSK and Differential Phase Shift Keying (DPSK) for M-ary orthogonal noncoherent modulation. In order to overcome the limitation of possible improvement in performance, we try to extend the idea by the use of coding. Some trials with hand designed codes show the possibility of gaining significantly in performance. This development gave the inspiration to the development of Noncoherent Coded Modulation (NCM), the core of this thesis.

Trellis coded modulation with two or multidimensional signal constellations, together with coherent maximum-likelihood detection, is considered an attractive solution for communications over the AWGN channel. In Chapter 6, a new noncoherent communication system is introduced called Noncoherent Coded Modulation as an alternative to coherent coded modulation. NCM achieves almost the same power efficiency without bandwidth expansion or an extensive increase in complexity. As a noncoherent system, the method does not need carrier phase estimation. Nonetheless, differential encoding is not required. High performance noncoherent detection is achieved by using multiple symbol observations. Unlike previous approaches, a sliding window for the observations is used, with each observation covering several branches of the trellis, such that the observations are time-overlapped. We define a new type of noncoherent Maximum Likelihood Sequence Estimator (MLSE) and analyze its performance over the AWGN channel by numerical calculation of the union bound. We perform a computerized search and present new high performance codes for noncoherent detection with their performance. The new codes cover many useful rates and complexities and achieve higher performance than existing codes for noncoherent detection. The method can also be used for multiple symbol demodulation of M-ary DPSK with better results than existing methods. We evaluate the performance of NCM in the presence of fast randomly time varying phase in the channel. We compare the results with those of a Phase-Locked Loop (PLL)-based system. The error probability analysis requires the evaluation of the distribution of the indefinite non-central Hermitian quadratic forms in complex normal random variables. A new series expansion is developed for this distribution function and the cumulative distribution function. Three different versions are presented for evaluating either the probability distribution function or the cumulative distribution function. The series are fast converging and computationally efficient. The series have a rapid convergence when the eigenvalues are separated adequately. However, since we cannot guarantee this property, we have developed another numerical method. This second numerical method works well for all ranges of eigenvalues and is computationally efficient. In addition, several approximations are given.

In Chapter 7, Several decoding methods for the IO-NMLSE are described. The optimal implementation of the decoder, using the Viterbi Algorithm (VA), is given. For L-symbols observation, it requires a number of states that grows exponentially with L. Then, three novel sub-optimal algorithms are presented, whose number of states is the same as the original code so their complexity has a relatively weak dependence on L. For practical values of L, these algorithms are substantially less complex than the optimal algorithm.

The first suboptimal algorithm to be described is called the Basic Decision Feedback Algorithm (BDFA). In this algorithm, the symbols from the decisions are fed back to be used in the subsequent decisions. This algorithm suffers from increased error event probability and from error propagation. However, by a small modification of the BDFA, we get another improved algorithm, which will be called Modified DFA (MDFA).

To obtain close to optimal performance, the third algorithm, the Estimated Future Decision Feedback Algorithm (EFDFA) is offered. This sophisticated algorithm, which uses the BDFA as a basic building block, is based on a novel concept called "estimated future." Performance analysis and simulation results are given.

In Chapter 8, we apply the noncoherent decoding technique proposed in previous chapters to CPM. As in the case of coded PSK, we show that this scheme achieves almost the same power efficiency as coherent CPM using VA. A new trellis diagram for noncoherent CPM is suggested for simplifying the analysis and the decoder structure. The error performance is evaluated by using the union bound technique applied to the symbols-difference trellis diagram. The very efficient sub-optimal decoding algorithms of Chapter 7, with very small degradation, are also used. The complexity and power efficiency of this new method is superior to all past schemes known to the author. Both full and partial response CPM schemes with arbitrary modulation index are considered as well as convolutionally coded CPM.

1.2 Publications List

Parts of the material of this thesis will appear in coming journal and conference publications. A preliminary publications list is given here. Unless mentioned otherwise, there are no joint authors to the following publications.

- The material in Chapter 3 and 4:
  - "Uniform Error Linear Coded Modulation for Noncoherent Channels," The Annual Conference on Information Sciences and Systems (CISS), Princeton, NJ, March 16–18, 1994.
  - "Uniform Error Group Codes for Generalized Decoder and Their Application to Noncoherent Detection," IEEE Transactions on Information Theory, submitted.
- Part of the material in Chapter 5:
  - "Improvement of Noncoherent Orthogonal Coding by Time-Overlapping," IEE Proceedings, accepted for publication.
- Parts of the material in Chapter 6:

  - "Noncoherent Coded Modulation," IEEE Transactions on Communications, submitted.
  - "Combined Noncoherent Detection and Decoding of Coded Modulation," The Annual Conference on Information Sciences and Systems (CISS), Princeton, NJ, March 16–18, 1994.
  - "The Performance of Noncoherent Coded PSK Modulation in the Presence of Phase Noise," IEEE International Conference on Universal Personal Communications, submitted.

- Part of the material in Chapter 7:
  - "Decoding Algorithms for the Noncoherent Coded Modulation," IEEE Transactions on Communications, submitted.
  - "Efficient Decoding Algorithms for the Noncoherent Coded Modulation," IEEE GLOBECOM, San Francisco, Nov. 1994, accepted for publication.

- Part of the material in Chapter 8:
  - "Noncoherent Detection of Continuous Phase Modulation Using Overlapped Observations," with D. Divsalar, Communications Theory Mini-Conference included in GLOBECOM'94.
  - "Efficient Noncoherent Decoding of Uncoded and Convolutionally Coded Continuous Phase Modulation," with D. Divsalar, IEEE Transactions on Vehicular Technology, will be submitted.

Bibliography

[1] Ungerboeck G., "Channel Coding with Multilevel/Phase Signals," IEEE Trans. Info. Theory, Vol. IT-28, pp. 55–67, Jan. 1982.

[2] Forney G.D., "Geometrically Uniform Codes," IEEE Trans. Info. Theory, Vol. 37, No. 5, pp. 1241–1260, Sep. 1991.

Chapter 2

Coding, Modulation and Detection

2.1 Review of Modern System Representation

Digital communication systems used in practical applications are often very complicated. For example, the receiver for the NASA Deep Space Network took over one decade to develop and cost millions of dollars. Moreover, every communication system is different. There are many elements to be defined including the code, the modulation, the equalizer, the transmitted frequency, etc. For any such element, there are many possible methods of implementation. To investigate the theory of the operation of a digital communication system, it would be desirable to develop a unifying representation. Working in such a framework will enable us to treat a large class of communication systems as one. Next, we describe the system representation which will be used in this thesis.

A baseband equivalent block diagram of a digital communication system is shown in Figure 2.1. All the signals are represented by their complex envelopes [1]. Our goal is to derive a vectorized version, as shown in Figure 2.2. Let $x(t)$ be one of the possible transmitted signals and T_s the duration of an individual symbol. Let us choose a complete orthogonal set $\{\varphi_n\}$ of complex functions such that

$$\int\limits_{-\infty}^{\infty} \varphi_n^*(t)\varphi_m(t)dt = \begin{cases} 1, & n = m\ ; \\ 0, & n \neq m. \end{cases} \tag{2.1}$$

It is convenient to assume that the transmitter is turned on at $t = 0$. Then we can represent $x(t)$ as

$$x(t) = \sum_{i=0}^{\infty} x_i(t), \tag{2.2}$$

where

$$x_i(t) = \sum_{j=0}^{D-1} x_{i,j}\varphi_j(t - iT_s), \tag{2.3}$$

and D is the number of complex dimensions needed to represent each symbol $x_i(t)$ in the specific coordinate system $\{\varphi_n\}$. The symbols $x_i(t)$ may have an infinite duration, like in the case of Nyquist pulses. Most of the energy of $x_i(t)$ is usually found within the time interval $iT_s < t < (i+1)T_s$ (with proper alignment of the time origin). For certain modulation types or choices of $\{\varphi_n\}$, D can be infinite. Each symbol of $x(t)$ will be written as a complex vector of dimension D, $\mathbf{x}_i = (x_{i,0}, \ldots, x_{i,D-1})$. It follows that we can describe the whole transmitter as a code which maps the sequence of input symbols (which are commonly bits) to a sequence of output symbols which are complex vectors, each of D dimensions. The code is often a block code, operating on a finite number of symbols, or a trellis code, operating on an infinite input sequence and outputting an infinite sequence. Practically everything is finite, but for a long message the infinite length is a good approximation. A block code is a mapping of a block of input symbols to a block of output symbols. A trellis code is a state machine. The input symbol and the current state determine the next state, and for each such transition one or more output symbols are assigned. We will assume that $\{\varphi_i\}$ are chosen such that there is no Inter-Symbol Interference (ISI) [1] at the matched filter output. This condition can be defined as

$$\int_{-\infty}^{\infty} \varphi_n^*(t)\varphi_m(t - lT_s)dt = 0 \quad \forall l \neq 0, n, m. \tag{2.4}$$

If no such representation is possible, then it is always possible to form an equivalent system where the ISI is converted into a trellis code in which output symbols have no ISI.

Let us try some examples. The first is Binary Phase Shift Keying (BPSK) modulation without coding.

$$x(t) = \sum_{i=0}^{\infty} \alpha_i q(t - iT_s), \tag{2.5}$$

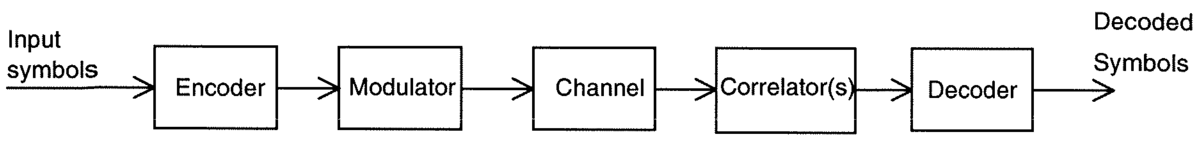


Figure 2.1: A base-band model of a digital communication system.

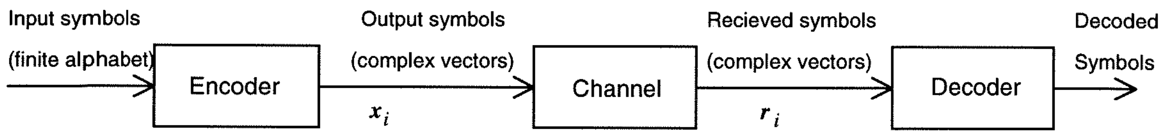

Input symbols (finite alphabet)
Encoder
Output symbols (complex vectors)
x_i
Channel
Recieved symbols (complex vectors)
r_i
Decoder
Decoded Symbols

Figure 2.2: An equivalent model for digital communication systems.

where $q(t)$ is some waveform which does not lead to ISI, and $\alpha_i = \pm 1$ corresponds to the binary input symbols. Naturally, we choose $\varphi_0(t) = q(t)$ and $D = 1$. Then we have $\mathbf{x}_i = \alpha_i$. The code is reduced to a one-to-one mapping from input bits to output symbols. The next example is DPSK (Differential Phase Shift Keying) modulation. Here

$$x(t) = \sum_{i=0}^{\infty} \beta_i q(t - iT_s), \quad \beta_i = \beta_{i-1}\alpha_i, \tag{2.6}$$

with initial conditions $\beta_{-1} = 1$. Here, we use again $\varphi_0(t) = q(t)$ and $\mathbf{x}_i = \beta_i$. There is a trellis code which maps the input bits to the output symbols β_i. This code has two states, but has no redundancy (looked upon as a convolutional code, this encoder is non-minimal. The minimal encoder sends the bits uncoded, i.e., it has only one state). Its trellis is shown in Figure 2.3.

Any modulator with memory is equivalent to a code, most often a trellis code. Adding a binary encoder in front of this modulator merely generates an equivalent code. A good example to such modulator is CPM (Continuous Phase Modulation) [2]. Some people prefer to separate a code into two parts. The first is a binary encoder and the second is a memoryless mapper from bits to complex valued vectors.

Let us move on to discuss the channel. A typical channel attenuates, filters, adds noise and causes a phase shift to the signal. Coherent channels do not cause a phase

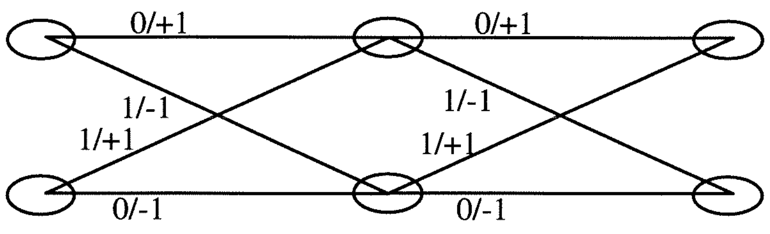


Figure 2.3: Trellis representation of DPSK.

shift to the signal (a phase shift of ϕ is mathematically a multiplication by $e^{j\phi}$ of the base-band signal). Noncoherent channels cause random time-varying phase shifts.

The final block, the receiver, serves to decode the input signal back to the transmitted information. Let $r(t)$ be the complex envelope of the received signal. $r(t)$, as $x(t)$, is decomposed by the orthonormal functions as

$$r(t) = \sum_{i=0}^{\infty} r_i(t) = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} r_{i,j}\varphi_j(t - iT_s). \tag{2.7}$$

Note that an infinite number of dimensions is needed in general to represent $r(t)$ due to the infinite dimensionality of the noise, but later we will only use D of them since all the components of the received signal outside the signal space are irrelevant for the decision. For an additive noise channel, we can express the received noise $n(t)$ in the same way.

A large class of receiving systems, either coded or uncoded can be represented as follows. We will assume that there is no timing error between the transmitter and the receiver. The receiver performs symbol by symbol correlations between the received signal, $r(t)$, and a candidate signal $\hat{x}(t)$ (a possible transmitted signal). The results of these correlations are the arguments of a real-valued likelihood function $\Lambda(\hat{x})$[1]. The receiver chooses $\hat{x}$ which maximizes this function. The correlator output for the i'th symbol is defined as

$$\mu_i = \int_{-\infty}^{\infty} r(t)^* x_i(t)dt = \mathbf{r}_i^\dagger \mathbf{x}_i, \tag{2.8}$$

where $\mathbf{r}_i = (r_{i,0}, r_{i,1}, \cdots, r_{i,D-1})$ and '$\dagger$' denotes the conjugate and transpose of a vector

[1] Λ is often called a "metric".

or matrix. The vector $\underline{\mu} = (\mu_0, \cdots, \mu_{N-1})$, representing the sufficient statistics for decision, is the input to the likelihood function

$$\Lambda(\hat{\mathbf{x}}) = f(\underline{\mu}). \tag{2.9}$$

Here, N is the number of symbols the decoder operates on. For a block code, N is the size of the output block. For the case of uncoded modulation N is usually equal to 1, and for a trellis code $N \to \infty$. For a coherent additive noise channel we can express $\mathbf{r}_i$ as

$$\mathbf{r}_i = \mathbf{x}_i + \mathbf{n}_i, \tag{2.10}$$

where $\mathbf{n}_i = (n_{i,0}, n_{i,1}, \cdots, n_{i,D-1})$. For a coherent decoder, for the case of equiprobable and equal-energy waveforms over the AWGN channel, the optimal likelihood function, f, is

$$f(\mu) = \sum_{i=0}^{N-1} Re\{\mu_i\}. \tag{2.11}$$

We see that the receiver is composed of two parts. The front end is the correlator, which correlates the input signal with all the possible symbols. Its output is fed into a decoder (or decision device for the uncoded case) which operates by maximizing the likelihood function, $\Lambda(\hat{\mathbf{x}})$ and finding the information bits that correspond to the sequence which achieves this maximum; see Figure 2.4. The correlation at the front end is, most often, accomplished by matched filters. A matched filter will either output the component $r_{i,j}$ using φ_j as the template, or will output μ_i using $x_i(t)$ as the template. In the first case, some signal processing block will compute the vector product $\mathbf{r}_i^\dagger \mathbf{x}_i$. We will call this type of receiver a *correlator-based receiver.* Hereafter, a decoder will always refer to a device which maximizes a likelihood function as described above. The definition of the code is general and it includes also the uncoded case as a special case. Since we have included the uncoded modulations as special case of the coded modulations, "decoding" will include the detection using a symbol-by-symbol decision device.

Next, we will discuss coherent decoding, noncoherent decoding, and partially coherent decoding. A noncoherent decoder ignores the phase shift $\phi(t)$ of the channel with the restriction that the phase is slowly varying. The noncoherent decoder is built under the assumption that we have no information about $\phi(t)$, except its slow variation. When some information is available about the phase, for example if $\phi(t)$ is the residual phase error of a Phase-Locked Loop (PLL), then the resulting decoder will be called partially coherent. We postpone the discussion on how to find the optimal noncoherent decoder until Chapter 6.

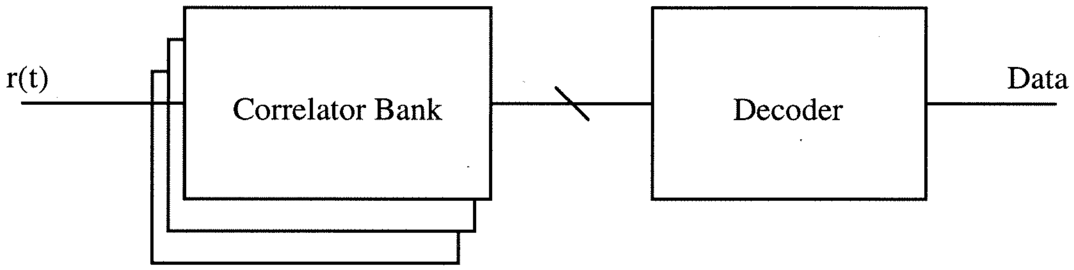


Figure 2.4: A correlator-based receiver.

2.2 Noncoherent Channels

The noncoherent channel introduces a phase shift $\phi(t)$ to the received waveform. Assuming additive noise is independent of the phase shift of the signal, noise statistics remain unchanged, whether the noise is added before the phase shift, i.e., $r(t) = (x(t) + n(t))e^{j\phi(t)}$ or after the phase shift, i.e., $r(t) = x(t)e^{j\phi(t)} + n(t)$. For analysis throughout the thesis we will use the latter form (see Figure 2.5a). The channel phase, $\phi(t)$, is a random process. Its statistics depend on its physical origin. In many cases, it is very difficult to find the exact model of the phase process (also referred as phase noise or phase jitter). We use the assumption that the phase is slowly varying relative to the symbol rate. If this is not the case, then communication becomes difficult if not impossible, and large degradation is unavoidable. We will constrain ourselves to cases where the phase variation is negligible over the duration of at least one symbol[1]. Under this assumption we can express the received signal in a vector form as

$$\mathbf{r}_i = \mathbf{x}_i e^{j\phi(iT_s)} + \mathbf{n}_i, \tag{2.12}$$

[1]More specifically, over the duration where most of the energy of the symbol is concentrated.

see Figure 2.5b. The duration over which the phase is assumed constant will be called an "observation."

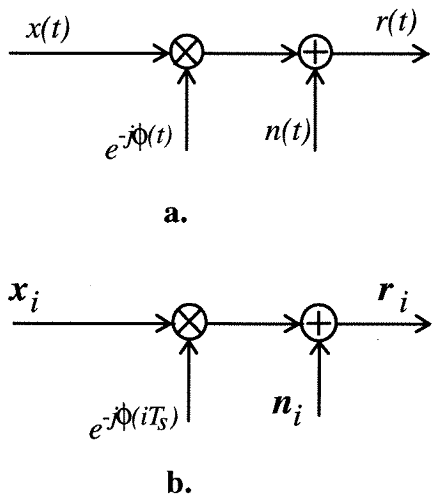


Figure 2.5: A noncoherent additive noise channel.

2.3 Existing Noncoherent Decoders

One of the oldest types of noncoherent modulation is M-ary Frequency Shift Keying (MFSK). In MFSK, M frequencies $\omega_0, \cdots, \omega_{M-1}$ are used to transmit the information. Each output symbol is a sinusoidal wave $s_n(t) = \sin(\omega_n t + \theta)$. For optimal noncoherent performance[1], the frequencies should be separated from each other by multiples of $\frac{2\pi}{T_s}$. This makes the symbols orthogonal[2] to each other. If the phase of the sinusoid is kept continuous in the transition between symbols, we call the modulation Continuous Phase FSK (CPFSK). The phase continuity introduces memory to the modulator and has the effects of coding. A decoder that utilizes this memory performs better than one that looks at independent symbols. There are many decoders for noncoherent MFSK. One popular suboptimal detector is based on a discriminator. A discriminator is a

[1]Optimal with respect to a detector which looks at independent symbols.

[2]Orthogonality between different symbols in their complex envelope representation is defined as $\int_0^{T_s} s_n^*(t)s_m(t)dt = 0$.

device that measures instantaneous frequency. Obviously, one can use this device to determine which one of the possible waveforms was transmitted. This method works especially well when CPFSK is used, and in particular for $M = 2$. In fact, for $M = 2$, the discriminator detector for the optimal choice of parameters has better performance than the optimal detector for FSK with independent symbols, since it uses (indirectly) the modulation memory.

If we assume that the channel phase is independent from one symbol to the next (in this case the phase continuity in the modulator does not make any difference), then the optimal decoder is as follows (this case is most commonly referred to as the only form of noncoherent detection). We assume orthogonal waveforms, equally likely, having equal energy and the channel is AWGN. Since the symbols are orthogonal, we can choose

$$\varphi_n(t) = s_n(t), \quad n = 0, \cdots, M-1. \tag{2.13}$$

The optimal detector is a correlator-based receiver as defined in Section 2.1. Its choice of f is

$$f(\mu) = |\mu|^2. \tag{2.14}$$

Here $N = 1$, so μ is not a vector, and we perform a separate decision on each symbol. The performance of MFSK over an AWGN channel becomes better and better as M grows, and in the limit of $M \longrightarrow \infty$ it achieves the channel capacity [1]. However, we must remember that the observation time length also grows with M. For each output symbol, $\log_2 M$ bits are sent. Keeping the bit rate, $\frac{1}{T_b}$, constant, the observation length is $T_b \log_2 M$. For a large M, the assumption made that the phase is constant over the length of the symbol may no longer be satisfied, so it is not useful over practical channels.

Instead of using different frequencies for making the symbols orthogonal (orthogonality guaranties optimum performance), we can form other sets of orthogonal waveforms to be used as output symbols. For example, we can use the Walsh functions [5]. These functions take only values of $+1$ or -1, and thus we benefit from using BPSK transmitters and matched filters when using these functions as symbols. We also get easier symbol time synchronization in the receiver compared to MFSK. Let us call this modulation Walsh Shift Keying (WSK). The matched filter bank for MFSK can be implemented using a Fast Fourier Transform (FFT). Likewise, the matched filter bank for WSK can be implemented using the Fast Walsh Transform (FWT) [5]. The FWT involves only additions and substractions, thus is easier to implement than the FFT. An example of a set of Walsh functions is shown in Figure 2.6.

The next type of noncoherent modulation we will discuss is DPSK. In DPSK the information is encoded in the difference between the phase of two consecutive symbols. Even though DPSK is a coded scheme, it is possible to make symbol-by-symbol decisions. The optimal noncoherent decoder for DPSK and its performance can be found by making the following observation. Two consecutive symbols will switch sign if the input bit is 1, or stay the same if the input bit is 0. Refer to Figure 2.7. We can say that when the input bit is 1, we transmit the waveform $\pm s_1(t)$ and when it is 0, we transmit $\pm s_2(t)$. The sign depends on the previous modulator output value. The resulting output waveform can be seen as an overlap of $s_1(t)$ and $s_2(t)$ in time. The decoder has to make the decision, which one of $s_1(t)$ or $s_2(t)$ was transmitted by looking at the received signal over two symbols. Since $s_1(t)$ and $s_2(t)$ are two orthogonal waveforms, the receiver is equivalent to the receiver for binary FSK (the decoder can be simplified to the familiar form $Re\{r_i r_{i-1}^*\} \gtrless 0$). Its performance is the same as that of the FSK receiver. There is a dependency between consecutive decisions, but this does not affect the per-bit error probability. Notice that the energy for each of the orthogonal waveforms used is the energy of two bits. The energy in binary FSK symbols, on the other hand, is the energy of one bit. Thus, we obtain an improvement of exactly 3 dB, over binary FSK.

In the case of MFSK, the observation length grows with M, and the performance improves. Comparing DPSK to binary FSK, we increase the observation length from one bit to two bits. Hence we conclude that increasing the observation length leads to better performance. This is indeed the case for most noncoherent decoders. Recently, various authors have investigated the use of multiple symbol observation length to improve the performance of noncoherent systems. This subject will be covered later in this thesis.

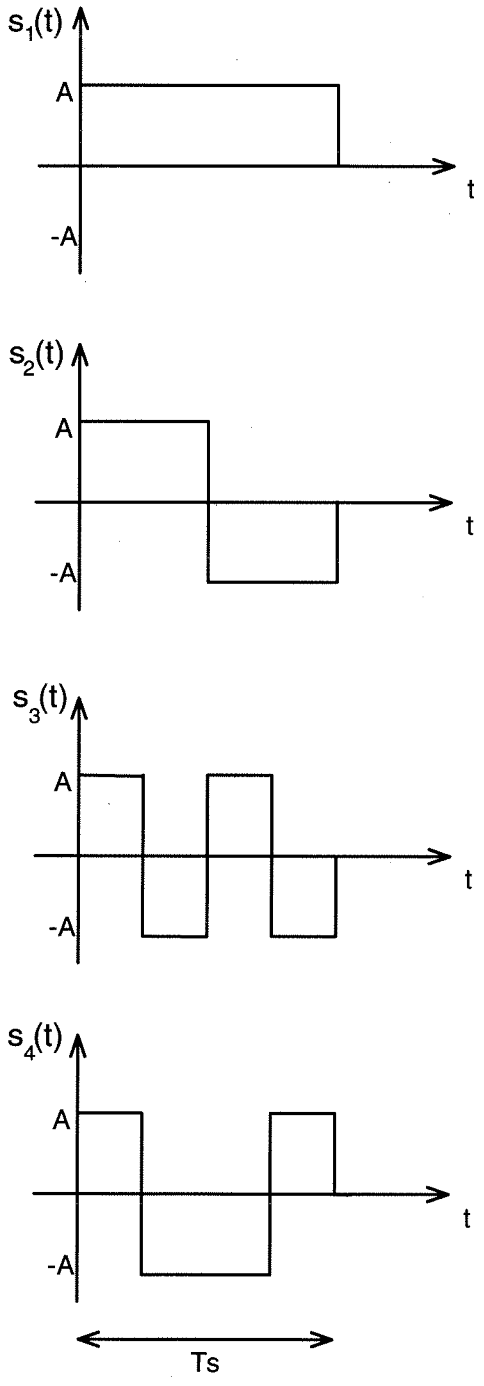


Figure 2.6: A set of 4 Walsh functions.

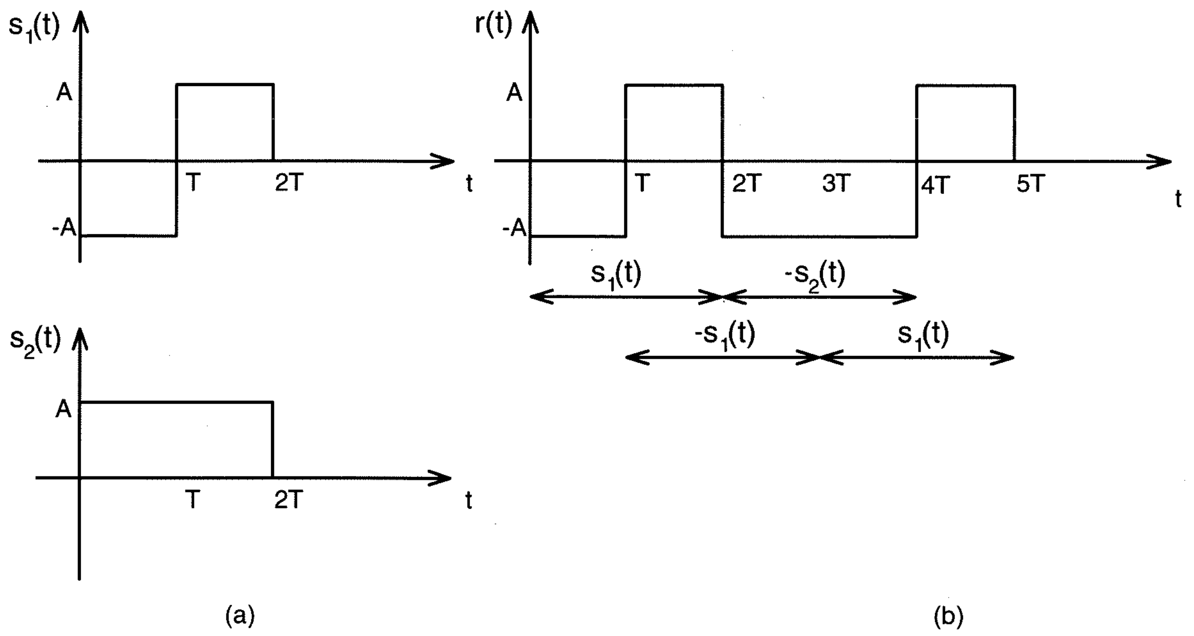


Figure 2.7: DPSK viewed as FSK with time-overlapping. (a) The two orthogonal signals $s_1(t)$ and $s_2(t)$ that will be used to construct DPSK waveform. (b) DPSK waveform in baseband, $r(t)$, is constructed by time-overlapping sequence from the set $s_1(t)$, $s_2(t)$.

In Figure 2.8, we compare the performance of uncoded schemes over the AWGN channel. We can notice a 1 dB degradation of DPSK compared to coherent BPSK at high Signal to Noise Ratio (SNR). 2FSK loses 3 dB compared to DPSK, but the performance of 4FSK is close to that of DPSK. Note that both DPSK and 4FSK have the same observation length. Some coded modulations are compared in Figure 2.9, which is re-drawn from [4]. Here we see 3 dB difference between coherent and noncoherent modulations. To make the outputs of the DPSK detector independent for optimal use of Viterbi Algorithm (VA), we use interleaving. Without interleaving we suffer about 1 dB of additional degradation. When coding is used, traditional noncoherent modulations suffer high degradation compared to coherent ones, over the AWGN.

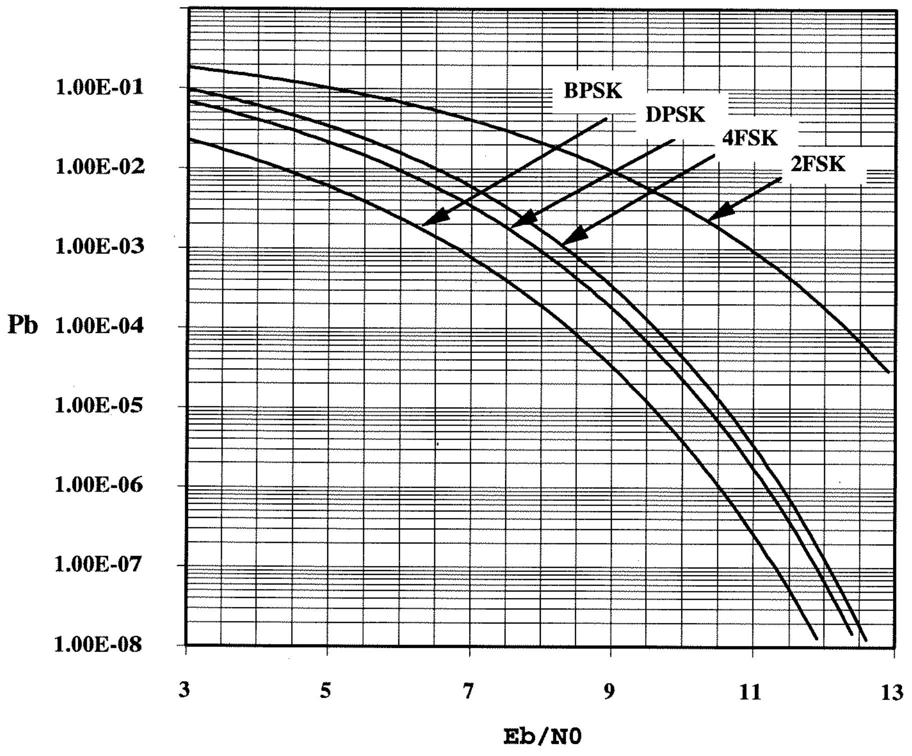


Figure 2.8: Performances of uncoded coherent BPSK and of noncoherent DPSK, 2FSK and 4FSK.

2.4 Advantages of Noncoherent Decoding

There are many situations where noncoherent detection is preferred. In some of them, the elimination of the phase estimation circuitry simplifies the receiver. In others, the channel includes oscillator phase noise, fast fading or fast Doppler effects which cause the carrier phase to change rapidly, and do not allow its estimation and tracking at the receiver [1, page 289]. Channels which are limited or degraded by oscillator phase noise are found in systems employing high carrier frequencies in conjunction with low bit rate. See for example the Space Shuttle communications [6] and the INMARSAT standard C ship stations [7]. In [7], noncoherent techniques, in particular MFSK, are shown to be the best choice for this channel, which includes fast fading and oscillator phase noise, together with a low bit rate. In the cellular-mobile radio systems, multipath fading and Doppler shifts significantly degrade the performance of

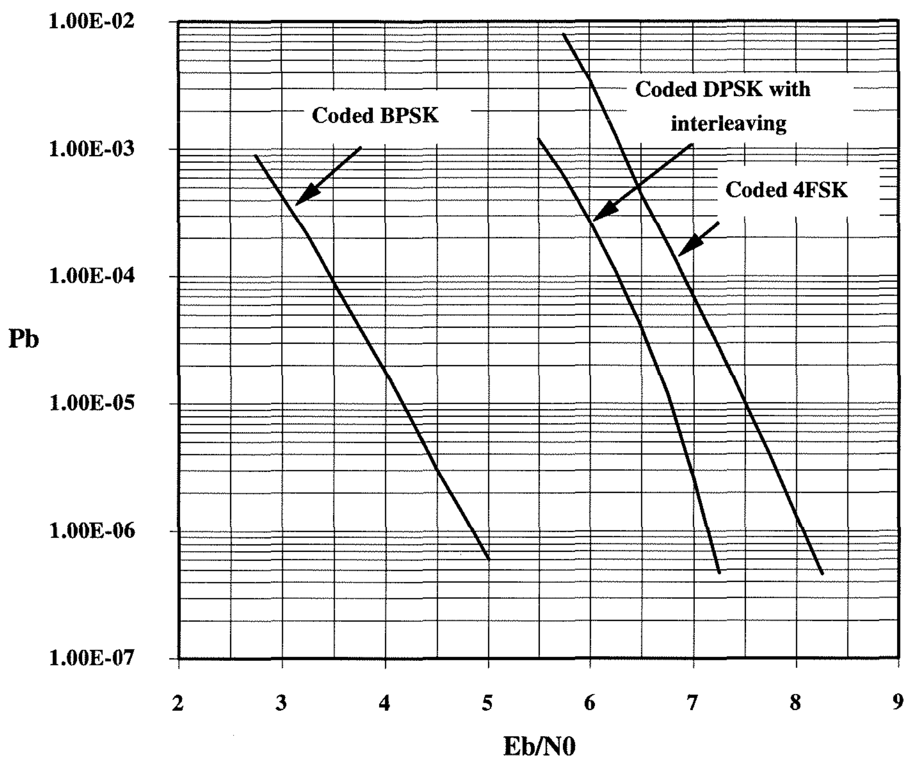


Figure 2.9: Performance of rate 1/2 convolutionally coded BPSK, DPSK and 4FSK. Infinite decoder memory, and infinite quantization is assumed. Ideal interleaving is used for the DPSK.

coherent detection [8]. Mobile communication channels are characterized by fading and Doppler shifts which make them in many cases unsuitable for coherent communications. Therefore, noncoherent techniques are preferred candidates for this type of channels [9]–[11]. Noncoherent detection may also be more suitable for burst operated Time Division Multiple Access (TDMA) systems requiring fast synchronization and re-synchronization [12],[13, Section 11.6]. Another environment that benefits from noncoherent decoding is military spread-spectrum communication [14], [15].

There are many additional advantages of noncoherent detection over coherent detection. There is no acquisition time so that no data would be lost until tracking is established. There is no acquisition mode and system requirement involved, like lock detection and mode switching. Fast recovery from severe fading occurs in noncoherent modulation. There is no false-lock, phase slipping and loss of lock associated with a

PLL circuit. In many cases, coherent modulations suffer high degradation from phase estimation errors.

Phase estimation is a problem, especially when bandwidth efficient modulations, like Trellis Coded Modulation (TCM) [16] and CPM [17], [18], are used. Due to the difficulty in achieving coherency in CPM systems, many noncoherent receivers were suggested in the literature [17]. The carrier phase tracking of CPM or TCM can be performed by a PLL. There are two ways to implement the phase tracking. One is to remove the data by a nonlinear operation, and then lock to one or more of the discrete spectral components produced. The other is to decode the data and feed it back into the loop (data aided loop). Data aided loop may be the better way when the TCM employs Quadrature Amplitude Modulation (QAM). If discrete spectral components are to be generated from a CPM scheme with M phase states or from MPSK modulation, the signal has to be raised to the Mth power. Since this process results in a large SNR drop, very narrow loops must be employed. If a data aided loop is employed, the large decoding delay caused by the VA will only allow the operation of a PLL with a large time constant (narrow-band). The use of extremely narrow-band PLL, as required, is not allowed in cases where fast phase variations caused by phase noise, fading or Doppler effects occur. The large time constant of the PLL causes additional problems like slow acquisition.

Bibliography

[1] Proakis J.G., *Digital Communications.* NY: McGraw Hill, 1989.

[2] Anderson J.B., Aulin T., Sundberg C.E., *Digital Phase Modulation.* NY: Plenum Press, 1986.

[3] Viterbi A.J., Omura J.K., *Principles of Digital Communication and Coding.* McGraw-Hill, 1979.

[4] Clark G.C., Cain J.B., *Error-Correction Coding for Digital Communications.* NY: Plenum Press, 1981.

[5] Beauchamp K.G., *Walsh Functions and Their Applications.* NY: Academic Press, 1975.

[6] Lindsey W.C., Kwei Tu, "Phase Noise Effects on Space Shuttle Communications Link Performance," IEEE Trans. Comm., Vol. COM-29, No. 11, Nov. 1978.

[7] Rhodes S., Hagmann W., "Signal Design for INMARSAT Standard C Ship Earth Stations," in *Proc. GLOBECOM'82*, Miami FL, Vol. 3, pp. 1051–60, Nov. 1982.

[8] Feher K., "MODEMS for Emerging Digital Cellular-Mobile Radio System," IEEE Trans. Veh. Tech., Vol. 40, No. 2, May 1991.

[9] Mackenthun K.M., "Modulation Selection for the Mobile Satellite Experiment (MSAT-X)," COMSAT Tech. Rev., Vol. 17, No. 1, pp. 87–103, Spring 1987.

[10] Rafferty W., Divsalar D., "Modulation and Coding for Land Mobile Satellite Channels," Proc. ICC'88, pp. 1105–1111, 1988.

[11] Lodge J.H., Moher M.L., Crozier S.N., "A Comparison of Data Modulation Techniques for Land Mobile Satellite Channels," IEEE Trans. Veh. Tech., Vol. VT-36, pp. 28–34, Feb. 1987.

[12] Lombard D., Imbeaux J.C., "Multidifferential PSK-demodulation for TDMA transmission," Int. Conf. Satellite Comm. Syst. Technol., London, England, pp. 207–213, 1975.

[13] Gagliardi R.M., *Introduction to Communications Engineering.* Wiley-Interscience, 1988.

[14] Milstein L.B., Davidovici S., Schilling D.L., "Coding and Modulation techniques for Frequency-hopped Spread Spectrum Communications over a Pulse-Burst Jammed Rayleigh Fading Channel," IEEE J. Select. Areas. Comm., Vol. SAC-3, pp. 644–651, Sep. 1985.

[15] Geraniotis E.A., "Performance of Noncoherent Direct-Sequence Spread-Spectrum Multiple-Access Communications," IEEE J. Selec. Areas Comm., Vol. SAC-3, pp. 687–694, Sep. 1985.

[16] Ungerboeck G., "Trellis-Coded Modulation with Redundant Signal Sets Part 2: State of the Art," IEEE Comm. Magazine, Vol. 25, No. 2, pp. 12–21, Feb. 1987.

[17] Aulin T., Sundberg C.E., "Partially Coherent Detection of Digital Full Response Continuous Phase Modulated Signals," IEEE Trans. Comm., Vol. COM-30, No. 5, pp. 1096–1117, May 1982.

[18] Svensson A., Aulin T., Sundberg C.E., "Symbol Error Probability Behavior for Continuous Phase Modulation with Partially Coherent Detection," AEU, Vol. 40, No. 1, pp. 37–45, 1986.

[19] Sklar B., *Digital Communications, Fundamental and Applications.* NJ: Prentice Hall, 1988.

[20] Alexovich J.R., Gagliardi R.M., "The Effect of Phase Noise on Noncoherent Digital Communications," IEEE Trans. Comm., Vol. 38, No. 9, Sep. 1990.

[21] Golomb S.W., Baumert L.D., Easterling M.F., Stiffler J.J., Viterbi A.J., *Digital Communication with Space Applications.* NJ: Prentice Hall, 1964.

Chapter 3

Uniform Error Property and Linear Noncoherent Codes

3.1 Introduction

Most commonly used channel encoders are linear over some group, commonly the binary field $GF(2)$. The linearity property helps us considerably in the analysis, design and decoder implementation. In particular, linear codes, which are used over a Binary Symmetric Channel (BSC), satisfy a nice property called Uniform Error Property (UEP). UEP means that the error probability is the same for all the transmitted codewords; $P_r(\text{error}) = P_r\{\text{error}|x_i\}$ were x_i is any transmitted codeword. By using this property we do not have to check all possible pairs of codewords in order to find the decoder error probability when evaluated by the union bound or similar techniques. Hence, we like to have codes which satisfy UEP when used with a noncoherent decoder (see also the argument, of why UEP codes tend to be good codes, given in Chapter 1). However, when linear codes are used in conjunction with carrier modulation, the resulting coded modulation might no longer be linear (depending on how we define the group addition between any two output symbols). Furthermore, in the case where we have linearity, UEP does not necessarily hold.

In this chapter, the term "general decoder" will refer to any decoder that can be represented as the decoder part of a correlator-based receiver as defined in Chapter 2.

Unlike the linearity of the code, which does not depend on the decoder, the UEP depends on the decoder's metric and the channel. In example 3.1.1 we demonstrate a linear code which satisfies UEP when decoded coherently, but when decoded noncoherently, strongly disobeys UEP.

Example 3.1.1 *This example presents a linear code with mapping to Quadrature Phase Shift Keying (QPSK) modulation which does not satisfy the UEP under noncoherent decoding. Consider a code which consists of the following* 4 *binary codewords:*

$$\{W_k\} = \{0000, 0001, 0100, 0101\}.$$

This linear code is not a useful code; it is constructed for demonstration only. Let us group every two bits and map them to QPSK symbols by Gray mapping. The QPSK symbols are $1, j, -1$ *and* $-j$*. It is known, and also easy to show, that for QPSK modulation Gray mapping conserves the UEP when the detection is coherent. Gray mapping maps the pairs* $[00, 01, 11, 10]$ *to the symbols* $[1, j, -1, -j]$ *correspondingly. The resulting symbols for each codeword are:*

$$\{\mathbf{x}_k\} = \{(1,1), (1,j), (j,1), (j,j)\}.$$

Now let us have a block noncoherent decoder (the observation covers one block, i.e., one codeword, and for each block the carrier phase is independent; see [9, Appendix 4c]) which operates by choosing i *which maximizes (in vector notation)* [1]

$$\Lambda(\mathbf{x}_i) = |\mathbf{r}^{\dagger}\mathbf{x}_i|^2, \tag{3.1}$$

where $\mathbf{r}$*is the received signal. If there is no single maximum, the decoder chooses randomly from the maximal ones. If* $\mathbf{x}_0$ *or* $\mathbf{x}_3$ *is transmitted, the error probability will be greater than* $1/2$ *even in the case of very high SNR, since* $\Lambda(\mathbf{x}_0) = \Lambda(\mathbf{x}_3)$*. However, if* $\mathbf{x}_1$ *or* $\mathbf{x}_2$ *is transmitted, the error probability is difficult to compute, but since it is zero in the noiseless case, it approaches zero in High SNR. This code clearly does not satisfy the UEP.*

In [1], sufficient conditions for a code to satisfy UEP are given for the case of a coherent receiver in which case the error probability is determined by the Euclidean distance.

[1] $\mathbf{A}^{\dagger} = (\mathbf{A}^T)^*$

Another approach is found in [2]. We will generalize the idea for general noncoherent receiver (with arbitrary metric) and show a way to construct multidimensional coded modulations such that UEP will hold. These codes will be called "Linear Noncoherent Coded Modulation" (LNCM). If the codes can be described by a trellis diagram, we can appropriately name them "Linear Noncoherent Trellis Coded Modulation" (LNTCM). Here, "linear" does not mean that binary arithmetic is used. Linear codes over groups are a special case of group codes. A good analysis of group codes is found in [3]. For LNTCM codes, linearity guaranties UEP, the same as in the cases of the BSC and binary linear codes. We will treat the case of equal-energy symbols transmitted over an AWGN channel. We use a general correlator-based receiver, of which a coherent receiver is a special case. The use of such codes not only simplifies the design and analysis of a coded system significantly, but it also enables us to solve the problem of noncoherently catastrophic codes (see next chapter). With some limitations, coded PSK, differential PSK and coded differential PSK are included as a special case in the family of LNCM. The theorems presented here are especially important in the analysis and design of multidimensional coded modulations which are to be decoded noncoherently with multiple symbol observations; refer to [4]–[7] and the method proposed in this thesis.

A large class of receiving systems, either coded or uncoded, can be represented as a complex correlator front end and a decoder that uses the output of the correlator for making its decisions; see Chapter 2 for details. This class includes all the coherent modulations and optimum noncoherent detection of DPSK, MFSK, CPM, multiple symbol detection of DPSK and many others [9]–[13]. We will use this model in the following analysis.

3.2 Determination of the Error Probability

In our model the decoder makes its decisions by maximizing some function $f(\underline{\mu})$ of the outputs of the correlator. f is an arbitrary real valued function. The decoder chooses $\hat{\mathbf{x}}$ which maximizes

$$\Lambda(\hat{\mathbf{x}}) = f(\mathbf{r}_0^\dagger\hat{\mathbf{x}}_0, \mathbf{r}_1^\dagger\hat{\mathbf{x}}_1, \cdots, \mathbf{r}_{N-1}^\dagger\hat{\mathbf{x}}_{N-1}). \tag{3.2}$$

We can write equation (3.2) in a more convenient form as

$$\Lambda(\hat{\mathbf{x}}) = f(\{\mathbf{r}_i^\dagger \hat{\mathbf{x}}_i\}). \tag{3.3}$$

We constrain ourselves to the case of equal energy symbols, such that $\mathbf{s}^\dagger \mathbf{s} = E$ for every symbol $\mathbf{s}$. For these systems we can prove the following theorem.

Theorem 3.2.1 *The pairwise error probability between two codewords* $\mathbf{x}$ *and* $\mathbf{y}$ *assuming* $\mathbf{x}$ *is transmitted is completely determined by* E*, the noise variance and the sequence of symbol-by-symbol complex correlations* $\rho_i = \mathbf{x}_i^\dagger \mathbf{y}_i$.

Proof: Let $\mathbf{x}_i$, $i = 0, \cdots, N-1$ be the transmitted signal and $\mathbf{r}_i = \mathbf{x}_i + \mathbf{n}_i$ be the received signal components, where $\mathbf{n}_{i,j}$, $i = 0, \cdots, N-1$, $\quad j = 0, \cdots, D-1$ are complex Gaussian i.i.d. (independent and identically distributed) random variables with zero mean and variance σ^2. The probability that the decoder makes an incorrect decision on sequence $\mathbf{y}$is

$$P_e(\mathbf{x}, \mathbf{y}) = P_r[f(\{\mathbf{r}_i^\dagger \mathbf{x}_i\}) < f(\{\mathbf{r}_i^\dagger \mathbf{y}_i\})] =$$

$$= P_r[f(\{\mathbf{x}_i^\dagger \mathbf{x}_i + \mathbf{n}_i^\dagger \mathbf{x}_i\}) < f(\{\mathbf{x}_i^\dagger \mathbf{y}_i + \mathbf{n}_i^\dagger \mathbf{y}_i\})]. \tag{3.4}$$

Note that $\tilde{\mathbf{n}}_{1,i} = \mathbf{n}_i^\dagger \mathbf{x}_i$ and $\tilde{\mathbf{n}}_{2,i} = \mathbf{n}_i^\dagger \mathbf{y}_i$ are two zero mean complex Gaussian vectors, composed of independent r.v.'s. As such, their joint probability distribution is completely determined by their components' variances and the cross-correlation of their components:

$$\sigma^2_{\tilde{\mathbf{n}}_{1,i}} = \sigma^2 \mathbf{x}_i^\dagger \mathbf{x}_i = \sigma^2 E, \quad \sigma^2_{\tilde{\mathbf{n}}_{2,i}} = \sigma^2 \mathbf{y}_i^\dagger \mathbf{y}_i = \sigma^2 E, \tag{3.5}$$

$$C_{\tilde{\mathbf{n}}_{1,i}, \tilde{\mathbf{n}}_{2,i}} = E[\tilde{\mathbf{n}}_{1,i}^* \tilde{\mathbf{n}}_{2,i}] = E[\mathbf{x}_i^\dagger \mathbf{n}_i \mathbf{n}_i^\dagger \mathbf{y}_i] = \sigma^2 \mathbf{x}_i^\dagger \mathbf{y}_i = \sigma^2 \rho_i, \tag{3.6}$$

$$C_{\tilde{\mathbf{n}}_{1,i}, \tilde{\mathbf{n}}_{2,j}} = 0 \quad \forall i \neq j \tag{3.7}$$

We conclude that $P_e(\mathbf{x}, \mathbf{y})$ depends only on E, σ^2 and $\rho_0, \rho_1, \cdots, \rho_{N-1}$. □

If $\mathbf{a}$ and $\mathbf{b}$ are two other codewords, which satisfy

$$\mathbf{a}_i^\dagger \mathbf{b}_i = \mathbf{x}_i^\dagger \mathbf{y}_i \quad \forall i \tag{3.8}$$

then ρ_i are equal for the two pairs of sequences, so

$$P_e(\mathbf{a}, \mathbf{b}) = P_e(\mathbf{x}, \mathbf{y}). \tag{3.9}$$

By extending this theorem, it can be shown that the total error probability, when a specific output sequence $\mathbf{x}^{(m)}$ is transmitted, is a function, g_m, of all the pairwise correlations of all possible output sequences. We can write the above statement by

$$P_e(\mathbf{x}^{(m)}) = P_r\{max_{\forall j \neq m}[f(\{\mathbf{r}_i^\dagger \mathbf{x}_i^{(j)}\})] > f(\{\mathbf{r}_i^\dagger \mathbf{x}_i^{(m)}\})\} = g_m(\{\bar{\rho}(u^{(k)}, u^{(l)})\}), \quad (3.10)$$

where $u^{(k)}$ and $u^{(l)}$ are two input sequences encoded to $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(l)}$, $\bar{\rho}(u^{(k)}, u^{(l)})$ denotes the vector $(\cdots, \mathbf{x}_{i-1}^{(l)\dagger}\mathbf{x}_{i-1}^{(k)}, \mathbf{x}_i^{(l)\dagger}\mathbf{x}_i^{(k)}, \mathbf{x}_{i+1}^{(l)\dagger}\mathbf{x}_{i+1}^{(k)}, \cdots)$ and $\{\bar{\rho}(u^{(k)}, u^{(l)})\}$ denotes the ordered set of all possible combinations of k and l with $k \neq l$. For coherent demodulation of constant energy signals [9], the optimal f is

$$f(\{\mathbf{r}_i^\dagger \mathbf{x}_i\}) = \sum_{i=0}^{N-1} \mathrm{Re}\{\mathbf{r}_i^\dagger \mathbf{x}_i\}. \quad (3.11)$$

In this case only the real part of ρ_i is of interest, thus the condition $\mathrm{Re}\{\mathbf{a}_i^\dagger \mathbf{b}_i\} = \mathrm{Re}\{\mathbf{x}_i^\dagger \mathbf{y}_i\}$ is sufficient for equation (3.9) to hold. Since the last requirement is weaker than equation (3.8), a coded modulation might have the UEP for coherent decoding and not satisfy the UEP for other types of decoders; see Example 3.1.1.

