A Novel Receiver Structure for Data Detection in the Presence of Rapidly Changing Nuisance Parameters

Carl R Nassar

McGill University, Montreal

December 1996

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

Abstract

This thesis introduces a novel receiver structure for the detection of data in the presence of rapidly changing nuisance parameters. Underlying equations characterizing the novel receiver are presented first. This is followed by a presentation and explanation of the receiver implementation; the implementation uses a parallel structure to facilitate a real time processing.

A theoretical analysis of this receiver is provided. Here, we introduce an existence condition; this condition suggests the broad applicability of our receiver. Next, we present two algorithms, one based on rate distortion theory, and the other on the Generalized Lloyd Algorithm (GLA). These two algorithms facilitate the creation of the novel receiver's variables for many practical applications.

We apply our receiver to four communication environments of practical interest. These environments can be described briefly as follows: (1) an MPSK signal is sent across a channel introducing noise and a phase offset; here, the phase offset is constant over only N symbols, where N is small but greater than two (e.g., $N = 3$); (2) as in (1), an MPSK signal is sent across a channel introducing noise and a phase offset; this time, the phase offset is constant over only $N = 2$ symbols; (3) a *coded* MPSK signal is sent across a channel adding noise and rapidly changing phase; and, finally, (4) independent data symbols are sent across a channel introducing timing offset and noise; here, the timing offset changes in every received burst of data. We show that, in these environments, our receiver is able to offer gains when compared to the receivers in the current literature; our receiver gains in terms of performance, complexity, or both.

There exists a great potential for future research. The novel receiver introduced in this thesis can be applied to many other communication environments of practical interest.

Résumé

Cette thèse présente un nouveau type de récepteur pour la détection de données en présence de paramètres nuisibles variant de façon rapide. Les équations charactérisant le nouveau récepteur sont présentées d'abord, suivies par la présentation et l'explication de l' implémentation du récepteur. L'implémentation utilise une structure en parallèle afin de faciliter le traitement en temps réel.

Une analyse théorique de ce récepteur est fournie. Une condition d'existence est introduite qui suggère une large application du récepteur. Par ailleurs, deux algorithmes sont présentés, l'un basé sur la théorie de la distortion de débit, l'autre sur l'algorithme généralisé de Lloyd. Ces deux algorithmes facilitent la création des variables du nouveau récepteur en vue d'applications pratiques.

Nous appliquons notre récepteur à quatre environements de communication d'intérêt pratique. Ces environements peuvent être décrits sommairement comme suit : (1) un signal MPSK est transmis sur un canal qui introduit du bruit et un changement rapide de la phase; pour ce cas, on considèrera la phase constante sur $N > 2$ symboles; (2) comme dans le cas (1), un signal MPSK est transmis sur un canal introduisant du bruit et un changement rapide de la phase; cette fois, on considèrera la phase constante sur $N=2$ symboles; (3) un signal MPSK codé est transmis sur un canal ajoutant du bruit et des changements de phase rapides; et finalement (4) des symboles indépendants sont transmis sur un canal introduisant des écarts de synchronisation et du bruit; ici, les écarts de synchronisation changent pour chaque 'burst' de données. Nous démontrons que, dans ces environements, notre récepteur est capable d'offrir des gains en comparaison des récepteur décrits dans la littérature; notre récepteur gagne en terme de performance, complexité ou les deux.

Il existe un grand potentiel pour des recherches futures. Le nouveau récepteur introduit dans cette thèse peut être appliqué à plusieurs autres environements de communication d'intérêt pratique.

Acknowledgements

There are many people who I would like to thank for their love and support throughout this four-year journey.

First there are those who facilitated my academic growth. Dr Reza Soleymani has been both my research supervisor and good friend. A gentle man, Reza always encouraged me to seek my own answers, and facilitated this search with the wisdom of his experience. He has helped me to build a sense of self-confidence and self-trust in my academics that I will carry with me long after my Ph D days.

Thanks also to Dr Maier Blostein, my co-supervisor. Dr Blostein has been a constant source of sound advice and solid support. His ongoing input throughout my Ph D was of immeasurable value.

Finally, I would like to acknowledge the remaining members of my Ph D committee: Dr Harry Leib, thanks for your support; Dr George Zames, thanks for your input; and Dr Peter Kabal, thanks for your kindness.

My Ph D experience was not only an academic journey, but an intimate journey of personal growth as well. Here, again, there are many people who I want to acknowledge for their love and support.

First and foremost, thank you Gretchen Brooks, for being my very best friend - you are my spiritual companion in my walk through the wonder of life. Your support, love, and acceptance have helped me find that same place inside of me. You are an endless source of joy in my life.

Thanks mom, dad, and sister Christine. You have provided me a place called home, where you love me so very much. It is great to know that you are there.

And thanks to the many other people who have been there along the way, who provided me with warm companionship. There are so many of you in the last four years that I can only begin to name you all: David, Sylvie, Denis, Alexi, Daniel, JP, James, Gita, Chris, Judy, Glynnis, Tom, Paul, Angela, Kassandra, Larry, Cecile, Teta, Nana, Nick, Gabby, Monica, Andreas, Andrea, Alok, Ann, Brant, John, Florance, Else, Uwe, Martin, Cornelia, Jules, Peter, and many others.

May you all find peace and joy in your lives. Thank you all for helping me find mine.

Contents

List of Figures

Part I

Introduction

Chapter 1

Introduction

This first chapter introduces the area of interest in this thesis, namely data detection in the presence of nuisance parameters. This is followed by a summary of the original contributions of this thesis.

1.1 Data Detection in the Presence of Nuisance Parameters

In all communication environments, the signal that arrives at the receiver differs from the information-bearing signal output by the transmitter. This is because the transmission medium, also called the channel, introduces some impairments.

A primary impairment introduced by the channel is *noise*. Noise refers to an unwanted signal added, by the channel, to the transmitted signal. A common example of noise is *thermal noise*, a noise caused by the thermal motion of electrons in electronic equipment (e.g., wires and antennas). This noise is modeled as a zero-mean Gaussian random process. Additionally, this noise is modeled as a *white noise*, meaning it has a flat power spectral density, flat for all frequencies.

Receivers detect data in the presence of noise using well known methods such as the commonly used Bayesian criteria. The Bayesian criteria can generate data

symbols that are most likely to match the originally transmitted data.

In addition to noise, channels also introduce other impairments, called *nuisance parameters*. Nuisance parameters refer to a phase offset, a frequency offset, a timing offset, and a channel gain (or attenuation).

A final impairment, which can be viewed as a vector of nuisance parameters, is intersymbol interference (ISI). ISI is described as follows. Channels act as linear filters having a limited bandwidth. This causes the spreading of pulse waveforms that pass through it. Whenever the channel bandwidth is close to the pulse bandwidth, the pulse spreading will exceed a pulse duration, and neighboring pulses will overlap. This overlapping is called intersymbol interference (ISI).

Receivers that detect data in the presence of the nuisance parameters and noise are the topic of this thesis.

1.1.1 Traditional Receivers for Data Detection in the Presence of Nuisance Parameters

Receivers capable of detecting data in the presence of the nuisance parameters and noise have been around a long time. The traditional receivers, usually quite effective for continuous communication, are based on a simple idea: track the nuisance parameters using some tracking circuitry, remove these parameters from the received signal, and then use the optimal Bayesian methods to detect the data in the presence of the noise.

In what follows, we present the traditional tracking circuitry used to follow the nuisance parameters.

Phase Tracking

The circuitry used to track a phase offset is called the Phase Locked Loop (PLL) [1]. The PLL is made up of three basic components, as shown in Figure 1.1.

The multiplier and loop filter generate a difference signal, a signal indicating the

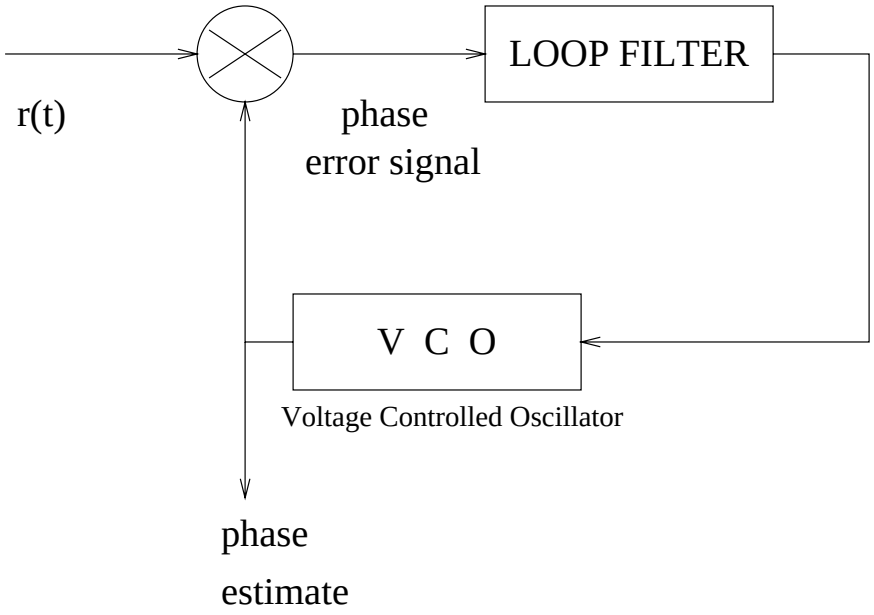

Figure 1.1: A basic phase locked loop (PLL).

difference between the phase to be tracked and the PLL's current estimate of the phase. The voltage controlled oscillator (VCO) updates the PLL's phase estimate using this difference signal [1][2, chapter 8].

Tracking the Timing Offset

There are two traditional schemes for tracking a timing offset.

The first method, deductive timing recovary, is shown in Figure 1.2. Here, a signal compensating for the timing offset (labeled the timing tone) is generated directly from the incoming signal. In many cases, this signal has an unacceptable jitter, and the jitter is reduced using a feedback loop [3, p.561].

The second traditional timing recovary method, inductive timing recovary, is based on a feedback loop much like the PLL. It is shown in Figure 1.3. Several different implementations of this loop exist, including early-late gate timing recovary [3, p.577], sample-derivative timing recovary [3, p.576], and Inphase/Midphase timing recovary [4, p.437].

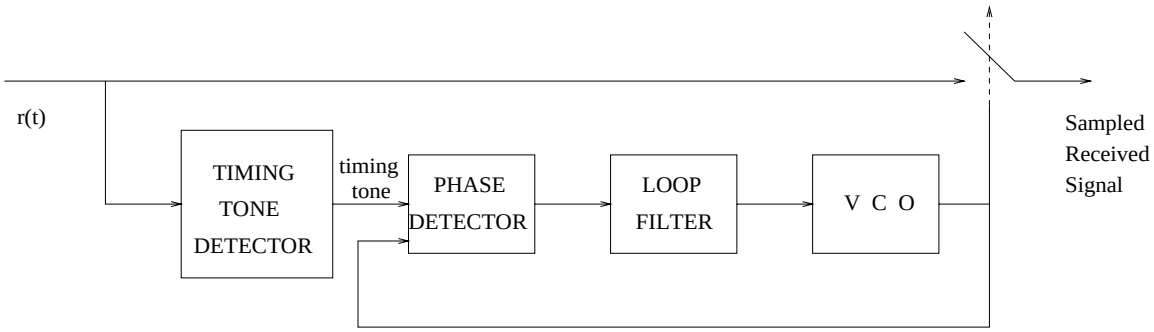

Figure 1.2: Deductive timing recovary.

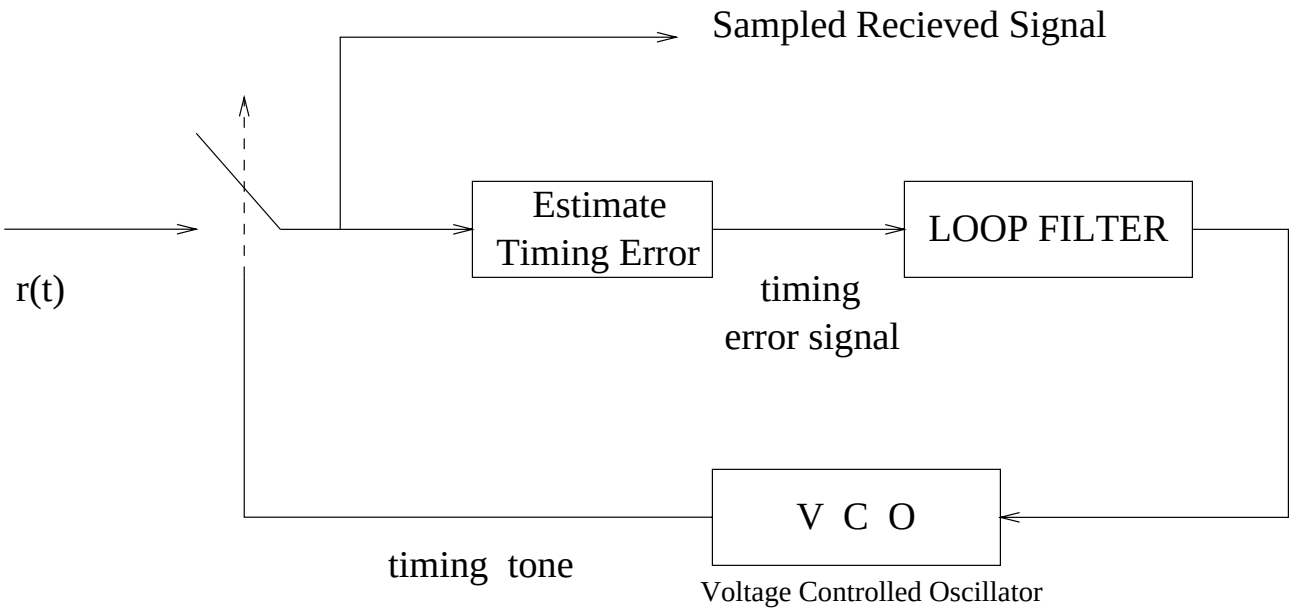

Figure 1.3: Inductive timing recovary.

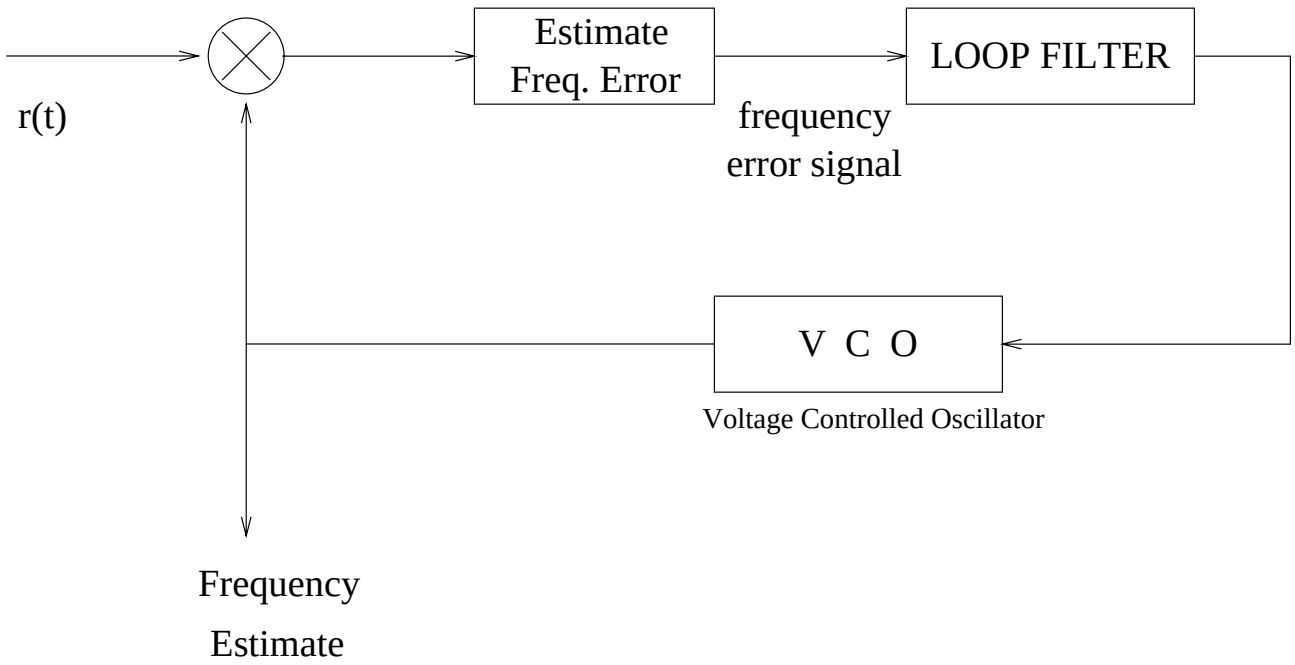

Figure 1.4: Automatic frequency control (AFC) loop.

Tracking the Frequency Offset

A frequency offset is traditionally tracked in one of two ways. First, in cases of small frequency offsets, the PLL used for phase tracking can also follow a frequency offset. Alternatively, in cases of large frequency offsets, an automatic frequency control (AFC) loop is used [5]. This loop, shown in Figure 1.4, demonstrates the same form as the PLL.

Dealing with the ISI

The set of devices designed by communication engineers to remove the effects of ISI in the received signal are collectively referred to as equalizers [6, chapter 6]. Traditionally, equalizers update a set of parameters, using a feedback loop. These parameters allow the equalizer to follow the channel characteristics which cause ISI, and undo their effects.

1.1.2 Limitations of the Traditional Data Detection Methods

The traditional methods of data detection, which first track and then remove the nuisance parameters, facilitate very good data detection in many communication environments of practical interest. For instance, these techniques are used effectively in modern day modems which receive data sent along telephone lines.

However, the rapid expansion of telecommunications has lead to a number of communication environments in which the traditional tracking methods are inappropriate. The key stumbling block limiting the application of traditional methods is easily explained. It sometimes happens that modern transmission environments provide a receiver with a signal containing nuisance parameters which change quickly. In these cases, the traditional methods, which use a feedback loop to track the nuisance parameters, are unable to follow the rapid changes in the nuisance parameters. These schemes become ineffective in such environments.

There are many modern-day examples of communication environments which provide a receiver with quickly changing nuisance parameters.

One example is a communication system employing burst transmission of digital data, such as in time division multiple access (TDMA). TDMA refers to a method of sharing the communication medium. Here, time is split into a number of narrow slots, and each user sends his or her message in bursts of data, with one burst fit into one slot. Only one user may transmit in each slot, and some system agreement is in place to determine who transmits in each slot. TDMA is currently in use in many satellite communication systems and mobile cellular communication systems worldwide.

The reason a TDMA system provides receivers with quickly changing nuisance parameters is as follows. A receiver in a TDMA system, call it A, is only given access to data in time slots addressed to A. As a result, a long interval of time may elapse between the bursts of data which arrive at A. During this time, nuisance parameters may drift, moving far away from their values in the earlier time slot. The nuisance parameters are effectively a uniform random variable at the beginning of each burst entering A. Hence, we have nuisance parameters which can be viewed as quickly changing. One nuisance parameter which behaves in this way is phase —

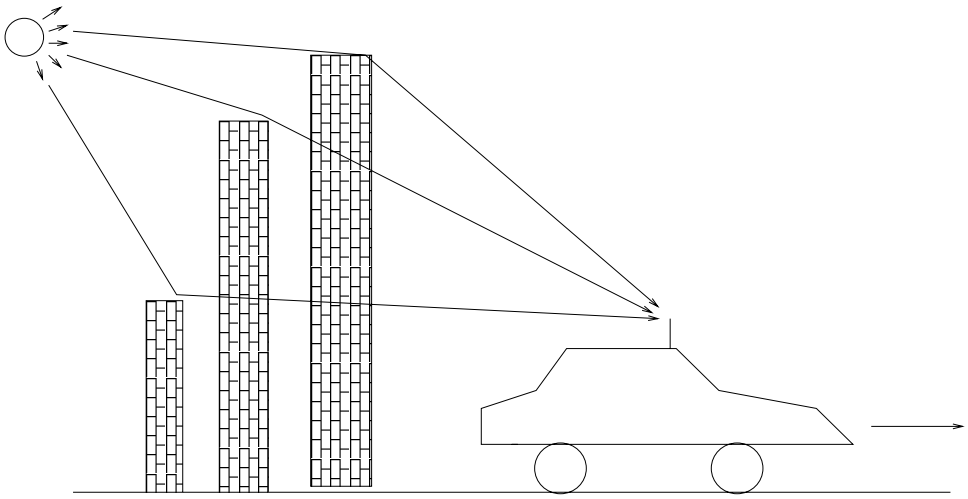

Figure 1.5: The received signal in a mobile communication environment.

small frequency offsets present in TDMA systems result in large phase shifts between received bursts. In this TDMA environment, traditional detection based on a tracking is inefficient.

Another communication system in which receivers experience rapidly changing nuisance parameters is the mobile cellular system. Here, a tower transmitter sends an electromagnetic wave, through the air, to a vehicle. Buildings and houses surrounding the moving vehicle act as natural scatters, reflecting the radio signal. The terrain also reflects the radio signal. Hence, the signal picked up by a receiver mounted on the vehicle is the combination of many reflected versions of the originally transmitted signal, and possibly the original signal itself [7, chapter 1]. This is depicted in Figure 1.5.

The combining of the many reflected signals leads to nuisance parameters in the received signal. These include a phase offset, a channel gain, and possibly ISI. As the receiver moves, the houses and buildings that surround the receiver change, causing the reflected waves arriving at the receiver to demonstrate different properties. They now combine in a different way, leading to a received signal with different characteristics. The nuisance parameters contained in the received signal, such as phase offset, gain, and ISI, are among the characteristics in the received signal that change. These

changes can be very rapid, depending on the speed of the vehicle and the environment surrounding the vehicle. In this mobile environment, as in TDMA, traditional detection based on a tracking is ineffective.

1.1.3 Recent Data Detection Methods

The inability of traditional receivers to detect data when nuisance parameters change rapidly has sparked an interest in finding new receivers. To date, researchers have developed a number of receivers better able to deal with nuisance parameters that are rapidly changing.

In this thesis, data detection in the presence of a rapidly changing phase offset, and data detection in the presence of a rapidly changing timing offset, are of particular interest. In what follows, we briefly introduce these cases of interest, and then survey the recent receivers proposed to detect data in each case. We will see that many of the recent receivers demonstrate a performance degradation relative to detection with known nuisance parameters, or display a high complexity, limiting their practical applicability.

Data Detection in the Presence of a Rapidly Changing Phase

We first consider cases of data detection in the presence of a rapidly changing phase offset. Our interest here can be divided into three important, practical cases, each of which is detailed below.

Phase Constant Over Three (3) or More Symbol Intervals The first case of interest is the detection of differentially encoded MPSK in the presence of additive white Gaussian noise (AWGN) and a phase offset. Here, the phase offset is assumed constant over N symbol intervals ($N > 2$), representing one of two physical possibilities: (1) the phase changes slowly, so slowly that it can be modeled as constant over N symbol intervals, but not necessarily as constant over a longer interval; or, (2) the phase is constant over N symbol intervals, but there exist sudden phase jumps be-

tween blocks of N symbols. Henceforth, we will simply use the phrase phase *constant over N symbol intervals* to denote these two possibilities. The many receivers that have recently been proposed for use in this environment are detailed below.

One of the first receivers proposed for this phase offset environment was conventional differential detection, also called differential PSK, or DPSK for short [8]. Here, the phase reference for a symbol is simply the previously received symbol. Consequently, DPSK can be used whenever the phase is constant over two or more symbols, a very mild constraint. Unfortunately, DPSK experiences a substantial performance degradation – up to 3 dB may be lost when compared to coherent detection. This performance degradation comes about because the phase reference, namely the previous received symbol, is noisy.

The performance degradation experienced by DPSK led researchers to search for alternative receivers for the phase offset environment. Researchers came up with receivers which offered coherent-like performances, but only when the phase was constant over a mid-sized block of symbols, e.g., 20 or more symbols.

The first researchers to develop such a receiver were A.J. Viterbi and A.M. Viterbi [9]. They proposed a receiver wherein a feedforward carrier phase estimator tracked the channel phase. Their receiver demonstrates a performance comparable to coherent detection, as long as phase remains constant over a long enough duration, e.g. $N > 20$.

Another receiver, providing effective data detection only when phase is constant over a long interval (e.g. $N = 20$), is based on the application of the Expectation Maximization (EM) algorithm [10][11]. One drawback of this method is its iterative nature, which may make it impractical in some communication environments. Also, there exists a degenerate case which can only be avoided at a cost of doubling the receiver complexity [12].

Finally, some additional receivers have recently been proposed, again for the detection of data when phase is constant over a long block of symbols (e.g., [13]). Here, coherent-like performances are demonstrated whenever channel phase remains constant over 20 or more symbols.

The inability of the above receivers to offer effective data detection when phase

changes rapidly ($N < 20$), and DPSK's inability to achieve near-coherent performances, lead to the development of still more receiver structures.

One receiver, intended to deal with phase jitter in telephone lines, was able to detect data with a rapidly changing channel phase [14]. However, this receiver required a long preamble to get started, and, in addition, it employed a complexity that rendered it impractical [15].

Other receivers that detect data when phase is constant over only a few symbols were based on heuristic arguments [16]-[21]. They resulted in some performance gains over DPSK, but their performance remained far from coherent.

Recently, four groups of researchers independently generated a receiver, for the phase offset communication environment, by extending the ideas of DPSK [22]-[25]. Their receiver is commonly referred to as Multiple Symbol Differential Detection (MSDD). The performance of MSDD is far superior to that of DPSK, even with phase constant over as few as three symbols. As the number of symbols with a constant phase increases from three, the performance of their receiver tends rapidly toward coherent. In fact, the researchers show that the performance achieved by their receiver is optimal, optimal in the sense of minimizing symbol error rate given an unknown channel phase over a block of N received symbols. However, a drawback of this scheme is complexity. The complexity of their receiver increases exponentially as N increases. Specifically, the complexity of MSDD, in terms of computation per decoded symbol, is in the order of $M^N * N$ [23]. This limits the applicability of MSDD.

The quality performance but high complexity of MSDD lead researchers to search for a low complexity version of MSDD. In [26]-[28], a receiver was proposed that reduced the complexity of MSDD by using symbol feedback. However, this receiver experienced decision feedback errors, especially costly in fading channels during periods of deep fade (i.e., low SNR), because at these times the feedback error effect can lead to a burst of errors.

Mackenthun [29] succeeded in finding an implementation of MSDD which demonstrates the same optimal performance, while maintaining a lower complexity – here, the complexity increases linearly with block size N, rather than exponentially.

The above receiver schemes describe the recent receivers available for the detection of MPSK in the presence of a phase offset constant over three or more symbols.

Phase Constant Over Two (2) Symbol Intervals The second case of particular interest in this thesis is the detection of differentially encoded MPSK in the presence of AWGN and a phase offset that changes at the following rate: phase change over two symbol intervals is negligible, but it is not necessarily negligible over a longer interval. The recent receivers currently available to detect data in the presence of this channel phase are described below.

The previous literature survey introduced the many receivers available for data detection in the presence of a channel phase constant over $N \geq 2$ symbols. The literature survey for the case at hand, then, can be achieved by simply re-introducing some of these receivers; from among the previous list, we simply re-introduce those receivers that are able to provide data detection when channel phase is constant over $N \approx 2$ symbol intervals.

The first receiver, from among the receivers listed previously, that can detect data with channel phase constant over only two symbol intervals, is DPSK [8]. However, DPSK experiences a substantial performance degradation when compared to coherent – up to 3 dB may be lost.

The second receiver, listed in the previous literature survey, that can deal with a phase constant over only two symbol intervals, is the receiver provided in [14]. However, as mentioned earlier, this receiver suffers from two major drawbacks: it requires a preamble of fifty symbols, and its implementation demonstrates a complexity that renders it impractical [15].

Another receiver provided in the earlier literature survey, also of interest to us in this case, is MSDD [22]-[25], or alternatively, a scheme performing as well as MSDD without its high complexity (e.g., [29]). These receivers are of interest because they easily outperform DPSK, and their assumption on phase (phase is constant over N, e.g, $N = 3$, symbols) is not too different from a phase constant over only two symbols. Unfortunately, in the case at hand, the performance of MSDD decreases rapidly as

the rate of phase change increases.

The above three receivers represent the receivers currently available to detect data in the presence of a phase offset constant over only two symbol intervals. These receivers are unable to provide near-coherent performances at a reasonable complexity.

Coded Modulation, Phase Constant over Three (3) or More Symbols The third case of interest is the detection of coded MPSK (trellis coded modulation employing an MPSK modulation format) in the presence of AWGN and a phase offset constant over a short block of N symbols ($N > 2$). In what follows, we present the recently-introduced receivers available to detect data in this environment.

One of the first receivers proposed to detect coded MPSK symbols, in the presence of rapidly changing channel phase, was introduced in [30][31]. This receiver corresponds to an extension of the ideas of DPSK, and, hence, we call it Coded DPSK. The major drawback of Coded DPSK is its substantial performance degradation when compared to coherent detection; all the performance gain achieved by coded MPSK (when compared to uncoded MPSK) is lost.

A few years later, the same researchers introduced a receiver corresponding to an improved version of Coded DPSK [32]. However, this receiver was only able to achieve moderate gains over Coded DPSK, and these gains could only be achieved with certain codes.

More recently, a group of researchers proposed a novel receiver, for coded MPSK detection when a rapidly changing phase is present, based on an extension of MSDD to coded PSK [33]. We will call this receiver Coded MSDD. The major drawback of Coded MSDD is that it uses Multiple Trellis Coded Modulation (MTCM) [34] of multiplicity $k = N - 1$, and, as a result, its complexity grows quickly with increasing N. This high complexity limits the applicability of Coded MSDD.

Another group of researchers [35] introduced a receiver which applied MSDD to coded MPSK in a unique fashion; their receiver avoided the high complexity found in Coded MSDD. We will call their receiver low complexity Coded MSDD, or LC Coded MSDD. LC Coded MSDD comes with some drawbacks. First, this receiver employs

a detection algorithm that differs from the standard Viterbi Algorithm decoder, increasing its implementation cost. Furthermore, considering a rate $\frac{2}{3}$ Coded 8-PSK scheme, LC Coded MSDD can only be employed for $N \geq 30$, and with $N \approx 30$, this receiver experiences performance degradations of 1 dB or more.

Finally, many receivers have been proposed for data detection when N is large ($N > 500$) [13][36][37]. These receivers are able to achieve near-coherent performances for $N > 500$, but, unfortunately, their performances degrade rapidly at smaller N values.

The above list of receivers represent the receivers available to date to detect coded MPSK in the presence of a phase offset constant over N symbol intervals. These receivers are unable to provide near-coherent performances, at a reasonable complexity, for small N values (i.e., N values less than 30).

Data Detection in the Presence of a Rapidly Changing Timing Offset

This thesis also emphasizes cases of data detection in the presence of a rapidly changing timing offset. In particular, we are interested in the detection of independent data symbols in the presence of AWGN and a timing offset constant over a burst of N symbols. The following is a list of receivers recently introduced to detect data in this case.

Most of the current receivers employed for data detection in the presence of noise and a timing offset (changing from burst to burst) are based on a simple approach. First, *feedforward* tracking circuitry is used to remove the timing offset; then, the data is detected in the presence of the noise alone using well-known Bayesian methods. Some of the analog tracking circuits that are currently available can be found in [3, chapter 15][38].

One feedforward tracking circuit, presented by Oerder and Meyer [39], has recently gained widespread popularity. The popularity of this circuit is due in large part to its digital nature – the received signal is sampled by a free running oscillator and all the timing offset recovary is then done digitally. However, this tracking method has its drawbacks. First, as the roll-off factor α decreases, the performance of the tracking

circuit diminishes; in fact, when α is close to 0, a receiver using this tracking circuit is unable to achieve reliable data detection. Another drawback of this tracking circuit is that its performance diminishes substantially as burst lengths decrease.

Recently, a group of researchers devised a tracking circuit [40] which offered performance gains over the popular circuit described above. These gains are most notable at small values of α. Despite these gains, their tracking scheme still suffers from the same drawbacks that plagued its predecessor.

Receivers employing the above tracking circuits represent the receivers available to date to detect data in the presence of a timing offset that changes over each burst of N symbols. These receivers are unable to provide reliable data detection with roll off factors near 0, and at small burst lengths (i.e., small values of N).

1.2 The Contributions of this Thesis

The contributions of this thesis can be considered in two parts. First, this thesis introduces a novel receiver structure for the detection of data in the presence of rapidly changing nuisance parameters, a receiver that demonstrates widespread applicability. Second, this thesis presents four important, practical applications of the receiver structure, highlighting its gains when compared to the receivers in the current literature. In what follows, we provide more details regarding these contributions.

1.2.1 A Novel Receiver Structure

In 1963, Harman [41, p.271] noted in passing that a *parallel* receiver structure (specifically, a receiver employing a bank of detectors in parallel) can be applied when nuisance parameters are present in the received signal. However, at that time, Harman believed that the complexity of such a receiver would be extremely large; in his words, 'comparable to the complexity of a modern digital computer or telephone exchange.'

Recently, the work of Madhow and Pursley [42]-[45] showed that a *parallel* receiver

structure can act as a practical data detector for detection with unknown parameters. However, their work considered only *memoryless* channels. As a result, their receiver required the use of a bandwidth-expanding channel coding (a Reed-Solomon coding) to select from among parallel demodulators.

More recently, *parallel* receiver structures were proposed for data detection in the presence of nuisance parameters constant over a long sequence of symbols [46][47]. Here, each demodulator, in a bank of parallel demodulators, performs a sequence long data detection, and a decision between demodulators is made based on the knowledge that the nuisance parameters are constant.

This thesis introduces a novel, *parallel* receiver structure for the detection of data in the presence of rapidly changing nuisance parameters. Here, unlike the work of [46][47], the nuisance parameters are modeled as rapidly changing; in addition, unlike [42]-[45], where memoryless parameters are assumed, the memory inherent in the channel's quickly changing nuisance parameters is key to our receiver design. Specifically, the channel memory is used in our structure to select from among parallel demodulators – avoiding the need for a channel coding. We note that the works of [46][47] can be viewed as special applications of our proposed receiver structure (with some minor updates). These works correspond to two cases where the memory of the nuisance parameters translates to constant nuisance parameters over a long sequence of symbols.

Our novel parallel receiver structure is derived using Maximum *A Posteriori* arguments, a form of Bayesian Inference long known to lead to optimal data detection. The parallel nature of our receiver structure facilitates its rapid real-time processing.

The key theoretical contributions of this thesis are the two design algorithms we introduce for our receiver structure. The first, based on rate distortion theory, generates a bound on the minimum number of parallel demodulators required by our receiver structure to achieve a stated performance. The second algorithm, based on the Generalized Lloyd Algorithm, creates an optimal set of values for use in the parallel demodulator structure.

1.2.2 Applications of the Novel Receiver Structure

We apply our novel receiver structure to four (4) communication environments of practical interest, namely the four environments described in the literature surveys of Subsection 1.1.3.

In the first application, we apply our receiver structure to detect differentially encoded MPSK in the presence of AWGN and a channel phase constant over N symbols, $N > 2$ (e.g., $N = 3$). Here, our receiver demonstrates a performance matching theoretically optimal bounds, while maintaining a low complexity. Compared to receivers in the current literature, only two receivers are capable of achieving a comparable performance, namely MSDD and Mackenthun's low complexity version of MSDD; however, our receiver is available at a lower complexity (substantially lower in the case of MSDD).

Our second application is the detection of differentially encoded MPSK in the presence of AWGN and a rapidly changing phase; here, the phase change over two symbol intervals is negligible, but it is not necessarily negligible over a longer interval. We show that the receiver we introduce in this application is very beneficial. It is preferable to DPSK because it outperforms DPSK by 1.5 dB; preferred to MSDD because, except under very slow phase change conditions, it easily outperforms MSDD; and, finally, it is preferred to the scheme of [14] because it demonstrates a realizable complexity and avoids the need for a long preamble.

Our third application is the detection of coded MPSK in the presence of a phase offset constant over N symbols, $N > 2$. Here, our receiver structure demonstrates many gains. Most notably, our receiver displays a near-coherent performance, and a realizable complexity, when phase is constant over as few as 10 symbol intervals. No other receiver, to date, is able to achieve this.

Finally, in our fourth application, data detection in the presence of a timing offset changing from burst to burst, our receiver structure again demonstrates many gains when compared to the receivers in the current literature. Specifically, whenever the roll-off factor is close to zero (i.e., *sinc*-like pulse shapes are transmitted), or whenever the burst length is short, our receiver easily outperforms those in recent literature.

Furthermore, our receiver is easily realized.

1.3 Claims to Originality

This section summerizes, in point form, all the claims to originality of this thesis.

- Three novel data detection equations are derived starting from theoretically optimal arguments (Chapter 3 of this thesis, or [48]-[50]).
- A novel, parallel receiver structure is introduced which can implement any of the three new data detection equations (Chapter 3, [48]-[50]).
- It is proven that the novel receiver structure can be applied to most cases of data detection in the presence of noise and nuisance parameters (Chapter 4, [50]).
- A theoretical analysis, establishing the performance of our receiver *versus* the number of parallel demodulators in its implementation, is provided (Chapter 4, [50]).
- An algorithm is presented that generates a bound on the smallest number of parallel demodulators that can be used in our receiver's implementation, when it is required that our receiver structure achieve a stated performance (Chapter 4, [50]).
- An algorithm is introduced that generates the undetermined variables in the receiver structure. This algorithm generates the variables in a manner that optimizes the receiver's performance and complexity (Chapter 4, [50]).
- The novel receiver structure is applied to data detection in the presence of a phase offset constant over N symbols, $N > 2$. The receiver that results demonstrates a theoretically optimal performance at a low complexity, and offers gains compared to receivers in current literature (Chapter 5, [51][52]).

- The novel receiver structure is applied to detect MPSK in the presence of a phase offset that changes quickly; the phase offset's change over two symbol intervals is negligible, but its change is not necessarily negligible over a longer interval. Here, our receiver easily outperforms the many receivers proposed to date; specifically, our receiver gains 1.5 dB when compared to DPSK, gaining back half of the 3 dB degradation experienced by DPSK; furthermore, our receiver outperforms MSDD (and its low complexity implementations) by a significant amount, with its gain over these schemes increasing as the rate of phase change increases. Our receiver achieves these gains while maintaining a low (easy-to-implement) complexity (Chapter 6, [49][53][54]).

- The novel receiver is applied to detect trellis coded modulation (in particular coded MPSK) in the presence of a phase offset constant over N symbols. The receiver that results is able to outperform those introduced in the current literature. Specifically, for values of N as low as 10, our receiver is able to achieve a near-coherent performance, while maintaining an easily-realized implementation; other receiver schemes, at N values between 10 and 30, either experience a very significant performance degradation relative to coherent, or demonstrate an unmanageably high complexity (Chapter 7,[55][56]).

- Finally, the receiver structure is applied to detect data symbols in the presence of a timing offset that changes from burst to burst. In the important, practical cases of small roll-off factors or short burst durations, our receiver demonstrates performances close to coherent (within 0.5 dB); receivers in the current literature, on the other hand, are unable to achieve reliable data detection in these same cases. Additionally, our receiver is available at an easily realizable complexity. (Chapter 8,[57][58]).

1.4 Thesis Outline

The remainder of this thesis is presented in the following order. Chapter 2 summerizes the necessary theoretical background. Here, we begin by presenting a general

mathematical model of the communication environment under consideration. This is followed by an explanation of the theory underlying data detection.

Part 2 of this thesis, containing Chapters 3 and 4, introduces the novel receiver at the heart of this thesis. Chapter 3 gets things started with a derivation of our novel receiver's underlying equations. Chapter 3 also introduces the implementation of our novel receiver. Chapter 4 provides a theoretical analysis of the novel receiver, and includes design algorithms which allow an engineer to apply this receiver to most data detection scenarios involving nuisance parameters.

Part 3 of this thesis, consisting of Chapters 5 to 8, introduces some of the many applications of our novel receiver structure. In each chapter, we present a different application of practical interest. These chapters follow a consistent (and uniform) format. Specifically, each chapter provides a detailed modeling of the communication environment of interest; presents the application of our receiver structure; and compares our receiver with the receivers in the literature — here, we indicate the benefits of our receiver.

Part 4 concludes the thesis. It begins with a summary of the work presented in the thesis, highlighting its contributions, and ends with a discussion of possible work for the future.

Chapter 2

Background

In this chapter, we present a model of the communication environment of interest in this thesis. We also present the theory underlying receiver detection and estimation. With this in hand, we can go on in future chapters to present our novel receiver.

2.1 The Communication Environment Model

This section introduces the communication environment for which our novel receiver is intended. First, we introduce a model, and then we impose some conditions on this model.

2.1.1 The Model

The communication environment model is shown in Figure 2.1.

The term *source* refers to both the *physical source* and the *source encoder*. The *physical source* outputs a signal, $y(t)$, which may represent a voice, video, or data signal. This signal is characterized as a random process - that is, the source output is a sample function from a random process ensemble.

The *source encoder* maps the physical source output, $y(t)$, into a sequence (in

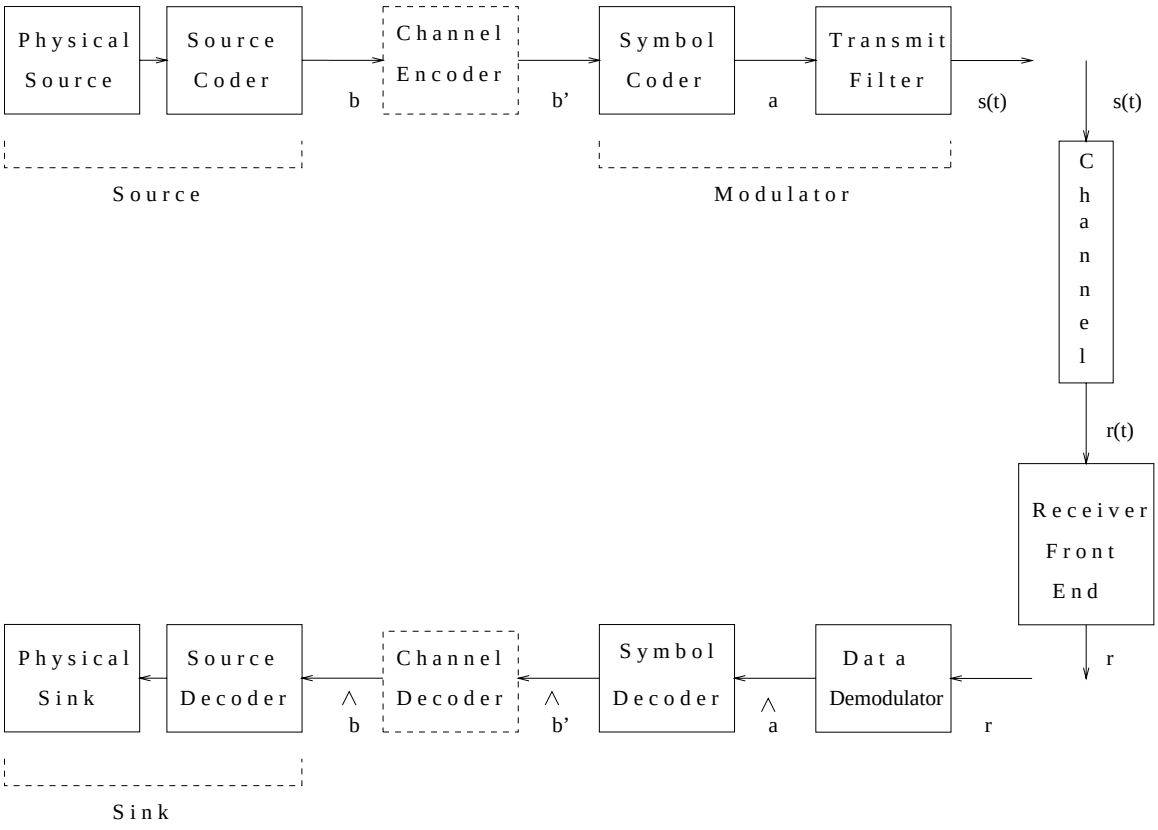

Figure 2.1: The communication system model.

time) of binary digits. This may be done by sampling the waveform $y(t)$, quantizing the samples, and then encoding the quantized samples using binary digits. The waveform $y(t)$ can not, in general, be precisely reconstructed from the binary digits. The source encoder is usually designed to output the fewest binary digits per unit time and still adequately represent the physical source's output. The sequence (in time) of binary digits created by the source encoder is

$$\underline{b} = (b_1, b_2, ..., b_X), \tag{2.1}$$

where $b_i \in B = \{0, 1\}$. Throughout this thesis, the subscript i will be used as a discrete time index.

The *channel encoder* maps the input sequence of bits, $\underline{b}$, into an output sequence of bits, called $\underline{b}'$. This mapping is carefully chosen; it adds redundancy to help the receiver reconstruct the original bit sequence $\underline{b}$. The channel encoder, unlike other components, is an optional part of the communication system model. Its presence has been introduced for completeness, and not out of necessity.

The *symbol coder* maps the input bit sequence into an output sequence of symbols. It generates these symbols at a fixed rate of one symbol every T seconds. Each symbol is a selection from the fixed alphabet $A = \{a^j; j = 1, 2, ..., M\}$. (Throughout this thesis, the superscript j will be used to indicate the j^{th} selection in a discrete set.) The sequence (in time) of output symbols generated by the symbol coder is represented by

$$\underline{a} = (a_1, a_2, ..., a_L), \tag{2.2}$$

where $a_i \in A$. This symbol coder mapping may represent, for instance, a one-to-one mapping of each n bits into one of the $M = 2^n$ symbols $\{e^{j\frac{2\pi}{M}}, e^{j\frac{2\pi}{M}\cdot 2}, ..., e^{j\frac{2\pi}{M}\cdot M}\}$.

The *transmit filter* transforms each symbol from the symbol coder into a waveform which is ready for transmission over the physical channel. The data modulator output, in response to input a_i, is labeled $s_i(t)$. Whenever $a_i = a^j$, $s_i(t)$ corresponds to the j^{th} waveform from the fixed set of waveforms $S = \{s^1(t), s^2(t), ..., s^M(t)\}$, shifted by iT seconds; that is $s_i(t) = s^j(t - iT)$. The entire waveform output by the transmit

filter, in response to input $\underline{a}$, is

$$s(t) = \sum_{i=1}^{L} s_i(t). \tag{2.3}$$

The *physical channel* maps the input waveform $s(t)$ into a random process $r(t)$. The mapping function generating $r(t)$ is described by

$$r(t) = w(s(t), \underline{c}(t)) + \eta(t); \tag{2.4}$$

here, $\underline{c}(t)$ is a set of random processes modeling the nuisance parameters introduced by the channel; $\eta(t)$ is a single random process representing the channel's noise; and $w(\cdot, \cdot)$ is a known mapping rule.

The *receiver* attempts to recreate the binary sequence $\underline{b}$ from the received signal $r(t)$. The receiver is usually subdivided into four parts: a *receiver front end*, a *data demodulator*, a *symbol decoder*, and a *channel decoder*. (The source decoder is included in the sink.)

The *receiver front end* maps the continuos-time random process $r(t)$ into a discrete time random process $\underline{r} = (r_1, r_2, ..., r_L)$. Here, each r_i is characterized by

$$r_i = v(\underline{a}_i, \underline{c}_i) + \eta_i; \tag{2.5}$$

η_i is a random value generated from the channel noise process $\eta(t)$; $\underline{c}_i$ is a set of random values $(c_{i,1}, c_{i,2}, ..., c_{i,I})$ generated from the nuisance parameter random processes $\underline{c}(t)$; and, finally, $\underline{a}_i$ is a subsequence of $\underline{a}$ which includes a_i. Oftentimes, $\underline{a}_i$ equals a_i.

The output sequence $\underline{r}$ is generated at a rate of one sample every T seconds. The receiver front end is carefully designed to insure that the sequence $\underline{r}$ contains all the information needed to make an optimal decision on the binary sequence $\underline{b}$. As a result, $\underline{r}$ is called a sufficient statistic for detection.

The *data demodulator* maps the receiver front end output, $\underline{r}$, into an estimate of the symbol sequence $\underline{a}$. This estimate is labeled $\hat{\underline{a}}$. The *symbol decoder* takes the estimate $\hat{\underline{a}}$ and regenerates the transmitted binary data $\underline{b}'$; this output is labeled $\hat{\underline{b}}'$. Finally, the *channel decoder* maps $\hat{\underline{b}}'$ into an estimate of the source bits, $\hat{\underline{b}}$. In some

communication environments, optimal receiver performance requires carrying out the operations of the data demodulator, symbol decoder, and channel decoder in a single block. In these cases, these devices will be put together in a single block, also called a *data demodulator*. To avoid confusion, unless explicitly indicated otherwise, the term data demodulator will refer to a device separate from the symbol decoder and channel decoder.

2.1.2 Conditions Assumed to be in Effect

This subsection introduces three conditions on the communication model which we assume to be in effect. All the receivers we are aware of, that are intended for data detection in the presence of nuisance parameters, are applicable only when the first two of the three conditions introduced here are satisfied. Our receiver structure will require that, in addition, the third condition is satisfied. We want to point out here that the conditions we introduce are not very restrictive. In fact, in most communication environments, with nuisance parameters of phase, timing, frequency, and ISI, the conditions we introduce in this section are satisfied.

The first restriction we impose on the communication model relates to the discrete-time random sequence $\underline{r}$, a sufficient statistic for detection. It must be possible to generate this sequence $\underline{r}$ such that the component function $v(\underline{a}_i, \underline{c}_i)$ is a one-to-one function of $\underline{a}_i$. This condition insures that, in the absence of noise (η_i), the symbol sequence $\underline{a}$ can be resolved, and hence the binary source output regenerated.

Our second restriction on the communication environment is also on the discrete-time random process $\underline{r}$. This restriction states that, if a data demodulator receives $\underline{r}$, and carries out the optimal detection scheme to generate $\hat{\underline{a}}$, this scheme is not independent of the nuisance parameter sequence $\underline{c}$. The mathematical expression representing this condition is provided in the next section, where optimal data detection methods are explained. This condition insures that the communication environment does not simplify the detection problem at the receiver to the point where the unknown parameters can, for purposes of optimal detection, be ignored.

Our final restriction on the communication environment again relates to the suf-

ficient statistic $\underline{r}$. We require either: (1) for cases of *dependent* noise samples, a characterization of $P(\epsilon|\underline{c},\underline{c}')$, the probability of error of a receiver, with input $\underline{r}$, that assumes $\underline{c}' = (\underline{c}'_1, \underline{c}'_2, ..., \underline{c}'_L)$ is the nuisance parameter vector when in reality $\underline{c} = (\underline{c}_1, \underline{c}_2, ..., \underline{c}_L)$ is in effect; or, (2) for cases of *independent* noise samples, a characterization of $P(\epsilon|\underline{c}_i,\underline{c}'_i)$, the probability of error of a receiver, with inputs r_i, assuming $\underline{c}'_i$ when in reality $\underline{c}_i$ is in effect. Either function is required in later chapters to verify the applicability of our proposed receiver structure, to establish complexity, and to generate unknown variables. If a complete mathematical characterization of $P(\epsilon|\underline{c},\underline{c}')$ or $P(\epsilon|\underline{c}_i,\underline{c}'_i)$ is unattainable, it usually suffices to characterize $P(\epsilon|\underline{c},\underline{c}')$ / $P(\epsilon|\underline{c}_i,\underline{c}'_i)$ using a function proportional to $P(\epsilon|\underline{c},\underline{c}')$ / $P(\epsilon|\underline{c}_i,\underline{c}'_i)$ (e.g., Section 7.2), or using the $P(\epsilon|\underline{c},\underline{c}')$ / $P(\epsilon|\underline{c}_i,\underline{c}'_i)$ of a simpler, but closely related, communication environment (e.g., Section 8.2).

2.2 Optimal Data Detection and Parameter Estimation

This section introduces theoretically optimal methods for a receiver to detect data and estimate nuisance parameter values.

2.2.1 Optimal Data Detection

In this subsection, we present an equation characterizing a receiver that carries out optimal data detection.

We begin by explaining what we mean by optimal data detection. A receiver carrying out optimal data detection refers to a receiver generating the data sequence $\hat{\underline{b}}$ that is most likely to match $\underline{b}$. Alternatively, we are referring to a receiver that generates the sequence $\hat{\underline{a}}$ most likely to match $\underline{a}$. (We can re-express the $\hat{\underline{b}}$ matching $\underline{b}$ as $\hat{\underline{a}}$ matching $\underline{a}$ because, in all communication environments we are aware of, these are directly proportional [2, p.180-181].) That is, a receiver implementing optimal

data detection refers to one creating the sequence $\underline{\hat{a}}$ that minimizes the risk

$$R = \sum_{\underline{a}} \sum_{\underline{\hat{a}}, \underline{\hat{a}} \neq \underline{a}} P(\underline{a}) \cdot Pr(say\ \underline{\hat{a}}\,|\,\underline{a}\ is\ true). \tag{2.6}$$

A receiver that minimizes the risk R corresponds to, after some computation [59, chapter 2], a receiver that implements

$$\underline{\hat{a}} = \arg\max_{\underline{a}} p(\underline{a}|\underline{r}). \tag{2.7}$$

That is, a receiver that minimizes the probability of a difference between $\underline{a}$ and $\underline{\hat{a}}$ computes the *a posteriori* probabilities $p(\underline{a}|\underline{r})$ for each $\underline{a}$, and outputs the $\underline{\hat{a}}$ corresponding to the $\underline{a}$ with the largest $p(\underline{a}|\underline{r})$. Receivers based on equation (2.7) are called maximum *a posteriori* (MAP) receivers.

2.2.2 Optimal Parameter Estimation

This section introduces two equations characterizing a receiver that generates $\underline{\hat{c}}$, an estimate of the nuisance parameter sequence $\underline{c} = (\underline{c}_1, ..., \underline{c}_L)$, from the received $\underline{r}$.

Oftentimes, we want a receiver to generate a $\underline{\hat{c}}$ that minimizes the average of the *squared error* cost function, i.e., the cost function

$$C(\underline{c}, \underline{\hat{c}}) = (\underline{c} - \underline{\hat{c}}) \cdot (\underline{c} - \underline{\hat{c}})^T. \tag{2.8}$$

A receiver can achieve this by carrying out [59, chapter 2]

$$\underline{\hat{c}}(\underline{r}) = \int_C \underline{c} \cdot p(\underline{c}|\underline{r}) d\underline{c}; \tag{2.9}$$

that is, the receiver generates the conditional mean of $\underline{c}$.

Other times, we want a receiver that generates $\underline{\hat{c}}$ minimizing the average of the *uniform* cost function, a cost function described by: whenever the Euclidian distance between $\underline{c}$ and $\underline{\hat{c}}$ is less than some small Δ, the cost is 0; otherwise, the cost is 1. In this case, the receiver should be built to carry out [59, chapter 2]

$$\underline{\hat{c}}(\underline{r}) = \arg\max_{\underline{c}} p(\underline{c}|\underline{r}); \tag{2.10}$$

that is, the receiver should generate the sequence which maximizes the *a posteriori* density. This receiver's estimate is called the MAP estimate. In many instances [59, chapter 2], the estimates generated by (2.9) and (2.10) are identical.

2.2.3 Joint Data Detection and Parameter Estimation

In this subsection, we present an equation characterizing a receiver that jointly detects the data sequence $\underline{a}$ and estimates the nuisance parameter sequence $\underline{c}$, given the received $\underline{r}$. The goal of this receiver is to generate the data sequence $\hat{\underline{a}}$ most likely to match $\underline{a}$, and create an estimate $\hat{\underline{c}}$ which minimizes the average of the uniform cost function.

The equation characterizing a receiver that generates the above $\hat{\underline{a}}$ and $\hat{\underline{c}}$ can be achieved as follows. First, restate the data detection as a data estimation. Specifically, it is easily shown that the data detection operation is equivalent to: estimate the value $\hat{\underline{a}}$ that minimizes the uniform cost function, where, here, the usually discrete $\underline{a}$ now corresponds to a continuous random variable with distribution $p(\underline{a}) = \sum_{\underline{a}^j} P(\underline{a}^j)\delta(\underline{a} - \underline{a}^j)$; the term $\delta(\underline{a} - \underline{a}^j)$ refers to $\delta(a_1 - a_1^j) \cdot ... \cdot \delta(a_L - a_L^j)$, where $\underline{a}^j = (a_1^j, ..., a_L^j)$ is the j^{th} possible data sequence, and $\delta(x)$ is the delta function, a function of value 0 whenever $x \neq 0$ and displaying unit area. With data detection now restated as the above data estimation, the joint detection of $\underline{a}$ and estimation of $\underline{c}$ corresponds simply to finding the joint estimate $(\hat{\underline{a}}, \hat{\underline{c}})$ which minimizes the average of the uniform cost function. Using the result of (2.10), the receiver should be built to carry out

$$\hat{\underline{a}}, \hat{\underline{c}} = \arg \max_{\underline{a}, \underline{c}} p(\underline{a}, \underline{c}|\underline{r}). \tag{2.11}$$

In (2.11), it is easily shown that $\underline{a}$ can again be interpreted as a discrete random variable with distribution $P(\underline{a})$. The detected data and estimated parameter sequence generated by equation (2.11) maximize the joint *a posteriori* density. Consequently, a receiver implementing (2.11) is called the joint MAP receiver.

2.2.4 Optimal Data Detection *vs* Joint Detection and Estimation

The received sequence $\underline{r}$ in the model of Section 2.1 contains both a data sequence $\underline{a}$ and nuisance parameter sequence $\underline{c}$. Hence, a receiver which generates the data symbols $\hat{\underline{a}}$ can be constructed using one of two possible starting equations: the optimal data detection of (2.7) or the joint data detection and parameter estimation of (2.11).

First, the receiver can be constructed using the principles of optimal data detection; this receiver is built to implement the MAP equation

$$\hat{\underline{a}} = \arg\max_{\underline{a}} p(\underline{a}|\underline{r}). \tag{2.12}$$

A receiver implementing (2.12) can be constructed in one of two ways. Unfortunately, as we will now explain, it is very hard to achieve a reasonable complexity in the case at hand.

First, we can try to build a receiver implementing (2.12) by expressing (2.12) in a form containing $p(\underline{r}|\underline{a},\underline{c})$, a known distribution, and hope that this suggests a realizable receiver structure. This is done as follows: applying $p(\underline{a}|\underline{r}) = \frac{p(\underline{r}|\underline{a})P(\underline{a})}{p(\underline{r})}$ in (2.12), and observing that the denominator is independent of $\underline{a}$, leads to

$$\hat{\underline{a}} = \arg\max_{\underline{a}} p(\underline{r}|\underline{a}) \cdot P(\underline{a}). \tag{2.13}$$

Introducing the nuisance parameter sequence $\underline{c}$, this receiver equation becomes

$$\hat{\underline{a}} = \arg\max_{\underline{a}} \int_C p(\underline{r}|\underline{a},\underline{c}) \cdot p(\underline{c}) \cdot P(\underline{a}) d\underline{c}. \tag{2.14}$$

The integral nature of this equation makes it hard to achieve a general receiver implementation at a reasonable complexity. For instance, one way to attempt to build a receiver using (2.14) is based on the use of numerical techniques to evaluate the integral. However, this implementation requires the evaluation of a numerical integral for each of the M^L possible sequences $\underline{a}$, an unreasonable complexity.

An alternative way to attempt to build a receiver based on (2.12) is the following. It has been shown in [60] that a receiver employing an estimator-correlator structure can almost always achieve optimal data detection, where optimality can be defined in the sense of minimizing the likelihood of error. Hence, with (2.12) corresponding to an equation that minimizes error likelihoods, we could consider the possibility of implementing (2.12) using an estimator-correlator structure. Unfortunately, two problems arise. First, the estimator-correlator structure requires generating an estimate that is often hard to find [60]. This problem may be resolved by using a suboptimal estimator (e.g. [61]), but no general form of such an estimator is available. Furthermore, in our case, where sequence long detection is of interest, the number of estimators

and correlators required increases exponentially as L increases, and is generally in the order of M^L. This is a prohibitive complexity.

A second starting point for a receiver detecting data in the presence of nuisance parameters is the joint MAP equation of (2.11). A receiver built from this starting point does not guarantee the optimal data sequence $\underline{\hat{a}}$ as output; instead, it balances optimal data detection with optimal estimation of the nuisance parameter sequence. Because the goal of a receiver is to achieve optimal data detection, this receiver can be considered suboptimal. However, the benefit of such a receiver is that it is based on a much easier to implement equation: the integral in (2.14) is replaced by the largest term of the integral in the joint MAP equation (2.11).

Furthermore, as we now point out, in many cases of practical interest, the joint MAP equation results in data detection which is indistinguishable from optimal data detection. Consider the following special case of practical interest: the data sequences $\underline{a}$ are equally likely; the nuisance parameters $\underline{c} = (\underline{c}_1, ..., \underline{c}_L)$ are modeled as time invariant, i.e., $\underline{c}_i = \underline{c}_k \ , \ \forall \ i, k$; the distribution of $\underline{c}_i$ is uniform; and the noise is Gaussian. In this case, the integral equation of (2.14) corresponds to $\underline{\hat{a}} = \arg\max_{\underline{\hat{a}}} \int_{C_0} p(\underline{r}|\underline{a}, \underline{c}_1) \cdot p(\underline{c}_1) d\underline{c}_1$. Now, with $p(\underline{c}_1)$ corresponding to a uniform distribution, and the noise distribution Gaussian, the dominant contribution to this integral comes from the region in the parameter space C_0 ($\underline{c}_1 \in C_0$) that maximizes the distribution $p(\underline{r}|\underline{a}, \underline{c}_1)$. It is therefore almost optimal data detection to base a decision on equation (2.11), the joint MAP detection equation, which, in this case, can be expressed as $\underline{\hat{a}} = \arg\max_{\underline{\hat{a}}}[\max_{\underline{c}_1} p(\underline{r}|\underline{a}, \underline{c}_1) \cdot p(\underline{c}_1)]$. A more detailed explanation, as well as further evidence of the closeness of joint MAP and optimal detection, can be found in [62, p.291].

2.2.5 A Mathematical Statement of the Second Condition in Effect

In subsection 2.1.2, we introduced three conditions on the communication system model of interest in this thesis. The second condition stated: optimal data detection is not independent of the nuisance parameter sequence $\underline{c}$. With the equation for optimal

data detection now in place, this section introduces the mathematical equivalent to the second constraint.

Equation (2.14) is the equation for optimal data detection when the nuisance parameter sequence $\underline{c}$ is present. The requirement, then, is that the right-hand side (RHS) of (2.14) is not independent of $\underline{c}$. In cases where the optimal data detection equation is well approximated by the joint MAP equation, the requirement can be expressed as : $\hat{\underline{a}} = \arg\max_{\underline{a}} p(\underline{a}, \underline{c}|\underline{r})$ is not independent of $\underline{c}$.

Part II

The Novel Parallel Receiver Structure: Theory

Chapter 3

The Parallel Receiver Structure: Underlying Equations and Implementation

In this chapter, we introduce a novel receiver structure. We do this in two parts. First, we present equations which characterize the operation of a novel receiver structure. Second, we present the corresponding implementation of the receiver structure.

3.1 The Underlying Equations

This section introduces three novel data detection equations, each characterizing a data demodulator detecting data in the presence of nuisance parameters. The first equation we present is a general detection equation, an equation which can be applied to any communication environment modeled by Section 2.1. Next, we present an equation useful for detection when the noise and data samples are both independent. Finally, we present an equation for detection whenever noise samples are independent, and data samples display state dependence.

3.1.1 General Data Detection Equation

The first receiver equation we derive is called the general data detection equation. It imposes no constraints on the communication system model of Section 2.1.

The derivation of this equation is as follows. A data demodulator tries to achieve optimal data detection; that is, given $\underline{r}$, it attempts to generate an $\hat{\underline{a}}$ which is most likely to match $\underline{a}$. This is achieved by generating $\hat{\underline{a}}$ from

$$\hat{\underline{a}} = \arg \max_{\underline{a}} p(\underline{a}|\underline{r}). \tag{3.1}$$

With nuisance parameters present, this equation corresponds to integral equation (2.14). This does not suggest a realizable implementation.

As pointed out in Chapter 2, a data demodulator can instead carry out jointly optimal data detection and parameter estimation. Such a data demodulator is more easily implemented. Furthermore, it oftentimes corresponds closely to optimal data detection. For these reasons, we restart the derivation of our data detection equation with the joint MAP equation

$$\hat{\underline{a}}, \hat{\underline{c}} = \arg \max_{\underline{a},\underline{c}} p(\underline{a}, \underline{c}|\underline{r}). \tag{3.2}$$

We are merely interested in generating the data sequence $\hat{\underline{a}}$ at the data demodulator; the $\hat{\underline{c}}$ is a by-product. To emphasize this, we rewrite the equation for data detection carried out by the demodulator as: choose the sequence $\hat{\underline{a}}$ which results from the joint maximization

$$\max_{\underline{a},\underline{c}} p(\underline{a}, \underline{c}|\underline{r}). \tag{3.3}$$

We now work this maximization to a more convenient form using simple statistical and mathematical arguments. This begins with the application of a statistical rule, namely Bayes Rule, which allows the maximization to be rewritten according to

$$\max_{\underline{a},\underline{c}} p(\underline{r}|\underline{a}, \underline{c}) \cdot p(\underline{a}, \underline{c}). \tag{3.4}$$

Next, we recognize that the sequence $\underline{a}$, generated by the symbol coder, and the sequence $\underline{c}$, generated by the physical channel, are independent random processes. This allows the maximization to be expressed according to

$$\max_{\underline{a},\underline{c}} p(\underline{r}|\underline{a}, \underline{c}) \cdot P(\underline{a}) \cdot p(\underline{c}). \tag{3.5}$$

The distribution $P(\underline{a})$ represents the *a priori* knowledge of the source and the channel encoder. However, the statistics at the source are, in general, not known. This lack of knowledge can be expressed statistically by assuming all messages $\underline{a}$ are equally likely; that is, $P(\underline{a})$ is a constant. Applying this to the maximization leads to

$$\max_{\underline{a},\underline{c}} p(\underline{r}|\underline{a},\underline{c}) \cdot p(\underline{c}). \tag{3.6}$$

Because logarithms are monotonic functions, this maximization can be rewritten as

$$\max_{\underline{a},\underline{c}} \{\ln\, p(\underline{r}|\underline{a},\underline{c}) + \ln\, p(\underline{c})\}. \tag{3.7}$$

Reordering the maximizations leads to

$$\max_{\underline{c}}\{[\max_{\underline{a}} \ln\, p(\underline{r}|\underline{a},\underline{c})] + \ln\, p(\underline{c})\}. \tag{3.8}$$

The first term in this equation, a joint maximization, corresponds to choosing the sequences $\underline{c}$ and $\underline{a}$ that are most likely based on the received sequence and the distribution of the noise; the second term, a maximization over $\underline{c}$, corresponds to choosing the $\underline{c}$ that is most likely given the known distribution $p(\underline{c})$. The overall equation corresponds to choosing an $\underline{a}$ and $\underline{c}$ that is a balance between these two choices.

The final equation, which sets up the novel parallel receiver implementation introduced in this thesis, is achieved by introducing an approximation to equation (3.8). The continuous nuisance parameter space, C ($\underline{c} \in C$), is approximated by a discrete nuisance parameter space, $\tilde{C} = \{\underline{c}^1, \underline{c}^2, ..., \underline{c}^m\}$. This discrete nuisance parameter space is a subset of the continuous nuisance parameter space, i.e., $\tilde{C} \subset C$.

The approximation of C by $\tilde{C}$ in equation (3.8) is a very good one, provided we can insure that the data detection achieved assuming $\tilde{C}$ is very close to that achieved using C. A more complete description of when the approximation of C by $\tilde{C}$ is valid, and algorithms for establishing the discrete space $\tilde{C}$, are the topic of the next chapter. For the time being, we assume that this approximation is a good one, and return to completing the derivation of the key equation. Applying the discrete space approximation to equation (3.8) leads to: choose the sequence $\hat{\underline{a}}$ which results from the joint maximization

$$\max_{\tilde{\underline{c}}}\{[\max_{\underline{a}} \ln\, p(\underline{r}|\underline{a},\tilde{\underline{c}})] + \ln\, P(\tilde{\underline{c}})\}, \tag{3.9}$$

where $\underline{\tilde{c}}$ refers to an element in the discrete parameter space $\tilde{C}$. Throughout this thesis, the notation $\tilde{}$ will be used to refer to the discrete domain.

Equation (3.9) corresponds to the first of three equations on which the novel receiver implementation is based.

3.1.2 Data Detection Equations for Cases of Independent Noise Samples

In many practical communication environments, the noise samples (η_i) contained in the receiver front end output $(\underline{r})$ are statistically independent; that is, $p(\eta_i, \eta_j) = p(\eta_i) \cdot p(\eta_j)\,, \forall\, i \neq j$. This modeling is very common in a wide variety of modern day communication environments. For instance, many satellite, mobile, and telephone line communication systems model the received noise samples (η_i) as independent.

The most common situation leading to independent η_i's is the following. Whenever the noise $\eta(t)$ is white and Gaussian, the output of the matched filter contained in the receiver front end satisfies Nyquist criteria, and the optimal sampling times are known to the receiver front end, the noise samples η_i are independent. Other situations may also lead to independent η_i's, but these are less common. For example, the receiver front end may employ a whitening filter.

Whenever the noise samples η_i are independent, equation (3.9) can be replaced by simpler equations. The new equations are generated as follows. We begin here with equation (3.8), which states that sequence $\underline{\hat{a}}$ should be selected from the maximization

$$\max_{\underline{c}}\{[\max_{\underline{a}} \ln p(\underline{r}|\underline{a},\underline{c})] + \ln p(\underline{c})\}. \tag{3.10}$$

The first probability in this equation, $p(\underline{r}|\underline{a},\underline{c})$, corresponds to the distribution of the noise sequence evaluated at the value $\underline{r} - \underline{v}(\underline{a},\underline{c})$; that is,

$$p(\underline{r}|\underline{a},\underline{c}) = p_{\underline{\eta}}(\underline{r} - \underline{v}(\underline{a},\underline{c})), \tag{3.11}$$

where $\underline{v}(\underline{a},\underline{c})$ refers to the sequence $(v(\underline{a}_1,\underline{c}_1), v(\underline{a}_2,\underline{c}_2), ..., v(\underline{a}_L,\underline{c}_L))$. Substituting equation (3.11) into (3.10) leads to

$$\max_{\underline{c}}\{[\max_{\underline{a}} \ln p_{\underline{\eta}}(\underline{r} - v(\underline{a},\underline{c}))] + \ln p(\underline{c})\}. \tag{3.12}$$

The noise samples are independent, that is, $p(\underline{\eta}) = p(\eta_1) \cdot p(\eta_2) \cdot ... \cdot p(\eta_L)$. Applying this to (3.12), we find

$$\max_{\underline{c}}\{[\max_{\underline{a}} \ln\, p_{\eta_1}(r_1 - v(\underline{a}_1, \underline{c}_1)) + \ln\, p_{\eta_2}(r_2 - v(\underline{a}_2, \underline{c}_2)) + ... + \ln\, p_{\eta_L}(r_L - v(\underline{a}_L, \underline{c}_L))] + \ln\, p(\underline{c})\}, \tag{3.13}$$

or, in short hand notation,

$$\max_{\underline{c}}\{[\max_{\underline{a}} \sum_{i=1}^{L} \ln\, p_{\eta_i}(r_i - v(\underline{a}_i, \underline{c}_i))] + \ln\, p(\underline{c})\}, \tag{3.14}$$

or, equivalently,

$$\max_{\underline{c}}\{[\max_{\underline{a}} \sum_{i=1}^{L} \ln\, p(r_i|\underline{a}_i, \underline{c}_i)] + \ln\, p(\underline{c})\}. \tag{3.15}$$

The term $p(\underline{c})$ can be expressed, by repetitive application of the statistical rule $p(A, B) = p(A|B)p(B)$, according to

$$p(\underline{c}) = \prod_{i=1}^{L} p(\underline{c}_i|\underline{c}_{i-1}, ..., \underline{c}_1). \tag{3.16}$$

Furthermore, the statistical dependence of $\underline{c}_i$ on $\underline{c}_j$ diminishes as the time interval between i and j increases. Consequently, equation (3.16) is well approximated by

$$p(\underline{c}) = \prod_{i=1}^{L} p(\underline{c}_i|\underline{c}_{i-1}, ..., \underline{c}_{i-J}), \tag{3.17}$$

where J is the value beyond which the additional statistical dependence is negligible. Substituting equation (3.17) into equation (3.15) leads to

$$\max_{\underline{c}}\{[\max_{\underline{a}} \sum_{i=1}^{L} \ln\, p(r_i|\underline{a}_i, \underline{c}_i)] + \sum_{i=1}^{L} \ln\, p(\underline{c}_i|\underline{c}_{i-1}, ..., \underline{c}_{i-J})\}. \tag{3.18}$$

We now make a key approximation. We approximate C_0, the space of nuisance parameter vectors ($\underline{c}_i$'s), by a discrete space $\tilde{C}_0 = \{\underline{c}^1, \underline{c}^2, ..., \underline{c}^m\}$; the notation $\underline{c}^j$ refers here to a vector in the space $\tilde{C}_0$ (rather than in $\tilde{C}$). This approximation is valid whenever the data detected assuming $\tilde{C}_0$ is very close to the data detected using the complete space C_0. The conditions under which this is true, and methods for generating $\tilde{C}_0$, are the subject of the next chapter. For the time being, we assume this approximation is valid and return to deriving the key equation. We apply the

discrete space approximation to equation (3.18), which leads to: choose the $\hat{\underline{a}}$ which results from the joint maximization

$$\max_{\underline{\tilde{c}} \in \tilde{C}_0^L} \{ [\max_{\underline{a}} \sum_{i=1}^{L} \ln\, p(r_i | \underline{a}_i, \underline{\tilde{c}}_i)] + \sum_{i=1}^{L} \ln\, P(\underline{\tilde{c}}_i | \underline{\tilde{c}}_{i-1}, ..., \underline{\tilde{c}}_{i-J}) \}, \tag{3.19}$$

where $\underline{\tilde{c}}_i$ refers to an element in the discrete space $\tilde{C}_0$. (In future chapters, we will drop the subscript and refer to $\tilde{C}_0$ as $\tilde{C}$.)

Equation (3.19) represents the general form of the data detection equation when noise samples are independent. This general form usually takes on one of two simpler forms, described next.

Case Independent Data Samples

In many practical situations, r_i's dependence on the sequence $\underline{a}_i$ corresponds to a dependence on the single, independent value a_i. In this case, the data detection equation of (3.19) reduces to: choose the $\hat{\underline{a}}$ from the joint maximization

$$\max_{\underline{\tilde{c}} \in \tilde{C}_0^L} \sum_{i=1}^{L} \{ [\max_{a_i} \ln\, p(r_i | a_i, \underline{\tilde{c}}_i)] + \ln\, P(\underline{\tilde{c}}_i | \underline{\tilde{c}}_{i-1}, ..., \underline{\tilde{c}}_{i-J}) \}. \tag{3.20}$$

This equation suggests that part of the data detection can be carried out using symbol-by-symbol detection. At each time i, for each $\underline{\tilde{c}}_i \in \tilde{C}_0$, we can carry out a symbol-by-symbol detection — this leads to m possible symbols at each time. This resolves the inner optimization of equation (3.20). We can then find the best data symbol at each time by carrying out a sequence long maximization over $\underline{\tilde{c}} \in \tilde{C}_0^L$.

Case Data Symbols Have a State Dependance

In many practical situations, the data symbols a_i demonstrate a state dependence. By this, we mean that r_i's dependence on $\underline{a}_i$ can be expressed as a dependence on two variables: a_i, the transmitted symbol at time i, and $S_i \in \{S^1, S^2, ..., S^V\}$, called the state at time i. The S_i summerizes the dependence of r_i on a_i's past and future. Some examples of r_i's containing such an $\underline{a}_i$ are found in communication systems using trellis coded modulation, or experiencing ISI.

In this case, the data detection of equation (3.19) can be rewritten according to: choose the $\underline{\hat{a}}$ from the joint maximization

$$\max_{\underline{\tilde{c}} \in \tilde{C}_0^L} \max_{\underline{S}} \sum_{i=1}^{L} \{[\max_{a_i} \ln\, p(r_i|a_i, \underline{\tilde{c}}_i)|_{S_i}] + \ln\, P(\underline{\tilde{c}}_i|\underline{\tilde{c}}_{i-1}, ..., \underline{\tilde{c}}_{i-J})\}; \tag{3.21}$$

here, the notation $\underline{S}$ refers to a state sequence $(S_1, S_2, ..., S_L)$, and $\max_{\underline{S}}$ refers to the maximization over the set of permissible state sequences. Also, the dependence of r_i on S_i is not explicitly indicated, i.e., we write $p(r_i|a_i, \underline{\tilde{c}}_i)$ rather than $p(r_i|a_i, S_i, \underline{\tilde{c}}_i)$. This equation states that the data detection can be carried out, in part, on a symbol-by-symbol level, generating one symbol for each possible state and each possible discrete nuisance parameter. This resolves the innermost maximization of (3.21). The final symbols are achieved by sequence maximization over the possible state sequences and over the nuisance parameter sequence.

3.2 Implementation

This section introduces a novel parallel receiver structure based on the three data detection equations (3.9), (3.20), and (3.21). The receiver structure presented can be used to implement any of these three key detection equations. The description of the processing at each receiver component depends on which equation is being implemented.

This section begins with a general presentation of the parallel receiver structure. This is followed by two subsections which describe the receiver components in detail. The first of these subsections explains the components when the receiver structure is used to carry out the general data detection equation - equation (3.9). A second subsection describes the components when the receiver implements the data detection equations for cases of independent noise - equations (3.20) and (3.21).

3.2.1 General Receiver Structure

Figure 3.1 shows the general receiver structure. This receiver can be used to carry out any one of the key data detection equations of Section 3.1.

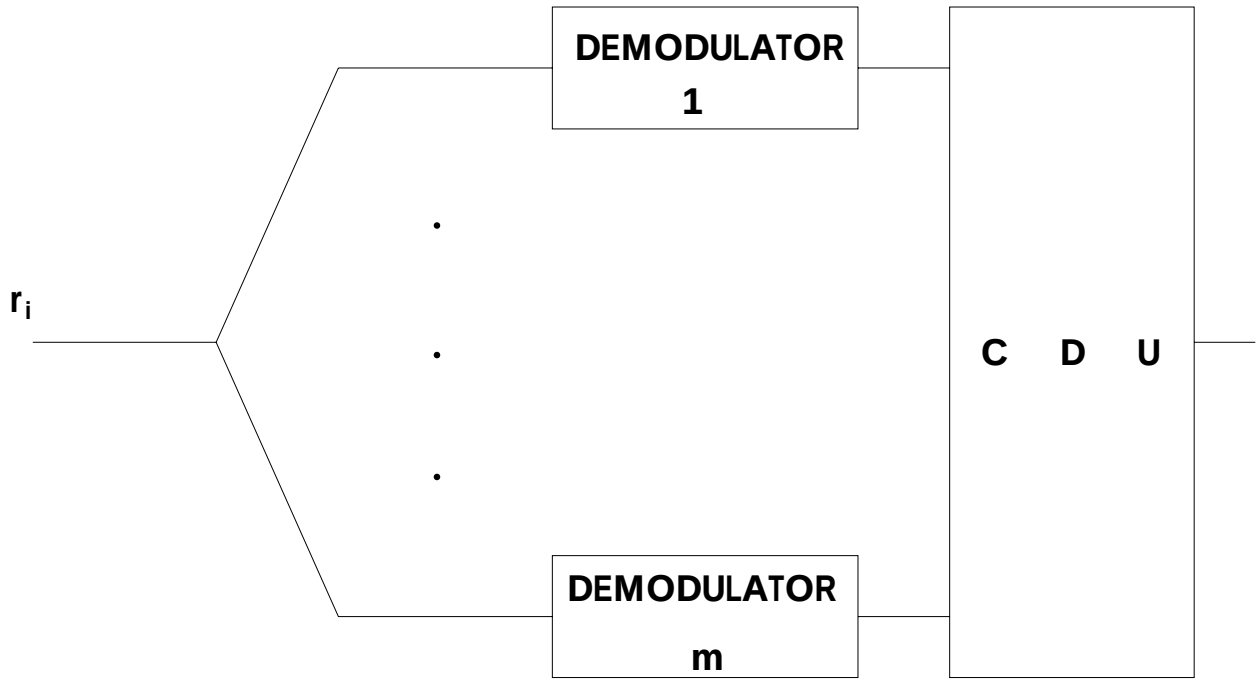

Figure 3.1: The general receiver structure.

The receiver consists of two basic parts: a bank of m demodulators which we call the *universal set of demodulators*, and a processing unit called the *computation and decision unit* (CDU). The *universal set of demodulators* carry out a maximization over the data sequence $\underline{a}$, the inner maximization in the joint optimization equations (3.9), (3.20), and (3.21). The *computation and decision unit* (CDU) carries out the optimizing over $\underline{\tilde{c}}$ (and possibly $\underline{S}$), the outer maximization of these three equations.

3.2.2 The Components in the General Data Detection Equation

This section describes the components of the receiver structure in Figure 3.1, when the equation being implemented corresponds to the general data detection equation - equation (3.9).