We believe that most practical decoders will have the property that the error probability is the same if we conjugate all the correlation sequences, i.e., the function f will get the same value if we conjugate all its arguments. We will call this property "conjugate-symmetry."

Theorem 3.2.2 *A code with only two codewords satisfies UEP if a conjugate-symmetric decoder is used.*

Proof: The error probability depends on the sequence of correlations $\rho_i = \mathbf{x}_i^\dagger \mathbf{y}_i$. Now if we exchange $\mathbf{x}$and $\mathbf{y}$we get ρ_i^*. Since the decoder is conjugate-symmetric we get the same error probability for transmitting $\mathbf{x}$and deciding on $\mathbf{y}$and for transmitting $\mathbf{y}$and deciding on $\mathbf{x}$. □

3.3 Algebraic Framework and the Construction of Linear Noncoherent Codes Satisfying UEP

Linear noncoherent codes will denote the codes which linearity implies UEP when decoded with a general noncoherent decoder (and also with coherent decoder as a special case). In order to facilitate the code construction, we will define a special algebraic structure. As we will show, the benefit of working with this algebraic framework is that every linear code also becomes a UEP one. First, we would like to write the definition of linear codes. Our definition is similar to [1], but we do not have to limit the input alphabet to be binary and the operations do not have to be performed over GF(2). For the input alphabet I, we will define a group operation denoted by '$\oplus$' under which I forms an Abelian group, denoted by

$$\mathcal{I} = \{I, \oplus\}. \tag{3.12}$$

If I is binary then $\oplus$ is the XOR operation. For the output alphabet O, we define another group operation, denoted by '$\circ$', under which O forms an Abelian group denoted by $\mathcal{O} = \{O, \circ\}$. The operations '$\oplus$' and '$\circ$' can be extended to apply to sequences of input or output symbols correspondingly, by performing them symbol-by-symbol.

Definition 3.3.1 *A code γ, which is a mapping from I^N to O^M, is said to be linear if and only if for any two input sequences $u, v \in I^N$,*

$$\gamma(u \oplus v) = \gamma(u) \circ \gamma(v), \tag{3.13}$$

In other words, γ is a homomorphism from I^N to O^M.

Definition 3.3.2 *A correlation group $\mathcal{U}$ is an Abelian group $\{O, \circ\}$ with alphabet $O \subset C^D$ (complex vectors of D components) such that for all $A, B, C \in O$:*

a. *The operation, '$\circ$', and the inner product (sum of the component by component complex multiplication) denoted by '$\cdot$' satisfy*

$$A \cdot (B \circ C) = (A \circ B) \cdot C. \tag{3.14}$$

b. *A^* which is the complex conjugate of the vector A, is a member of the group $\mathcal{U}$.*

c.

$$A \circ (A^*) = \mathcal{Z}, \tag{3.15}$$

where $\mathcal{Z}$ is the zero element of the group $\mathcal{U}$.

The reason for calling $\mathcal{U}$ a correlation group will be clarified in theorem 3.3.3. The notation $(s_0, s_1, \cdots, s_{D-1})$ will be used for describing an element of O which is a complex vector, where s_i are complex numbers. The energy of an element is defined as $E(A) = A \cdot A^*$ where $A \in \mathcal{U}$. Let us now show some consequences of definition 3.3.2.

Theorem 3.3.1 *All the elements of $\mathcal{U}$ have equal energy.*

Proof:

$$A \cdot A^* = A \cdot (\mathcal{Z} \circ A^*) = \mathcal{Z} \cdot (A \circ A^*) = \mathcal{Z} \cdot \mathcal{Z} = E, \tag{3.16}$$

where E is a constant. □

Theorem 3.3.2 *$\mathcal{Z}^* = \mathcal{Z}$, i.e., the components of $\mathcal{Z}$ are real numbers.*

Proof: From (c) $\mathcal{Z} \circ (\mathcal{Z}^*) = \mathcal{Z}$, but $\mathcal{Z}$, as the zero element, satisfies $\mathcal{Z} \circ \mathcal{Z} = \mathcal{Z}$ so $\mathcal{Z} = \mathcal{Z}^*$. □

The following theorem is the most important consequence of the definition of the correlation group $\mathcal{U}$.

Theorem 3.3.3 *Let X, Y and Z be elements of the correlation group $\mathcal{U}$ and let $Y = X \circ Z$, then $Y \cdot X^* = Z \cdot \mathcal{Z}$. This can be written also as $X^\dagger Y = \mathcal{Z}^\dagger Z$.*

Proof:

$$Y \cdot X^* = (X \circ Z) \cdot X^* = Z \cdot (X \circ X^*) = Z \cdot \mathcal{Z} = Z \cdot \mathcal{Z}^*. \quad \Box \tag{3.17}$$

The following theorem shows that every linear code has UEP if its output symbols are taken out of a correlation group.

Theorem 3.3.4 *Let $\mathcal{U} = \{O, \circ\}$ be a correlation group, $\mathcal{I} = \{I, \oplus\}$ an Abelian group and $\gamma : I^N \mapsto O^M$ is a linear code, then γ has UEP when the symbols are transmitted over an AWGN channel and a correlator-based receiver is used for the decoding.*

Proof: Let $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ be the codewords for two input sequences $u^{(i)}$ and $u^{(j)}$ respectively, namely $\mathbf{x}^{(i)} = \gamma(u^{(i)})$, $\mathbf{x}^{(j)} = \gamma(u^{(j)})$. For any input sequence $u^{(i)}, u^{(j)} \in I^N$, there is a sequence $u^{(k)} \in I^N$ such that $u^{(k)} = u^{(j)} \ominus u^{(i)}$, where '$\ominus$' is the inverse of the operation '$\oplus$' ($u^{(j)} = u^{(i)} \oplus u^{(k)}$), and is performed term by term over the sequences. From the linearity, $\mathbf{x}^{(k)} = \gamma(u^{(k)})$ satisfies $\mathbf{x}^{(j)} = \mathbf{x}^{(i)} \circ \mathbf{x}^{(k)}$. Let $\mathbf{x}^{(0)}$ be the codeword for the all-zero input sequence $u^{(0)}$. As such, $\mathbf{x}^{(0)}$ is the all-$\mathcal{Z}$ sequence. By theorem 3.3.3,

$$\mathbf{x}_t^{(i)\dagger}\mathbf{x}_t^{(j)} = \mathbf{x}_t^{(0)\dagger}\mathbf{x}_t^{(k)} \quad \forall t. \tag{3.18}$$

We can write equation (3.18) as

$$\bar{\rho}(u^{(j)}, u^{(i)}) = \bar{\rho}(u^{(j)} \ominus u^{(i)}, u^{(0)}), \tag{3.19}$$

where $\bar{\rho}$ is defined in equation (3.10). The same equality as equation (3.10) will hold if we exchange the labeling of the free variables in the equation. We use the exchange rules $u^{(n)} \rightarrow u^{(n)} \ominus u^{(m)}$ and $u^{(m)} \rightarrow u^{(0)}$, and get

$$P_e(\mathbf{x}^{(0)}) = g_m(\{\bar{\rho}[(u^{(k)} \ominus u^{(m)}), (u^{(l)} \ominus u^{(m)})]\}) =$$

$$= g_m(\{\bar{\rho}[(u^{(k)} \ominus u^{(m)}) \ominus (u^{(l)} \ominus u^{(m)}), u^{(0)}]\}) =$$

$$= g_m(\{\bar{\rho}(u^{(k)} \ominus u^{(l)}, u^{(0)})\}) = g_m(\{\bar{\rho}(u^{(k)}, u^{(l)})\}) = P_e(\mathbf{x}^{(m)}) \tag{3.20}$$

and thus the error probability for each transmitted sequence is the same. The UEP is satisfied. □

For the coherent case, [1] defined the term superlinearity, which guarantees UEP for the Euclidean distance. Since any linear noncoherent code satisfies UEP for a general correlator-based receiver, it is plausible that it is also superlinear w.r.t. Euclidean distance (but not necessarily otherwise). Remember that the Euclidean distance is used for a coherent decoder which is a special case of a correlator-based receiver. We will show this, but first we will repeat the definition of superlinearity.

Definition 3.3.3 *A superlinear code is a linear code in which it is possible to associate with each symbol $A \in O$ a real number $W(A)$ called symbol weight, so that the square of the distance between A and B satisfies $d^2(A, B) = W(A \circ B), \quad \forall A, B \in O$. The input in this case is restricted to be binary, i.e., the input group is GF(2).*

Theorem 3.3.5 *Every linear noncoherent code with binary input is superlinear.*

Proof: Taking the Euclidean distance as $d^2(A,B) = |A-B|^2$ (where '$-$' is an ordinary subtraction, remember that A and B are also complex vectors) we get

$$d^2(A,B) = |A-B|^2 = (A-B)^\dagger(A-B) =$$

$$A^\dagger A + B^\dagger B - B^\dagger A - A^\dagger B = 2E - 2\text{Re}\{A^\dagger B\}. \tag{3.21}$$

The input is binary, so for every codeword $\mathbf{x} = \gamma(u)$, $\mathbf{x} \circ \mathbf{x} = \gamma(u \oplus u) = \gamma(0)$, thus for every symbol A, $A \circ A = \mathcal{Z}$. From this, if $C = A \circ B$ then $B = A \circ C$. Using theorem 3.3.3 on the last relation we get

$$A^\dagger B = \mathcal{Z}^\dagger C. \tag{3.22}$$

Defining $W(A) = 2E - 2\text{Re}\{\mathcal{Z}^\dagger \mathcal{A}\}$ we get

$$W(A \circ B) = W(C) = 2E - 2\text{Re}\{\mathcal{Z}^\dagger C\} = 2E - 2\text{Re}\{A^\dagger B\} = d^2(A,B) \tag{3.23}$$

as required. □

We will now present a few examples to clarify the correlation group definition.

Example 3.3.1 *(Used for rate $\frac{1}{2}$ coded BPSK) Let $D = 2$ and our group of vectors (The output alphabet O) be the pairs $(1,1)$, $(-1,1)$, $(-1,-1)$, $(1,-1)$. Each of the pairs is one output symbol. In a practical implementation it may be a transmission of two orthogonal waveforms each with an appropriate sign or a serial transmission of two BPSK symbols. By doing so we get a rate $\frac{1}{2}$ coded BPSK transmission. The $\mathcal{Z}$ symbol is $(1,1)$ and the output group operation is defined as*

$$(a,b) \circ (c,d) = (ac, bd),$$

where a,b,c,d are each $+1$ or -1. The above definition represents an Abelian group. We want to show that it is also a correlation group:

$$(a,b) \cdot [(c,d) \circ (e,f)] = abc + bdf = (c,d) \cdot [(a,b) \circ (e,f)].$$

$(a,b)^* = (a,b)$ since a and b are real.

$$(a,b) \circ (a,b)^* = (a^2, b^2) = (1,1) = \mathcal{Z}. \tag{3.24}$$

Thus, this is a correlation group.

Example 3.3.2 *(MPSK modulation): Let $D = 1$, so that our vectors are merely complex numbers. In addition, let the output alphabet be of the form*

$$A_j = e^{j\frac{2\pi}{R}n}, \quad n = 0, \ldots, R-1, \tag{3.25}$$

where R (R=M for MPSK) is a positive integer. Let us define the group operation '$\circ$' as ordinary complex multiplication so that we can write $A_i \circ A_n = A_{(i+n) \bmod R}$. The zero element is $A_0 = 1$. Since scalar multiplication reduces to complex multiplication, and the last is equivalent to '$\circ$', condition **a** *of Def. 3.3.2 is satisfied. $A_n^* = A_{R-n}$ is a member of the group, and $A_n \circ A_n^* = A_n \circ A_{R-n} = A_0$. We conclude that this group is a correlation group.*

Example 3.3.3 *Let $D = 2$. We choose our group of vectors to be $(\pm 1, 0)$, $(\pm j, 0)$, $(0, \pm 1)$, $(0, \pm j)$, a total of 8 possible symbols. Let us denote the vector $(a, 0)$ by a^0 and $(0, a)$ by a^1. If two frequencies are used as the two orthogonal functions φ_0,φ_1, then the output of the encoder will result in an FSK-like waveform. Unlike in the case of noncoherent FSK, there is a certain phase relation between the symbols. Let us define the operation '$\circ$' as*

$$a^x \circ b^y = (ab)^{x \,\mathrm{XOR}\, y}, \tag{3.26}$$

where x and y get the values 0 or 1, then, as an example, $j^1 \circ j^1 = (-1)^0$. The zero element is $\mathcal{Z} = 1^0$. Let us show that this is a correlation group.
For any $a, b, c \in \{\pm 1, \pm j\}$

$$a^x \cdot (b^y \circ c^z) = a^x \cdot (bc)^{y \,\mathrm{XOR}\, z} = \begin{cases} abc, & \text{if } x = y \,\mathrm{XOR}\, z; \\ 0, & \text{otherwize.} \end{cases} \tag{3.27}$$

On the other hand

$$b^y \cdot (a^x \circ c^z) = b^y \cdot (ac)^{x \,\mathrm{XOR}\, z} = \begin{cases} abc, & \text{if } y = x \,\mathrm{XOR}\, z; \\ 0, & \text{otherwize.} \end{cases} \tag{3.28}$$

Since the condition $x = y \,\mathrm{XOR}\, z$ is equivalent to $y = x \,\mathrm{XOR}\, z$ then

$$a^x \cdot (b^y \circ c^z) = b^y \cdot (a^x \circ c^z), \tag{3.29}$$

thus condition **a** *of definition 3.3.2 is satisfied. Condition* **b** *is trivial and condition* **c** *is satisfied by $a^x \circ (a^*)^x = (aa^*)^0 = 1^0 = \mathcal{Z}$, therefore, we have a correlation group. It is* *also possible to generalize this group definition to cases where we have more frequencies or more phases.*

Next, we want to construct UEP convolutional coded modulation. We will achieve this goal by using the following well known construction [9], which results in linear convolutional code over a correlation group. The structure generates rate $1/D$ codes but the generalization to rate n/D codes for any integer n is straightforward.

Let K be the constraint length of the convolutional code, D the number of complex dimensions of the modulation, B the number of input bits per trellis branch, $R = 2^B$ the number of branches reaching a node, and $N = R^{K-1}$ the number of states. Let the input group $\{I, \oplus\}$ be the set of integers modulo R, Z_R, and the $\oplus$ operation be addition modulo R. For encoding we use a shift register of K stages, each stage contains an integer between 0 to $R-1$. Each branch of the trellis is assigned a number in the following manner. Let the current state and the next state be represented by numbers in base R, then the branch number (in base R) is the most significant digit of the current state followed by the digits of the next state. For example, we can write the branches from state *abcde* to state *gabcd* as

$$abcde \xrightarrow{gabcde} gabcd,$$

where the letters represent digits in base R (least significant $\cdots$ most significant). The trellis branch that connects state *abcde* with *gabcd* is assigned the number *gabcde*. This number will be used as the argument of the encoder output function val(x) described below. Let G_i, $i = 1, \cdots, K$, be a list of elements from a correlation group $\{O, \circ\}$ which also satisfy

$$RG_i \triangleq \underbrace{G_i \circ G_i \circ \cdots \circ G_i}_{R \text{ times}} = \mathcal{Z}, \tag{3.30}$$

i.e., the order of each G_i is R or a divisor of R. The encoder output symbol is defined as

$$\text{val}(p) = \sum_{i=1}^{K} p_i G_i \triangleq p_1 G_1 \circ p_2 G_2 \circ \cdots \circ p_K G_K, \tag{3.31}$$

where p_i are the digits of p in base R from least significant to most significant, and p is a branch enumerator. In (3.31) the product p_iG_i is to be interpreted by

$$\underbrace{G_i \circ G_i \circ \cdots \circ G_i}_{p_i \text{ times}}.$$

Since p_i is determined modulo R, and G_i satisfies equation (3.30), this definition is unambiguous. See Figure 3.1 for the encoder implementation.

Theorem 3.3.6 *The* val(p) *function is linear.*

Proof: Let $a, b \in I^K$ be two string of integers modulo R.

$$\text{val}(a) \circ \text{val}(b) = \sum_{i=1}^{K}(a_i \oplus b_i)G_i = \sum_{i=1}^{K}[(a_i \oplus b_i) \bmod R]G_i = \text{val}(a \oplus b). \qquad (3.32)$$

□

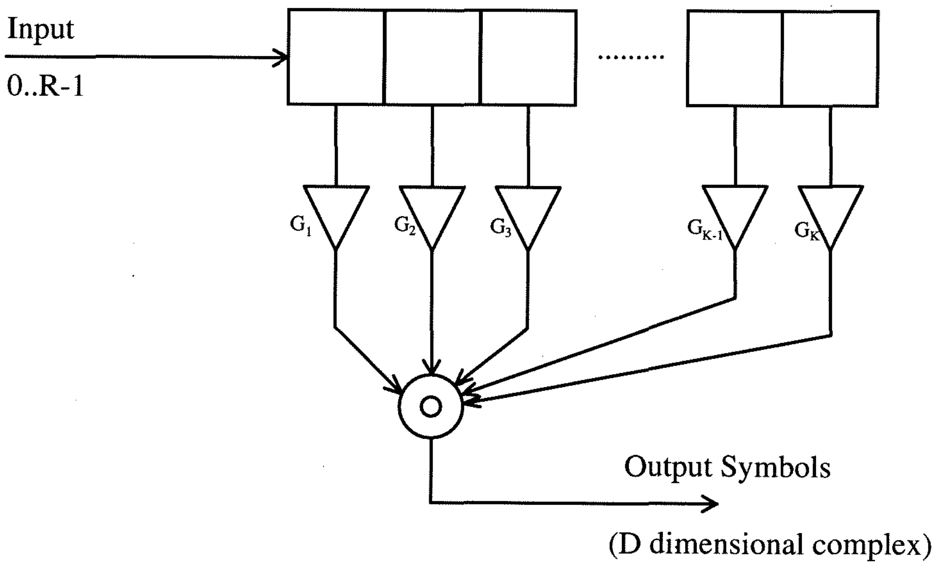


Figure 3.1: The trellis encoder implementation for LNCM.

Since the shift register is a linear operation, the defined code is linear. Some examples will clarify the code construction.

Example 3.3.4 *We use the correlation group defined in Example 3.3.1. With* $R = 2$ *equation (3.30) is satisfied, and the encoder input is binary. Let* $K = 3$ *(*$R^{K-1} = 4$

states) and $G_i = (-1,-1),(1,-1),(-1,-1)$. *This code has generators* 5 *and* 7 *(in octal) in the binary convolutional code description and has the maximal free distance of* 5 *[9, table 5.3.1]; see next sub-section.*

Example 3.3.5 *(Differential PSK): Let us use the correlation group defined in Example 3.3.2 to construct a code. This code has no redundancy because the number of possible output symbols is equal to the number of possible inputs per symbol. This code is noncoherently catastrophic but can be corrected, as described in Chapter 4, by an appropriate shuffling of the input bits according to the feedback encoder algorithm for the catastrophic codes. For* $K = 2$ *the corrected encoder is exactly the encoder for* M*-ary Differential Phase Shift Keying (MDPSK). For example, let* $R = 4$*, then* G_i *must be chosen out of the values* $1, j, -1$*, and* $-j$ *so all possible outcomes of* $\mathrm{val}(x)$ *are one of the values* $1, j, -1$*, and* $-j$*. We also choose* $K = 2$*,* $G_1 = j$*, and* $G_2 = -1$*. The resulting modulation, after the application of the algorithm in Chapter 4, is of Differential Quadratic PSK (DQPSK); see Figure 3.2. Note that although* 4 *states are required for the trellis representation, an optimal coherent decoder should use only one state, i.e., symbol by symbol decision. The explanation is that the* 4 *states encoder is not minimal; the minimal one has only one state.*

3.3.1 Convolutionally Coded PSK Modulation

For multiple-dimension modulation using PSK signals in each dimension, one possibility is to use the following correlation group: The vector $(a_1, a_2, \cdots, a_D)$ is composed of components of the form $e^{jn\frac{2\pi}{R}}$, where $n = 0, \cdots, R-1$ is an integer. The group operation '$\circ$' is defined as

$$(a_1, a_2, \cdots, a_D) \circ (b_1, b_2, \cdots, b_D) = (a_1 b_1, a_2 b_2, \cdots, a_D b_D) \tag{3.33}$$

and $\mathcal{Z} = (1, 1, \cdots, 1)$. In order to show that this group is a correlation group, we follow Example 3.3.1 and 3.3.2. Writing the elements of G as

$$G_i = \left(e^{j\frac{2\pi}{R}\alpha_{i,1}}, e^{j\frac{2\pi}{R}\alpha_{i,2}}, \cdots, e^{j\frac{2\pi}{R}\alpha_{i,D}}\right), \tag{3.34}$$

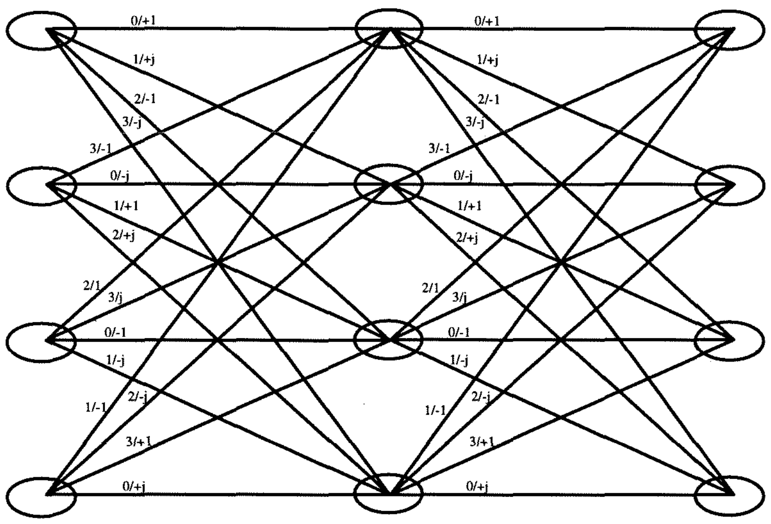


Figure 3.2: Trellis representation of DQPSK.

where $\alpha_{i,k}$ are integers in the range 0 to $R-1$, we find immediately that

$$\text{val}(a) = \sum_{i=1}^{K} a_i G_i = \left(e^{j\frac{2\pi}{R}\sum_{i=1}^{K} a_i \alpha_{i,1}}, e^{j\frac{2\pi}{R}\sum_{i=1}^{K} a_i \alpha_{i,2}}, \cdots, e^{j\frac{2\pi}{R}\sum_{i=1}^{K} a_i \alpha_{i,D}} \right). \tag{3.35}$$

Since the summations in each exponential expression can be performed modulo R, we recognize that our encoder is exactly the linear convolutional encoder, with its output mapped to PSK signals. However, we have to be careful about two things. First, the additions in the encoder implementation must be performed modulo R and not by XOR (binary additions are allowed only for $R = 2$). Second, the output of the code should be mapped linearly to the PSK signal. Note that the above conditions are sufficient but not necessary for UEP.

3.4 Conclusion

A sufficient condition for treating multidimensional coded modulation as linear (the code satisfies UEP) for a correlator-based receiver was found. Based on this condition, a construction method for a family of constant energy multidimensional convolutional coded modulations was described. By using this kind of codes, the analysis and design are considerably simplified. Also, among the codes that satisfy the UEP, there is a good chance to find high performance codes for noncoherent decoding as it is the case in other applications of coding.

As special cases, we saw that convolutionally coded BPSK and DPSK and uncoded MDPSK have UEP. For coded M-ary PSK we need both an appropriate encoder and mapping to symbols in order for UEP to hold.

The theorems described in this article are very important for the design of new coded modulation schemes which are made to be decoded by the emerging class of noncoherent decoders using multiple-symbol observation.

Bibliography

[1] Benedetto S., Marsan M.A., Albertengo G., Giachin E., "Combined Coding and Modulation: Theory and Applications," IEEE Trans. Info. Theory, Vol. 34, No. 2, pp. 223–236, Mar. 1988.

[2] Forney G.D., "Geometrically Uniform Codes," IEEE Trans. Info. Theory, Vol. 37, No. 5, pp. 1241–1260, Sep. 1991.

[3] Forney G.D., Trott M.D., "The Dynamics of Group Codes: State Spaces, Trellis Diagrams, and Canonical Encoders," IEEE Trans. Info. Theory, Vol. IT-39, No. 9, pp. 1491–1513, Sep. 1993.

[4] Divsalar D., Simon M.K., "Multiple-Symbol Differential Detection of MPSK," IEEE Trans. Comm., Vol. 38, No. 3, pp. 300–308, Mar. 1990.

[5] Divsalar D., Simon K.M., Shahshahani M. "The Performance of Trellis-Coded MDPSK with Multiple Symbol Detection," IEEE Trans. Comm., Vol. 38, No. 9, pp. 1391–1403, Sep. 1990.

[6] Leib H., Pasupathy S., "Optimal Noncoherent Block Demodulation of Differential Phase Shift Keying (DPSK)," AEU, Vol. 45, No. 5, pp. 299–305, Sep. 1991.

[7] Edbauer F., "Bit Error Rate of Binary and Quaternary DPSK Signals with Multiple Differential Feedback Detection," IEEE Trans. Comm., Vol. 40, No. 3, pp. 457–460, Mar. 1992.

[8] Divsalar D., Simon M.K., "Multiple Trellis Coded Modulation (MTCM)," IEEE Trans. Comm., Vol. 36, No. 4, pp. 410–419, Apr. 1988.

[9] Proakis J.G., *Digital Communications.* NY: McGraw Hill, 1989.

[10] Wozencraft J.M., Jacobs I.M., *Principles of Communication Engineering.* John Wiley and Sons, 1965.

[11] Stein S., "Unified Analysis of Certain Coherent and Noncoherent Binary Communication Systems," IEEE Trans. Info. Theory, Vol. IT-10, pp. 43–51, Jan. 1964.

[12] Fitz M.P., "Further Results in the Unified Analysis of Digital Communication Systems," IEEE Trans. Comm., Vol. 40, No. 3, pp. 521–532, Mar. 1992.

[13] Gagliardi R.M., *Introduction to Communications Engineering.* Wiley-Interscience, 1988.

Chapter 4

Noncoherently Catastrophic Codes

4.1 Introduction

Some codes cannot be decoded noncoherently. The simplest example is the memoryless code which maps input bits to BPSK symbols. For this case, there is no way to resolve the transmitted data without a phase reference. In other cases, even when the information is encoded in the phase of the symbol, it may be possible to resolve the data without a phase reference. The simplest example is DPSK, where the information is encoded in the phase differences between consecutive symbols. A decoder that "looks" at the phase difference between symbols does not need a phase reference. We see that noncoherent decoding can be achieved when encoding the data before transmission.

In contrast to a common belief, differential encoding is not a necessary operation to eliminate the need of a phase reference in a phase modulation. Many other encoders can be used for that purpose, including most of the binary convolutional encoders using BPSK modulation, which are commonly used in various applications. When these encoders are used, the only change that must be done to switch from coherent decoding to noncoherent is in the decoder, leaving the transmitter intact.

The property of a trellis-coded modulation that we define here, Noncoherent Catastrophicity (NC), prohibits noncoherent decoding of any kind. The other codes, for which noncoherent decoding is not prohibited, can in principle be decoded noncoherently. This means that there exists a decoder which can decode this code noncoherently. Such decoder should use in its decisions all of the phase transitions between symbols.

There can be other noncoherent decoders which cannot decode this code. For example, a decoder which does noncoherent demodulation on each symbol separately will not be able to decode any coded PSK modulation. However, any non-NC code can be decoded by the decoder defined in Chapter 6, which due to the overlapping of the observations uses all the phase transitions. A commonly used method in noncoherent communications with PSK symbols is differential encoding of the channel encoder's output. However, according to the above arguments, this differential encoder is not really necessary.

4.2 Definition of Noncoherently Catastrophic Codes and Discussion

Usually, catastrophic codes refer to convolutional codes in which an infinite number of decoding errors can happen as a result of a finite error event. This happens when two infinite length codewords, which have the same output for an infinite number of symbols, exist, and correspond to different input bits. We define NC codes as codes in which two infinite length codewords, which have the same output or a *constant phase difference* from each other, for an infinite number of symbols, exist, and correspond to input bits differing in infinitely many positions. In principle, codes which are non-NC can be detected noncoherently.

NC codes include the usual catastrophic codes as a special case. Every transparent binary convolutional code, i.e., one that inverts the input of the encoder to the output, is NC with BPSK modulation. When a code is not catastrophic in the usual sense, but is NC, we can equivalently say that the code has phase ambiguities. This means that if a coherent channel has phase ambiguities, this code cannot be decoded before resolving these ambiguities. A non-NC code on the other hand, if transmitted over a channel with phase ambiguities, can be decoded with no ambiguity. For example, suppose that a coherent channel has 180 degrees phase ambiguity. We can operate two decoders in parallel, one for each possible phase. We then choose the output of the decoder with the maximum metric. Over a sufficiently long time, the choice of the correct decoder becomes arbitrarily reliable. Another solution to phase ambiguities is to use rotationally transparent codes [1], codes which are insensitive to phase rotations. Here, we present a method which solves the problem of NC for the case of LNTCM codes. When used over coherent channels, there is no performance degradation of the probability of error events or increase in decoding complexity. There can be a small change in the bit error probability, since we change the bit assignments to the trellis branches.

4.3 Correction of Catastrophic Codes

For the LNTCM codes, almost every catastrophic code can be converted to a non-catastrophic one, by an appropriate shuffling of the input bits depending on the encoder states. We can already state here that if parallel transitions exist, with symbols that are phase shift of one another, this code cannot be "cured" without adding more states to the code (for example, by adding a differential encoder in front of the uncoded bits). The condition of which codes cannot be corrected will be clear shortly. We propose a feedback encoder, such that all the codewords that are a phase shifted version of other codewords are decoded to the same bits with no errors. The resulting code may be non-linear, but since it has the same output sequences as the original linear code, it will still have the UEP. The correct way of assigning input bits to the trellis branches is as follows.

We will call the codewords that happen to be a phase shift of the all-$\mathcal{Z}$ sequence "constant sequences." First, all the constant sequences are found. These can be found by a simple search. The encoder is assumed to have the structure of Figure 4.1, where the feedback function $O(x)$ is yet to be determined. Let $C_j, \quad j = 0, \cdots, P-1$ be the trellis states which belong to any constant sequence (including state 0 which is always a member of a constant sequence, namely the all-0 sequence of states). Let O_j be the input value that corresponds to the transition from the state C_j to the next state in the constant sequence which contains C_j. Let us separate the states into groups

$$P_k = \{x | x = a_k \oplus C_j, \quad j = 0, \cdots, P-1\},$$

where a_k is a non negative integer that defines the group P_k and the operation '$a \oplus b$' is performed by the input group operation applied between the digits of a and b in base R. Every state x will belong to one of the groups P_k and will have a corresponding j. We observe that there is a mapping $O(x)$ from x to O_j, defined mathematically as

$$O(x) = \{O_j | x \in P_k,\ C_j \oplus a_k = x\}. \tag{4.1}$$

In the modified encoder, we will add $O(x)$ to the input value (using '$\oplus$') when we are branching from state x. If multiple shift registers are used then $O(x)$ and O_j are vectors, with each component corresponds to one shift register. Notice that only in the rare situation of having two different constant sequences passing through the same state, $O(x)$ cannot be defined (or due to parallel transitions). In this case we will have to add more states to the code. See Figure 4.1 for the modified encoder implementation and Figure 4.2 for the flowchart for computing the function $O(x)$.

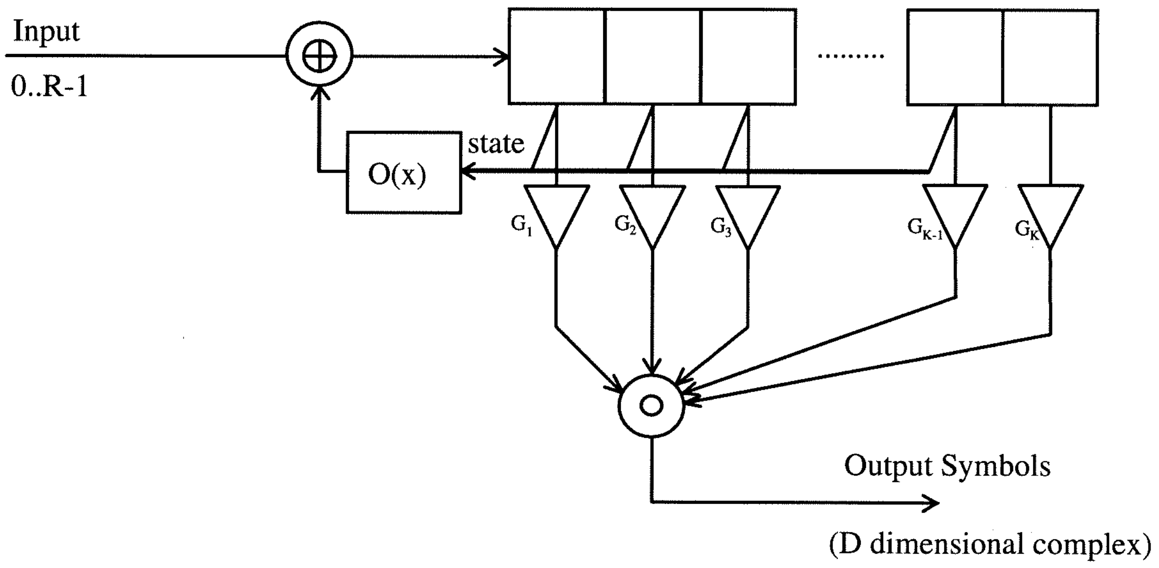


Figure 4.1: Modified encoder for correcting catastrophic codes.

Theorem 4.3.1 *If the encoder is corrected as described above, then every two possible output sequences which are a phase shift of each other are the encoding of equal input streams.*

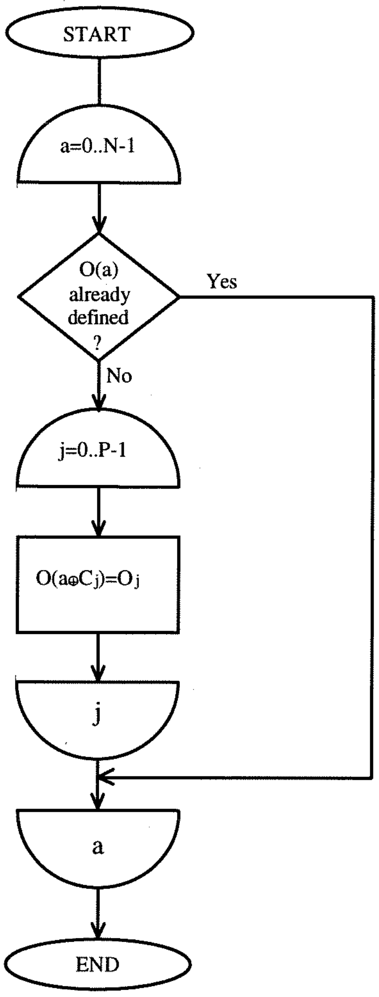


Figure 4.2: Flowchart for computing the function $O(x)$.

Proof: Let α and β be sequences of states which output symbols differ one from each other by a constant phase shift. Let i_α, i_β be the inputs to the encoder (before shuffling) at time t. From the linearity property, the sequence c, defined by the components $c_i = \alpha_i \ominus \beta_i$, is a valid codeword, where '$\ominus$' is the inverse of '$\oplus$'. By theorem 3.3.3 its output is a phase shift of the all-$\mathcal{Z}$ codeword, so c is a constant sequence, and there is m such that $C_m = C_t$. Let C_m be the state of c at time t. The state α_t belongs to one of the groups P_k and has a corresponding a_k and C_j. Let c' be the constant sequence which includes C_j, i.e., $C_j = c'_t$. From the linearity it follows that c'', which is defined by $c''_i = c'_i \ominus c_i$, is a constant sequence, there is a number n such that $C_n = c''_t$, and

$C_n = C_j \ominus C_m$. From $\alpha_t = a_k \oplus C_j$ and $\beta_t = \alpha_t \ominus C_m$, $C_n = C_j \ominus \beta_t$. Since $a_k \oplus C_n = \beta_t$ and C_n belongs to a constant sequence, then α_t and β_t belong to the same group P_k, and O_n will be used for shuffling the input for the sequence β at time t (O_n corresponds to C_n). Since $C_j = C_m \oplus C_n$, it follows that $O_j = O_m \oplus O_n$. The input to the shift register after adding $O(x)$ to the input is the input to the encoder of the uncorrected code and is $i_\alpha \oplus O_j$ for α and $i_\beta \oplus O_n$ for β. Since $\beta_i \oplus c_i = \alpha_i$, and O_m is the input value for c at time t, it follows that $O_m \oplus i_\beta \oplus O_n = i_\alpha \oplus O_j$. Using the fact that $O_j = O_m \oplus O_n$, we conclude that $i_\alpha = i_\beta$ so that α and β will be decoded to the same input bits, as desired. $\square$

Example 4.3.1 *Let us find the correction for the encoder which trellis is shown in Figure 4.3 and its implementation is shown in Figure 4.4. This encoder generates a rate* $\frac{1}{2}$*,* $K = 3$ *binary convolutional code with generators in octal* $1, 7$ *mapped to BPSK symbols. Equivalently,* $G_i = (-1,-1), (1,-1), (1,-1)$*. This code is transparent. The constant sequence that leads us to define this code as noncoherently catastrophic is the sequence of states* $\{...3,3,3,3...\}$*. Unless we make the correction, we will have an infinite number of errors when the sequence* $\{...3,3,3,3...\}$ *is received instead of the desired sequence* $\{...0,0,0,0...\}$*. The first step in the correction algorithm is to find* C_j *and* O_j*. The states that belong to any constant sequence are* 0 *and* 3*. With an arbitrary order we determine that* $C_0 = 0$ *and* $C_1 = 3$*. Also,* $O_0 = 0$ *and* $O_1 = 1$ *are the input values that correspondingly each brings us to the next state in the constant sequence. Setting* $a_0 = 0$*, we get the first group* $P_0 = \{0, 3\}$*. The rest of the states will belong to a second group* $P_1 = \{1, 2\}$ *with* $a_1 = 1$*. For each state, we determine the value of the corresponding offset* $O(x)$ *by determining to which group the state belongs and then what the corresponding* C_j *is:*

State	k	j	O_j
0	0	0	0
1	1	0	0
2	1	1	1
3	0	1	1

Finally,

$$O(x) = \begin{cases} 0, & x = 0, 1; \\ 1, & x = 2, 3. \end{cases}$$

The corrected encoder is found in Figure 4.4.

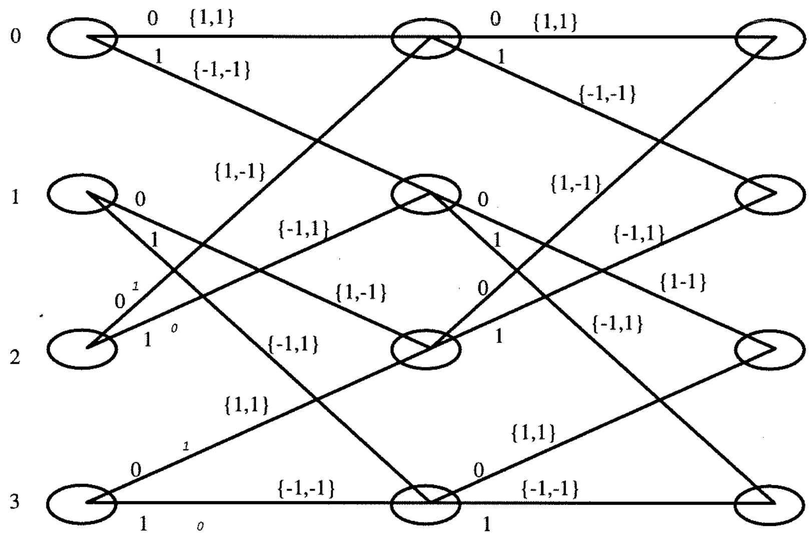


Figure 4.3: Trellis diagram of the code used in Example 4.3.1 for demonstrating an NC code. The constant sequences are ..., 0, 0, 0, 0, ... and ..., 3, 3, 3, 3, ... The corrected input assignments are shown in italic.

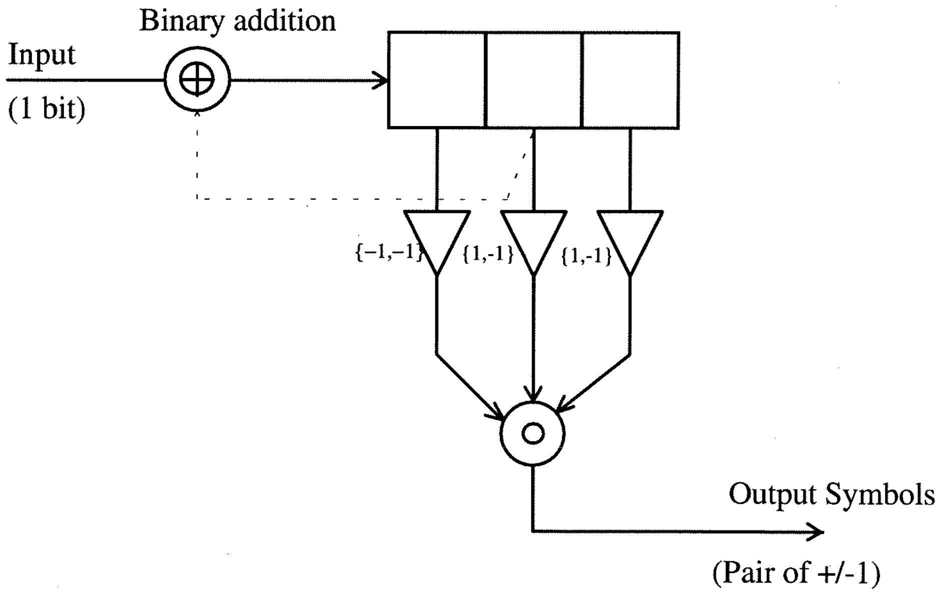


Figure 4.4: The encoder used in Example 4.3.1 with its correction (marked in dashed lines).

Bibliography

[1] Wei L.F., "Rotationally Invariant Trellis-Coded Modulations M-PSK," IEEE J. Selec. Comm., Vol. 7, No 9, pp. 1281–1295, Dec. 1989.

[2] Biglieri E., Divsalar D., McLane P.J., Simon M.K., "Introduction to Trellis Coded Modulation with Applications," NY: Macmillan, 1991.

Chapter 5

Improvement of Noncoherent Orthogonal Coding by Time-Overlapping

5.1 Introduction

Block orthogonal coding is a modulation technique that works either coherently or noncoherently [1, Sections 9.3 and 9.8]. Using either method of detection, we can approach the capacity of the AWGN channel as the number of symbols M grows [2, Section 4.3.2]. Noncoherent detection is used when the carrier phase is unknown at the receiver and no attempt is made to estimate its value.

M-ary block orthogonal coding is the optimal noncoherent modulation for the AWGN channel in the case in which decisions are made independently symbol by symbol [1, Section 9.8]. This modulation is simple to implement and has a relatively good performance, especially for large M. Each k bits of the input stream is assigned to one of possible 2^k signals. All the signals of the set are orthogonal to each other. The signals are transmitted over the AWGN channel and received by an optimal noncoherent receiver. As for now, we do not include coding in the transmission. Later on we will try to benefit from the use of coding.

The optimal M-ary receiver chooses the signal which corresponds to the maximum of the energy among the outputs of $M = 2^k$ matched filters, each matched to one of the possible transmitted signals. Specifically, the receiver finds i which maximizes

$$\left|\int_0^{T_s} x(t)s_i^*(t)dt\right|^2, \tag{5.1}$$

where $x(t)$ is the received waveform, $s_i(t), \quad i = 1 \ldots M$ are the set of the possible transmitted signals, and T_s is the symbol duration.

Let us now try to overlap the transmitted symbols in time. This means that the "tail" of the current symbol is a part of the next symbol and so on. By doing so, and choosing the appropriate set of orthogonal signals which can be overlapped, we will see that we actually can have an improvement in performance. This improvement, we show, comes in the expense of only a very small increase in the complexity of the receiver or the transmitter. The only difference in the receiver is that the integration window in equation (5.1) advances in time in smaller steps than T_s. The receiver, as before, makes decisions for each symbol independently. We will first show that improvement in performance over conventional noncoherent block orthogonal signaling can be actually achieved.

In order to describe the idea, let us take as an example the case of binary transmission ($k = 1$). DPSK can be described as the time-overlapping version of the binary orthogonal transmission. As a reminder, the probability of symbol error for FSK (or any other binary orthogonal signaling) is [3, table 3.1],

$$P_e = \frac{1}{2}e^{-\frac{E_b}{2N_0}}, \tag{5.2}$$

and for DPSK we have

$$P_e = \frac{1}{2}e^{-\frac{E_b}{N_0}}. \tag{5.3}$$

We can notice clearly the 3 dB improvement: half the $\frac{E_b}{N_0}$ is needed in DPSK relative to FSK to get the same error probability.

For DPSK the two transmitted signals are: $+A, +A$ or $+A, -A$ in baseband. The duration of those signals is $2T$, when $\frac{1}{T}$ is the bit rate. There is an overlapping in time between each signal and the one that follows. The amount of overlap is half of the signal duration as described in Figure 5.1. Because of the overlapping in time, we are transmitting with DPSK at twice the rate we would have transmitted without any overlapping. Yet the probability of error per bit still stays the same as in the non-overlapping case. This error probability is the error probability of FSK in which the symbol duration is $2T$.

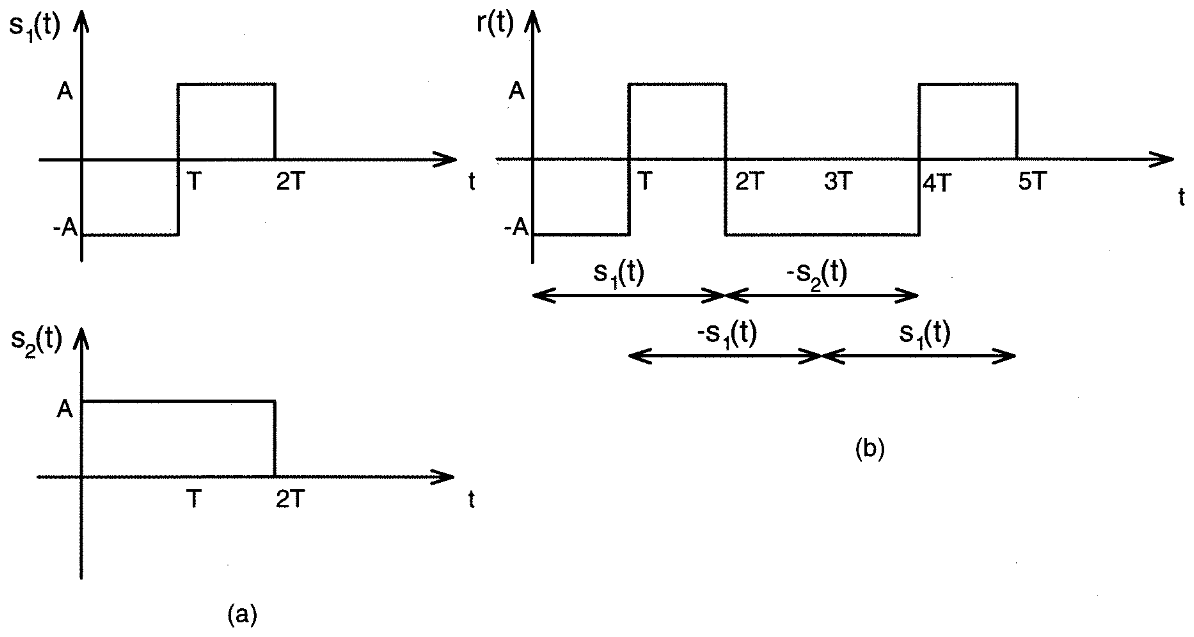


Figure 5.1: DPSK viewed as binary orthogonal modulation with time-overlapping. (a) The two orthogonal signals $s_1(t)$ and $s_2(t)$ that will be used to construct a DPSK waveform. (b) DPSK waveform in baseband, $r(t)$, is constructed by time-overlapping sequence from the set $s_1(t)$, $s_2(t)$.

The result is that without degrading any other aspect of performance, we get an improvement of 3 dB with DPSK over FSK. The 3 dB corresponds to the extended duration of the symbol: instead of T the duration is $2T$. However, the errors that will occur are no longer independent; they tend to happen in pairs, but this fact does not change the error probability per bit.

In the following sections we will generalize this notion to M-ary block orthogonal signaling and we will find an attainable bound to the improvement in $\frac{E_b}{N_0}$ for a constant probability of error.

5.2 Generalization to M-ary Transmission

Not any set of symbols can be overlapped. We will present the constraints that a set must satisfy and show that the set of the Walsh functions [6] satisfies those constraints and thus can be used to utilize the performance gain. It is also important to note that the phases of the transmitted symbols are not independent when using overlapped signals. This situation is similar to the case of continuous phase FSK (CPFSK).

Assume that the duration of the overlap in time between a given symbol and the following one is x. In order that any two consecutive independent signals be able to overlap in time, the set of signals must satisfy the following requirements:

1. For a window in time $0 \leq t \leq x$, all the signals must have the same waveform except for a possible phase shift.

2. For a window in time $T_s - x \leq t \leq T_s$, all the signals must have the same waveform except for a possible phase shift, where T_s is the symbol duration.

In order to understand the first requirement, let the signal $s_k e^{j\theta}$ be transmitted. Here s_k is a signal from the set and θ is a phase shift that we introduce to comply with the previous symbol. Now we want to transmit the next symbol, s_i. The time segment $0 \leq t \leq x$ of this signal was already transmitted, so we just have to transmit the rest of the signal. It is clear then that all the signals that we might transmit have to have the same beginning or "head." If s_i does not have the required beginning section, but only a phase shift of it, we can still transmit it if we properly phase shift the whole signal such that its beginning will be equal to what was already transmitted. We are free to choose any phase shift in the transmitter since the receiver is not sensitive to phase shifts (see equation 5.1).

The following orthogonal set is an example of a set that can be used for time-overlapped orthogonal signaling. Let $M = 2^k$ and let the set of orthogonal functions be the set of M Walsh functions [6]. This set of functions also achieves the upper bound that we will prove next, so it is an optimal set.

Walsh functions are a set of functions which are binary codes. In each time slot they assume a value of either $-A$ or $+A$. The functions are defined by the following equation:

$$s_i(t) = A \sum_{n=0}^{M-1} a_{n,i} P(t - nT_c), \quad i = 1 \ldots M, \tag{5.4}$$

where $P(t)$ is a rectangular pulse of duration T_c that will be called a *chip*. The $a_{n,i}$ can have the value $+1$ or -1 and are computed by the following formula:

$$a_{n,i} = \prod_{j=0}^{k-1} (-1)^{b_j(n) b_{k-1-j}(i)}, \tag{5.5}$$

where $b_j(m)$ is the j'th bit in the binary representation of m ($j = 0$ for the least significant bit). This set of functions has the desired orthogonality property, as shown in the cited reference.

We will always be able to overlap the first chip of a certain signal with the last chip of the preceding signal. For example, consider $M = 4$, i.e., two bits per symbol. Denote T_s as the symbol duration, T_c as the chip duration, and T as the effective symbol duration after overlapping, i.e., the non-overlapped symbol duration. The following relations hold:

$$T_s = 4T_c, \quad T = 3T_c. \tag{5.6}$$

Because of the overlapping by $\frac{1}{4}$ symbol, the performance gain over conventional 4FSK is $\frac{4}{3}$ which is 1.25 dB; see Figure 5.2.

Asymptotically, it can be shown that both the bit error probability of coherent BPSK, DPSK and 4FSK approach the same expression as in equation (5.3) in terms of the needed E_b/N_0 per given bit error probability. Thus, with $M = 4$ and overlapping we can do up to 1.25 dB better than DPSK and even better than BPSK.

For example, we take the point $P_b = 10^{-5}$ (P_b is the bit error probability). With DPSK we need $\frac{E_b}{N_0} = 10.34$ dB, with BPSK we need $\frac{E_b}{N_0} = 9.59$ dB. With set of 4 orthogonal signals (like 4FSK) we need $\frac{E_b}{N_0} = 10.61$ dB, but with the 1.25 dB improvement we need only 9.36 dB, which is better than BPSK by 0.23 dB.