The Universal Set of Demodulators

The *universal set of demodulators* carry out the inner maximization in equation (3.9); that is, they carry out $\max_{\underline{a}} p(\underline{r}|\underline{a}, \underline{\tilde{c}})$, outputting both the resulting maximum value and the corresponding data sequence. Each demodulator performs this maximization using a different $\underline{\tilde{c}} \in \tilde{C}$, for a total of m demodulators. A more detailed description of the demodulators follows.

Each of the m demodulators receives, every T seconds, the sample r_i from the receiver front end. The j^{th} demodulator assumes that the sequence of nuisance parameter values contained in $\underline{r}$, namely $\underline{c} = (\underline{c}_1, ..., \underline{c}_L)$, is given by $\underline{c}^j$. The j^{th} demodulator waits until all the r_i's arrive before generating its output. Upon their arrival, the j^{th} demodulator outputs its choice of $\underline{a}$, based on $\underline{r}$ and the assumption that the nuisance parameter values are $\underline{c}^j$. This output sequence is called $\hat{\underline{a}}^j$; it is given by

$$\hat{\underline{a}}^j = \arg\max_{\underline{a}} p(\underline{r}|\underline{a}, \underline{c}^j). \tag{3.22}$$

This sequence is sent to the CDU accompanied by a value which indicates its likelihood:

$$l^j = \max_{\underline{a}} p(\underline{r}|\underline{a}, \underline{c}^j) = p(\underline{r}|\hat{\underline{a}}^j, \underline{c}^j). \tag{3.23}$$

In summary, it can be said that the j^{th} demodulator carries out optimal, *maximum likelihood* (ML) sequence detection assuming $\underline{c}^j$ is the nuisance parameter sequence in effect.

The CDU

The *CDU* carries out the outer maximization of equation (3.9); that is, it implements the maximization over the nuisance parameter sequence. Its operation can be described, in terms of the demodulator outputs, as follows.

The CDU decides which one of the m demodulators is best, and selects its output as the system output. The CDU's decision is based on both the likelihood of the demodulators' decisions and the nuisance parameter statistics. The output of the CDU is given by

$$\hat{\underline{a}} = \hat{\underline{a}}^{j^*}, j^* = \arg\max_j [\ln(l^j) + \ln P(\underline{c}^j)]. \tag{3.24}$$

3.2.3 The Components in Cases of Independent Noise Samples

This section describes the components of the receiver in Figure 3.1 in the cases when the noise samples η_i are independent. In these cases, the receiver corresponds to an implementation of either equation (3.20) or (3.21).

The Universal Set of Demodulators

The *universal set of demodulators* carry out the inner maximization of equation (3.20) or (3.21), a maximization over a_i. The operation of the demodulators, when they implement (3.20), is slightly different from their operation when they carry out (3.21). In what follows, we first provide a paragraph describing the operation of the demodulators that is common to both cases. We then provide two subsections which detail the demodulators' operation: one corresponding to the implementation of (3.20), the other for (3.21).

Regardless of whether the demodulators implement equation (3.20) or (3.21), each of the m demodulators receive, every T seconds, the sample r_i. The j^{th} demodulator assumes that $\underline{c}_1 = \underline{c}_2 = ... = \underline{c}_L = \underline{c}^j$, i.e., $\underline{c}_i = \underline{c}^j$. (The notation $\underline{c}^j$ now refers to an assumption on $\underline{c}_i$, rather than an assumption on the entire sequence $\underline{c} = (\underline{c}_1, ..., \underline{c}_L)$.) The j^{th} demodulator, using r_i, and this claim of $\underline{c}_i = \underline{c}^j$, generates, every T seconds, one or more decisions regarding a_i, the transmitted symbol.

Case Independent Data Symbols Whenever r_i's dependence on $\underline{a}_i$ corresponds to a dependence on the single, independent value a_i, the demodulators implement the inner maximization of equation (3.20). Here, the j^{th} demodulator generates a single decision on a_i, using r_i and the assumption $\underline{c}_i = \underline{c}^j$. This is called $\hat{a}_i^j$. It is generated according to

$$\hat{a}_i^j = \arg\max_{a_i} p(r_i|\underline{c}^j, a_i). \tag{3.25}$$

The j^{th} demodulator also generates l_i^j, a value indicating the likelihood of $\hat{a}_i^j$, according to

$$l_i^j = \max_{a_i} p(r_i|\underline{c}^j, a_i) = p(r_i|\underline{c}^j, \hat{a}_i^j). \tag{3.26}$$

Both $\hat{a}_i^j$ and l_i^j are sent to the CDU. The entire sequence of outputs generated by the j^{th} demodulator (and sent to the CDU) is, simply, $\underline{\hat{a}}^j = (\hat{a}_1^j, ..., \hat{a}_L^j)$ and $\underline{l}^j = (l_1^j, ..., l_L^j)$.

In summary, in the case when $\underline{a}_i$ corresponds to an independent a_i, the j^{th} demodulator carries out optimal *maximum likelihood* (ML) symbol-by-symbol detection assuming the nuisance parameters in effect, at any time i, correspond to $\underline{c}^j$.

Case Data Symbols Have a State Dependence Sometimes r_i's dependence on $\underline{a}_i$ does not correspond simply to a dependence on an independent a_i, but rather its dependence can be expressed using two terms - the single value a_i and the state value S_i. In these cases the demodulators implement the inner maximization of equation (3.21). Here, the j^{th} demodulator generates many decisions on a_i, using r_i and the claim that $\underline{c}_i = \underline{c}^j$. It creates one decision for each of the V possible values of the state S_i. Let $\{\hat{a}_{i,k}^j, k = 1, 2, ..., V\}$ denote the V demodulator decisions generated by the j^{th} demodulator at time i; here, the decision $\hat{a}_{i,k}^j$ corresponds to the decision when $S_i = S^k$. These decisions are generated according to

$$\hat{a}_{i,k}^j = \arg\max_{a_i} p(r_i|\underline{c}^j, a_i)|_{S_i = S^k}. \tag{3.27}$$

The dependence of r_i on S_i is not written explicitly in equation (3.27); instead, this dependence is implicit. Corresponding values, indicating the likelihood of the demodulator's decisions, are generated according to

$$l_{i,k}^j = \max_{a_i} p(r_i|\underline{c}^j, a_i)|_{S_i = S^k} = p(r_i|\underline{c}^j, \hat{a}_{i,k}^j). \tag{3.28}$$

These are sent, along with $\{\hat{a}_{i,k}^j\}$, to the CDU. The entire sequence of outputs generated by the j^{th} demodulator and sent to the CDU corresponds to $\underline{\hat{a}}^j = \{\hat{a}_{i,k}^j,\ i = 1, 2, ..., L,\ k = 1, 2, ..., V\}$ and $\underline{l}^j = \{l_{i,k}^j,\ i = 1, 2, ..., L,\ k = 1, 2, ..., V\}$.

The CDU

The *CDU* implements the outer maximization of equation (3.20) or (3.21); that is, the CDU implements the maximization over the discrete nuisance parameter sequence $\underline{\tilde{c}}$ (and possibly $\underline{S}$). The CDU outputs the data sequence that results from performing this maximization. With the values of the inner maximization, and the corresponding data sequences, available from the universal set of demodulators, the CDU's operation can be described as follows.

The CDU decides which of the demodulator decisions to output. At one sample time, the CDU may decide to output a decision received from demodulator A, and at the next time the CDU may choose a decision received from demodulator B. Overall, the CDU's output data sequence is a mixing of the decisions from the m demodulators. The CDU generates its output only after a consideration of the likelihood of the demodulators' decisions and the statistical characterization of the nuisance parameter sequence.

Case Independent Data Symbols

Whenever r_i's dependence on $\underline{a}_i$ corresponds to a dependence on an independent a_i, the data sequence output by the CDU is selected based on the outer maximization of equation (3.20). The CDU's output is expressed here in terms of the decisions generated by the bank of demodulators.

The $\underline{\hat{a}}$ generated by the CDU can be expressed according to

$$\underline{\hat{a}} = (\hat{a}_1, \hat{a}_2, ..., \hat{a}_L) \text{ where } \hat{a}_i = \hat{a}_i^{j_i^*}; \tag{3.29}$$

here, $j_i^* \in \{1, 2, ..., m\}$ corresponds to the index of the demodulator whose output is selected by the CDU at time i. These j_i^* values are generated according to

$$\underline{j}^* = (j_1^*, j_2^*, ..., j_L^*) = \arg\max_{\underline{j}} \sum_{i=1}^{L} \{\ln(l_i^{j_i}) + \ln P(\underline{c}^{j_i} | \underline{c}^{j_{i-1}}, ..., \underline{c}^{j_{i-J}})\}. \tag{3.30}$$

That is, the j_i^*'s are chosen based on two considerations: the likelihood of the demodulator outputs, and the probability that a demodulator is chosen given the channel parameters assumed by the previously selected demodulators.

Case Data Symbols Have a State Dependence When r_i's dependence on $\underline{a}_i$ can be expressed as a dependence on a_i and a state variable S_i, the CDU's output data sequence is based on the outer maximizations of equation (3.21). We express the CDU's output here in terms of the inputs provided by the bank of demodulators.

The set of decisions $\{\hat{a}^j_{i,k},\ k = 1, ..., V\}$ are received from each demodulator at each time i, along with a corresponding set of likelihood values. The CDU output $\underline{\hat{a}}$ is chosen according to

$$\underline{\hat{a}} = (\hat{a}_1, \hat{a}_2, ..., \hat{a}_L) \text{ where } \hat{a}_i = a^{j_i^*}_{i,k_i^*}; \tag{3.31}$$

here, $j_i^* \in \{1, 2, ..., m\}$ again corresponds to the index of the demodulator whose output is chosen by the CDU at time i; similarly, $k_i^* \in \{1, 2, ..., V\}$ corresponds to the index of the state whose output is selected by the CDU at time i. These k_i^* and j_i^* values are selected according to

$$\underline{k}^*, \underline{j}^* = (k_1^*, ..., k_L^*, j_1^*, ..., j_L^*) = \arg \max_{\underline{j}, \underline{k} \in K_P} \sum_{i=1}^{L} \{\ln(l^{j_i}_{i,k_i}) + \ln P(\underline{c}^{j_i} | \underline{c}^{j_{i-1}}, ..., \underline{c}^{j_{i-J}})\}, \tag{3.32}$$

where K_P represents the set of indices corresponding to all permissible state sequences. This equation in simply an alternative expression for the outer maximizations of equation (3.21).

3.2.4 The Components — Example

In this section we provide a description of the components in the general receiver structure for an example which is of practical interest. In this example, the received signal r_i is described as

$$r_i = v(a_i, c_i) + \eta_i. \tag{3.33}$$

Here, η_i is assumed independent of its history and future; similarly, a_i is independent of $a_j\ \forall\, i \neq j$; and, finally, c_i represents a single nuisance parameter value.

The receiver components here will correspond to a special case of the receiver components which carry out equation (3.20) — the only difference is $\underline{c}_i$ is replaced by the single value c_i.

The *universal set of demodulators* operate, in this example, as follows. The j^{th} demodulator assumes that the sample c_i, present in r_i, has a value of c^j. It assumes that this is the correct parameter value regardless of the time index i. This j^{th} demodulator generates, every T seconds, the decision

$$\hat{a}_i^j = \arg\max_{a_i} p(r_i|a_i, c^j). \tag{3.34}$$

This equation corresponds to the *maximum likelihood* decision on a_i when c_i is assumed to be c^j. The demodulator transmits its decision to the CDU, accompanied by the likelihood value

$$l_i^j = \max_{a_i} p(r_i|a_i, c^j) = p(r_i|\hat{a}_i^j, c^j); \tag{3.35}$$

this indicates the likelihood of $\hat{a}_i^j$.

The CDU determines which of the demodulator decisions to select as an output at each time. The decisions it outputs are chosen according to

$$\underline{\hat{a}} = (\hat{a}_1, ..., \hat{a}_L) \text{ where } \hat{a}_i = \hat{a}_i^{j_i^*}; \tag{3.36}$$

here, the j_i^* are generated using

$$\underline{j}^* = (j_1^*, ..., j_L^*) = \arg\max_{\underline{j}} \sum_{i=1}^{L} \{\ln(l_i^{j_i}) + \ln P(c^{j_i}|c^{j_{i-1}}, ..., c^{j_{i-J}})\}. \tag{3.37}$$

This completes the description of the receiver components.

Chapter 4

The Parallel Receiver Structure: Analysis

In this chapter, we analyze the parallel receiver structure. We begin by introducing a condition on the discrete parameter space $\tilde{C} = \{\underline{c}^1, \underline{c}^2, ..., \underline{c}^m\}$. Whenever $\tilde{C}$ satisfies this condition, it can be used in our receiver structure.

Next, we generate the existence condition for our receiver. When a communication environment satisfies this existence condition, it is possible to find a $\tilde{C}$ satisfying its key condition, and hence it is possible to apply the proposed receiver structure to this environment.

The chapter concludes with two algorithms: the first algorithm provides insight into the number of parallel demodulators (m) required by the receiver structure; the second algorithm, using the results of the first, establishes the discrete parameter space $\tilde{C} = \{\underline{c}^j, j = 1, 2, ..., m\}$.

4.1 Constraints on $\tilde{C}$ for the Existence of the Receiver

In Chapter 3, we generated three key equations which served as the basis for the general receiver structure. These three equations, (3.9), (3.20), and (3.21), were all generated from joint detection and estimation equations using the discrete space approximation; here, the continuous parameter space, either C or C_0, was approximated by the discrete parameter space $\tilde{C}$. It was assumed at the time that the $\tilde{C}$ was carefully chosen to insure that the approximation was a good one. That is, the $\tilde{C}$ insured that data detection achieved by the equations using C or C_0 was very close to the data detection achieved by the equations using $\tilde{C}$.

This section provides a mathematical description of the condition on the discrete parameter space $\tilde{C}$. Whenever the $\tilde{C}$ satisfies this new condition, the discrete parameter space approximation is valid, and the $\tilde{C}$ can be used in building our receiver; conversely, when $\tilde{C}$ fails the condition, the discrete space approximation is invalid, and the $\tilde{C}$ should not be used in the receiver structure.

We want to formalize the statement regarding $\tilde{C}$: data detection using C (or C_0) is close to data detection using $\tilde{C}$. We begin by defining a performance measure, a value which allows us to measure how well a receiver, implementing a data detection equation, is doing. The measure of performance that has become a standard for receivers is probability of error, denoted $P(\epsilon)$. This value refers to the likelihood of a difference between the receiver symbol $\hat{a}_i$ and the transmitter symbol a_i. This probability reflects the likelihood that binary symbols b_i and $\hat{b}_i$ are different [2, p.180-181]. A larger probability of error indicates a diminished performance. With this well defined performance measure, we can now restate our starting condition on the discrete parameter space $\tilde{C}$. We want $\tilde{C}$ to insure that: (i) the $P(\epsilon)$ achieved in (3.8) (using C) is close to the $P(\epsilon)$ achieved in (3.9) (using $\tilde{C}$); or, (ii) in cases of independent noise samples, the $P(\epsilon)$ achieved in (3.18) (using C_0) is close to the $P(\epsilon)$ achieved in (3.20) or (3.21) (using $\tilde{C}$). We will consider the first of these two conditions on $\tilde{C}$, as the second condition can be studied analogously.

Our requirement on $\tilde{C}$ can be restated as

$$P_{\tilde{C}}(\epsilon) \approx P_C(\epsilon), \tag{4.1}$$

where $P_{\tilde{C}}(\epsilon)$ refers to the probability of error that results from equation (3.9) (using $\tilde{C}$), and $P_C(\epsilon)$ refers to the probability of error generated by equation (3.8) (using C). We can state our requirement on $\tilde{C}$ more precisely using

$$P_{\tilde{C}}(\epsilon) \gtrsim P_C(\epsilon); \tag{4.2}$$

equation (4.2) indicates that $P_{\tilde{C}}(\epsilon)$ is greater than, but very close to, $P_C(\epsilon)$ (e.g., $P_C(\epsilon) < P_{\tilde{C}}(\epsilon) < P_C(\epsilon) + \epsilon$, where ϵ is some small positive value). The terms of equation (4.2) are functions of the nuisance parameter vector in effect, $\underline{c}$. Hence, we state, more specifically, that we want the probabilities of error averaged over $\underline{c}$ to be close, i.e.,

$$E_{\underline{c}}[P_{\tilde{C}}(\epsilon|\underline{c})] \gtrsim E_{\underline{c}}[P_C(\epsilon|\underline{c})]; \tag{4.3}$$

here, $E_{\underline{c}}[\cdot]$ refers to an averaging over $\underline{c}$; $P_{\tilde{C}}(\epsilon|\underline{c})$ refers to the probability of error in (3.9) (using $\tilde{C}$) when $\underline{c}$ is the parameter vector in effect; and $P_C(\epsilon|\underline{c})$ refers to the error probability in (3.8) (using C) when $\underline{c}$ is in effect. Hence our requirement on $\tilde{C}$ corresponds to

$$E_{\underline{c}}[P_{\tilde{C}}(COR|\underline{c})] \lesssim E_{\underline{c}}[P_C(COR|\underline{c})], \tag{4.4}$$

where $P(COR|\underline{c})$ refers to the probability of a correct decision when $\underline{c}$ is in effect, $P_{\tilde{C}}(COR|\underline{c})$ refers to the probability of correct decision in equation (3.9) (using $\tilde{C}$) when $\underline{c}$ is in effect, and $P_C(COR|\underline{c})$ is the correct decision probability in (3.8) (using C) with $\underline{c}$ in effect.

The final condition on $\tilde{C}$ is generated from (4.4) using two assumptions, which we now state. First, we assume that there exists a set $N_{\underline{c}} \subset C$, defined by: given $\underline{c}$, $N_{\underline{c}} = \{\hat{\underline{c}} \in C : \quad P(COR|\underline{c}, \hat{\underline{c}}) \geq P(COR|\underline{c}, \underline{c}) - \epsilon\}$; here, $P(COR|\underline{c}, \hat{\underline{c}})$ refers to the probability of a correct decision when a demodulator assumes $\hat{\underline{c}}$, but in reality $\underline{c}$ is in effect; also, ϵ is some small positive value. In words, this assumption states that, for any given $\underline{c}$, there exists a set of points, $N_{\underline{c}}$, such that the performance of a demodulator assuming $\hat{\underline{c}} \in N_{\underline{c}}$ is very close to coherent. Our second assumption regards receivers implementing joint MAP detection. We assume that such receivers

will output an estimate of $\underline{c}$, $\underline{\hat{c}}$, that satisfies $P(\underline{\hat{c}} \in N_{\underline{c}}) >> P(\underline{\hat{c}} \notin N_{\underline{c}})$; that is, the estimate of $\underline{c}$ that results from joint MAP detection, $\underline{\hat{c}}$, is very likely to correspond to near coherent data detection. This is simply the claim that joint MAP detection achieves a good performance.

With the above two assumptions in hand, we begin to simplify the condition on $\tilde{C}$ in (4.4) by working first with the RHS. We can restate the expectation on the RHS according to $E_{\underline{c}}[P_C(COR|\underline{c})] = E_{\underline{c}}[\int_{\underline{\hat{c}} \in N_{\underline{c}}} P_C(COR|\underline{c}, \underline{\hat{c}}) \cdot p(\underline{\hat{c}}) d\underline{\hat{c}} + \int_{\underline{\hat{c}} \notin N_{\underline{c}}} P_C(COR|\underline{c}, \underline{\hat{c}}) \cdot p(\underline{\hat{c}}) d\underline{\hat{c}}]$. Here, $P_C(COR|\underline{c}, \underline{\hat{c}})$ refers to the probability of a correct decision in the joint MAP equation of (3.8), when $\underline{c}$ is in effect, and $\underline{\hat{c}}$ is the estimate on $\underline{c}$ that results in (3.8). Our assumption that joint MAP provides reliable data detection indicates $P(\underline{\hat{c}} \in N_{\underline{c}}) >> P(\underline{\hat{c}} \notin N_{\underline{c}})$; furthermore, by definition of $N_{\underline{c}}$, $P_C(COR|\underline{c}, \underline{\hat{c}} \in N_{\underline{c}}) > P_C(COR|\underline{c}, \underline{\hat{c}} \notin N_{\underline{c}})$. Hence, we can rewrite the expectation on the RHS according to $E_{\underline{c}}[P_C(COR|\underline{c})] \approx E_{\underline{c}}[\int_{\underline{\hat{c}} \in N_{\underline{c}}} P_C(COR|\underline{c}, \underline{\hat{c}}) \cdot p(\underline{\hat{c}}) d\underline{\hat{c}}]$.

Finally, using $P_C(COR|\underline{c}, \underline{\hat{c}} \in N_{\underline{c}}) \geq P_C(COR|\underline{c}, \underline{\hat{c}} = \underline{c}) - \epsilon$, where ϵ is some small value, we can rewrite the RHS according to $E_{\underline{c}}[P_C(COR|\underline{c})] \approx E_{\underline{c}}[\int_{\underline{\hat{c}} \in N_{\underline{c}}} P_C(COR|\underline{c}, \underline{\hat{c}} = \underline{c}) \cdot p(\underline{\hat{c}}) d\underline{\hat{c}}] = E_{\underline{c}}[P_C(COR|\underline{c}, \underline{\hat{c}} = \underline{c}) \cdot \int_{\underline{\hat{c}} \in N_{\underline{c}}} p(\underline{\hat{c}}) d\underline{\hat{c}}] = E_{\underline{c}}[P_C(COR|\underline{c}, \underline{\hat{c}} = \underline{c}) \cdot P(\underline{\hat{c}} \in N_{\underline{c}})]$. Using this in equation (4.4) leads to the requirement on $\tilde{C}$:

$$E_{\underline{c}}[P_{\tilde{C}}(COR|\underline{c})] \lesssim E_{\underline{c}}[P_C(COR|\underline{c}, \underline{\hat{c}} = \underline{c}) \cdot P(\underline{\hat{c}} \in N_{\underline{c}})]. \tag{4.5}$$

We now rework the left hand side (LHS) of this equation. We first restate the expectation on the LHS according to $E_{\underline{c}}[P_{\tilde{C}}(COR|\underline{c})] = E_{\underline{c}}[P_{\tilde{C}}(COR|\underline{c}, \underline{\hat{c}} = Q(\underline{c})) \cdot P(\underline{\hat{c}} = Q(\underline{c})) + \sum_{\underline{c}^j \in \tilde{C}, \underline{c}^j \neq Q(\underline{c})} P_{\tilde{C}}(COR|\underline{c}, \underline{\hat{c}} = \underline{c}^j) \cdot P(\underline{\hat{c}} = \underline{c}^j)]$. Here, $P_{\tilde{C}}(COR|\underline{c}, \underline{\hat{c}} = \underline{c}')$ refers to the probability of a correct decision in equation (3.9) (a joint MAP equation using the discrete space $\tilde{C}$), when $\underline{c}$ is in effect and $\underline{\hat{c}} = \underline{c}'$ is the estimate that results from (3.9). Also, $Q(\underline{c})$ refers to the $\underline{\tilde{c}} \in \tilde{C}$ closest to $\underline{c}$, where the closeness of $\underline{\tilde{c}}$ and $\underline{c}$ is measured by the size of the metric: probability of error when $\underline{c}$ is in effect and $\underline{\tilde{c}}$ is assumed. Now, assuming again that joint MAP provides reliable data detection, $P(\underline{\hat{c}} = Q(\underline{c})) >> P(\underline{\hat{c}} \neq Q(\underline{c}) \mid \underline{\hat{c}} \in \tilde{C})$; also, by definition of $Q(\underline{c})$, $P_{\tilde{C}}(COR|\underline{c}, \underline{\hat{c}} = Q(\underline{c})) > P_{\tilde{C}}(COR|\underline{c}, \underline{\hat{c}} = \underline{c}^j)$ for all $\underline{c}^j \in \tilde{C}$ ($\underline{c}^j \neq Q(\underline{c})$). Hence, the expectation on the LHS is bounded tightly by $E_{\underline{c}}[P_{\tilde{C}}(COR|\underline{c})] \gtrsim E_{\underline{c}}[P_{\tilde{C}}(COR|\underline{c}, \underline{\hat{c}} = Q(\underline{c})) \cdot P(\underline{\hat{c}} = Q(\underline{c}))]$. We can update the LHS of (4.5) by using the above bound; this leads to the tighter

bound on $\tilde{C}$:

$$E_{\underline{c}}[P_{\tilde{C}}(COR|\underline{c}, \hat{\underline{c}} = Q(\underline{c})) \cdot P(\hat{\underline{c}} = Q(\underline{c}))] \lesssim E_{\underline{c}}[P_C(COR|\underline{c}, \hat{\underline{c}} = \underline{c}) \cdot P(\hat{\underline{c}} \in N_{\underline{c}})]. \quad (4.6)$$

In general, $P(\hat{\underline{c}} \in N_{\underline{c}})$ in (3.8) and $P(\hat{\underline{c}} = Q(\underline{c}))$ in (3.9) are close, and hence the above bound is satisfied when $E_{\underline{c}}[P_{\tilde{C}}(COR|\underline{c}, \hat{\underline{c}} = Q(\underline{c}))] \lesssim E_{\underline{c}}[P_C(COR|\underline{c}, \hat{\underline{c}} = \underline{c})]$; that is, $E_{\underline{c}}[P_{\tilde{C}}(\epsilon|\underline{c}, \hat{\underline{c}} = Q(\underline{c}))] \gtrsim E_{\underline{c}}[P_C(\epsilon|\underline{c}, \hat{\underline{c}} = \underline{c})]$. This requirement on $\tilde{C}$ can be expressed more precisely by the bound on $\tilde{C}$:

$$E_{\underline{c}}[P_{\tilde{C}}(\epsilon|\underline{c}, \hat{\underline{c}} = Q(\underline{c}))] \leq h\{E_{\underline{c}}[P_C(\epsilon|\underline{c}, \hat{\underline{c}} = \underline{c})]\}; \quad (4.7)$$

here, $h\{\cdot\}$ represents $h : [0, 1] \to [0, 1]$ which satisfies $h(s) > s$ and $h(s) \approx s$ (e.g., $h(s) = s + \epsilon$ where ϵ is small) – we call $h\{\cdot\}$ the degradation function [42]. Finally, using a shorthand notation, we have the bound on $\tilde{C}$:

$$E_{\underline{c}}[P(\epsilon|\underline{c}, Q(\underline{c}))] \leq h\{E_{\underline{c}}[P(\epsilon|\underline{c}, \underline{c})]\}; \quad (4.8)$$

here, $P(\epsilon|\underline{c}, Q(\underline{c}))$ represents the probability of error of an ML demodulator assuming $Q(\underline{c})$ (the $\tilde{\underline{c}} \in \tilde{C}$ closest to $\underline{c}$) when in reality $\underline{c}$ is in effect; and $P(\epsilon|\underline{c}, \underline{c})$ represents the probability of error of an ML demodulator with complete information regarding the $\underline{c}$ in effect. Equation (4.8) corresponds to the mathematical equation that we require $\tilde{C}$ satisfy.

In the case of independent noise samples, in which case we want $\tilde{C}$ to insure the proximity of equation (3.18) to equation (3.20) and (3.21), the defining condition we will require $\tilde{C}$ satisfy can be shown to be the analogous

$$E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, Q(\underline{c}_i))] \leq h\{E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, \underline{c}_i)]\}. \quad (4.9)$$

In both equations (4.8) and (4.9), the degradation function $h(s)$ is not explicitly defined. This is because $h(s)$ determines how close our receiver structure's performance, using $\tilde{C}$, is to that of joint MAP detection and estimation; and there is no one universally good choice for this. Rather, its choice depends on the intended application of the receiver structure. For instance, if we want to apply our receiver structure to an environment where probability of errors less than 10^{-5} are not required at the receiver output, a good choice for $h(s)$ may be $h(s) = \max(s + 10^{-5}, 2 \cdot s)$.

In terms of the receiver structure in Figure 3.1, the conditions on $\tilde{C} = \{\underline{c}^1, \underline{c}^2, ..., \underline{c}^m\}$ in (4.8) and (4.9) can be expressed as follows. The nuisance parameter values assumed by each demodulator are carefully chosen. They insure that the performance of the best demodulator (the demodulator assuming a $\underline{c}^j$ closest to the channel parameters in effect) is close to the performance of an omniscient, or *coherent*, demodulator (a demodulator using the actual parameter values in effect).

The evaluation of an exact set $\tilde{C}$, satisfying either (4.8) or (4.9), is detailed later, in Section 4.4. There, we provide an algorithm for generating $\tilde{C}$, an algorithm based on quantization theory.

4.2 Existence Condition for the Proposed Receiver

In this section, we introduce an existence condition for our receiver structure. Whenever a communication environment meets this existence condition, our proposed receiver structure can be used to carry out data detection in the environment.

In what follows, we present the existence condition as a condition which insures that there exists a $\tilde{C}$ to satisfy (4.8) or (4.9). We do this because, whenever a $\tilde{C}$ can be found that satisfies (4.8) or (4.9), then this means that a $\tilde{C}$ can be found that can be used in our receiver structure; this, in turn, implies that our receiver exists.

The existence condition can be described as follows. *(1) For cases of dependent noise samples, if the communication environment demonstrates a* $P(\epsilon|\underline{c}, \underline{c}')$ *such that* $\lim_{\underline{c}' \to \underline{c}} P(\epsilon|\underline{c}, \underline{c}') = P(\epsilon|\underline{c}, \underline{c})$ *(for every* $\underline{c}' \in C$*), then the receiver structure exists (i.e., then there exists an integer* m *and set* $\tilde{C} = \{\underline{c}^1, ..., \underline{c}^m\}$ *satisfying (4.8)); similarly, (2) for cases of independent noise samples, if the communication environment demonstrates a* $P(\epsilon|\underline{c}_i, \underline{c}_i')$ *such that* $\lim_{\underline{c}_i' \to \underline{c}_i} P(\epsilon|\underline{c}_i, \underline{c}_i') = P(\epsilon|\underline{c}_i, \underline{c}_i)$ *(for every* $\underline{c}_i' \in C_0$*), then the receiver structure exists (i.e., then there exists an integer* m *and set* $\tilde{C} = \{\underline{c}^1, ..., \underline{c}^m\}$ *satisfying (4.9)).*

We now provide a brief intuitive argument, explaining how the condition $\lim_{\underline{c}' \to \underline{c}} P(\epsilon|\underline{c}, \underline{c}') = P(\epsilon|\underline{c}, \underline{c})$ insures the existence of a $\tilde{C}$ satisfying (4.8) (a complete proof is provided later). We begin by assuming that the communication environment demonstrates

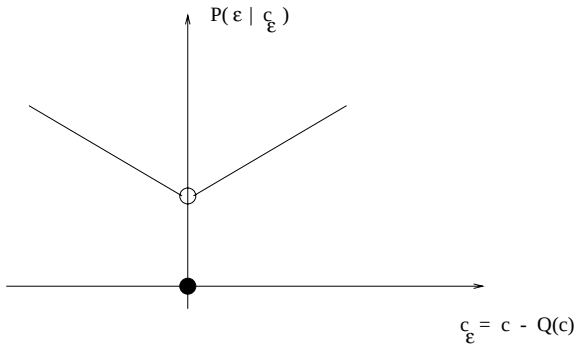

Figure 4.1: $P(\epsilon|\underline{c}_\epsilon)$, $\lim_{\underline{c}_\epsilon \to \underline{0}} P(\epsilon|\underline{c}_\epsilon) \neq P(\epsilon|\underline{0})$.

$P(\epsilon|\underline{c},\underline{c}') = P(\epsilon|\underline{c}-\underline{c}') = P(\epsilon|\underline{c}_\epsilon)$, and that this probability does *not* satisfy $\lim_{\underline{c}' \to \underline{c}} P(\epsilon|\underline{c},\underline{c}') = P(\epsilon|\underline{c},\underline{c})$ (for every $\underline{c}' \in C$), i.e., it does *not* always satisfy $\lim_{\underline{c}_\epsilon \to \underline{0}} P(\epsilon|\underline{c}_\epsilon) = P(\epsilon|\underline{0})$. An example of this is shown in Figure 4.1. (We note here that this is not a practical example, since it requires a receiver have exact knowledge of the continuous nuisance parameters to achieve near-coherent performances; never-the-less, it serves to illustrate our point regarding existence.) In this example, for *any* $\underline{c}$ in effect, regardless of the number of vectors we add to the discrete nuisance parameter space $\tilde{C}$, we can in no way guarantee that the vector $Q(\underline{c}) \in \tilde{C}$ achieves a $P(\epsilon|\underline{c},Q(\underline{c})) = P(\epsilon|\underline{c}-Q(\underline{c}))$ very close to $P(\epsilon|\underline{c},\underline{c}) = P(\epsilon|\underline{0})$; hence, we can not insure that (4.8) is satisfied, i.e, that $E_{\underline{c}}[P(\epsilon|\underline{c},Q(\underline{c}))] \leq h\{E_{\underline{c}}[P(\epsilon|\underline{c},\underline{c})]\}$.

Consider now the case when the communication environment demonstrates a $P(\epsilon|\underline{c},Q(\underline{c})) = P(\epsilon|\underline{c}-Q(\underline{c})) = P(\epsilon|\underline{c}_\epsilon)$, and this time the probability satisfies $\lim_{\underline{c}' \to \underline{c}} P(\epsilon|\underline{c},\underline{c}') = P(\epsilon|\underline{c},\underline{c})$, i.e., $\lim_{\underline{c}_\epsilon \to \underline{0}} P(\epsilon|\underline{c}_\epsilon) = P(\epsilon|\underline{0})$. An example of this is shown in Figure 4.2. Here, by adding more and more uniformly spaced vectors $\underline{c}^j$ into $\tilde{C}$, we can move the value $P(\epsilon|\underline{c},Q(\underline{c})) = P(\epsilon|\underline{c}-Q(\underline{c}))$ arbitrarily close to $P(\epsilon|\underline{c},\underline{c}) = P(\epsilon|\underline{c}_\epsilon = \underline{0})$. This is because, as we increase the number of vectors in a uniform manner, we can insure that there exists in $\tilde{C}$ a vector which moves $Q(\underline{c})$ very close to $\underline{c}$, in turn moving the performance $P(\epsilon|\underline{c},Q(\underline{c})) = P(\epsilon|\underline{c}-Q(\underline{c}))$ very close to $P(\epsilon|\underline{c},\underline{c}) = P(\epsilon|\underline{c}_\epsilon = \underline{0})$. This implies that equation (4.8) can be satisfied.

We now provide a formal proof of the existence condition. Specifically, we prove that the requirement on $P(\epsilon|\underline{c},\underline{c}')$ is *sufficient* to insure the satisfaction of equation

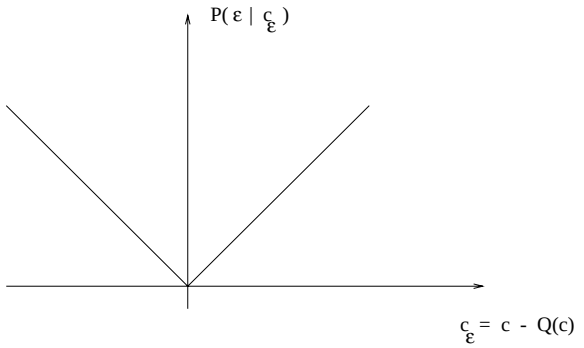

Figure 4.2: $P(\epsilon|\underline{c}_\epsilon)$, $\lim_{\underline{c}_\epsilon \to \underline{0}} P(\epsilon|\underline{c}_\epsilon) = P(\epsilon|\underline{0})$.

(4.8). Our proof demonstrates *sufficiency*, rather than both *sufficiency* and *necessity*. This is adequate, because all communication environments we are aware of satisfy this *sufficient* condition, insuring the applicability of our receiver to almost all communication environments. (The proof for the part of the existence condition relating to the satisfaction of (4.9) is carried out analogously.)

Proof:

We want to prove: if $\lim_{\underline{c}' \to \underline{c}} P(\epsilon|\underline{c}, \underline{c}') = P(\epsilon|\underline{c}, \underline{c})$ (for every $\underline{c}' \in C$), then there exists a set $\tilde{C}$ with a finite m satisfying

$$E_{\underline{c}}[P(\epsilon|\underline{c}, Q(\underline{c}))] \leq h\{E_{\underline{c}}[P(\epsilon|\underline{c}, \underline{c})]\}. \tag{4.10}$$

An equivalent proof is: if $\lim_{\underline{c}' \to \underline{c}} P(\epsilon|\underline{c}, \underline{c}') = P(\epsilon|\underline{c}, \underline{c})$, then

$$\lim_{m \to \infty} E_{\underline{c}}[P(\epsilon|\underline{c}, Q(\underline{c}))] = E_{\underline{c}}[P(\epsilon|\underline{c}, \underline{c})]. \tag{4.11}$$

This is equivalent because (4.11) implies, for any $\delta > 0$, that there exists a $\tilde{C}$ with a large (but finite) m such that $|E_{\underline{c}}[P(\epsilon|\underline{c}, Q(\underline{c}))] - E_{\underline{c}}[P(\epsilon|\underline{c}, \underline{c})]| < \delta$; this, in turn, implies that, for a $\tilde{C}$ with large but finite m, (4.10) can be satisfied.

We resolve the equivalent proof of (4.11) by reference to the Dominated Convergence Principle [63, p.111]. This principle states, for the case at hand: if $\lim_{Q(\underline{c}) \to \underline{c}} P(\epsilon|\underline{c}, Q(\underline{c})) = P(\epsilon|\underline{c}, \underline{c})$, then $\lim_{Q(\underline{c}) \to \underline{c}} E_{\underline{c}}[P(\epsilon|\underline{c}, Q(\underline{c}))] = E_{\underline{c}}[P(\epsilon|\underline{c}, \underline{c})]$. It follows that, if $\lim_{Q(\underline{c}) \to \underline{c}} P(\epsilon|\underline{c}, Q(\underline{c})) = P(\epsilon|\underline{c}, \underline{c})$, then $\lim_{m \to \infty} E_{\underline{c}}[P(\epsilon|\underline{c}, Q(\underline{c}))] = E_{\underline{c}}[P(\epsilon|\underline{c}, \underline{c})]$ is satisfied. $\diamondsuit$

4.3 Performance as a Function of m

In this section, we introduce an algorithm which generates m, the number of parallel demodulators employed in the receiver structure, for a stated communication environment. Specifically, given a particular communication environment and a stated receiver performance, the algorithm provided here generates a bound on the smallest value of m at which this performance may be achieved. This algorithm is developed for cases of independent noise samples and independent data symbols.

Establishing m may be the single most important criteria in determining the practical applicability of the proposed receiver. The previous section tells us whether or not it is theoretically possible to apply the receiver to a particular communication environment. However, the practical applicability of a receiver is determined by the complexity of its implementation. The primary factor in the complexity of our receiver is the number of parallel demodulators (m). If thousands are needed, it is clear that our proposed receiver is inapplicable. Fortunately, using the results of this section, we show, in later chapters (and in an example provided in this section) that, in many cases of practical interest, only a few parallel demodulators (usually less than 10) are required to achieve performances indistinguishable from those attained with $m \to \infty$. As a result, our proposed receiver is not only theoretically applicable, but also practical.

4.3.1 Performance Measure as a Function of m

In this subsection, we re-introduce the performance measure, and attempt to characterize this measure in terms of m. We hope to derive a simple equation describing the relationship between the performance measure and m.

As we pointed out earlier, the standard measure for evaluating the performance of a receiver is the probability that a_i and $\hat{a}_i$ differ, denoted $P(\epsilon)$. For the proposed parallel receiver, this probability can be expressed according to

$$P(\epsilon) = E_{\underline{c}_i, \underline{c}^j}[P(\epsilon|\underline{c}_i, \underline{c}^j)], \tag{4.12}$$

where $P(\epsilon|\underline{c}_i, \underline{c}^j)$ refers to the likelihood of an error at the j^{th} demodulator when $\underline{c}_i$ is in effect. Restating this expectation leads to

$$P(\epsilon) = \sum_{j=1}^{m} \int_{C_0} [P(\epsilon|\underline{c}_i, \underline{c}^j)] \cdot P(\underline{c}^j|\underline{c}_i) \cdot p(\underline{c}_i) d\underline{c}_i; \tag{4.13}$$

here, $P(\underline{c}^j|\underline{c}_i)$ denotes the probability that the CDU output corresponds to the j^{th} demodulator's decision, given that $\underline{c}_i$ is in effect. It is difficult to come up with a simple expression for this probability. This is because the CDU decisions are based on the statistics of the entire sequence of L symbols, and hence the likelihood of a decision at a particular time i is hard to resolve.

As a result of the difficulty in simplifying $P(\underline{c}^j|\underline{c}_i)$, the evaluation of the m at which a stated performance $P(\epsilon)$ can be attained is not easily resolved by a brute force evaluation of $P(\epsilon)$ as a function of m.

4.3.2 A New Equation Relating Performance Measure to m

In this subsection, we introduce a realization regarding the $P(\epsilon) - m$ relationship in our receiver structure. This realization leads to a simpler equation describing the $P(\epsilon) - m$ relationship.

The realization, regarding the $P(\epsilon) - m$ relationship of our receiver structure, is generated as follows. It begins with the recognition that detection errors in our proposed parallel receiver can be attributed to two sources of error – errors created by the best demodulator, and errors introduced by the CDU.

The first source of errors, errors in the best demodulator, refer to the errors created by the demodulator assuming a $\underline{c}^j$ closest to $\underline{c}_i$. These errors are a result of both additive noise and the demodulator's assumption of $\underline{c}^j$ when in fact $\underline{c}_i$ is in effect. By increasing m, the $\underline{c}^j$ of the best demodulator is likely to move closer to $\underline{c}_i$; this reduces the errors made by the best demodulator.

The errors created by the CDU, the second source of errors, refer to the errors made in addition to those at the best demodulator. That is, it refers to the additional errors made when the CDU selects, as its output, the decisions of a demodulator with

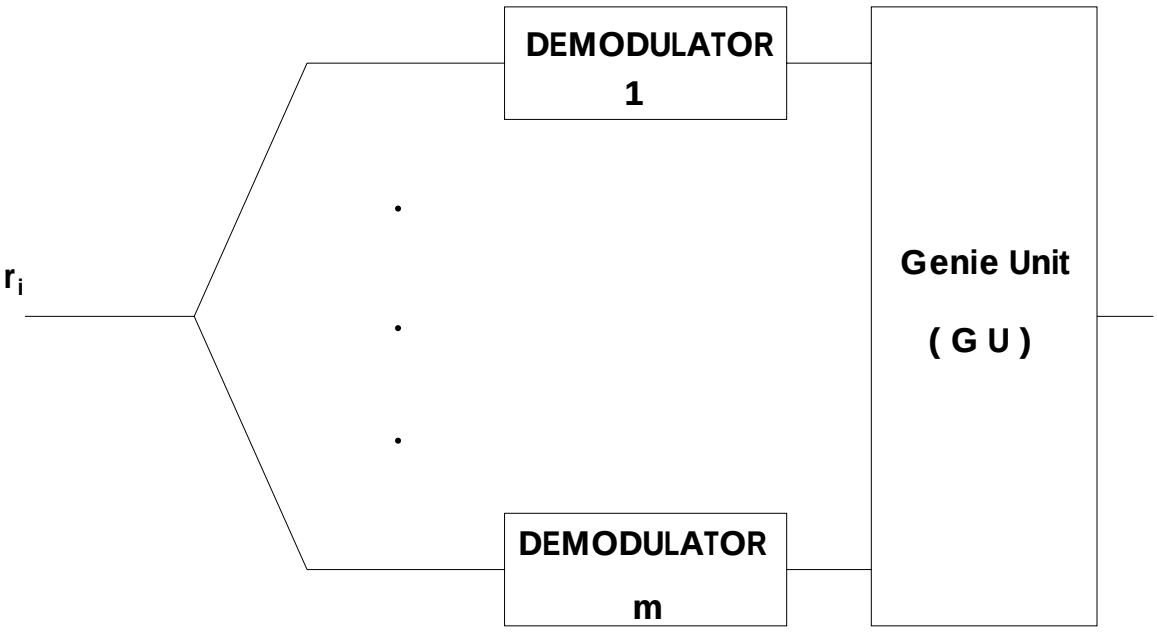

Figure 4.3: Receiver characterizing $P(\epsilon) - m$ relationship

a $\underline{c}^j$ which is not closest to $\underline{c}_i$. The CDU's method of choosing from among the m demodulator decisions is uneffected by the value of m. In fact, regardless of m, the CDU is usually able to select the demodulator with a $\underline{c}^j$ closest to $\underline{c}_i$.

For the purposes of characterizing the $P(\epsilon) - m$ relationship in our receiver structure, rather than the complete $P(\epsilon)$, we can omit the contributions to error which are independent of m. Hence, from the arguments presented in the above two paragraphs, it suffices to focus on the errors created by the best demodulator, and ignore those of the CDU. That is, we can characterize the $P(\epsilon) - m$ relationship of our parallel receiver structure by characterizing the $P(\epsilon) - m$ relationship of the receiver in Figure 4.3. Here, the *Genie Unit* (or *GU*, for short) refers to an ideal processing unit that always outputs the decisions generated by the best demodulator, i.e., by the demodulator using an assumed $\underline{c}^j$ closest to $\underline{c}_i$. This receiver only experiences the errors created by the best demodulator.

With this key realization in hand, we now try to characterize the $P(\epsilon) - m$ relationship by a brute force computation of the $P(\epsilon)$ of the receiver in Figure 4.3.

We hope to be able to achieve an expression for this $P(\epsilon)$ which clearly indicates its dependence on m. The overall $P(\epsilon)$ of the parallel receiver of Figure 4.3 is described by

$$P(\epsilon) = E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, Q(\underline{c}_i))]; \tag{4.14}$$

that is, it is the averaging, over $\underline{c}_i$, of the probability of error achieved by the demodulator assuming $Q(\underline{c}_i)$ (the $\underline{c}^j$ closest to the $\underline{c}_i$). Restating the expectation in integral form leads to

$$P(\epsilon) = \int_C P(\epsilon|\underline{c}_i, Q(\underline{c}_i))p(\underline{c}_i)d\underline{c}_i; \tag{4.15}$$

or, equivalently,

$$P(\epsilon) = \sum_{j=1}^{m} \int_{R_j} P(\epsilon|\underline{c}_i, \underline{c}^j)p(\underline{c}_i)d\underline{c}_i, \tag{4.16}$$

where $R_j = \{\underline{c}_i \in C_0 \ : \ Q(\underline{c}_i) = \underline{c}^j\}$, i.e., R_j corresponds to the subset of C_0 made up of the elements $\underline{c}_i \in C_0$ which are closer to $\underline{c}^j$ than any other $\underline{c}^k, \ k \neq j$.

The $P(\epsilon)$ of equation (4.16) could be used to establish the $P(\epsilon) - m$ relationship. This, however, would be a long process. For each value of m, we could generate a set $\tilde{C} = \{\underline{c}^1, ..., \underline{c}^m\}$, using either intuitive arguments or the algorithm provided in the next section (Section 4.4); using this $\tilde{C}$, and numerical integration methods, we could solve for $P(\epsilon)$ from equation (4.16). Repeated application of these two steps, each time with a new value of m, establishes a $P(\epsilon) - m$ relationship.

4.3.3 $P(\epsilon) - m$ Evaluated Using Rate Distortion Theory

This subsection introduces a practical algorithm which establishes the $P(\epsilon)$ vs m relationship for the proposed parallel receiver structure. This algorithm is generated starting from the key realization of the previous subsection, which explains that for the sake of characterizing $P(\epsilon) - m$, our proposed receiver can be replaced by Figure 4.3. From this starting point, we show how *rate distortion theory* can be applied to create an algorithm that establishes $P(\epsilon)$ vs m.

The work that follows is subdivided into five parts. First, we describe the receiver of Figure 4.3 in a form more appropriate for rate distortion theory analysis. Next, we provide a brief introduction to rate distortion theory. Third, we apply the ideas of

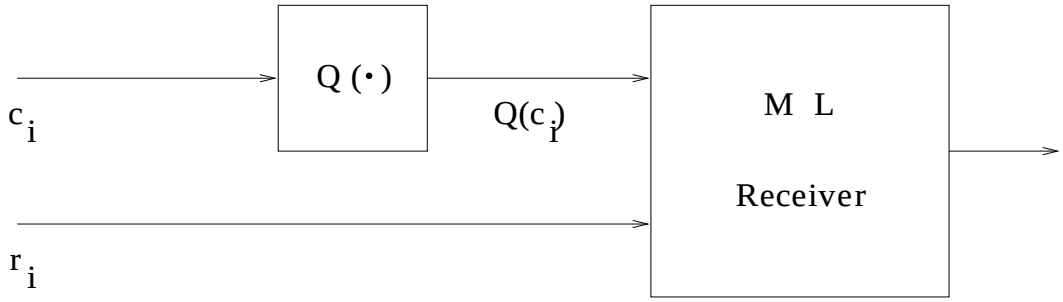

Figure 4.4: Alternative implementation of genie receiver.

rate distortion theory to the new description of the receiver of Figure 4.3, resulting in an algorithm which establishes the $P(\epsilon) - m$ relationship. Two final subsections provide some insights regarding this $P(\epsilon) - m$ algorithm.

An Alternative Description of the Genie Receiver

The receiver of Figure 4.3, which we will now refer to as the *genie receiver*, serves as our starting point in characterizing the $P(\epsilon) - m$ relationship. We now express the operation of the genie receiver in a form more suitable for the upcoming rate distortion analysis.

An alternative implementation of the genie receiver is shown in Figure 4.4. Here, we show the genie receiver with two inputs: the nuisance parameter vector $\underline{c}_i$, and the received sample r_i. The leftmost component of the genie receiver is an m-level quantizer, which carries out the mapping $Q : C_0 \rightarrow \tilde{C}$; that is, given $\underline{c}_i$, it generates $Q(\underline{c}_i) = \underline{c}^j$, where $\underline{c}^j$ is the element in $\tilde{C}$ closest to $\underline{c}_i$. The ML receiver component carries out ML data detection assuming $Q(\underline{c}_i)$ is the correct nuisance parameter vector, i.e., it outputs $\hat{a}_i = \arg\max_{a_i} p(r_i|a_i, Q(\underline{c}_i))$.

The quantizer is characterized (in part) by a rate R, which indicates the number of bits needed to represent its output vectors; that is, R represents the number of bits the quantizer uses in representing the input vector. In our case, $R = \log m$, where all logarithms in this subsection are base 2.

The overall performance of this receiver is $P(\epsilon) = E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, Q(\underline{c}_i))]$. We will now

refer to this term as the average distortion, and label it D.

Using this characterization of the receiver of Figure 4.3, we can now express the relationship between $P(\epsilon)$ and m as a relationship between D and R.

Rate Distortion Theory

One branch of information theory, that can help in establishing the $R-D$ relationship, is *rate distortion theory*. The foundations for rate distortion theory (as well as the whole of information theory) were established by Claude Shannon in 1948, in his much-celebrated journal article 'A Mathematical Theory of Communication' [64]. Following this introduction in Shannon's article, the literature on rate distortion theory grew steadily. In 1972, Toby Berger's *Rate Distortion Theory* [65] pieced together much of the diverse literature on this topic.

At the heart of rate distortion theory lies a function, called the *rate distortion function*, and denoted $R(D)$. This function defines the minimum rate R which can be used, to represent an information-bearing signal, such that it produces an average distortion of D. Whenever we are given the statistics of the information-bearing signal, and a distortion measure, rate distortion theory explains how to generate this $R(D)$.

Applying Rate Distortion Theory

In this subsection, we apply rate distortion theory to the receiver of Figure 4.4, and establish a rate distortion function, $R(D)$, for the receiver. In this case, with R corresponding to $\log m$, and D to $P(\epsilon)$, this $R(D)$ function establishes the smallest m at which we can achieve a performance of $P(\epsilon)$. This serves as our characterization of the $P(\epsilon) - m$ relationship.

Before we can generate the rate distortion function, $R(D)$, for our receiver, we must introduce two characteristics: a distortion measure; and the statistics of the random vector being represented at rate R.

A distortion measure refers to a non-negative cost function; it indicates the cost when an input, say $\underline{x}$, is represented by the value $\underline{y}$. The average distortion of a system, D, is the average of the distortion measure over all possible inputs. In our case, the input $\underline{c}_i$ is quantized to $Q(\underline{c}_i)$. We want to define a distortion measure to describe the cost of representing $\underline{c}_i$ by $Q(\underline{c}_i)$. We choose the distortion measure

$$\rho(\underline{c}_i, Q(\underline{c}_i)) = P(\epsilon|\underline{c}_i, Q(\underline{c}_i)). \tag{4.17}$$

This leads to an average distortion D which is consistent with the probability of error measure introduced earlier, namely $D = P(\epsilon) = E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, Q(\underline{c}_i))]$.

A second characteristic, which must be introduced before we can establish $R(D)$, is the statistical characterization of the random vector being quantized at a rate R. In our case, the random vector being quantized is $\underline{c}_i$. The genie receiver, as described in Figure 4.4, quantizes each $\underline{c}_i$ independently of all other $\underline{c}_k$'s, $k \neq i$; additionally, its quantization operation is identical at each sample time i. Here, the receiver is acting as if each $\underline{c}_i$ is an independent, identically distributed (i.i.d.) random vector, fully characterized by $p(\underline{c}_i)$. Since we want the $R(D)$ relationship for this receiver, we will make these very same claims regarding the statistics of $\underline{c}_i$.

With these two characteristics in hand, we can now apply rate distortion theory to generate $R(D)$. Referring to [65, p.88], we find that, for our application, with the two characteristics provided above, the function $R(D)$ is given by

$$R(D) = \inf_{p \in Q_D} I(\underline{c}_i; \underline{y}) \tag{4.18}$$

where

$$Q_D = \{p(\underline{y}|\underline{c}_i) : \int_{C_0}\int_{C_0} p(\underline{c}_i)p(\underline{y}|\underline{c}_i)\rho(\underline{c}_i, \underline{y})d\underline{c}_i d\underline{y} = D\}; \tag{4.19}$$

$$I(\underline{c}_i; \underline{y}) = \int_{C_0}\int_{C_0} p(\underline{c}_i)p(\underline{y}|\underline{c}_i)\log\frac{p(\underline{y}|\underline{c}_i)}{p(\underline{c}_i)}d\underline{c}_i d\underline{y}; \tag{4.20}$$

here, $\underline{y} \in C_0$. The units of $R(D)$ are bits per vector $\underline{c}_i$.

This equation states that the minimum rate R at which a distortion D can be achieved corresponds to: the smallest amount of average information that an output $\underline{y}$ can convey about an input $\underline{c}_i$, and still attain an average distortion of D. This equation, due in part to its integral nature, is not easy to compute.

Fortunately, whenever we have a difference distortion measure, that is, the distortion measure $\rho(\underline{c}_i, \underline{y}) = P(\epsilon|\underline{c}_i, \underline{y}) = P(\epsilon|\underline{c}_i - \underline{y}) = P(\epsilon|\underline{c}_\epsilon)$, then this $R(D)$ is lower bounded by a Shannon bound. The good news here is that the Shannon bound is both a tight bound and is easier to evaluate than the exact $R(D)$. This tight bound, which we call $R_L(D)$ (L for lower bound), can be generated in parametric form, for our case of interest, as follows. For a given $s \in (-\infty, 0]$, compute $D = D(s)$ according to

$$D = \int g_s(\underline{c}_\epsilon) P(\epsilon|\underline{c}_\epsilon) d\underline{c}_\epsilon \tag{4.21}$$

where

$$g_s(\underline{c}_\epsilon) = \frac{e^{sP(\epsilon|\underline{c}_\epsilon)}}{\int e^{sP(\epsilon|\underline{c}_\epsilon)} d\underline{c}_\epsilon}. \tag{4.22}$$

The corresponding $R = R(D)$ is lower bounded (tightly) by

$$R_L = R_L(D) = R_L(s) = h(\underline{c}_i) - h(g_s), \tag{4.23}$$

where $h(\underline{c}_i)$ refers to the entropy of $\underline{c}_i$, that is, $h(\underline{c}_i) = -\int p(\underline{c}_i) \log p(\underline{c}_i) d\underline{c}_i$; and $h(g_s) = -\int g_s(\underline{c}_\epsilon) \log g_s(\underline{c}_\epsilon) d\underline{c}_\epsilon$. In this way, the $R - D$ relationship can be established, one point at a time. (The $s \in (-\infty, 0]$ value, used to generate $(D(s), R_L(s))$, itself corresponds to the slope of the $R - D$ curve at the point approximated by $(D(s), R_L(s))$.)

The terms D and R_L (in equations (4.21) and (4.23) respectively), used to generate the Shannon lower bound, can only be evaluated numerically. In what follows, we provide a detailed algorithm which explains how the Shannon lower bound is generated using numerical methods. Briefly, the algorithm varies s over a range of values between $-\infty$ and 0, and evaluates one point on the $R - D$ curve at each s value by numerical means. All logarithms in the algorithm that follows are base e.

Algorithm 1:

A1. **(i)** Evaluate an analytical expression for the distortion measure $P(\epsilon|\underline{c}_\epsilon)$ for the communication environment of interest.

(ii) Select a Signal-to-Noise (SNR) ratio at which we want to establish the $P(\epsilon) - m$ relationship. A good choice for this SNR value is the intended system operating point. The reason we fix the SNR value is because $P(\epsilon)$ is a function of SNR, and we are not currently interested in this dependence.

(iii) Choose a starting s, s_0; e.g., $s_0 = -1.0$.

(iv) Choose a rule for generating a new s value, s_{i+1}, from the previous s value, s_i; e.g., $s_{i+1} = s_i * 2$.

(v) Set a minimum (final) value for s, s_{min}; e.g., $s_{min} = -4096$.

(vi) Set $n = 0$.

B1. Compute $I1 = \int e^{s_n P(\epsilon|\underline{c}_\epsilon)} d\underline{c}_\epsilon$ according to

$$I1 \approx \sum_{i=1}^{K} e^{s_n P(\epsilon|\underline{c}_{\epsilon_i})} \Delta\underline{c}_\epsilon. \tag{4.24}$$

C1. Compute $D = P(\epsilon) = \frac{1}{I1} \int e^{s_n P(\epsilon|\underline{c}_\epsilon)} P(\epsilon|\underline{c}_\epsilon) d\underline{c}_\epsilon$ according to

$$D = P(\epsilon) \approx \frac{1}{I1} \sum_{i=1}^{K} e^{s_n P(\epsilon|\underline{c}_{\epsilon_i})} P(\epsilon|\underline{c}_{\epsilon_i}) \Delta\underline{c}_\epsilon. \tag{4.25}$$

D1.(i) Compute $h(g_s) = -1.0 * \frac{1}{I1} \int e^{s_n P(\epsilon|\underline{c}_\epsilon)} [s_n P(\epsilon|\underline{c}_\epsilon) - \log(I1)] d\underline{c}_\epsilon$ according to

$$h(g_s) \approx -1.0 * \frac{1}{I1} \sum_{i=1}^{K} e^{s_n P(\epsilon|\underline{c}_{\epsilon_i})} [s_n P(\epsilon|\underline{c}_{\epsilon_i}) - \log(I1)] \Delta\underline{c}_\epsilon. \tag{4.26}$$

(ii) Compute

$$R_L(D) = \frac{h(\underline{c}_i) - h(g_s)}{\log(2)}; \tag{4.27}$$

the $\log(2)$ insures that $R(D)$ is generated in units of bits per vector.

(iii) Compute

$$m = 2^{R_L(D)}. \tag{4.28}$$

E1. If $s_n \neq s_{min}$, let $n \leftarrow n + 1$, and return to **B1**;
otherwise, stop – the set of points $(m, P(\epsilon))$, one point generated at each s_n, can now be used to plot a $P(\epsilon) - m$ curve. $\diamondsuit$

Before proceeding, we pause here to emphasize that the $P(\epsilon) - m$ curve generated by the above algorithm provides a lower bound on the minimum m that achieves a performance $P(\epsilon)$ (and does *not* provide an actual minimum m that can achieve this

performance). There are two reasons for this. First, our algorithm generates R_L *vs* D (i.e., m_L *vs* $P(\epsilon)$), a lower bound on R *vs* D (i.e., on m *vs* $P(\epsilon)$). In addition, the values of the R *vs* D (m *vs* $P(\epsilon)$) curve are only attainable in the limit of large blocklength [65], i.e., in the limit as the number of $\underline{c}_i$'s quantized simultaneously tends toward the infinite; however, in the receiver at hand (Figure 4.4), each $\underline{c}_i$ is quantized independently. As a result, R *vs* D (m *vs* $P(\epsilon)$) acts as a lower bound for our receiver.

In our experience, the m value required to achieve a stated performance $P(\epsilon)$, generated by the $P(\epsilon)$ *vs* m algorithm, serves as a very good starting estimate for the actual m value required. For instance, in the case of AWGN and phase offset, when the $P(\epsilon)$-m curve indicated that the m value required to achieve a stated $P(\epsilon)$ was $m = 5$, the actual m value that was found to achieve the $P(\epsilon)$ was $m = 8$ (see Section 5.2 for details).

Application of the $P(\epsilon) - m$ Algorithm

In this subsection, we apply the algorithm designed to characterize the $P(\epsilon) - m$ relationship to a particular example of practical interest. This serves to demonstrate how to use this algorithm, the practical nature of the algorithm, and the practical applicability of our receiver structure.

We consider the case of differentially encoded MPSK symbols, with $M = 8$, transmitted over a channel which introduces both a carrier phase offset and AWGN. In this case, the received samples generated from the receiver front end correspond to

$$r_i = d_i e^{j\theta_i} + \eta_i; \tag{4.29}$$

here $d_i = a_i \cdot d_{i-1}$, where $d_0 = \sqrt{E_s}$ and a_i corresponds to an information-bearing MPSK symbol, that is, $a_i = e^{j\frac{2\pi}{M}l_i}$, where $l_i \in \{1, ..., M\}$; θ_i represents the unknown phase offset introduced by the channel at time sample i; and the η_i's correspond to complex random noise samples with real and imaginary parts that are i.i.d. Gaussian random variables of zero mean and variance $\frac{N_o}{2}$. We assume, for reasons which we will explain in a later chapter, that the phase offset is uniform on $[0, \frac{2\pi}{M})$, that is,

$$p(\theta_i) = \begin{cases} \frac{M}{2\pi}, & \theta_i \in [0, \frac{2\pi}{M}) \\ 0, & \text{else.} \end{cases} \tag{4.30}$$

We are now ready to apply our Shannon bound algorithm to establish the $P(\epsilon)-m$ relationship for the case at hand. The first part of the algorithm, **A1 (i)**, requires that we achieve an analytic expression of the distortion measure. Using union bounds and geometric arguments, we acquire

$$P(\epsilon|\theta_\epsilon) \approx \frac{1}{2}\text{erfc}(\sqrt{\frac{E_s}{N_o}}\sin(\frac{\pi}{M}+\theta_\epsilon)) + \frac{1}{2}\text{erfc}(\sqrt{\frac{E_s}{N_o}}\sin(\frac{\pi}{M}-\theta_\epsilon)). \tag{4.31}$$

We now choose parameter values as explained in the remainder of step **A1**. Specifically, we select $\frac{E_s}{N_o}$, a representative of the SNR, to be 16dB; at this value, the probability of error with perfect knowledge of the phase offset θ_i is 6.3×10^{-4}. We also choose $s_0 = -500$, $s_{i+1} = s_i * 2$, and $s_{min} = -128000$.

We next carry out steps **B1** through **E1**, using MATLAB. This leads to the $P(\epsilon)$ vs m curve of Figure 4.5. This curve suggests that when a small, finite number of demodulators are used in our receiver structure, almost all the performance to be had is achieved.

The $P(\epsilon) - m$ Characteristics as a Function of SNR

The algorithm provided in an earlier subsection generates a $P(\epsilon) - m$ relationship for our receiver structure at a given SNR. In this final subsection, we examine how this $P(\epsilon) - m$ relationship changes as we vary SNR. To establish this, we consider a particular example, namely the example of unknown phase described by (4.29).

We begin by presenting several $P(\epsilon) - m$ relationships, each one generated using the provided algorithm with a unique SNR value. Figure 4.6 shows the $P(\epsilon) - m$ curve that results when the SNR is 13dB; Figure 4.7 shows the $P(\epsilon) - m$ curve that results at an SNR of 18dB; and the earlier Figure 4.5 displays this $P(\epsilon) - m$ curve at an SNR of 16dB. In all these curves, the leveling-off value of $P(\epsilon)$ corresponds to the coherent $P(\epsilon)$.

We can draw the following conclusions from these curves. As SNR increases, the $P(\epsilon) - m$ relationship of our proposed receiver changes. Specifically, at larger SNR values, a larger m value is required to achieve a $P(\epsilon)$ close to that attained with $m \to \infty$. For instance, if we want a $P(\epsilon)$ within 10% of that attained with $m \to \infty$,

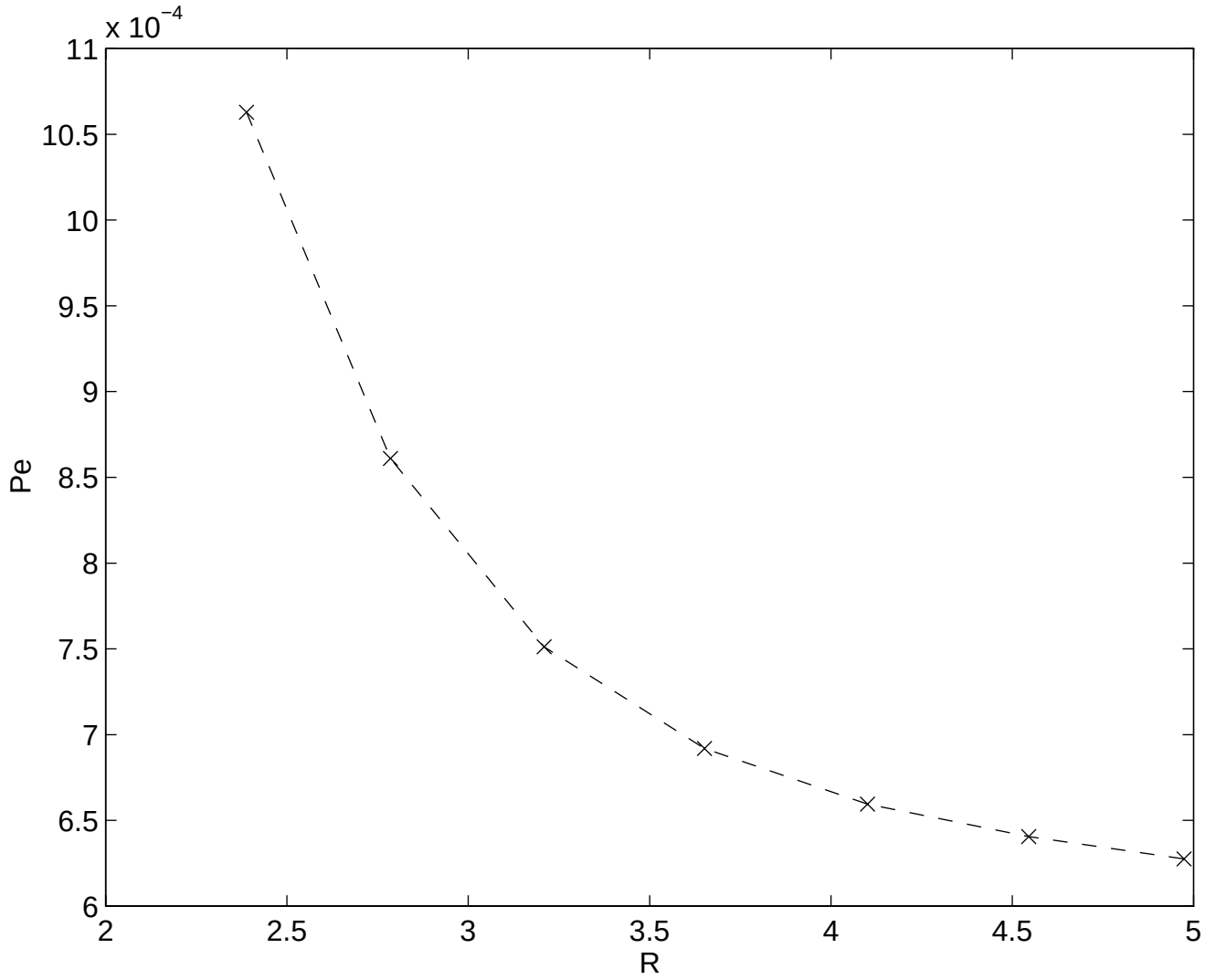

Figure 4.5: $P(\epsilon)$ *vs* $R = log(m)$ curve for 8-PSK with $\frac{E_s}{N_o} = 16$dB

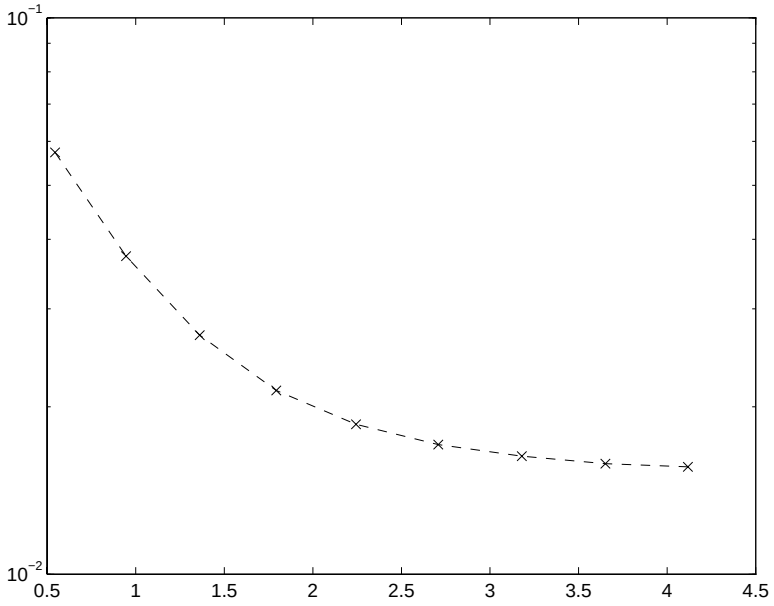

Figure 4.6: $P(\epsilon)$ *vs* $R = log(m)$ Curve for 8-PSK with $\frac{E_s}{N_o} = 13$dB

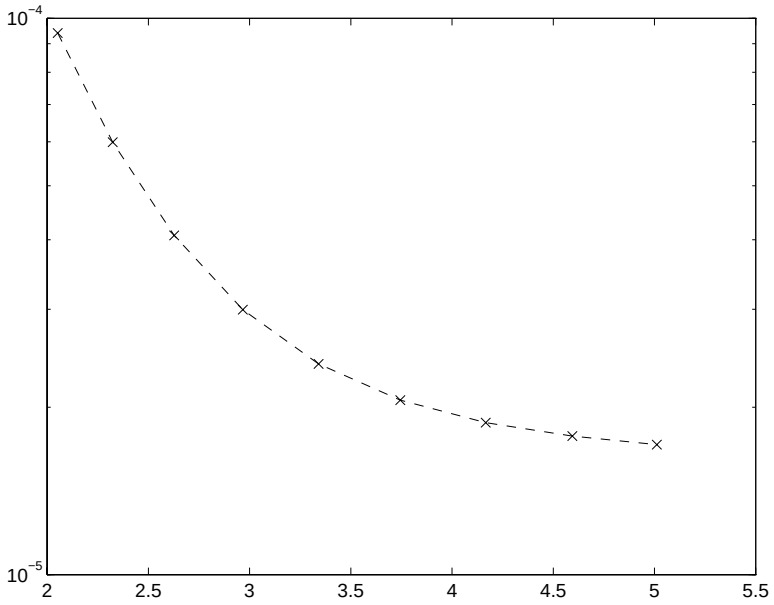

Figure 4.7: $P(\epsilon)$ *vs* $R = log(m)$ Curve for 8-PSK with $\frac{E_s}{N_o} = 18$dB

we require the following: our receiver must be built with $m \geq 7$ demodulators if the operating point of the system is 13dB; our receiver requires $m \geq 13$ demodulators if the system operating point is 16dB; this value rises to $m \geq 18$ if the system operating point is 18dB.

The reason our receiver requires a larger m at increased SNR's, to maintain a fixed percentage from the performance attained with $m \to \infty$, can be explained as follows. In the receiver of Figure 4.4, the receiver characterizing the $P(\epsilon) - m$ relationship, there are two sources of error. First, there are errors due to noise; this refers to errors that would be made even if the exact θ_i were known. Second, there are errors due to phase quantization; these are errors that result, in addition to those due to noise alone, because the phase value θ_i is approximated by $Q(\theta_i)$ at the receiver.

At low to mid SNR values, the errors due to noise are substantial. Hence, the errors due to phase quantization are negligible, even when m is a small value. As SNR values rise, the errors due to noise diminish. Hence, errors due to phase quantization become more noticeable. In these large SNR cases, a bigger m value is needed to diminish the errors due to phase quantization to the point where they are again negligible compared to the errors due to noise.

We want to point out that, assuming the genie receiver's $P(\epsilon)$ is a good approximation to that of our receiver (i.e., assuming the CDU usually selects the best demodulator's output), then as $SNR \to \infty$, our receiver structure demonstrates $P(\epsilon) \to 0$ regardless of m ($m > 1$) (for the example at hand). That is, there is no error floor for our receiver as $SNR \to \infty$ (regardless of m). Both coherent $P(\epsilon)$ and the $P(\epsilon)$ of our receiver demonstrate waterfall-like shapes as a function of SNR; an increased m simply moves our receiver's waterfall toward the coherent one. This is shown in Figure 4.8.

4.4 Evaluating a Discrete Parameter Space $\tilde{C}$

In this section, we provide an algorithm that establishes a set of values $\tilde{C} = \{\underline{c}^1, ..., \underline{c}^m\}$ for use in our proposed receiver. We do this in two parts. First, we establish what set

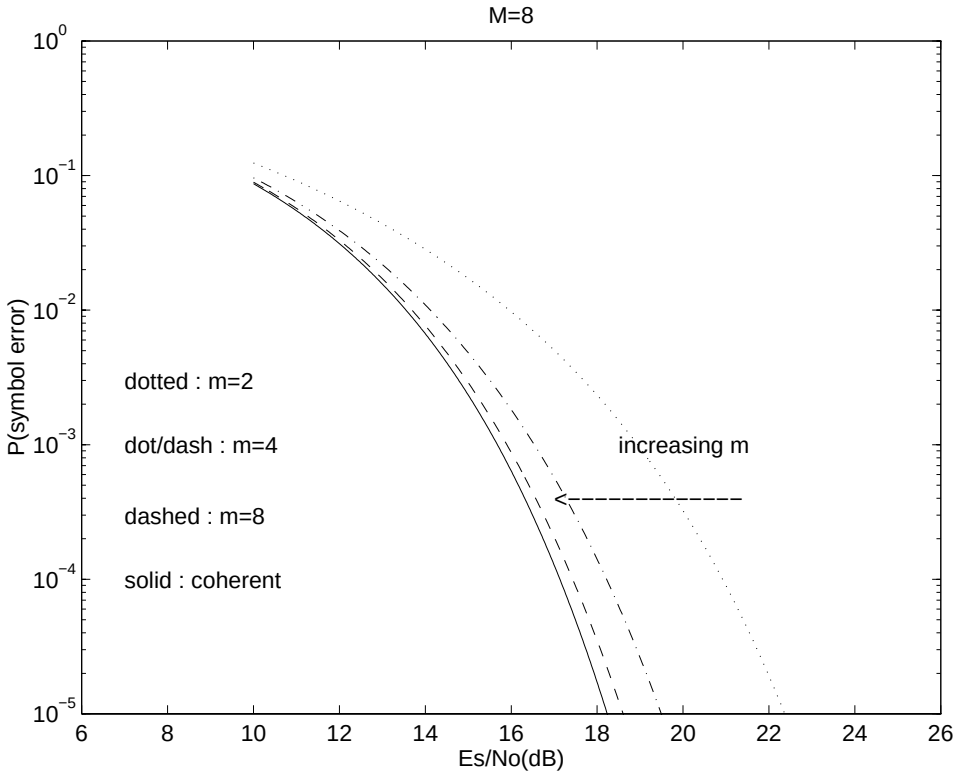

Figure 4.8: $P(\epsilon) - \frac{E_s}{N_o}$ as a function of m for 8-PSK

of values $\tilde{C}$ would be best in our receiver structure. Next, we provide an algorithm for generating this $\tilde{C}$. In the presentation that follows, we assume independent noise samples.

4.4.1 The Best Set of Values for $\tilde{C}$

In the first section (Section 4.1), we explain that the set $\tilde{C}$ must insure that the discrete space approximation used in Chapter 3 is valid. We go on to show that an equivalent requirement on $\tilde{C}$ is: $\tilde{C}$ must meet the performance constraint

$$E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, Q(\underline{c}_i))] \leq h\{E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, \underline{c}_i)]\}, \tag{4.32}$$

where $Q(\underline{c}_i)$ is the element in $\tilde{C} = \{\underline{c}^1, \underline{c}^2, ..., \underline{c}^m\}$ that is closest to $\underline{c}_i$. Hence, a best set of values for $\tilde{C}$ will satisfy equation (4.32).

The complexity of our receiver increases as m, the size of $\tilde{C}$, increases. As a result, the best set $\tilde{C} = \{\underline{c}^1, ..., \underline{c}^m\}$ has a low value of m; specifically, the best set $\tilde{C}$ uses the smallest m possible while still satisfying equation (4.32).

We narrow this definition of the best set $\tilde{C}$. The terms of equation (4.32) are functions of SNR. Hence, different best sets $\tilde{C}$ may result at different SNR's. One choice for the best set $\tilde{C}$ is the set, with the smallest size (m), satisfying (4.32) *at the SNR corresponding to the operating point of the system.*

Alternatively, we may want the best set $\tilde{C}$ to demonstrate robustness to large changes in SNR. Our work in Section 4.3 indicates that, as SNR decreases, a $\tilde{C}$ with fewer elements will satisfy (4.32); conversely, as SNR values rise, a $\tilde{C}$ containing a greater number of elements is required to meet (4.32). Hence, if our goal is a receiver that will satisfy (4.32) for a wide range of SNR's, $\tilde{C}$ should be chosen as the set, with the smallest possible m, satisfying (4.32) *at an SNR corresponding to the largest value in the range of anticipated SNR's.*

4.4.2 Algorithm

This section introduces an algorithm which generates a set $\tilde{C}$ having a small m and satisfying equation (4.32) at a selected SNR; it approximates the best set $\tilde{C}$. This algorithm is developed using quantization theory in general, and codebook design algorithms in particular [66].

A brief description of the algorithm follows. The algorithm begins by providing a good starting m value. It then works on the task of evaluating the vectors in $\tilde{C}$ such that they satisfy equation (4.32) at the selected SNR. There are two important conditions that a $\tilde{C}$ minimizing $E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, Q(\underline{c}_i))]$, and hence satisfying (4.32), will meet. These two conditions are applied, in our algorithm, in an iterative manner.

The details of the iterative algorithm for generating $\tilde{C}$, a modified version of the Generalized Lloyd Algorithm [66, chapter 11], are presented below.

Algorithm 2:

A2.(i) Select an SNR; a good choice for this value is the SNR at the operating point of the system. This algorithm will insure that the set $\tilde{C}$ satisfy equation (4.32) at the selected SNR.

(ii) Select, as a starting value for m, the smallest m value that will satisfy equation (4.32). This can be generated by: (1) create a $P(\epsilon)-m$ curve using the *rate distortion analysis* of the previous section; then, (2) using the $P(\epsilon)-m$ curve, select the smallest m such that the corresponding $P(\epsilon)$ is less than $h\{E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, \underline{c}_i)]\}$.

(iii) Select an initial set $\tilde{C}(0) = \{\underline{c}^1(0), ..., \underline{c}^m(0)\}$; a uniformly spaced set often serves as a good starting point.

(iv) Set $n = 0$.

B2. Generate m sets, called nearest neighbor cells, according to

$$R_j = \{\underline{c}_i \in C_0 : P(\epsilon|\underline{c}_i, \underline{c}^j(n)) < P(\epsilon|\underline{c}_i, \underline{c}^k(n)) \; \forall \; k \neq j\}. \tag{4.33}$$

C2. Establish a new set of values for $\tilde{C}$ according to the centroid condition

$$\underline{c}^j(n+1) = \text{cent}(R_j) = \arg\min_{\underline{c}^j} E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, \underline{c}^j)|\underline{c}_i \in R_j] = \arg\min_{\underline{c}^j} \frac{\int_{R_j} P(\epsilon|\underline{c}_i, \underline{c}^j) p(\underline{c}_i) d\underline{c}_i}{\int_{R_j} p(\underline{c}_i) d\underline{c}_i}. \tag{4.34}$$

D2. Compute the average distortion for the newly derived set $\tilde{C}(n+1) = \{\underline{c}^1(n+1), ..., \underline{c}^m(n+1)\}$ according to

$$D_{n+1} = E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, Q(\underline{c}_i))], \tag{4.35}$$

where $Q(\underline{c}_i)$ is the element in $\tilde{C}(n+1)$ closest to $\underline{c}_i$. This can be evaluated by restating the expectation as an integral, and then using numerical techniques. **If** D_{n+1} is within some small predetermined neighborhood of D_n, go to **E2**; **otherwise**, set $n \leftarrow n+1$, and go to **B2**.

E2. If $D_{n+1} < h(E_{\underline{c}_i}[P(\epsilon|\underline{c}_i, \underline{c}_i)])$, **stop**; **otherwise**, set $m \leftarrow m+1$, set $n \leftarrow n+1$, choose a value for $\underline{c}^{m+1}(n)$, and go to **B2**.$\diamondsuit$

In practice, an alternative ending criteria for this algorithm, i.e., an alternative step **E2**, might be: if the percent change between the distortion achieved using m and the distortion achieved using $m+1$ is below a predetermined threshold, stop.