5.2.1 Finding an Upper Bound

In this section, we will prove that the maximum gain possible is $\frac{M}{M-1}$. We are considering only equal energy signals. For the case of constant-envelope signals, this means that the maximum possible overlap in time is $\frac{1}{M}T_s$, where T_s is the symbol duration. Notice that in DPSK, $M = 2$, and the improvement is $\frac{2}{2-1} = 2$. According to the statement above, we cannot do better than this.

Assume that the duration of the overlap in time between a given symbol and the following one is x. The set of signals must satisfy the requirement that from $t = 0$ to $t = x$ they are the same waveform except for a possible phase difference. For the following discussion, without loss of generality, align (by phase shifting) the set of symbols such that there is no phase difference. Adding an arbitrary phase shift to the symbols does not affect their orthogonality, the only property that we are going to use.

Let us find the variation in the average power caused by the overlapping. The average power of the signal without overlapping is

$$P_0 = \frac{1}{T_s}\Big(\int_0^x \mid s_i(t) \mid^2 dt + \int_x^{T_s} \mid s_i(t) \mid^2 dt\Big). \tag{5.7}$$

The average power of the signal after overlapping is

$$P_1 = \frac{1}{T_s - x}\int_x^{T_s} \mid s_i(t) \mid^2 dt. \tag{5.8}$$

Here P_0 and P_1 are independent of i since we have equal-energy symbols.

Also let us define the improvement factor Γ to be the improvement in the signal energy performance due to the overlapping in time:

$$\Gamma = \frac{T_s}{T_s - x}\frac{P_0}{P_1} = \frac{\int_0^x \mid s_i(t) \mid^2 dt + \int_x^{T_s} \mid s_i(t) \mid^2 dt}{\int_x^{T_s} \mid s_i(t) \mid^2 dt} \tag{5.9}$$

$$= 1 + \frac{\int_0^x \mid s_i(t) \mid^2 dt}{\int_x^{T_s} \mid s_i(t) \mid^2 dt}. \tag{5.10}$$

Let us now calculate the correlation between a pair of different symbols. The correlation has to be 0 in order to satisfy the orthogonality of the set. Thus,

$$\rho = \int_0^{T_s} s_i(t)s_j^*(t)dt = 0, \quad \forall i \neq j. \tag{5.11}$$

Because the symbols are identical in the window of time $0 \leq t \leq x$ we will get

$$\rho_1 = \int_0^x | s_i(t) |^2 \, dt$$

as the contribution of this window to ρ (independent of i). The rest of the symbol, that is the section from x to T_s, contributes $\rho_2 = \int_x^{T_s} s_i(t)s_j^*(t)dt, \quad \forall i \neq j$. From equation (5.11), $\rho_1 + \rho_2 = \rho = 0$.

In general, for arbitrary equal energy symbols, the correlation between each pair of symbols from a set of M symbols has to satisfy

$$\rho \geq -\frac{1}{M-1}E, \tag{5.12}$$

where E is the energy of the symbol. This follows from the following well-known derivation:

$$0 \leq \int_0^{T_s} \left| \left(\sum_{i=1}^{M} s_i(t) \right) \right|^2 dt = \int_0^{T_s} \left[\sum_{i=1}^{M} | s_i^2(t) | + \sum_{j=1}^{M} \sum_{\substack{i=1 \\ i \neq j}}^{M} s_i(t)s_j^*(t) \right] dt \tag{5.13}$$

$$= ME + (M^2 - M)\rho = ME + M(M-1)\rho.$$

Solving for ρ, we get equation (5.12).

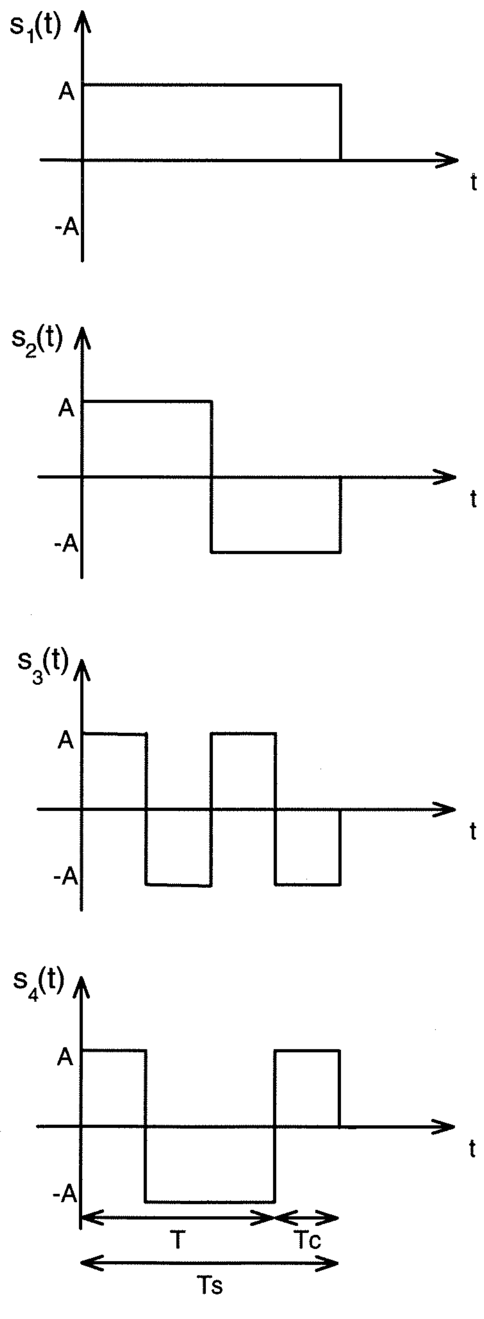
Codes which satisfy equation (5.12) with equality are called *Simplex Codes* [1] and they exist for each $M \geq 2$ [4]. By applying this theorem to the set of M signals which correspond to the window $x \leq t \leq T_s$ of the orthogonal symbols, we get

$$\rho_2 = -\rho_1 \geq -\frac{1}{M-1} \int_x^{T_s} | s_i(t) |^2 \, dt. \tag{5.14}$$

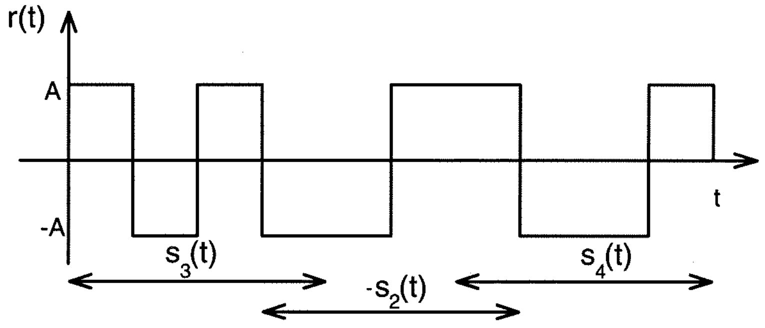
Now we find the desired upper bound:

$$\Gamma = 1 + \frac{\int_0^x | s_i(t) |^2 \, dt}{\int_x^{T_s} | s_i(t) |^2 \, dt} = 1 + \frac{\rho_1}{\int_x^{T_s} | s_i(t) |^2 \, dt} \leq 1 + \frac{1}{M-1} = \frac{M}{M-1}, \tag{5.15}$$

$$\Gamma \leq \frac{M}{M-1}.$$



(a)



(b)

Figure 5.2: 4-ary orthogonal modulation with time-overlapping using Walsh functions. (a) The 4 Walsh functions. (b) The transmitted signal using time-overlapping of 4 Walsh functions.

5.3 Trials with Hand-designed Codes

We have shown that it is possible to improve the performance of noncoherent orthogonal signaling by using a set of orthogonal signals which can be overlapped in time. The maximum possible improvement is given by

$$10 \log\left(\frac{M}{M-1}\right)$$

in dB. Although we have bounded the possible improvement, we have not considered the use of coding. Let us present simple code constructions, which enable us to virtually overlap by 50% a set of M orthogonal waveforms (M is a power of 2) by using a trellis structure. Let us begin with an example using $M = 4$. We begin by choosing a set of 4 orthogonal waveforms $s_0(t), \ldots, s_3(t)$, each defined for the duration of T seconds, where $T = 2T_b$, and is the span of one trellis branch. We place those waveforms on a 4-states trellis as shown in Figure 5.3a. The encoder is shown in Figure 5.3b.

We claim that this code is UEP. Let us construct the following correlation group. The group includes 8 elements, $s_0, -s_0, s_1, -s_1, s_2, -s_2, s_3, -s_3$, and uses the following rule for addition.

$$\alpha s_i \circ \beta s_j = \alpha\beta s_{i \oplus j}, \tag{5.16}$$

where α and β get the values $+1$ or -1, and '$\oplus$' is binary addition. This group is similar to the group of 3.3.3. If a bit represents the value of α or β, then their multiplication is equivalent to binary addition. We see that the addition rule between the group elements can be made equivalent to binary addition by assigning a binary word to each of the elements αs_i such that one bit field contains the sign α and the rest contain the index i. In other words, the output group is isomorphic to $(Z_2)^3$. Hence, a linear binary encoder, with an appropriate mapping of the output to group elements, satisfies UEP.

Let us use an observation covering 2 trellis branches ($L = 2$). Since the code satisfies UEP, without loss of generality, we can assume that the all-0 path is transmitted. We can look at paths originating from state 0 and ending at state 0. Let us investigate the shortest error events, those covering 2 branches of the trellis. In state 0, the decoder will choose the most likely among 4 such competitors, which carry the following lengthed

$2T$ signals: $\{s_0, s_0\}, \{-s_0, s_0\}, \{s_1, s_2\}, \{-s_1, s_2\}$. It is easy to verify that these 4 signals are orthogonal. Thus, we expect the decision reliability to be at least that of uncoded 4FSK with symbol energy equal to 4 times the bit energy instead of 2 times the bit energy in uncoded 4FSK. We have virtually overlapped by 50% 4 orthogonal symbols and gained 3 dB. This result is only asymptotically correct since we have to consider the contribution of error events longer than 2 branches. In the method described above, we can add 3 dB to the performance of uncoded MFSK for any M. However, the number of waveforms that we have to use is equal to $(\frac{M}{2})^2$ and the number of states is equal to M. For the case of $M = 4$, we do not increase the number of waveforms, use a very simple code, and still obtain results which are better than high complexity codes with DPSK modulation. However, for a large M, The use of a large number of waveforms adds to the system complexity and requires a large bandwidth.

Let us try to see if it is possible to use only $\frac{M}{2}$ waveforms. Refer to the trellis diagram of Figure 5.4. In this figure, for simplicity, the states related to the sign switching were grouped together to form super-states; also every branch in the figure represents two actual branches. This code looks promising, as all 4 signals carried by the shortest paths are orthogonal. However, a new kind of error event appears which is the sequence of super-states $\{\cdots, 0, 0, 0, 1, 1, \cdots\}$ carrying the output sequence $\{\cdots, s_0, s_0, s_1, s_0, s_0, \cdots\}$. So we lose our 3 dB advantage leading to a performance similar to uncoded 4PSK. Before rejecting this code, let us try to "fix" the problem. We will multiply the branches emerging from super-state 1 by $+j$ ($j = \sqrt{-1}$) every second symbol. The new encoder is shown in Figure 5.5. Forming this new time varying code, we did not change the distances between the error events that end in super-state 0—those stay orthogonal (provided that the decoder pre-multiplies the appropriate branches by $-j$). Note that the decoder has to be time synchronized to the sequence which multiplies the encoder output. This synchronization can be performed the same way as node synchronization in the Viterbi algorithm. By the modified encoder, the sequence $\{\cdots, s_0, s_0, s_1, s_0, s_0, \cdots\}$ is replaced by $\{\cdots, s_0, s_0, s_1, js_0, s_0, js_0, s_0, \cdots\}$ so its distance from the sequence $\{s_0, \cdots, s_0, \cdots\}$ is infinite. The error event $\{\cdots, s_0, s_1, s_0, s_1, s_0, \cdots\}$ is possible but its error probability is approximately the square of the error probability of uncoded 4FSK. For larger values of M it is required to generate $\frac{M}{2} - 1$ sequences, appropriately chosen, to multiply the encoder. Note that for a large M the number of error events of the form of the last mentioned may be large, and the gain will be reduced.

The actual performance results are shown in Figure 5.6 using the tools that will be developed later in Chapter 6. Also shown for reference, the performance of uncoded 4FSK, shifted 3 dB to the left. Note that the uncoded performance is exact, while the other curves are an upper union bound. They are tight only for low bit error rate. The decoding implementation details will also be covered in the following chapters.

The reason why we have a problem when using $\frac{M}{2}$ waveforms is that we want to map $\log_2 \frac{M}{2}$ bits to $\frac{M}{2}$ waveforms so no redundancy exist. If we increase the number of waveforms to M, we can use a rate $\frac{\log_2 M-1}{\log_2 M}$ binary code to map the input bits to the orthogonal waveforms except of one bit which feeds the differential encoder and chooses the sign of the waveform. Our only requirement from the code is that it will contain at least two consecutive non-zero branches in every error event, which can be easily satisfied with every short constrain length. We can extend the above ideas to the case of L larger than 2. This will require more states in the encoder, but will hopefully achieve very good results. For $L > 2$ we cannot have the benefit of the differential encoder, so we will eliminate it from the encoder. The minimum number of states in the encoder is M^L. The number of waveforms needed is $2M$ or larger (without the multiplying by $+j$ method). The codes to be used are those optimal for MFSK modulation in general. Let us now try another construction, which has much less bandwidth requirements and may have the potential for good performance.

Let us investigate the case where L, the observation length, is 4; K, the constraint length is 3; and R, the number of branches reaching a trellis state, is 4. Having four sequences of four symbols each, we can make them orthogonal one to each other. Let us place those four sequences on four possible paths on the trellis, originating from state 0 and ending at state 0 again. The UEP of the codes guarantees that every 4 paths of 4 symbols originating from some state A and reaching a state B are also orthogonal to each other. The decision at point B, which one of the sequences is the received one, is close to deciding which of possible four orthogonal waveforms was transmitted, and is equivalent to the decoding of MFSK for $M = 4$ with symbol energy equal to eight times the bit energy ($E_s = 8E_b$). The expected performance is 6 dB better than uncoded 4FSK. Let us choose the sequences such that they will be orthogonal to $\{1, 1, 1, 1\}$ (for our current discussion modulation is two-dimensional, i.e., each symbol is a complex number). The above sequences will be $\{1, j, -1, j\}$, $\{1, -1, 1, -1\}$, $\{1, -j, -1, -j\}$ and $\{1, -j, -1, -j\}$. In fact these four sequences are a Discrete Fourier Transform (DFT) matrix of dimension 4×4. One can immediately notice that each component of the sequences can be expressed as α_i^k, where i is the index within the sequence and k is the sequence number. The generator of this code is $G_i = j, -1, j$. Notice that there is no redundancy in this code because the number of possible symbols and the number of possible input combinations per symbol is equal and is 4. There is a need to add another two dimensions (so that the modulation will be four dimensional), i.e., $D = 2$.

We choose our group of vectors to be $\{\pm 1, 0\}$, $\{\pm j, 0\}$, $\{0, \pm 1\}$, $\{0, \pm j\}$, total of 8 possible symbols as in Example 3.3.3. It is shown in that example that this group is a correlation group. Let us test the following code (see explanations for the notations in Example 3.3.3):

$$G_i = -j^1, -1^0, j^1.$$

The sequences that are created when we start from state 0 and immediately return to state 0 are:

$$\begin{array}{lcrrrr}
S_0 & = & 1^0 & 1^0 & 1^0 & 1^0 \\
S_1 & = & 1^0 & -j^1 & -1^0 & j^1 \\
S_2 & = & 1^0 & -1^0 & 1^0 & -1^0 \\
S_3 & = & 1^0 & j^1 & -1^0 & -j^1
\end{array}$$

The first symbol in each sequence line is 1^0 and it belongs to the common branch of all four paths. Note that all four sequences are orthogonal, so we can expect a decision between them as good as in 4FSK with symbol energy four times larger than one symbol energy of our code. This will result in an extremely good performance. Here, we are only checking the case of the shortest error path. After further examination we concluded that the performance of the code was less than what was expected from the above discussion.

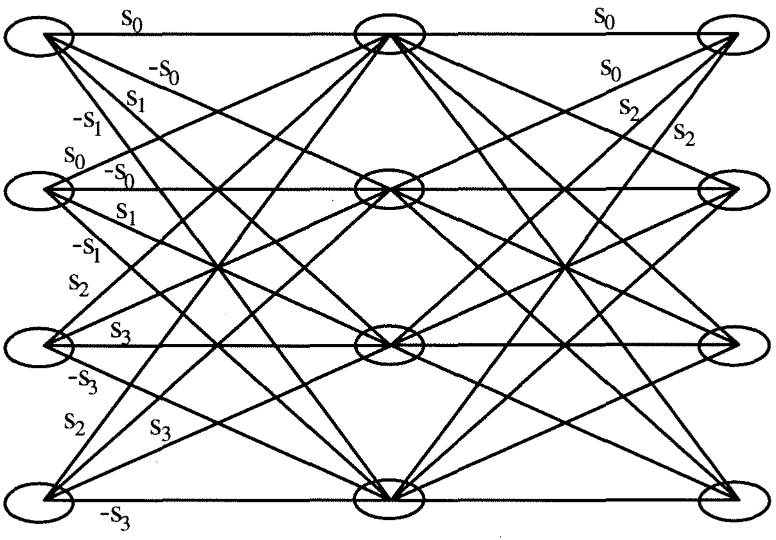
We argue that these design methods described above are sub-optimal. The reason is that these codes can always be generated by a regular convolutional encoder and PSK symbols. For example, a rate $\frac{1}{4}$ encoder with BPSK output, if restricted to have generators that get the values $G_i = \{+1,+1,+1,+1\}$,$\{+1,+1,-1,-1\}$, $\{+1,-1,-1,+1\}$ or $\{+1,-1,+1,-1\}$, will generate orthogonal outputs. Allowing G_i to be arbitrary, we enlarge our allowed codes number and hopefully find better codes. Due to the complexity of the code performance evaluation under noncoherent decoding, we have performed a systematic computer search. Surprisingly, some of the best codes found for optimum noncoherent decoding are also the best for coherent decoding. For those codes, it means that we can approach the coherent performance as the observation length increases.

5.4 Conclusion

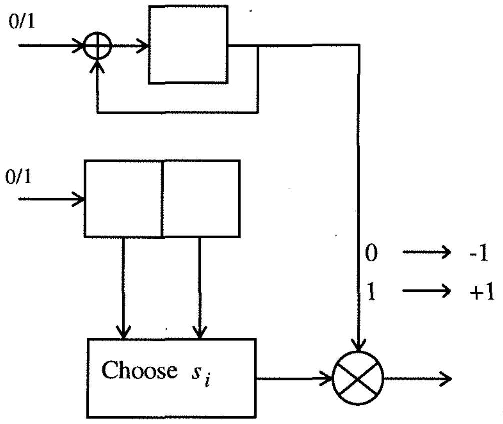
It is possible to improve the performance of noncoherent orthogonal signaling by using a set of orthogonal signals which can be overlapped in time. In particular, we suggest the use of the Walsh functions. The maximum possible improvement is given by

$$10\log(\frac{M}{M-1}) \tag{5.17}$$

in dB. This is valuable especially for small M. We note that with M=4, the performance of noncoherent block orthogonal transmission with overlapping is better than DPSK and even better than coherent BPSK for large SNR. This case might be particularly useful. When coding is included in the transmission, we see that it is possible to virtually overlap the orthogonal symbols to a much larger extent, leading to a large performance gain. However, we pay the price of higher complexity and in general wider bandwidth (overlapping in the uncoded case does not lead to an increase in complexity). The systematic approach presented in the next chapter will lead to higher performance codes with less bandwidth and complexity requirements.



(a)



(b)

Figure 5.3: A code for 4FSK which makes the shortest error sequences (of length 2) orthogonal. (a) The trellis diagram. (b) The encoder.

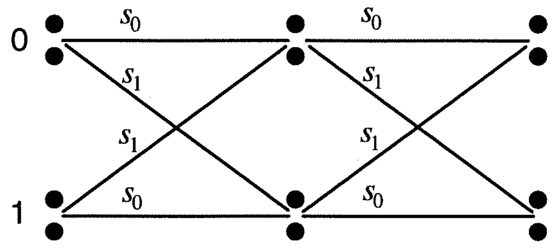


Figure 5.4: A code using two orthogonal waveforms.

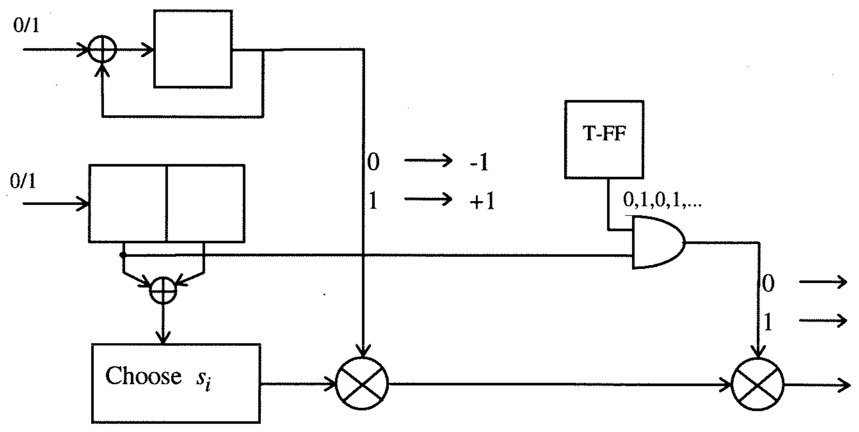


Figure 5.5: A time varying encoder for 2 orthogonal waveforms.

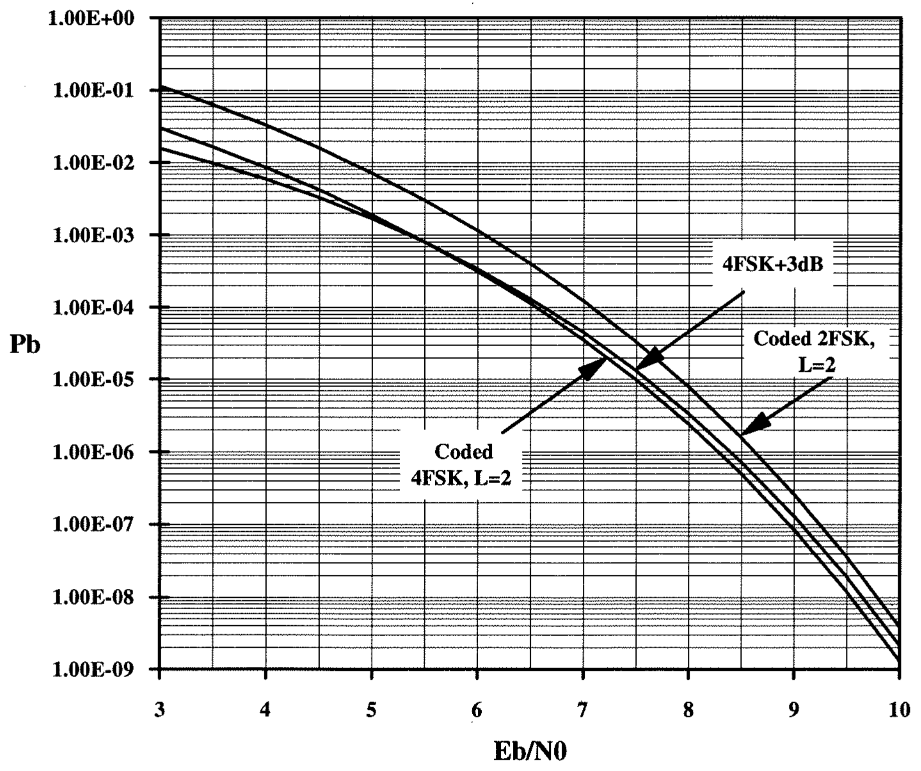


Figure 5.6: Performance of the codes described in this chapter. The encoder of Figure 5.3 which uses 4 orthogonal waveforms and the encoder of Figure 5.5 which uses 2 orthogonal waveforms are evaluated for an observation length L of 2 symbols.

Bibliography

[1] Gagliardi R.M., *Introduction to Communications Engineering.* Wiley-Interscience, 1988.

[2] Proakis J.G., *Digital Communications.* NY: McGraw Hill, 1989.

[3] Sklar b., *Digital Communications, Fundamental and Applications.* NJ: Prentice Hall, 1988.

[4] Taylor H., "On The Existence of Binary Simplex Codes," JPL Publication N78-15074, 4800 Oak Grove Dr., Pasadena, CA 91109, 1978.

[5] Golomb S.W., Baumert L.D., Easterling M.F., Stiffler J.J., Viterbi A.J., *Digital Communication with Space Applications.* NJ: Prentice Hall, 1964.

[6] Beauchamp K.G., *Walsh Functions and Their Applications.* NY: Academic Press, 1975.

Chapter 6

Definition and Analysis of Noncoherent Coded Modulation

6.1 Introduction

There are many advantages of using noncoherent decoding; these are summarized in Chapter 2. Currently, there is a gap between the performance of coded noncoherent systems and the coherent ones. For example, there is about 3 dB difference in performance on the AWGN channel between constraint length 7 rate 1/2, coded PSK and DPSK (differential PSK), even when interleaving is used for the DPSK [1]. Moreover, there is not much available tradeoff between bandwidth and power efficiency with currently used noncoherent modulations as it is with coherent ones.

Recently it has been shown that the performance of noncoherent detection can be improved by using a longer observation time[1] (one symbol for FSK and two symbols for differential detection). Multiple symbol noncoherent detection of uncoded DPSK is considered in [2],[3],[4] and multiple symbol noncoherent detection of uncoded CPM is considered in [5], [6], [7] and [8]. Application to block coded MSK is found in [7]; and a new block coded MPSK for noncoherent detection is considered in [9]. Multiple-symbol differential detection of trellis coded MDPSK is considered in [10] and [11]. Multiple-symbol noncoherent detection of trellis coded modulation has been confined to applications which use differential encoding. Also, no optimization of the code selection has been performed. In all of the above contributions, the observations are

[1]The observation is the time window in which the carrier phase is assumed to be constant.

independent. In the case of uncoded or block coded modulation, one observation was used for each decision. When convolutional coded modulations were decoded, independent observations were used. In some methods the observations overlap in one symbol and interleaving was used for making them independent. The interleaving function adds extra complexity to the system and causes unwanted delay in the received data, although in some applications such as fading channels, interleaving is required.

We show a new noncoherent decoding scheme which can be applied to almost all types of coded modulation, without changing the encoder, and with performance that in general approaches the coherent decoding of these codes as the observation length grows. We also show new codes with high noncoherent performance which are also optimal for coherent decoding (have maximum free distance), and their performance with the efficient noncoherent decoding.

Unlike the previous approaches, we use maximally overlapped observations (see Figure 6.1), which will not and cannot be made independent. We thus utilize the information from all possible (time shifted) observations of length T. When applying this technique to trellis coded modulation, each observation spans several branches of the code trellis. This can be viewed as a generalization of the ideas presented in Chapter 5. The resulting novel sequence estimator is derived, and its performance for equal energy symbols over the AWGN channel is found by the union bound. The bound is computed numerically. We show by examples and prove analytically that the performance of the noncoherent sequence estimator approaches the coherent one as the observation length grows. Also, we suggest the use of a type of coded modulation scheme, the Linear Noncoherent Trellis Coded Modulation (LNTCM), defined in Chapter 3, which exhibits Uniform Error Property (UEP) when used with our sequence estimator. The LNTCM uses constant energy symbols, e.g., MPSK. Note that the proposed decoding algorithm can be applied to many other codes and modulations. Its application to CPM (Continuos Phase Modulation) will be covered in Chapter 8.

The above efforts lead to a new approach for designing noncoherent systems. We call this coded modulation and detection scheme "Noncoherent Coded Modulation (NCM)" (see Figure 6.2). This scheme provides tradeoff between robustness with re-

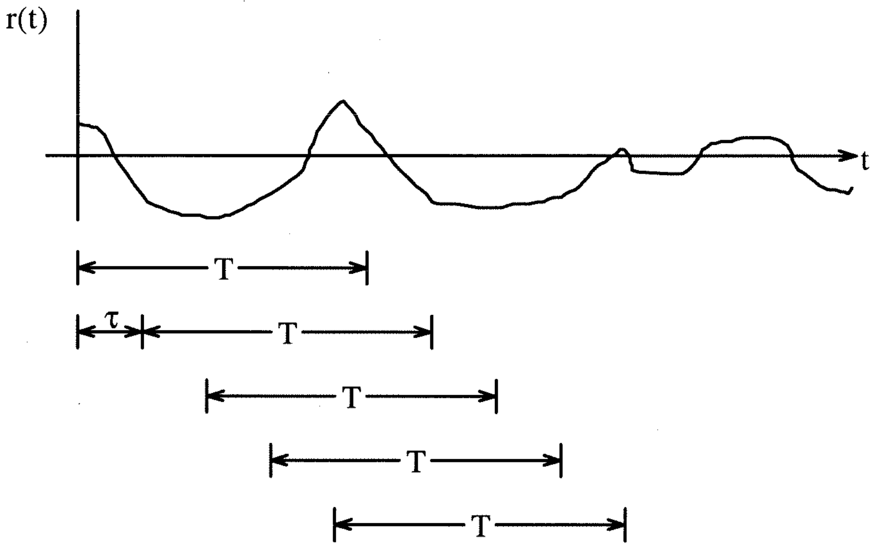


Figure 6.1: Demonstration of overlapped observations.

spect to phase variations (or frequency uncertainty) and power efficiency, by controlling the observation length of the detector. The existence of decoding algorithms, whose complexity depends only slightly on the observation length (which will be described in the next chapter) makes this tradeoff efficient. NCM may be attractive even in cases where phase synchronization is not a major problem, since its degradation relative to coherent demodulation can be smaller than the degradation caused by imperfect phase estimation.

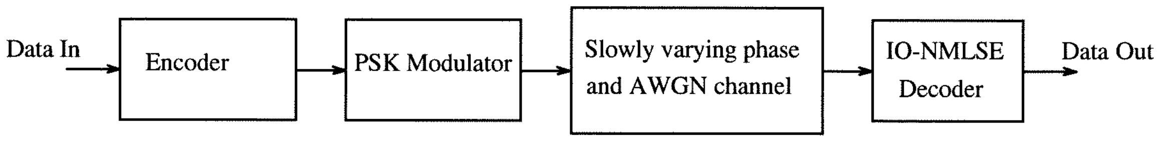


Figure 6.2: The noncoherent coded modulation system.

The noncoherent codes include the linear convolutional codes used with BPSK modulation. The existing coded systems with coherent detection (for example the codes used in the NASA Deep Space Network) can be modified to be used with the suggested detection with negligible degradation. The only limitation to this is that the code should not be a noncoherently catastrophic one.

In coherent coded modulations, in cases where enough bandwidth is available, better performance can be achieved with lower code rates without considerably increasing the complexity. In the same way, it is possible to trade bandwidth efficiency with power efficiency in NCM. Note that this trade is not efficient when using conventional detection of coded DPSK or MFSK.

This chapter is organized as follows: In Section 6.2, the general problem of detection is discussed, and the Independent Overlapped observations Noncoherent Maximum-Likelihood Sequence Estimator (IO-NMLSE) is derived. In Section 6.3, we briefly describe the codes which will be used. Section 6.4 is devoted to the error probability analysis for a coded system employing the IO-NMLSE decoder. The search for good NCM codes is described in Section 6.5. Finally, in Section 6.6, the results for several good codes, and for DPSK, are presented. In Section 6.7, some simulation results for the case where phase noise is included are presented.

In Appendix A, the proof is given for the claim that the IO-NMLSE performance approaches the coherent MLSE Maximum Likelihood Sequence Estimator performance as the observation length grows. In Appendix B, the derivation of several series expansions for the distribution of the general non-central indefinite Hermitian quadratic form in normal random variables is given. In Appendix C, a numerical method for evaluating this distribution is described.

6.2 The Noncoherent Sequence Estimation

The Optimal Noncoherent Maximum Likelihood Sequence Estimator (NMLSE) depends on the statistics of the time varying carrier phase. When such statistics are unavailable, the derivation of the optimal NMLSE must start from some broad assumptions. The commonly used assumption is that the carrier phase is constant (but completely unknown) during some observation interval $(t, t+T)$ for any t. When trying to use this assumption for the derivation of the NMLSE, we notice that the problem is ill posed. If the phase is constant over any interval of a certain length, then it has to be constant everywhere. It is clear that some phase variation should be allowed over the observation time. To make the optimal NMLSE a well defined problem, the allowed phase variation over the observation interval, and the way to measure it, should be defined. Then, the worst-case phase random process among those which fit into the constraint can be found and used for deriving the maximum of the likelihood function. This approach seems too complex and probably will not lead to any simple implementable solution.

In previous approaches, the observations were either independent or overlapping in one symbol. In our approach, maximally overlapped observations, which make use of the fact that the carrier phase can be assumed to be constant for *any* observation of length T, are used. Thus, the channel memory is utilized in a more efficient manner. In fact, it will be shown in the results that the decoding performance can be improved by increasing the overlapping ratio κ (which will be defined shortly). It is further assumed that the observations, even when they overlap in time, are independent, and have independent phases. We call the resulting estimator Independent Overlapped observations NMLSE (IO-NMLSE). Note that the observations are not made independent by any means. They are only treated as such for the derivation of the estimator, thus leading to a sub-optimal solution for the NMLSE.

In the case of coded DPSK, which uses two-symbols observations, the overlapping is inherent. However, the observations are commonly made independent by interleaving. Recently [12] showed that the non-interleaved DPSK can achieve higher performance in terms of cutoff rate and capacity, thus supporting our approach. The importance of using overlapped observations is also demonstrated in Example 6.2.1.

The IO-NMLSE discriminates between a set of possible transmitted waveforms $\{x_i(t)\}$ by choosing the signal m which maximizes the following metric

$$\eta(x_m(t), \tau) = \sum_{k=-\infty}^{\infty} \log Q_m(k\tau, k\tau + T), \tag{6.1}$$

where $Q_m(T_a, T_b)$ is the ML metric for one observation interval, $T_a \leq t \leq T_b$, and is found in [13, equation 4c.12], k is the observation number, τ is the observations spacing and T is the observation length. The choice of τ is a tradeoff between maximizing BER performance and minimizing system complexity. The complexity is approximately proportional to $\frac{1}{2}$.

$$Q_m(T_a, T_b) = \exp\left[-\frac{\alpha^2}{2}\int_{T_a}^{T_b} x_m(t)q_m^*(t)dt\right] I_0\left[\alpha\left|\int_{T_a}^{T_b} r(t)q_m^*(t)dt\right|\right], \tag{6.2}$$

where $r(t)$ is the received waveform (both $x(t)$ and $r(t)$ appear in the baseband representation), α is the channel attenuation and $q_m(t)$ is defined in [13].

The overlapping ratio is $\kappa = (T - \tau)/T$ and has values between 0 to 1. In the case of $\kappa = 0$, we get non-overlapped observations.

In the case of additive white Gaussian noise with one sided spectral density N_0, $q_m(t) = x_m(t)/N_0$ [13, pp. 344]. In the case of equal energy signals, the estimator can as well maximize

$$\eta(x_m(t), \tau) = \sum_{k=-\infty}^{\infty} \log I_o\left[\frac{\alpha}{N_0}\left|\int_{k\tau}^{k\tau+T} r(t)x_m^*(t)dt\right|\right]. \tag{6.3}$$

For low SNR (small argument) the $\log I_0(x)$ function is approximated by $x^2/4$, leading to an estimator which maximizes the metric

$$\eta(x_m(t), \tau) = \sum_{k=-\infty}^{\infty}\left|\int_{k\tau}^{k\tau+T} r(t)x_m^*(t)dt\right|^2. \tag{6.4}$$

We have confirmed by simulations that the use of this approximation does not lead to any noticeable performance degradation. In a digital implementation, where $x_m(t)$ is a sequence of symbols of duration T_s and each symbol is constructed using a $2 \times D$ dimensional signal space, the metric can be written as

$$\eta(\bar{\mathbf{x}}^{(m)}, l) = \sum_{k=-\infty}^{\infty} \eta_k = \sum_{k=-\infty}^{\infty}\left|\sum_{i=0}^{S-1} \bar{\mathbf{r}}_{lk-i}^{\dagger}\bar{\mathbf{x}}_{lk-i}^{(m)}\right|^2, \tag{6.5}$$

where S is the observation length in symbols, l (an integer) is the observations spacing in symbols, and for every symbol i, $\bar{\mathbf{r}}_i$ is a complex vector which assumes the output of D complex matched filters, each for one complex dimension of modulation. The sequence of vectors of dimension D, $\bar{\mathbf{x}}^{(m)}$, is the signal space representation of $x_m(t)$.

Let us define L as the number of trellis branches which are covered by one observation, i.e., $\lceil L = S/n \rceil$ (assuming l is a multiple of n; or else, in some cases it is necessary to add 1), where n is the number of symbols in a trellis branch. L is more important than S since it determines the complexity of the decoder and also relates more closely to the actual observation time T. Unless stated otherwise, we will use $l = n$ for maximal overlapping and convenient decoder implementation.

If the code is not noncoherently catastrophic (see next section), then as S increases (and the allowed phase variations are reduced appropriately), the performance of the IO-NMLSE approaches that of the MLSE with a completely known phase. This provides a tradeoff between robustness to phase variations and power efficiency. The proof is given in Appendix A.

Example 6.2.1 *Suppose that we want to discriminate between two possible received signals* $\{\mathbf{r}_1, \cdots, \mathbf{r}_8\}$*; one is* $\{1,1,1,1,1,1,1,1\}$ *and the other is* $\{1,1,1,1,-1,-1,-1,-1\}$. *Assume that our observation ignores the absolute phase. By taking the non-overlapping observation* $\{\mathbf{r}_1, \cdots, \mathbf{r}_4\}$ *and* $\{\mathbf{r}_5, \cdots, \mathbf{r}_8\}$*, it is clear that the two possible received signals are indistinguishable. By adding the observation* $\{\mathbf{r}_2, \cdots, \mathbf{r}_5\}$*, we are able to distinguish between the two hypotheses; for the first case we get* $\{1,1,1,1\}$*, and for the second case we get* $\{1,1,-1,-1\}$*. This shows the importance of the overlapped observations.*

We would like to state some insights and performance predictions supported by comparisons of actual schemes. Suppose that a constant phase is transmitted, $x_0(t)$, usually corresponding to the all-0 input to a code. Let us have another candidate sequence, $x_1(t)$, which metric compete with the metric of the constant phase sequence. The operation

$$\left| \int_{k\tau}^{k\tau+T} r(t) x_m^*(t) dt \right|^2$$

is equivalent to the signal passed through a filter with an impulse response $h(t) = x_m^*(T-t)$ and taking the squared magnitude of the result, sampled at time $t = T$. The operation of equation (6.5) is to consider all possible sampling points, with some finite resolution τ. The impulse response in the case of the candidate x_0 is

$$h(t) = 1, \quad 0 \le t \le T.$$

This corresponds to a low-pass filter. In order to have the largest distance between x_0 and x_1 in one observation (here distance serves as a qualitative term only), x_1 has to contain most of its energy in high frequency. Then for this observation the calculation of the metric of x_1 will be equivalent to the energy output of a high pass filter. Passing x_0, a low pass signal through this filter will result in a low output. Hence to get low error probability performance using the shortest observations, the energy of x_1 should be concentrated at as high energy as possible. See Example 6.2.2. Note that as the observation length T decreases, the equivalent low pass filter of the observations of x_0 broadens and the same happens to the high pass properties of x_1. Instead of using the concept of filtering, we can consider the correlation between the signals over an observation. For a one observation noncoherent decision, the correlation determines the error probability. Both views are mathematically equivalent.

For convolutionally coded PSK we expect to get good results for a typical good code (for coherent detection). Observations shorter than an error event length, which are located inside the event, have a random-like distribution of values. This gives high pass properties. Some specific codes may exist so that their most likely error events do not have a random-like distribution, and these codes will require larger observations to obtain good performance. Examining the results for coded PSK at the end of this chapter, we see that observation length of $S = 6$ to $S = 8$ is always needed to achieve close to coherent performance. We can think of it as the number of symbols that is needed to randomize the values in the observation enough so that they will become high pass (we can assume, that for our good codes, the error event symbols for all-0 input transmitted are distributed in a close to random manner). The high pass property of the events is also indirectly related to the bandwidth occupancy and the spectral properties of the scheme as whole. As the scheme becomes more bandwidth efficient, larger observations (measured in input bits) are required to achieve good error performance. This is true for the coded PSK schemes and also for the CPM (Continuous Phase Modulation) schemes analyzed in Chapter 8. For example, GMSK (Gaussian Minimum Shift Keying) [14] with parameter $BT = 0.25$ requires $L = 10$ to be 0.5 dB close to coherent, whereas in MSK (Minimum shift keying) only $L = 3$ is required. GMSK has much better spectral properties than MSK.

There are other approaches for explaining why bandwidth efficiency implies longer observations. We can view the noncoherent decoder as one that estimates the phase and then uses its phase estimate to coherently detect the same symbols. The quality of the phase estimation is dependent on the SNR per observation, and the last is a function of the length of the observation. As the scheme becomes more bandwidth efficient, it is typically more sensitive to phase reference errors. Thus, the observation length should be increased.

The principle underlying the operation of any noncoherent decoder is that the phase noise spectrum is narrower than the modulation spectrum. We will call this spectrum separation. The phase noise bandwidth is inversely related to the maximum time constant of the detector allowed for low degradation. As the modulation spectrum become narrower, the phase noise spectrum should become narrower as well to insure the spectral separation property. As a result, the allowed maximum time constant of the detector is increased. If the detector time constant is not increased correspondingly, the degradation cannot stay low. If this is not the case, it leads to a contradiction, since the phase noise spectrum is then allowed to be wider, disobeying the spectrum separation property, hence impossible. The same argument applies to any receiver, including coherent ones (when the phase tracking circuit is included, the coherent is essentially equivalent to a noncoherent one). Noncoherent or coherent schemes in which the time constant is variable (for the coherent this will be the PLL time constant) can be distinguished by the different spectral separation required for a given loss.

Example 6.2.2 *Let us have a code with the following three codewords (in one complex dimension), transmitted over the AWGN:*

$$\mathbf{x}^{(0)} = \{\ldots, 1, 1, 1, 1, \ldots\},$$

$$\mathbf{x}^{(1)} = \{\ldots, 1, 1, j, j, 1, 1, \ldots\},$$

and

$$\mathbf{x}^{(2)} = \{\ldots, 1, 1, j, -j, 1, 1, \ldots\}.$$

Let $\mathbf{x}^{(0)}$ *be the transmitted codeword.* $\mathbf{x}^{(2)}$ *has more high energy content than* $\mathbf{x}^{(1)}$*. Both* $\mathbf{x}^{(1)}$ *and* $\mathbf{x}^{(2)}$ *have the same probability to be detected instead of the true message when* *coherent decoding is used. However, when noncoherent decoding is used, we have found that for* $\mathbf{x}^{(1)}$ *we need* $L = 7$ *for getting close (within* 0.5 *dB) to coherent error probability but for* $\mathbf{x}^{(2)}$ *we need only* $L = 2$*!*

6.3 Linear Noncoherent Coded Modulation

We suggest the use of a specific type of trellis coded modulation called Linear Noncoherent Trellis Coded Modulation (LNTCM) defined in Chapter 3, in conjunction with the above described sequence estimator. This code exhibits the UEP (Uniform Error Property) for noncoherent detection. This property simplifies the analysis and helps us in the search for good codes. The UEP means that the error probability is not dependent on the specific codeword that was sent. By using this property, we do not have to check all possible pairs of codewords in order to find the decoder error probability when using the union bound.

When using coded modulation and a noncoherent metric, the linearity of the code does not imply that the code exhibits the UEP. The LNTCM are codes who indeed satisfy the UEP under noncoherent detection. In this paper we limit ourselves to convolutionally coded PSK modulation. For these codes the LNTCM reduces to the following codes.

Let K be the constraint length of the code, B be the number of input bits per trellis branch and $R = 2^B$ be the number of branches reaching a node, and also the number of phases in the MPSK modulation. R^{K-1} is the number of states and n is the number of PSK symbols in one branch. Let the input group and the output group be the modulo R set of integers Z_R, i.e., the input field is not GF(2) for $R > 2$. The encoder consists of a shift register of K stages; each stage contains an integer between 0 to $R - 1$. Each output i of the encoder is generated by a modulo R weighted sum of the shift register contents and the generator $\mathbf{G}_i = (\alpha_{i,1}, \alpha_{i,2}, \cdots, \alpha_{i,K})$, $0 \le \alpha_{i,j} < R$.

The output is mapped to the phase of the PSK signals linearly (not by set partitioning). The rate of the code is B/nB, which means that for every trellis branch, B bits are entering the encoder and nB bits are getting out, followed by a mapping to n PSK symbols. Other rates are also possible by having multiple shift registers or using punctured codes.

6.4 Error Probability Analysis

In this section we evaluate the IO-NMLSE error probability for a coded modulation transmitted over a channel with AWGN and slowly varying carrier phase. We will treat the case of UEP, while the result can be extended to non-UEP systems, but will require much more computation time.

Since for any slow varying phase process, in which the assumption of constant phase over S symbols approximately holds, the metric (6.5) will give the same result, we can assume without loss of generality that the phase is constant everywhere. This analysis holds for any arbitrary slowly varying phase process.

The decoder compares the metrics of all the possible paths and chooses the path with the maximal metric. For error probability computation, we assume that a specific codeword is transmitted, namely the all-$\mathcal{Z}$ sequence, $\bar{\phi}$, and find the probability that another path on the trellis will get a larger metric. The symbol $\mathcal{Z}$ is the output symbol which corresponds to the all-0 input to the encoder. When LNTCM is used, $\mathcal{Z}$ is the zero element of the output group. The derivation can be easily modified to any other transmitted sequence.

We call the codewords that happen to be a phase shift of the all-$\mathcal{Z}$ sequence "constant sequences." If constant sequences exist, and the code is not noncoherently catastrophic, i.e., the input assignments for all the constant sequences are the same, paths branching from $\bar{\phi}$ and reaching a constant sequence are also competitors.

We will use two bounds for estimating the error event probability. The first is the union bound which is the sum of all the pairwise error probabilities of the possible error events. The union bound is an upper bound. The second is the maximum among all the pairwise error probabilities which is a lower bound for the error event probability.

We will assume that S is an integer multiple of l and that $l = n$. For the case of $l = np$, where p is an arbitrary integer, the error events have p different positions relative to the beginning of the observations. In this case, one should average over all the possible positions while computing the union bound.

Unfortunately, the pairwise error probability cannot be expressed in a form suitable for the transfer function method [15]. Instead, we numerically approximate the error event probability as

$$P_{seq} \leq \sum_{m} P_e\{\bar{\mathbf{x}}^{(m)}\} \cong \sum_{|\bar{\mathbf{x}}^{(m)}|<\mathcal{M}} P_e\{\bar{\mathbf{x}}^{(m)}\}, \tag{6.6}$$

and approximate the bit error probability as

$$P_b \cong \frac{1}{B} \sum_{|\bar{\mathbf{x}}^{(m)}|<\mathcal{M}} b(\bar{\mathbf{x}}^{(m)}) P_e\{\bar{\mathbf{x}}^{(m)}\}, \tag{6.7}$$

where $\bar{\mathbf{x}}^{(m)}$ are all possible competing sequences, $|\bar{\mathbf{x}}^{(m)}|$ is the length of the error event in $\bar{\mathbf{x}}^{(m)}$ ($\bar{\mathbf{x}}^{(m)}$ itself has infinite length), and $b(\bar{\mathbf{x}}^{(m)})$ is the number of non-zero bits in the decoding of $\bar{\mathbf{x}}^{(m)}$. $P_e\{\bar{\mathbf{x}}^{(m)}\}$ is the pairwise error probability between $\bar{\phi}$ and $\bar{\mathbf{x}}$. $\mathcal{M}$ is a sufficiently large number such that the residual contribution of the larger error events can be neglected. Usually, an error event is defined as one starting from state 0 and returning back to state 0 without "visiting" (passing through) state 0. Here, we allow the error event to visit state 0 for a maximum of $L-1$ consecutive places. The shortest among the error events that visit state 0 has length of twice the shortest regular error event. Commonly, the contribution of such error events to the error probability is small and can be neglected. It is also possible to show that regular error events, i.e., those which do not visit the zero state, can be used when the trellis diagram is the augmented trellis defined in Section 7.2.

Let us derive $P_e\{\bar{\mathbf{x}}\}$. The derivation can be easily modified to any 2 sequences. Let $\bar{\mathbf{r}}_i$ be the received sequence,

$$Y_2 = \sum_{k=0}^{N-S} \left| \sum_{i=0}^{S-1} \bar{\mathbf{x}}_{i+kl}^{\dagger} \bar{\mathbf{r}}_{i+kl} \right|^2, \tag{6.8}$$

and

$$Y_1 = \sum_{k=0}^{N-S} \left| \sum_{i=0}^{S-1} \bar{\phi}^{\dagger} \bar{\mathbf{r}}_{i+kl} \right|^2. \tag{6.9}$$

$\bar{\mathbf{x}}$ is an infinite encoder output sequence containing an error event of length $N-2(S-1)$. $\bar{\mathbf{x}}$ is formed such that the $S-1$ symbols $\{\bar{\mathbf{x}}_0, \cdots, \bar{\mathbf{x}}_{S-2}\}$ are $\mathcal{Z}$ and the symbols $\{\bar{\mathbf{x}}_{N-S+1}, \cdots, \bar{\mathbf{x}}_{N-1}, \bar{\mathbf{x}}_N, \cdots\}$ are equal to either $\mathcal{Z}$ or a phase shift of $\mathcal{Z}$. The latter is appropriate for sequences which diverge to a constant sequence. The pairwise error probability is $P_e(\bar{\mathbf{x}}) = \Pr\{Y_2 > Y_1\}$. Each element, $\bar{\mathbf{r}}_i$ or $\bar{\mathbf{x}}_i$, is a vector of dimension D.