One of the key computational steps in this algorithm, namely step **B2**, can be greatly simplified in many cases of practical interest. In particular, whenever $P(\epsilon|\underline{c}_i, \underline{c}^j)$ depends on $\underline{c}_i$ only through $|\underline{c}_i - \underline{c}^j|$, and $P(\epsilon|\underline{c}_i, \underline{c}^j)$ takes on a value proportional to $|\underline{c}_i - \underline{c}^j|$, this step can be simplified as follows. The R_j computation of step **B2** can be evaluated using the simple rule

$$R_j = \{\underline{c}_i \in C_0 : |\underline{c}_i - \underline{c}^j(n)| < |\underline{c}_i - \underline{c}^k(n)| \text{ for all } k \neq j\}. \tag{4.36}$$

Part III

Applications

Chapter 5

Data Detection in a Rapidly Changing Phase Environment, $N > 2$

In this chapter, we apply our proposed receiver structure to a particular communication environment of practical interest. The communication environment is described briefly as follows. A differentially encoded MPSK signal is transmitted over a channel introducing both an additive noise and a phase offset. The phase offset is constant over only a few symbols, N, e.g., $N = 3$ (it is assumed that $N > 2$).

The communication system model of interest in this chapter can describe burst mode communications with short burst lengths, mobile communications, and communication environments using frequency hopping. All of these environments are practical, modern-day environments, currently experiencing a rapid growth. As a result, many receivers [8]-[29] have already been proposed for this important communication environment. These receivers, surveyed in the introduction, act as a benchmark for comparison.

We will show that our receiver structure is able to perform as well as the best of the receivers in the literature, and is available at a lower complexity. Specifically, our proposed receiver structure, when applied to the above environment, is able to

achieve a performance matching theoretically optimal bounds, while maintaining a low complexity.

This chapter proceeds in the following manner. We first provide a detailed description of the communication environment under consideration in this chapter. We then introduce the application of our receiver structure to this communication environment, describing in detail the receiver which results, and comparing it, in terms of performance and complexity, to the previously proposed receiver schemes – it is herein that we detail the benefits of our receiver.

5.1 The Communication System Model

This section provides a detailed modeling of the communication environment of interest. The model is shown in Figure 5.1; it is simply a special case of the general communication model shown in Figure 2.1.

The *source* outputs a sequence of binary digits $\underline{b} = (b_1, b_2, ..., b_X)$. These digits represent a voice, video, or data signal. It is assumed that binary digits 0 and 1 are equally likely, i.e., $P(b_i = 0) = P(b_i = 1) = 0.5$.

The *symbol coder*, or *MPSK encoder*, maps the sequence of bits $\underline{b}$ into a sequence of letters $\underline{a} = (a_1, a_2, ..., a_L)$. Specifically, this coder carries out a one-to-one mapping of each set of $n = log_2 M$ bits $(b_{i \cdot n-(n-1)}, ..., b_{i \cdot n})$ into the symbol a_i, $a_i \in A = \{a^1, ..., a^M\}$. The k^{th} term in the set A, a^k, corresponds to a complex value with unit magnitude and a unique phase $2\pi \cdot \frac{k}{M}$; that is, a^k corresponds to

$$a^k = e^{j2\pi \cdot \frac{k}{M}}. \tag{5.1}$$

The a_i's are generated at a rate of one a_i each T seconds.

The *transmit filter* consists of two parts: a *differential encoder* and a *pulse shaping filter* $h_s(t)$. The *differential encoder* carries out a one-to-one mapping of a_i to d_i according to

$$d_i = a_i \cdot d_{i-1}, \tag{5.2}$$

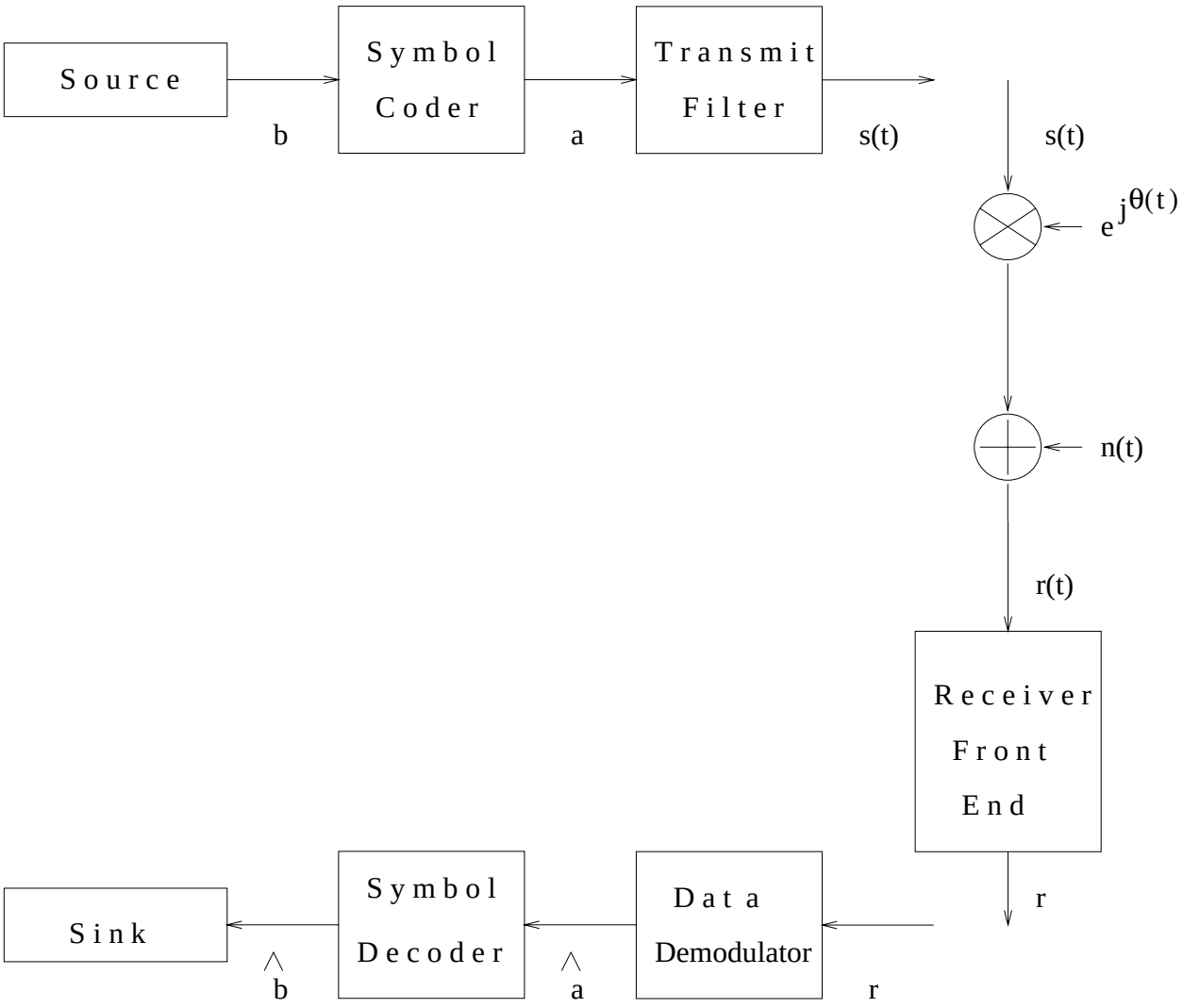

Figure 5.1: The communication system model

where $d_0 = \sqrt{E_s}$. It follows from this equation that each d_i has magnitude $\sqrt{E_s}$ and a phase ϕ_i corresponding to the addition of ϕ_{i-1} with the phase of a_i. This operation is carried out to avoid an M-fold phase ambiguity at the receiver side [2, p.160].

The *pulse shape filter* maps each letter d_i into a waveform ready for transmission over the channel. Specifically, it carries out the one-to-one mapping of each d_i into the waveform $d_i h_s(t - iT)e^{j\omega_c t}$. The entire waveform output by the pulse shape filter, in response to $\underline{d}$, is

$$s(t) = \sum_{i=0}^{L} d_i h_s(t - iT)e^{j\omega_c t}. \tag{5.3}$$

(In practice, $s(t)$ is simply the real component of equation (5.3). However, we use complex notation here because it simplifies the presentation without impacting the receiver design.)

The *physical channel* introduces two forms of degradation. First, the channel introduces a random phase $\theta(t)$. Additionally, the channel adds an additive white Gaussian noise to the transmitted signal. The resulting random process output by the channel is

$$r(t) = s(t)e^{j\theta(t)} + \eta(t). \tag{5.4}$$

It is assumed that the phase $\theta(t)$ is constant over a duration exceeding two symbol intervals.

The *receiver front end* consists of a mixer, which returns the signal to baseband, a matched filter $h_s^*(-t)$, and a sampler, which generates one sample per interval T. Assuming $p_s(t) = h_s(t) * h_s^*(t)$ (where '$*$' denotes convolution) satisfies Nyquist's criteria, then the output of the receiver front end is the sufficient statistic for detection described by [3, p.157]

$$r_i = d_i e^{j\theta_i} + \eta_i. \tag{5.5}$$

Here, η_i is an i.i.d. complex Gaussian noise, with variance $\frac{N_o}{2}$. The phase θ_i is a constant value over each block of N symbols, but its exact value is not known. This θ_i can be characterized as a random variable with statistical characterization

$$p(\theta_i) = \begin{cases} \frac{1}{2\pi}, & \theta_i \in \Theta = [0, 2\pi) \\ 0, & \text{else} \end{cases} ; \tag{5.6}$$

and, whenever θ_i and θ_{i-1} are in the same block of N symbols,

$$p(\theta_i|\theta_{i-1}) = \delta(\theta_i - \theta_{i-1}), \tag{5.7}$$

where $\delta(x)$ is the delta function, a function of value 0 whenever $x \neq 0$, and displaying unit area. Furthermore, if θ_i and θ_{i-1} are not in the same block of symbols, they are assumed independent.

The *demodulator* maps the received samples $\underline{r} = (r_0, r_1, ..., r_L)$ into an estimate of the MPSK sequence $\underline{a}$. This estimate is labeled $\underline{\hat{a}}$.

The following realization allows us to simplify the demodulator operation. Between one block of N received samples and the next, the phases are independent, as are the noise samples. Furthermore, the data symbols a_i are independent of one another, and hence the d_i values are also independent. This implies that the received samples r_i are independent from one block of N samples to the next. It follows, then, that optimal demodulation can be achieved by carrying out demodulation over each block of N received samples independently. The demodulator built for this communication environment can therefore act as if the entire sequence of symbols is of duration N; it simply repeats this assumption for each new N symbols that arrive. Hence, the demodulator operation can be described as a mapping of each N received samples into an estimate of the corresponding MPSK symbols. For the remainder of this chapter, $\underline{r}$ will refer to a block of N symbols, $\underline{\hat{a}}$ will refer to the demodulator's estimate on this block of N symbols, and, in general, vector notation will refer to a vector of length N.

The demodulator is separated into two parts: a *data detector* and a *differential decoder*. The *data detector* maps the received signal $\underline{r}$ into the data sequence $\underline{d}$; the *differential decoder* maps the symbols d_i into the corresponding a_i's. The data detector can be built using the general receiver structure introduced earlier in this thesis. The differential decoder simply implements the inverse to the mapping of a_i to d_i in (5.2), namely

$$a_i = d_i/d_{i-1}; \tag{5.8}$$

this can be implemented as a subtraction of the phase of d_{i-1} from that of d_i.

Finally, a *symbol decoder*, consisting of a simple one-to-one mapping of each symbol $\hat{a}_i$ to the $n = log_2 M$ bits $(\hat{b}_{i \cdot n-(n-1)}, ..., \hat{b}_{i \cdot n})$, completes the receiver. This mapping is simply the inverse of the mapping carried out at the symbol coder. The final output sequence is denoted $\underline{\hat{b}}$.

5.2 Data Detector Based on General Receiver Structure

This section introduces a *data detector* which corresponds to the application of our proposed receiver structure to the case at hand.

5.2.1 Existence

We begin by verifying that the phase offset communication environment, presented in the previous section, satisfies the existence condition of Section 4.2. If it does, then our proposed receiver structure can be applied to this communication environment.

The general existence condition provided in Section 4.2 can be stated, for the case of independent noise samples, according to: our proposed receiver exists in a given communication environment whenever $\lim_{\underline{c}_i' \to \underline{c}_i} P(\epsilon|\underline{c}_i, \underline{c}_i') = P(\epsilon|\underline{c}_i, \underline{c}_i)$ (for every $\underline{c}_i' \in C_0$).

We first restate this condition in terms of the current communication environment. The vector $\underline{c}_i$ represents the nuisance parameters introduced by the channel at time i. In the current environment, this term is simply the single phase value θ_i. Hence, the existence condition is stated here as: our proposed receiver exists if $\lim_{\theta_i' \to \theta_i} P(\epsilon|\theta_i, \theta_i') = P(\epsilon|\theta_i, \theta_i)$.

We now use an important realization to simplify this existence condition. For an MPSK constellation, the probability of error term $P(\epsilon|\theta_i, \theta_i')$ depends only on the difference between θ_i and θ_i', and not on the absolute values of θ_i and θ_i'. That is,

$P(\epsilon|\theta_i, \theta_i')$ can be expressed according to

$$P(\epsilon|\theta_i, \theta_i') = P(\epsilon|\theta_i - \theta_i') = P(\epsilon|\theta_{\epsilon_i}). \tag{5.9}$$

Hence, the existence condition can be restated as: our proposed receiver exists if $\lim_{\theta_{\epsilon_i} \to 0} P(\epsilon|\theta_{\epsilon_i}) = P(\epsilon|0)$.

We now evaluate $P(\epsilon|\theta_{\epsilon_i})$ to determine if it indeed satisfies our existence condition, and hence if our proposed receiver exists. Using geometric arguments and union bounds, we find that, for MPSK, $P(\epsilon|\theta_{\epsilon_i})$ is given by: for $M = 2$

$$P(\epsilon|\theta_{\epsilon_i}) = \frac{1}{2}\text{erfc}(\sqrt{\frac{E_s}{N_o}}\cos(\theta_{\epsilon_i})); \tag{5.10}$$

and, for $M > 2$,

$$P(\epsilon|\theta_{\epsilon_i}) \approx \frac{1}{2}\text{erfc}(\sqrt{\frac{E_s}{N_o}}\sin(\frac{\pi}{M} + \theta_{\epsilon_i})) + \frac{1}{2}\text{erfc}(\sqrt{\frac{E_s}{N_o}}\sin(\frac{\pi}{M} - \theta_{\epsilon_i})). \tag{5.11}$$

The $P(\epsilon|\theta_{\epsilon_i})$ functions, in (5.10) and (5.11), both satisfy $\lim_{\theta_{\epsilon_i} \to 0} P(\epsilon|\theta_{\epsilon_i}) = P(\epsilon|0)$. Consequently, the existence condition, which we restated in terms of $P(\epsilon|\theta_{\epsilon_i})$, is satisfied, and, hence, our receiver structure can be applied to this communication environment.

5.2.2 The Data Detector's Underlying Equation and Implementation

In this section, we introduce a novel equation characterizing the *data detector*, and present a corresponding data detector implementation. The equation and implementation we present correspond to the application of our proposed receiver structure, presented in Chapter 3, to the particular case at hand.

The Data Detector's Underlying Equation

This subsection introduces a novel equation characterizing the data detector. We generate this equation by applying the general detection equations of Chapter 3 to the case at hand.

In Chapter 3, we derived three equations for data detection – (3.9), (3.20), and (3.21). The appropriate detection equation depends on the particular application. In the case at hand, we have independent noise samples and independent data symbols. Consequently, the data detection equation which is appropriate to this case is equation (3.20). This equation states that the data symbols $\underline{\hat{a}}$ should correspond to the values that result from the maximization

$$\max_{\underline{\tilde{c}} \in \tilde{C}^L} \sum_{i=1}^{L} \{[\max_{a_i} \ln \ p(r_i|a_i, \underline{\tilde{c}}_i)] + \ln \ P(\underline{\tilde{c}}_i|\underline{\tilde{c}}_{i-1}, ..., \underline{\tilde{c}}_{i-J})\}. \tag{5.12}$$

The variables in this equation can be expressed in terms of the phase-offset case at hand as follows. First, the data detector attempts to regenerate the sequence of the differentially encoded symbols, d_i, and not the symbol coder symbols, a_i (this task is left for the differential decoder); hence, the a_i term in (5.12) can be replaced by d_i. Second, as noted earlier, the vector $\underline{c}_i$ is the single value θ_i in the case at hand; it follows that $\underline{\tilde{c}}_i$, a vector approximating $\underline{c}_i$, corresponds to the single value $\tilde{\theta}_i$, a value approximating θ_i, in this case. Additionally, the continuous parameter space C_0 becomes the continuous phase space $\Theta = [0, 2\pi)$, and, correspondingly, the discrete approximation to C_0, called $\tilde{C}$, becomes a discrete approximation to Θ, which we label $\tilde{\Theta} = \{\theta^1, ..., \theta^m\}$. Finally, the sequence duration L can be replaced by the block length N. These insights lead to the following data detection equation for the case at hand: choose the sequence $\underline{\hat{d}}$ that results from the maximization

$$\max_{\underline{\tilde{\theta}} \in \tilde{\Theta}^N} [\sum_{i=1}^{N} \{[\max_{d_i} \ln \ p(r_i|d_i, \tilde{\theta}_i)] + \ln \ P(\tilde{\theta}_i|\tilde{\theta}_{i-1}, ..., \tilde{\theta}_{i-J})\}]. \tag{5.13}$$

Furthermore, for the case at hand, the statistics of the θ_i are provided in equations (5.6) and (5.7). Applying this to the above equation leads to: choose $\underline{\hat{d}}$ from the maximization

$$\max_{\tilde{\theta} \in \tilde{\Theta}} \sum_{i=1}^{N} [\max_{d_i} \ln \ p(r_i|d_i, \tilde{\theta})]. \tag{5.14}$$

This equation characterizes the data detector that results from the application of the general data detection equations of Chapter 3.

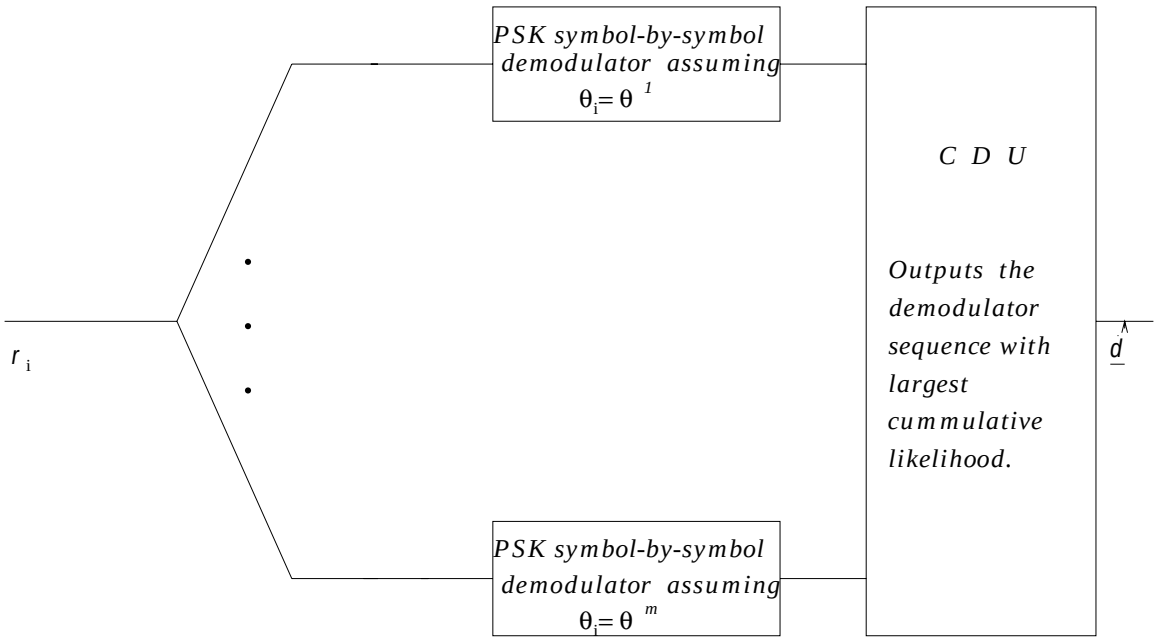

Figure 5.2: The data detector implementation.

Data Detector Implementation

In this subsection, we introduce an implementation for the data detector. This data detector implementation corresponds to a parallel evaluation of equation (5.14).

The detector's implementation is shown in Figure 5.2. This implementation corresponds to the general structure provided in Chapter 3 (Figure 3.1), when applied to the case at hand.

The components in the data detector of Figure 5.2 are now summerized. There are two main components – the universal set of demodulators, and the CDU. The universal set of demodulators carry out the inner maximization in data detection equation (5.14), a maximization over the data symbols; meanwhile, the CDU performs the outer maximization of detection equation (5.14), maximizing over channel phase.

The operation of the universal set of demodulators is now described in detail. Each of the m demodulators in the universal set of demodulators acquires the sample r_i every T seconds. The j^{th} demodulator assumes that the channel phase θ_i, present

in r_i, has a value θ^j, regardless of sample time i. This j^{th} demodulator, using r_i and $\theta_i = \theta^j$, generates an ML decision on d_i. This decision corresponds to

$$\hat{d_i^j} = \arg\max_{d_i} p(r_i|d_i, \theta^j). \tag{5.15}$$

The j^{th} demodulator also generates the corresponding likelihood value

$$l_i^j = \max_{d_i} p(r_i|d_i, \theta^j) = p(r_i|\hat{d_i^j}, \theta^j). \tag{5.16}$$

Each demodulator sends its output symbol, along with the corresponding likelihood value, to the CDU.

The CDU generates its output from among the demodulators' decisions. It selects an output sequence $\underline{\hat{d}}$ according to the outer maximization of equation (5.14). Using equation (5.15) and (5.16), the operation of the CDU can be written in terms of the demodulators' outputs according to: over each block of duration N, the CDU chooses N d_i's according to

$$\underline{\hat{d}} = (\hat{d}_1^{J^*}, ..., \hat{d}_N^{J^*}) \tag{5.17}$$

where the value $J^* \in \{1, 2, ..., m\}$ is generated using

$$J^* = \arg\max_{J} \sum_{i=1}^{N} \ln(l_i^J). \tag{5.18}$$

The CDU implements (5.17) and (5.18) as follows. The CDU outputs, over each block of duration N, N symbols generated from a single demodulator – the demodulator with the largest sum of log probabilities $\sum_{i=1}^{N} \ln(l_i^j)$.

5.2.3 The Discrete Space $\tilde{\Theta}$

The data detection equation and corresponding data detector implementation provided above are incomplete. We have not yet established a set of values $\tilde{\Theta} = \{\theta^1, ..., \theta^m\}$ for use in the data detector. This section sets out to generate this $\tilde{\Theta}$ by applying the algorithms provided in Chapter 4 to the case at hand.

In what follows, we establish the $\tilde{\Theta}$ in three parts. First, we redefine the continuous phase space Θ, using a realization regarding differential decoding to reduce its size.

Second, applying the algorithm introduced in Section 4.3, we examine the size of $\tilde{\Theta}$, m, and establish the range of m values that can lead to a near-optimal performance. Finally, we apply the algorithm introduced in Section 4.4 and establish the set of phase values for $\tilde{\Theta}$.

Redefining the Continuous Phase Space Θ

The space $\tilde{\Theta}$ represents a discrete approximation to the continuous phase space Θ. Before we establish $\tilde{\Theta}$, we first redefine the continuous space Θ. The continuous phase space was introduced in Section 5.1 as $\Theta = [0, 2\pi)$. However, in the case at hand, our data detector is followed by a differential decoder. It is well known that a differential decoder resolves an M-fold phase ambiguity; that is, it maps the information bearing phase to the correct $[\frac{2\pi \cdot i}{M}, \frac{2\pi \cdot (i+1)}{M})$ sector of space. Hence, it suffices, for the purposes of our data detector, to represent the continuous phase space by $\Theta = [0, \frac{2\pi}{M})$, because: the only effect of this assumption at the detector is an information bearing phase in the wrong $\frac{2\pi}{M}$ phase sector, and the differential decoder can correct for this (by mapping the information bearing phase to the correct $\frac{2\pi}{M}$ sector of space).

This redefining of the phase space Θ is very important in our receiver implementation: it allows us to use a set $\tilde{\Theta}$ (to approximate Θ) that has the same size regardless of the constellation size, M. This is because increased phase sensitivity at larger M values is now offset by a decreased continuous phase space.

The Size of $\tilde{\Theta}$

We now examine the size of the set $\tilde{\Theta} = \{\theta^1, ..., \theta^m\}$; that is, we examine possible values for m. Specifically, we provide $P(\epsilon) - m$ curves, and, using these curves, we establish m values that can achieve a quality $P(\epsilon)$ performance.

The $P(\epsilon) - m$ curves are generated by applying the algorithm provided in Section 4.3 to the case at hand. Fortunately, under the title 'Application of the $P(\epsilon) - m$ Algorithm' in Subsection 4.3.3, we already applied the algorithm to the unknown phase case of interest in this chapter. Curves indicating the $P(\epsilon)$ vs m relationship,

at SNR's of 13 dB, 16 dB, and 18 dB, and with $M = 8$, are provided in Figures 4.6, 4.5, and 4.7, respectively.

These curves clearly indicate that performances very close to that attained with $m \to \infty$ can be achieved with small, finite values for m, e.g., $m = 8$. This implies that the proposed data detector can be implemented using only a few parallel demodulators — our receiver implementation is realizable.

The Discrete Phase Space $\tilde{\Theta}$

In this section, we establish the set $\tilde{\Theta} = \{\theta^1, \theta^2, ..., \theta^m\}$ for use in our novel data detector. We do this by applying the general algorithm provided in Section 4.4 to the phase case at hand. In what follows, we assume $M = 8$.

The algorithm of Section 4.4 begins with an initialization step, **A2**, which requests that we: (a) select an SNR; and (b) generate a starting m value, by (b1) deciding on an $h(s)$ for use in equation (4.32), and (b2) evaluating the smallest m at which it is possible to satisfy equation (4.32). We begin by selecting an SNR of 18 dB. We then decide that we want $h(s) = 2 \cdot s$ in (4.32). Using this $h(s)$, the SNR of 18 dB, and the $P(\epsilon) - m$ curve of Figure 4.7, we find that the smallest possible m value that can satisfy (4.32) is $m = 5$.

The algorithm then goes on to a two step iterative process, steps **B2** and **C2** with stopping criteria **D2**. These steps generate a good set of m values for $\tilde{\Theta}$.

Steps **B2**, **C2**, and **D2** can be simplified in the case at hand. This is because it is a special case: the continuous space being approximated is a finite space, namely $\Theta = [0, \frac{2\pi}{M})$; and, $p(\theta_i)$ is a uniform random variable. In this case (following the example of [66, p.183]), the set of m phase values that results from **B2**, **C2**, and **D2** are, simply, the set of phases uniformly spaced over the interval $\Theta = [0, \frac{2\pi}{M})$. That is, in the special case at hand, the two step iterative process **B2** and **C2**, with stopping criteria **D2**, results in

$$\theta^j = \frac{2\pi}{M} \cdot \frac{2 \cdot j - 1}{2 \cdot m}; \tag{5.19}$$

i.e., for the starting $m = 5$ case, $\tilde{\Theta} = \{\frac{\pi}{4}\frac{1}{10}, \frac{\pi}{4}\frac{3}{10}, ..., \frac{\pi}{4}\frac{9}{10}\}$.

The algorithm then moves to step **E2**. Step **E2** states that we should check the performance achieved by the $\tilde{\Theta}$ of steps **B2** to **D2**, and determine if it matches (or exceeds) the performance criteria of (4.32). In the case at hand, we find that the performance criteria of (4.32) is not satisfied with $m = 5$. With this realization, step **E2** requires we increase m by one and return to redo the iterations of steps **B2** to **D2**, which in this case simplifies to using equation (5.19).

After a few computations of equation (5.19) and distortion comparisons in step **E2**, we find: $m = 8$, with the discrete space $\tilde{\Theta} = \{\frac{\pi}{4}\frac{1}{16}, \frac{\pi}{4}\frac{3}{16}, ..., \frac{\pi}{4}\frac{15}{16}\}$, satisfies the performance criteria. This ends the algorithm, and provides us with a good set of values for $\tilde{\Theta}$.

5.3 Performance and Complexity

In this section, we detail the performance and complexity of the data detector introduced in the Section 5.2, a data detector corresponding to the application of our general receiver structure. The performance is characterized in terms of probability of error as a function of SNR, while complexity is measured in operations per symbol.

5.3.1 Performance

The performance of our receiver, in terms of $P(\epsilon) - SNR$ plots, is generated by simulation. By simulation, we are referring to Monte Carlo simulation carried out on computer using C code. In cases of low error probability, importance sampling [67]-[69] was used to reduce the simulation times.

Figure 5.3 contains our simulated performance results for an MPSK constellation of size M=8. Each set of points in this figure corresponds to the simulated performance results with a different value of block length N: the '+' set is for $N = 2$, the 'x' set for $N = 3$, the '*' set for $N = 5$, and the 'o' set is for $N = 10$.

Figure 5.3 also contains a series of short solid lines, one line plotted alongside each set of simulation points. These lines correspond to the theoretical bounds on

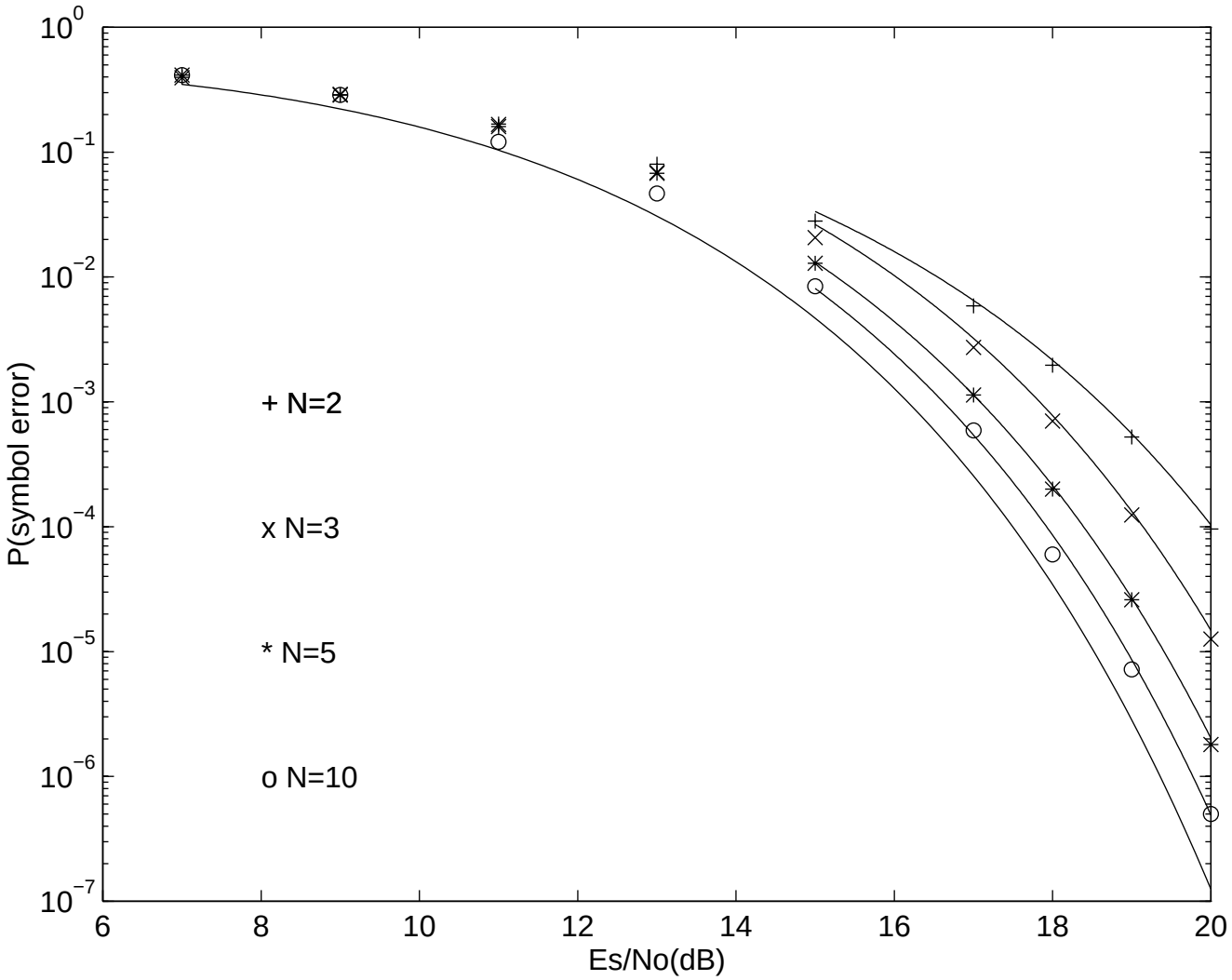

Figure 5.3: $P(\epsilon) - \frac{E_s}{N_o}$ for 8-PSK, m=8.

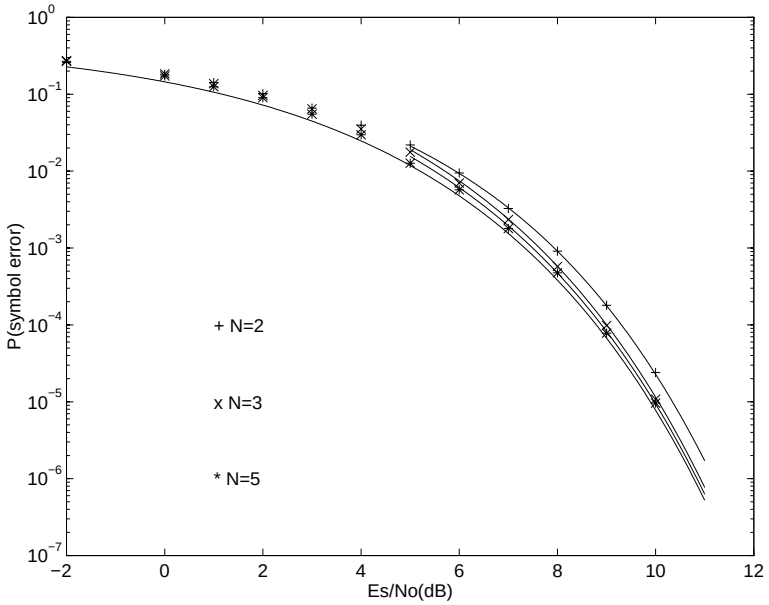

Figure 5.4: $P(\epsilon) - \frac{E_s}{N_o}$ for BPSK, m=8.

performance, given a block of N received symbols with an unknown phase [22]. These lines clearly highlight the fact that the performance of our receiver matches the theoretically attainable performance.

Figure 5.3 also contains a long solid line running just below all the simulation points. This curve corresponds to the performance of a coherent receiver employing a differential encoder-decoder. We see that as N increases, even slightly, from the value 2, the performance results move rapidly toward coherent.

Figures 5.4, 5.5, and 5.6 show the simulated performance results for an MPSK constellation of size $M = 2$, 4, and 16, respectively. These curves highlight the fact that the performance of our data detector matches the theoretically attainable performance regardless of constellation size.

5.3.2 Complexity

A key consideration in determining the usefulness of a receiver is its complexity. A simple receiver with a performance matching optimal would be a very attractive

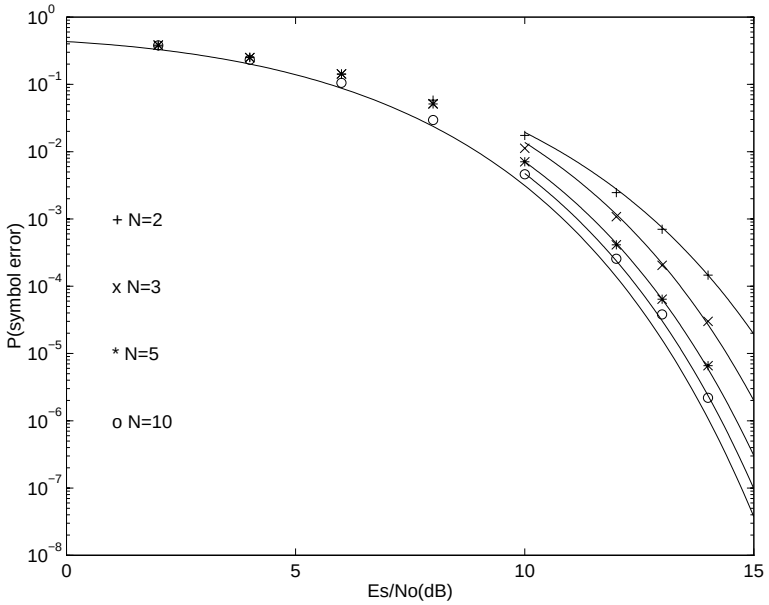

Figure 5.5: $P(\epsilon) - \frac{E_s}{N_o}$ for QPSK, m=8.

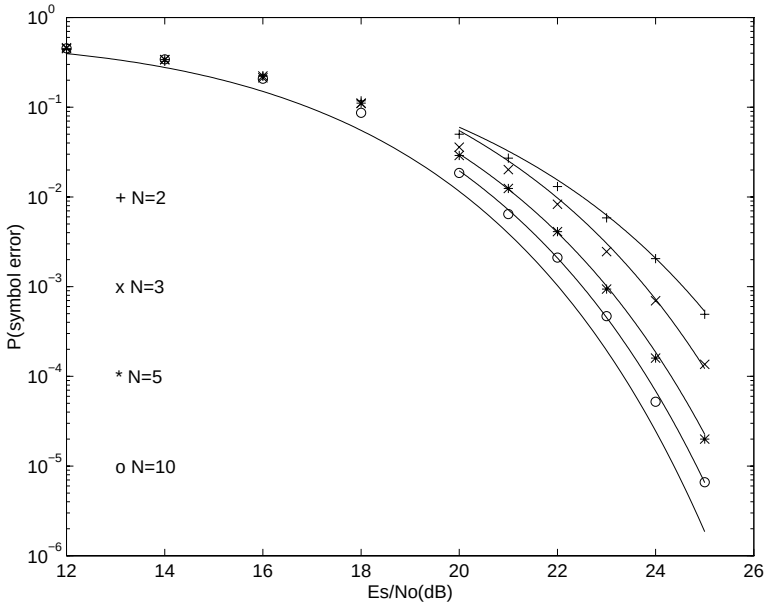

Figure 5.6: $P(\epsilon) - \frac{E_s}{N_o}$ for 16-PSK, m=8.

package.

The data detector which we propose consists of two main computational components. First, 8 demodulators carry out 8 MPSK symbol-by-symbol demodulations, and compute 8 values for $p(r_i|\hat{d}_i^j, \theta^j)$, or, alternatively, 8 values for $Re\{r_i^* e^{j\theta^j} \hat{d}_i^j\}$. Second, the CDU carries out 8 additions per received symbol, and $m-1$ comparisons every N^{th} symbol. Hence, for each received symbol, the data detector carries out 8 MPSK demodulations, 8*2 additions, 8*2 multiplies, and $\frac{m-1}{N}$ of a comparison. This is a low complexity, and one which is essentially independent of N.

5.3.3 Comparison with Other Proposed Receivers

In this subsection, we compare the receiver introduced in this chapter to other receivers proposed to date. Our comparison is one of performance and complexity.

First, we consider performance. Our receiver is able to achieve a performance matching theoretically optimal results for any N value. Hence, no scheme, considering a block of N symbols with an unknown phase, can outdo the performance of our receiver. However, MSDD [22]-[25] and Mackenthun's low complexity version of MSDD [29] also achieve a performance matching optimal for any N value.

The second point of comparison is complexity. The total number of computations that our receiver must carry out, for each decoded symbol in a block of N symbols, is essentially independent of N. The complexity of the MSDD scheme, on the other hand, increases exponentially with increasing block size. Hence, our scheme is far more attractive than MSDD.

Furthermore, Mackenthun's low complexity implementation of MSDD requires more computations than our proposed receiver. Specifically, Mackenthun's receiver and our proposed receiver demonstrate a comparable number of additions and comparisons, but our receiver avoids the magnitude operations required by Mackenthun's receiver, operations that involve complex multiplications.

Chapter 6

Data Detection in a Rapidly Changing Phase Environment, $N = 2$

In this chapter, we again apply our proposed receiver structure to a particular communication environment of practical interest; here, the communication environment models mobile communications.

The communication environment under consideration in this chapter is closely related to the environment introduced in Chapter 5. In Chapter 5, we presented an environment where a differentially encoded MPSK signal was sent across a channel introducing two disturbances, additive noise and channel phase. The channel phase was modeled as constant over a block of $N > 2$ symbols. In this chapter, we again consider the environment in which a differentially encoded MPSK symbol is sent across a channel introducing noise and a phase. The key difference is the modeling of the phase: here, the phase is constant over two symbols, and not necessarily any more. That is, in this chapter, we are interested in the communication environment presented in Chapter 5, with the one difference: the phase is assumed to change at a faster rate, a rate such that phase change over two symbols is negligible, but it is not necessarily negligible over a longer interval.

Only a few receivers [8][14][22]-[25] are currently available to detect data in the above environment. These receivers, summerized in the introduction, will act as a benchmark for comparison. Specifically, we will show that our receiver structure, applied to the case at hand, easily outperforms all the receivers currently available, while maintaining a low complexity.

This chapter proceeds as follows. First, we provide a detailed modeling of the communication environment. We then present the application of our proposed receiver structure to this environment: we detail the receiver that results, provide an analysis of this receiver in terms of complexity and performance, and explain the benefits of this receiver when compared to other receivers available to date.

6.1 The Communication Environment Model

In this subsection, we provide a detailed model of the communication environment of interest. Figure 6.1 displays the communication environment model, and, in what follows, we explain each component in this model.

The descriptions of the first few components in the communication model of Figure 6.1 match the descriptions provided in Section 5.1. The descriptions of the *source, MPSK encoder,* and *transmit filter* (consisting of a *differential encoder* and a *pulse shape filter*) are identical to those in Section 5.1, and, hence, for brevity, these descriptions are not repeated here.

The difference between the communication model in Chapter 5 and that of this chapter begins at the *channel*. Here, as in Chapter 5, the channel introduces two forms of degradation, noise and channel phase, and the channel outputs

$$r(t) = s(t)e^{j\theta(t)} + \eta(t). \tag{6.1}$$

However, in a break from the modeling of Chapter 5, it is assumed here that $\theta(t)$ is constant over two symbol intervals, and not necessarily constant over a longer interval.

The *receiver front end* of Figure 6.1 generates the output

$$r_i = d_i e^{j\theta_i} + \eta_i, \tag{6.2}$$

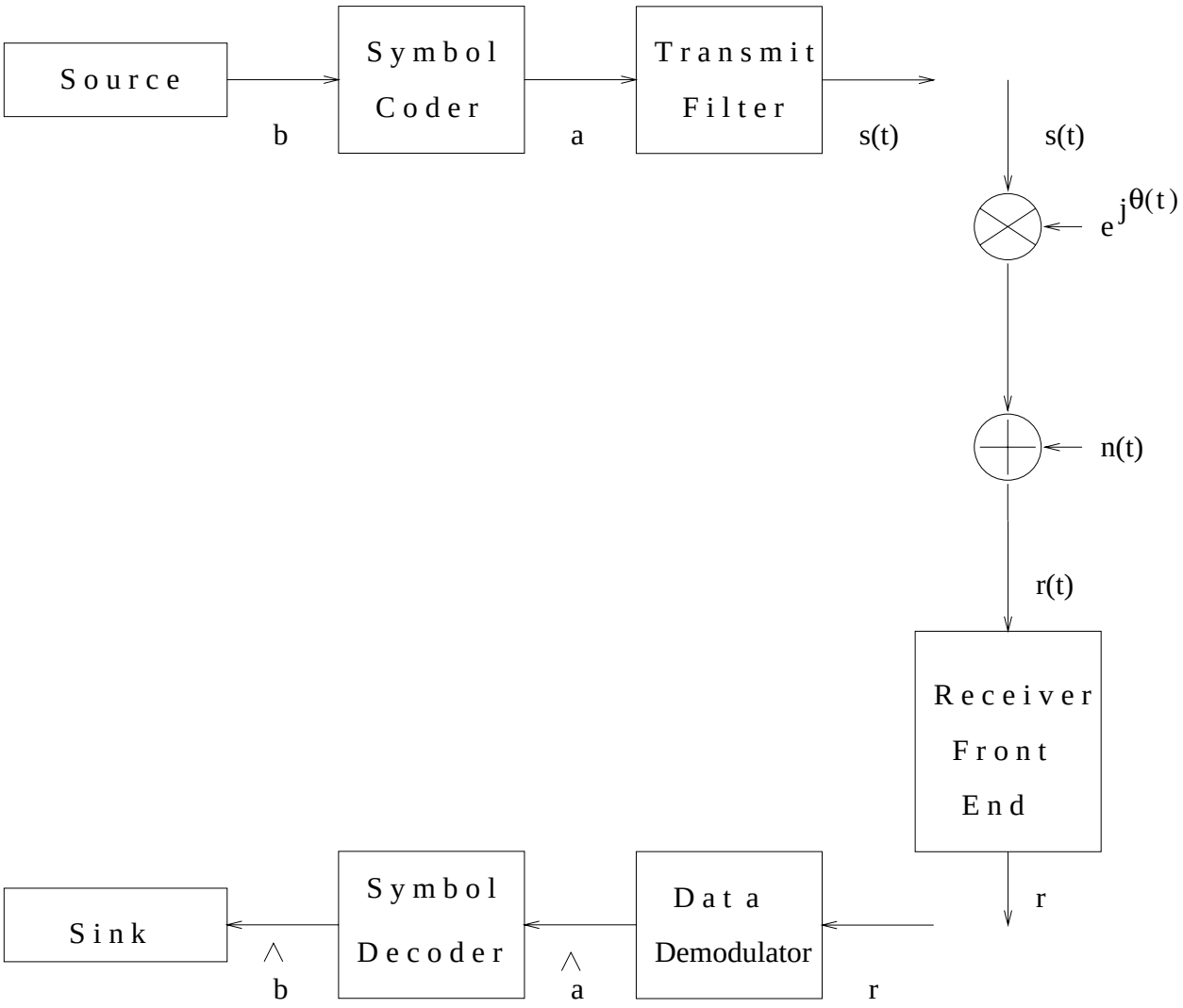

Figure 6.1: The communication system model.

identical in appearance to the r_i in Chapter 5. However, a key difference exists in the modeling of the phases θ_i. Here, the phases θ_i can only be assumed constant over two symbols.

We want to generate a statistical characterization of the channel phases θ_i present in r_i. We proceed as follows. First, the phase value at any sample time i, θ_i, when considered in isolation, is an unknown value between $[0, 2\pi)$. This θ_i can be characterized as a random variable with a distribution given by

$$p(\theta_i) = \begin{cases} \frac{1}{2\pi}, & \theta_i \in \Theta = [0, 2\pi) \\ 0, & \text{otherwise} \end{cases} . \tag{6.3}$$

Additionally, the phase change $(\theta_i - \theta_{i-1})$ is negligible; that is, mathematically, $|\theta_i - \theta_{i-1}| < \beta$, where β is small. A good choice for β is the largest value at which DPSK [8] still performs well; we have found that this β is $\frac{2\pi}{8 \cdot M}$. Hence, the phase change can be described as $|\theta_i - \theta_{i-1}| < \beta = \frac{2\pi}{8 \cdot M}$. We represent this information statistically by assuming that phase change is uniformly distributed in the range $|\theta_i - \theta_{i-1}| < \beta$, i.e.,

$$p(\theta_i | \theta_{i-1}) = \begin{cases} \frac{1}{2\beta}, & |\theta_i - \theta_{i-1}| \leq \beta = \frac{2\pi}{8 \cdot M} \\ 0, & \text{otherwise} \end{cases} . \tag{6.4}$$

Equations (6.3) and (6.4) characterize the channel phase θ_i in the received sample r_i of (6.2).

Continuing with the description of the components in Figure 6.1, the *demodulator* maps the received samples $\underline{r} = (r_0, r_1, ..., r_L)$ into an estimate of the MPSK sequence $\underline{a}$. This estimate is labeled $\underline{\hat{a}}$. This is done using two components: a *data detector* and a *differential decoder*. The *data detector* maps $\underline{r}$ into an estimate of the data sequence $\underline{d}$. This detector can be built by applying our proposed receiver structure to the case at hand. The *differential decoder* maps d_i's into a_i's, carrying out the inverse of the mapping at the differential encoder.

Finally, a *symbol decoder*, or *MPSK decoder*, recreates the binary sequence from the received symbols $\underline{\hat{a}}$. It does this by implementing the inverse of the MPSK encoder's one-to-one mapping.

6.2 Data Detector Based on our Proposed Receiver Structure

This section introduces a data detector which corresponds to the application of our proposed receiver structure to the case at hand.

6.2.1 Existence

We begin by determining if the communication environment in hand, presented in Section 6.1, satisfies the existence condition introduced in Section 4.2. If it does, then our proposed receiver can be applied to this communication environment.

The existence condition introduced in Section 4.2 states, for cases of independent noise samples: in a given communication environment, our proposed receiver exists if $\lim_{\underline{c}_i' \to \underline{c}_i} P(\epsilon|\underline{c}_i, \underline{c}_i') = P(\epsilon|\underline{c}_i, \underline{c}_i)$. In the communication environment at hand, $\underline{c}_i$ corresponds to the single phase value θ_i; hence, the existence condition can be restated here according to: in the environment at hand, our proposed receiver exists if $\lim_{\theta_i' \to \theta_i} P(\epsilon|\theta_i, \theta_i') = P(\epsilon|\theta_i, \theta_i)$.

We now want to determine whether $P(\epsilon|\theta_i, \theta_i')$ satisfies the above condition. Fortunately, we can determine this without any computation; we simply look at the results achieved in Chapter 5. There, in Subsection 5.2.1, we demonstrated that the $P(\epsilon|\theta_i, \theta_i')$ indeed satisfies $\lim_{\theta_i' \to \theta_i} P(\epsilon|\theta_i, \theta_i') = P(\epsilon|\theta_i, \theta_i)$. Hence, in the case at hand, the existence condition is satisfied; consequently, we can apply our receiver structure to this communication environment.

6.2.2 The Data Detector's Underlying Equation – Part 1

In this subsection, we introduce an equation characterizing the operation of the data detector. This equation is generated by applying the general detection equations of Chapter 3, equations which lead to our proposed receiver structure, to the case at hand.

Chapter 3 introduced three equations for data detection, namely (3.9), (3.20), and (3.21). In the case at hand, with independent noise samples and independent data symbols, the detection equation which is applicable is equation (3.20). This equation states: choose the data sequence $\underline{\hat{a}}$ from the joint maximization

$$\max_{\underline{\tilde{c}} \in \tilde{C}^L} \sum_{i=1}^{L} \{[\max_{a_i} \ \ln \ p(r_i|a_i, \underline{\tilde{c}}_i)] + \ln P(\underline{\tilde{c}}_i|\underline{\tilde{c}}_{i-1}, ..., \underline{\tilde{c}}_{i-J})\}. \tag{6.5}$$

We now restate this equation using the terminology of the phase case at hand. First, the nuisance parameter vector $\underline{c}_i$ corresponds to the single phase value θ_i in this case. This implies that the vector $\underline{\tilde{c}}_i$, an approximation to $\underline{c}_i$, corresponds to the single value $\tilde{\theta}_i$, an approximation to θ_i. Also, the nuisance parameter space C_0 becomes the phase space $\Theta = [0, 2\pi)$ in the case at hand. Consequently, $\tilde{C}$, a discrete space approximation to the continuous space C_0, becomes $\tilde{\Theta} = \{\theta^1, ..., \theta^m\}$, a discrete phase space approximation to Θ. Finally, the data detector here generates an estimate of d_i, rather than a_i. Using the above realizations, we can rewrite (6.5) according to: choose the data sequence $\underline{\hat{d}}$ from the joint maximization

$$\max_{\underline{\tilde{\theta}} \in \tilde{\Theta}^L} \sum_{i=0}^{L} \{[\max_{d_i} \ \ln \ p(r_i|d_i, \tilde{\theta}_i)] + \ln P(\tilde{\theta}_i|\tilde{\theta}_{i-1}, ..., \tilde{\theta}_{i-J})\}. \tag{6.6}$$

It is possible to further simplify the detection equation in (6.6). This is done by introducing equation (6.4) into detection equation (6.6), using the following two steps. First, derive the discrete probability mass function $P(\tilde{\theta}_i|\tilde{\theta}_{i-1})$ from the probability density function $p(\theta_i|\theta_{i-1})$ of equation (6.4). Second, replace the term $P(\tilde{\theta}_i|\tilde{\theta}_{i-1}, ..., \tilde{\theta}_{i-J})$ in the detection equation by this $P(\tilde{\theta}_i|\tilde{\theta}_{i-1})$; this can be done because the $P(\tilde{\theta}_i|\tilde{\theta}_{i-1})$ term represents all the information available regarding the discrete phase.

We are currently unable to carry out the simplification to the data detection equation (6.6) described above. This is because we have not yet defined the discrete space $\tilde{\Theta}$, and hence we can not generate $P(\tilde{\theta}_i|\tilde{\theta}_{i-1})$ from $p(\theta_i|\theta_{i-1})$. So, we leave the derivation of the data detection equation for the time being, and proceed as follows. In the next subsection, we define the discrete space $\tilde{\Theta}$. We then return and complete the derivation of the data detection equation, the equation underlying the operation of our data detector.

6.2.3 The Discrete Space $\tilde{\Theta}$

In this subsection, we generate a discrete phase space $\tilde{\Theta}$ for use at the data detector. We do this by first redefining the continuous phase space Θ, and then applying the algorithms of Chapter 4 to the case at hand.

Redefining the Continuous Phase Space Θ

The discrete space $\tilde{\Theta}$ represents a discrete approximation to the continuous phase space $\Theta = [0, 2\pi)$. We begin the process of generating $\tilde{\Theta}$ by redefining this continuous phase space Θ.

In the communication model at hand, the data detector is followed by a differential decoder. Hence, for reasons already explained in Chapter 5 (Subsection 5.2.3), it suffices, for the purposes of our data detector, to represent the continuous phase space by $\Theta = [0, \frac{2\pi}{M})$.

The Size of $\tilde{\Theta}$

In this subsection, we evaluate sizes of the space $\tilde{\Theta} = \{\theta^1, ..., \theta^m\}$, i.e., values for m. Specifically, we generate the sizes of $\tilde{\Theta}$ (i.e., m values) that, when used at the data detector, allow for a performance near that attained with $m \to \infty$.

In what follows, we determine the m values in two steps. First, we apply the algorithm of Section 4.3 to the phase case at hand; this leads to $P(\epsilon) - m$ curves indicating the smallest possible m that can achieve a stated $P(\epsilon)$ performance. Second, reading from these curves, we determine the range of m values that achieve a performance near that attained with $m \to \infty$.

The algorithm provided in Section 4.3 generates curves characterizing the $P(\epsilon) - m$ performance. This algorithm, which stems from rate distortion theory, has an outcome dictated by two functions: the probability of error term $P(\epsilon|\underline{c}_{\epsilon_i})$, and the probability density $p(\underline{c}_i)$. In the case at hand, the two functions that determine the algorithm's outcome can be restated as $P(\epsilon|\theta_{\epsilon_i})$ and $p(\theta_i)$.

An important realization allows us to generate the $P(\epsilon) - m$ curves from the algorithm without any computation. We recognize that the functions $P(\epsilon|\theta_{\epsilon_i})$ and $p(\theta_i)$, in the communication model of this chapter, are identical to the corresponding functions in the communication environment of Chapter 5. Hence, the algorithm generates the same outcome ($P(\epsilon) - m$ curves) in both the communication model of Chapter 5 and that of this chapter. It follows that the $P(\epsilon) - m$ curves shown in Figures 4.5, 4.6 and 4.7 are also applicable to the communication model at hand.

The $P(\epsilon) - m$ curves indicate that small values of m can be used at the data detector to achieve performances very close to optimal. For instance, all these curves indicate that $m \geq 8$ achieves a performance very close to that attained with $m \to \infty$.

The Elements of $\tilde{\Theta}$

In this subsection, we establish the exact size of, and the elements in, the discrete set $\tilde{\Theta} = \{\theta^1, ..., \theta^m\}$. We achieve this by applying the algorithm of Section 4.4 to the case at hand.

The algorithm of Section 4.4, applied to the case at hand, will generate an outcome $\tilde{\Theta}$ dictated by two distributions: $P(\epsilon|\theta_{\epsilon_i})$ and $p(\theta_i)$. This realization allows us to establish the $\tilde{\Theta}$ (from the algorithm of Section 4.4) without any computation. Specifically, we simply recognize that the terms $P(\epsilon|\theta_{\epsilon_i})$ and $p(\theta_i)$ in the communication model of this chapter are the same as those in the communication model of Chapter 5. Hence, the $\tilde{\Theta}$ result achieved in Chapter 5 (by applying the algorithm of Section 4.4) is also applicable to the communication model of this chapter.

We now restate the $\tilde{\Theta}$ result achieved in Chapter 5 for convenience. From Chapter 5, we have: the discrete set of phases $\tilde{\Theta} = \{\theta^1, ..., \theta^m\}$ will always have the form

$$\theta^j = \frac{2\pi}{M} \cdot \frac{2j-1}{2 \cdot m}, \tag{6.7}$$

where the exact value for m depends on the constellation size M, the SNR, and the stopping criteria. Specifically, for $M = 8$, at an SNR of 18 dB, and with a stopping criteria based on $h(s) = 2 \cdot s$, we have, from Chapter 5, that $m = 8$, and hence $\tilde{\Theta}$ is

$$\tilde{\Theta} = \{\frac{2\pi}{16} \cdot \frac{1}{16}\ ,\ \frac{2\pi}{16} \cdot \frac{3}{16}\ ,\ ...,\ \frac{2\pi}{16} \cdot \frac{15}{16}\}. \tag{6.8}$$

We have also found that, for other constellation sizes, a good choice for $\tilde{\Theta}$ are the $m = 8$ uniformly spaced phases

$$\tilde{\Theta} = \{\frac{2\pi}{M}\frac{1}{16}\ ,\ \frac{2\pi}{M}\frac{3}{16}\ ,\ ...,\frac{2\pi}{M}\frac{15}{16}\}; \tag{6.9}$$

that is,

$$\theta^j = \frac{2\pi}{M} \cdot \frac{2 \cdot j - 1}{16}. \tag{6.10}$$

6.2.4 The Data Detector's Underlying Equation – Part 2

In this subsection, with the discrete phase space $\tilde{\Theta}$ in hand, we establish a final equation to characterize the data detector.

We began creating a data detection equation in Subsection 6.2.2, and got as far as: choose the $\underline{\hat{d}}$ that results from the joint maximization

$$\max_{\underline{\tilde{\theta}} \in \tilde{\Theta}^L} \sum_{i=0}^{L} \{[\max_{d_i}\ \ln\ p(r_i|d_i, \tilde{\theta}_i)] + \ln P(\tilde{\theta}_i|\tilde{\theta}_{i-1}, ..., \tilde{\theta}_{i-J})\}. \tag{6.11}$$

We stopped here because we did not know the discrete space $\tilde{\Theta}$. With $\tilde{\Theta}$ now at hand, we simplify equation (6.11) and generate a final detection equation.

The final detection equation is generated by applying the phase statistics of equation (6.4) to equation (6.11). We begin this process by restating equation (6.4) here for convenience:

$$p(\theta_i|\theta_{i-1}) = \begin{cases} \frac{1}{2\beta}, & |\theta_i - \theta_{i-1}| \le \beta = \frac{2\pi}{8 \cdot M} \\ 0, & \text{otherwise} \end{cases} . \tag{6.12}$$

When the phases $\theta_i \in \Theta$ are approximated by the discrete phases $\tilde{\theta}_i \in \tilde{\Theta}$, the corresponding probability mass function describing $\tilde{\theta}_i$ is

$$P(\tilde{\theta}_i|\tilde{\theta}_{i-1}) = \begin{cases} \frac{1}{3}, & \tilde{\theta}_i - \tilde{\theta}_{i-1} \in \{-\Delta\tilde{\theta}, 0, \Delta\tilde{\theta}\} \\ 0, & \text{otherwise} \end{cases} , \tag{6.13}$$

where $\Delta\tilde{\theta} = \frac{2\pi}{8 \cdot M}$, i.e., $\Delta\tilde{\theta}$ is the difference between two neighboring phases in $\tilde{\Theta}$. This equation represents all the available information regarding the channel phase, when it is approximated to the discrete phase space $\tilde{\Theta}$. We can substitute (6.13) into

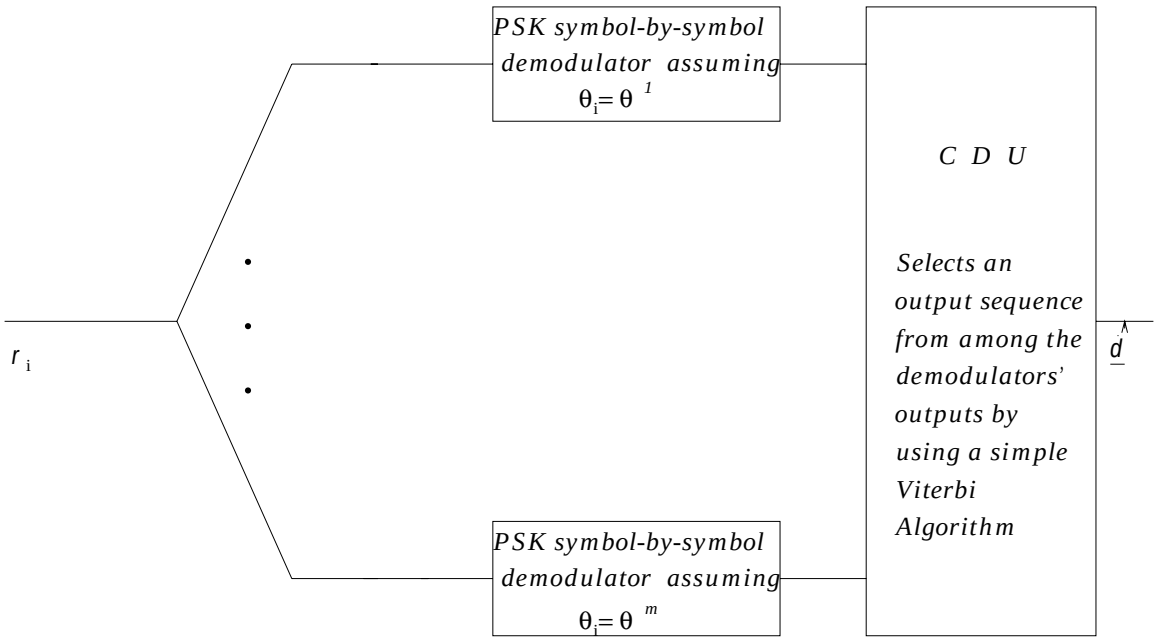

Figure 6.2: The data detector implementation

detection equation (6.11), using (6.13) to replace the $P(\tilde{\theta}_i|\tilde{\theta}_{i-1}, ..., \tilde{\theta}_{i-J})$ term. This leads to: choose $\underline{\hat{d}}$ from the joint maximization

$$\max_{\underline{\tilde{\theta}} \in \tilde{\Theta}^L} \sum_{i=0}^{L} \{\max_{d_i} [\ln\ p(r_i|d_i, \tilde{\theta}_i)]\} \text{ subject to } \tilde{\theta}_i - \tilde{\theta}_{i-1} \in \{-\Delta\tilde{\theta}, 0, \Delta\tilde{\theta}\}. \tag{6.14}$$

This equation characterizes the operation of the data detector generated by applying the general receiver equations of Chapter 3 to the case at hand.

6.2.5 The Data Detector Implementation

In this subsection, we introduce the implementation of the data detector. This implementation corresponds to a parallel evaluation of the data detection equation (6.14).

The implementation is shown in Figure 6.2. This implementation matches the general receiver structure in Chapter 3 (Figure 3.1).

The Components

We begin with a brief description of the components of the data detector implementation. The data detector implementation consists of two main components: the universal set of demodulators and the CDU. The universal set of demodulators carry out the inner maximization of equation (6.14), a maximization over the data symbols d_i. Meanwhile, the CDU carries out the outer optimization, optimizing over the phase sequence $\underline{\tilde{\theta}}$.

We now describe the universal set of demodulators in detail. The j^{th} demodulator in the set of demodulators assumes that θ_i has a value of θ^j, regardless of the time index i. This j^{th} demodulator, using r_i, and this claim of $\theta_i = \theta^j$, generates, at each time i, the decision

$$\hat{d_i^j} = \arg\max_{d_i} p(r_i|d_i, \theta^j). \tag{6.15}$$

The j^{th} demodulator also generates the corresponding likelihood value

$$l_i^j = \max_{d_i} p(r_i|d_i, \theta^j) = p(r_i|\hat{d_i^j}, \theta^j). \tag{6.16}$$

These two values are sent to the CDU.

We want to point out that the universal set of demodulators described above are identical to the universal set of demodulators introduced in the data detector of Chapter 5. This is because the demodulators carry out the inner maximization of the detection equation, and this inner maximization is identical in the case at hand and in the environment of Chapter 5. The difference between the data detector for the case at hand and that of Chapter 5 lies with the CDU's operation, which we examine next.

The CDU selects its output symbols from among the many demodulator decisions. It generates its output based on the outer maximization of equation (6.14). The CDU's operation can be stated in terms of the demodulator outputs as follows. The CDU generates the sequence

$$\underline{\hat{d}} = (\hat{d}_0, ..., \hat{d}_L),\ \hat{d}_i = \hat{d}_i^{j_i^*}, \tag{6.17}$$

where $j_i^* \in \{1, 2, ..., m\}$ is the index of the demodulator whose output is selected at time i. This value is generated according to the sequence long maximization

$$\underline{j}^* = (j_0^*, ..., j_L^*) = \arg\max_{\underline{j}} \sum_{i=0}^{L} \ln(l_i^{j_i}) \text{ subject to } (j_i - j_{i-1}) \in \{-1, 0, 1\}. \quad (6.18)$$

We now describe the CDU's operation in a way which suggests an easy construction. First, we explore the constraint $(j_i - j_{i-1}) \in \{-1, 0, 1\}$. This states that if the decision of demodulator k (the demodulator assuming θ^k) is selected by the CDU at one time (i.e., $j_i^* = k$), then at the next time the CDU will select as its output the decision generated by either demodulator $k-1$ (assuming θ^{k-1}), demodulator k (assuming θ^k), or demodulator $k+1$ (assuming θ^{k+1}). That is, the CDU chooses which demodulators' decisions to output under the restriction that the only discrete phase transitions permitted are as indicated in Figure 6.3.

Equations (6.17) and (6.18) state that the CDU outputs the sequence of decisions corresponding to the path through the trellis of Figure 6.3 which maximizes $\sum_{i=0}^{L} \ln(l_i^{j_i})$; here, $l_i^{j_i}$ are the likelihood values generated by the $m = 8$ demodulators. This can be implemented using a rather simple Viterbi Algorithm (VA).

A Slight Revision for Phase Movement from one $\frac{2\pi}{M}$ Sector to Another

In this subsection, we introduce a shortcoming of the data detector implementation, and then introduce an update to the implementation, namely an update to the CDU, which allows us to overcome this shortcoming.

The Issue The underlying equation, and corresponding implementation, of our data detector is based on the approximation of the continuous phase space Θ by the discrete space $\tilde{\Theta}$. In the underlying equation and implementation generated above, the $\tilde{\Theta}$ we used assumed that $\Theta = [0, \frac{2\pi}{M})$.

The equality $\Theta = [0, \frac{2\pi}{M})$ was generated in Subsection 6.2.3; it was based on the understanding that a differential decoder immediately follows our data detector, and that this decoder is able to undo phase ambiguities of $\frac{2\pi}{M}$. This understanding allowed us to use $\Theta = [0, \frac{2\pi}{M})$ at the data detector rather than the usual $\Theta = [0, 2\pi)$.

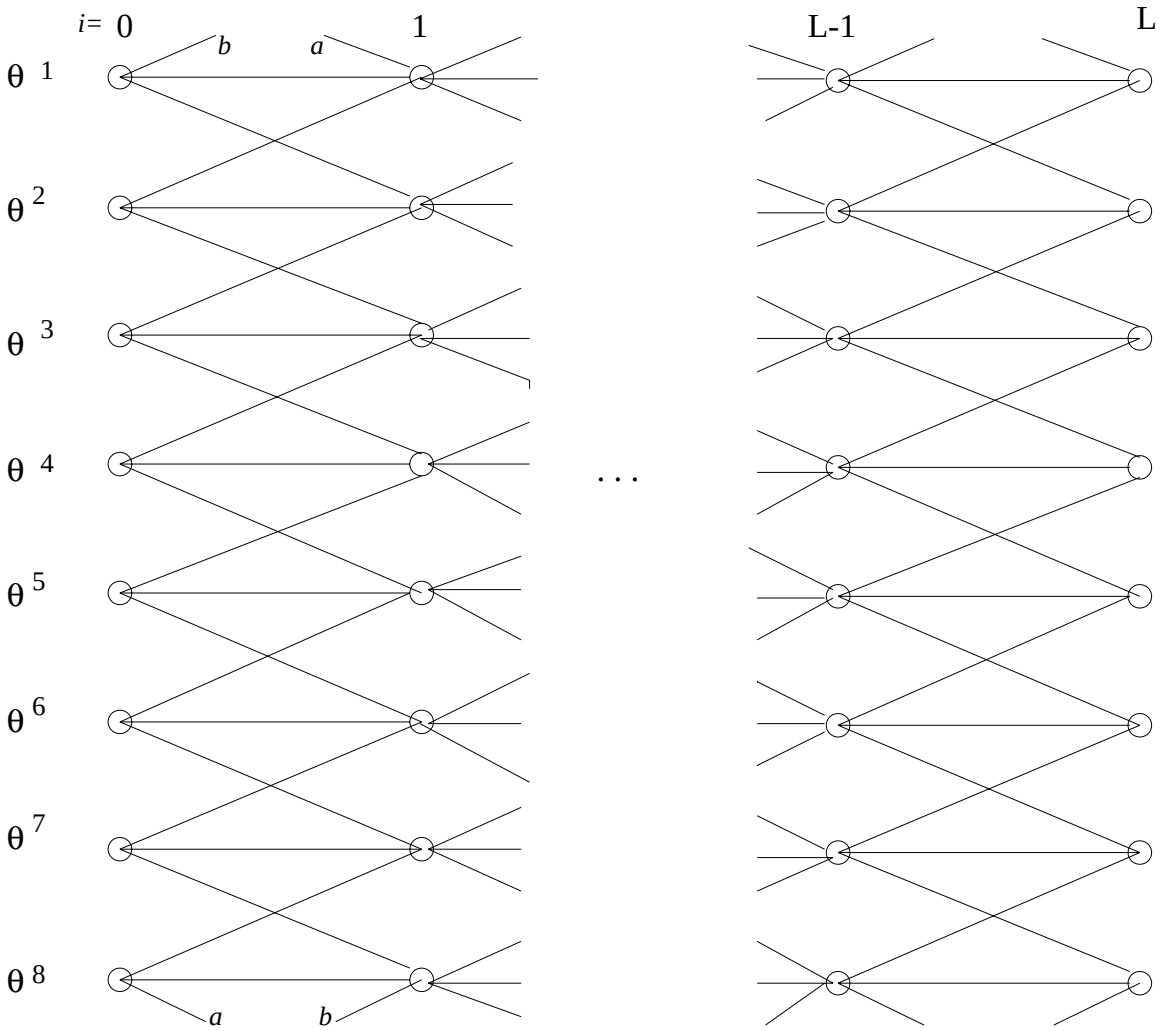

Figure 6.3: The trellis of permitted phases.

However, at sample times i when θ_{i-1} and θ_i are in different $\frac{2\pi}{M}$ sectors of the phase space $[0, 2\pi)$ (e.g., $\theta_{i-1} \in [0, \frac{2\pi}{M})$ and $\theta_i \in [\frac{2\pi}{M}, \frac{2\pi \cdot 2}{M})$), the differential decoder is unable to undo phase ambiguities of $\frac{2\pi}{M}$. At these sample times i, the phase space representation $\Theta = [0, \frac{2\pi}{M})$ is not valid, and the usual $\Theta = [0, 2\pi)$ should be used at the data detector. Unfortunately, the data detector we introduced assumes $\Theta = [0, \frac{2\pi}{M})$ at all sample times i. Hence, at times i, when θ_{i-1} and θ_i are in different $\frac{2\pi}{M}$ sectors, our data detector will introduce errors.

Resolution by Modified CDU Implementation In this subsection, we introduce a minor change in the implementation of the CDU. This allows the data detector to avoid the errors that it would otherwise make at times i when θ_{i-1} and θ_i move between $\frac{2\pi}{M}$ phase sectors.

In brief, we modify the CDU implementation as follows. The CDU is modified to insure that the data sequence $\underline{\hat{d}}$, arriving at the differential decoder, matches the sequence that would arrive if no θ_{i-1} to θ_i movements between $\frac{2\pi}{M}$ sectors occurred. Hence, the differential decoder is, at all times, able to undo phase ambiguities of $\frac{2\pi}{M}$, and our data detector is always applicable.

In what follows, we detail the change we make to the CDU's implementation. We begin by assuming that the best path through the CDU's trellis of Figure 6.3 tracks phase correctly. Then, when the phase movement $\theta_{i-1} \rightarrow \theta_i$ is between $\frac{2\pi}{M}$ phase sectors, the phases of the CDU's trellis move either from $\theta^8 \rightarrow \theta^1$, or from $\theta^1 \rightarrow \theta^8$. Knowing this, the CDU is able to update the data sequence $\underline{\hat{d}}$, sent to the differential decoder, so that, as far as the differential decoder is concerned, all the channel phases θ_i remain in the same $\frac{2\pi}{M}$ sector of space.

6.3 Complexity and Performance

In this section, we present the complexity and performance of the data detector generated in Section 6.2 – the data detector representing the application of our proposed receiver structure to the case at hand.

6.3.1 Complexity

We begin by introducing the complexity of our data detector. Our measure of complexity will be computations per decoded symbol.

The computations of our data detector can be divided into two parts. First, there is the computation carried out by the bank of demodulators. Second, there are the computations of the CDU.

In the bank of $m = 8$ demodulators, at each sample time i, each demodulator generates $\hat{d}_i^j$ and l_i^j. The $\hat{d}_i^j$ is evaluated using an MPSK symbol-by-symbol demodulation. The l_i^j computation can be replaced by the evaluation of $Re\{r_i^* \hat{d}_i^j e^{j\theta^j}\}$, which requires only 2 multiplications and 1 addition.

The CDU finds the best path through a trellis made up of 8 nodes, one node for each discrete phase $\theta^j \in \tilde{\Theta}$. Three branches enter into each one of these nodes. When using a VA to determine the best path through this trellis, one requires, at each time i, 8 comparisons of 3 values each, and 8 additions.

Putting it all together, the overall computation required by our data detector, at each sample time i, is: 8 MPSK symbol-by-symbol demodulations, 16 multiplies, 16 additions, and 8 comparisons of 3 values each. That is, at each time i, the computational requirement of the data detector is in the order of 8 times that of a symbol-by-symbol PSK decoder.

6.3.2 Performance

In this subsection, we present the performance of our data detector, achieved by computer simulation. This performance is displayed using plots showing probability of symbol error, $P(\epsilon)$, versus $\frac{E_s}{N_o}$.

The computer-simulated performance results are presented in two parts. First, we introduce the communication environment model we used in our computer simulations. Then, we describe the performance results generated by simulation, and compare these results to the performances of DPSK [8] and MSDD [22]-[25].

The Communication Environment used in Simulation

The communication model was shown in Figure 6.1, and was described in Section 6.1. To generate simulation results, we made a few assumptions regarding the variables in this model. This subsection details the assumptions that we made.

First, we selected a size for the MPSK constellation. We chose $M = 8$.

Next, we selected a channel phase model. We wanted to choose a model which was practical, and one which allowed us to achieve a θ_i constant over two symbol intervals, but not necessarily constant over a longer interval. We chose the channel phase model described by

$$\theta_i = \theta_{i-1} + w_i, \tag{6.19}$$

where w_i represent samples from an AWGN process. This models channel phase, θ_i, as a random walk on a circle.

This phase model is practical. Specifically, this rather general channel phase model has been employed successfully in modeling slowly fading channels, as well as heterodyne optical communications and telephone communications [70].

Additionally, this phase model can be used to achieve a θ_i constant over two symbol intervals, but not necessarily constant over a longer interval. This is done by the choice of the variance of w_i, which we label σ_w^2. Specifically, in our simulations, we generate performance results at the following σ_w^2. We first consider $\sigma_w^2 = 0$; in this case, the channel phases θ_i are constant. We also consider $\sigma_w^2 = 0.0009$, 0.0025, 0.0049, and 0.0081 radians^2; or, equivalently, a standard deviation of $\sigma_w = 0.03$, 0.05, 0.07, and 0.09 radians. In degrees, $\sigma_w = 1.71$, 2.88, 4.01, and 5.15 degrees. In these cases, the phase movement from θ_{i-1} to θ_i is within 1.71, 2.88, 4.01, and 5.15 degrees, respectively, 60% of the time, and within 6.84, 11.53, 16.04, and 20.6 degrees, respectively, 98% of the time.

The implementation of our data detector, which we consider in this simulation, assumes $L = 100$. If the data sequence is longer than $L = 100$, than the data detector just runs over each new set of $L = 100$ symbols. This is a slightly suboptimal approximation to our data detector, but it keeps the memory requirement of the VA

fairly low, and, at $L = 100$, the end effects of the VA are negligible.

There exists an alternative way to implement the data detector, which would also keep the memory requirement of the VA low. Here, we allow the data detector to run over the entire sequence of length L, but we implement the VA at the CDU using a lookback depth of 12 to 15. This implementation is also a slightly suboptimal version of our proposed data detector.

The Performance: Curves and Analysis

In this subsection, we present the performance curves for our data detector. We compare these curves to the performance curves of DPSK, a receiver using the previous symbol as a phase reference [8], and to the performance curves of MSDD, a receiver extending the ideas of DPSK to blocks of N symbols [22]-[25]. Our performance curves are generated using the communication system modeling described in the previous subsection.

The performance curves for our data detector are presented in Figures 6.4 to 6.11. In Figures 6.4 to 6.8, the performance of our data detector is plotted alongside the performance curves for DPSK and coherent detection. In Figures 6.9 to 6.11, the performance of our data detector is plotted alongside the curves for DPSK, coherent detection, MSDD using $N = 3$, MSDD using $N = 5$, and MSDD using $N = 10$.

Figure 6.4 presents the performance curve of our data detector, along with the curves for DPSK and coherent detection, when the channel phase θ_i is constant from symbol to symbol. This curve indicates that, at a $P(\epsilon)$ of 10^{-3} and 10^{-4}, our detector achieves gains of 1.41 and 1.43 dB respectively over DPSK.

Figures 6.5, 6.6, and 6.7 present the performance curves of our data detector, along with DPSK and coherent detection curves, when the phase movement has variance 0.0009, 0.0025, and 0.0049. Here, our data detector again demonstrates gains over DPSK: at a $P(\epsilon)$ of 10^{-3}, we gain 1.40 dB, 1.53 dB, and 1.46 dB, respectively; and at a $P(\epsilon)$ of 10^{-4}, we gain 1.72 dB, 1.50 dB, and 1.32 dB.

Figure 6.8 presents the performance curves for our data detector, DPSK, and coherent detection, when the phase movement displays a variance of $\sigma_w^2 = 0.0081$. In

this case, our scheme performs poorly, at times performing worse than DPSK. However, this is a case where phase change is so rapid that even DPSK is not applicable.

The results presented in Figures 6.4 to 6.8 can be summerized as follows. Overall, over the range of phase changes where DPSK is applicable, our scheme is able to outperform DPSK by about 1.5 dB at $P(\epsilon)$'s of 10^{-3} and 10^{-4}.

Our data detector's improvement over DPSK can be explained as follows. The detector we implement considers a longer phase history than DPSK. That is, DPSK assumes a channel phase constant over two symbols r_i and r_{i-1}, but it ignores the implied phase correlation that would then exist between r_i and r_{i-2}. Our detector considers this phase correlation via the trellis of Figure 6.3, and thus gains over DPSK.

Figures 6.9, 6.10, and 6.11 present the performance of our data detector alongside the performance of MSDD with $N = 3$, $N = 5$, and $N = 10$. Figure 6.9 displays these performances when the phase θ_i is constant from symbol to symbol. It is seen that, in this case, the performance of our detector is superior to that of MSDD with $N = 3$, matches that of MSDD with $N = 5$, but is inferior to MSDD using $N = 10$.

Figure 6.10 displays the performance of our detector alongside that of MSDD when the channel phase θ_i changes slowly ($\sigma_w^2 = 0.0009$). In this case, our detector is again superior to MSDD with $N = 3$, and is again as good as MSDD with $N = 5$. MSDD with $N = 10$ no longer outperforms our data detector. The performance of MSDD with $N = 10$ deteriorates quite rapidly as phase variation increases, because its assumption of constant phase over $N = 10$ symbols does not hold any longer.