If D is large or infinite, the following derivation might not be useful since it will require computation with very large matrices. For the case of large D, one should refer to the alternative derivation presented in Section 6.4.3. Let us map $\bar{\mathbf{r}}$, $\bar{\mathbf{x}}$ and $\bar{\phi}$ each to a vector, $\mathbf{r}$, $\mathbf{x}$ and ϕ, of length DN by

$$\mathbf{r}_{Di+j} = \bar{\mathbf{r}}_{i,j}, \quad \mathbf{x}_{Di+j} = \bar{\mathbf{x}}_{i,j} \quad \text{and} \quad \phi_{Di+j} = \bar{\phi}_{i,j} = \mathcal{Z}_j$$

$$0 \le i \le N-1, \quad 0 \le j \le D-1. \tag{6.10}$$

$$Y_2 = \sum_{k=0}^{N-S} \left| \sum_{i=0}^{S-1} \sum_{j=0}^{D-1} \mathbf{r}_{Dkl+Di+j} \mathbf{x}^*_{Dkl+Di+j} \right|^2 =$$

$$= \sum_{k=0}^{N-S} \left| \sum_{i=0}^{DS-1} \mathbf{r}_{i+Dkl} \mathbf{x}^*_{i+Dkl} \right|^2 =$$

$$= \sum_{k=0}^{N-S} \left(\sum_{i=0}^{DS-1} \mathbf{r}_{i+Dkl} \mathbf{x}^*_{i+Dkl} \right)^* \left(\sum_{j=0}^{DS-1} \mathbf{r}_{j+Dkl} \mathbf{x}^*_{j+Dkl} \right) =$$

$$= \sum_{k=0}^{N-S} \sum_{i=0}^{DS-1} \sum_{j=0}^{DS-1} \mathbf{r}^*_{i+Dkl} \mathbf{r}_{j+Dkl} \mathbf{x}_{i+Dkl} \mathbf{x}^*_{j+Dkl} =$$

$$= \sum_{k=0}^{N-S} \sum_{p=Dkl}^{Dkl+DS-1} \sum_{q=Dkl}^{Dkl+DS-1} \mathbf{x}_p \mathbf{x}^*_q \mathbf{r}^*_p \mathbf{r}_q. \tag{6.11}$$

Define

$$w(i,j,k) = \begin{cases} 1, & \text{if } Dkl \le i,j \le Dkl + DS - 1; \\ 0, & \text{otherwise.} \end{cases} \tag{6.12}$$

Then

$$Y_2 = \sum_{k=0}^{N-S} \sum_{p=0}^{DN-1} \sum_{q=0}^{DN-1} \mathbf{x}_p \mathbf{x}^*_q \mathbf{r}^*_p \mathbf{r}_q w(p,q,k) =$$

$$= \sum_{p=0}^{DN-1} \sum_{q=0}^{DN-1} \mathbf{r}^*_p \mathbf{r}_q \mathbf{x}_p \mathbf{x}^*_q \sum_{k=0}^{N-S} w(p,q,k) = \sum_{p=0}^{DN-1} \sum_{q=0}^{DN-1} a_{2,p,q} \mathbf{r}_q \mathbf{r}^*_p. \tag{6.13}$$

In the same way we can express Y_1 as

$$Y_1 = \sum_{p=0}^{DN-1} \sum_{q=0}^{DN-1} a_{1,p,q} \mathbf{r}_q \mathbf{r}^*_p, \tag{6.14}$$

where

$$a_{1,p,q} = \mathcal{Z}_{p \bmod D} \mathcal{Z}^*_{q \bmod D} \sum_{k=0}^{N-S} w(p,q,k). \tag{6.15}$$

Let

$$a_{p,q} = a_{2,p,q} - a_{1,p,q} = \left(\mathbf{x}_p\mathbf{x}_q^* - \mathcal{Z}_{p \bmod D}\mathcal{Z}_{q \bmod D}^*\right) \sum_{k=0}^{N-S} w(p,q,k), \tag{6.16}$$

and $\mathbf{A} = \{a_{p,q}\}$, a $DN \times DN$ matrix, then

$$Y = Y_2 - Y_1 = \mathbf{r}^\dagger \mathbf{A} \mathbf{r}. \tag{6.17}$$

We want to find the probability of $Y > 0$. The vector $\mathbf{r}$ is a complex Gaussian random vector of length DN, with $E[\mathbf{r}] = \frac{1}{\sigma}\phi$ and a covariance matrix $\mathbf{C} = 2\mathbf{I}$ (unit variance in the real and the imaginary parts of each of the vector components), where

$$\sigma^2 = \frac{DN_0 n}{2BE_b}, \tag{6.18}$$

and $\mathbf{I}$ is the identity matrix. Let us express $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-1}, \tag{6.19}$$

where $\mathbf{\Lambda}$ is the diagonal eigenvalue matrix, and define $\mathbf{Z} = \mathbf{Q}^{-1}\mathbf{r}$, then

$$Y = \mathbf{r}^\dagger \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-1}\mathbf{r} = \mathbf{Z}^\dagger \mathbf{\Lambda} \mathbf{Z} = \sum_{i=0}^{DN-1} \lambda_i |\mathbf{Z}_i|^2. \tag{6.20}$$

It is easy to show that $E[\mathbf{Z}] = \mathbf{Q}^{-1}E[\mathbf{r}]$ and that the covariance matrix of $\mathbf{Z}$ is $2\mathbf{I}$.

The form (6.20) resembles the χ^2 distribution. It would have been exactly χ^2 if all the eigenvalues of $\mathbf{A}$ were equal, which is generally not the case. The distribution of Y is that of a *non-central general quadratic form in Gaussian variables* [16]. The general case, which we have, is the most difficult case to treat. Some special cases are easier; among them are the case where $\mathbf{A}$ is positive definite and the case where we have a central general quadratic form, defined as $E[\mathbf{Z}] = 0$. $\{a_{1,p,q}\}$ and $\{a_{2,p,q}\}$ are each a positive definite matrix, but their difference is not.

For the general quadratic form expressed as

$$Y = \sum_{j=0}^{n-1} \lambda_j (W_j - \beta_j)^2, \tag{6.21}$$

where W_i are unit real normal random variables and β_i are real constants, the characteristic function is given by [16] and is:

$$\Phi(i\omega) = E[e^{i\omega y}] = e^{-\frac{1}{2}\sum_{j=0}^{n-1}\beta_j^2}\, e^{\frac{1}{2}\sum_{j=o}^{n-1}\frac{\beta_j^2}{1-2i\omega\lambda_j}} \prod_{j=0}^{n-1}(1-2i\omega\lambda_j)^{-\frac{1}{2}}. \tag{6.22}$$

In our case $\mathbf{Z}_i$ is complex, so we can write

$$|\mathbf{Z}_i|^2 = (W_{2i} - \beta_{2i})^2 + (W_{2i+1} - \beta_{2i+1})^2, \quad \lambda_{2i} = \lambda_{2i+1}, \tag{6.23}$$

$$\beta_{2i} = \mathrm{Re}\{\mu_i\}, \quad \beta_{2i+1} = Im\{\mu_i\}, \quad \mu_i = E[Z_i].$$

From equation (6.22) and (6.23), we get

$$\Phi(j\omega) = e^{-\frac{1}{2}\sum_{i=0}^{DN-1}|\mu_i|^2} e^{\frac{1}{2}\sum_{i=0}^{DN-1}\frac{|\mu_i|^2}{1-2\lambda_i j\omega}} \prod_{i=0}^{DN-1} \frac{1}{1-2\lambda_i j\omega}. \tag{6.24}$$

This characteristic function should be inverted in order to find the distribution of Y.

There is no known closed form solution for the distribution. There is not even a good series expansion for the general case. We have developed a series expansion which, for fast convergence, requires that the eigenvalues be distinct (see Appendix B). Since we cannot guarantee this property, we have developed another numerical method (see Appendix C). This second numerical method works well for all ranges of eigenvalues.

6.4.1 High SNR Approximation

Let us look at the expression for Y.

$$Y = \mathbf{r}^\dagger \mathbf{A}\mathbf{r} = (\phi + \mathbf{n})^\dagger \mathbf{A}(\phi + \mathbf{n}) = \phi^\dagger \mathbf{A}\phi + 2Re\{\mathbf{n}^\dagger \mathbf{A}\phi\} + \mathbf{n}^\dagger \mathbf{A}\mathbf{n}, \tag{6.25}$$

where $\mathbf{n}$ is a complex Gaussian vector with zero mean and covariance matrix $\mathbf{C} = 2\sigma^2\mathbf{I}$. One possible approximation is to ignore the $\mathbf{n} \times \mathbf{n}$ term which is the third term. In high SNR this term will be smaller than the other noise term, the second term, since the noise is relatively small and it appears in second order. The second term is a Gaussian random variable (a scalar) with zero mean, and variance

$$\sigma^2 = 4|\mathbf{A}\phi|^2\sigma_n^2 = |\mathbf{A}\phi|^2 \frac{2Dn}{B}\frac{N_0}{E_b}. \tag{6.26}$$

We can see that the third term has also zero mean:

$$E[\mathbf{n}^\dagger \mathbf{A}\mathbf{n}] = E[\mathbf{n}^\dagger \mathbf{Q}\Lambda\mathbf{Q}^\dagger \mathbf{n}] = E[\mathbf{Z}^\dagger \Lambda \mathbf{Z}] = \sum_i \lambda_i E[|\mathbf{Z}_i|^2] = 2\sigma_n^2 \sum_i \lambda_i, \tag{6.27}$$

where $\mathbf{Z}$ is another complex Gaussian vector with zero mean and same covariance, which relates to $\mathbf{n}$ by $\mathbf{Z} = \mathbf{Q}^\dagger \mathbf{n}$. The sum over the eigenvalues of $\mathbf{A}$ is equal to the trace of $\mathbf{A}$. It can be shown that the trace of $\mathbf{A}$ is zero by performing the sum $\sum_{p=0}^{DN-1} a_{p,p}$ and using $\bar{\mathbf{x}}_i^\dagger \bar{\mathbf{x}}_i = \mathcal{Z}^\dagger \mathcal{Z} \quad \forall i$. Moreover, it can be shown that the eigenvalues of $\mathbf{A}$ satisfy the property that if λ is an eigenvalue, then $-\lambda$ is also an eigenvalue.

The error probability after neglecting the $\mathbf{n} \times \mathbf{n}$ term is

$$Pr\{Y > 0\} \approx Q\Big(\frac{\phi^\dagger \mathbf{A} \phi}{|\mathbf{A}\phi|}\sqrt{\frac{BE_b}{2DnN_0}}\Big). \tag{6.28}$$

The computation of the above expression is much easier compared to the exact computation of the error probability which involves a calculation of the eigenvalues of a matrix. Unfortunately, the result is not very accurate when working in moderate SNR, but it is good for the process of the preselection of codes before testing them more rigorously.

Surprisingly, the result of the exact calculation of the probability was always found to be lower than this estimation. It was reasonable to assume that the extra noise term, the third term, will be responsible for increasing the error probability. It can be shown that the correlation between the second and third term is zero. However, there is a negative correlation between the absolute value of the second term and the value of the third term. When the value of the second term is positive high, the third term tends to be negative and thus reduces the total of the first, second and third terms. The decrease of the error probability is thus understood. The conclusion is that the equation (6.28) can be used as an upper bound on the error probability.

6.4.2 Approximation by Using the Largest Eigenvalues

A good approximation to the error probability is based on equation (6.20). In the summation we can keep only the terms of λ_{max} and $-\lambda_{max}$ (as it was mentioned before the eigenvalues always come in pairs of positive and negative).

$$Pr\{Y > 0\} \cong Pr\{\lambda_{max}|Z_1|^2 - \lambda_{max}|Z_2|^2 > 0\} = Pr\{|Z_1|^2 > |Z_2|^2\}, \tag{6.29}$$

where the indices 1 and 2 correspond to $+\lambda_{max}$ and $-\lambda_{max}$ respectively. Equation (6.29) is of the same form as the error probability of FSK when the waveforms are not orthogonal. It is the probability of having one Rician r.v. greater than another Rician r.v.; see for example [13]. It is evaluated by the Markov Q function. The approximation is good only for cases where λ_{max} is much larger than the other eigenvalues; examining numerous error events showed that this is most likely the case. The main difficulty is to find μ_1 and μ_2, the expected values of Z_1 and Z_2. The following procedure can be used to find them, without having to find the other eigenvalues. For large N,

$$A^N x \cong \lambda_{max}^N (q_1 a_1 + q_2 a_2 (-1)^N),$$

where a_1 and a_2 are unknown constants, x is an arbitrary vector, and q_1 and q_2 are the normalized eigenvectors corresponding to λ_{max} and $-\lambda_{max}$. Let

$$v_1 = A^{N-1}x, \quad v_2 = A^N x. \tag{6.30}$$

Since q_1 and q_2 are orthogonal,

$$|v_1| = \lambda_{max}^{N-1}(|a_1|^2 + |a_2|^2)^{0.5}, \quad |v_2| = \lambda_{max}^{N}(|a_1|^2 + |a_2|^2)^{0.5}. \tag{6.31}$$

Let

$$V_1 = \frac{v_1}{|v_1|} + \frac{v_2}{|v_2|} = \frac{2a_1 q_1}{(|a_1|^2 + |a_2|^2)^{0.5}}, \tag{6.32}$$

and

$$V_2 = \frac{v_1}{|v_1|} - \frac{v_2}{|v_2|} = \frac{2a_2 q_2}{(|a_1|^2 + |a_2|^2)^{0.5}}. \tag{6.33}$$

As a result, using $|q_1| = 1$,

$$|\mu_1| = |q_1^\dagger \phi| = \frac{|V_1^\dagger \phi|}{|V_1|}, \tag{6.34}$$

where ϕ is defined in (6.10). Similarly,

$$|\mu_2| = |q_2^\dagger \phi| = \frac{|V_2^\dagger \phi|}{|V_2|}. \tag{6.35}$$

6.4.3 Evaluating the Pairwise Error Probability for Symbols with Large or Infinite Number of Dimensions

The pairwise error probability computation in the previous sections involved $DN \times DN$ matrices. When D is large or infinite, the method becomes computationally too extensive. Here, we develop a method which uses only the values of the correlations between the symbols and thus is independent of D. In fact, the signal space presentation is not required; signal waveforms can be used for the computation of the correlations between symbols. The size of the matrixes involved will always be $2N$. We know that such a derivation is possible using the results in Theorem 3.2.1. Let $\alpha_p = \bar{\mathbf{x}}_p^\dagger \bar{\mathbf{r}}_p$ and $\beta_p = \mathcal{Z}^\dagger \bar{\mathbf{r}}_p$. Let

$$Y_1 = \sum_{k-0}^{N-L} \left| \sum_{i=0}^{S-1} \beta_{i+kl} \right|^2 \tag{6.36}$$

and

$$Y_2 = \sum_{k=0}^{N-S} \left| \sum_{i=0}^{S-1} \alpha_{i+kl} \right|^2 . \tag{6.37}$$

$$Y_2 = \sum_{k=0}^{N-S} (\sum_{i=0}^{S-1} \alpha_{i+kl})^* (\sum_{j=0}^{S-1} \alpha_{j+kl}) = \sum_{k=0}^{N-1} \sum_{i=0}^{S-1} \sum_{j=0}^{S-1} \alpha_{i+kl}^* \alpha_{j+kl} = \tag{6.38}$$

$$= \sum_{k=0}^{N-1} \sum_{p=kl}^{kl+S-1} \sum_{q=kl}^{kl+S-1} \alpha_p^* \alpha_q = \sum_{p=0}^{N-1} \sum_{q=0}^{N-1} \alpha_p^* \alpha_q \sum_{k=0}^{N-1} w(p,q,k), \tag{6.39}$$

where

$$w(p,q,k) = \begin{cases} 1, & \text{if } kl \le p,q \le kl+S-1; \\ 0, & \text{otherwize} \end{cases} . \tag{6.40}$$

Let

$$a_{p,q} = \sum_{k=0}^{N-1} w(p,q,k) \tag{6.41}$$

and let $\mathbf{A} = \{a_{p,q}\}$, a $N \times N$ matrix, then

$$Y_2 = \alpha^\dagger \mathbf{A} \alpha, \tag{6.42}$$

where α and β are vectors with components α_p and β_p, $0 \le p \le N-1$. Similarly,

$$Y_1 = \beta^\dagger \mathbf{A} \beta. \tag{6.43}$$

Thus, we get

$$Y = Y_2 - Y_1 = \alpha^\dagger \mathbf{A} \alpha - \beta^\dagger \mathbf{A} \beta. \tag{6.44}$$

Let us define the vector U of length $2N$ as

$$U = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}, \tag{6.45}$$

and the $2N \times 2N$ matrix P as

$$P = \begin{pmatrix} A & 0 \\ 0 & -A \end{pmatrix}. \tag{6.46}$$

Then,

$$Y = U^\dagger P U. \tag{6.47}$$

Let us assume that $\bar{\mathbf{r}}_i$ is normalized such that each component has unit variance, and the symbols $\bar{\mathbf{x}}_i$ have unit energy. Let V be the covariance matrix of U. Its nonzero entries are

$$\begin{gathered} V_{p,p} = E\{[\alpha_p - E(\alpha_p)][\alpha_p^* - E(\alpha_p)^*]\} = \\ = E\{\bar{\mathbf{x}}_p^\dagger[\bar{\mathbf{r}}_p - E(\bar{\mathbf{r}}_p)][\bar{\mathbf{r}}_p - E(\bar{\mathbf{r}}_p)]^\dagger \bar{\mathbf{x}}_p\} = \bar{\mathbf{x}}_p^\dagger I \bar{\mathbf{x}}_p = \bar{\mathbf{x}}_p^\dagger \bar{\mathbf{x}}_p = 1, \end{gathered} \tag{6.48}$$

$$\begin{gathered} V_{p,p+N} = E\{[\alpha_p - E(\alpha_p)][\beta_p^* - E(\beta_p)^*]\} = \\ E\{\bar{\mathbf{x}}_p^\dagger[\bar{\mathbf{r}}_p - E(\bar{\mathbf{r}}_p)][\bar{\mathbf{r}}_p - E(\bar{\mathbf{r}}_p)]^\dagger \mathcal{Z}\} = \bar{\mathbf{x}}_p^\dagger \mathcal{Z} \triangleq u_p, \end{gathered} \tag{6.49}$$

and

$$V_{p+N,p} = V_{p,p+N}^*. \tag{6.50}$$

$$V = \begin{pmatrix} 1 & & & & u_0 & & & \\ & 1 & & & & u_1 & 0 & \\ & & \ddots & & 0 & & \ddots & \\ & & & \ddots & & & & u_{N-1} \\ u_0^* & & 0 & & \ddots & & & \\ & u_1^* & & & & \ddots & & \\ & 0 & \ddots & & & & \ddots & \\ & & & u_{N-1}^* & & & & 1 \end{pmatrix}. \tag{6.51}$$

Here we have a quadratic form in random variables like in equation (6.17). However, there is a difference. Here the components of the random vector U are correlated. We proceed in a method similar to [16]. Let us find a non-singular matrix L such that $V = L^\dagger L$. Since V is Hermitian, such decomposition always exists [19] and is not unique. Let us define another random vector related to U as

$$W = L^{-1^\dagger} U. \tag{6.52}$$

The following relations are useful to note: $L^{-1^\dagger} = L^{\dagger^{-1}}$ and $(L^\dagger L)^{-1} = L^{-1} L^{-1^\dagger}$. The covariance matrix of W is

$$E\{[W - E(W)][W - E(W)]^\dagger\} = L^{-1^\dagger} E\{[U - E(U)][U - E(U)]^\dagger\} L^{-1} =$$

$$= L^{-1^\dagger} V L^{-1} = L^{-1^\dagger} L^\dagger L L^{-1} = I. \tag{6.53}$$

Hence, the components of W are uncorrelated. Let us use (6.52) in (6.47) to get

$$Y = W^\dagger L P L^\dagger W. \tag{6.54}$$

From this point we can proceed similarly to Section 6.4. Let Λ be the diagonal eigenvalues matrix of $LPL^\dagger$ and Q the unitary eigenvectors matrix. If we set $Z = Q^\dagger W$, then

$$Y = W^\dagger Q \Lambda Q^\dagger W = Z^\dagger \Lambda Z = \sum_{i=0}^{2N-1} \lambda_i |Z_i|^2. \tag{6.55}$$

Finally,

$$P_e\{\mathbf{x}\} = \Pr\{Y > 0\} = \Pr\left\{\sum_{i=0}^{2N-1} \lambda_i |Z_i|^2 > 0\right\}. \tag{6.56}$$

Now we can write the characteristic function as in equation (6.24) and use the techniques described in the appendices for finding the cumulative distribution of Y. Note that the eigenvalues of $LPL^\dagger$ are the same as those of PV. Since Y in (6.55) has to have the same distribution as in equation (6.20), λ_i should be the same (up to a scale factor) under both derivations and the additional N values should be zero. This was always the case in our tests. Another quantity that needs to be evaluated is $\bar{\mu}^T = [\mu_0, \ldots, \mu_{2N-1}]^T$.

$$\bar{\mu}^T = E[Z] = Q^\dagger L^{-1\dagger} E[U], \tag{6.57}$$

where

$$E[U] = \frac{1}{\sigma} \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}, \tag{6.58}$$

and

$$\sigma^2 = \frac{N_0 n}{2BE_b}. \tag{6.59}$$

The matrix L can be found by standard triangular factorization techniques. However, for our particular case we were able to find a close form for L (one of many possible solutions). The validity of the following expressions can be confirmed by inspection.

$$L = \begin{pmatrix} \frac{u_0^*}{\sqrt{1-|u_0|^2}} & 0 & \dots & \dots & 1 & 0 & \dots & 0 \\ \sqrt{1-|u_0|^2} & 0 & \dots & \dots & 0 & 0 & \dots & 0 \\ 0 & \frac{u_1^*}{\sqrt{1-|u_1|^2}} & \dots & \dots & 0 & 1 & \dots & 0 \\ 0 & \sqrt{1-|u_1|^2} & \dots & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \frac{u_{N-1}^*}{\sqrt{1-|u_{N-1}|^2}} & 0 & 0 & \dots & 1 \\ 0 & 0 & \dots & \sqrt{1-|u_{N-1}|^2} & 0 & 0 & \dots & 0 \end{pmatrix}, \tag{6.60}$$

and

$$L^{-1} = \begin{pmatrix} 0 & \frac{1}{\sqrt{1-|u_0|^2}} & 0 & 0 & \dots & \dots & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{1-|u_1|^2}} & \dots & \dots & 0 \\ \vdots & \vdots & \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 & \frac{1}{\sqrt{1-|u_{N-1}|^2}} \\ 1 & -\frac{u_0^*}{\sqrt{1-|u_0|^2}} & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & -\frac{u_1^*}{\sqrt{1-|u_1|^2}} & \dots & 0 & 0 \\ \vdots & \vdots & & & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 & -\frac{u_{N-1}^*}{\sqrt{1-|u_{N-1}|^2}} \end{pmatrix}. \tag{6.61}$$

6.5 Searching for Good Codes

Given a specific observation length and code parameters, the optimal code was found by a computer search. The minimum Euclidean distance serves only as a lower bound, thus it is not very useful for NCM selection.

We have used the bit error probability bound (6.7) as the optimization criterion, but we took care to minimize the maximum error event probability as much as possible. For small K, R and n, the number of possible codes is not too large and a one by one search was performed to check all possible codes in order to find the optimal one. For larger values of the parameters, this method is not practical. In this case, the codes to be tested were produced at random. Since there is a large number of good codes, the probability of finding one of these good codes is not too small.

With either method, a large number of codes had to be checked. By observing symmetry properties of the codes, we can reduce the number of codes to be checked.

For example, a code and its reverse (the generators are reversed) have exactly the same performance, since all the possible error events are reversed. Permutating the generators yields to an equivalent code, and the same thing happens if we conjugate all the generators' coefficients.

For each code, we have to compute the error probability of each of the possible error events. Since there is an infinite number of error events, we will consider only error events limited to a finite length. Error events longer than those are assumed to have only a small contribution to the error event probability.

The exact error probability computation for an error event involves the computation of the eigenvalues and eigenvectors of a matrix and the computation of the quadratic form cumulative distribution. We have used simple lower bound criteria for fast preselection of codes to speed up the search. The search was performed in 4 stages, alternating preselection rules with exact computation, and in each stage $\mathcal{M}$ is increased.

The first preselection is based on the computation of the correlation (absolute squared) of the sequence with the all-$\mathcal{Z}$ sequence,

$$\rho = \left| \sum_{i=0}^{N-1} \bar{\phi}_i^\dagger \bar{\mathbf{x}}_i \right|^2 . \tag{6.62}$$

The total correlation can be used to compute a lower bound on the error probability. Our decoder cannot be better than the optimal way of deciding between two hypotheses under random phase conditions. The latter has an error probability (for a constant symbol energy) which is a function of the correlation ρ and the sequence length N. For the fast preselection, ρ was compared to a threshold which is a pre-computed function of N. The error probability of binary noncoherent block detection is found in [13, Section 4.3.1]. This lower bound is tighter than the one based on the Euclidean distance which is the coherent error probability.

The second preselection rule was based on the high SNR approximation, equation (6.28), which upperbounds the pairwise error probability. The argument of the Q function was compared to a threshold.

The code search procedure consists of four stages. Every code must pass three stages, and then in the fourth stage the final calculation of bit error probability is being made. In the first stage, which is the fastest, all the error events of length up to $K+2$ are checked against the preselection rules described above. Having passed the first stage, the code reaches the second stage, in which an exact error probability calculation using all error events up to length $K+2$ is performed. To pass the second stage, both the maximal probability and the union bound must pass thresholds. In the third stage, the fast preselection is applied again, but now, to all error events up to a maximal length depending on the allowed computing time. In the fourth stage, an exact probability computation is done using error events of length less or equal to a certain length, again limited by computing power. The result of the final stage is the base for choosing the optimal code. The probability calculation is done for a specific E_b/N_0, and L whose values will slightly influence the optimal code selection.

The codes to be presented in the next section are some of the results of this computer search. For the BPSK and QPSK case we have used $L=4$, and for the 8PSK case we have used $L=3$. The codes that use BPSK or QPSK are also found to be maximum free distance codes. This shows their optimality to either coherent or noncoherent detection.

Table 6.1: Best noncoherent codes found (each represents a class of equivalent codes).

R	D	K	N	$\alpha_{1,j}$	$\alpha_{2,j}$	$\alpha_{3,j}$
2	2	5	16	10011	11101	
2	2	7	64	1000101	1101111	
4	2	3	16	133	231	
4	2	4	64	2123	1312	
4	3	4	64	2122	1323	3311
8	2	4	64	727	562	

6.6 Results and Discussion

We present the performance of some good NCM using BPSK, QPSK and 8PSK modulations when detected noncoherently with various observation lengths, and coherently for comparison. The bit error probability is computed by using equation (6.7). For computing the union bound we do an exact calculation of the pairwise error probability and sum over all error events up to a certain length, where the contribution of all other longer error events is negligible. All the results presented are computed for codes, given in Table 6.1, found through the search for good codes. Note that each code in this table is a representative of a class of equivalent codes.

Fortunately, except for the case $K = 7$, rate $1/2$ convolutionally coded BPSK, the optimum codes found also have maximum free distance, thus they are also optimal for coherent detection. The best convolutional codes are tabulated in [13, table 5.3]. For the $K = 7$, rate $1/2$ case the free distance of the best found NCM is 9, while the maximum free distance possible for $K = 7$, rate $1/2$ is 10. For the BPSK and QPSK cases, we get a fair comparison between noncoherent and coherent detection. By fair comparison we mean that the best found NCM is also optimal for coherent detection, thus the performance curves for this code will reflect the best possible for either case. For the case of 8PSK, it is difficult to make fair comparison since to the best of our knowledge no optimal rate 3/6 TCM using 8PSK has been published. However, the coherent results, for the 8PSK code found, are very good (compared to BPSK and QPSK). Thus, we get a good comparison between coherent and noncoherent detection also in this case.

We see that as the observation length grows, the performance of the noncoherent detection approaches that of the coherent, and we observe that the rate of convergence seems to be only slightly dependent on the SNR.

In Figure 6.3, a $K = 5$ rate $1/2$ coded BPSK (from Table 6.1) is evaluated. For comparison, when we decode the same code with non-overlapped observations, we get poor performance. The computation of the non-overlapped metric bit error rate is performed in a similar fashion to the derivation of the performance of the overlapped metric treated in Section 6.4. We repeat the derivation, assuming $l = S$ in equation 6.5.

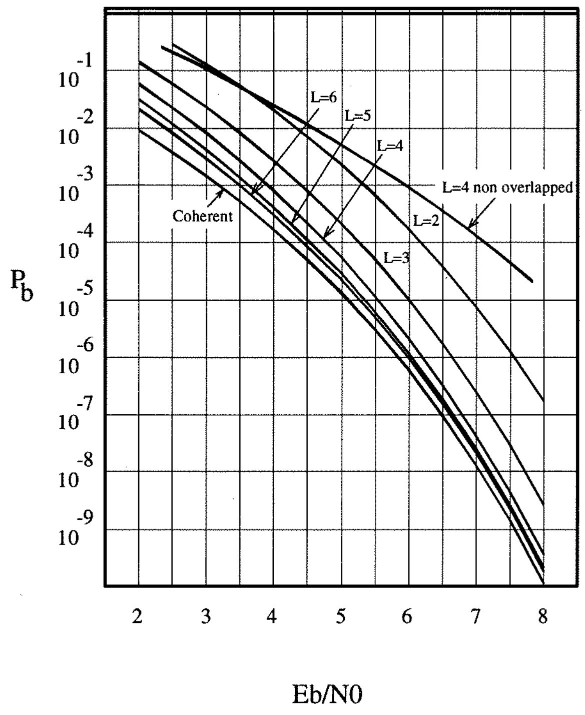


Figure 6.3: BER performance with noncoherent and coherent detection of rate 1/2 coded BPSK with $K = 5$ (16 states). This code is the best code for noncoherent detection. Moreover, because $d_{\text{free}} = 7$, it is a maximum free distance code. For comparison, the performance of non-overlapped observations ($\kappa = 0$) with $L = 4$ is included.

In Figure 6.4 and 6.5, two codes with $B = 2$ and $n = 2$ (n is the number of symbols per trellis branch), which means rate 2/4 coded QPSK (2 bit in and 4 bit out at each clock tick mapped to two 2 QPSK symbols), are evaluated for $K = 3$ (16 states) and $K = 4$ (64 states). We have simulated the code of $K = 3$ (Figure 6.4) using the VA (we have used the optimal algorithm, Section 7.2) to confirm the results of the analysis.

Figure 6.6 presents the performance of a code with $K = 5$, $B = 2$ and $n = 3$, which is rate 2/6 coded QPSK. We use this example to show the possibility of using larger bandwidth in order to improve the results. This code has a free distance of 16 which is even better than that of the tabulated best rate 1/3 convolutional code with 64 states

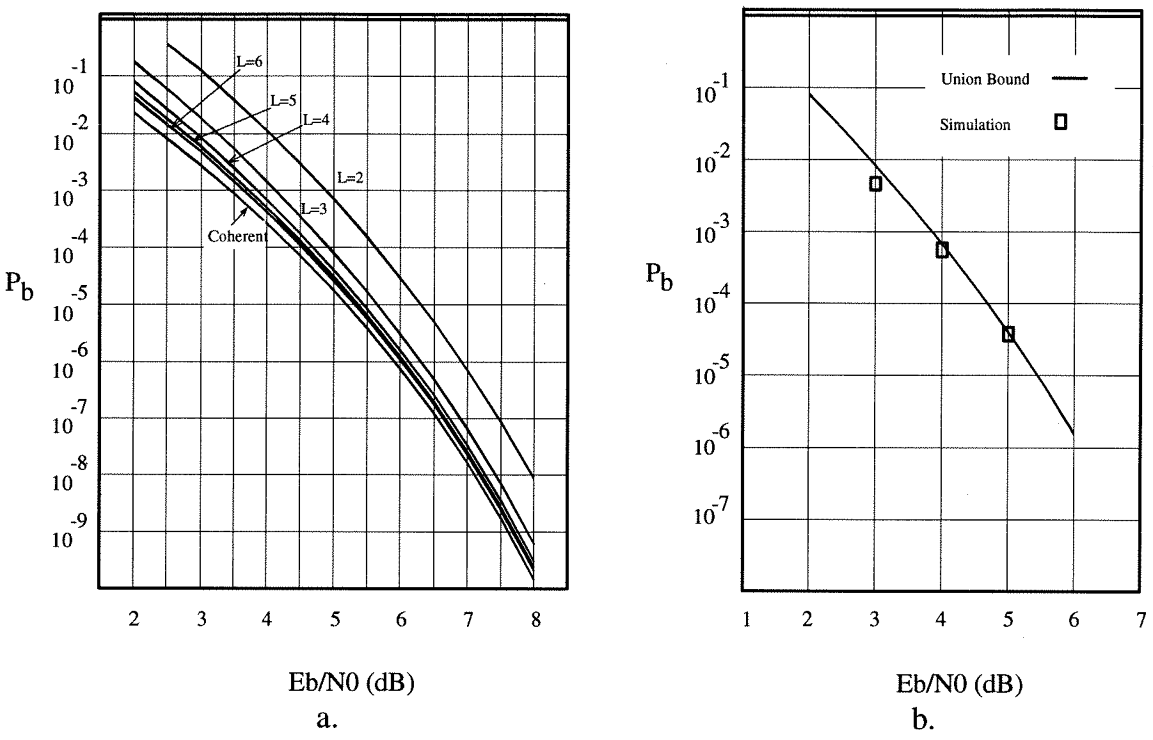


Figure 6.4: a. BER performance with noncoherent and coherent detection of 2/4 coded QPSK with $K = 3$ (16 states). This code is the best code for noncoherent detection. Moreover, because $d_{\text{free}} = 7$, it is a maximum free distance code. b. Confirmation of analysis results by simulation for $L = 4$.

which is 15. In fact, the use of 2/6 code enables us to get larger free distance than the maximum possible in 1/3 code (the bound for a rate 2/6 code is 16). An example using 8PSK is given in Figure 6.7. In this case $B = 3$, $n = 2$, $R = 8$ and $K = 3$ (64 states).

We see that the optimal code for noncoherent detection is very close to the optimal one for the coherent case. Can we reverse the argument and say that the best coherent code will be the best also if decoded noncoherently? The answer is no. First, a code which contains constant sequences (with the same input assignment for all of them such that no error occurs if we chooses a wrong one) will have more sequences as candidates for error events and so will have a higher error probability when decoded noncoherently. Observation lengths significantly larger are needed in order to make their contribution negligible. This is the case with the commonly used $K = 7$ rate 1/2 convolutional code with generators in octal 133, 171 which is a noncoherently catastrophic code. With

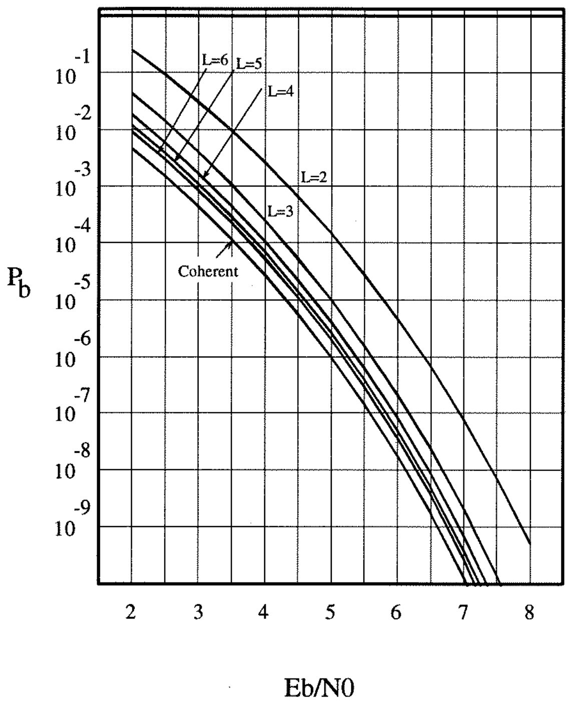


Figure 6.5: BER performance with noncoherent and coherent detection of rate 2/4 coded QPSK with $K = 4$ (64 states). This code is the best code for noncoherent detection. Moreover, because $d_{\text{free}} = 10$, it is a maximum free distance code.

correct input assignments to the trellis branches, we can make a modified (feedback) encoder which is non-catastrophic. This code has free distance of 10. The best NCM for the same parameters has free distance of 9, so its performance is slightly lower than the above code. Second, two codes with equal coherent performances might have a different noncoherent performance. For example, in Figure 6.8, two codes with $B = 2$, $D = 2$ and $R = 4$. One is $111, 312$ and the other is $133, 231$ (the generators in base 4). Both have the best free Hamming distance of 7 and almost the same coherent performance. When decoded noncoherently their performances differ considerably; $133, 231$ is much better.

Looking at the performance of the different codes, we conclude that for a degradation of less than 0.5 dB relative to the coherent case, $L = 3$ or 4 is sufficient. It is interesting to note that $L = 3$ in $B = 3$ code is equivalent in SNR per observation to $L = 9$ in $B = 1$. However, as was discussed in Section 6.2, the observation length

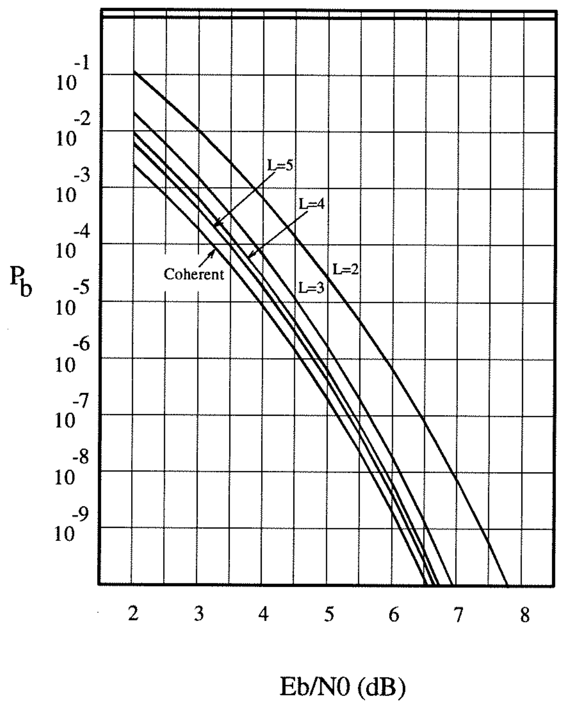


Figure 6.6: BER performance with noncoherent and coherent detection of 2/6 coded QPSK with $K = 4$ (64 states). This code is the best code for noncoherent detection. Moreover, because $d_{\text{free}} = 16$, it is a maximum free distance code.

needed to achieve the randomizing effect is measured in symbols. This is also related to the bandwidth efficiency of the scheme. As the scheme becomes more bandwidth efficient, larger observation (measured in input bits) is required.

In Figure 6.9 we compare the cases of $l = 1$ and $l = 2$. In the case of $l = 1$ and $S = 2$, the decoder is similar to the decoder for coded DPSK. Note that no differential encoding takes place. We notice the slight improvement for the case of $l = 1$ which correspond to higher overlapping ratio κ.

Differentially encoded MPSK (DMPSK) can also be decoded by the IO-NMLSE. We show the results for the case of $M = 4$ as an example. We compare our results to the optimal block Multi-Symbol differential detector which was suggested by Divsalar et al. [2]. In both cases, the performance for $S = 2$ is equal to the performance of conventional DMPSK as expected. Also in both cases as $S \to \infty$, the performance is equal to that of coherent detection. Unlike the block detector which utilizes one

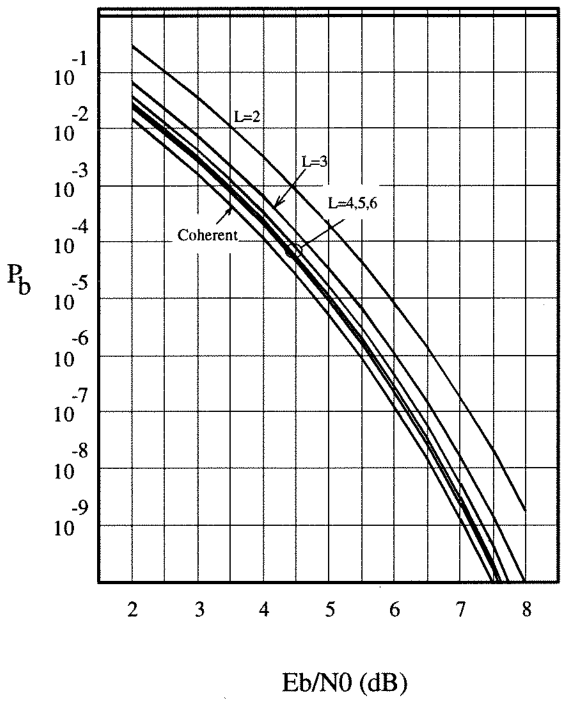


Figure 6.7: BER performance with noncoherent and coherent detection of 3/6 coded 8PSK with $K = 4$ (64 states). This code is the best for noncoherent detection.

observation at a time, the IO-NMLSE utilizes all possible observations of length S and by this achieves a superior performance for any $S > 2$, leading to a faster convergence to the coherent case as S increases. The comparison is presented in Figure 6.10.

In order to compute the bit error rate of DQPSK with IO-NMLSE, we use the truncated sequence union bound of equation (6.7) with $\mathcal{M} = 3$ since the contribution of larger error events was found to be negligible. The main contributors to the error probability for the cases $S = 2$ and $S = 3$ are the error events of the form $\{\cdots, +1, +1, +j, +j, \cdots\}$ which correspond to one bit error. The main contributor for larger S is $\{\cdots, +1, +1, +j, +1, +1, \cdots\}$, the same error event that dominates in the coherent case.

The bit error probability in the presence of phase noise is evaluated by simulation of the IO-NMLSE detector. In Figure 6.11 and 6.12, we present the performance of the optimal 16-states rate 1/2 LNTCM with QPSK modulation (the code of Figure 6.4)

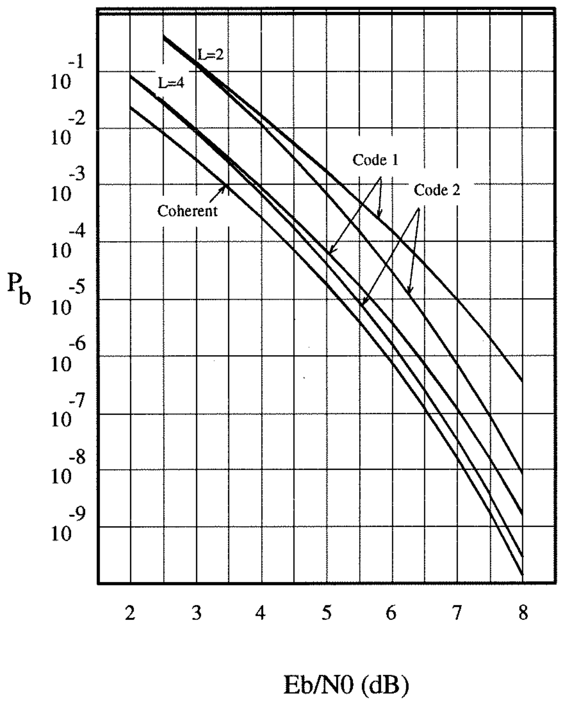


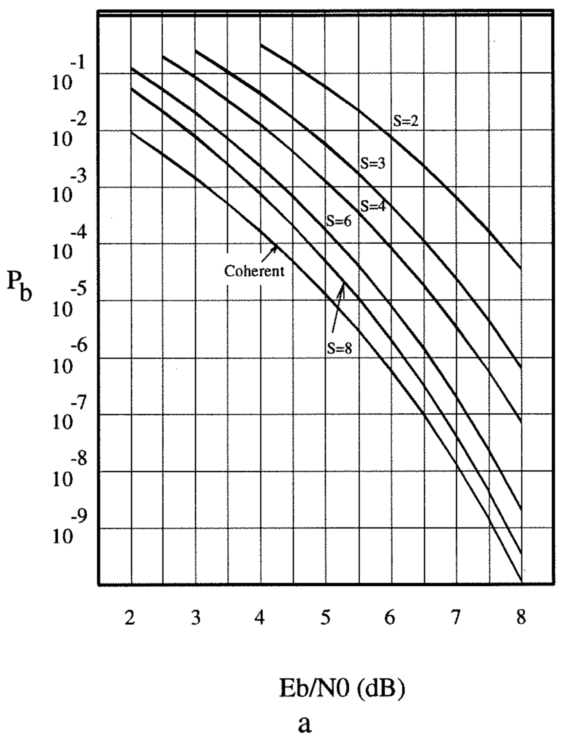
Figure 6.8: Comparison of two maximal free distance codes with rate 2/4 and QPSK. It is clear that one of them is better for noncoherent detection.

when detected noncoherently, with observation lengths $S = 6$ and $S = 8$ with various levels of phase noise. The coherent performance is also shown for comparison.

For the phase noise model we assume a very simplified model where the phase noise is a first order Markov process with Gaussian transition probability distribution. This corresponds to a frequency spectrum that behaves as $1/f^2$. In practice this means that the phase variation between successive symbols is an independent normal r.v., with mean zero and specified variance δ^2.

In Figure 6.13, we present the results of bit error probability, at $E_b/N_0 = 4$ dB, versus δ. In this graph the increased sensitivity of the larger observation cases is clearly demonstrated. However, it is somewhat surprising to note that the degradation at low values of δ seems to be independent of the observation length.

We see that very low degradation occur for $\delta < 5°/symbol$ even for a large observation as $S = 12$, where the loss from the coherent case is less than 0.3 dB. We like to compare this result to a conventional coherent technique. As checked by simulation



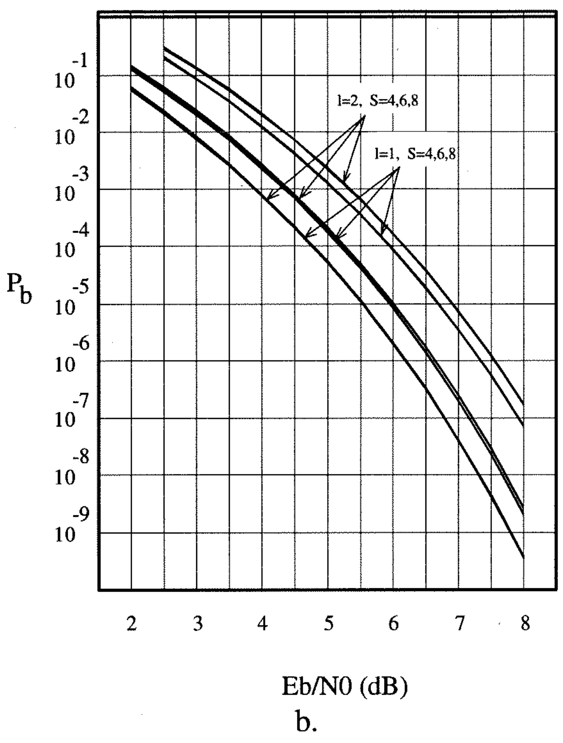


Figure 6.9: a. The code of Figure 6.3 decoded with $l = 1$. b. Comparison between the decoding with $l = 1$ and $l = 2$. Note that the $l = 2$ case was plotted in Figure 6.3 where $L = S/2$.

of a QPSK Costas loop (at $E_b/N_0 = 3$ dB, rate 1/2 coded QPSK), it is not possible to achieve stable lock at $\sigma > 0.5°/symbol$, whereas the degradation of the noncoherent scheme is very small even when $\sigma = 5°/symbol$. This means that the specifications of the allowed phase instability can be relaxed by more than 20 dB when switching from coherent to noncoherent.

6.7 Conclusion

We have introduced a noncoherent coded modulation system which approaches the performance of the coherent ones over the AWGN without the need for carrier phase estimation. The method promises robustness to carrier phase noise and frequency uncertainty, where there is a possible tradeoff between robustness and performance by varying the observation length. It also might be appropriate with fading channels, since

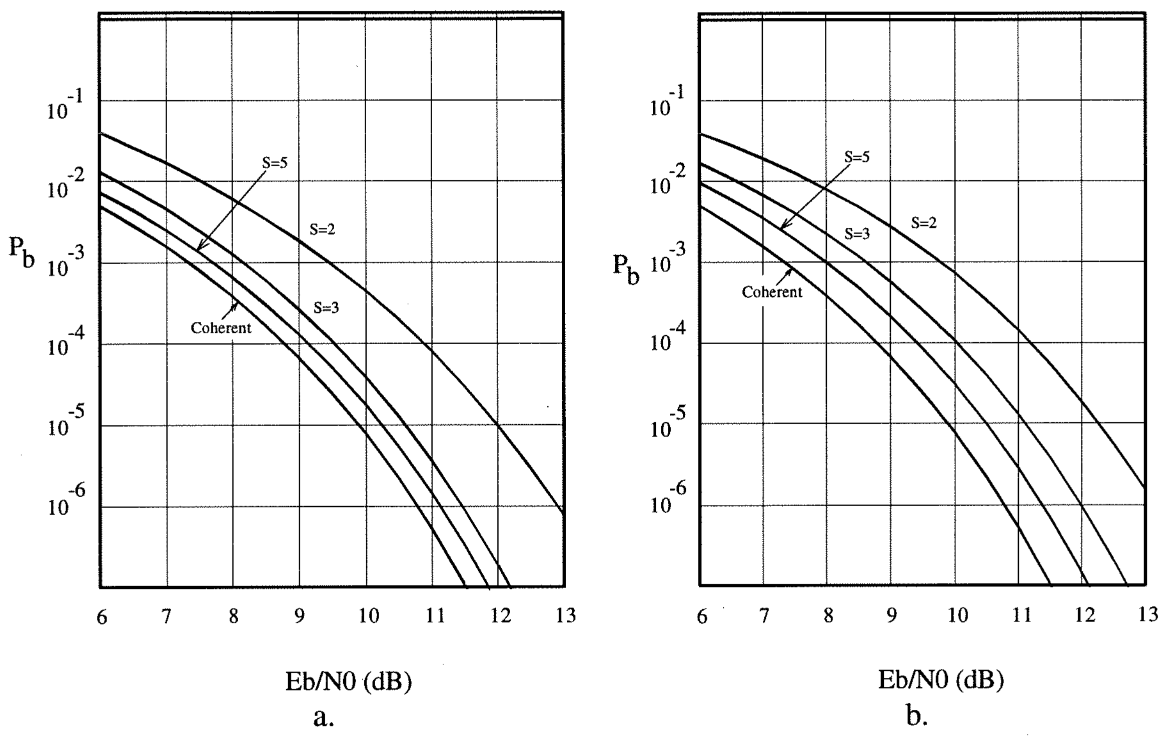


Figure 6.10: a. Performance of DQPSK when decoded by the IO-NMSLE. b. Performance of DQPSK when detected by the block Multiple-Symbol Differential Detector [2].

phase synchronization is difficult in those environments. Efficient sub-optimal decoding algorithms for the NCM are presented in the next chapter.

We have shown the importance of overlapping the observations. We do not need to use interleaving which adds to the system complexity and causes unwanted delay in the received data.

Our method can also be applied to differentially encoded MPSK. We have demonstrated higher performance and faster convergence to coherent detection performance than previously used methods. Our detection method can be applied to other known modulations with promising results.

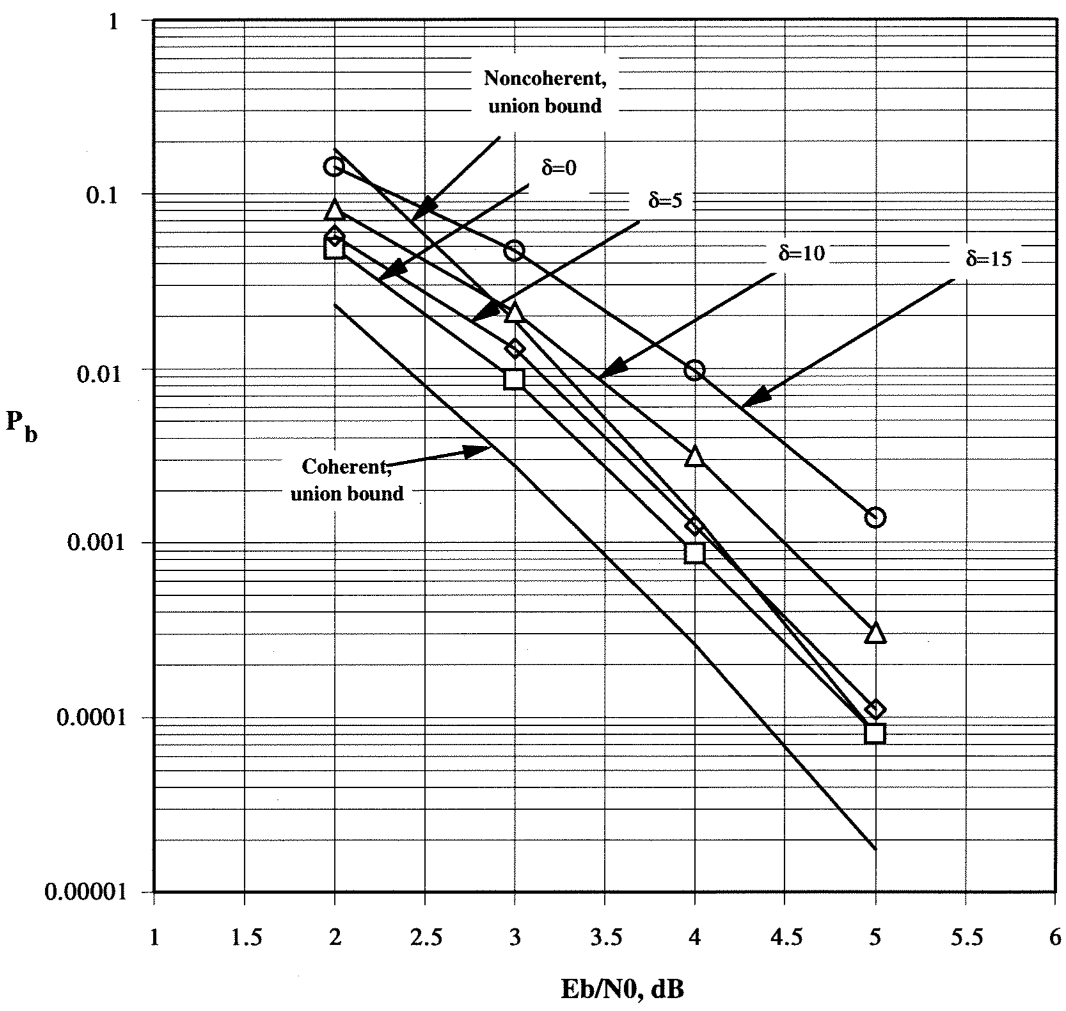


Figure 6.11: Performance in the presence of phase noise, $N = 16$, $R = 4$, $S = 6$.