Figure 6.11 presents the performance of our data detector along with that of MSDD when channel phase changes rapidly ($\sigma_w^2 = 0.0049$). In this case, the performance of our detector is always substantially better than MSDD. This is because the constant phase assumptions of MSDD prove unrealistic, even at lower values of N.

The results presented in Figures 6.9 to 6.11 can be summerized as follows. These results demonstrate that our data detector is at least as good as MSDD under slow phase change conditions, and outperforms MSDD under rapid phase changes.

The gains of our detector over MSDD can be explained as follows. MSDD assumes that channel phase is constant over N symbols, where typical values for N are values

such as 3, 5, and 10. Hence, as phase change becomes more and more rapid, MSDD's performance degrades quickly. Our scheme considers a block of L symbols, where L is very large (e.g., $L = 100$), and, unlike MSDD, we do not assume a constant phase over the block, but rather a correlated one. This leads to two effects. For a truly constant phase (not a practical situation in many environments), MSDD with $N = 10$ will outperform our scheme because its assumption of constant phase is better than our assumption of correlated phase. However, once there is a phase movement, even a small one, we achieve a better performance, because our assumption of correlated phase is superior, and, additionally, we consider a longer history of phase.

6.3.3 The Benefits of Our Data Detector – Complexity and Performance

The application of our general receiver structure, to the communication environment of Section 6.1, leads to the data detector described in Section 6.2. This data detector is able to achieve substantial performance gains over MSDD and DPSK, as explained above. The complexity of this data detector is in the order of 8 times that of a coherent PSK receiver. Hence, in any environment in which a DPSK or MSDD receiver is currently employed, this receiver can be replaced by our data detector, resulting in substantial performance gains at a practical complexity.

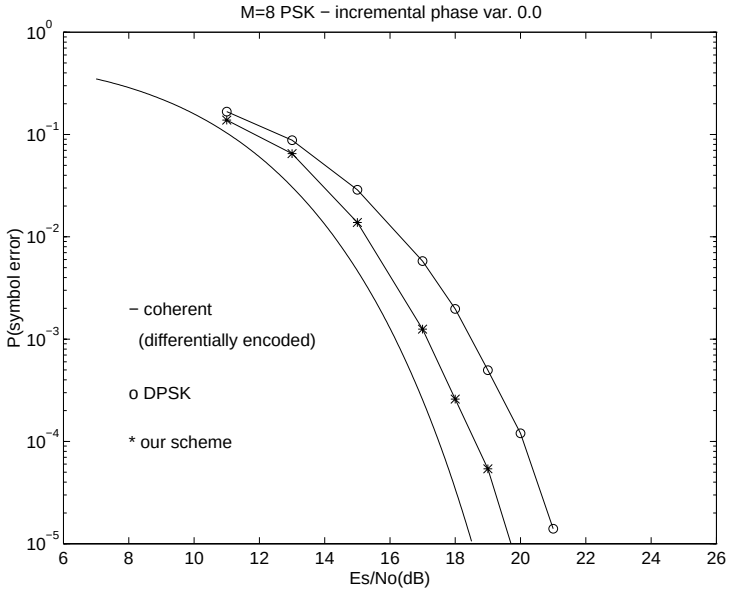

Figure 6.4: $P(\epsilon) - \frac{E_s}{N_o}$ with $\sigma = 0.0$.

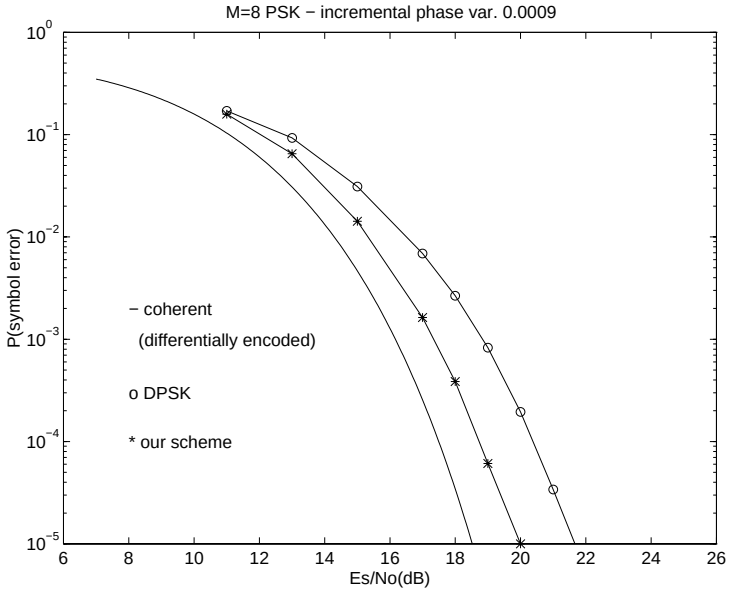

Figure 6.5: $P(\epsilon) - \frac{E_s}{N_o}$ with $\sigma = 0.03$.

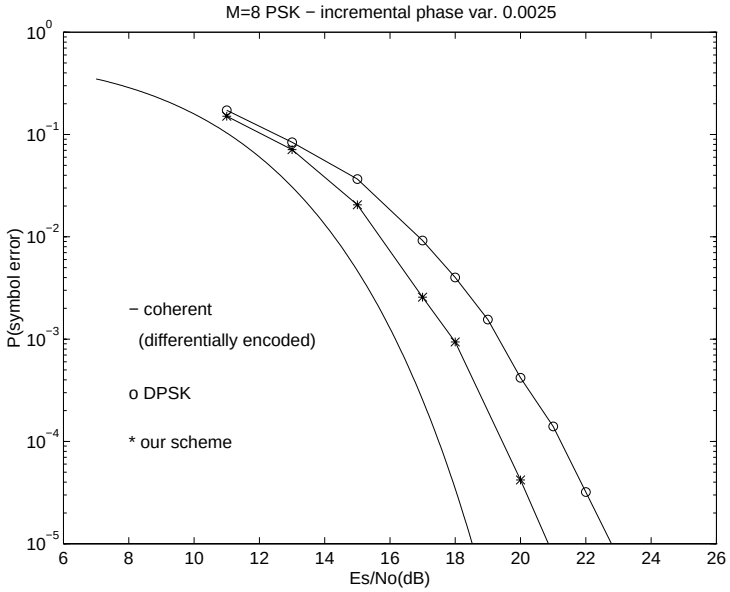

Figure 6.6: $P(\epsilon) - \frac{E_s}{N_o}$ with $\sigma = 0.05$.

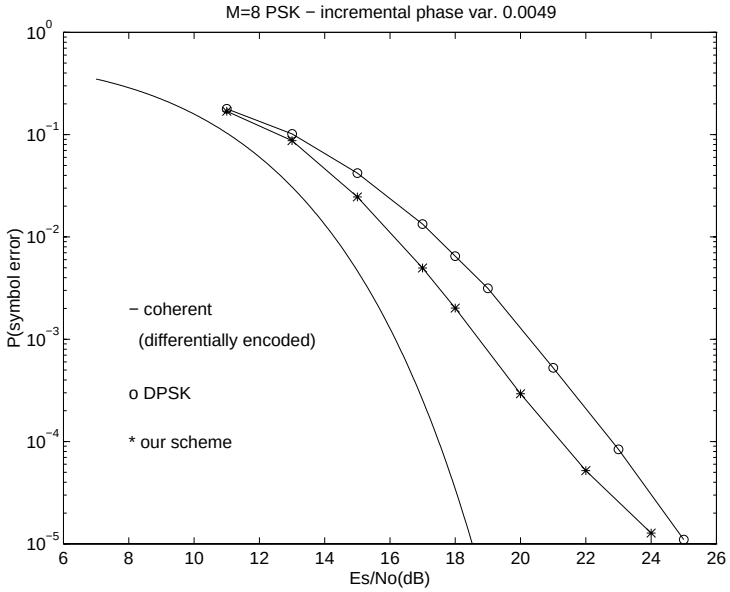

Figure 6.7: $P(\epsilon) - \frac{E_s}{N_o}$ with $\sigma = 0.07$.

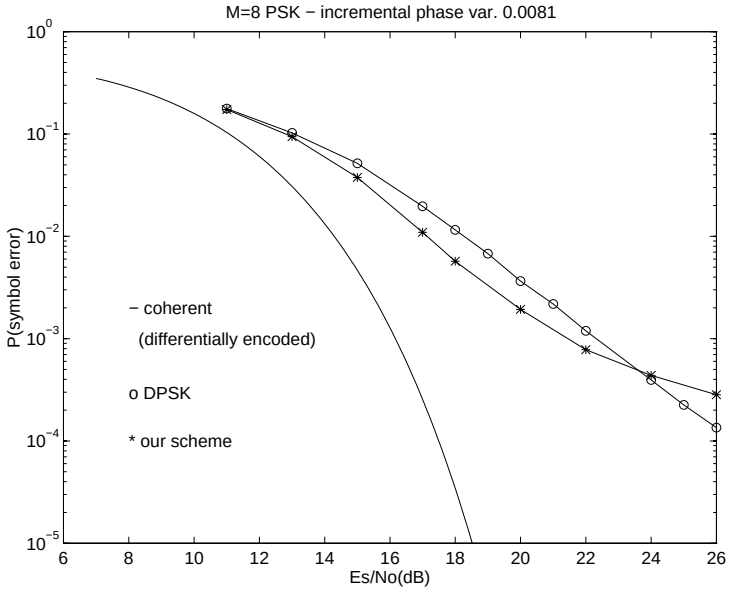

Figure 6.8: $P(\epsilon) - \frac{E_s}{N_o}$ with $\sigma = 0.09$.

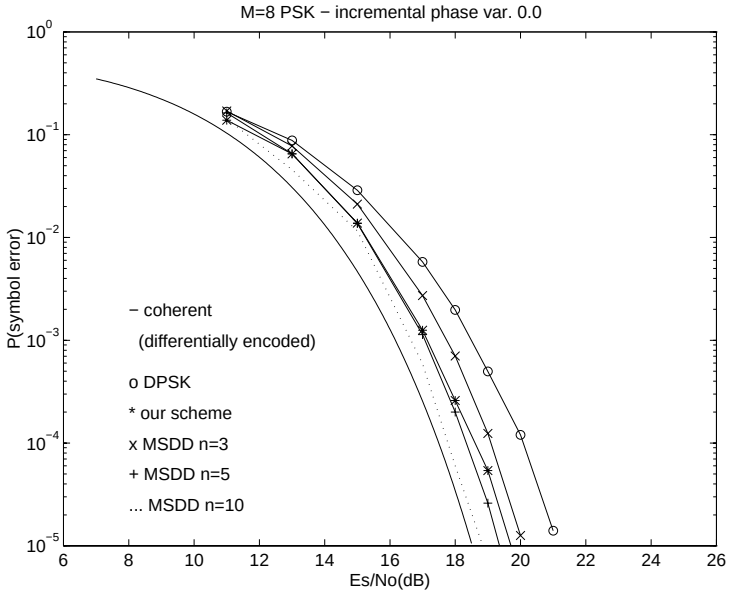

Figure 6.9: $P(\epsilon) - \frac{E_s}{N_o}$ with $\sigma = 0.0$; also included are MSDD curves.

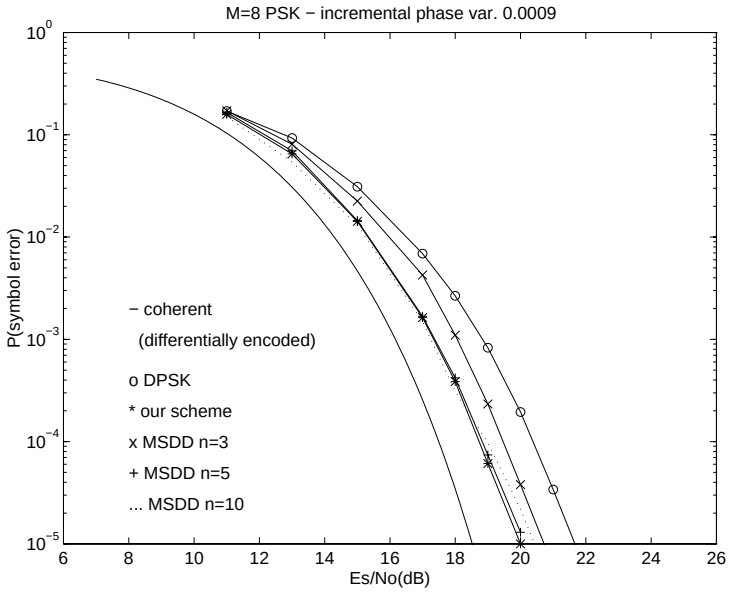

Figure 6.10: $P(\epsilon) - \frac{E_s}{N_o}$ with $\sigma = 0.03$; also included are MSDD curves.

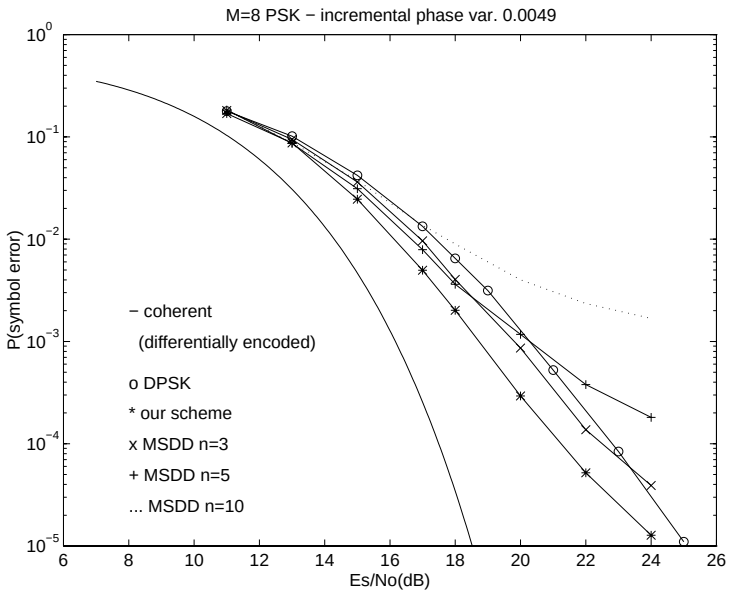

Figure 6.11: $P(\epsilon) - \frac{E_s}{N_o}$ with $\sigma = 0.07$; also included are MSDD curves.

Chapter 7

Data Detection of Coded Modulation in a Rapidly Changing Phase Environment

This chapter presents the application of our proposed receiver structure to another practical, modern-day communication environment.

The communication environment of interest in this chapter is similar to the environments presented in the previous two chapters. Specifically, in the earlier two chapters, we considered a communication environment wherein an MPSK signal was sent across a channel introducing both a noise and a channel phase. In this chapter, we examine the communication environment wherein a *coded* MPSK signal is sent across the same channel, adding noise and phase. Here, the channel phase is assumed to be constant over a short block of N symbols, $N > 2$.

The communication environment of interest in this chapter is not yet well summerized; what remains is a brief introduction to *coded* MPSK, which we provide here. In 1982, Ungerboeck [71] introduced Trellis Coded Modulation (TCM), a joint channel coding and modulation scheme. This scheme offers a considerable performance gain over uncoded modulations, a gain achieved without the cost of increased bandwidth, but rather at a cost of increased complexity. *Coded* MPSK is a subset of TCM;

specifically, coded MPSK refers to TCM when the modulation format used in TCM corresponds to MPSK.

Several receivers [13][30]-[37] are currently available for data detection in the above communication environment. These receivers, surveyed in the introduction, serve as a benchmark for comparison. We show that in many important cases of practical interest, our receiver structure is able to outperform the receivers in current literature, gaining in performance or complexity.

This chapter proceeds as follows. We begin with a detailed description of the communication environment. With this in hand, we present the application of our receiver structure.

7.1 The Communication Environment Model

This section presents a detailed model of the communication environment of interest. The model is shown in Figure 7.1; in what follows, we provide a description of each component.

The *source* creates a sequence (in time) of binary digits. These are labeled $\underline{b} = (b_1, b_2, ..., b_X)$.

The *channel encoder* and *symbol coder* are closely linked, and together they map the bit sequence $\underline{b}$ into the symbol sequence $\underline{a} = (a_1, ..., a_L)$. The symbol sequence $\underline{a}$ is generated from $\underline{b}$ by following the two mapping rules [36]

$$a_i = g(\underline{b}_i, \varsigma_i) \tag{7.1}$$

and

$$\varsigma_{i+1} = f(\underline{b}_i, \varsigma_i). \tag{7.2}$$

Here, $\underline{b}_i$ refers to the bits used by the mapping at time i. Specifically, $\underline{b}_i$ corresponds to the set of n bits $(b_{i\cdot n-(n-1)}, ..., b_{i\cdot n})$, where the value of n is usually between 1 and 4. Also in (7.1) and (7.2), ς_i refers to the state at time i, and a_i refers to the MPSK symbol $a_i = \sqrt{E_s}e^{j\frac{2\pi}{M}l_i}$, $l_i \in \{1, 2, ..., M\}$. The two mappings $f(\cdot,\cdot)$ and $g(\cdot,\cdot)$ insure

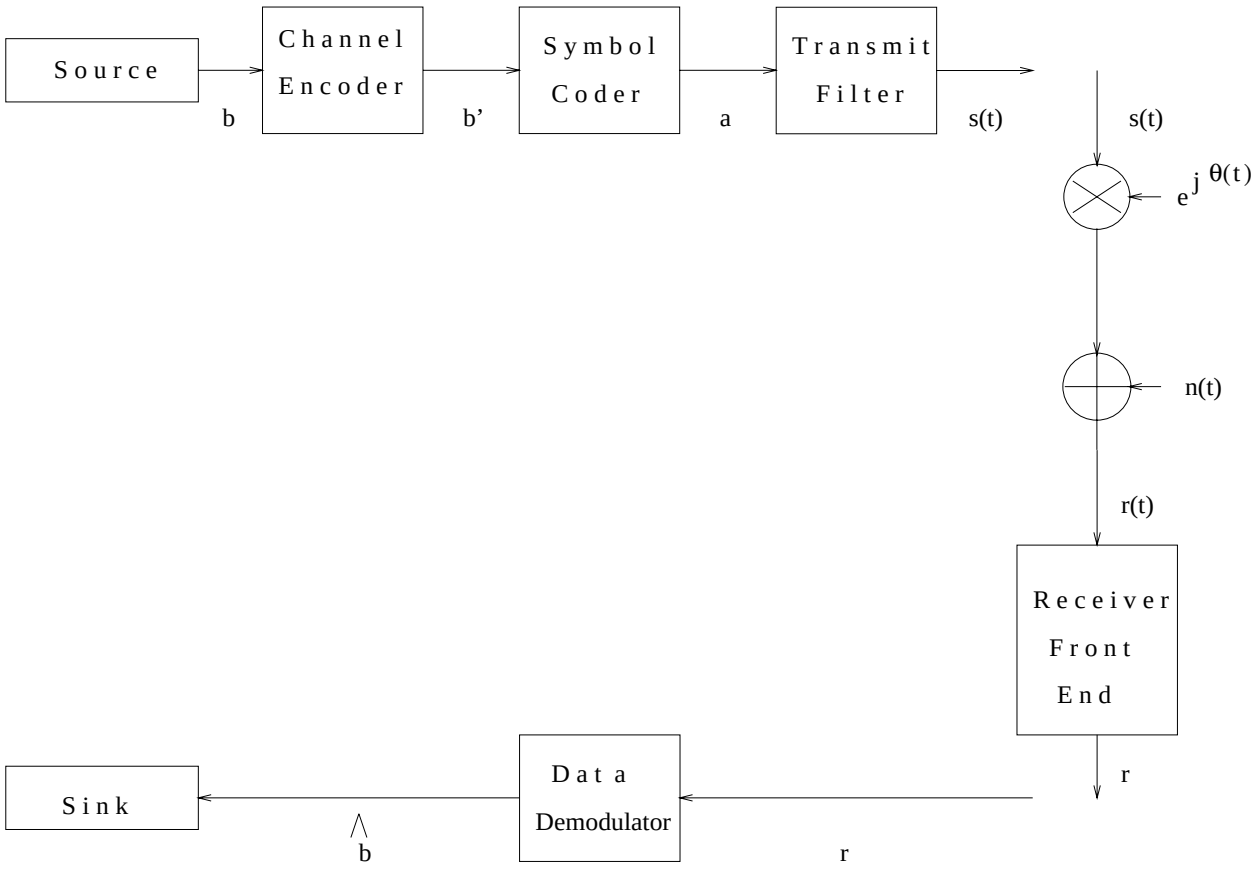

Figure 7.1: The communication system model.

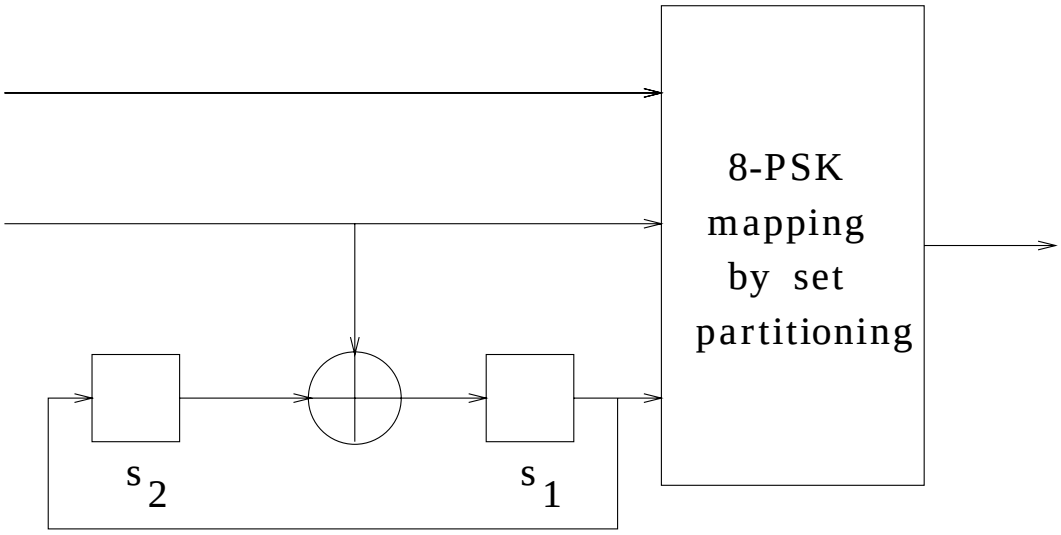

Figure 7.2: Schematic representing the TCM encoding.

that there is a one-to-one correspondence between the input binary sequence $\underline{b}$ and the output symbol sequence $\underline{a}$.

These two mappings represent a rate $\frac{n}{n+1}$ convolutional coder followed by a bit-to-symbol mapping called mapping-by-set-partitioning. This can be well explained by example, so we borrow an example from [71] to facilitate our explanation. Here, the two mappings are represented by Figure 7.2. From this figure, it is evident that there are $n = 2$ bits input at each time i, i.e., $\underline{b}_i = (b_{2i-1}, b_{2i})$. The value of ς_i is an index indicating the bits held in the two delay elements, e.g., $\varsigma_i = 2 \cdot s_2 + s_1$, $\varsigma_i \in \{0, 1, 2, 3\}$. Finally, the value a_i is an 8-PSK value, i.e., $a_i = \sqrt{E_s} e^{j\frac{2\pi}{M} l_i}$, $l_i \in \{1, 2, ..., 8\}$. This value is generated by a one-to-one mapping of the three binary values $(b_{2 \cdot i-1}, b_{2 \cdot i}, s_1)$ to an 8-PSK value.

The mapping of $\underline{b}$ to $\underline{a}$ can alternatively be described by a trellis. For example, the mapping described by Figure 7.2 can also be represented by the trellis of Figure 7.3. Here, each node of the trellis represents a possible state at time i, i.e., a possible ς_i value. Each branch, labeled with $n = 2$ bits, indicates the transition between states that results when the $n = 2$ bits are the input. Additionally, each branch is labeled with a symbol a_i. This indicates the symbol that is output given the starting state

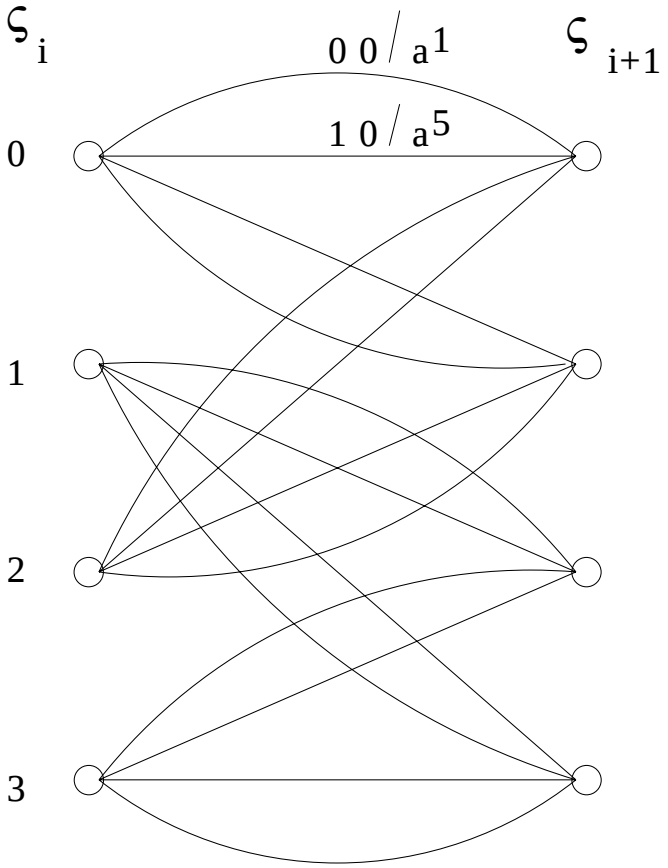

Figure 7.3: Trellis representing the TCM encoding.

and input bits.

Every channel encoder and symbol coder can be described by a finite state machine similar to the one of Figure 7.2, or, alternatively, by a trellis diagram similar to that of Figure 7.3.

We restrict the set of channel encoder/symbol coder mappings that we consider. We only consider mappings which are able to resolve phase ambiguities of $\frac{2\pi}{M}$. Some examples of these mappings, as well as general rules for creating them, are provided in [72]-[75].

Continuing with the description of the components in the communication system model, the *transmit filter* maps the symbol sequence $\underline{a}$ into the waveform $s(t) =$

$\sum_{i=1}^{L} a_i h_s(t - iT)e^{j\omega_c t}$. This waveform is ready for transmission over the channel.

The *channel* introduces two effects, an additive noise and a channel phase. The waveform output by the channel is described by

$$r(t) = s(t)e^{j\theta(t)} + \eta(t); \tag{7.3}$$

here, $\eta(t)$ represents an additive white Gaussian noise, and $\theta(t)$ represents the channel phase, assumed to be constant over $N > 2$ symbol intervals.

The *receiver front end* maps the continuous-time waveform $r(t)$ into a sequence of L symbols $\underline{r} = (r_1, ..., r_L)$, a sufficient statistic for detection. Here, each r_i is characterized by

$$r_i = a_i e^{j\theta_i} + \eta_i. \tag{7.4}$$

The η_i represent i.i.d. Gaussian random variables with variance $\frac{N_0}{2}$, and θ_i corresponds to an unknown phase value constant over N symbols, $N > 2$. This phase can be modeled statistically according to

$$p(\theta_i) = \begin{cases} \frac{1}{2\pi}, & \theta_i \in \Theta = [0, 2\pi) \\ 0, & \text{otherwise} \end{cases}; \tag{7.5}$$

also, whenever θ_i and θ_{i-1} are in the same block of N symbols

$$p(\theta_i|\theta_{i-1}) = \delta(\theta_i - \theta_{i-1}); \tag{7.6}$$

and, finally, whenever θ_i and θ_{i-1} are in different blocks of N symbols, they are assumed to be independent.

The *data demodulator* maps the received sequence $\underline{r}$ into $\hat{\underline{b}}$, an estimate of the binary sequence $\underline{b}$. In the case at hand, the channel decoder and symbol decoder are included in the data demodulator. This component can be designed by applying our proposed receiver structure.

7.2 Application of our Proposed Receiver Structure

In this section, we introduce a novel data demodulator for the coded PSK/unknown phase environment. We generate this receiver by applying our general receiver of Chapter 3 to the case at hand.

We present the data demodulator as generating the output $\underline{\hat{a}}$, an estimate of $\underline{a}$. The true demodulator output, $\underline{\hat{b}}$, is easily generated from $\underline{\hat{a}}$ by using the one-to-one mapping between these two sequences.

Our novel demodulator is presented in three parts. First, we introduce an underlying equation characterizing the operation of the data demodulator. Next, we present a corresponding data demodulator implementation. Finally, we create the discrete space $\tilde{\Theta}$ for use in the demodulator implementation.

7.2.1 The Data Demodulator's Underlying Equation

In this subsection, we present an underlying equation characterizing the data demodulator. We generate this equation by applying the general data detection equations provided in Chapter 3 to the case at hand.

In Chapter 3, we introduced three general data detection equations – (3.9), (3.20), and (3.21). The equation best suited to the communication environment at hand is equation (3.21). This states that data detection should be carried out according to: choose the sequence $\underline{a}$ that results from the joint maximization

$$\max_{\underline{\tilde{c}} \in \tilde{C}^L} \max_{\underline{S}} \sum_{i=1}^{L} \{ [\max_{a_i} \ln p(r_i | a_i, \underline{\tilde{c}}_i)|_{S_i}] + \ln P(\underline{\tilde{c}}_i | \underline{\tilde{c}}_{i-1}, ..., \underline{\tilde{c}}_{i-J}) \}. \tag{7.7}$$

We first rewrite this equation by introducing the terminology used in the case at hand. Here, the nuisance parameter vector $\underline{c}_i$ corresponds to the single phase value θ_i. Consequently, the vector $\underline{\tilde{c}}_i$, an approximation to $\underline{c}_i$, is replaced by the value $\tilde{\theta}_i$, an approximation to the phase θ_i. Additionally, the nuisance parameter space

C_0 corresponds to the phase space $\Theta = [0, 2\pi)$ in this application. It follows, then, that $\tilde{C}$, a discrete space approximation to C_0, is replaced by $\tilde{\Theta}$, a discrete space approximating Θ. Finally, the state S_i in (7.7) refers to a state summerizing the dependence of a_i on its past and future; in the case at hand, we have ς_i denoting the impact of a_i on its past, and ς_{i+1} denoting the dependence of a_i on its future; hence, in the case at hand, the S_i of (7.7) is replaced by the pair $(\varsigma_i, \varsigma_{i+1})$. Using these realizations in equation (7.7) leads to: choose the sequence $\underline{a}$ that results from the joint maximization

$$\max_{\underline{\tilde{\theta}} \in \tilde{\Theta}^L} \max_{\underline{\varsigma}} \sum_{i=1}^{L} \{[\max_{a_i} \ln p(r_i|a_i, \tilde{\theta}_i)|_{\varsigma_i, \varsigma_{i+1}}] + \ln P(\tilde{\theta}_i|\tilde{\theta}_{i-1}, ..., \tilde{\theta}_{i-J})\}. \tag{7.8}$$

We can further simplify this equation. This is achieved by applying the available phase statistics, namely equation (7.5) and (7.6), to equation (7.8). This leads to: choose the data sequence $\underline{a}$ that results from the joint maximization

$$\max_{\tilde{\theta}_1, \tilde{\theta}_{N+1}, ..., \tilde{\theta}_{\beta N+1}} \max_{\underline{\varsigma}} \{\sum_{i=1}^{N} [\max_{a_i} \ln p(r_i|a_i, \tilde{\theta}_1)|_{\varsigma_i, \varsigma_{i+1}}] + \sum_{i=N+1}^{2 \cdot N} [\max_{a_i} \ln p(r_i|a_i, \tilde{\theta}_{N+1})|_{\varsigma_i, \varsigma_{i+1}}] + ... + \sum_{i=\beta N+1}^{L} [\max_{a_i} \ln p(r_i|a_i, \tilde{\theta}_{\beta N+1})|_{\varsigma_i, \varsigma_{i+1}}]\} \tag{7.9}$$

where $\tilde{\theta}_i \in \tilde{\Theta} = \{\theta^1, ..., \theta^m\}$, and β is the integer closest to, but smaller than, $\frac{L}{N}$.

7.2.2 Implementation

In this subsection, we introduce an implementation for the data demodulator. This implementation corresponds to the parallel evaluation of equation (7.9).

The data detector implementation is shown in Figure 7.4. From this figure, it is apparent that this implementation demonstrates the same structure as the general receiver in Figure 3.1.

There are two main components in the implementation, the *universal set of demodulators* and the *CDU*. The universal set of demodulators carry out the inner maximization of equation (7.9), maximizing over the data symbols a_i; the CDU

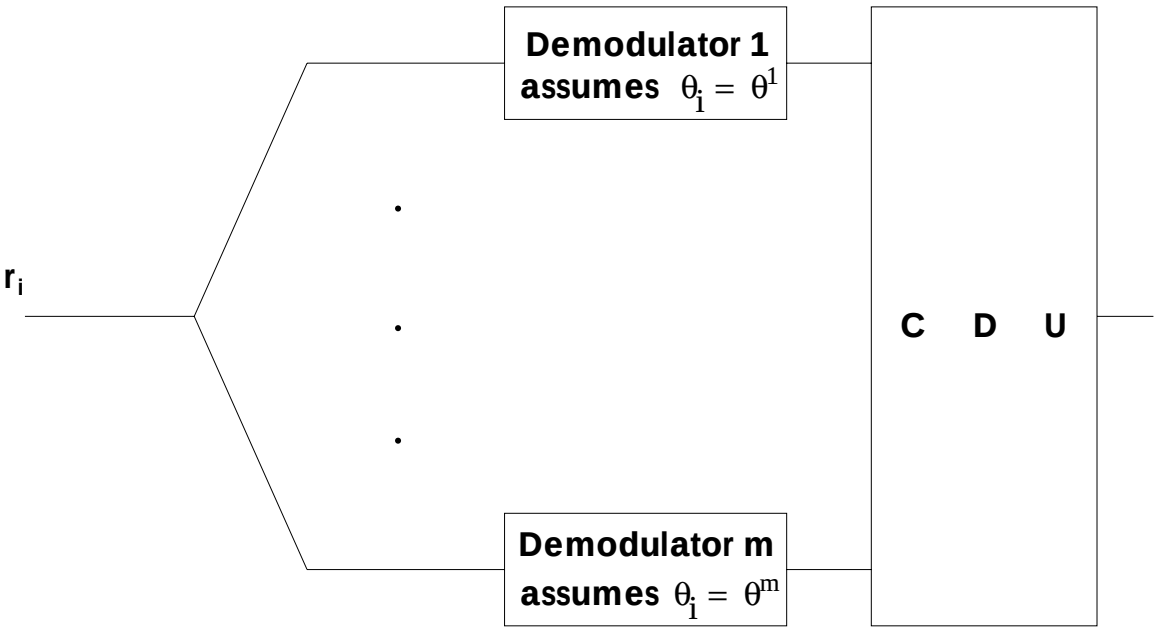

Figure 7.4: Data demodulator implementation.

carries out the two outer optimizations, optimizing over $\underline{\varsigma}$ and the discrete phases $\tilde{\theta}_1, \tilde{\theta}_N, ..., \tilde{\theta}_{\beta N+1}$.

The Universal Set of Demodulators

The universal set of demodulators can be described in detail as follows.

At each time i, all m demodulators receive the input sample r_i. The j^{th} demodulator assumes that the channel phase θ_i corresponds to θ^j, i.e., it assumes $\theta_i = \theta^j$, $\forall\ i$.

The j^{th} demodulator outputs several data symbols at each time i. Specifically, the j^{th} demodulator generates one output symbol for each possible ς_i (starting state) and ς_{i+1} (ending state). That is, considering the trellis (e.g., Figure 7.3), the j^{th} demodulator generates one data symbol output for each possible start node and end node. The data symbols generated at time i are created using the received r_i, the assumption $\theta_i = \theta^j$, and ς_i and ς_{i+1}. Specifically, the symbols correspond to

$$\hat{a}^j_{i,k} = \arg\max_{a_i} p(r_i|a_i, \theta^j)|_k,\ k = (1, 2, ..., V), \tag{7.10}$$

where k indicates that the k^{th} possible $(\varsigma_i, \varsigma_{i+1})$ pair is in effect, and V denotes the total number of $(\varsigma_i, \varsigma_{i+1})$ pairs.

In addition to the data symbol outputs, the j^{th} demodulator also generates the corresponding likelihood values

$$l_{i,k}^{j} = \max_{a_i} p(r_i|a_i, \theta^j)|_k = p(r_i|\hat{a}_{i,k}^{j}, \theta^j),\ k = (1, 2, ..., V). \tag{7.11}$$

The two sets of values, data symbols and corresponding likelihoods, are sent to the CDU at each time i.

The CDU

The CDU generates the output sequence $\underline{\hat{a}}$ by choosing from among the many demodulator decisions. It does this according to the outer maximizations of equation (7.9). The output generated by the CDU, based on the outer maximizations of (7.9), can be written in terms of demodulator outputs according to

$$\underline{\hat{a}} = (\hat{a}_{1,k_1^*}^{j_1^*}, \hat{a}_{2,k_2^*}^{j_1^*}, ..., \hat{a}_{N,k_N^*}^{j_1^*}, \hat{a}_{N+1,k_{N+1}^*}^{j_{N+1}^*}, \hat{a}_{N+2,k_{N+2}^*}^{j_{N+1}^*}, ..., \hat{a}_{2N,k_{2N}^*}^{j_{N+1}^*}, ..., \hat{a}_{\beta N+1,k_{\beta N+1}^*}^{j_{\beta N+1}^*}, \hat{a}_{\beta N+2,k_{\beta N+2}^*}^{j_{\beta N+1}^*}, ...\hat{a}_{L,k_L^*}^{j_{\beta N+1}^*}) \tag{7.12}$$

here, the $\underline{k}^*$ and the $j_{i\cdot N+1}^*$ values are selected according to

$$\underline{k}^*, j_1^*, j_{N+1}^*, ..., j_{\beta N+1}^* = \arg\max_{\underline{k} \in K_P, j_1, j_{N+1}, ..., j_{\beta N+1}} \sum_{i=1}^{N} \ln l_{k_i}^{j_1} + \sum_{i=N+1}^{2\cdot N} \ln l_{k_i}^{j_{N+1}} + ... + \sum_{i=\beta N+1}^{L} \ln l_{k_i}^{j_{\beta N+1}} \tag{7.13}$$

We now explain how to implement a CDU to carry out these two equations.

The CDU can be viewed as implementing m Viterbi Algorithms (VA's) in parallel, with a brief interaction between the VA's after each block of N symbols. In what follows, we describe this implementation in more detail using the illustrative trellis diagram (e.g., Figure 7.3).

The j^{th} VA finds the best data sequence through the trellis assuming that the phase $\theta_i = \theta^j$. This VA is aided by the j^{th} demodulator, which provides it with branch metrics, namely the best branch metric between each two nodes.

The m parallel VA's pause after running through a block of N symbols. They pause to allow the CDU to compare their end node metrics. First, the end node metric associated with state 1 is considered. Here, m metrics are available, one from each VA. The CDU finds the largest end node metric, and replaces all the VA's current end node metrics with this largest value. The CDU makes a point of remembering from which VA they all got their end node metric. This is repeated for each possible end node. When this process is complete, each VA sits with the same set of end node metrics, each having the largest possible end node metrics.

Hence, we have m parallel VA's, the j^{th} assuming $\theta_i = \theta^j$. After running through each block of N symbols, the VA's pause, and the CDU replaces each end node metric with the largest available end node metric. They then carry on.

When the VA's reach the end of the data sequence, they stop. At this point, the CDU backtracks through the trellis, outputting the optimal sequence. This backtracking proceeds as a normal Viterbi backtracking through a trellis, with one difference: after each N symbols, the backtracking pauses; the VA, whose output is being selected, changes; it changes to the VA that generated the current end node metric.

The backtracking through the trellis may be followed by a differential decoding. Specifically, a differential decoding is included if differential encoding is used to generate the a_i's. The differential decoding, which may be in the CDU, was not included in equations (7.12) and (7.13).

7.2.3 The Discrete Phase Space $\tilde{\Theta}$

In this subsection, we generate the discrete phase space $\tilde{\Theta} = \{\theta^1, ..., \theta^m\}$. This $\tilde{\Theta}$ is used in the data detector's underlying equation and corresponding implementation.

Redefining the Continuous Phase Space Θ

We begin the evaluation of $\tilde{\Theta}$ by first redefining Θ, the continuous phase space which $\tilde{\Theta}$ approximates. In previous sections, we identified Θ as $[0, 2\pi)$. However, in Section 7.1, we explained that the channel encoder mapping is carefully chosen to insure invariance

to $\frac{2\pi}{M}$ phase ambiguities. Consequently, at the data demodulator, it suffices to assume that the continuous channel phase space is $\Theta = [0, \frac{2\pi}{M})$, rather than $\Theta = [0, 2\pi)$.