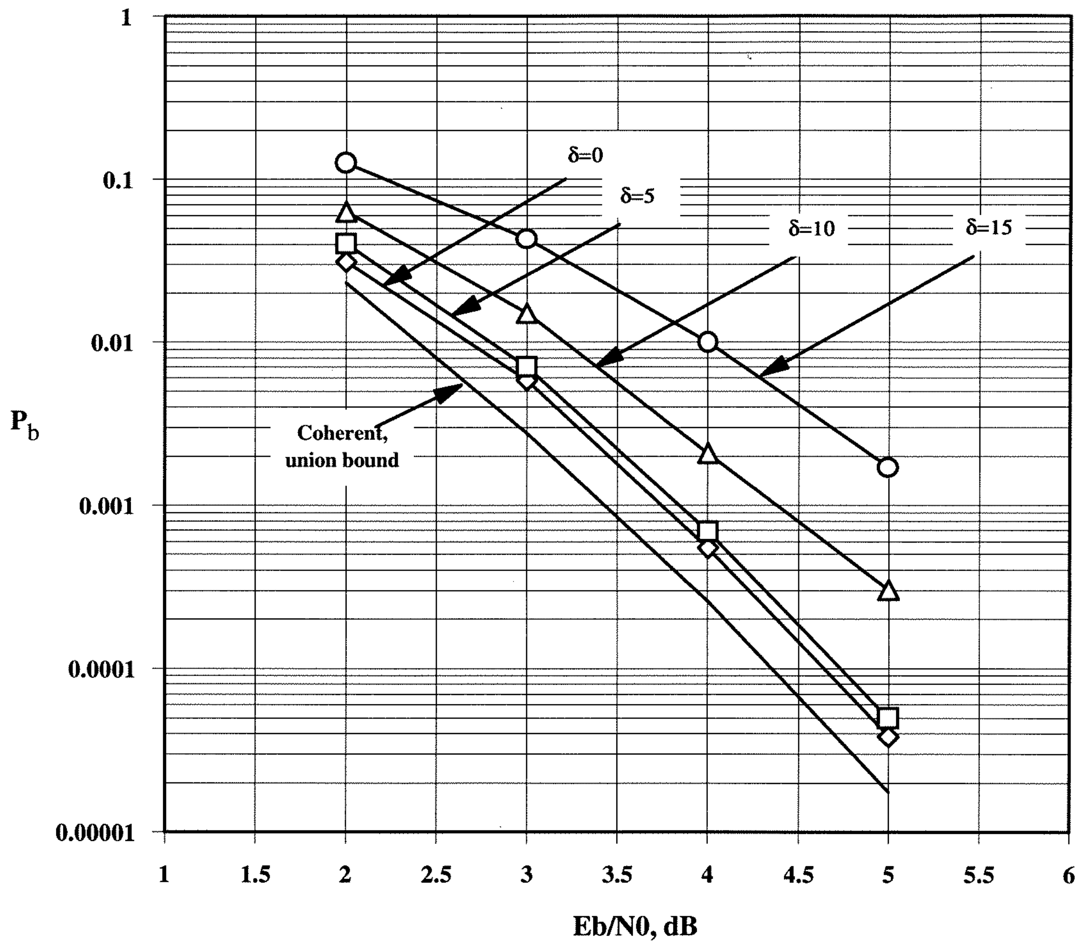


Figure 6.12: Performance in the presence of phase noise, $N = 16$, $R = 4$, $S = 8$.

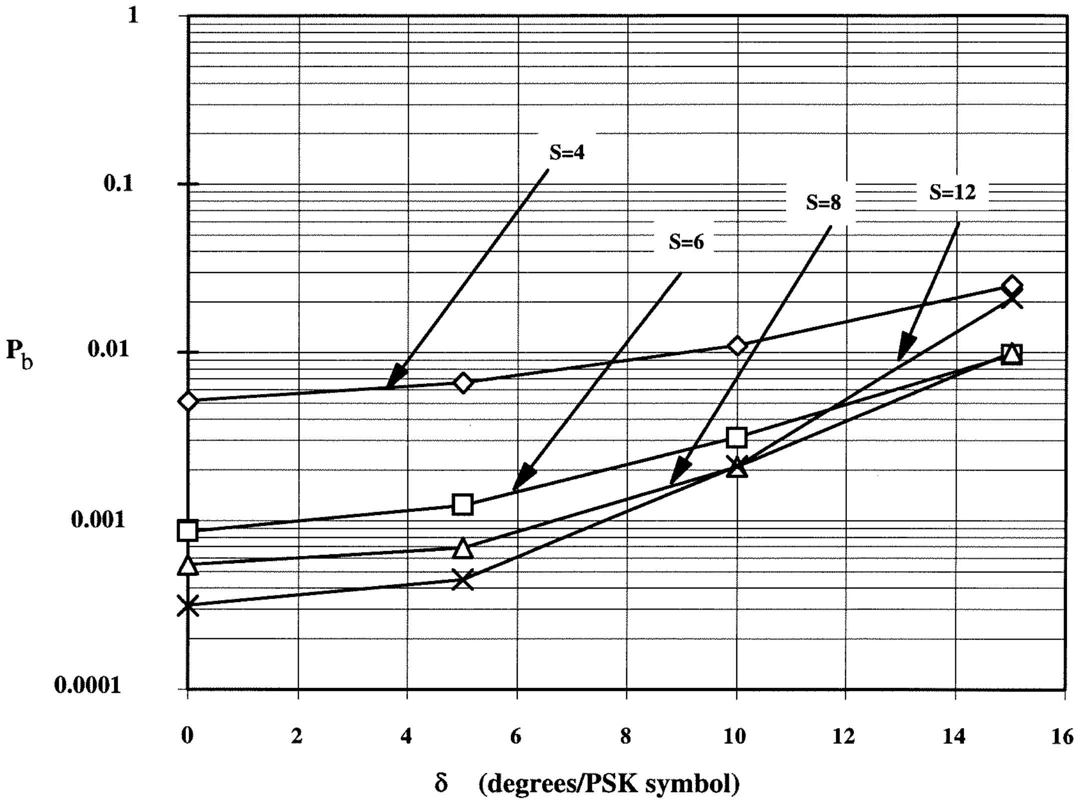


Figure 6.13: Bit error probability versus δ at $E_b/N_0 = 4$ dB.

Bibliography

[1] Clark G.C., Cain J.B., *Error-Correction Coding for Digital Communications* NY: Plenum Press, 1981.

[2] Divsalar D., Simon M.K., "Multiple-Symbol Differential Detection of MPSK," IEEE Trans. Comm., Vol. 38, No. 3, pp. 300–308, Mar. 1990.

[3] Leib H., Pasupathy S., "Optimal Noncoherent Block Demodulation of Differential Phase Shift Keying (DPSK)," AEU Vol. 45, No. 5, pp. 299–305, Sep. 1991.

[4] Edbauer F., "Bit Error Rate of Binary and Quaternary DPSK Signals with Multiple Differential Feedback Detection," IEEE Trans. Comm., Vol. 40, No. 3, pp. 457–460, Mar. 1992.

[5] Aulin T., Sundberg C.E., "Partially Coherent Detection of Digital Full Response Continuous Phase Modulated Signals," IEEE Trans. Comm. Vol. COM-30, No. 5, pp. 1096–1117, May 1982.

[6] Svensson A., Aulin T., Sundberg C.E., "Symbol Error Probability Behavior for Continuous Phase Modulation with Partially Coherent Detection," AEU, Vol. 40, No. 1, pp. 37–45, 1986.

[7] Leib H., Pasupathy S., "Noncoherent Block Demodulation of MSK with Inherent and Enhanced Encoding," IEEE Trans. Comm., Vol. 40, No. 9, pp. 1430–1441, Sep. 1992.

[8] Simon M.K., Divsalar D., "Maximum Likelihood Block Detection of Noncoherent Continuous Phase Modulation," IEEE Trans. Comm., Vol. 41, No. 1, pp. 90–98, Jan. 1993.

[9] Knopp R., Leib H., "Module-Phase Codes with Noncoherent Detection," ICC'93, Geneva, pp. 1054–1058, May 1993.

[10] Divsalar D., Simon M.K., Shahshahani M., "The Performance of Trellis-Coded MDPSK with Multiple Symbol Detection," IEEE Trans. Comm., Vol. 38, No. 9, pp. 1391–1403, Sep. 1990.

[11] Yu K., Ho P., "Trellis Coded Modulation with Multiple Symbol Differential Detection," ICC'93 Geneva, pp. 1414–1418, May 1993.

[12] Kaplan G., Shamai S., "On the Achievable Information Rate of DPSK," IEE Proc., Vol. 139, pp. 311–318, No. 3, 1992.

[13] Proakis J.G., *Digital Communications.* NY: McGraw Hill, 1989.

[14] Murota K., Hirade K., "GMSK modulation for digital mobile radio telephony," IEEE Trans. Comm, vol. COM-29, pp. 1044–1050, July 1981.

[15] Viterbi A.J., Omura J.K., *Principles of Digital Communication and Coding.* McGraw-Hill, 1979.

[16] Johnson N. I., Kotz S., *Continuous Univariate Distributions-2.* NJ: John Wiley & Sons, 1970.

[17] Mathai A.M., Provost S.B., *Quadratic Forms in Random Variables,* New York: Marcel Dekker Inc., 1992.

[18] Korn G.A., Korn T.M., *Mathematical Handbook for Scientists and Engineers*, New York: McGraw-Hill, 1968.

[19] Franklin J.N., *Matrix Theory*, New Jersey: Prentice-Hall, 1993.

Chapter 7

Decoding the Noncoherent Trellis Coded Modulation

7.1 Introduction

In the previous chapter we have introduced the notion of Noncoherent Trellis Coded Modulation (NTCM), the noncoherent decoding of multidimensional trellis coded modulation. The sequence estimator used for the noncoherent decoding is the Independent Overlapped observations Noncoherent Maximum Likelihood Sequence Estimator (IO-NMLSE). In this chapter, we present and evaluate several practical decoding algorithms for NTCM. First, we describe the use of the Viterbi Algorithm (VA) for an optimal decoding by the IO-NMLSE. Optimal decoding requires the use of an augmented trellis diagram with a number of states that grows exponentially with L, the observation length in symbols. Thus, it is practical only for small L. Then, we present three suboptimal algorithms that perform close to optimal, yet with complexity which does not grow exponentially with L. These algorithms, which are based on the VA, use the trellis diagram of the original code. Thus, the number of states does not increase, and the dependence of the complexity on L has linear affinity. The first suboptimal algorithm to be described is called the Basic Decision Feedback Algorithm (BDFA). In this algorithm, the symbols from the decisions are fed back to be used in the subsequent decisions. This algorithm suffers from increased error event probability and from error propagation (to be abbreviated as e.p.). However, by a small modification of the BDFA, we obtain another improved algorithm, which will be called Modified DFA

(MDFA). For some practical codes, degradation of 0.5–1 dB relative to the optimum is demonstrated.

The MDFA still has degradation relative to the optimal algorithm, thus a better algorithm is desired. The third algorithm, the Estimated Future Decision Feedback Algorithm (EFDFA), which uses the BDFA as a basic building block, is based on a novel concept called "estimated future," and performs very close to the optimal in most practical cases. Its degradation in high SNR ($P_b < 10^{-3}$) is negligible. The performance of EFDFA is analyzed by using approximate models due to its complexity.

The degradation of the suboptimal algorithms can be overcome by employing error detection, and processing erroneous blocks off-line using an optimal algorithm. If the probability of error is low, the off-line process can be complex, since more time is available for its completion (causing delay in the decoding of that block).

This chapter is organized as follows. In Section 7.2, we describe the optimal implementation of the IO-NMLSE by the Viterbi algorithm. In Section 7.3, the BDFA is presented, followed by an explanation of its sources of degradation. An improvement of the BDFA called MDFA is outlined in Section 7.4. The EFDFA is presented in Section 7.5 and followed by a detailed example. Performance analysis of the EFDFA is given in Section 7.6, and in Section 7.7 we give a proposed software implementation method. In Section 7.8 we give simulation results which compare the various algorithms.

The encoder uses trellis coded modulation which has the following parameters:

- N - number of states.
- B - number of input bits per branch.
- $R = 2^B$ - number of branches reaching a node.

The trellis is assumed not to contain parallel transitions. However, only minor changes in the algorithms are required to remove this restriction.

7.2 Optimal IO-NMLSE Implementation by the Viterbi Algorithm

It seems natural to choose the Viterbi Algorithm (VA) [1] for implementing the IO-NMLSE. However, the VA cannot be used without modification. We cannot use $\Delta\eta_k$ as the branch metric since it is a function of the current branch value $\mathbf{x}_k$ together with the previous branch values $\{\mathbf{x}_{k-1}, \ldots, \mathbf{x}_{k-L+1}\}$. Since the tentative decisions made by the VA should not affect the following branch metrics, this choice of metric will not cause optimal operation of the VA as a maximization algorithm. If we insist on using this metric in the VA, we get a suboptimal algorithm that will be described in the next section. In order to make the branch metrics independent of previous decisions, we can construct a new trellis diagram with NR^{L-1} states as follows.

A state in the new trellis will be assigned to each of the possible sequences of L consecutive states $\{z_0, \cdots, z_{L-1}\}$ that can be produced by the original trellis. The original transitions from state z_{L-1} to state z_L are mapped to transitions from state $\{z_0, \cdots, z_{L-1}\}$ to state $\{z_1, \cdots, z_L\}$ for all possible choices of $\{z_0, \cdots, z_L\}$. The corresponding branch value is the sequence of symbols $\{\mathbf{x}_0, \cdots, \mathbf{x}_{L-1}\}$ which is the output of the path $\{z_0, \cdots, z_L\}$ on the original trellis. For an example, see Figure 7.1. When using the new trellis, $\Delta\eta_k$ is a function of the branch value only, enabling correct maximizations of the metric (6.5) by the VA. Note that having NR^{L-1} states is sufficient but not necessary.

Note that in some cases the number of states can be reduced since what is important to the independence of the branches' metrics is that the state should correspond to the $L-1$ previous symbols and not necessarily to the $L-1$ previous states.

An alternative, but essentially equivalent, trellis diagram can be constructed in a more simple way as follows. The equivalent code is produced by adding $L-1$ unconnected stages to each of the shift registers which generates the code. These stages are only used for the purpose of delaying the merge of two sequences on the trellis by $L-1$ symbols. During this "waiting period" the outputs of the two sequences are equal and in this way we eliminate the influence of the unknown future after the merge. By

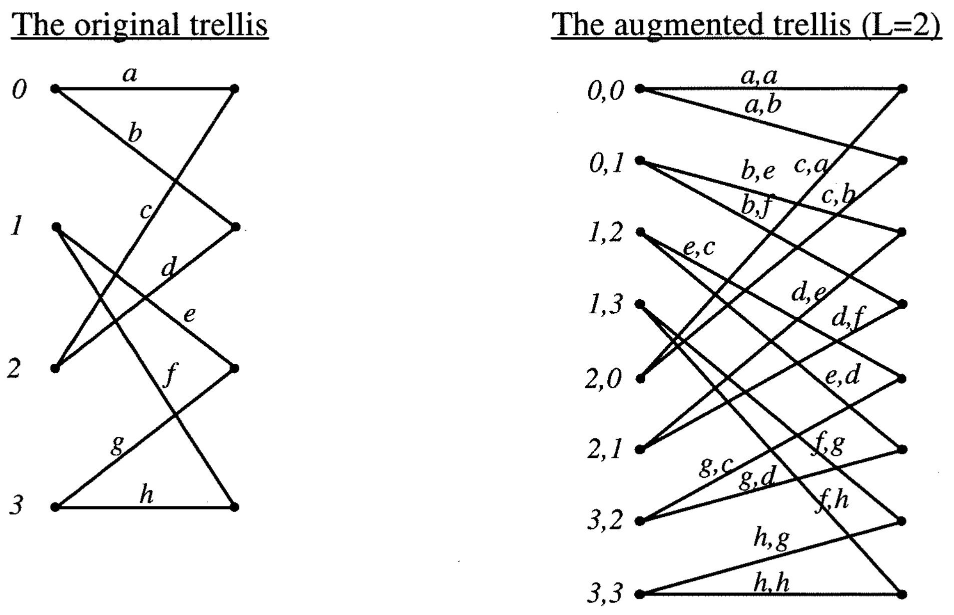


Figure 7.1: An example of the augmented trellis for using the Viterbi Algorithm.

using an example, let us demonstrate that this code has no future dependence. Let us have one shift register in the encoder, $L = 4$ and $K = 3$. We change K to 6 by adding 3 unconnected stages. We have the following two paths leading to the same final state.

$$abcde \xrightarrow{fabcde} fabcd \xrightarrow{gfabcd} gfabc \xrightarrow{hgfabc} hgfab,$$

$$abijk \xrightarrow{fabijk} fabij \xrightarrow{gfabij} gfabi \xrightarrow{hgfabi} hgfab.$$

The numbers are represented in base R (Least significant .. Most significant) such that every letter represents a digit which is contained in one stage of the shift register. The states are shown between the arrows and above the arrows are the shift register contents which determines the output. We can see that in the last three $(L-1)$ steps, the shift register contents differ only in the last three stages, and since they are unconnected, the corresponding outputs are equal. With any common future path, these two candidate paths have the same output symbols sequence for time $t - L + 1$ to $t + L - 1$ where the decision is made at time t. As a result, any two sequences of length L which include future symbols, each taken from one of the two candidate paths, are the same. We see that here the unknown future cannot influence the decision, thus the decisions are optimal.

If the complexity of the decoder is measured by the number of correlations per symbol, we get NR^L correlations. We would like to compare the number of correlations needed to implement the multiple symbol differential detector of Simon and Divsalar [2]. There the number of states is not increased, but every L consecutive symbols (we ignore the overlapped PSK symbol for large L) become one branch. We get $\frac{NR^L}{L}$ correlations per symbol. We see that both complexities are asymptotically comparable.

7.3 The Basic Decision Feedback Algorithm

As was mentioned in the previous section, we would like to use $\Delta\eta_k$ as the metric in the VA, using the original code trellis instead of the augmented one. Doing so, the number states as well as the number of correlations stay constant as L increases. Only the number of complex multiplication per correlation grows with L.

In order to get the metric for the branch connecting the previous state to the current state, the received symbols are correlated with the L last symbols of the candidate path (the concatenation of the survivor path ending in the previous state with the branch connecting the previous to the current state); see Figure 7.2. This metric is added to the accumulated metric of the survivor path. The accumulated metric, like in the VA, is saved in the accumulator array indexed by the previous state number. Note that the metric computation for time t makes use of the decisions made at times $t-1, \ldots, t-L+1$. This can be viewed as if the decisions are fed back to the decoder. Let us now describe the Basic Decision Feedback Algorithm (BDFA) in detail; refer to Figure 7.2.

Let $Z = \{z_0, \cdots, z_k\}$ be a path, i.e., a sequence of states of the trellis. Then define the accumulated metric of the path at time t ($t \leq k$) as

$$\eta(Z)_t = \sum_{k=0}^{t} \left| \sum_{j=0}^{L-1} \mathbf{r}_{k-j}^{\dagger} \mathbf{x}_{k-j} \right|^2, \tag{7.1}$$

where $\mathbf{x}_k$ is the output associated with the trellis branch connecting z_{k-1} with z_k. For $k \leq 0$, $\mathbf{x}_k = 0$ by definition. If t is omitted, then we use the entire path, i.e., $\eta(Z) = \eta(Z)_k$.

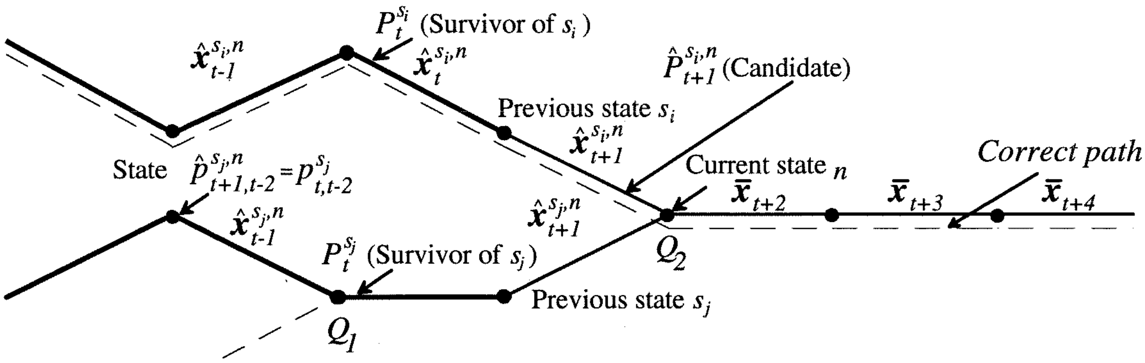


Figure 7.2: Operation of the BDFA and the MDFA.

For each state s and time t, the algorithm keeps track of the associated survivor, i.e., the most likely path of previous states,

$$P_t^s = \{p_{t,0}^s, \cdots, p_{t,t-1}^s, p_{t,t}^s\}, \quad p_{t,t}^s \equiv s, \tag{7.2}$$

where $p_{t,k}^s$ denotes the state indexed by k in the list P_t^s. The last state in the list, $p_{t,t}^s$, is always the current state s. Only the last M (truncation length) states are actually stored by the decoder. The algorithm also keeps track of an accumulator metric

$$J_t^s = \eta(P_t^s). \tag{7.3}$$

Out of all N data paths P_t^s, the one with the largest metric value, J_t^s, has the greatest likelihood. The input data which corresponds to the branch in that path connecting the two "oldest" states $p_{t,t-M}^s$ and $p_{t,t-M+1}^s$ serves as the final estimate of the BDFA of the transmitted data delayed by $M-1$ symbols. To achieve near-MLSE performance, M should be much larger than the memory, K, of the encoder. In our simulations we have used $M > 5K$, like in the VA, and the survivors have always converged as expected. Since the BDFA is a recursive process, it contains provisions for the recursive update of P_t^s and J_t^s. Denote the next state that follows the present state s for the input symbol $0 \leq i < R$ by next(s,i). Each path P_t^s is appended with each one of the states $n_i = \text{next}(s,i)$, $i = 0, \ldots, R-1$, to form the candidate paths $\hat{P}_{t+1}^{s,n_i} = \{p_{t,0}^s, \cdots, p_{t,t-1}^s, s, n_i\}$. There are R paths $\hat{P}_{t+1}^{s_i,n}$ which end in state n. For each of them, $\hat{J}^{s_i,n}_{t+1}$ is computed by the formula

$$\hat{J}^{s_i,n}_{t+1} = J^{s_i}_t + \left| \sum_{j=0}^{L-1} \mathbf{r}^{\dagger}_{t+1-j} \hat{\mathbf{x}}^{s_i,n}_{t+1-j} \right|^2 , \tag{7.4}$$

where $\{\hat{\mathbf{x}}^{s_i,n}_k\}$ are the output symbols of the path $\hat{P}^{s_i,n}_{t+1}$. The index i which maximizes $\hat{J}^{s_i,n}_{t+1}$ is used to update J^n_{t+1} and P^n_{t+1}, by $J^n_{t+1} = \hat{J}^{s_i,n}_{t+1}$ and $P^n_{t+1} = \hat{P}^{s_i,n}_{t+1}$ correspondingly.

This algorithm seems likely to result in a survivor path that has the largest metric. This is true most of the time, but not always. The VA assumes that when a decision is made, it does not depend on future decisions. This will not apply in our case. We are deciding $L-1$ symbols too early. We should have decided on the branch of a state in time t, after we included the contribution of the symbols on the path emerging from this state continuing up to time $t+L-1$. After time $t+L-1$, there no longer is a dependence between the future symbols and the current decisions. If we want to make the correct decision, we may want to include the future path of $L-1$ symbols to the candidate paths of our current decision. But, for every state, there are R^{L-1} possible future paths for which we will have to consider each (or at least consider the most probable one, as will be done in the EFDFA). Thus, the complexity of the decoder is increased at least by R^{L-1}, as in Section 7.2. Alternatively, we can convert the code to an equivalent one by adding $L-1$ unconnected stages to the encoder shift register. When such code is used, the BDFA becomes the optimal algorithm, as was explained in Section 7.2.

The suboptimal algorithm causes degradation through two processes. The first is an increase in the error event probability and the second is e.p. The increase in the error event probability can be explained as follows. Each time we make a decision we essentially compare two truncated sequences instead of infinite sequences, as in the IO-NMLSE. The sequences are truncated at the current symbol when we compare and make the decision. In order to make the correct decision, we have to compare the metric of two infinite sequences (both of which are assumed to follow the correct path after the current decision point). However, instead of using infinite paths, it is sufficient to include only the future $L-1$ symbols in the comparison to make the optimal decision. For example, with $L=4$ it is better to compare the metric (equation 6.5) of the two sequences $\{1,1,1,-1,-1,-1,1,1,1\}$ and $\{1,1,1,1,1,1,1,1,1\}$ than to compare the metric of the truncated sequences $\{1,1,1,-1,-1,-1\}$ and $\{1,1,1,1,1,1\}$. The first error event probability of the BDFA can be evaluated by the union bound when using the truncated sequences instead of infinite sequences. Only two sequences which pass through the same state are truncated as described when the decoder makes its decision at that state. If there exists a path which has output symbols which are a constant phase shift of the transmitted path symbols, this path is equally likely to be decoded instead of the correct path. This leads to an essentially infinite number of decoding errors if the incorrect path is decoded to different output bits. However, if that incorrect path is decoded to the same output bits, then there is no catastrophic behavior (see Chapter 4). In this case, additional error events are possible: The paths that diverge from the true path and reach a path with symbols which are a constant phase shift of the transmitted path symbols. For these error events, there is no truncation because the decisions are being made at separate states.

Let us now discuss the e.p., which is the main cause of degradation. After an error occurs in the decision process, the error probability for future decisions increases. If a new error occurs, further future decisions may be in error in a chain reaction-like fashion. This process of e.p. leads to large error bursts. For some applications this phenomena is not crucial. For example, if a message must be transmitted without error, and any number of errors cause the message to be discarded and re-transmitted.

The mechanism of the e.p. is explained by referring to Figure 7.3. Suppose that Path 1 was transmitted. At point B, a wrong decision has been made (due to the noise), i.e., $\eta(\text{path }1)_B < \eta(\text{path }2)_B$. Suppose that we could have decided at point B which of the sequences 1 or 2 was transmitted using the accumulated metric up to point A, not up to point B, i.e., using $\eta(\cdot)_A$ instead of $\eta(\cdot)_B$. Then we might have decided on Path 1 instead. Let us assume that this is the case, i.e., $\eta(\text{path }1)_A > \eta(\text{path }2)_A$. Returning to the algorithm, at point A we now decide between Path 3 and Path 2. Path 3 might win even though it would have lost had it been compared to Path 1 at point A. Even though Path 1 has the maximal metric up to point A, Path 3 has won due to the wrong decision made at point B. Path 3 can deviate from the true path by more symbols than the original error, i.e., Path 2, and still win. After making the first wrong decision, candidate paths will not have to win over the maximal path but over a path whose metric is less than the maximal. The e.p. can continue as long as there are candidates that pass the lower threshold. It is very difficult to predict when this process stops.

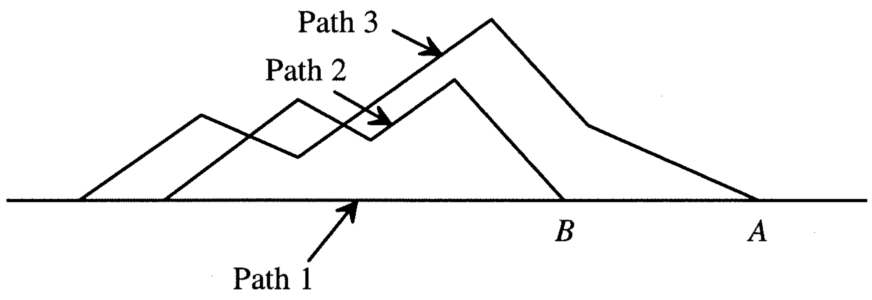


Figure 7.3: Explanation of the error propagation. Path 3 may win at point A due to an error that was made at point B.

Let us investigate the case where we can use the future symbols of the path in the decisions. In this case we argue that even if we make an error, no e.p. will result. Let us suppose that we could have decided at point B which of the sequences 1 or 2 were transmitted using $\eta(\cdot)_A$ instead of $\eta(\cdot)_B$. Suppose that Path 2 won again. In this case no e.p. can occur because when Path 2 was decided at point B (although it is an error), Path 2 had indeed the largest metric (up to point A) so we are truly maximizing the path metric between these three paths. Note that if point A is less than $L-1$ symbols away from point B, e.p. can still occur in later decisions. An important and not very intuitive fact is that for the same input, error events that occur both in the optimal algorithm and in the BDFA will not cause e.p. in the BDFA.

In the analysis of the following EFDFA algorithm, we will assume that the e.p. length can be modeled by a geometric distribution. This model results from the assumption that whenever we are already in an e.p. event, the probability that it will stop is the same no matter when this e.p. began. In other words the e.p. is assumed to be memoryless. This model holds well for large e.p. events. The validity of the model is demonstrated in Figure 7.4, where the model fits the measured data taken by simulation of the BDFA.

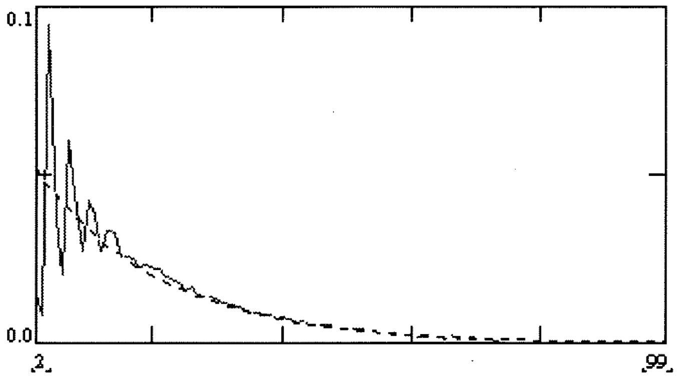


Figure 7.4: Measured data (solid line) against model (geometric distribution, dashed line) of the e.p. length. The code used has 4 states, $L = 3$, and $E_b/N_0 = 5$ dB.

7.4 The Modified Decision Feedback Algorithm

We can improve the decision process of the BDFA even without knowing the future $L-1$ states emerging from the current state. The new algorithm will be referred to as Modified Decision Feedback Algorithm (MDFA). Let $\bar{\mathbf{x}}_i$ be the transmitted symbols. Let us assume that the symbols $\bar{\mathbf{x}}_{t+2}, \cdots, \bar{\mathbf{x}}_{t+L}$ are known to the decoder when it has to make a decision at time $t+1$; see Figure 7.2. If n is on the correct path, the optimal decision rule is then the following. For each of the new paths $\hat{P}_{t+1}^{s_i,n}$, we compute $\hat{J}_{t+L}^{s_i,n}$ by

$$\hat{J}_{t+L}^{s_i,n} = J_t^{s_i} + \sum_{k=t+1}^{t+L} \left| \sum_{j=0}^{L-1} \mathbf{r}_{k-j}^{\dagger} \hat{\mathbf{x}}_{k-j}^{s_i,n} \right|^2, \tag{7.5}$$

where $\hat{\mathbf{x}}_j^{s_i,n} = \bar{\mathbf{x}}_j$ (the known symbols) if $j > t+1$. If the next state n where we make the decision is not on the correct path, then there is no meaning to this expression, and using it may cause wrong decisions. Such a wrong decision can only change the paths competing with the correct path in later decisions. For example, in Figure 7.2 wrong decision at point Q_1 will change the candidate path competing with the correct path at point Q_2. Assuming that the correct path has the largest metric, no other path will win above it no matter which path it is. Thus, the decoding error probability will not increase.

We can express the received signal as $\mathbf{r}_j = \alpha\bar{\mathbf{x}}_j e^{j\theta} + \mathbf{n}_j$, where θ is the carrier phase and α is the channel attenuation (both assumed constant over $2L$ symbols, but unknown). Since we use constant amplitude symbols such that $\bar{\mathbf{x}}_j^\dagger \bar{\mathbf{x}}_j = 1$,

$$\mathbf{r}_j^\dagger \bar{\mathbf{x}}_j = \alpha e^{-j\theta} + \mathbf{n}_j^\dagger \bar{\mathbf{x}}_j, \quad j = t-L+2, \cdots, t+L. \tag{7.6}$$

For the correct path (n is correct and s_i is correct), $\hat{\mathbf{x}}_j^{s_i,n}$ for $j = t-L+1, \cdots, t+1$, are the correct symbols. Let us define

$$\mu = \sum_{j=t-L+1}^{t+1} \mathbf{r}_j^\dagger \hat{\mathbf{x}}_j^{s_i,n} = L\alpha e^{-j\theta} + \sum_{j=t-L+1}^{t+1} \mathbf{n}_j \hat{\mathbf{x}}_j^{s_i,n}. \tag{7.7}$$

Provided that the SNR in L symbols is high enough, $\alpha e^{-j\theta}$ can be estimated by $\frac{\mu}{L}$. Then we can use $\frac{\mu}{L}$ as an estimate for $\mathbf{r}_j^\dagger \bar{\mathbf{x}}_j$, whenever $j > t+1$ and use it in (7.5). We tested several other ways to estimate $\alpha e^{-j\theta}$, but none were more successful than this simple method. To summarize, the change in the algorithm is the following: Compute

$$\mu = \sum_{j=0}^{L-1} \mathbf{r}_{t+1-j}^\dagger \hat{\mathbf{x}}_{t+1-j}^{s_i,n}, \tag{7.8}$$

$$\hat{J}_{t+1}^{s_i,n} = J_t^{s_i} + |\mu|^2, \tag{7.9}$$

and

$$\tilde{J}_{t+L}^{s_i,n} = J_t^{s_i} + \sum_{k=t+1}^{t+L} \left| \sum_{j=0}^{L-1} \left\{ \begin{array}{ll} \mathbf{r}_{k-j}^\dagger \hat{\mathbf{x}}_{k-j}^{s_i,n}, & \text{if } k-j \le t+1 \\ \frac{\mu}{L}, & \text{otherwise} \end{array} \right\} \right|^2. \tag{7.10}$$

$\tilde{J}_{t+L}^{s_i,n}$ is used as an estimate for $\hat{J}_{t+L}^{s_i,n}$. The index i which maximizes $\tilde{J}_{t+L}^{s_i,n}$ is used to update J_{t+1}^n and P_{t+1}^n, by $J_{t+1}^n = \hat{J}_{t+1}^{s_i,n}$ and $P_{t+1}^n = \hat{P}_{t+1}^{s_i,n}$. The algorithm was found to reduce the number of e.p. events significantly. Although this algorithm cannot stop e.p. once it has begun (since then $\frac{\mu}{L}$ is not a good estimate for $\alpha e^{-j\theta}$), it can reduce the probability of continuing the e.p. event. It was found that the length of the error bursts was reduced considerably compared to the BDFA.

7.5 Estimated Future Decision Feedback Algorithm

The MDFA still has degradation compared to the optimal algorithm. Thus, a better algorithm is desired. A novel algorithm, which performs very close to the optimal algorithm, but with significantly lower complexity, was found. This algorithm, like the BDFA, uses the original code trellis. On the other hand, the optimal algorithm, uses an augmented trellis with a large number of states. The EFDFA complexity is roughly 4 times that of the BDFA.

The algorithm uses a novel concept called *estimated future* to improve the decision process. We have previously recognized that we need to include the future path to make the current decision in order to make optimal decisions in the BDFA. If such a future path is given, but it is not completely reliable, we call it estimated future. The algorithm works as follows. In each trellis state, at each time, two independent decisions are being made. One is called the no-future (n.f.) decision, which is similar to the BDFA, and the other is a decision using the $L-1$ estimated future symbols and it is called the with-future (w.f.) decision. The first suffers from e.p. and increased sequence error as discussed in the previous section. The second will make the best decisions as long as the future sequence which is being used is the true one. On the other hand, if wrong future is used for the w.f. decision, then it can cause an error. However, utilizing both the n.f. and the w.f. decisions, one can arrive at a combined decision which chooses the right sequence in an optimal way.

How can one possibly know the future? The approach is to save a block of the input signal in memory and perform the DFA (BDFA or MDFA) backwards, starting from the end of the block. After the backward process ends, we have the survivor paths belonging to each state at each time in the trellis within the block. These paths will be used as future estimates. The future estimation performance by this method is as good as the performance of the reversed code (reversing the generators) when decoded with a DFA. The performances of the code and its reversed version can be different only because of the suboptimality of the DFA.

The input stream is divided into overlapped blocks, each block having a length of

$A+W$ symbols and starts A symbols after the beginning of the previous block; see Figure 7.5. The backward process (to be abbreviated by b.p.) operates first and processes the whole block. Then the forward process (to be abbreviated by f.p.) operates on the first A symbols of the block. The f.p. is continuous from block to block. The blocking is intended only for the operation of the b.p. The section of length W is intended for letting the b.p. converge from the initial conditions, in which no particular state is used as a beginning state. This convergence region can be eliminated by inserting $K + B(L-2)$ known input bits at the end of each block of BA input bits, where K is the code memory, and in that way reach a known state at the end of each block.

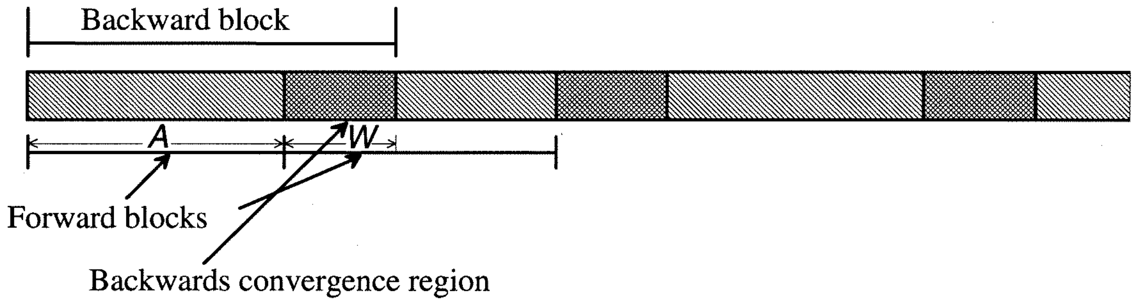


Figure 7.5: How to get the estimated future.

The f.p. and the b.p. operating on the same block are highly correlated. When there is an error event in the f.p., similar error event is also likely to occur in the b.p. However, the initial error and the e.p. following it in the b.p. is going to the other direction—towards the past, leaving the estimated future intact at the time needed for correcting the forward e.p.; see Figure 7.6.

Returning to the issue of the convergence of the b.p., the first decisions in the b.p. are made using very few symbols. In particular, the first decision is based only on the last symbol in the block (remember, we are going backwards so this last symbol is our first). This means that it is probable that we start with an error. Since the b.p. suffers from e.p., the convergence time is similar to the time needed to recover from e.p. This is why we recommend to use the MDFA for the b.p. instead of BDFA.

For the algorithm description that follows, we are using the BDFA decisions in

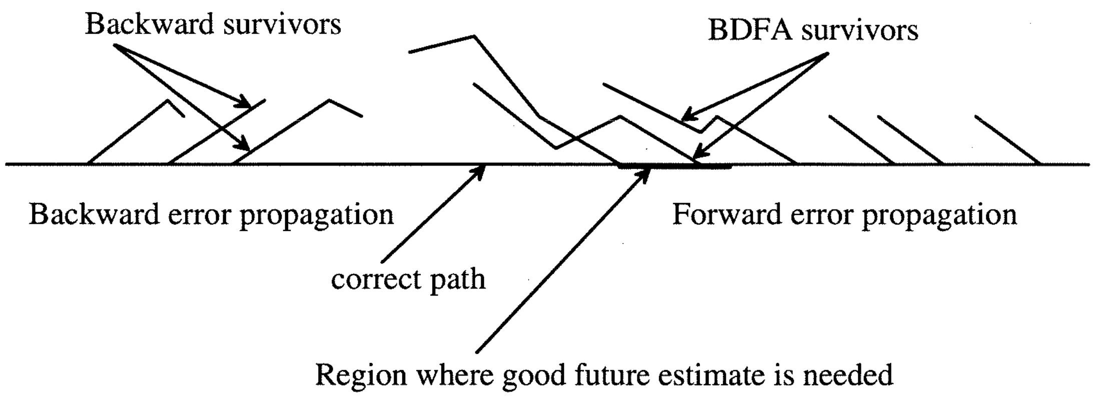

Backward survivors
BDFA survivors
Backward error propagation
Forward error propagation
correct path
Region where good future estimate is needed

Figure 7.6: Explanation of why the backward errors resulting from the same noise instance does not harm the future estimate needed for solving the forward e.p.

both the backward decisions and the forward n.f. decisions. Each one of them can be replaced by the modified version without altering the principle of operation.

7.5.1 The Backward Process

The b.p. operates on a block of length $A+W$, $kA \leq t < (k+1)A+W$. This is exactly the BDFA going back in time. For each state s, the algorithm keeps track of the associated survivor, i.e., the most likely path of previous states

$$Q_t^s = \{q^s_{t,(k+1)A+W}, q^s_{t,(k+1)A+W-1}, \cdots, q^s_{t,t+1}, q^s_{t,t}\}, \quad q^s_{t,t} \equiv s, \tag{7.11}$$

and also of an accumulator metric

$$E_t^s = \sum_{k=t}^{(k+1)A+W} \left| \sum_{j=1}^{L} \mathbf{r}^\dagger_{k+j} \mathbf{x}^s_{k+j} \right|^2, \tag{7.12}$$

where $\mathbf{x}^s_k$ is the output of the trellis branch connecting $q^s_{t,k-1}$ with $q^s_{t,k}$. Denote the previous state which produced s by moving one step forward in the code trellis for the input symbol $0 \leq i < R$ by previous(s,i). For the recursive update of Q_t^s and E_t^s, each path Q_t^s is appended with each one of the states $p_i = \text{previous}(s,i)$, $i = 0, \ldots, R-1$ to form the candidate paths $\hat{Q}^{s,p_i}_{t-1} = \{q^s_{(k+1)A+W}, q^s_{(k+1)A+W-1}, \cdots, q^s_{t-1}, s, p_i\}$. There are R paths $\hat{Q}^{s_i,p}_{t+1}$ which end at state p. For each of them, $\hat{E}^{s_i,p}_{t-1}$ is computed by the formula

$$\hat{E}^{s_i,p}_{t-1} = E^{s_i}_t + \left| \sum_{j=0}^{L-1} \mathbf{r}^\dagger_{t+j} \hat{\mathbf{x}}^{s_i,p}_{t+j} \right|^2, \tag{7.13}$$

where $\{\hat{\mathbf{x}}_k^{s_i,p}\}$ are the output symbols of the path $\hat{Q}_{t-1}^{s_i,p}$. The index i which maximizes $\hat{E}_{t-1}^{s_i,p}$ is used to update E_{t-1}^p and Q_{t-1}^p.

7.5.2 The Forward Process

Given the estimated future (contained in Q), the f.p. works as follows:

For each state s, the algorithm keeps track of two associated survivors. The first, C_t^s, is called the n.f. survivor and is our best guess of the ML path.

$$C_t^s = \{\cdots, c_{t,t-k}^s, c_{t,t-k+1}^s, \cdots, c_{t,t-1}^s, c_{t,t}^s\}, \quad c_{t,t}^s \equiv s. \tag{7.14}$$

Here $c_{t,k}^s$ denotes the state indexed by k in the list C_t^s. For every path C_t^s there is an associated accumulator metric

$$G_t^s = \eta(C_t^s). \tag{7.15}$$

The second survivor, F_t^s, is called the w.f. survivor and is used as a temporary variable.

$$F_t^s = \{\cdots, f_{t,t-k}^s, f_{t,t-k+1}^s, \cdots, f_{t,t-1}^s, f_{t,t}^s, f_{t,t+1}^s, \cdots, f_{t,t+L-1}^s\}. \tag{7.16}$$

Its associated accumulator metric is

$$H_t^s = \eta(F_t^s). \tag{7.17}$$

F is constructed such that

$$f_{t,t}^s = s \quad \text{and} \quad f_{t,t+i}^s = q_{t,t+i}^s, \quad i = 1, \cdots, L-1. \tag{7.18}$$

F is the path that includes the future estimate. It extends from the state s at time t, $L-1$ symbols, towards the future. As in the implementation of the VA, only M (truncation length) last states need to be saved in each path list. The algorithm works recursively. Each time we do the following:

Step 1. For each state s, form the w.f. candidate path $\hat{F}_{t+1}^{s,m}$, $\quad m = q_{t,t+1}^s$, by appending the state $q_{t,t+L}^s$ to F_t^s.

Step 2. For each $\hat{F}_{t+1}^{s,m}$, compute the accumulated metric $\hat{H}_{t+1}^{s,m}$ by

$$\hat{H}_{t+1}^{s,m} = \eta(\hat{F}_{t+1}^{s,m}) = H_t^s + \left|\sum_{j=0}^{L-1} \mathbf{r}_{t+1}^\dagger \hat{\mathbf{y}}_{t+1-j}^{s,m}\right|^2, \tag{7.19}$$

where $\{\hat{\mathbf{y}}_k^{s,m}\}$ denotes the output symbols of $\hat{F}_{t+1}^{s,m}$.

Step 3. For each state s, find the next states $n_i = \text{next}(s,i) \quad i = 0, \ldots, R-1$. For every $n_i \neq m$ form the w.f. candidate path $\hat{F}_{t+1}^{s,n_i}$ by appending the sequence $\{n_i, q_{t+1,t+2}^{n_i}, q_{t+1,t+3}^{n_i}, \cdots, q_{t+1,t+L}^{n_i}\}$ to C_t^s.

Step 4. For each $\hat{F}_{t+1}^{s,n_i}$, compute the accumulated metric $\hat{H}_{t+1}^{s,n_i}$ by

$$\hat{H}_{t+1}^{s,n_i} = \eta(\hat{F}_{t+1}^{s,n_i}) = G_t^s + \sum_{k=t+1}^{t+L} \left| \sum_{j=0}^{L-1} \mathbf{r}_{k-j}^{\dagger} \hat{\mathbf{y}}_{k-j}^{s,n_i} \right|^2, \tag{7.20}$$

where $\{\hat{\mathbf{y}}_k^{s,n_i}\}$ denotes the output symbols of $\hat{F}_{t+1}^{s,n_i}$.

Step 5. (w.f. decision) For each state w, there are a total of R values $\hat{H}_{t+1}^{s_i,w}$. The index i which maximizes $\hat{H}_{t+1}^{s_i,w}$ is used to update H_t^w and F_t^w by $H_{t+1}^w = \hat{H}_{t+1}^{s_i,w}$ and $F_{t+1}^w = \hat{F}_{t+1}^{s_i,w}$.

Step 6. For each state s and input $i = 0, \ldots, R-1$, form the n.f. candidate path $\hat{C}_{t+1}^{s,n_i}$, by appending the state $n_i = \text{next}(s,i)$ to C_t^s.

Step 7. For each $\hat{C}_{t+1}^{s,n_i}$, compute the accumulated metric $\hat{G}_{t+1}^{s,n_i}$ by

$$\hat{G}_{t+1}^{s,n_i} = \eta(\hat{C}_{t+1}^{s,n_i}) = G_t^s + \left| \sum_{j=0}^{L-1} \mathbf{r}_{t+1-j}^{\dagger} \hat{\mathbf{x}}_{t+1-j}^{s,n_i} \right|^2, \tag{7.21}$$

where $\{\hat{\mathbf{x}}_k^{s,n_i}\}$ denotes the output symbols of the path $\hat{C}_{t+1}^{s,n_i}$.

Step 8. (n.f. decision) For each state w, find all states u_j, $j = 0, \cdots, l-1$, such that the path $F_{t-L+2}^{u_j}$ ends with w, i.e., $q_{t-L+2,t+1}^{u_j} = w$. For an empty set, $l = 0$. Including the R values of the form $\hat{G}_{t+1}^{s_i,w}$, we define

$$\alpha_i = \left\{ \begin{array}{ll} \hat{G}_{t+1}^{s_i,w}, & \text{if } i < R \\ H_{t-L+2}^{u(i-R)}, & \text{if } R \leq i < R+l \end{array} \right\}, \quad i = 0, \cdots, R+l-1. \tag{7.22}$$

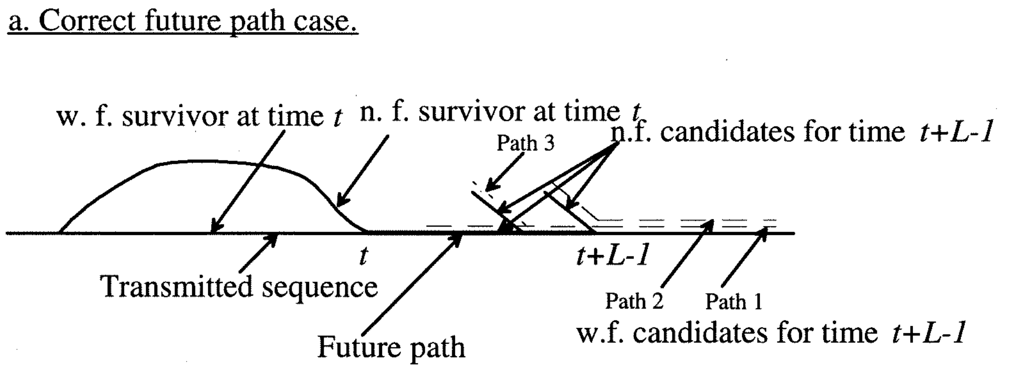
The index i which maximizes α_i is used to update C_{t+1}^w and G_{t+1}^w by $G_{t+1}^w = \alpha_i$ and

$$C_{t+1}^w = \left\{ \begin{array}{ll} \hat{C}_{t+1}^{s_i,w}, & \text{if } i < R \\ F_{t-L+2}^{u(i-R)}, & \text{if } R \leq i < R+l \end{array} \right\}. \tag{7.23}$$

Step 9. Find the s which maximizes G_{t+1}^s. The input data which corresponds to the branch connecting the two "old" states $c_{t+1,t-M}^s$ and $c_{t+1,t-M+1}^s$ serves as the decoder output, where M is the decoder memory length.

The operation of the algorithm is as follows (refer also to Figure 7.7). C_t^s always holds the best survivor path that we have at time t to state s. Its update is done in Step 8, the n.f. decision, where two kinds of competitors are compared: The n.f. candidate paths which are extensions by one symbol of the previous n.f. survivor and are not using the future estimate, and the w.f. candidate paths which are found in F. The w.f. candidate paths are the survivors of past decisions that used the future estimates. These decisions were made at time $t - L + 1$, while the path is used at time t. When the w.f. survivor is used, the $L - 1$ states that used to be the future at time $t - L + 1$ are the past for time t, and the last state becomes the current one. In case the future estimation was correct, that w.f. path is a result of an optimal decision. In this case this path will either win or its metric will be equal to the maximal n.f. path metric. A win condition indicates that wrong decisions have been made which might lead to e.p., unless corrected by the winning path from F. Equality between the w.f. path and the n.f. candidate indicates that the previous n.f. decisions were correct leading to the same survivor. Correct decisions mean that the decisions are leading to the ML path, not necessarily to the transmitted path. In order to update F, we make the decisions about the survivor path at time t, using the future $L - 1$ symbols which are assumed to be correct. The candidates that pass through the state s at time t are divided into two categories. The first category contains paths that are extensions of previous w.f. paths, by appending the next state from the future estimate path (Step 1), like Path 1 in Figure 7.7. The second category contains n.f. paths combined with the estimated future path (Step 3), like Path 2 and Path 3 in Figure 7.7. The paths of the second category which share their last L states with paths of the first category, like Path 3, are eliminated from the comparison since those two candidates have already been compared in the previous w.f. decision. This elimination is taking place in Step 3 checking that the next state n_i is not equal to the next state from the estimated future, m.

Example 7.5.1 *Refer to Figure 7.8. We would like to decode a rate $\frac{1}{2}$, $K = 3$ code with BPSK modulation, and we choose $L = 3$. The input to the decoder is as shown, where the numbers are real for simplicity. The symbols are in general complex vectors.*


a. Correct future path case.
w. f. survivor at time t
n. f. survivor at time t
Path 3
n.f. candidates for time $t+L-1$
t
$t+L-1$
Transmitted sequence
Future path
Path 2
Path 1
w.f. candidates for time $t+L-1$

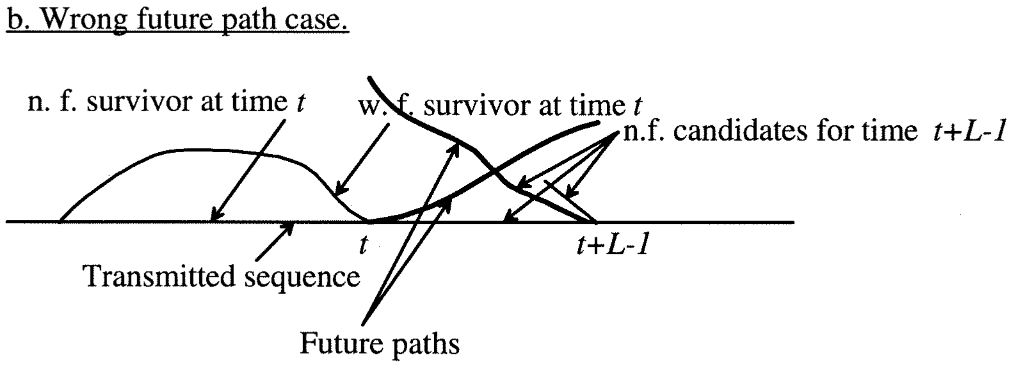

b. Wrong future path case.
n. f. survivor at time t
w. f. survivor at time t
n.f. candidates for time $t+L-1$
t
$t+L-1$
Transmitted sequence
Future paths

Figure 7.7: Explanation of the Estimated Future Decision Feedback Algorithm. In the first example (a) the n.f. decision at time t was in error. Correct future estimate enabled the w.f. decision to be correct. The n.f. error caused error propagation in the n.f. decisions. The w.f. survivor of time t became a candidate for the n.f. decision at time $t+L-1$, and wins over all the wrong n.f. candidates, leading to elimination of the error. In the second example (b), the n.f. decision was correct, but the future estimate was wrong, leading to w.f. decision error. At time $t+L-1$, the n.f. candidates did not include the wrong w.f. survivor, but has included other (arbitrary) w.f. candidate. The candidate which coincides with the transmitted path had the largest metric and no error occurred.

Initial conditions: 0 is received for $t < 0$ and the state of the encoder is 0 at time $t = -1$, where all paths originate (it is not needed for the algorithm operation, only for the example). For $t < 2$ we only compute the metrics—there are no decisions.

At $\mathbf{t} = \mathbf{0}$:

$$F_0^0 = \{0, 0, 0, 0\},$$

$$H_0^0 = (-0.5-1.5)^2+(-0.5-1.5+0.5+0.8)^2+(-0.5-1.5+0.5+0.8-0.9-0.9)^2 = 10.74,$$

$$F_0^1 = \{0, 1, 2, 0\},$$

$$H_0^1 = (0.5+1.5)^2 + (0.5+1.5-0.5+0.8)^2 + (0.5+1.5-0.5+0.8+0.9+0.9)^2 = 26.1,$$

$$C_0^0 = \{0, 0\}, \quad G_0^0 = (-0.5-1.5)^2 = 4,$$

$$C_0^1 = \{0, 1\}, \quad G_0^1 = 4.$$

Step 1,2: *$s = 0 \to m = 0$ and $s = 1 \to m = 2$.*

$$\hat{F}_1^{0,0} = \{0, 0, 0, 0, 0\}, \quad \hat{H}_1^{0,0} = 10.74 + (0.5+0.8-0.9-0.9-1-1)^2 = 16.99,$$

$$\hat{F}_1^{1,2} = \{0, 1, 2, 0, 0\}, \quad \hat{H}_1^{1,2} = 26.1 + (-0.5+0.8+0.9+0.9-1-1)^2 = 26.11,$$

Step 3,4: *For $s = 0$, $\{n_i\} = \{0, 1\}$. $m = 0$ so we take only $n_i = 1$.*

$$\hat{F}_1^{0,1} = C_0^0 + \{1, 2, 0\} = \{0, 0, 1, 2, 0\},$$

$$\{\hat{y}_i^{0,1}\} = \{(1,1), (-1,-1), (-1,1), (-1,-1)\}, \quad \text{for } i = 0 \ldots 3,$$

$$\hat{H}_1^{0,1} = G_0^0 + \sum_{k=1}^{3} \left| \sum_{j=0}^{2} r_{k-j}^{\dagger} \hat{y}_{1,k-j}^{1} \right|^2 = 4 + (-0.5-1.5-0.5-0.8)^2 +$$

$$+(-0.5-1.5-0.5-0.8+0.9-0.9)^2 + (-0.5-0.8+0.9-0.9+1+1)^2 = 26.27.$$

For $s = 1$, $\{n_i\} = \{2, 3\}$. $m = 2$ so we use only $n_i = 3$.

$$\hat{F}_1^{1,3} = \{0, 1, 3, 2, 0\},$$

$$\hat{H}_1^{1,3} = 12.67.$$

Step 5: *There are no decisions to be made, only updating: $F_1^0 = \hat{F}_1^{0,0}, F_1^1 = \hat{F}_1^{0,1}$, etc.*

Step 6,7:

$$\hat{C}_1^{0,0} = \{0, 0, 0\}, \quad \hat{G}_1^{0,0} = G_0^0 + (-0.5-1.5+0.5+0.8)^2 = 4.49,$$

$$\hat{C}_1^{0,1} = \{0,0,1\}, \quad \hat{G}_1^{0,1} = G_0^0 + (-0.5 - 1.5 - 0.5 - 0.8)^2 = 14.89,$$

$$\hat{C}_1^{1,2} = \{0,1,2\}, \quad \hat{G}_1^{1,2} = 9.29,$$

$$\hat{C}_1^{1,3} = \{0,1,3\}, \quad \hat{G}_1^{1,3} = 6.89.$$

Step 8: *There are no decisions, only updates:* $C_1^0 = \hat{C}_1^{0,0}, C_1^1 = \hat{C}_1^{0,1}$ *etc. The part that involves* $F_{t-L+2} = F_1$ *is not relevant since* F *is not defined at* $t = -1$*.*

At $\mathbf{t = 1}$:

Step 1,2: $s = 0 \rightarrow m = 0, s = 1 \rightarrow m = 2, s = 2 \rightarrow m = 0, s = 3 \rightarrow m = 2.$

$$\hat{F}_2^{0,0} = \{0,0,0,0,0,0\}, \quad \hat{H}_2^{0,0} = 50.63,$$

$$\hat{F}_2^{1,2} = \{0,0,1,2,0,0\}, \quad \hat{H}_2^{1,2} = 26.27,$$

$$\hat{F}_2^{2,0} = \{0,1,2,0,0,0\}, \quad \hat{H}_2^{2,0} = 30.95,$$

$$\hat{F}_2^{3,2} = \{0,1,3,2,0,0\}, \quad \hat{H}_2^{3,2} = 12.67.$$

Step 3,4:

$$\hat{F}_2^{0,1} = \{0,0,0,1,2,0\}, \quad \hat{H}_2^{0,1} = 29.75in,$$

$$\hat{F}_2^{1,3} = \{0,0,1,3,2,0\}, \quad \hat{H}_2^{1,3} = 31.47,$$

$$\hat{F}_2^{2,1} = \{0,1,2,1,2,0\}, \quad \hat{H}_2^{2,1} = 11.83,$$

$$\hat{F}_2^{3,3} = \{0,1,3,3,2,0\}, \quad \hat{H}_2^{3,3} = 13.87.$$

Step 5:

$$\hat{H}_2^{0,0} > \hat{H}_2^{2,0} \rightarrow F_2^0 = \{0,0,0,0,0,0\}, H_2^0 = 50.63,$$

$$\hat{H}_2^{0,1} > \hat{H}_2^{2,1} \rightarrow F_2^1 = \{0,0,0,1,2,0\}, H_2^1 = 29.75in,$$

$$\hat{H}_2^{1,2} > \hat{H}_2^{3,2} \rightarrow F_2^2 = \{0,0,1,2,0,0\}, H_2^2 = 26.27,$$

$$\hat{H}_2^{1,3} > \hat{H}_2^{3,3} \rightarrow F_2^3 = \{0,0,1,3,2,0\}, H_2^3 = 31.47.$$

Step 6,7:

$$\hat{C}_2^{0,0} = \{0,0,0,0\}, \quad \hat{G}_2^{0,0} = 10.74,$$

$$\hat{C}_2^{0,1} = \{0,0,0,1\}, \quad \hat{G}_2^{0,1} = 5.7,$$

$$\hat{C}_2^{1,2} = \{0,0,1,2\}, \quad \hat{G}_2^{1,2} = 25.78,$$

$$\hat{C}_2^{1,3} = \{0,0,1,3\}, \quad \hat{G}_2^{1,3} = 25.78,$$

$$\hat{C}_2^{2,0} = \{0,1,2,0\}, \quad \hat{G}_2^{2,0} = 26.1,$$

$$\hat{C}_2^{2,1} = \{0,1,2,1\}, \quad \hat{G}_2^{2,1} = 9.54,$$

$$\hat{C}_2^{3,2} = \{0,1,3,2\}, \quad \hat{G}_2^{3,2} = 9.78,$$

$$\hat{C}_2^{3,3} = \{0,1,3,3\}, \quad \hat{G}_2^{3,3} = 9.78.$$

Step 8: *For $w = 0$, we have two paths from the list F at time $t - L + 2 = 0$ that end with 0, i.e., $l = 2$. These paths are F_0^0 and F_0^1.*

$$\alpha_i = \{\hat{G}_2^{0,0}, \hat{G}_2^{2,0}, H_0^0, H_0^1\} = \{10.74, 26.1, 10.74, 26.1\}.$$

The maximal α_i is $\hat{G}_2^{2,0}$ so $G_2^0 = 26.1$ and $C_2^0 = \{0,1,2,0\}$. Note that at this point the decoder made an error. Let us look at the w.f. survivor at this point (state $= 0$, $t = 2$). This path is $F_2^0 = \{0,0,0,0,0\}$. We can see that the w.f. decisions were correct, as opposed to the n.f. decisions. Since the future estimate is correct, the w.f. decision will override all the wrong n.f. decisions, as we will see later. Continuing Step 8; for $w = 1,2$ and 3, we have no paths F_0^s that end with w, so we get:

$$C_2^1 = \{0,1,2,1\}, \quad G_2^1 = 9.54,$$

$$C_2^2 = \{0,0,1,2\}, \quad G_2^2 = 25.78,$$

$$C_2^3 = \{0,0,1,3\}, \quad G_2^3 = 25.78.$$

At $\mathbf{t = 2}$:

The paths computed at steps 1–5 are not needed for this example since we will stop at $t = 4$ to read the final results.

Step 6,7:

$$\hat{C}_3^{0,0} = \{0,1,2,0,0\}, \quad \hat{G}_3^{0,0} = 26.11,$$

$$\hat{C}_3^{0,1} = \{0,1,2,0,1\}, \quad \hat{G}_3^{0,1} = 42.91,$$

$$\hat{C}_3^{1,2} = \{0,1,2,1,2\}, \quad \hat{G}_3^{1,2} = 11.79,$$

$$\hat{C}_3^{1,3} = \{0,1,2,1,3\}, \quad \hat{G}_3^{1,3} = 11.79,$$

$$\hat{C}_3^{2,0} = \{0,0,1,2,0\}, \quad \hat{G}_3^{2,0} = 26.27,$$

$$\hat{C}_3^{2,1} = \{0,0,1,2,1\}, \quad \hat{G}_3^{2,1} = 36.67,$$

$$\hat{C}_3^{3,2} = \{0,0,1,3,2\}, \quad \hat{G}_3^{3,2} = 27.47,$$

$$\hat{C}_3^{3,3} = \{0,0,1,3,3\}, \quad \hat{G}_3^{3,3} = 27.47.$$

Step 8: *For $w = 0$ we have 4 paths ending with w: F_1^0, F_1^1, F_1^2 and F_1^3.*

$$\alpha_i = \{\hat{G}_3^{0,0}, \hat{G}_3^{2,0}, H_1^0, H_1^1, H_1^2, H_1^3\} = \{26.11, 26.27, 16.99, 26.27, 26.11, 12.67\}$$

The maximal is $\hat{G}_3^{2,0}$ or H_1^1 and without ambiguity, we get $C_3^0 = \{0,0,1,2,0\}$ and $G_3^0 = 26.27$. Here the n.f. decision is again an error. Continuing with Step 8, there is no path F_1^s that ends with $w > 0$ and we get

$$C_3^1 = \{0,1,2,0,1\}, \quad G_3^1 = 42.91,$$

$$C_3^2 = \{0,0,1,3,2\}, \quad G_3^2 = 27.47,$$

$$C_3^3 = \{0,0,1,3,3\}, \quad G_3^3 = 27.47.$$

At $\mathbf{t = 3}$:

Again, we will omit steps 1–5.

Step 6,7: *We are interested only in the decision at state 0.*

$$\hat{C}_4^{0,0} = \{0,0,1,2,0,0\}, \quad \hat{G}_4^{0,0} = 26.27,$$

$$\hat{C}_4^{2,0} = \{0,0,1,3,2,0\}, \quad \hat{G}_4^{2,0} = 31.47,$$

$$\alpha_i = \{\hat{G}_4^{0,0}, \hat{G}_4^{2,0}, H_2^0, H_2^1, H_2^2, H_2^3\} = \{26.27, 31.47, 50.63, 29.75in, 26.27, 31.47\}.$$

The maximal is H_2^0, so we have

$$C_4^0 = F_2^0 = \{0,0,0,0,0,0\}, \quad G_4^0 = H_2^0 = 50.63.$$

Eventually the correct path is decided on and overrides the wrong decisions of the past. Note that without this help from the w.f. candidate, the n.f. decision would have made another wrong decision by taking the maximal among the G's. This is a part of an **endless** *e.p. that occurs in this example if the future estimate is not used or is wrong and the input remains -1.*

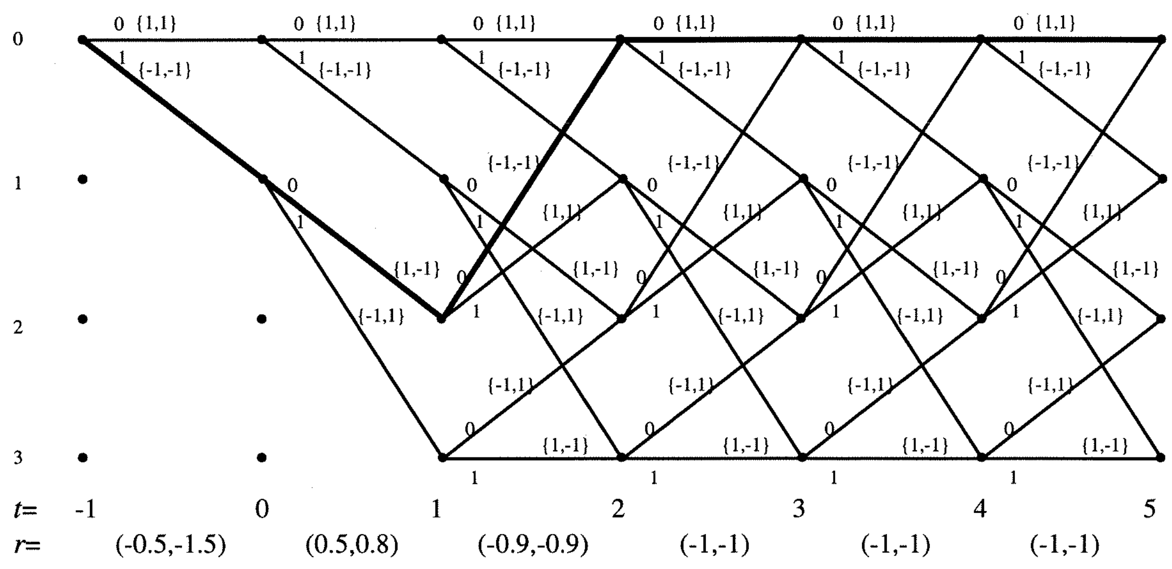


Figure 7.8: The trellis of the code used in Example 7.4.1 and the input signal.

7.6 Software Implementation

A direct implementation of the algorithm requires the maintenance of a list for each state and each time, so the memory use is infinite. It is clear, however, that only the last list C_t^s and the $L-1$ lists $F_{t-L+2}^s, \cdots, F_t^s$ need to be saved per state. For an efficient software implementation, the amount of required memory is minimized. The first thing to note is that instead of saving a list of states, we only need to save B bits per state. Also, we want to avoid copying whole lists, to save CPU time. In our implementation, we only maintain two lists per state and we avoid copying whole lists. We describe the implementation method next. The backward DFA which is implemented very similar to the VA (therefore will not be elaborated on) provides us with its path history $Q_t^s, \quad t = kA, \ldots, (k+1)A + W, \quad s = 0, \ldots N-1$. Since each list of time t is an extension of a list of time $t+1$, it is sufficient to save for each state and each time one B-bits pointer. This pointer is used to choose the trellis branch when tracking a survivor path. Hence, the whole path history has a tree structure. In the f.p. we maintain two tree-structured path history arrays, which are implemented as two two-dimensional cyclic buffers, *nf_History* and *wf_History.* The two are not independent: there are links between the two, as we will describe later. The first, *nf_History*, saves the n.f. decisions while *wf_History* saves the w.f. decisions. A path starting in *wf_History* will correspond to a list in F (except the future symbols in F) and a path starting in *nf_History* will correspond to a list in C. Each entry, in either arrays, contains one extra bit to connect to the other array. Thus, while tracking a typical path, we move back and forth between *nf_History* and *wf_History.* Two double-buffer, one-dimensional arrays are used to maintain G_t^s and H_t^s. For a flowchart of this implementation, refer to Figure 7.9 and 7.10. For the w.f. decision (Step 5), we need to form the candidate paths and their metric. These can be formed by steps 1 and 2 or by steps 3 and 4. The routine starts at a given state w and needs to find all $\hat{H}_{t+1}^{s_i,w}$. First, we find all s_i, the previous states to w on the trellis. For each s_i we decide whether to use steps 1 and 2 or steps 3 and 4 by checking if $w = q_{t,t+1}^{s_i}$. In steps 1 and 2 we use the path from *wf_History* and in steps 3 and 4 we use the path from *nf_History.* After making the decision which value of $\hat{H}_{t+1}^{s_i,w}$ is maximal, we update H_t^w and *wf_History.* If the candidate path formed from the *nf_History* path in Step 3 was decided on, then we set the appropriate entry in *wf_History* to point to that path in *nf_History.* Hence, we form a link between *wf_History* and *nf_History.* The new survivor path formed, F_{t+1}^w, will be used at time $t + L$ in the n.f. decisions. Thus, we have to save some information for future use. For this purpose, we maintain two additional data structures, cyclic buffers, each of size $L - 1$ per state. The cyclic buffers are used to implement a delay of $L - 1$ iterations. The first cyclic buffer *Projecting_path* saves the estimated future path used in the w.f. decisions (instead of saving the path we can also save w, and read the path from Q when needed) and the second cyclic buffer *Projected_H* saves H_{t+1}^w for future use. The estimated future path of state w leads to (projects on) state x. Since the above saved information is needed when making the n.f. decisions at state x, it helps to index the information in these arrays by x instead of w. We update *Projecting_path*$[x]$ and *Projected_H*$[x]$ only if $H_{t+1}^w >$ *Projected_H*$[x]$ (there is a value different from zero in *Projected_H*$[x]$ if the previous state already led to state x). Maximization of the saved value is needed since several states may lead to one state, but in Step 8 only the maximum is needed. Here, we did part of the work of Step 8, and also saved memory.

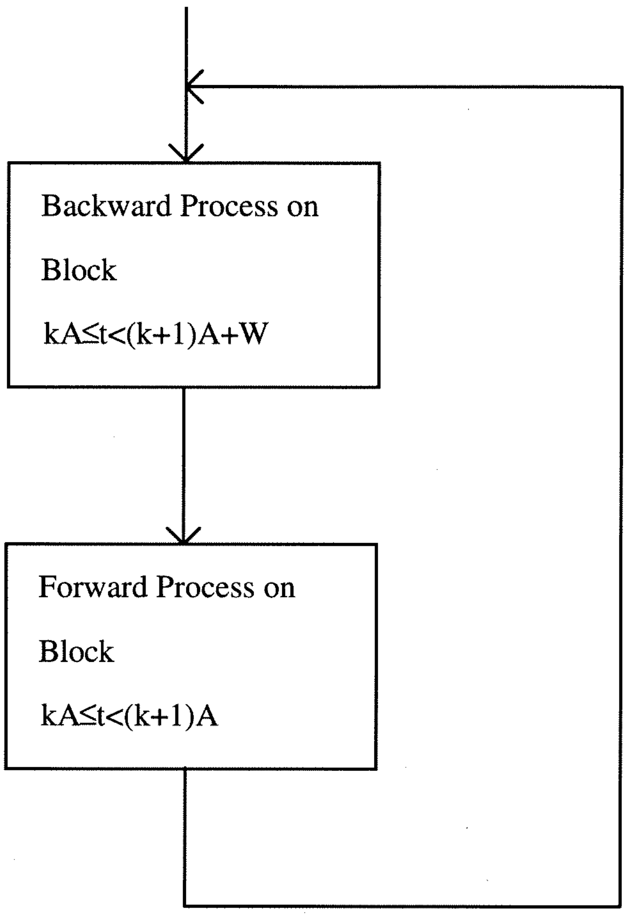
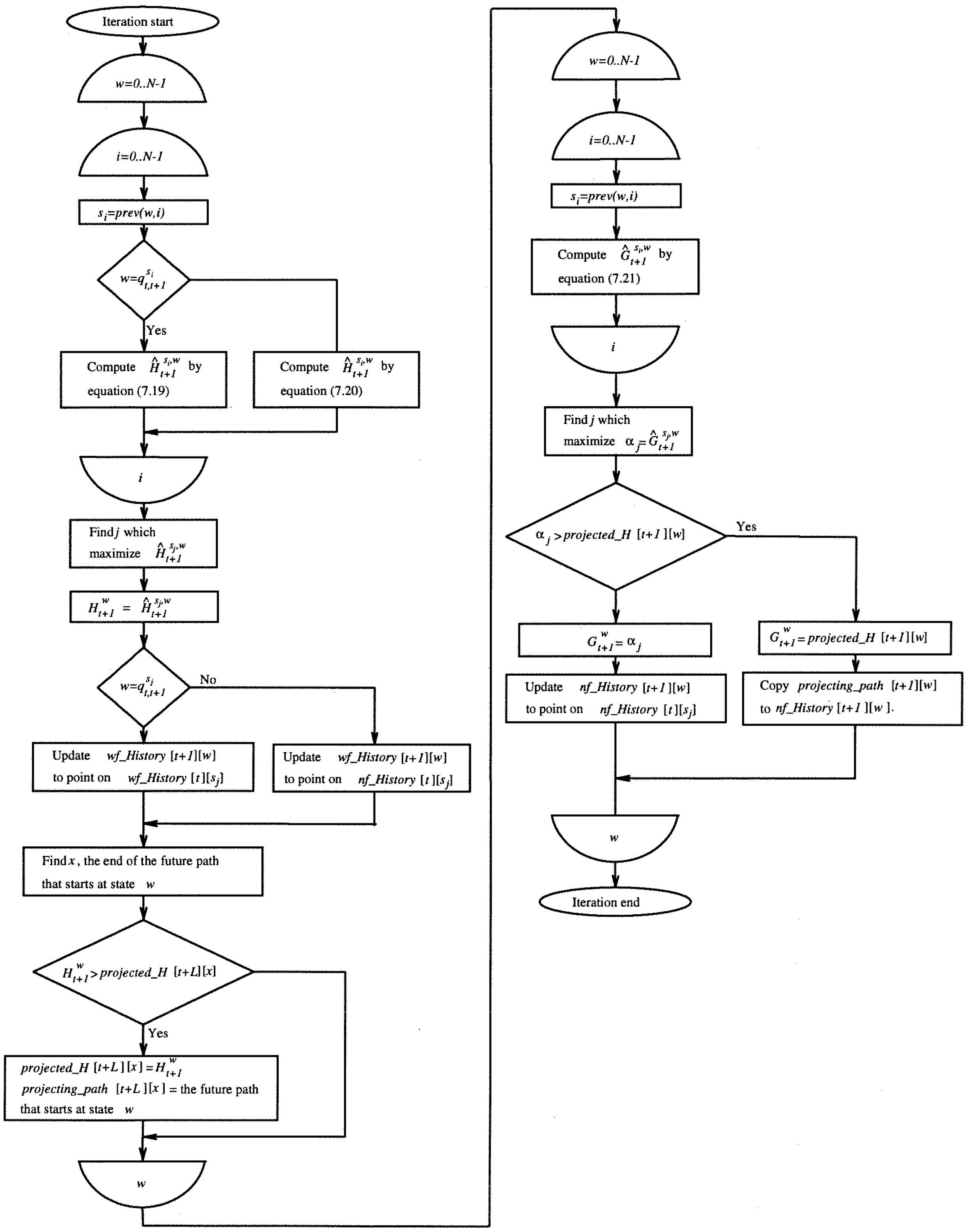

Backward Process on Block kA≤t<(k+1)A+W
Forward Process on Block kA≤t<(k+1)A

Figure 7.9: High level flow chart of the EFDFA algorithm.

Now, let us move on to describe the n.f. decisions. Here we have $R+1$ candidate paths. The first R are the paths formed from the previous states and their survived paths from *nf_History*, and the last is the path containing the saved estimated future path. For a state w, the value found in *Projected_H*$[w]$ (after the delay of $L-1$) is already the maximum element of $\{\alpha_i\}, \quad i=R,\cdots,R+L-1$ (equation (7.22)). If this value is also the maximum element of $\{\alpha_i\} \quad \forall i$, then we should update *nf_History* to point to the path $F_{t-L+2}^{u(i-R)}$. This path is composed of two parts. $F_{t-L+2}^{u(i-R)}$ is a concatenation of the survivor in *wf_History* at time $t-L+2$ and the saved estimated future path. Thus, the latter path is copied into *nf_History* (each entry in *nf_History* should be



Iteration start
w=0..N-1
i=0..N-1
s_i=prev(w,i)
$w=q^{s_i}_{t,t+1}$
Yes
Compute $\hat{H}^{s_i,w}_{t+1}$ by equation (7.19)
Compute $\hat{H}^{s_i,w}_{t+1}$ by equation (7.20)
i
Find j which maximize $\hat{H}^{s_j,w}_{t+1}$
$H^w_{t+1} = \hat{H}^{s_j,w}_{t+1}$
$w=q^{s_j}_{t,t+1}$
No
Update wf_History [t+1][w] to point on wf_History [t][s_j]
Update wf_History [t+1][w] to point on nf_History [t][s_j]
Find x, the end of the future path that starts at state w
H^w_{t+1} > projected_H [t+L][x]
Yes
projected_H [t+L][x] = H^w_{t+1}
projecting_path [t+L][x] = the future path that starts at state w
w
w=0..N-1
i=0..N-1
s_i=prev(w,i)
Compute $\hat{G}^{s_i,w}_{t+1}$ by equation (7.21)
i
Find j which maximize $\alpha_j = \hat{G}^{s_j,w}_{t+1}$
α_j > projected_H [t+1][w]
Yes
$G^w_{t+1} = \alpha_j$
G^w_{t+1} = projected_H [t+1][w]
Update nf_History [t+1][w] to point on nf_History [t][s_j]
Copy projecting_path [t+1][w] to nf_History [t+1][w].
w
Iteration end

Figure 7.10: Simplified flowchart of one iteration of the EFDFA forward process.

long enough) and a pointer is set to point to *wf_History*. This creates the second link between *nf_History* and *wf_History*.

Now we will explain how to extract the decoded bits. Like in the VA, we start from the maximal metric node and backtrack. After backtracking a length M (the decoder truncation length), we assume convergence to the optimal path and start reading data. While tracking the survivor path, we move back and forth between *nf_history* and *wf_History* since the survivor path is the combination of the formers.

We start with the n.f. survivor. We continue backtracking in *nf_history* until the extra bit that is used to link the two buffers is set. Then, we backtrack through the $L-1$ states of the estimated future path saved in *nf_History* and then switch to *wf_History*. We continue backtracking in *wf_History* as long as the extra bit used to link the two buffers is off. If it is on that means that we have used n.f. information in this path, so we now backtrack in *nf_history*.

7.7 Performance Analysis

First, let us introduce several variables. Let p_e and N_e denote the probability per symbol and the average number of bits in error respectively of an error event of the IO-NMLSE. Let p_f denote the first event error probability of the forward DFA (or the n.f. decisions) and p_b, the first event error probability of the backward DFA. p'_f is the conditional probability of a forward error event given that it is correctable, i.e., this error can be corrected by using a correct future estimate. Let γ_b denote the conditional probability that e.p. occurs in the b.p. given the first error occurred. γ'_f denotes the e.p. conditional probability given a correctable forward error event occurred. Let N'_f be the average number of bit errors in a correctable forward first error event (first meaning that it is not a part of e.p.).

We will use the following assumptions. For the worst-case, an incorrect future path will prohibit correction of n.f. errors (any n.f. error which is the result of not making the optimal decision is corrected by the algorithm when correct future estimate is used, but also can sometimes be corrected by using an incorrect future estimate). e.p. continues until a correct future is encountered, then stops within a few symbols. For the derivation it is assumed that the e.p. stops immediately when the correct future is encountered. We assume that $p_b, p'_f, p_e << \frac{1}{A}$, so there is a maximum of one error per block. $A >> L, K$ so that edge effects can be ignored. Finally, we assume that if a convergence region W is used for the b.p., it converges before reaching the end of the block A.

Worst-case Model for the Error Propagation

We begin with the simplest, but worst-case, model for the e.p. It assumes endless e.p. until the end of the block. The resulting bound for the error event probability per symbol, P_{seq}, is

$$P_{seq} = \frac{P_A}{A} \leq p_e + (L-1)(1-\gamma_b)p'_f p_b + \frac{1}{A}\sum_{i=0}^{A-1} p'_f p_b \gamma_b (A-i) \underset{A>>L}{\cong} p_e + \frac{A}{2} p'_f p_b \gamma_b, \quad (7.24)$$

where P_A is the block error probability. The first term stands for the errors in the w.f. decisions even with correct estimated future. We equate this error probability with the error probability of the natural errors, which were to be produced by the optimal algorithm given the received signal. It is interesting to note that p_e is not equal to the error probability of the w.f. decisions but it is a tight upper-bound. The probability of the w.f. decisions given correct future is *lower* than the probability of the optimal algorithm (the other terms will compensate so the total will not lower than the optimal). The explanation goes as follows. Suppose that a natural error occurs, i.e., the ML path is not the transmitted path. By our assumption, the estimated future paths are correct i.e. there is no error in the b.p. This applies only to the states on the transmitted path. For the states of the ML path which do not coincide with the transmitted path, the estimated future generated by the b.p. will not necessarily agree with the ML path. Thus, the ML path may not be decided on and not included as a candidate for decision on the transmitted path. Hence the transmitted path can win, and error will not occur. In one of our simulations where we forced the estimated future to be correct on the transmitted path, the bit error probability was reduced slightly from $3.84 \cdot 10^{-5}$ in the optimal decoder to $3.6 \cdot 10^{-5}$.

The second term is the contribution of the uncorrected forward errors due to backward errors that occur in the near future ($< L$ symbols ahead) and does not cause e.p. The third term reflects the uncorrected forward errors which are not corrected due to e.p. in the b.p., one which originated in the region $i < t < A$ of the block. For the bit error probability, we get the following expression:

$$P_{bit} \leq \frac{p_e N_e}{B} + \frac{1}{AB}\sum_{i=0}^{A-1} p_b\gamma_b(A-i)p'_f[N'_f + \gamma'_f\frac{(A-i)B}{2}] \underset{A>>1}{\cong} \frac{p_e N_e}{B} +$$

$$+ p_b\gamma_b p'_f[\frac{N'_f}{B}\frac{A}{2} + \gamma'_f\frac{A^2}{12}]. \qquad (7.25)$$

$\frac{A-i}{2}$ is the average number of symbols in the forward e.p. This e.p. starts at time i and ends at the first backward error. Its location is uniformly distributed leading to the reduction of the e.p. length by half. For close to optimal operation, the second term should be much smaller than the first term. We can immediately see that the block size, A, should be as small as possible. On the other hand, if framed transmission is used, then we have to keep $A >> L + K$ to minimize degradation due to throughput reduction. If a convergence region is used, then $A >> W$ is needed to validate the convergence assumption. Otherwise, backward errors originating in that region will dominate. Thus, an optimum value for A exists.

The Refined Model

By using a probabilistic model for the e.p. length, we can get results which are more close to the actual performance. As mentioned previously, we will model the e.p. length by a geometric distribution with mean $\frac{1}{\lambda}$ (specifically λ_f for forward and λ_b for backward). With this model, the probability that the e.p. length (excluding the first error) is equal to k, is $\lambda(1-\lambda)^{k-1}$. The probability that at time i we are in a backward error is

$$\Pr(i \text{ is in error}) = p_b(1-\gamma_b) + p_b\gamma_b\sum_{j=i+1}^{A-1}(1-\lambda_b)^{j-i-1} =$$

$$= p_b(1-\gamma_b) + \frac{p_b\gamma_b}{\lambda_b}[1-(1-\lambda_b)^{A-i-1}]. \qquad (7.26)$$

The probability, h_i, that one or more of the backward decisions at times $i < t < \min(i+L-2, A-1)$ are in error is

$$h_i = \left\{ \begin{array}{ll} (L-1)p_b + \frac{p_b\gamma_b}{\lambda_b}[1-(1-\lambda_b)^{A-i-L+1}], & \text{if } i \leq A-L \\ (A-i)p_b, & \text{otherwise} \end{array} \right\}. \tag{7.27}$$

Since $A >> L$, we can approximate h_i by (despite the approximation, we leave the -1 term for future simplifications)

$$h_i = (L-1)p_b + \frac{p_b\gamma_b}{\lambda_b}[1-(1-\lambda_b)^{A-i-1}]. \tag{7.28}$$

The probability that the complete algorithm will make an error event (per symbol) is

$$P_{seq} = \frac{P_A}{A} < p_e + \frac{1}{A}\sum_{i=0}^{A-1} p'_f h_i =$$

$$= p_e + (L-1)p'_f p_b + \frac{p'_f p_b \gamma_b}{A\lambda_b^2}[(1-\lambda_b)^A + A\lambda_b - 1]. \tag{7.29}$$

The forward e.p., once started, continues until it either stops naturally or encounters a correct future. Then the bit error probability is

$$P_{bit} < \frac{p_e N_e}{B} + \frac{p'_f}{BA}\sum_{i=0}^{A-1}\left\{h_i N'_f + \gamma'_f p_b \frac{LB}{4}(L-1)(1-\gamma_b) + \right.$$

$$\left. + \gamma'_f \gamma_b p_b \sum_{j=i+1}^{A-1}\left[\sum_{k=1}^{j-i}\lambda_f(1-\lambda_f)^{k-1}\frac{kB}{2} + (j-i)(1-\lambda_f)^{j-i-1}\frac{B}{2}\right](1-\lambda_b)^{j-i-1}\right\}. \tag{7.30}$$

Let us explain this formula term by term. The first term represents the real errors, i.e., the uncorrectable ones. The second term (the sum over i) is the contribution of all the uncorrected forward errors. The second term consists of the following components. The first component reflects the fact that each uncorrected error event will produce at least N'_f bit errors. The next component evaluates the contribution of all backward errors which do not produce e.p. and are in the range $i \ldots i+L-1$. For those errors, we get an average length of forward e.p. (assuming $1/\lambda_f >> L$) of $L/2$ which translates to $LB/4$ bit errors on average. The next component (the summation over j) represents the forward e.p. caused by the backward e.p. The forward e.p. begins at i, and the backward e.p., at j. The first term inside the brackets is the naturally stopping forward e.p., the second is the case where the e.p. continues until time j, where it stops due to correct future. Evaluation of (7.30) yields

$$P_{bit} < \frac{p_e N_e}{B} + (L-1)p_b p'_f[\frac{N'_f}{B} + \frac{L}{4}\gamma'_f(1-\gamma_b)] + \frac{p'_f p_b \gamma_b(\frac{N'_f}{B} + \frac{\gamma'_f}{2\lambda_f})}{A\lambda_b^2}[(1-\lambda_b)^A + A\lambda_b - 1] +$$
$$+ \frac{p'_f\gamma'_f\gamma_b p_b}{2A\lambda_f}\Bigg[\frac{[\lambda_b - 1)\lambda_f^3 + (3-2\lambda_b)\lambda_f^2 + (2\lambda_b - 1)\lambda_f - \lambda_b](1-Q^A)}{(Q-1)^3} +$$
$$+ \frac{A[(Q^A+2)\lambda_f^2 - (1-\lambda_f)^2\lambda_b - \lambda_f]}{(Q-1)^2}\Bigg], \tag{7.31}$$

where

$$Q = (1-\lambda_b)(1-\lambda_f). \tag{7.32}$$

We can investigate two possible limiting cases. The first is the case of $A >> \frac{1}{\lambda_f}, \frac{1}{\lambda_b}$ which means that the block length is much larger than the e.p. average length. The second limit will be the opposite case where the block length is much shorter than the average e.p. length. This is equivalent to the worst-case e.p. model. For the first case $A >> \frac{1}{\lambda_b}, \frac{1}{\lambda_f}$.

Taking the limit $A \to \infty$ in (7.29) gives

$$P_{seq} \le p_e + (L-1)p'_f p_b + \frac{p'_f p_b \gamma_b}{\lambda_b}. \tag{7.33}$$

The result can be explained as follows. The probability that a particular decision will be in backward e.p. is $\frac{p_b\gamma_b}{\lambda_b}$. This is the result one obtains for any physical problem of memoryless arrivals of events with memoryless distributed time duration (for example, the model used for the availability of telephone lines). Taking $A \to \infty$ in (7.31) we get

$$P_{bit} \le \frac{p_e N_e}{B} + (L-1)p_b p'_f\left[\frac{N'_f}{B} + \frac{L}{4}\gamma'_f(1-\gamma_b)\right] + \frac{1}{\lambda_b}p'_f p_b \gamma_b \frac{N'_f}{B} +$$
$$+ \frac{p'_f\gamma'_f\gamma_b p_b(\lambda_b+\lambda_f)}{\lambda_b(\lambda_b\lambda_f - \lambda_b - \lambda_f)^2} \underset{\lambda_f,\lambda_b<<1}{\cong} \frac{p_e N_e}{B} + \frac{p'_f\gamma'_f p_b\gamma_b}{\lambda_b(\lambda_b+\lambda_f)}. \tag{7.34}$$

Here the probability of forward e.p., $\frac{p'_f\gamma'_f\gamma_b p_b}{\lambda_b}$, is multiplied by the average length of the e.p. which is $\frac{1}{\lambda_b+\lambda_f}$, the average minimum lifetime of two processes of rate λ_b and λ_f. The second limiting case is $A << \frac{1}{\lambda_b}, \frac{1}{\lambda_f}$. In this case the result should be the same as the worst-case e.p. case. We take the limit as $\lambda_f, \lambda_b \to 0$ and get

$$P_{seq} \le p_e + \frac{(A-1)}{2}p'_f p_b \gamma_b \underset{A>>1}{\cong} p_e + \frac{A}{2}p'_f p_b \gamma_b, \tag{7.35}$$

and

$$P_{bit} \leq \frac{p_e N_e}{B} + (L-1)p_b p'_f \Big[\frac{N'_f}{B} + \frac{L}{4}\gamma'_f(1-\gamma_b)\Big] +$$
$$+ p'_f p_b \gamma_b \Big[\frac{A-1}{2}\frac{N'_f}{B} + \gamma'_f(\frac{A^2-1}{12})\Big] \underset{A>>L}{\cong} \frac{p_e N_e}{B} + p_b \gamma_b p'_f \Big[\frac{A}{2}\frac{N'_f}{B} + \frac{A^2}{12}\gamma'_f\Big]. \quad (7.36)$$

Example 7.7.1 *Let us investigate the performance of the algorithm for the code of rate $\frac{1}{2}$, 4 states code with BPSK modulation with $L = 3$ and $\frac{E_b}{N_0} = 5\,dB$. By the BDFA simulation program, we have measured the following parameter values (The code is symmetric so all backward and forward parameters are the same):*

$$p_e = 0.00017, \quad p_b = p_f = 0.002, \quad \frac{1}{\lambda_f} = \frac{1}{\lambda_b} = 19, \quad \gamma_f = \gamma_b = 0.845, \quad N_e = 1.72.$$

Since $p_f >> p_e$ the majority of the forward errors are correctable and we can assume $p'_f = p_f$, $\gamma'_f = \gamma_f$ and $N'_f = N_f$. In addition, we can assume $N_f = N_e$. Measuring these parameters exactly is difficult and approximation is sufficient. By simulating the above parameters and choosing $A = 50$ and the framed input option, we get

$$P_{seq} = 0.00022, \quad P_{bit} = 0.00056.$$

Actual values measured by simulation are

$$P_{seq} = 0.00021, \quad P_{bit} = 0.00049.$$

7.8 Simulation Results

The algorithms presented were evaluated by simulation. The channel used was AWGN with a slowly varying phase (10° change over one observation of length L symbols generated by a frequency offset). The codes used were the quaternary ($R = 4$) Linear Noncoherent Trellis Coded Modulation (LNTCM) family. These codes are built over the modulo 4 group and use QPSK symbols. Both have rate $\frac{1}{2}$. The first code has 16 states and generators (in base 4) 133, 231. The second has 64 states and generators 2123, 1312. The codes are maximum d_{free} codes with d_{free} of 7 and 10 respectively. Their performance was also shown in Figure 6.4 and 6.5. For additional application examples, see Chapter 8 where all the simulations points were produced by using the

EFDFA. For proper operation of the MDFA, it is necessary to have enough SNR per observation. For $\frac{E_b}{N_0} = 2$ dB, we need $L \geq 5$ and for $\frac{E_b}{N_0} = 3$ dB, we need $L \geq 4$. $L = 4$ was used in the 16 states code simulation and $L = 5$ for the 64 states code. We chose $W = 100$ for the EFDFA to achieve a good convergence. It was convenient to choose also $A = 100$.

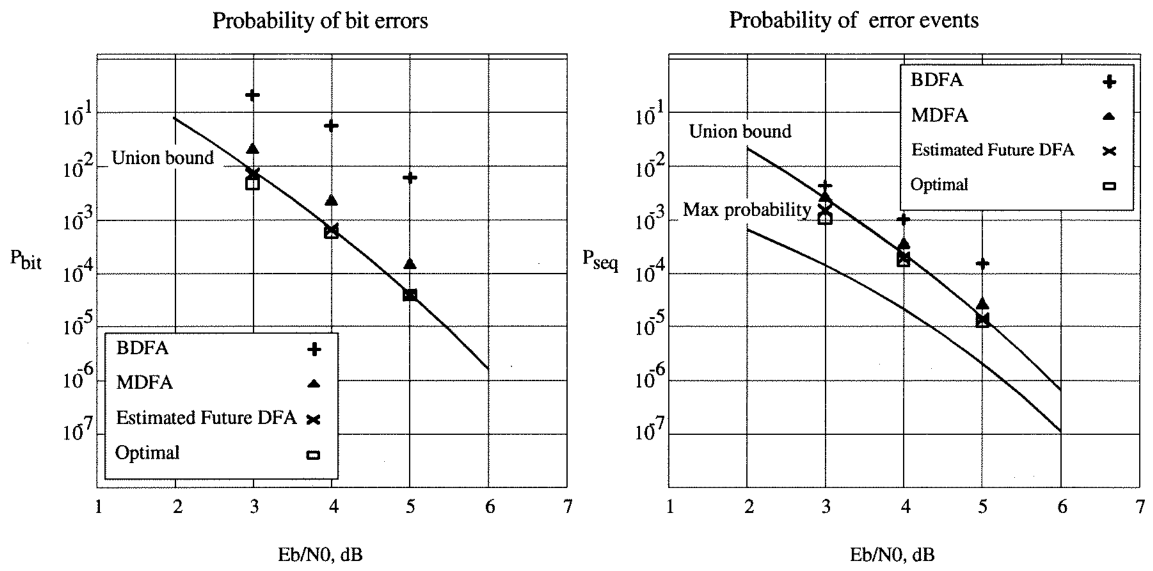
For the 16 states code, 1024 states are required for the optimal algorithm but only 16 states are required in the suboptimal algorithms. For the 64 states example, 16384 states are required, so the optimal algorithm could not been simulated. Since the suboptimal algorithms require only 64 states, their complexity is two orders of magnitude lower in this example. Compared to the BDFA, the CPU time requirements of the EFDFA is $4-5$ times larger.

As shown in Figure 7.11 in the bit error performance curves, due to mainly e.p. effects, the BDFA lost 2 dB compared to the optimal. The MDFA worked well, but degradation of up to 0.5 dB in one case and 0.8 dB in the other remains. The EFDFA algorithm essentially achieved the optimal performance in moderate and high SNR ($P_b < 10^{-3}$). In low SNR ($P_b \cong 10^{-2}$), we see 0.2–0.3 dB degradation. The reason for the degradation in low SNR is that the ratio $\frac{p_b p_f}{p_e}$ becomes higher.

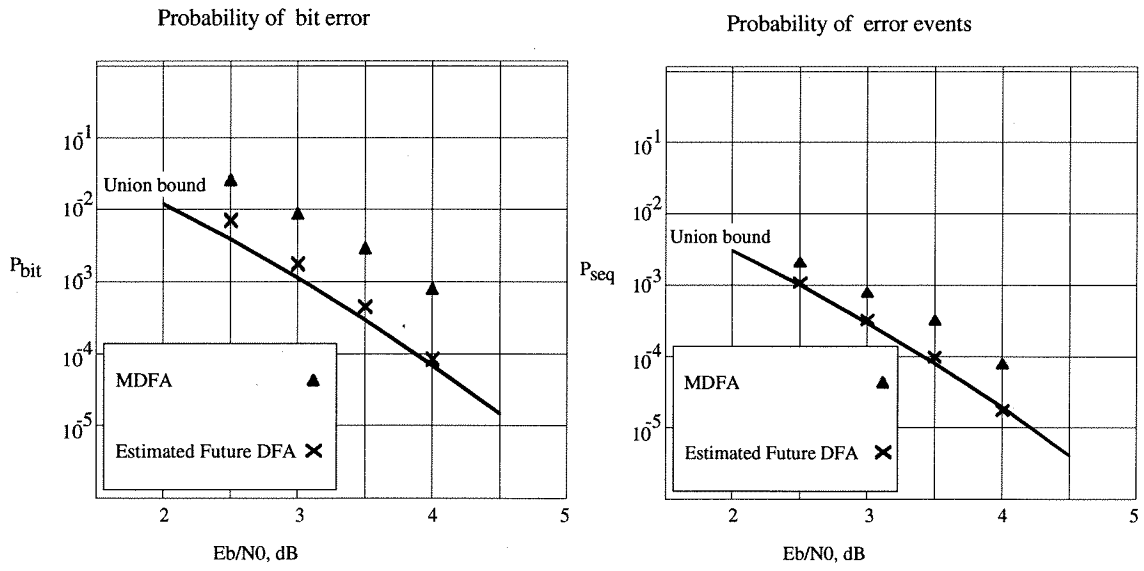
Examining the error event performance curves, we see that all the algorithms suffer less degradation compared to the bit error rate case. The reason is that all the algorithms suffer from increased length error bursts when they fail. We also note that the modification in the MDFA reduces the error events probability of the BDFA.

7.9 Conclusion

Several algorithms for implementing the decoder for NCM were introduced. Their performance was explained and evaluated analytically and via computer simulations. The BDFA, although simple to implement, is not recommended for actual use, since with only a slight increase in complexity we can implement the MDFA which can reduce the degradation to less than 0.5 dB (depending on the code). The EFDFA algorithm essentially achieved the optimal performance in moderate and high SNR ($P_b < 10^{-3}$). Its complexity for a practical case is two order of magnitude lower than that of the



a.



b.

Figure 7.11: Performance results of the several algorithms by simulation for codes with (a) 16 and (b) 64 states. The same codes were evaluated in Figure 6.4 and Figure 6.5. The union bound was derived analytically in Chapter 6 and is a tight upper bound on the performance of the optimal implementation of the IO-NMLSE. The maximum among all the pairwise error probabilities serves as a lower bound on the probability of error events.

optimal implementation. Since the IO-NMLSE performance is very close to that of the coherent MLSE, the algorithm shows a practical noncoherent decoding which performs very close to the optimal coherent one.

Bibliography

[1] Forney G. D., "The Viterbi Algorithm," Proc. IEEE, Vol. 61, pp. 268–278, 1973.

[2] Divsalar D., Simon M.K., Shahshahani M., "The Performance of Trellis-Coded MDPSK with Multiple Symbol Detection," IEEE Trans. Comm., Vol. 38, No. 9, pp. 1391–1403, Sep. 1990.

[3] Viterbi A.J., Omura J.K., *Principles of Digital Communication and Coding.* McGraw-Hill, 1979.

[4] Proakis J.G., *Digital Communications.* NY: McGraw Hill, 1989.

Chapter 8

Applying the New Noncoherent Decoder to Continuous Phase Modulation

8.1 Introduction

Continuous Phase Modulation (CPM), is a class of constant envelope modulation schemes. CPM can have power and bandwidth efficiency, so it is an attractive scheme to be used whenever a nonlinear channel is employed (requires constant amplitude signal). With the addition of an encoder in front of the modulator, even better combinations of power and bandwidth efficiency are possible. However, CPM in general requires complex receivers, and carrier tracking is difficult. Due to the difficulty in achieving coherency in CPM systems, many noncoherent receivers were suggested in the literature (see Section 8.7). Here, we propose a new method for combined noncoherent detection and decoding of CPM which can be also used for coded CPM.

The separate problems of data detection and carrier phase tracking have been studied extensively. Uncoded CPM or coded CPM over the Additive White Gaussian Noise (AWGN) channel with coherent detection can be optimally decoded by the Viterbi Algorithm (VA) [1]. The carrier phase tracking can be performed by a PLL. There are two ways to implement the phase tracking for CPM. One is to remove the data by a nonlinear operation, and then lock to one or more of the discrete spectral components produced [2]. The other is to decode the data and feed it back into the loop (data aided loop) [3]. If discrete spectral components are to be generated from a CPM scheme with

R phase states, the signal has to be raised to the Rth power. Since this process results in a large SNR drop, very narrow loops must be employed. For the data aided loop, the large decoding delay caused by the VA will only allow the operation of a PLL with a large time constant (narrow-band). The use of extremely narrow band PLL is required, but such a PLL cannot be used for the cases where fast phase variations in the channel occur due to phase noise or Doppler effects. Also such PLL may cause other problems like slow acquisition and delayed recognition of loss of lock.

In the past reports on the noncoherent detection of CPM signals, independent block decisions were mainly used. In those schemes, like in [28], very large observations and a large complexity receiver must be used in order to approach the coherent performance. Other types of noncoherent decoders have also been proposed for CPM, but have much lower performance than the method examined here. Here we successfully apply the IO-NMLSE which was introduced in Chapter 6 to the decoding of CPM. Since CPM has a trellis structure, many of the concepts in Chapters 6 and 7 can be applied to CPM. Remarkably good results were achieved. For example, MSK (Minimum Shift Keying) can be decoded noncoherently with 3 symbol overlapped observations to within 0.5 dB of the coherent case. We are one of the first to apply noncoherent decoding to coded CPM. CPM is clearly not NC (noncoherently catastrophic). Although many sequences which are a constant phase shift of another exist, all of these are the encoding of the same input symbols.

8.2 CPM Schemes

In CPM schemes, the envelope of the RF signal is constant and its phase varies in a continuous manner. CPM signals are described by

$$s_n(t) = \exp\left\{j2\pi \sum_{i=0}^{n} h a_i q(t - iT)\right\} \quad \text{for} \quad nT < t < (n+1)T. \tag{8.1}$$

The data $\{a_n\}$ are M-ary data symbols, M is a power of 2, taken from the alphabet $\pm 1, \pm 3 \cdots, \pm(M-1)$, h is a modulation index and $q(t)$ is the phase response function. CPM schemes are denoted by their phase response function or by its derivative $g(t)$, the frequency response function. Let G be the number of symbol intervals over which the frequency response function is not zero. If $G = 1$, the scheme is called full response, and when $G > 1$ it is called partial response. The full response CPM with rectangular frequency pulse is called CPFSK. Binary CPFSK with $h = 1/2$ is called MSK (Minimum Shift Keying). In schemes denoted by multi-h, the modulation index h is replaced by a variable modulation index h_i.

CPM with rational h can be represented as a trellis code [1]. In order to use the representation of Chapter 2, we can choose an orthonormal basis such that D, the number of dimensions, is finite. In practice we can use the Gram-Schmidt orthogonalization procedure for finding the basis functions. There are M^G possible waveforms generated for the interval of one symbol. Hence, the maximum number of basis functions required is M^G.

8.3 A Trellis Structure for Use in Noncoherent CPM

A CPM scheme with a rational modulation index h can be represented by a trellis diagram. In a full response scheme, the states are the phase states, and in a partial response scheme, a state corresponds to the phase state and $G-1$ last symbols [1]. We can apply the methods described in Chapter 7 directly using this trellis. However, specifically for CPM, we can make some simplifications to reduce the number of states and to improve the understanding of the factors influencing the performance of the noncoherent system.

We can notice that for the noncoherent metric, the phase states are not required as they contain information which is not going to be used. Hence, for the noncoherent detection, a different trellis diagram than the one used in the coherent case should be adopted. In this trellis diagram, the phase states disappear, since noncoherent detection is used. Let us now define a trellis diagram for CPM in which the phase states are eliminated, but otherwise all the waveforms can be generated without any change. We will call this trellis the primitive noncoherent trellis. The relation between the primitive trellis and the optimal trellis is analogous to the relation between the original code trellis and the augmented one in Chapter 7. The primitive trellis is the optimal one for the case of $L = 1$, i.e., symbol by symbol noncoherent detection.

Each state in the primitive noncoherent trellis corresponds to the last $G-1$ input symbols $\{a_{n-1}, a_{n-2}, \cdots, a_{n-G+1}\}$. The $G-1$ symbols correspond to the memory required by the partial response scheme. Note that for a full response scheme, the trellis has only one state and M parallel transitions. Let $s_n(t)$ be the output waveform for the duration of the symbol a_n,

$$s_n(t) = \exp\left\{j2\pi h \sum_{i=-\infty}^{n} a_i q(t - iT_s)\right\} =$$

$$= \exp\left\{j2\pi h \sum_{i=0}^{G-1} a_{n-i} q(t - iT_s) + jh\pi \sum_{i=-\infty}^{n-G} a_i\right\},$$

$$nT_s \le t \le (n+1)T_s, \tag{8.2}$$

and let $s'_n(t)$ be the same waveform as $s_n(t)$ but with a zero phase state,

$$s'_n(t) = \exp\left\{j2\pi h \sum_{i=0}^{G-1} a_{n-i} q(t - iT_s)\right\}. \tag{8.3}$$

Each branch of the trellis is assigned a record with two information fields. The first field contains $s'_n(t)$, and the second field contains the amount of phase rotation induced by the input symbol a_{n-G+1},

$$\theta_n = \pi h a_{n-G+1}. \tag{8.4}$$

Given a sequence $\{s'_n(t), \theta_n\}$, starting at time n_0 (which may be the beginning of an observation), the signal $s_n(t)$ can be derived, up to a constant phase shift ϕ as

$$s_n(t) = s'_n(t) \exp\left\{j \sum_{i=n_0}^{n-1} \theta_i + j\phi\right\}. \tag{8.5}$$

The optimal noncoherent trellis must allow for the additional memory introduced by the overlapping as was explained in Chapter 7. We have to construct a new trellis diagram, such that using this trellis, η_k is a function of the branch value only. Each state in the optimal noncoherent trellis corresponds to the last $L-1+G-1$ input symbols $\{a_{n-1}, a_{n-2}, \cdots, a_{n-L-G+2}\}$. The $L-1$ symbols correspond to the amount of overlapping between the observations which can be considered as amount of memory, and $G-1$ symbols correspond to the memory required by the partial response. The assignments of the branches, as in the primitive trellis, follow equations (8.3) and (8.4). If a shift register is used as an encoder, then the most recent $G-1$ symbols together with the current input are used to choose one of the possible M^G waveforms (or complex vectors representing these waveforms). The rest of the shift register stages are left unconnected. These stages are only used for the purpose of delaying the merge of two sequences on the trellis by $L-1$ symbols. During this "waiting period" the outputs of the two sequences are equal. In this way we eliminate the influence of the unknown future after the merge (see Chapter 7 for more elaboration on this method). See Figure 8.1a-d for all the various trellis diagrams and encoders mentioned in this section. The CPM scheme used in this example is of $h = \frac{1}{5}$ binary CPFSK.

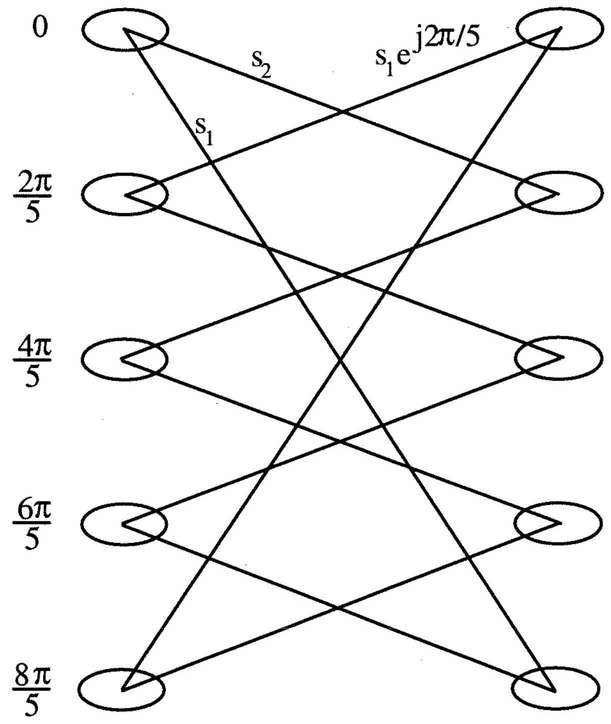


Figure 8.1-a: The coherent trellis of $h = 2/5$ binary CPFSK, $s_1(t) = e^{-j\frac{2\pi}{5}\frac{t}{T_s}}$, $s_2(t) = e^{j\frac{2\pi}{5}\frac{t}{T_s}}$.

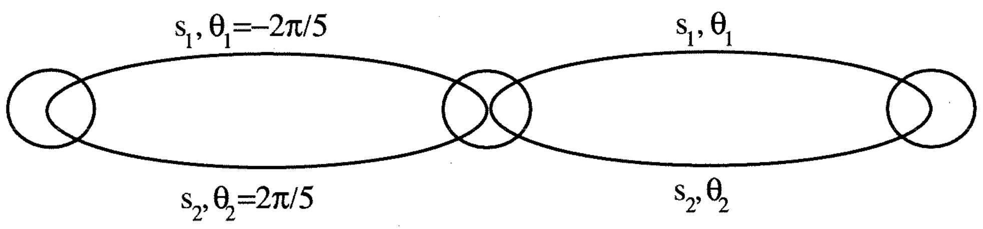
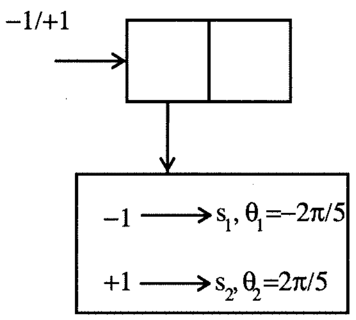


Figure 8.1-b: The primitive noncoherent trellis of $h = 2/5$ binary CPFSK.



$-1/+1$
$-1 \longrightarrow s_1, \theta_1 = -2\pi/5$
$+1 \longrightarrow s_2, \theta_2 = 2\pi/5$

Figure 8.1-c: The optimal noncoherent encoder of $h = 2/5$ binary CPFSK, $L = 2$.

8.4 Computing the Error Probability of the IO-NMLSE for CPM Signals

CPM is not UEP, i.e., the error probability is not independent of the transmitted sequence. There is a way to overcome this problem, by using a trellis diagram that represents the phase difference between two CPM signals. The error probability of a constant energy coded modulation can be completely specified by the pairwise complex correlations of all possible waveforms (which it is equivalent to the phase difference here). This was proved in Chapter 3 for any general noncoherent detector.

Since there is a linear mapping from input symbols to output phases, the phase difference signal can be generated simply by feeding the difference of the input symbols through the CPM transmitter. Also, we can form a different trellis diagram which is similar to the previously defined trellis, but where the input symbols a_n are replaced by difference input symbols Δa_n. Each symbol a_n takes the values $\pm 1, \pm 3, \cdots, \pm(M-1)$,

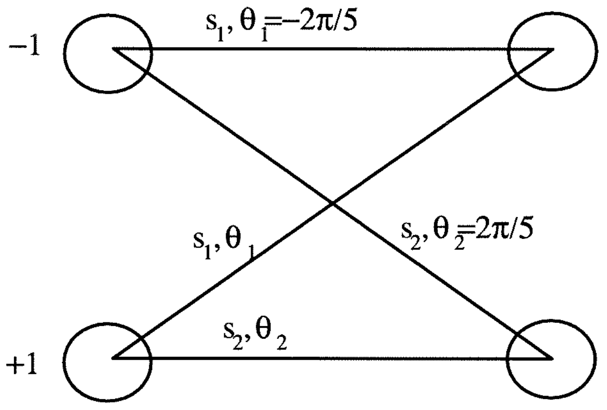
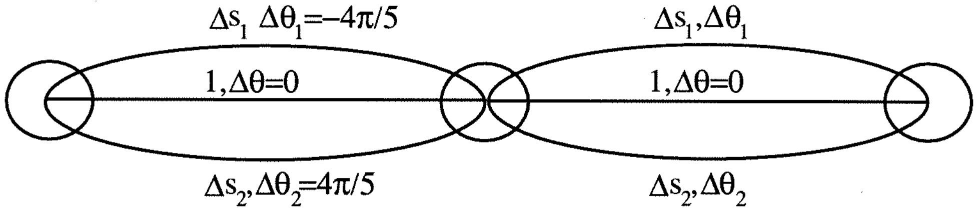


Figure 8.1-d: The optimal noncoherent trellis of $h = 2/5$ binary CPFSK, $L = 2$.

and Δa_n takes the values $0, \pm 2, \pm 4, \cdots, \pm 2(M-1)$, i.e., $2M-1$ values. Refer to Figure 8.1e-f for an example.



$\Delta s_1\ \Delta\theta_1 = -4\pi/5$
$1, \Delta\theta = 0$
$\Delta s_2, \Delta\theta_2 = 4\pi/5$
$\Delta s_1, \Delta\theta_1$
$1, \Delta\theta = 0$
$\Delta s_2, \Delta\theta_2$

Figure 8.1-e: The symbol-difference primitive noncoherent trellis of $h = 2/5$ binary CPFSK, $\Delta s_1(t) = e^{-j\frac{4\pi}{5}\frac{t}{T_s}}$, $\Delta s_2(t) = e^{j\frac{4\pi}{5}\frac{t}{T_s}}$.

Having this symbols-difference trellis, we can use the analysis developed in Chapter 6. Since D for many cases is large, the large D method for the computation of the pairwise error probability is more useful for CPM. Due to the fact that every difference sequence may represent more than one pair of input sequences, there is a minor change in equation (6.7) due to the fact that every difference sequence may represent more than one pair of input sequences. The corrected formula is

$$P_b \cong \frac{1}{B} \sum_{|\bar{\mathbf{x}}^{(m)}| < \mathcal{M}} b(\bar{\mathbf{x}}^{(m)}) P_e\{\bar{\mathbf{x}}^{(m)}\} \prod_{n=0}^{N-1} \frac{M - |\Delta a_n|/2}{M}, \tag{8.6}$$

where $B = \log_2 M$, the number of bits per symbol.

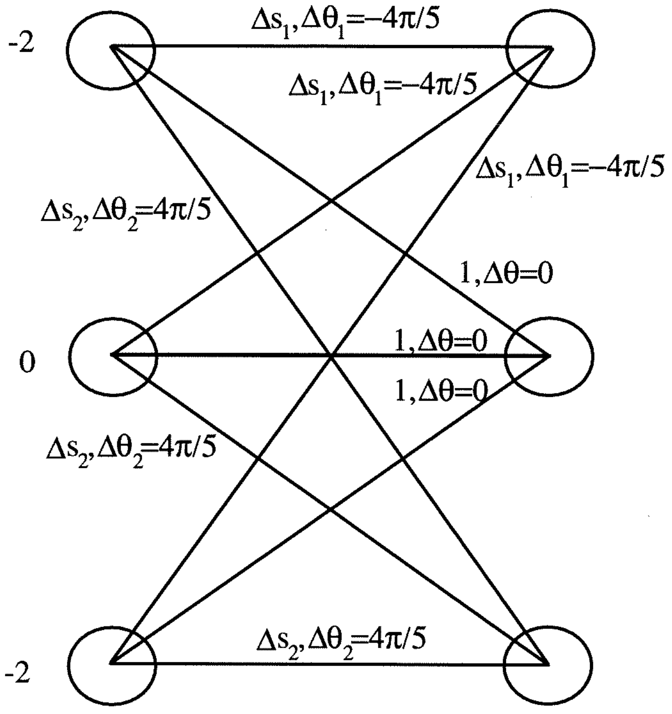


Figure 8.1-f: The symbol-difference optimal noncoherent trellis of $h = 2/5$ binary CPFSK, $L = 2$.

8.5 Performance Predictions

The degradation of the noncoherent methods versus the coherent ones comes from two sources. One is the increase in the error probability for error events which also occur in the coherent case and the other source is due to the inclusion of additional error events, which cannot occur in coherent reception. The additional error events are those which start in the transmitted sequence and diverge to a sequence with a constant phase shift from the transmitted one. In a noncoherent CPM trellis diagram, these error events can be distinguished by having a total phase rotation, $\sum \theta_n$, different from that of the transmitted sequence. In a coherent CPM trellis, this will correspond to reaching a different phase state, so that there would not be a merge, and thus no error event. The contribution of the additional error events is lower as the observation length, L, grows. In particular, when L is larger than the error event length, there are some observations that span the region of before and after the error event. These observations will have a particularly large distance and thus low error probability. Error events with larger difference in the total phase rotation tend to have lower pairwise error probability for the same event length.

For this reason CPM schemes which use h near a "weak" point will suffer a larger degradation (only if this weak point causes poor performance). A weak modulation index, [1] means that a merge can happen prior to the first inevitable merge at $t = (G+1)T_0$. Suppose that a weak modulation index h_w, causes a reduction in the minimum distance. Let **a** and **b** be two sequences of input symbols which correspond to two paths on the coherent trellis which start at state 0 at time $t = 0$ and merges at time $t = t_0 \leq GT$, for $h = h_w$. Now, if we use for the same CPM scheme h close to h_w, then at time t_0 the two sequences **a** and **b** will reach two different phase states ϕ_a and ϕ_b. It is clear that for $h \sim h_w$, $\phi_a - \phi_b \cong 0$. On the primitive noncoherent trellis, **a** and **b** correspond to two paths that merge at t_0 or prior to that. At t_0 the total phase rotations of **a** and **b** are equal to ϕ_a and ϕ_b respectively. Thus, the difference in the total phase transition for this error event is small, possibly having a large probability to occur in the noncoherent decoder.

8.6 Coded CPM

It is possible to encode the input data prior to feeding it to the CPM modulator and achieve a significant coding gain. Some good convolutional encoders for CPM (coherent detection) were found using a systematic search [4]-[9]. All the suggested codes are based on a binary field. The output of the encoder is mapped by a specific mapping rule to one of the M modulator inputs. Rimoldi [9] attempted to find a more "natural" combination of code and modulator. The naturality of his scheme is not clear; maybe he wanted to achieve UEP. Indeed, he assumed UEP in his analysis. Unfortunately, this assumption was invalid.

Evaluating the performance of binary encoded CPM is more difficult than uncoded CPM since it is not possible to use the symbols-difference trellis and assume the all-0 sequence is transmitted. The encoder is implemented over GF(2) and not over the real field as the input symbols of the CPM modulator. It may be the subject of future research to find an equivalent method to the symbols-difference trellis for coded CPM. Evaluating the error probability for coded CPM will require to compute the expression

$$P_b \cong \frac{1}{B}\frac{1}{|\lambda|}\sum_{m\in\lambda}\sum_{|\bar{\mathbf{x}}^{(n)}|<\mathcal{M},m\neq n} b(\bar{\mathbf{x}}^{(m)},\bar{\mathbf{x}}^{(n)})\Pr\{\eta(\bar{\mathbf{x}}^{(m)})<\eta(\bar{\mathbf{x}}^{(n)})\}, \tag{8.7}$$

where λ is the set of all the sequences differing in their M output symbols. It is possible to simplify the form of this expression to one summation, but then a pair-state trellis diagram is used (the amount of computations is not reduced). This computation was too extensive for us to evaluate even for short constraint length codes, so we preferred to evaluate coded CPM only by simulation.

We observed previously for the coded PSK case, that different codes, which have the same coherent performance, suffer different amounts of degradation in noncoherent decoding. Since we have not performed a systematic search for the best code for noncoherent CPM, we have to rely on codes good for the coherent decoding and hope that their degradation in noncoherent decoding is small even for a short observation. Of course we can always get close to the coherent performance if large observation is used, and for this case the codes best for coherent decoding will be the best for noncoherent. Specifically, the code which we evaluate here suffer low degradation, hence probably it is close or equal to the best noncoherent code. The encoder for this code is shown in Figure 8.2, and its equivalent for simulation is shown in Figure 8.3, and the equivalent encoder for noncoherent decoding, $L = 3$ (for example), is shown in Figure 8.4.

We must verify that the coded CPM is not NC (Noncoherently Catastrophic). This is true since the encoder only generates a subset of the uncoded CPM outputs, and uncoded CPM is not NC. Hence, unless the encoder itself is catastrophic, the coded CPM will not be NC.

8.7 Previously Proposed Schemes for Noncoherent Detection of CPM

We can divide the noncoherent detection schemes into four families. The first family is of symbol-by-symbol noncoherent detection, which was usually applied to binary

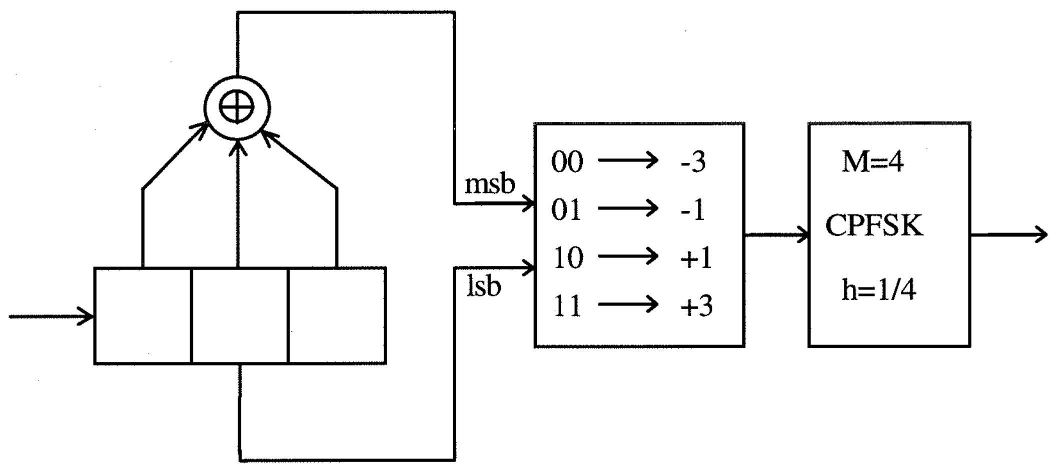


Figure 8.2: A coded CPFSK encoder, coherent detection.

CPFSK [10]. The second is a family of schemes based on differential detection. Applications to CPM signals on AWGN can be found in [1], [11]–[20]. The third family is of schemes based on Limiter-Discriminator (LD), [1], [13],[15], [17], [21], [22]. The fourth is multiple-symbol MLSE based approaches [1], [24], [25], [26], [27], [28], and [29], to which the IO-NMLSE introduced here also belongs.

The symbol-by-symbol noncoherent detection can be implemented simply by using two bandpass filters each centered at one frequency of the modulator and followed by an energy detector. A comparison is made between the output of the energy detector to determine which frequency was sent. This detection, although simple, does not use the phase continuity of the transmitted signal, resulting in poor performance. The differential and LD detectors, in their basic forms, can reach degradation as low as 3 dB relative to coherent, and this, only for certain binary CPM schemes. For other, more powerful schemes, the degradation is even higher. Many methods were used to improve the performance including zero forcing equalizer, decision feedback, error correction and VA. A good comparison is found in [27] for the case of MSK. The best of these schemes provides 2 dB degradation for MSK with LD and decision feedback. [20] have reached 1.5 dB degradation for GMSK (Gaussian MSK [30]), $BT = 0.25$ by combining 3 differential detectors for delays of 1,2 and 3 symbols and using the VA with similar

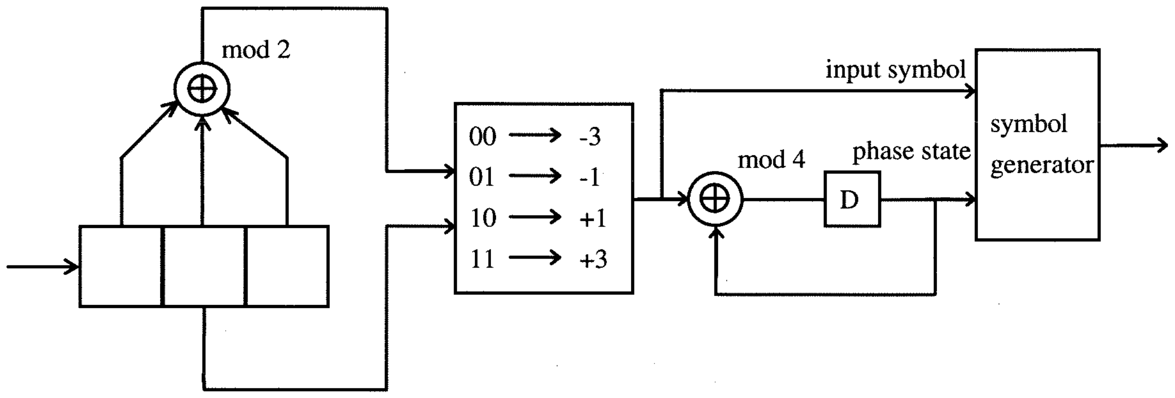


Figure 8.3: Equivalent encoder of the system of Figure 8.2, coherent detection.

complexity to the optimal decoding of the IO-NMLSE. Note that the lower degradation mentioned in [20] is due to comparison to sub-optimal coherent detection of [30] instead of to the optimal. The observation length is equivalent then to 4 symbols. This is about the same degradation expected in our scheme. Note that GMSK, $BT = 0.25$ (see Figure 8.9) in particular, suffers an increased degradation relative to other CPM schemes when using the IO-NMLSE. It will be interesting to know the performance of Makrakis et al. schemes for other CPM schemes, including multilevel schemes. Note that their scheme requires the addition of differential decoder in front of GMSK. This alone causes unrecoverable degradation of 0.5 dB even if the number of differential detectors is increased to infinity.

In the fourth family, most of the schemes solve a noncoherent MLSE on an individual block with complexity (except when using feedback) exponential in L. Svensson [25] makes an MLSE decision on a block of symbols, but decides on one symbol in the middle of the observation. Simon and Divsalar [28], made an MLSE decision on all L symbols. Leib and Pauspathy [27], and Abrardo et al. [29] for GMSK have recognized that the other symbols contribute more to the error probability and they ignore them, getting improved results. In [27], the degradation of noncoherent detection of MSK is 1.4 dB even for $N = 8$. [29] has shown degradation of less than 0.5 dB for $L = 7$ in GMSK, $BT = 0.5$. This compares to $L = 4$ in our scheme (see Figure 8.8). In [1],

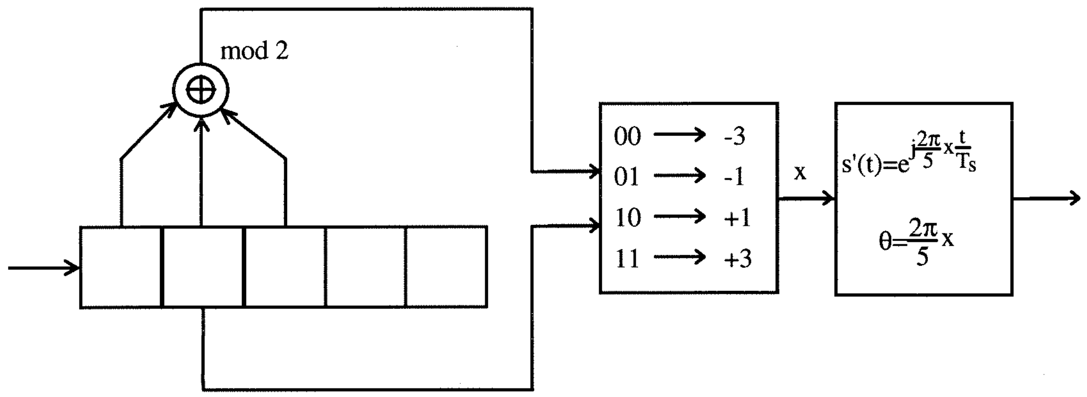


Figure 8.4: Equivalent encoder of the system of Figure 8.2, noncoherent detection with $L = 3$.

$L = 10$, for some h values even more is required to approach coherent in other partial response schemes. Harrold and Kingsbury [26] have suggested another scheme, based on the VA algorithm where a phase estimation is made from the best survivor and used as a reference to evaluating a partially coherent metric on an observation. In their scheme, $L = 12$–25 is required to be close within 0.5 dB to coherent with various partial response, but the complexity is not exponential in L. There is no data to compare to for noncoherent decoding of coded CPM.

Our scheme outperforms all of the above schemes. $L = 3$ only is required for MSK, and $L = 5$–6 for good 4 level schemes including partial response. For GMSK, $BT = 0.25$, we show less improvement, but we offer our reduced complexity algorithms to save in complexity. Our method can also be applied to coded CPM as described, which is a major benefit. Poor results are expected when applying differential detection or LD to coded CPM since the working point is at low symbol SNR. The previously proposed MLSE methods cannot be applied to coded CPM, unless the observations do not overlap. In this case worse performance is expected as was the case in the uncoded case and as was demonstrated for coded BPSK.

8.8 The Application of the Decoding Algorithms to CPM

The algorithms described in Chapter 7 are applicable to CPM or coded CPM. Some points are worth noting. To apply the noncoherent trellis, we should first regenerate $S_n(t)$ out of $\{S'_n(t)\}$ and $\{\theta_n\}$, given on the trellis, for correlation with the input signal. In principle, we can use the primitive trellis to apply the algorithms. However, the performance results are not so good for CPFSK schemes when using only a one state trellis. For CPFSK, we have found that the performance is significantly improved when using an M states trellis, the one optimal for $L = 2$. This trellis might also be used for $L > 2$ in the BDFA, MDFA or EFDFA, and the results are good considering the low complexity. For the partial response schemes (2RC and 3RC were tested), it is sufficient to use the primitive trellis to get good results. This means that for noncoherent decoding, the complexity for the 2RC scheme is equal to that of CPFSK ! We can note that for the case of CPM, in some cases the complexity of the noncoherent decoder is lower than that of the coherent one (due to the removal of the phase states). For example, for binary 2RC scheme, with $h = \frac{2}{5}$, we use 2 states in noncoherent decoding compared to 10 states for coherent. However, for the noncoherent, more processing is required per branch. On the other hand, we save the acquisition, tracking and phase ambiguity resolving circuitry.

When applying the EFDFA, we note the following. In Figure 8.1c we see the equivalent encoder for CPFSK, in which one unconnected stage was added to the shift register to improve the DFA performance (its optimal for $L = 2$, here $L > 2$). Note what happens when we apply the backward process. The encoder is reversed in time as shown in Figure 8.5. Here, the extra stage does not provide any memory and is useless. Thus, we get the performance of the DFA applied on the primitive trellis of CPFSK, obtaining poor error performance and large error propagation bursts. To solve the problem we have to use extra unconnected stages both before and after the primitive encoder, as shown in Figure 8.6. Note that for the b.p. there is almost no complexity increase in doing so, despite the addition of more states. In the b.p. we can operate on

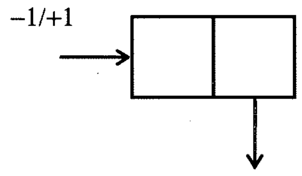


Figure 8.5: The reversed encoder.

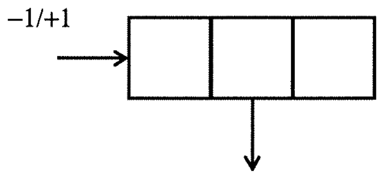


Figure 8.6: The corrected encoder for using in EFDFA.

2 states and copy the results to the other 2 states since they are completely equivalent. In the f.p., due to the dependency on the b.p. decisions, we have to process the full 4 states. For all the CPM schemes we have tested, the error propagation was minor and the backward convergence was extremely rapid (when a $L = 2$ trellis is used for CPFSK and primitive trellis is used for partial response). The e.p. is relatively low also for coded CPM. Thus, the EFDFA has negligible degradation when applied to CPM. Some simulation results using the EFDFA were added to Figure 8.7 (full response, 2-level, $L = 5$) and Figure 8.10 (partial response, 4-level, $L = 10$).

The low e.p. seems to be related to the low power efficiency per complexity meaning that there is more memory in the encoder than is required to achieve the same power efficiency as compared to PSK. This memory serves the same purpose as the memory introduced by the extra unconnected stages in the encoder. The distance between two candidates merging at a given state in their last few symbols is smaller than in the high performance coded PSK. Using this qualitative argument, we can predict that coded CPM will have larger e.p. than uncoded CPM, but lower than high performance coded PSK. This is indeed the case.

8.9 Results and Discussion

Here we present the performance analysis and simulation of the overlapped observations noncoherent decoding of CPM schemes. In the following examples, maximal overlapping, i.e., $l = 1$, is used. The first example is MSK modulation; see Figure 8.7. Using the analysis described in Section 8.4, we computed the bit error rate performance of noncoherent detection of MSK. Simulation points for $L = 5$ were added to confirm the analysis. The difference between the simulation points and the analysis is due to the use of the union bound technique in the analysis. In Figure 8.8 and 8.9, we show the performance of GMSK. With $BT = \infty$, GMSK becomes MSK. As BT becomes lower, the side-lobes become smaller, and also, to a lesser extent, the spectrum become narrower. We recognize by comparing Figure 8.7, 8.8 and 8.9 that as the spectral properties improve, the degradation in the noncoherent decoding is higher for the same observation length. The reason, as explained intuitively in Section 6.2, is that the high frequency components are important to the noncoherent decoder.

In Figure 8.10 we show the performance of a partial response scheme. Here we use 4-level 2RC with $h = 1/3$. This scheme is chosen to have the same power efficiency as MSK but have much better bandwidth efficiency (almost twice the throughput for 99% power bandwidth). In Figure 8.11 we use a 4-level CPFSK with $h = 2/5$. This scheme is chosen such that the bandwidth requirements are close to that of MSK, but the power efficiency is much better.

Comparing Figure 8.11 and 8.12, we see that as the modulation index h get close to a weak point $h = 0.5$, the degradation of the noncoherent decoder increases as expected. The shorter observations, $L = 2$, $L = 3$, as expected, do not "feel" the effect of being near weak h point (see Section 8.5). Note that although the degradation is higher, the performance of the $h = 4/9$ scheme is better than the $h = 2/5$ scheme. Since, unlike the coherent case, the number of phase states does not affect the decoder complexity, $h = 4/9$ should be chosen among the two.

Several other examples are shown in Figures 8.13–8.16. A coded CPM scheme is evaluated by simulations in Figure 8.17. The coherent points were simulated as well.

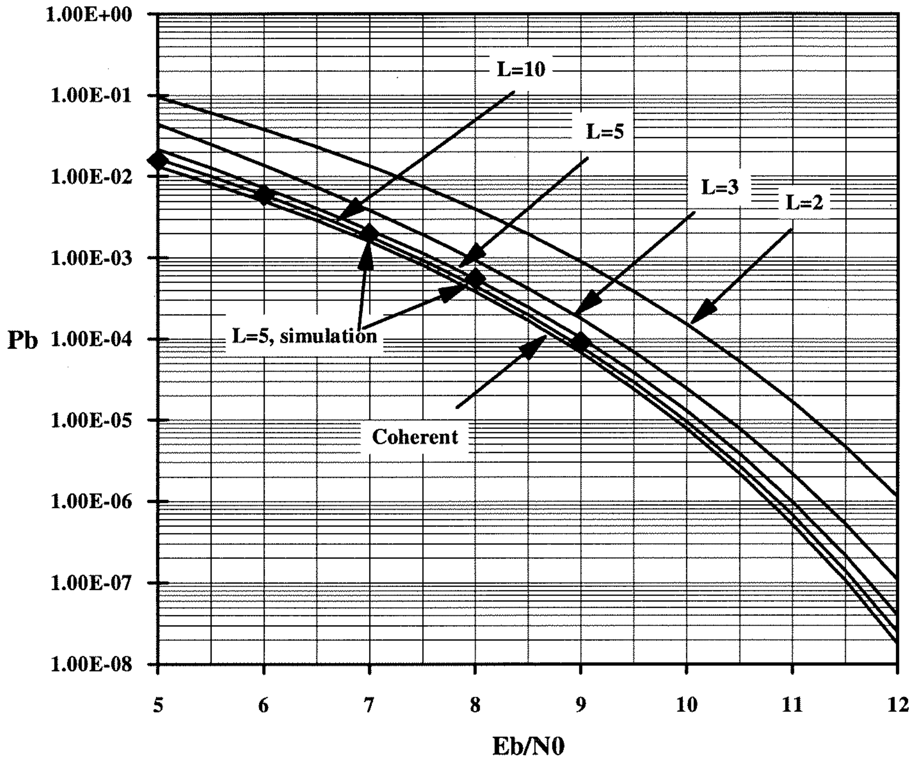


Figure 8.7: Noncoherent decoding performance of MSK.

They match with the results in [4] which were computed using union bound techniques. We see degradation of about 0.4 dB when $L = 10$ is used. Compared to the coded PSK schemes, considering the bandwidth efficiency which tend to increase to observation length needed, this is a good result.

The simulation points in Figure 8.7, 8.10 and 8.17 were produced using the Estimated Future Decision Feedback Algorithm. For this algorithm to decode MSK, a 4 states trellis diagram is used instead of 16 state which are necessary for the optimal algorithm with $L = 5$. For $L = 10$ we should use 512 states for optimal implementation (still compared to 2 states in the EFDFA) by the VA. For the simulation points shown in Figure 8.10, 4 states were used in the EFDFA, compared to $4^{10} = 1048576$ states required for the optimal algorithm! For the coded CPM, 8 states were used in the EFDFA. Four states belong to the code, and one extra unconnected stage was added to

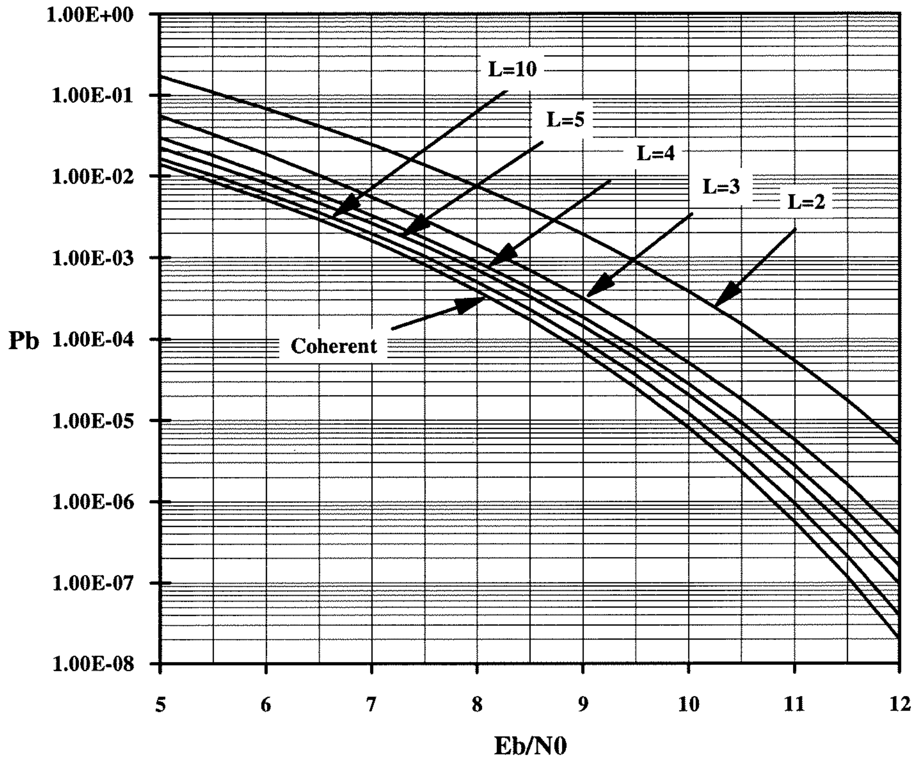


Figure 8.8: Noncoherent decoding performance of GMSK, $BT = 0.5$.

improve the convergence of the b.p. We note that this algorithm is very well suited for noncoherent decoding of CPM schemes. With this algorithm we can use observation lengths of 10 and above and get extremely close to the optimal coherent performance with a relatively low complexity receiver.

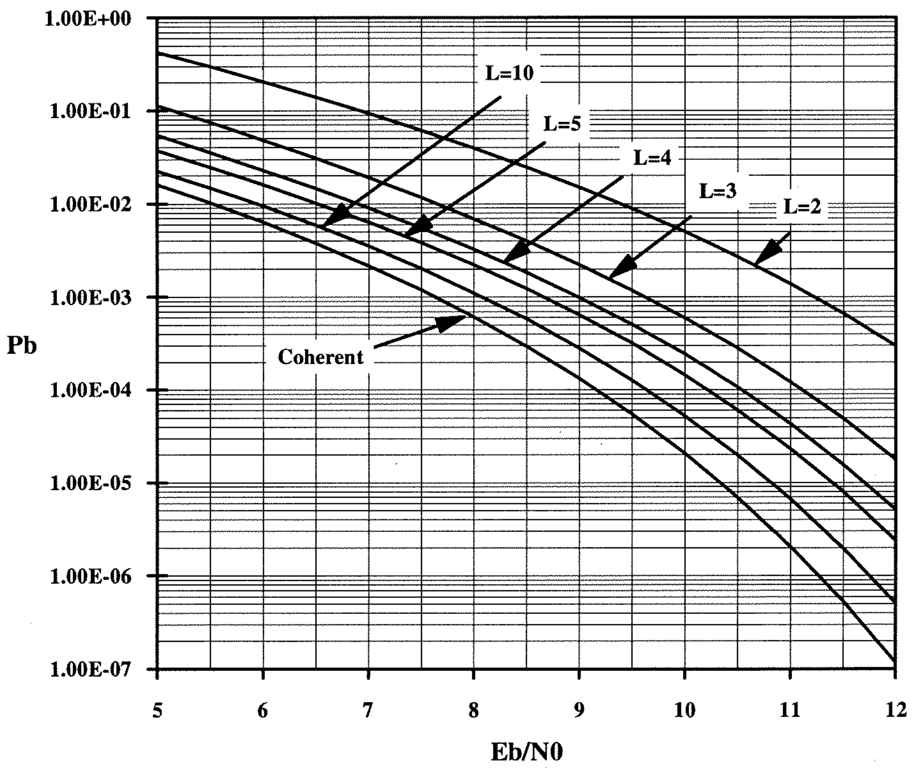


Figure 8.9: Noncoherent decoding performance of GMSK, $BT = 0.25$.

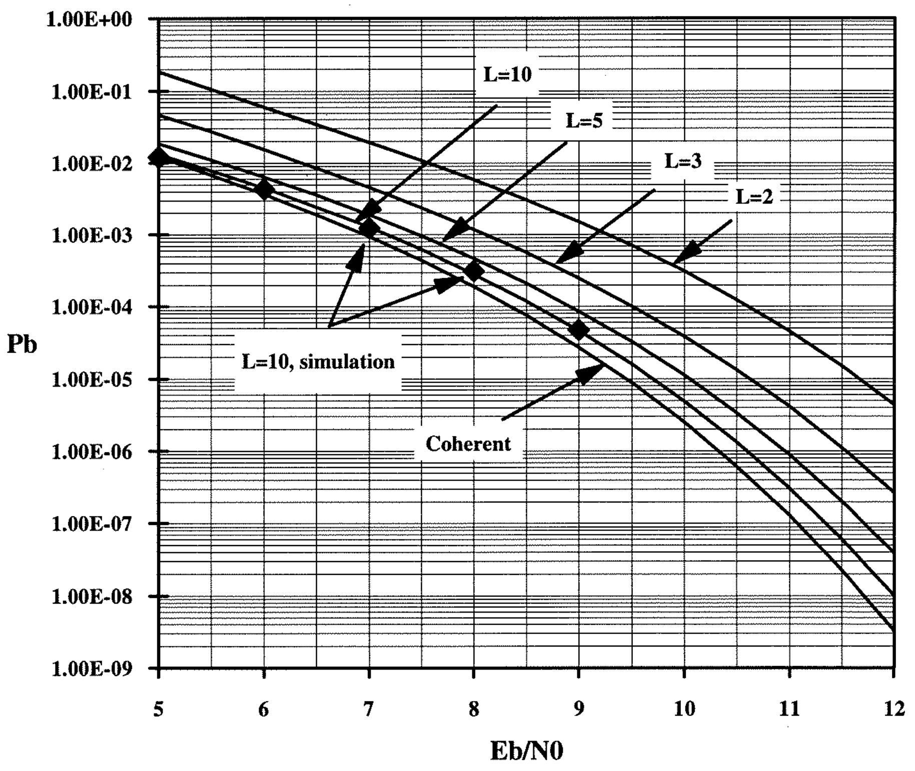


Figure 8.10: Noncoherent decoding performance of 2RC, $h = 1/3$, 4-level CPM.

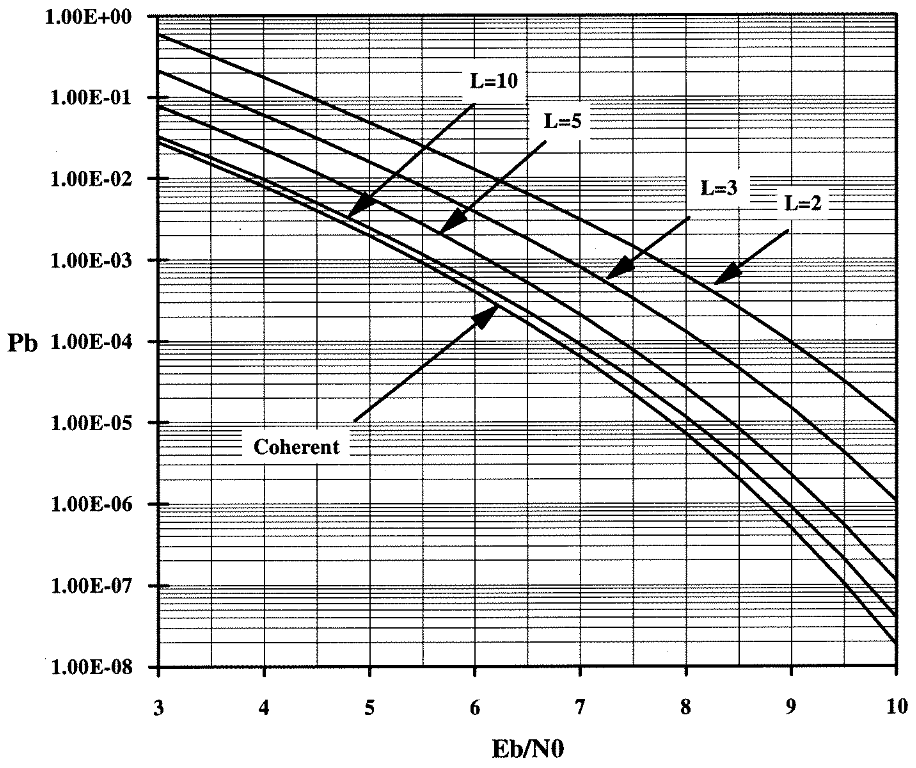


Figure 8.11: Noncoherent decoding performance of 1REC (CPFSK), $h = 2/5$, 4-level CPM.

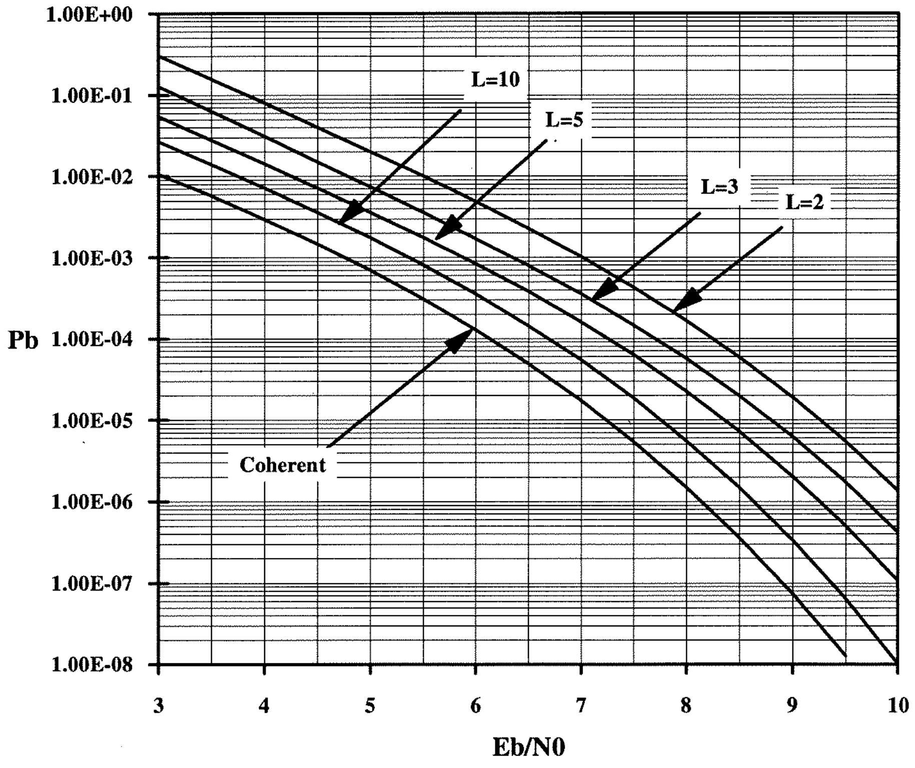


Figure 8.12: Noncoherent decoding performance of 1REC (CPFSK), $h = 4/9$, 4-level CPM.

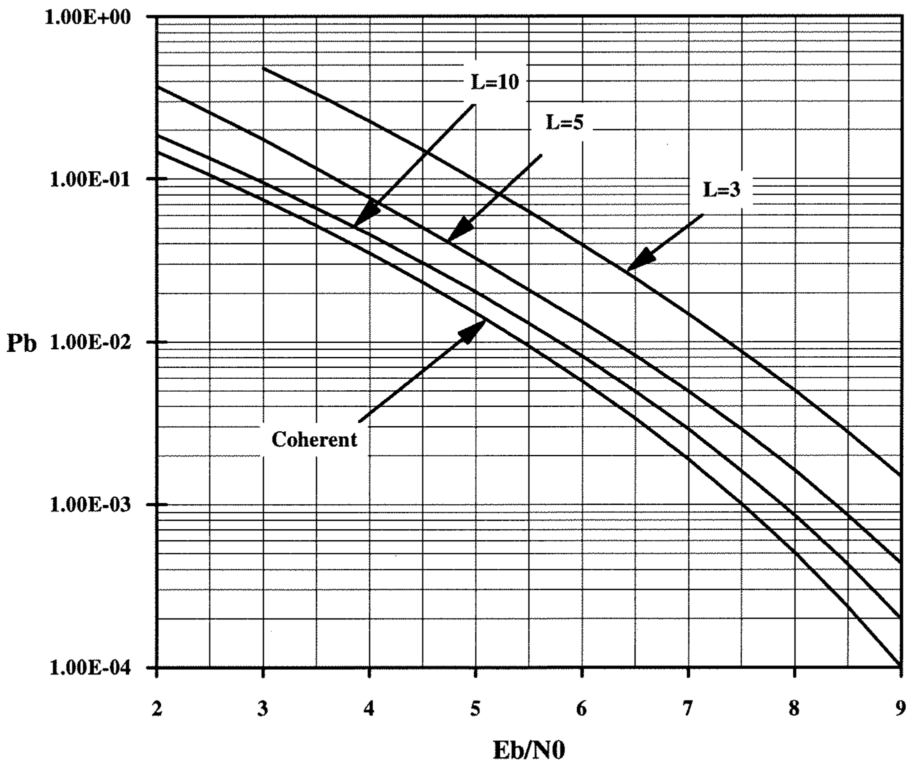


Figure 8.13: Noncoherent decoding performance of 3RC, $h = 1/2$, 2-level CPM.

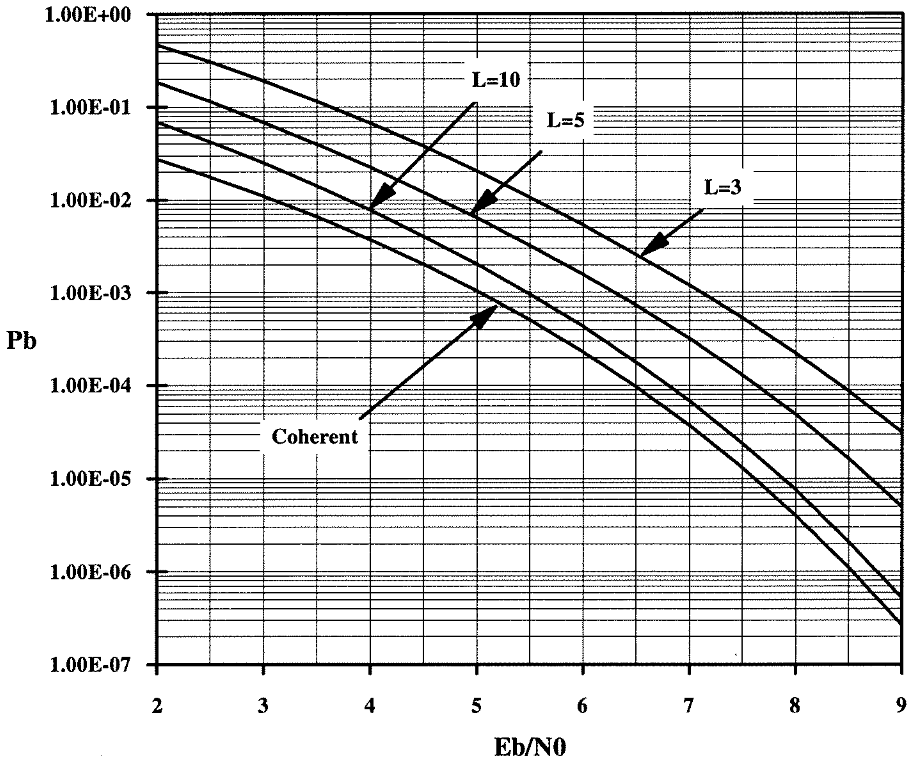


Figure 8.14: Noncoherent decoding performance of 3RC, $h = 4/5$, 2-level CPM.

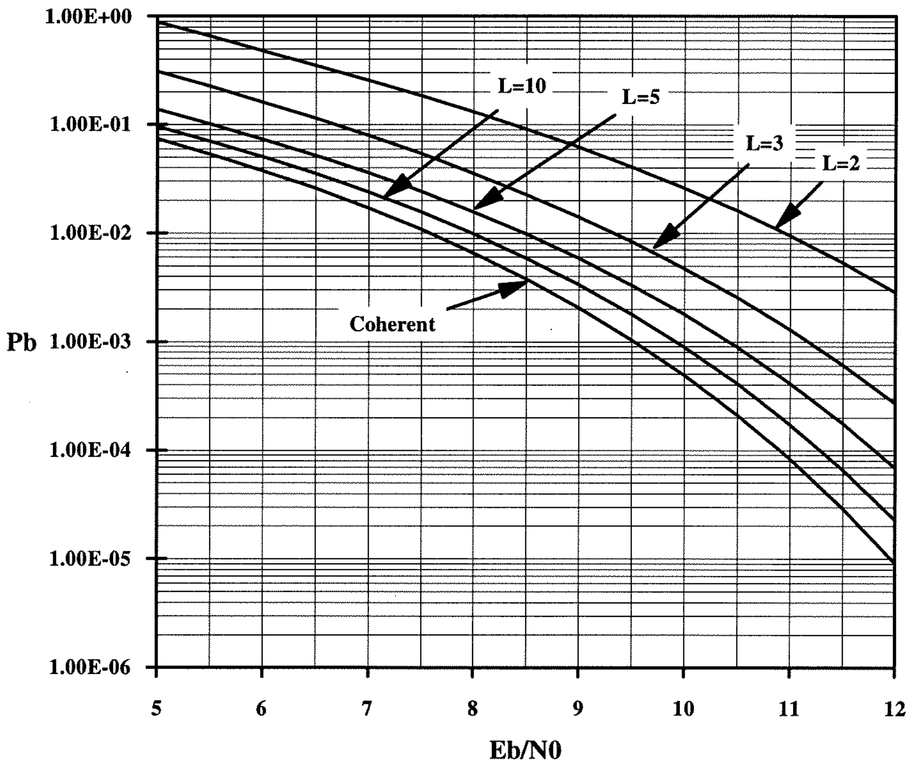


Figure 8.15: Noncoherent decoding performance of 3RC, $h = 2/7$, 4-level CPM.

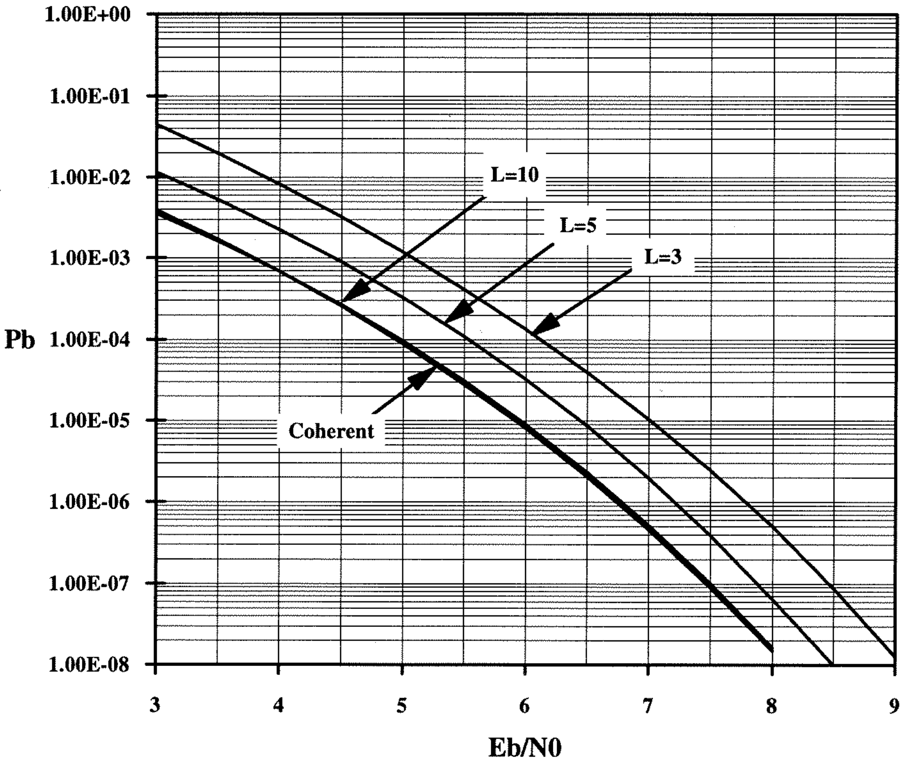


Figure 8.16: Noncoherent decoding performance of 3RC, $h = 4/5$, 4-level CPM.

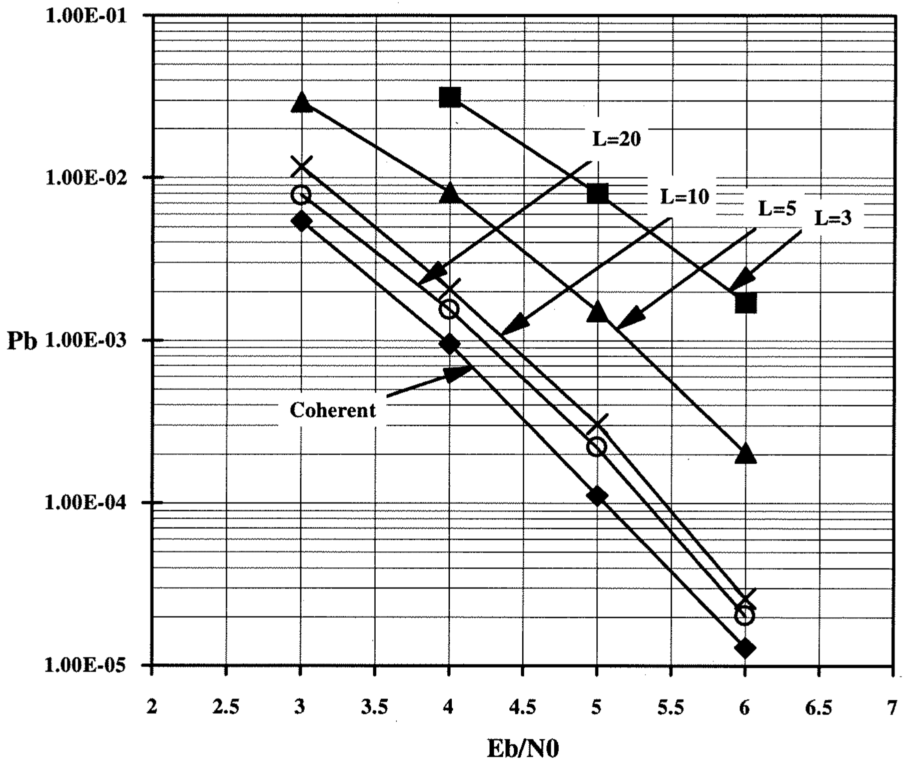


Figure 8.17: Simulation results of coded CPM with noncoherent detection. A 4 states binary rate 1/2 code is used with $h = 1/4$ 4-level CPFSK. The encoder is shown in Figure 8.2.

Bibliography

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

[2] Aulin T., Rydbeck N., Sundberg C.E., "Continuous Phase Modulation (CPM) Parts 1 and 2," IEEE Trans. Comm., Vol. 29, No. 3, pp. 196–225, Mar. 1981.

[3] Mazur B.A., Taylor D.P., "Demodulation and Carrier Synchronization of Multi-h Phase Codes," IEEE Trans. Comm., Vol. 29, No. 3, pp. 257–266, Mar. 1981.

[4] Lindell G., Sundberg C.W., "An Upper Bound on the Bit Error Probability of Combined Convolutional Coding and Continuous Phase Modulation," IEEE Trans. Comm., Vol. 34, No. 5, pp. 1263–1269, Sep. 1988.

[5] Pizzi S.V., Wilson S.G., "Convolutional Coding Combined with Continuous Phase Modulation," IEEE Trans. Comm., Vol. COM-33, No. 1, pp. 20–29, Jan. 1985.

[6] Lindell G., Sundberg C.E., Aulin T., "Minimum Euclidean Distance for Combinations of Short Rate 1/2 Convolutional Codes and CPFSK Modulation," IEEE Trans. Comm., Vol. IT-30, No. 3, pp. 509–519, May 1984.

[7] Naraghi-Pour M., "Trellis Codes for 4-ary Continuous Phase Frequency Shift Keying," IEEE Trans. Comm., Vol. 41, No. 11, pp. 1582–1587, Nov. 1993.

[8] Luise M., Reggiannini R., "Combined Modulation and Coding for Continuous Phase FSK with Nonuniform Alphabet," IEEE Trans. Comm., Vol. 41, No. 8, pp. 1201–1207, Aug. 1993.

[9] Rimoldi B., “Design of Coded CPFSK Modulation Systems for Bandwidth and Energy Efficiency,” IEEE Trans. Comm., Vol. 37, No. 9, pp. 897–905, Sep. 1989.

[10] Proakis J.G., *Digital Communications.* NY: McGraw Hill, 1989.

[11] Masamura T., Samejima S., Morihiro Y., Fuketa H., “Differential Detection of MSK with Nonredundant Error Correction,” IEEE Trans. Comm. COM-27(6), 912, 1979.

[12] Hirade K., Ishizuka M., Adachi F., Ohtani K., “Error-Rate Performance of Digital FM with Differential Detection in Mobile Radio Channels,” IEEE Trans. Veh. Tech., VT-28, pp. 204–212, 1979.

[13] Simon M.K., Wang C.C., “Differential Versus Limiter-Discriminator Detection of Narrow-band FM,” IEEE trans. Comm., Vol. 31, No. 11, pp. 1227–1234, Nov. 1983.

[14] Aulin T., Sundberg C.E., “Detection Performance of band-Limited continuous Phase Modulation,” GLOBECOM’82, Miami, FL, Conf. Rec., pp. E7.6.1–E7.6.7, Nov. 1982.

[15] Stjernvall J.E., Uddenfeldt J., “Gaussian MSK with different Demodulators and Channel Coding for Mobile Telephony,” ICC’84, Amsterdam, The Netherlands, Conf. Rec., pp. 1219–1222, May 1984.

[16] Hirade K., Murota K., Hata M., “GMSK Transmission Performance In Land Mobile Radio,” GLOBECOM’82, Conf. Rec., pp. B3.1.4–B3.4.6.

[17] Svensson A., Sundberg C.E., “On error Probability for Several Types of Noncoherent Detection of CPM,” Globecom’84, Atlanta, GA, Conf. Rec., pp. 22.5.1–22.5.7, Nov. 1984.

[18] Makrakis D., Mathiopoulos P.T., “Differential Detection of Correlative Encoded Continuous Phase Modulation Schemes Using Decision Feedback,” IEE Proceedings, Vol. 138, No. 5, pp. 473–480, Oct. 1991.

[19] Kaleh G.K., "Differential Detection Via the Viterbi Algorithm for Offset Modulation and MSK-Type Signals," IEEE Trans. Comm., Vol. 41, No. 4, pp. 401–406, Nov. 1992.

[20] Makrakis D., Feher K., "Multiple Differential Detection of Continuous Phase Modulation Signals," IEEE Trans. Comm., Vol. 42, No. 2, pp. 186–196, May 1993.

[21] Chung K.S., "A Noncoherent Receiver for GTFM Signals," GLOBECOM'82, Miami, FL, Conf. Rec., pp. B3.5.1–B3.5.5, Dec. 1982.

[22] Chung K.S., Zegers L.E., "Generalized Tamed Frequency Modulation," Proc. IEEE Inter. Conf. on Acoustics, Speech and Signal Processing, Vol. 3, pp. 1805–1808, Paris, May 1982.

[23] Andrisano O., Chiani M., Verdone R., "Performance of Narrowband CPM Systems with Limiter-Discriminator-Integrator Detection and Decision Feedback Equalization in Mobile Radio Channels," IEEE Trans. Comm., Vol. 42, No. 2, pp. 166–176, May 1993.

[24] Aulin T., Sundberg C.E., "Partially Coherent Detection of Digital Full Response Continuous Phase Modulated Signals," IEEE Trans. Comm., Vol. COM-30, No. 5, pp. 1096–1117, May 1982.

[25] Svensson A., Aulin T., Sundberg C.E., "Symbol Error Probability Behavior for Continuous Phase Modulation with Partially Coherent Detection," AEU, Vol. 40, No. 1, pp. 37–45, 1986.

[26] Harrold W., Kingsbury N., "A Partially Coherent Detector for Continuous Phase Modulation," IEEE Selec. Comm., Vol. 7, No. 9, pp. 1415–1426, Dec. 1989.

[27] Leib H., Pasupathy S., "Noncoherent Block Demodulation of MSK with Inherent and Enhanced Encoding," IEEE Trans. Comm., Vol. 40, No. 9, pp. 1430–1441, Sep. 1992.

[28] Simon M.K., Divsalar D., "Maximum Likelihood Block Detection of Noncoherent Continuous Phase Modulation," IEEE Trans. Comm., Vol. 41, No. 1, pp. 90–98, Jan. 1993.

[29] Abrardo A., Benelli G., Cau G., "Multiple-Symbols Differential Detection of GMSK," Elect. Letters, Vol. 29, No. 25, pp. 2167–2168, Dec. 1993.

[30] Murota K., Hirade K., "GMSK Modulation for Digital Mobile Radio Telephone," IEEE Trans. Comm., Vol. 29, No. 7, pp. 1044-1050, July 1981.

[31] Viterbi A.J., Omura J.K., *Principles of Digital Communication and Coding.* McGraw-Hill, 1979.

Chapter 9

Epilogue

Many investigations in the past dealt with the problem of reliable communication. Most of these considered ideal coherent channels, while others were consumed in trying to achieve coherency by phase tracking. Both areas have reached maturity. Practical channels are time varying. In particular, we note the areas of mobile and satellite communications. In these environments coherent techniques are either pushed to their limits or completely fail. Therefore, in a system where power efficiency is also a very important factor, there is a need, now more than ever, to minimize the existing performance gap between coherent and noncoherent modulation techniques. We have succeeded in minimizing this gap. While doing so, we do not have to use impractical schemes. Many designers of future systems will probably like to investigate the schemes proposed in this thesis for practical applications especially for mobile, satellite, personal and deep space communications.

In the course of this research many interesting, challenging, unsolved problems in mathematics and communication theory have aroused. We hope that this work will stimulate interest in noncoherent communications in general and in the methods developed in this thesis in particular.

Appendix A

Asymptotic Performance of the IO-NMLSE

We will show that the pairwise error probability between two sequences $\mathbf{x}$ and $\mathbf{y}$ of the output of a trellis encoder, when IO-NMLSE is used for the decoding, approaches the coherent error probability for $S \to \infty$. The encoder is assumed not to be noncoherently catastrophic.

While S increased, the allowed phase variations are simultaneously reduced, such that the phase stays approximately constant over the S-symbols observations. In the limit the phase must be constant everywhere. With a given phase process which is not constant, the noncoherent performance *cannot* approach the performance of coherent decoder that completely knows the phase (the phase is given to the decoder as a side information).

Let $\mathbf{x}$ be sent with an arbitrary phase shift θ. Let $\mathbf{x}$ and $\mathbf{y}$ differ in d consecutive places, $i \in \gamma = \{t, t+1, \cdots, t+d-1\}$, and assume d is small, since it is clear that the average error probability is determined mainly by the short error events. Error events (if they exist) which cause $\mathbf{y}_i$ to be a phase shift of $\mathbf{x}_i$ $\forall i \geq t+d$ will have diminishing contribution for large S. Choose S large enough such that $S >> d$. Let $\mathbf{r}_i = \mathbf{x} + \mathbf{n}$ and

$$s = \sum_{k=-\infty}^{\infty} \left| \sum_{i=k}^{k+S-1} \mathbf{r}_i^\dagger \mathbf{y}_i \right|^2 - \sum_{k=-\infty}^{\infty} \left| \sum_{i=k}^{k+S-1} \mathbf{r}_i^\dagger \mathbf{x}_i \right|^2, \tag{A.1}$$

then the pairwise error probability of deciding $\mathbf{y}$ instead of $\mathbf{x}$ is $\Pr\{s > 0\}$.

$$s = \sum_{k=-\infty}^{\infty} \left\{ \left| \sum_{\substack{i=k \\ i \notin \gamma}}^{k+S-1} \mathbf{r}_i^\dagger \mathbf{y}_i + \sum_{\substack{i=k \\ i \in \gamma}}^{k+S-1} \mathbf{r}_i^\dagger \mathbf{y}_i \right|^2 - \left| \sum_{\substack{i=k \\ i \notin \gamma}}^{k+S-1} \mathbf{r}_i \mathbf{x}_i + \sum_{\substack{i=k \\ i \in \gamma}}^{k+S-1} \mathbf{r}_i \mathbf{x}_i \right|^2 \right\} =$$

$$\sum_{k=-\infty}^{\infty}\Big\{\Big|\sum_{\substack{i=k\\ i\notin\gamma}}^{k+S-1}\mathbf{r}_i^\dagger\mathbf{y}_i\Big|^2+\Big|\sum_{\substack{i=k\\ i\in\gamma}}^{k+S-1}\mathbf{r}_i^\dagger\mathbf{y}_i\Big|^2+2\mathrm{Re}\Big[\Big(\sum_{\substack{i=k\\ i\notin\gamma}}^{k+S-1}\mathbf{r}_i^\dagger\mathbf{y}_i\Big)^*\sum_{\substack{i=k\\ i\in\gamma}}^{k+S-1}\mathbf{r}_i^\dagger\mathbf{y}_i\Big]-$$
$$-\Big|\sum_{\substack{i=k\\ i\notin\gamma}}^{k+S-1}\mathbf{r}_i^\dagger\mathbf{x}_i\Big|^2-\Big|\sum_{\substack{i=k\\ i\in\gamma}}^{k+S-1}\mathbf{r}_i^\dagger\mathbf{x}_i\Big|^2-2\mathrm{Re}\Big[\Big(\sum_{\substack{i=k\\ i\notin\gamma}}^{k+S-1}\mathbf{r}_i^\dagger\mathbf{x}_i\Big)^*\sum_{\substack{i=k\\ i\in\gamma}}^{k+S-1}\mathbf{r}_i^\dagger\mathbf{x}_i\Big]\Big\}. \tag{A.2}$$

For $S >> d$

$$\sum_{\substack{i=k\\ i\notin\gamma}}^{k+S-1}\mathbf{r}_i^\dagger\mathbf{x}_i \cong Se^{j\theta}\cdot E, \tag{A.3}$$

where $E=\overline{\mathbf{x}_i^\dagger\mathbf{x}_i}$. Let

$$\beta=\frac{1}{S}\mathrm{Re}\Big\{\sum_{k=-\infty}^{\infty}\Big\{\sum_{\substack{i=k\\ i\in\gamma}}^{k+S-1}e^{-j\theta}\mathbf{r}_i^\dagger(\mathbf{y}_i-\mathbf{x}_i)\Big\}\Big\}=\mathrm{Re}\Big\{\sum_{k=-\infty}^{\infty}e^{-j\theta}\mathbf{r}_k^\dagger(\mathbf{y}_k-\mathbf{x}_k)\Big\}, \tag{A.4}$$

and

$$\epsilon=\sum_{k=-\infty}^{\infty}\Big\{\Big|\sum_{\substack{i=k\\ i\in\gamma}}^{k+S-1}\mathbf{r}_i^\dagger\mathbf{y}_i\Big|^2-\Big|\sum_{\substack{i=k\\ i\in\gamma}}^{k+S-1}\mathbf{r}_i^\dagger\mathbf{x}_i\Big|^2\Big\}. \tag{A.5}$$

For $S\geq d$, $\epsilon=\epsilon_0+(S-d)\epsilon_1$, where

$$\epsilon_0=\sum_{k=t}^{t+d-1}\Big\{\Big|\sum_{i=t}^{k}\mathbf{r}_i^\dagger\mathbf{y}_i\Big|^2-\Big|\sum_{i=t}^{k}\mathbf{r}_i^\dagger\mathbf{x}_i\Big|^2\Big\}+\sum_{k=t+1}^{t+d-1}\Big\{\Big|\sum_{i=k}^{t+d-1}\mathbf{r}_i^\dagger\mathbf{y}_i\Big|^2-\Big|\sum_{i=k}^{t+d-1}\mathbf{r}_i^\dagger\mathbf{x}_i\Big|^2\Big\}, \tag{A.6}$$

and

$$\epsilon_1=\Big|\sum_{i=t}^{t+d-1}\mathbf{r}_i^\dagger\mathbf{y}_i\Big|^2-\Big|\sum_{i=t}^{t+d-1}\mathbf{r}_i^\dagger\mathbf{x}_i\Big|^2. \tag{A.7}$$

$$s=2S^2E\beta+\epsilon=2S^2E\beta+\epsilon_0+(S-d)\epsilon_1. \tag{A.8}$$

β,ϵ_0 and ϵ_1 are not a function of S so as $S\to\infty$, $s=2S^2E\beta$. Finally, for $S\to\infty$,

$$\Pr\{s>0\}=\Pr\Big\{\mathrm{Re}\{\sum_{k=-\infty}^{\infty}e^{-j\theta}\mathbf{r}_k^\dagger(\mathbf{y}_k-\mathbf{x}_k)\}>0\Big\}=\Pr(\text{coherent detection}). \tag{A.9}$$

Appendix B

Series Expansions for the Distribution of Non-central Indefinite Hermitian Quadratic Forms

A new series expansion is developed for the probability distribution function (p.d.f.) and the cumulative distribution function (c.d.f.) for indefinite non-central Hermitian quadratic forms in complex normal random variables. The moment generating function is inverted by contour integration using the residue theorem. The function is separated into two parts; one part, containing an essential singularity, is expanded by Laurant series and the other part is expanded by Taylor series. The series are combined for evaluating the residue of the complete function. Three different versions are presented for evaluating either the p.d.f. or the c.d.f. The series are computationally efficient. The rate of convergence depends on the eigenvalues separation and is usually very fast.

The Quadratic Form (QF) in normal random variables appears in many applications in statistics and in communication theory. Much attention has been given over the years to the problem of evaluating its distribution, and various series expansions were developed [3]–[8], [12]–[18], [25]–[26] [22], [10], [21]. However, most of the authors considered positive definite or semidefinite cases. Relatively, little attention has been devoted to the problem of obtaining the distribution of indefinite QF. Several series expansions to the indefinite case were developed [3]–[13]. Most of these are complex so their practical usefulness is limited. Here, we present a new technique for developing series expansions which result in some computationally efficient expansions.

Let A be a $N \times N$ Hermitian matrix, and $\mathbf{r}$ be a complex normal random vector with covariance matrix V and expected value η. A Quadratic Form (QF) in $\mathbf{r}$ is

$$Y = \mathbf{r}^{\dagger} A \mathbf{r}. \tag{B.1}$$

A QF is said to be definite or indefinite depending whether the matrix A is definite or indefinite. If $\eta = 0$, the QF is said to be central; otherwise, it is noncentral.

According to Section (6.4.3) and [1] and [2], the QF (B.1) can be expressed as

$$Y = Z^{\dagger} \Lambda Z = \sum_{i=0}^{N-1} \lambda_i |Z_i|^2, \tag{B.2}$$

where Z is a Gaussian vector with independent components each having a unit variance in the real and imaginary parts and expected value $\mu = E[Z] = Q^{\dagger} L^{-1\dagger} \eta$. The matrix L is any non-singular factorization of V, such that $V = L^{\dagger} L$. λ_i are the eigenvalues of $LAL^{\dagger}$ and Q is the corresponding unitary eigenvectors matrix.

Here, we develop a new series expansion for the p.d.f. and the c.d.f.. We treat the case of Hermitian QF in complex random vectors. This is equivalent to a special case of real valued QF, where the eigenvalues are always given in pairs. The resulting series is computationally efficient. Distinct eigenvalues are assumed, and their separation determines the rate of convergence. However, the method can easily be extended to the multiple eigenvalue case.

For the general quadratic form expressed as

$$Y = \sum_{j=0}^{n-1} \lambda_j (W_j - \omega_j)^2,$$

the moment generating function is given by [1] and is:

$$\Phi(s) = E[e^{sy}] = e^{-\frac{1}{2}\sum_{j=0}^{n-1} \omega_j^2} e^{\frac{1}{2}\sum_{j=o}^{n-1} \frac{\omega_j^2}{1 - 2s\lambda_j}} \prod_{j=0}^{n-1} (1 - 2s\lambda_j)^{-\frac{1}{2}}. \tag{B.3}$$

W_i are unit real normal random variables and ω_i are real constants. In our case Z_i is complex, so we can write

$$|Z_i|^2 = (W_{2i} - w_{2i})^2 + (W_{2i+1} - w_{2i+1})^2, \quad \lambda_{2i} = \lambda_{2i+1}, n = 2N, \tag{B.4}$$

and

$$\omega_{2i} = Re\{\mu_i\}, \quad \omega_{2i+1} = Im\{\mu_i\}. \tag{B.5}$$

From (B.4) and (B.3) we get[1]

$$\Phi(s) = e^{-\frac{1}{2}\sum_j |\mu_j|^2} e^{\frac{1}{2}\sum_j \frac{|\mu_j|^2}{1-2\lambda_j s}} \prod_j \frac{1}{1-2\lambda_j s}. \tag{B.6}$$

By the Laplace inversion theorem [26], we get the p.d.f.

$$p(y) = \frac{1}{2\pi i} \lim_{R\to\infty} \int_{\sigma-iR}^{\sigma+iR} e^{-si}\Phi(s)ds. \tag{B.7}$$

We choose $\sigma = 0$, where the function is always analytic. Since $\Phi(s)$ vanishes at infinity, we can replace the integral by a contour integral C circling the whole right half plane for $y \geq 0$ and the left half plane for $y < 0$. Furthermore, we can use the residue theorem to evaluate this contour integral. For $y \geq 0$,

$$p(y) = -\frac{1}{2\pi i} \oint_C e^{-sy}\Phi(s)ds = -\sum_{\lambda_k>0} \mathrm{Res}_{e^{-sy}\Phi}(\frac{1}{2\lambda_k}), \tag{B.8}$$

and for $y < 0$

$$p(y) = \sum_{k,\lambda_k<0} \mathrm{Res}_{e^{-sy}\Phi}(\frac{1}{2\lambda_k}). \tag{B.9}$$

Let us proceed with the case of $y > 0$. We can alternatively write

$$p(y) = -\sum_{\lambda_k>0} \mathrm{Res}_{F_k}(0), \tag{B.10}$$

where

$$F_k(s) = e^{-(\frac{1}{2\lambda_k}+s)y}\Phi(\frac{1}{2\lambda_k}+s). \tag{B.11}$$

The pole at $\frac{1}{2\lambda_k}$ was translated to $s = 0$ for convenience. We will assume that the eigenvalues are distinct.

[1]In all the following summations and products, when the range is not specified, it is from 0 to $N-1$.

$$F_k(s) = e^{-\frac{1}{2}\sum_j |\mu_j|^2 + \frac{1}{2}\sum_j \frac{|\mu_j|^2}{1-2\lambda_j(\frac{1}{2\lambda_k}+s)}} \cdot$$

$$\cdot \prod_j \frac{1}{1-2\lambda_j(\frac{1}{2\lambda_k}+s)} e^{-\frac{1}{2\lambda_k}y - sy} =$$

$$= -e^{-\frac{1}{2}\sum_j |\mu_j|^2 + \frac{1}{2}\sum_{j\neq k} \frac{|\mu_j|^2}{1-\frac{\lambda_j}{\lambda_k}-2\lambda_j s} - \frac{|\mu_k|^2}{4\lambda_k s}} \cdot$$

$$\cdot \prod_{j\neq k} \frac{1}{1-\frac{\lambda_j}{\lambda_k}-2\lambda_j s} \cdot \frac{1}{2\lambda_k s} \cdot e^{-\frac{1}{2\lambda_k}y - sy}. \tag{B.12}$$

By following the same principle we will present three different methods for obtaining series expansions for the p.d.f. and the c.d.f.

Method 1:

Let us define

$$g_k(s) = e^{\frac{1}{2}\sum_{j\neq k} \frac{|\mu_j|^2}{1-\frac{\lambda_j}{\lambda_k}-2\lambda_j s}} \cdot \prod_{j\neq k} \frac{1}{1-\frac{\lambda_j}{\lambda_k}-2\lambda_j s} \tag{B.13}$$

and

$$f_k(s) = \frac{1}{s} e^{-sy - \frac{|\mu_k|^2}{4\lambda_k s}}, \tag{B.14}$$

then

$$F_k(s) = -\frac{1}{2\lambda_k} e^{-\frac{1}{2}\sum_j |\mu_j|^2 - \frac{1}{2\lambda_k}y} \cdot g_k(s) \cdot f_k(s). \tag{B.15}$$

From this point and on, we omit the k whenever the dependence of k is understood. We write $f(s)$ as Laurant series and $g(s)$ as Taylor series,

$$f(s) = \sum_{n=0}^{\infty} b_n s^{-n-1} + \sum_{n=0}^{\infty} a_n s^n \tag{B.16}$$

$$g(s) = \sum_{n=0}^{\infty} a_n s^n, \quad a_n = \frac{1}{n!} g^{(n)}(0). \tag{B.17}$$

Since $f(s)$ is analytic for all $s \neq 0$, equation (B.16) converges uniformly for all $s \neq 0$. The expansion (B.17) converges uniformly in a circle around $s = 0$. Since the residue of $F(s)$ at $s = 0$ is the coefficient of the s^{-1} term in the Laurant series expansion of $F(s)$, we get

$$Res_F(0) = -\frac{1}{2\lambda_k} e^{-\frac{1}{2}\sum_j |\mu_j|^2 - \frac{1}{2\lambda_k}y} \cdot \sum_{n=0}^{\infty} b_n a_n. \tag{B.18}$$

Now we find a recursive formula method to compute $g^{(n)}(s)$.

$$\ln g(s) = \frac{1}{2}\sum_{j\neq k} \frac{|\mu_j|^2}{\alpha_j - 2\lambda_j s} + \sum_{j\neq k} \ln\left(\frac{1}{\alpha_j - 2\lambda_j s}\right), \tag{B.19}$$

where $\alpha_j = 1 - \frac{\lambda_j}{\lambda_k}$.

$$[\ln g(s)]' = \sum_{j\neq k} \frac{\lambda_j |\mu_j|^2}{(\alpha_j - 2\lambda_j s)^2} + \sum_{j\neq k} \frac{2\lambda_j}{\alpha_j - 2\lambda_j s}. \tag{B.20}$$

By induction we get

$$[\ln g(s)]^{(n)} = \sum_{j\neq k} \frac{n!\lambda_j^n 2^{n-1}|\mu_j|^2}{(\alpha_j - 2\lambda_j s)^{n+1}} + \frac{(n-1)!\lambda_j^n 2^n}{(\alpha_j - 2\lambda_j s)^n}. \tag{B.21}$$

Since $g' = g(\ln g)'$, taking its $n-1$ derivative yields to

$$g^{(n)} = \sum_{l=0}^{n-1} \binom{n-1}{l} g^{(l)} (\ln g)^{(n-l)}. \tag{B.22}$$

Now we can compute $g^{(n)}$ out of $g, g', \ldots, g^{(n-1)}$. Let us move on to evaluate b_n by contour integration. We choose a circular contour around the pole at $s = 0$. Since the function $f(s)$ is analytic everywhere except at $s = 0$, we can arbitrarily choose any contour that circles the point $s = 0$.

$$s = \varepsilon e^{i\phi}, \quad ds = \varepsilon e^{i\phi} i d\phi. \tag{B.23}$$

Then,

$$b_n = \frac{1}{2\pi i}\oint s^n f(s) ds = \frac{1}{2\pi}\int_0^{2\pi} \varepsilon^n e^{in\phi - \varepsilon y e^{i\phi} - \frac{|\mu_k|^2}{4\lambda_k \varepsilon} e^{-i\phi}} d\phi. \tag{B.24}$$

Choosing $\varepsilon = \frac{|\mu_k|}{2\sqrt{y\lambda_k}}$, we get,

$$b_n = \frac{\varepsilon^n}{2\pi}\int_0^{2\pi} e^{in\phi - \sqrt{\frac{|\mu_k|^2 y}{\lambda_k}}\cos\phi} d\phi = \varepsilon^n I_n\left(-\sqrt{\frac{|\mu_k|^2 y}{\lambda_k}}\right). \tag{B.25}$$

$I_n(x)$ is the modified Bessel function of the first kind of order n.

$$p(y) = -\sum_{\lambda_k>0} \mathrm{Res}_{F_k}(0) =$$

$$= e^{-\frac{1}{2}\sum_k |\mu_k|^2} \sum_{\lambda_k>0} \frac{1}{2\lambda_k} e^{-\frac{y}{2\lambda_k}} \sum_{n=0}^{\infty} \frac{1}{n!} g_k^{(n)}(0) \left(\frac{|\mu_k|^2}{4y\lambda_k}\right)^{\frac{n}{2}} I_n\left(-\sqrt{\frac{|\mu_k|^2 y}{\lambda_k}}\right). \tag{B.26}$$

Let

$$B_{k,n} = \frac{1}{2\lambda_k} \int_0^{\infty} e^{-\frac{y}{2\lambda_k}} \left(\frac{|\mu_k|^2}{4y\lambda_k}\right)^{\frac{n}{2}} I_n\left(-\sqrt{\frac{|\mu_k|^2 y}{\lambda_k}}\right) =$$

$$= \begin{cases} e^{\frac{|\mu_k|^2}{2}}, & \text{if } n = 0; \\ (-2\lambda_k)^{-n} \cdot \frac{1}{(n-1)!} e^{\frac{|\mu_k|^2}{2}} \Gamma(n, 0, \frac{|\mu_k|^2}{2}), & \text{otherwise.} \end{cases} \tag{B.27}$$

Then,

$$\Pr\{y \geq 0\} = \int_0^{\infty} p(y)dy = e^{-\frac{1}{2}\sum_k |\mu_k|^2} \cdot \sum_{\lambda_k>0} \sum_{n=0}^{\infty} \frac{1}{n!} g_n^{(n)}(0) B_{k,n}. \tag{B.28}$$

In order to find $\Pr\{y \geq \gamma\}$, $\gamma > 0$, we do the following changes: We substitute $y + \gamma$ for y in (B.12). Then we define

$$g_k(s) = e^{\frac{1}{2}\sum_{j\neq k} \frac{|\mu_j|^2}{1 - \frac{\lambda_j}{\lambda_k} - 2\lambda_j s}} \cdot \prod_{j\neq k} \frac{1}{1 - \frac{\lambda_j}{\lambda_k} - 2\lambda_j s} \cdot e^{-\gamma s}, \tag{B.29}$$

and (B.15) becomes

$$F_k(s) = -\frac{1}{2\lambda_k} e^{-\frac{1}{2}\sum_j |\mu_j|^2 - \frac{\gamma + y}{2\lambda_k}} \cdot g_k(s) \cdot f_k(s). \tag{B.30}$$

Equation (B.19) and (B.20) change to

$$\ln g(s,\gamma) = \frac{1}{2} \sum_{j\neq k} \frac{|\mu_j|^2}{\alpha_j - 2\lambda_j s} + \sum_{j\neq k} \ln\left(\frac{1}{\alpha_j - 2\lambda_j s}\right) - \gamma s, \tag{B.31}$$

$$\frac{\partial}{\partial s}[\ln g(s,\gamma)] = \sum_{j\neq k} \frac{\lambda_j |\mu_j|^2}{(\alpha_j - 2\lambda_j s)^2} + \sum_{j\neq k} \frac{2\lambda_j}{\alpha_j - 2\lambda_j s} - \gamma. \tag{B.32}$$

Then,

$$\Pr\{y \geq \gamma\} = e^{-\frac{1}{2}\sum_k |\mu_k|^2} \sum_{\lambda_k>0} e^{-\frac{\gamma}{2\lambda_k}} \sum_{n=0}^{\infty} \frac{1}{n!} g_k^{(n)}(0) B_{k,n}. \tag{B.33}$$

Method 2:

By differentiating (B.33) with respect to γ, we get the following alternative series expansion for $p(y)$ (for a positive argument only).

$$p(\gamma) = e^{-\frac{1}{2}\sum_k |\mu_k|^2} \sum_{\lambda_k>0} e^{\frac{-\gamma}{2\lambda_k}} \sum_{n=0}^{\infty} \frac{1}{n!} B_{k,n} \left(\frac{1}{2\lambda_k} \frac{\partial^{(n)}}{\partial s^n} g - \frac{\partial}{\partial \gamma} \frac{\partial^{(n)}}{\partial s^n} g \right) \bigg|_{s=0}. \tag{B.34}$$

A recursive formula for $\frac{\partial}{\partial \gamma}\frac{\partial^{(n)}}{\partial s^n} g\Big|_{s=0}$ is obtained by differentiating (B.19)–(B.22):

$$\frac{\partial}{\partial \gamma} g \bigg|_{s=0} = 0, \tag{B.35}$$

$$\frac{\partial}{\partial \gamma} \frac{\partial^{(n)}}{\partial s^n} [\ln g] \bigg|_{s=0} = \begin{cases} \text{-1}, & \text{if } n = 1; \\ 0, & \text{if } n \neq 1 \end{cases}, \tag{B.36}$$

$$\frac{\partial}{\partial \gamma} \frac{\partial^{(n)}}{\partial s^{(n)}} g \bigg|_{s=0} = \sum_{l=1}^{n-1} \binom{n-1}{l} \frac{\partial}{\partial \gamma} \frac{\partial^{(l)}}{\partial s^l} g \frac{\partial^{(n-l)}}{\partial s^{(n-l)}} [\ln g] \bigg|_{s=0} - (n-1) \frac{\partial}{\partial s} g \bigg|_{s=0} \quad n > 0, \tag{B.37}$$

$$\frac{\partial}{\partial \gamma} \frac{\partial}{\partial s} g \bigg|_{s=0} = -g(0,0). \tag{B.38}$$

Method 3:

Let us repeat method 1, but now let us move the e^{-sy} term from $f(s)$ to $g(s)$. The form of $g(s)$ is exactly the same as in (B.29), replacing γ with y. Equations (B.31) and (B.32) change accordingly. Let $\beta = \frac{|\mu_k|^2}{4\lambda_k}$, then

$$f(s) = \frac{1}{s} e^{-\frac{\beta}{s}} = \sum_{n=0}^{\infty} \frac{1}{n!} \frac{(-\beta)^n}{s^{n+1}}. \tag{B.39}$$

Hence, $b_n = \frac{1}{n!}(-\beta)$. From (B.18) we get

$$Res_{F_k} = -\frac{1}{2\lambda_k} e^{-\frac{1}{2}\sum_j |\mu_j|^2 - \frac{1}{2\lambda_k} y} \sum_{n=0}^{\infty} \frac{g^{(n)}}{(n!)^2} (-\beta)^n. \tag{B.40}$$

Using (B.10), we get

$$p(y) = e^{-\frac{1}{2}\sum_j |\mu_j|^2} \sum_{\lambda_k>0} \frac{1}{2\lambda_k} e^{-\frac{1}{2\lambda_k} y} \sum_{n=0}^{\infty} \frac{\partial^{(n)}}{\partial s} \frac{g_k(0,y)}{(n!)^2} \left(-\frac{|\mu_k|^2}{4\lambda_k} \right)^n. \tag{B.41}$$

In order to get the c.d.f., we can integrate $p(y)$. However, we can get the c.d.f. series directly.

$$\int_{\gamma}^{\infty} p(y)dy = -\frac{1}{2\pi i}\oint_{C} \frac{1}{s}e^{-s\gamma}\Phi(s)ds, \tag{B.42}$$

where $\gamma > 0$. Let

$$F_k(s) = \frac{1}{\frac{1}{2\lambda_k}+s} e^{-(\frac{1}{2\lambda_k}+s)\gamma}\Phi\Big(\frac{1}{2\lambda_k}+s\Big), \tag{B.43}$$

$$\begin{aligned} g(s,\gamma) &= e^{\frac{1}{2}\sum_{j\neq k}\frac{|\mu_j|^2}{\alpha_j+s}} \prod_{j\neq k}\frac{1}{\alpha_j - a\lambda_j s}\cdot\frac{1}{\frac{1}{2\lambda_k}+s}e^{-s\gamma} = \\ &= -2\lambda_k e^{\frac{1}{2}\sum_{j\neq k}\frac{|\mu_j|^2}{\alpha_j+s}} \prod_{j}\frac{1}{\alpha_j - 2\lambda_j s}\cdot e^{-s\gamma}, \end{aligned} \tag{B.44}$$

where $\alpha_k \triangleq -1$, and

$$f(s) = \frac{1}{s}e^{-\frac{|\mu_k|^2}{4\lambda_k s}}. \tag{B.45}$$

Then

$$\int_{\gamma}^{\infty} p(y)dy = -e^{-\frac{1}{2}\sum_j |\mu_j|^2} \sum_{\lambda_k>0} e^{-\frac{\gamma}{2\lambda_k}} \sum_{n=0}^{\infty}\frac{1}{(n!)^2}\frac{\partial^{(n)}g(0,\gamma)}{\partial s^n}.\Big(-\frac{|\mu_k|^2}{4\lambda_k}\Big)^n. \tag{B.46}$$

B.1 Convergence Rate

For values of $|\mu_k|^2$, λ_k and y on the same order, the summand value is dominated by $\alpha_{\text{min}}/(n!)^2$ where α_{min} is the smallest α_{jk}. If there is a pair of eigenvalues close together, $\alpha_{\text{min}} << 1$ and then the convergence will be slow. If the eigenvalues are separated by a ratio of at least 1.5, we observe a very fast convergence, and only a few terms will be sufficient for practical applications. In order to overcome this problem, close pairs of eigenvalues can be approximated by a multiple eigenvalue. The methods described can easily be extended to handle multiple eigenvalues.

Appendix C

Numerical Method for the Distribution of Non-central Indefinite Hermitian Quadratic Forms

The characteristic function of the random variable Y is given by equation (6.24). We would like to find the value of the inverse Fourier transform of $\Phi(j\omega)$ which is $p(y)$, integrated from γ to infinity, where γ is a positive number (for finding the pairwise error probability, $Pr\{Y > 0\}$, take $\gamma = 0$). A straightforward way to find our p.d.f., $p(y)$, is by performing an Inverse Fast Fourier Transform (IFFT) on the samples of $\Phi(j\omega)$. After we get the samples in "time" domain, we sum the points that correspond to $y \geq \gamma$ for approximating $\Pr\{y \geq \gamma\} = \int\limits_{\gamma}^{\infty} p(y)dy$. With this approach we have to take a large number of points to get a good approximation and not to suffer from aliasing. In previous approaches [2],[5] numerical integration was used in various forms. Here we offer an alternative method to the above, which requires neither numerical integration nor FFT. When using numerical integration we always get an approximation to the result. Here, we get an exact formula which is of the form of an infinite series. The method is general enough to apply to many other probability functions. This method demands much less computation time for the same accuracy compared to the existing methods.

Let $\pi(y)$ be a square wave of period P which assumes the values 0 and 1; the period starts with 1 at $y = 0$. For large enough P and assuming $p(y)$ vanishes for large $|y|$, we get (see Figure C.1)

$$\int_{\gamma}^{\infty} p(y)dt \cong \int_{-\infty}^{\infty} \pi(y-\gamma)p(y), \tag{C.1}$$

The Fourier transform of $\pi(y)$ is

$$0.5\delta(\omega) + j\frac{1}{\pi}\sum_{n=-\infty, n\ odd}^{n=\infty} \frac{1}{n}\delta\Big(\omega - n\frac{2\pi}{P}\Big). \tag{C.2}$$

Let

$$g(t) = \int_{-\infty}^{\infty} p(y+\tau)\pi(\tau-\gamma)d\tau. \tag{C.3}$$

Then its Fourier transform is

$$G(j\omega) = 2\pi\Phi(j\omega)\Big(0.5\delta(\omega) - j\frac{1}{\pi}\sum_{n=-\infty, n\ odd}^{n=\infty} \frac{1}{n}\delta(\omega - n\frac{2\pi}{P})\Big)e^{j\gamma\omega}. \tag{C.4}$$

$$\int_{-\infty}^{\infty} \pi(y)p(y) = g(0) = \frac{1}{2\pi}\int_{-\infty}^{\infty} G(j\omega)d\omega =$$

$$= \Big(0.5\Phi(0) - j\frac{1}{\pi}\sum_{n=-\infty, n\ odd}^{n=\infty} \frac{1}{n}\Phi(\omega - n\frac{2\pi}{P})\Big)e^{j\gamma(\omega - n\frac{2\pi}{P})}. \tag{C.5}$$

Since $\Phi(j\omega)$ is a Fourier transform of a real function, its imaginary part is odd and its real part is even. Using also $\Phi(0) = 1$, we get

$$\int_{0}^{\infty} p(t)dt \cong 0.5 - \frac{2}{\pi}\sum_{n=1, n\ odd}^{n=\infty} \frac{1}{n}\mathrm{Im}\Big\{\Phi(\omega - n\frac{2\pi}{P})e^{j\gamma(\omega - n\frac{2\pi}{P})}\Big\}. \tag{C.6}$$

The result is a nicely converging sum that is very efficient in computation. We have not decided yet what the value of P will be. If we choose P to be too small, $p(y)$ will not vanish in the next cycle of $\pi(y)$, especially in the negative y part where we find most of the weight of the probability. On the other hand if P is too large, the frequency points in the summation will be spaced close together, and large number of points will have to be summed. The last will cause a long computation time and accumulation of numerical errors. A way to estimate the value of P that works well in practice is to choose P as 10 times the roughly approximated "center of mass of the negative part of $p(t)$":

$$P = 10\sum_{i=0, \lambda_i<0} -|\mu_i|^2\lambda_i. \tag{C.7}$$

For a certain accuracy, in order to know how many elements of the series there are to sum, we have to check the current summed expression to see if it is small enough relative to the total. The expression itself is not monotonously decreasing in i so we use the part which have the major influence on the size of the expression and still decrease monotonously for large enough ω. This part is:

$$e^{\frac{1}{2}Re\{\sum_j \frac{|\mu_i|^2}{1-2j\lambda_j\omega}\}} \quad . \tag{C.8}$$

The number of elements to sum for 1% accuracy is between 10 to 40 depending mainly on the value of the final result. As the result approaches zero, we have to sum more elements. The computing time needed for this method is very short and is much less than the eigenvalues' computation time. Note that the evaluation of Φ may not require diagonalization of A. We can alternatively use the form [2]

$$\Phi(s) = |I - 2sAV|^{-\frac{1}{2}} e^{-\frac{1}{2}\eta^{\dagger}[I-(I-2sAV)^{-1}]V^{-1}\eta} \quad . \tag{C.9}$$

However, in this case we have to perform a matrix inversion for each new point, so it will probably not lead to a faster computation.

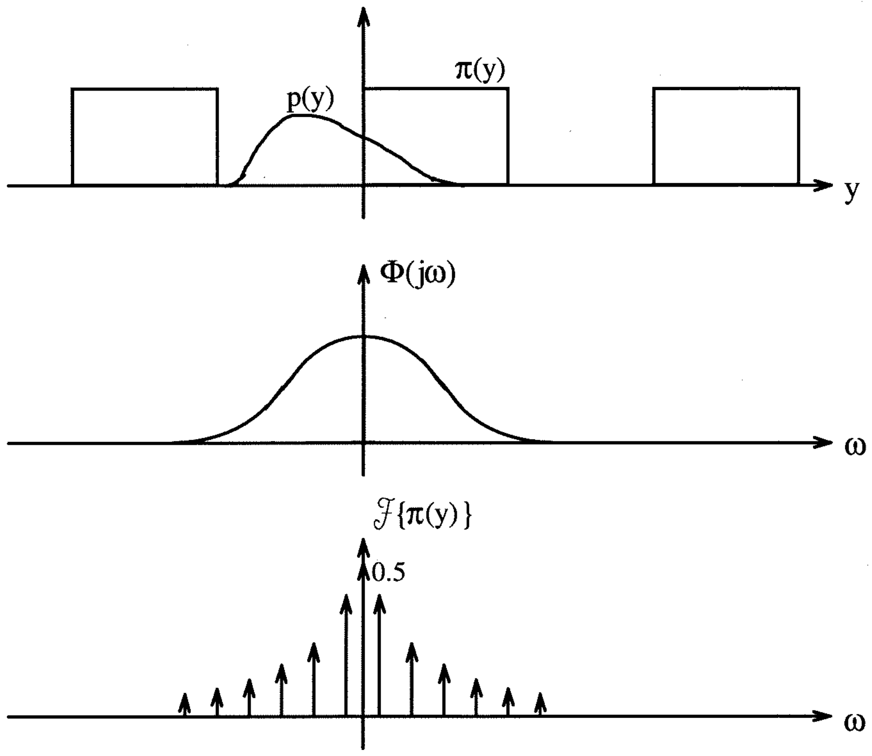


Figure C.1: Waveforms used in Appendix C ($\gamma = 0$).

Bibliography

[1] Johnson N. I., Kotz S., *Continuous Univariate Distributions-2.* NJ: John Wiley & Sons, 1970.

[2] Mathai A.M., Provost S.B., *Quadratic Forms in Random Variables*, New York: Marcel Dekker Inc., 1992.

[3] Gurland J., "Distribution of Definite and of Indefinite Quadratic Forms, "Annals of Mathematical Statistics, Vol. 26, pp. 122–127, 1955.

[4] Shah B.K., "Distribution of Definite and Indefinite Quadratic Forms from a Non-central Normal Distribution," Annals of Mathematical Statistics, Vol. 34, pp. 186–190, 1963.

[5] Imhof J.P., "Computing the Distribution of Quadratic Forms in Normal Variables," Biometrika, Vol. 48, pp. 419–426, 1961.

[6] Press S.J., "Linear Combinations of Non-central Chi-square Variates," Annals of Mathematical Statistics, Vol. 37, pp. 480–487, 1966.

[7] Grad A., Solomon H., "Distribution of Quadratic Forms and Some Applications," Annals of Mathematical Statistics, Vol. 26, pp. 464–477, 1955.

[8] Robinson J., "The Distribution of a General Quadratic Form in Normal Variates," Australian Journal of Statistics, Vol. 7, pp. 110–114, 1965.

[9] Wang Y.Y. "A Comparison of Several Variance Component Estimators," Biometrika, Vol. 54, pp. 301–305, 1967.

[10] Gurland J., "Quadratic Forms in Normally Distributed Random Variables," Sankhya, Vol. 17, pp. 37–50, 1956.

[11] Provost S.B., "The Distribution Function of a Statistic for Testing the Equality of Scale Parameters in Two Gamma Populations," Metrika, Vol. 36, pp. 337–345, 1989.

[12] Mathai A.M., "On Linear Combinations of Independent Exponential Variables," Communications in Statistics-Theory and Methods, Vol. 12(6), pp. 625–632, 1983.

[13] Khatri C.G., "Quadratic Forms in Normal Variables," *Handbook of Statistics,* Vol. 1, P.R. Krishnaiah ed., pp. 443–469, 1980.

[14] Robbins H. E., "The Distribution of a definite Quadratic Form," Annals of Mathematical Statistics, Vol. 19, pp. 266–270, 1948.

[15] Ruben H., "A new Result on the Distribution of Quadratic Forms," Annals of Mathematical Statistics, Vol. 34, pp. 1582–1584, 1963.

[16] Robbins H.E., Pitman E.J.G., "Application of the Method of Mixtures of Quadratic Forms in Normal Variates," Annals of Mathematical Statistics, Vol. 20, pp. 552–560, 1949.

[17] Kotz S., Johnson N.L., Boyd D.W., "Series Representation of Distributions of Quadratic Forms in Normal Variables 1. Central Case," Annals of Mathematical Statistics, Vol. 38, pp. 832–837, 1967.

[18] Pachares J., "Note on the Distribution of a Definite Quadratic Form," Annals of Mathematical Statistics, Vol. 26, pp. 128–131, 1955.

[19] Grenander U., Pollak H.O., Slepian D., "The Distribution of Quadratic Forms in Normal Variates: A Small Sample Theory with Applications to Spectral Analysis," SIAM Journal, Vol. 7, pp. 374–401, 1959.

[20] Siddiqui M.M., "Approximations to the Distribution of Quadratic Forms," Annals of Mathematical Statistics, Vol. 36, pp. 677–682, 1965.

[21] Shah B.K., Khatri C.G., "Distribution of a Definite Quadratic Form for Non-central Normal Variates," Annals of Mathematical Statistics, Vol. 32, pp. 883–887, 1961.

[22] Gurland J., "Distribution of Quadratic Forms and Ratios of Quadratic Forms," The annals of Mathematical Statistics, Vol. 24, pp. 416–427, 1953.

[23] Ruben H., "Probability Content of Regions Under Spherical Normal Distributions 1," Annals of Mathematical Statistics, Vol. 31, pp. 598–619, 1960.

[24] Ruben H., "Probability Content of Regions Under Spherical Normal Distribution iv," The Annals of Mathematical Statistics, Vol. 33, pp. 542–570, 1962.

[25] Kotz S., Johnson N.L., Boyd D.W., "Series Representations of Distributions of quadratic Forms in normal Variables 2. Non-Central Case.," The Annals of Mathematical Statistics, Vol. 38, pp. 838–848, 1967.

[26] Korn G.A., Korn T.M., *Mathematical Handbook for Scientists and Engineers*, New York: McGraw-Hill, 1968.

[27] Franklin J.N., *Matrix Theory*, New Jersey: Prentice-Hall, 1993.