The Space $\tilde{\Theta}$

We now evaluate the discrete space $\tilde{\Theta} = \{\theta^1, ..., \theta^m\}$.

Overview Previously, in Chapters 5 and 6, we generated $\tilde{\Theta}$ in two parts. First, we applied the algorithm of Section 4.3 to get a good starting value for m, the size of $\tilde{\Theta}$. Then we applied the algorithm of Section 4.4 to generate an exact value for m along with the θ^j elements. However, in the case at hand, we can not establish $\tilde{\Theta}$ in this manner.

First, consider the algorithm of Section 4.3. This algorithm is presented for cases of independent noise samples and independent data symbols, and, in the case at hand, the data symbols are *not* independent. Hence, in its current form, the algorithm is not applicable. An analogous algorithm can be established for cases where the data symbols demonstrate state dependence, but this algorithm requires the analytical evaluation of $P(\epsilon|\theta_i, \theta^j)$ (the probability of a single symbol error) which is not attainable.

Second, consider the algorithm of Section 4.4. This can not be carried out in the case at hand, again because of the need to evaluate $P(\epsilon|\theta_i, \theta^j)$.

Hence, we proceed as follows. First, we generate the elements $\{\theta^1, ..., \theta^m\}$ (for any m value) by using an approximation to the algorithm of Section 4.4. Next, we generate m by heuristic means.

The Elements $\{\theta^1, ..., \theta^m\}$ We now detail how we achieve $\{\theta^1, ..., \theta^m\}$ by an approximation to the algorithm of Section 4.4.

At the heart of the algorithm of Section 4.4 lies an iterative two-step process, namely steps **B2** and **C2**, with stopping criteria **D2**. These steps establish the set $\tilde{\Theta}$ at a given m value.

Step **B2** computes nearest neighbour cells, R_j. In the case at hand, this can be carried out according to equation (4.36).

Step **C2** computes values for θ^j using a centroid computation and R_j. Unfortunately, the centroid computation uses $P(\epsilon|\theta_i, \theta^j)$, a term which is not attainable. Consequently, we use an approximate centroid condition. The probability $P(\epsilon|\theta_i, \theta^j)$ is proportional to $|\theta_i - \theta^j|$; hence, for the sake of facilitating a centroid computation, we replace $P(\epsilon|\theta_i, \theta^j)$ by another function, also proportional to $|\theta_i - \theta^j|$, which allows for an easy centroid computation. Specifically, we replace $P(\epsilon|\theta_i, \theta^j)$ by either $|\theta_i - \theta^j|$ or $|\theta_i - \theta^j|^2$ — in the case at hand, these both lead to the same centroid result.

The result of a two-step iterative process using the above **B2** and **C2**, with stopping criteria **D2**, and with a $p(\theta_i)$ described by (7.5), can be evaluated analytically [66, pg 183]. Specifically, this iterative process leads to $\tilde{\Theta} = \{\frac{2\pi}{M} \cdot \frac{1}{2 \cdot m}, \frac{2\pi}{M} \cdot \frac{3}{2 \cdot m}, ..., \frac{2\pi}{M} \cdot \frac{2 \cdot m - 1}{2 \cdot m}\}$, i.e., $\tilde{\Theta}$ corresponds to $\{\theta^j = \frac{2\pi}{M} \cdot \frac{2j-1}{2 \cdot m}, j = 1, 2, ..., m\}$.

Establishing m We generate m by heuristic means as follows. First, we provide some $P(\epsilon)$ results for our data detector at different m values. We do this using a computer simulation that assumes the following conditions: SNR corresponds to the value that yields a coherent $P(\epsilon)$ of 10^{-4}; phase is constant over 10 symbol intervals; and the coded MPSK scheme corresponds to a typical rate $\frac{2}{3}$ convolutional coder with 4 states, followed by a careful mapping to 8-PSK. Next, using the simulation results, we determine the value of m, where, increasing m beyond this point leads to negligible performance gains. This value becomes our m value. Applying this heuristic method leads to $m = 4$. Hence, the discrete set of phases corresponds to $\tilde{\Theta} = \{\frac{2\pi}{M} \cdot \frac{1}{8}, \frac{2\pi}{M} \cdot \frac{3}{8}, \frac{2\pi}{M} \cdot \frac{5}{8}, \frac{2\pi}{M} \cdot \frac{7}{8}\}$.

We briefly explain why the m value is so small, half of the m value in Chapters 5 and 6. The explanation is based on the value of d_{free}, a value in trellis coded modulation that is key in determining performance; d_{free} refers to the shortest distance between two paths (in the trellis) with the same start and end node.

Our data detector corresponds to m VA's running over each block of N symbols; each VA assumes a different discrete phase value in $\tilde{\Theta}$. We choose from among the m

VA's decision sequences after each block of N symbols. Hence, in our data detector, d_{free} corresponds to: the smaller of d_1 and d_2; here, d_1 is the shortest distance between pathes with the same start and end node in a single VA; d_2 is the shortest distance between a path of N symbols in a VA using one discrete phase, and a different N-symbol-long path (with the same start and end node) in a VA assuming a second discrete phase.

When m is kept small, d_2 is large, and d_1 is always smaller than d_2; hence, the data detector maintains the traditional value of $d_{free} = d_1$. As m increases, d_2 gets smaller, and it becomes possible that d_{free} becomes d_2, a value that diminishes as m increases. For this reason, a small m value achieves a quality performance: an increased m, offering a performance gain due to smaller phase error, has this performance gain offset by a reduced d_{free}.

7.3 Complexity and Performance

In this subsection, we present the complexity and performance of the data demodulator introduced in Section 7.2, an application of our general receiver to the case at hand.

7.3.1 Complexity

We begin by introducing the complexity of our data demodulator.

The data demodulator, shown in Figure 7.4, consists of two main computational components: the set of $m = 4$ demodulators, and the CDU.

The j^{th} demodulator in the set of $m = 4$ demodulators generates, for each received sample, several data symbols and corresponding likelihood values, creating one pair for each possible start and end state. In terms of a Viterbi Algorithm (VA), the j^{th} demodulator computes the branch metrics for a VA, and it decides between parallel transitions in the VA.

The CDU computation can be separated into two parts. First, the CDU implements $m = 4$ Viterbi Algorithms, using the branch metrics and parallel transition decisions already available from the $m = 4$ demodulators. Second, it carries out additional computations at the end of each block of N symbols; here, comparisons of $m = 4$ values are performed for each state.

Putting it all together, the computations of the data demodulator are: 4 traditional Viterbi Algorithms, each assuming a different channel phase θ^j; and, once every N symbols, a comparison of 4 values for each state. That is, the overall computation is in the order of 4 traditional Viterbi Algorithms.

7.3.2 Performance

In this subsection, we introduce the performance of our data detector. This performance is generated, by computer simulation, using a rate $\frac{2}{3}$ 4-state 8-PSK channel encoder.

Figure 7.5 shows the probability of event error plotted against the signal to noise ratio $\frac{E_s}{N_o}$. The points marked 'x' provide the results at $N = 500$; these results match coherent, and hence serve as a benchmark for comparison.

The points marked 'o' present the performance of our data demodulator when $N = 50$. These points indicate a performance that is very close to coherent (less than 0.1 dB degradation).

The points marked '*', representing the performance of our data demodulator with $N = 20$, again indicate a performance very close to coherent. Only about 0.1 dB separates the performance of our demodulator ($N = 20$) from coherent.

Finally, the '+' set of points, generated using $N = 10$, indicate that our data detector's performance remains close to coherent even at this small N value. About $\frac{1}{2}$ of a dB separates the performance of our data detector from coherent.

Figure 7.6 presents another performance curve, one of greater practical interest. This curve introduces the probability of bit error, denoted $P(\text{bit error})$, as a function of $\frac{E_s}{N_o}$. Here, the 'x' points once again serve as our benchmark, as they correspond to

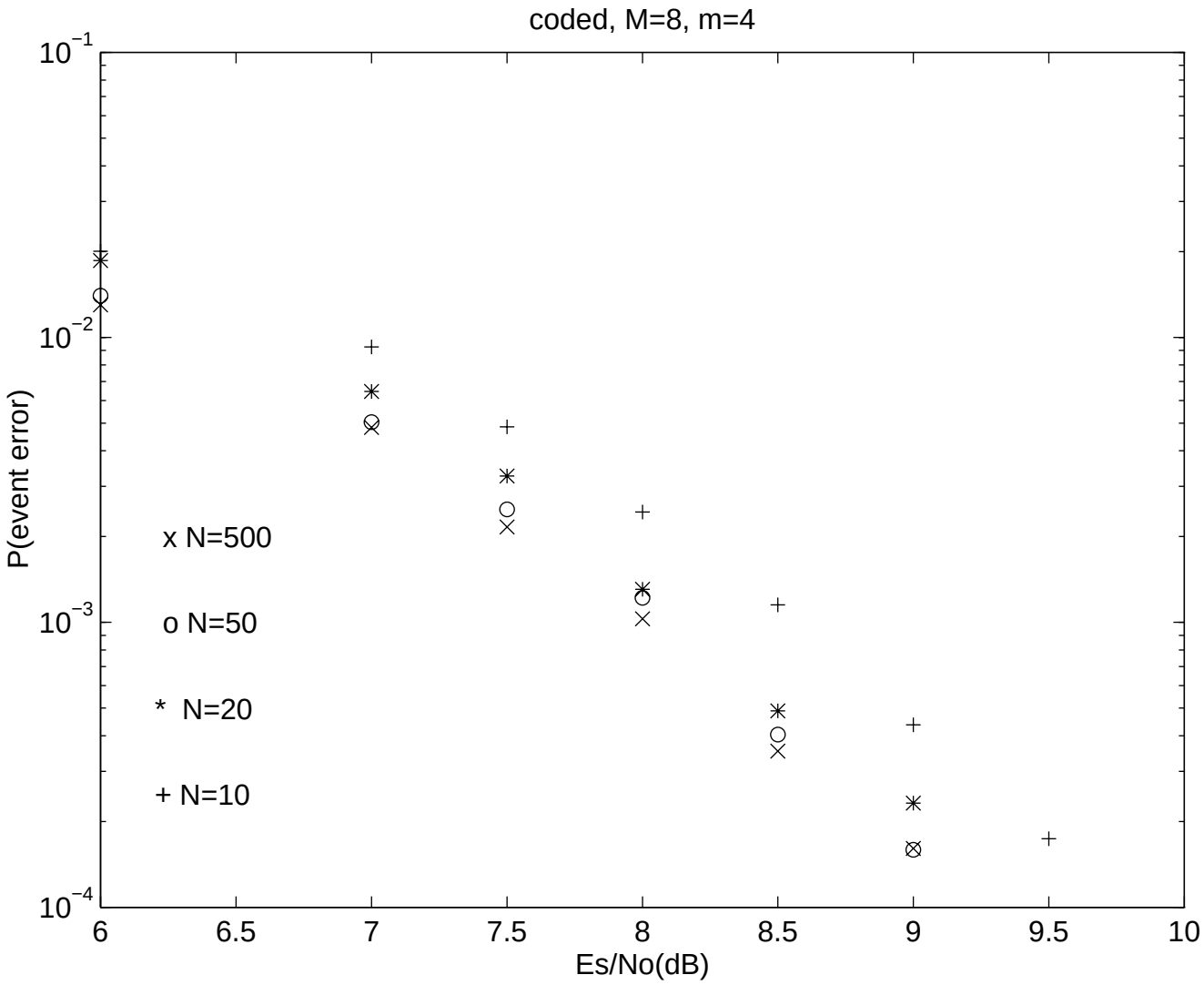

Figure 7.5: $P(event\ error) - \frac{E_s}{N_o}$ curve.

a performance matching coherent.

The 'o' points, showing the performance of our data detector with $N = 50$, indicate a performance very close to coherent (approximately a 0.1 dB degradation). The '*' points present our detector's performance with $N = 20$; these points indicate a performance about $\frac{1}{4}$ of a dB away from coherent. Finally, the '+' points show the performance when $N = 10$; this indicates that, at this low N value, our data demodulator loses about $\frac{3}{4}$ of a dB when compared to coherent.

7.3.3 Comparison to Other Receivers

We now compare our data demodulator to other receivers available to date.

Our data demodulator achieves performances close to coherent with N as low as 10. Furthermore, its performance essentially matches coherent with N as small as 20. To date, no other receiver (e.g., [13][30]-[37]) has demonstrated a comparable performance with N values of 10 or 20 at a realizable complexity. Hence, our detector offers substantial performance gains when compared to any receiver available to date, when phase is constant over only a handful of symbols.

Furthermore, the implementation of our data detector is very close to the implementation of 4 parallel Viterbi Algorithms. The availability of (and ease of implementing) Viterbi Algorithms makes this an easily realizable receiver.

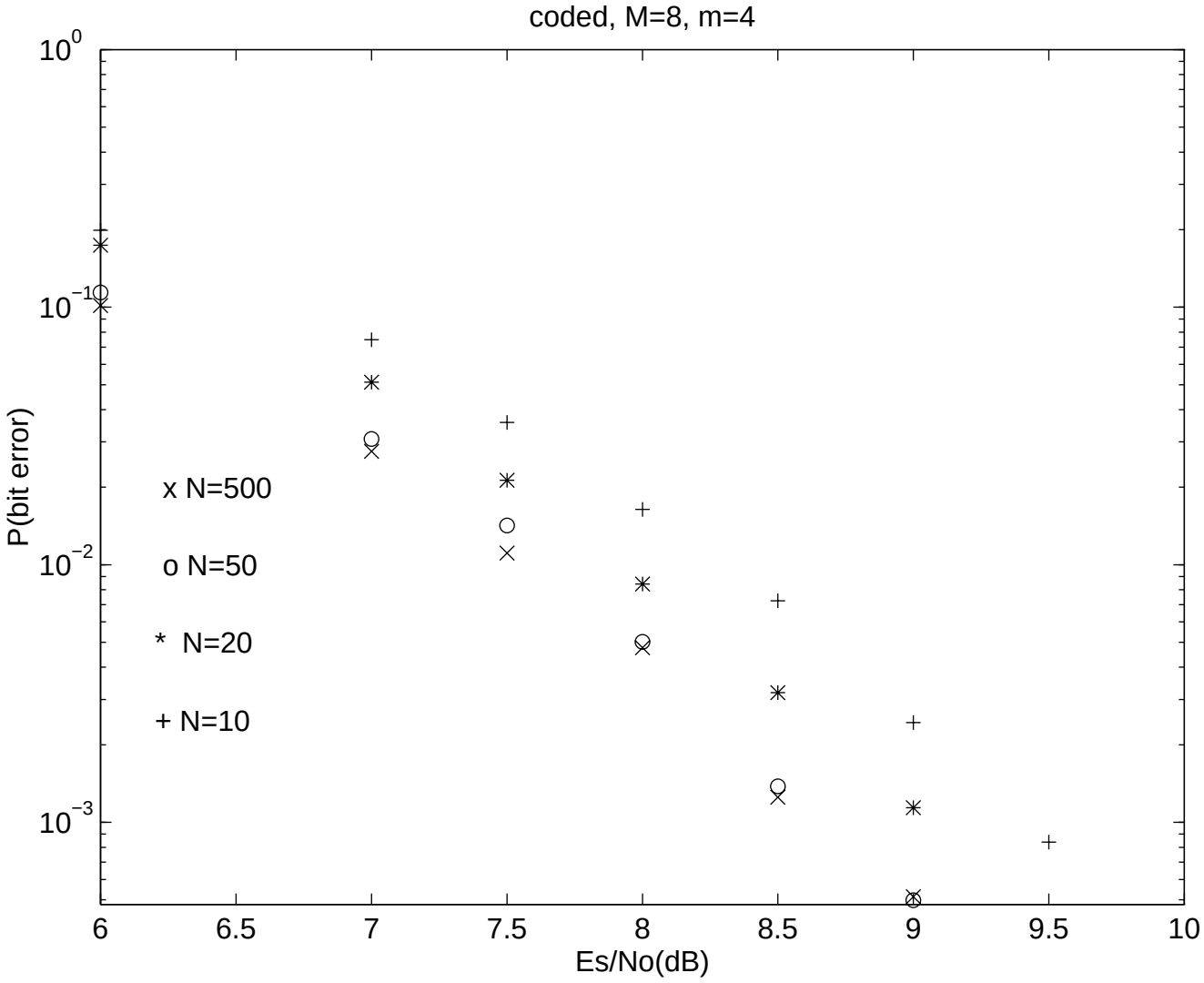

Figure 7.6: $P(bit\ error) - \frac{E_s}{N_o}$ curve.

Chapter 8

Data Detection in a Changing Timing Offset Environment

This chapter presents another application of our proposed receiver structure. Here, we apply our receiver to the practical communication environment described briefly as follows. A transmitter outputs independent data symbols, and these symbols are sent over a channel introducing two effects. The channel adds a noise and introduces a timing offset. The timing offset is constant over a burst of N symbols (e.g., 100 symbols); this models the timing offset of channels in burst mode communications. For simplicity in presentation, we will henceforth use the term *timing offset* to refer specifically to a timing offset constant over N symbols.

This chapter proceeds as follows. First, we provide a detailed description of the timing offset communication environment. This is followed by the application of our proposed receiver structure; we show that the receiver which results demonstrates many benefits when compared to the receivers in current literature ([38]-[40]), receivers summerized in Chapter 1.

8.1 The Communication Environment Model

In this section, we introduce a detailed model for the timing-offset communication environment. This model is shown in Figure 8.1; in what follows, we describe each component in this model.

First, the *source* creates a sequence of binary digits. This sequence is labeled $\underline{b} = (b_1, b_2, ..., b_X)$.

The *symbol coder* then maps these bits into the symbol sequence $\underline{a} = (a_1, a_2, ..., a_L)$. Here, each element a_i corresponds to a value in $A = \{a^1, ..., a^M\}$; an example of an $a^k \in A$ is $a^k = e^{j\frac{2\pi}{M}k}$.

The *transmit filter* maps the symbol sequence $\underline{a}$ into the waveform $s(t)$, a waveform ready for transmission over the channel. This mapping is described as follows. First, each data symbol a_i is mapped to the continuous time waveform $a_i h_s(t - iT)e^{j\omega_c t}$, where $h_s(t)$ corresponds to a *square-root raised-cosine waveform*, i.e.,

$$h_s(t) * h_s^*(-t) = p_s(t) = \frac{\sin(\pi t/T)}{\pi t/T} \cdot \frac{cos(\alpha\pi t/T)}{1 - (2\alpha t/T)^2}; \tag{8.1}$$

here, '$*$' denotes convolution, and α is a value in $[0, 1]$ called the *roll-off factor*. The waveform $s(t)$ is then created by adding together all the $a_i h_s(t - iT)e^{j\omega_c t}$ terms, i.e.,

$$s(t) = \sum_{i=1}^{L} a_i h_s(t - iT)e^{j\omega_c t}. \tag{8.2}$$

(In practice, $s(t)$ is simply the real component of equation (8.2). However, we use complex notation here because it simplifies the presentation without impacting the receiver design.)

The *channel* introduces two effects: it delays the signal, and it adds a noise. The output of the channel is described by

$$r(t) = s(t - \tau(t)) + \eta(t). \tag{8.3}$$

Here, $\eta(t)$ represents the additive noise, modeled as AWGN. Additionally, $\tau(t)$ represents the channel delay. This delay represents a portion of the symbol duration, i.e., $0 \leq \tau(t) < T$; furthermore, this delay is constant over N symbol intervals.

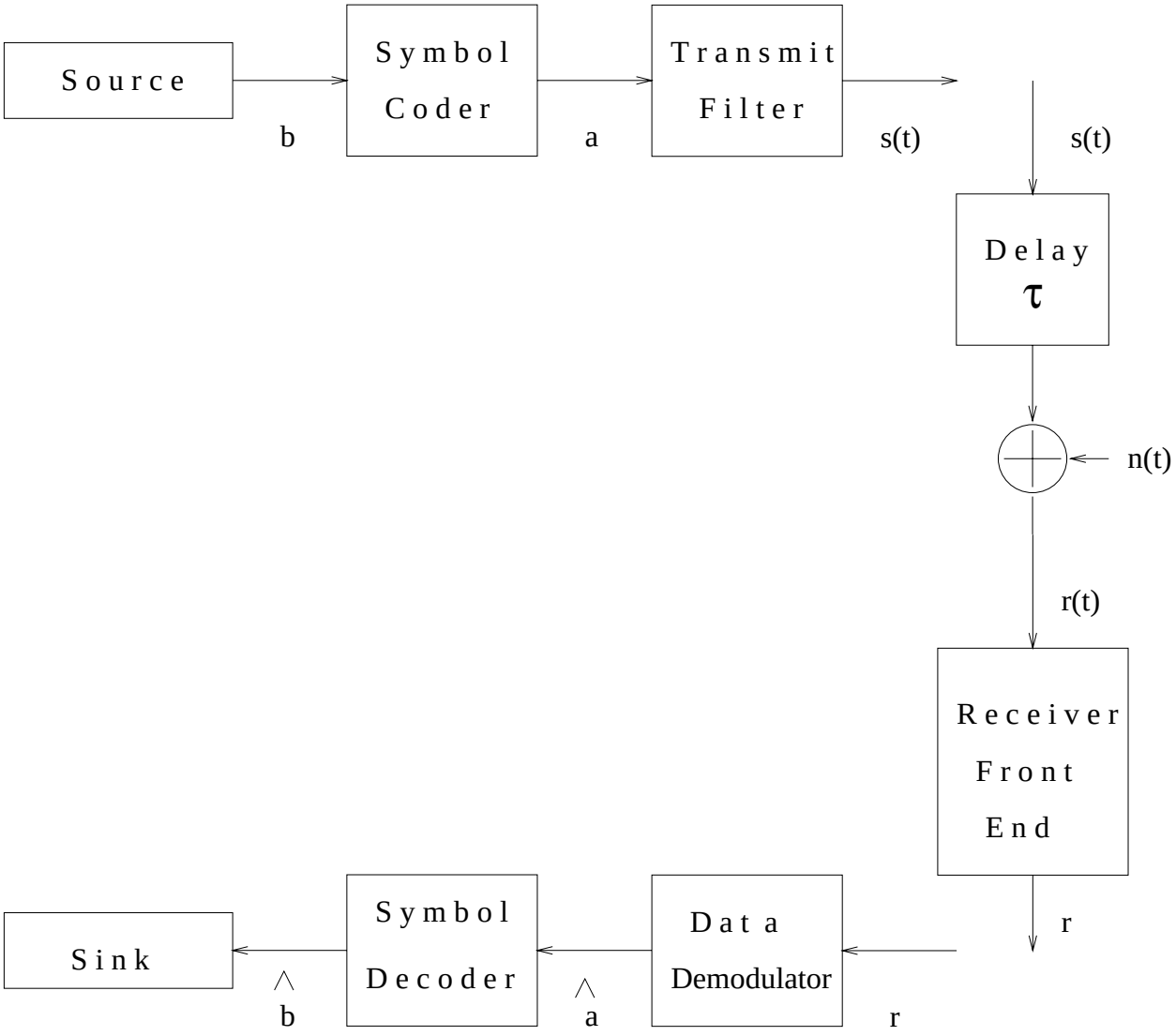

Figure 8.1: The communication system model.

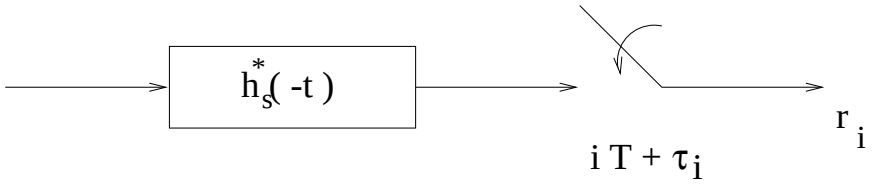

Figure 8.2: A typical receiver front end.

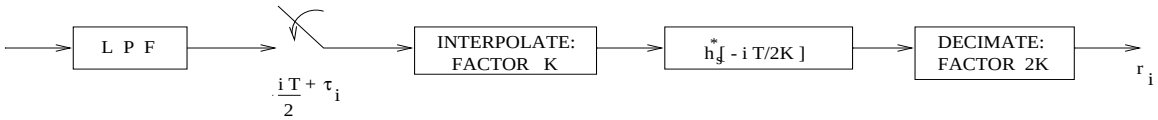

Figure 8.3: A digital receiver front end.

The *receiver front end* maps the waveform $r(t)$ into the sequence $\underline{r}$. It does this using three components: a mixer, a matched filter and a sampler, usually in that order. The mixer translates the symbol to baseband; we assume that the mixer has perfect knowledge of frequency and phase. The output of this mixer is

$$r'(t) = \sum_{i=1}^{L} a_i h_s(t - \tau_i - iT) + \eta'(t), \tag{8.4}$$

where $\eta'(t)$ represents the noise after the mixer operation, and $\tau_i = \tau(iT)$. The remainder of the receiver front end is shown in Figure 8.2. Here, the matched filter demonstrates an impulse response of $h_s^*(-t)$, and the sampler generates one sample every T second interval by sampling at time instants $iT + \tau_i$.

In the case at hand, we employ a digital version of this receiver front end. Here, dropping the mixer for simplicity in presentation, the receiver front end is shown in Figure 8.3. A low pass filter removes the noise components outside of the information-bearing signal's bandwidth. This filtered signal is sampled at time instants $\frac{iT}{2} + \tau_i$, a rate greater than (or equal to) the Nyquist rate. The sampled signal is then interpolated to achieve $2 \cdot K$ samples per symbol interval T, where K is typically 2 or 3. This upsampled signal is filtered by a digital implementation of the matched filter. Lastly, the filtered signal is downsampled by a factor of $2K$, leaving only the samples at time instants $iT + \tau_i$.

The receiver front end we employ corresponds to an alternative (equivalent) imple-

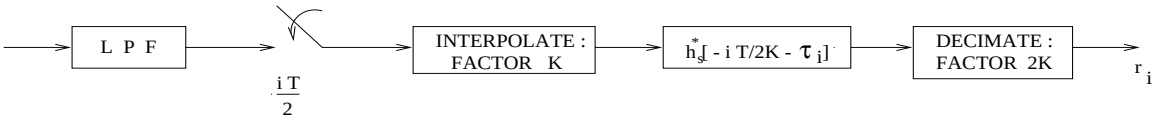

Figure 8.4: A final receiver front end

mentation of the digital receiver front end described by Figure 8.3. This alternative implementation is shown in Figure 8.4 (without the mixer). Here, the delay of τ_i has moved from the sampling instant to the digital matched filter.

We present the receiver front end of interest in the case at hand by making a final realization. The value of τ_i is not available for the matched filter of Figure 8.4. Consequently, this matched filter is not easily realized. We will include the design of the matched filter (along with the interpolator and decimator) in the demodulator. Hence, the receiver front end we consider consists simply of a mixer, a low pass filter, and a sampler generating samples at time instants $\frac{iT}{2}$.

Returning to describe the components in Figure 8.1, the *demodulator* implements the discrete time matched filter (along with the interpolator and decimator), outputting the sequence $\underline{r}$; it then maps $\underline{r}$ into $\underline{\hat{a}}$, an estimate of the transmitted symbol sequence. A simple *symbol decoder* follows, mapping the symbols $\underline{\hat{a}}$ into bits $\underline{\hat{b}}$, an estimate of the source's binary sequence.

8.2 Demodulator Based on Our Proposed Receiver

In this section, we present the design of the demodulator based on the general receiver structure presented in Chapter 3.

8.2.1 Underlying Equation

We begin by presenting an underlying equation characterizing the demodulator operation. This equation is generated by applying the general receiver equations of Chapter 3 to the case at hand.

In Chapter 3, we introduced three equations for data detection, namely equations (3.9), (3.20), and (3.21). The equation best suited to the case at hand is equation (3.20). According to this equation, data detection should be carried out as follows: select the data sequence $\underline{\hat{a}}$ from the joint maximization

$$\max_{\underline{\tilde{c}} \in \tilde{C}^L} \sum_{i=1}^{L} \{ [\max_{a_i} \ln p(r_i | a_i, \underline{\tilde{c}}_i)] + \ln P(\underline{\tilde{c}}_i | \underline{\tilde{c}}_{i-1}, ..., \underline{\tilde{c}}_{i-J}) \}. \tag{8.5}$$

We now rewrite this equation using the terminology appropriate to the communication environment at hand. First, the nuisance parameter vector $\underline{c}_i$ corresponds to the single timing offset value τ_i. This implies that $\underline{\tilde{c}}_i$ corresponds to $\tilde{\tau}_i$. Additionally, the nuisance parameter space C_0 is replaced by the timing offset space $\Upsilon = [0, T]$. This suggests that $\tilde{C}$ is replaced by $\tilde{\Upsilon} = \{\tau^1, ..., \tau^m\}$, a discrete space approximation to Υ. Finally, in the case at hand, the sufficient statistic for detection, r_i, is replaced by the set of samples (samples at time instants $\frac{iT}{2}$) required, at the demodulator, to generate r_i using interpolation, the digital matched filter, and decimation; we will label this set $\underline{r}(i)$. Using these realizations, equation (8.5) can be rewritten according to

$$\max_{\underline{\tilde{\tau}} \in \tilde{\Upsilon}^L} \sum_{i=1}^{L} \{ [\max_{a_i} \ln p(\underline{r}(i) | a_i, \tilde{\tau}_i)] + \ln P(\tilde{\tau}_i | \tilde{\tau}_{i-1}, ..., \tilde{\tau}_{i-J}) \}; \tag{8.6}$$

here, $p(\underline{r}(i) | a_i, \tilde{\tau}_i)$ refers to $p(r_i^{\tilde{\tau}_i} | a_i, \tilde{\tau}_i)$, where $r_i^{\tilde{\tau}_i}$ is the sufficient statistic r_i generated by a digital matched filter that is given $\underline{r}(i)$ and $\tilde{\tau}_i$.

We can further simplify this equation by using the available information regarding τ_i. This information is the following. The value τ_i corresponds to $\tau(iT)$, and it is known that $\tau(t)$ is constant over a block of N symbol intervals. This implies that τ_i is a constant, unknown value over each N symbols. This τ_i can be characterized statistically according to:

$$p(\tau_i) = \begin{cases} \frac{1}{T}, & \tau_i \in \Upsilon = [0, T) \\ 0, & \text{else} \end{cases} ; \tag{8.7}$$

additionally, if τ_i and τ_{i-1} are in the same block of N symbols

$$p(\tau_i | \tau_{i-1}) = \delta(\tau_i - \tau_{i-1}); \tag{8.8}$$

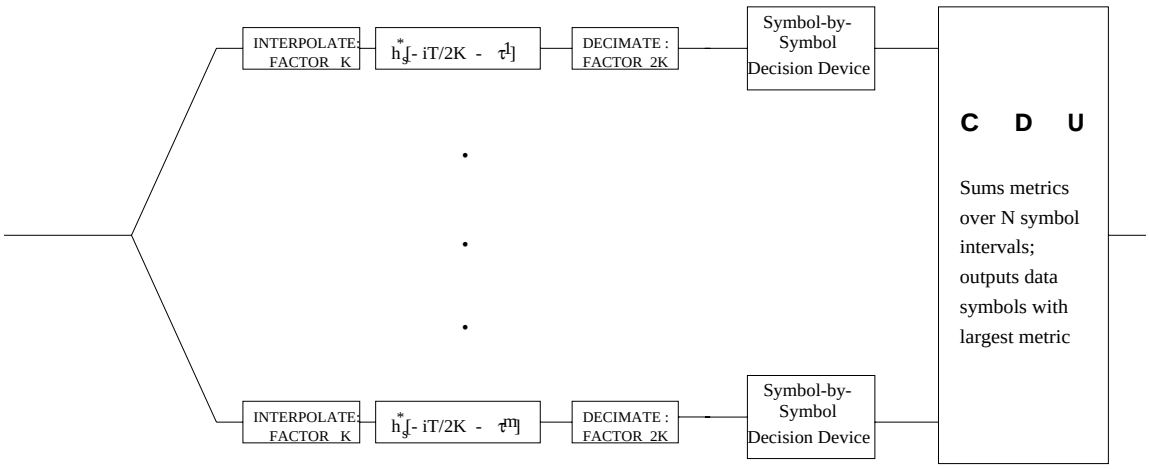

Figure 8.5: The demodulator implementation.

and, finally, if τ_i and τ_{i-1} are in different blocks of N symbols, they are assumed to be independent. Applying this τ_i information to equation (8.6) leads to: choose the $\underline{\hat{a}}$ from the maximizations

$$\{\max_{\tilde{\tau}_1 \in \tilde{\Upsilon}} \sum_{i=1}^{N} [\max_{a_i} \ln p(\underline{r}(i)|a_i, \tilde{\tau}_1)]\} + \{\max_{\tilde{\tau}_{N+1} \in \tilde{\Upsilon}} \sum_{i=N+1}^{2N} [\max_{a_i} \ln p(\underline{r}(i)|a_i, \tilde{\tau}_{N+1})]\} + \ldots \quad (8.9)$$

The joint maximizations are independent of one another, and hence we can rewrite this data detection equation according to: over each block of N symbols, choose the N symbol long sequence $\underline{\hat{a}}$ that results from the joint maximization

$$\max_{\tilde{\tau} \in \tilde{\Upsilon}} \sum_{i=1}^{N} [\max_{a_i} \ln p(\underline{r}(i)|a_i, \tilde{\tau})]. \quad (8.10)$$

8.2.2 Implementation

In this subsection, we provide an implementation for the demodulator. This implementation corresponds to a parallel evaluation of equation (8.10).

The implementation is presented in Figure 8.5. This figure shows that the receiver implementation at hand matches the general receiver structure of Figure 3.1.

The Universal Set of Demodulators

There are two main parts to the implementation. The first of the two is the *universal set of demodulators*, consisting of m parallel branches. These implement the inner maximization of equation (8.10), as described in the following.

In the i^{th} symbol interval $[iT, (i+1)T)$, the j^{th} branch receives $\underline{r}(i)$, the samples required to generate the matched filter output r_i. This j^{th} branch assumes that $\tau_i = \tau^j$, regardless of the time index i.

The j^{th} branch's processing of $\underline{r}(i)$ is done in two parts: first, a *matched filter* (preceded by an interpolator and followed by a decimator) is applied to generate r_i, and then a *decision device* is used to create $\hat{a}_i$. The *matched filter* demonstrates an impulse response of $h_s^*[-\frac{iT}{2K} - \tau_i]$ (see Figure 8.4). However, at the j^{th} branch, it is assumed that $\tau_i = \tau^j$. Hence, the matched filter at the j^{th} branch demonstrates an impulse response of $h_s^*[-\frac{iT}{2K} - \tau^j]$. The output of this matched filter (after decimation) is labeled r_i^j. This output is described by

$$r_i^j \simeq a_i p_s(\tau^j - \tau_i) + \sum_{k \neq i} a_k p_s((i-k)T + (\tau^j - \tau_i)) + \eta_i, \tag{8.11}$$

where η_i corresponds to an i.i.d. Gaussian random variable with variance $\frac{N_o}{2}$, and $p_s(\cdot)$ is defined in equation (8.1). In cases where τ_i does in fact truly correspond to τ^j, this output simplifies to

$$r_i^j \simeq a_i + \eta_i. \tag{8.12}$$

The *decision device* in the j^{th} branch receives, at each time i, the symbol r_i^j. As it also assumes that $\tau_i = \tau^j$, it believes that the received sample r_i^j corresponds to the value in equation (8.12). Consequently, it generates the simple ML decision

$$\hat{a}_i^j = \arg\max_{a_i} p(r_i^j | a_i) = \arg\max_{a_i} p_{\eta_i}(r_i^j - a_i). \tag{8.13}$$

It also generates the corresponding likelihood value

$$l_i^j = \max_{a_i} p(r_i^j | a_i) = \max_{a_i} p_{\eta_i}(r_i^j - a_i) = p_{\eta_i}(r_i^j - \hat{a}_i^j). \tag{8.14}$$

The two processings at the j^{th} branch correspond to an implementation of the inner maximization of equation (8.10), namely $max_{a_i} \ln p(\underline{r}(i) | a_i, \tilde{\tau})$, when evaluated at $\tilde{\tau} = \tau^j$; the logarithm in the maximization has not been included for convenience.

The CDU

The CDU generates the output $\underline{\hat{a}}$ by selecting from among the many decisions of the universal set of demodulators. Specifically, it generates its decision by implementing the outer maximization of equation (8.10). This outer maximization, which the CDU implements, can be expressed in terms of the universal set of demodulators' decisions according to

$$\underline{\hat{a}} = (\hat{a}_1^{j^*}, \hat{a}_2^{j^*}, ..., \hat{a}_N^{j^*}),\ j^* \in \{1, 2, ..., m\}; \tag{8.15}$$

here, j^* corresponds to the index of the branch (in the universal set of demodulators) whose decisions are selected by the CDU; this value is generated according to

$$j^* = \arg\max_j \sum_{i=1}^{N} \ln l_i^j. \tag{8.16}$$

The CDU implementing the above two equations is described as follows: the CDU sums the N likelihood values it receives from each of the m demodulators; it then determines which sum is the largest; finally, it outputs all the decisions of the demodulator with the largest sum.

8.2.3 The Discrete Space $\tilde{\Upsilon}$

In this subsection, we generate the discrete space $\tilde{\Upsilon} = \{\tau^1, ..., \tau^m\}$ for use in the underlying equation and corresponding implementation. This is carried out in two parts. First, we generate a starting value for m, the size of $\tilde{\Upsilon}$, by applying the rate distortion algorithm of Section 4.3. We then generate an exact value for m as well as values for τ^j by using the GLA-based algorithm of Section 4.4.

A Starting Value for m

We begin by creating a starting value for m.

The starting m value is generated by applying the algorithm of Section 4.3. This algorithm creates a curve plotting $P(\epsilon)$ on the y-axis, and the smallest m that can achieve this $P(\epsilon)$ on the x-axis. Using this plot, we can establish the value for (or

range of) m required to achieve a performance near that attained with $m \to \infty$. This serves as our starting m value.

The algorithm of Section 4.3 is applied to the case at hand as follows. Step **A1** requires that we generate some starting values. The set of starting values we choose are: a_i is a BPSK signal with an energy E_s creating $\frac{E_s}{N_o} = 9$ dB; with this starting choice, the coherent performance corresponds to 3.3×10^{-5}. We also choose $s_o = -1000$, $s_{i+1} = 2 \cdot s_i$, and $s_{min} = -64000$. The last starting value that step **A1** requests is $P(\epsilon|\tau_\epsilon) = P(\epsilon|\tau_i - Q(\tau_i))$. This value is not attainable for the pulse shape $h_s(t)$ at hand. Hence, to facilitate the evaluation of this term, we approximate the pulse shape $h_s(t)$ by a unit-energy rectangular pulse shape with a duration of T. Using this approximation, we find

$$P(\epsilon|\tau_\epsilon) = \frac{1}{4}\text{erfc}(\sqrt{\frac{E_s}{N_o}}) + \frac{1}{4}\text{erfc}(\sqrt{\frac{E_s}{N_o}(1-2\tau_\epsilon)}\ ) \tag{8.17}$$

These starting values are then used in the remaining steps of the algorithm in Section 4.3, namely **B1** to **E1**. This leads to the $P(\epsilon) - m$ curve of Figure 8.6. This curve indicates that, with $m \geq 4$, nearly all the performance to be had can be achieved. Specifically, this curve shows that, with $m \geq 4$, a probability of error performance can be achieved that is less than twice the probability of error attained with $m \to \infty$.

We also generate starting values for m using a different set of values in the algorithm of Section 4.3. This time, in step **A1**, we choose: a_i's are QPSK signals with an energy E_s creating $\frac{E_s}{N_o} = 12$ dB; with this choice, coherent $P(\epsilon)$ corresponds to 3.4×10^{-5}. We also choose $s_0 = -1000$, $s_i = 2 \cdot s_{i-1}$, and $s_{min} = -64000$. The final value that is requested in **A1** is $P(\epsilon|\tau_\epsilon)$. As in the earlier case, this value is not attainable for the $h_s(t)$ of interest; hence, we approximate $h_s(t)$ by a rectangular pulse shape, which leads to (for QPSK)

$$\begin{aligned} P(\epsilon|\tau_\epsilon) \approx \frac{1}{8}\text{erfc}(\sqrt{\frac{E_s}{N_o}}\sin(\frac{\pi}{4})) + \frac{1}{8}\text{erfc}(\sqrt{\frac{E_s}{N_o}(1-2\tau_\epsilon)}\sin(\frac{\pi}{4})) \\ + \frac{1}{4}\text{erfc}(\sqrt{\frac{E_s}{N_o}(1-2\tau_\epsilon+2\tau_\epsilon^2)}\sin(\frac{\pi}{4}-\phi)), \end{aligned} \tag{8.18}$$

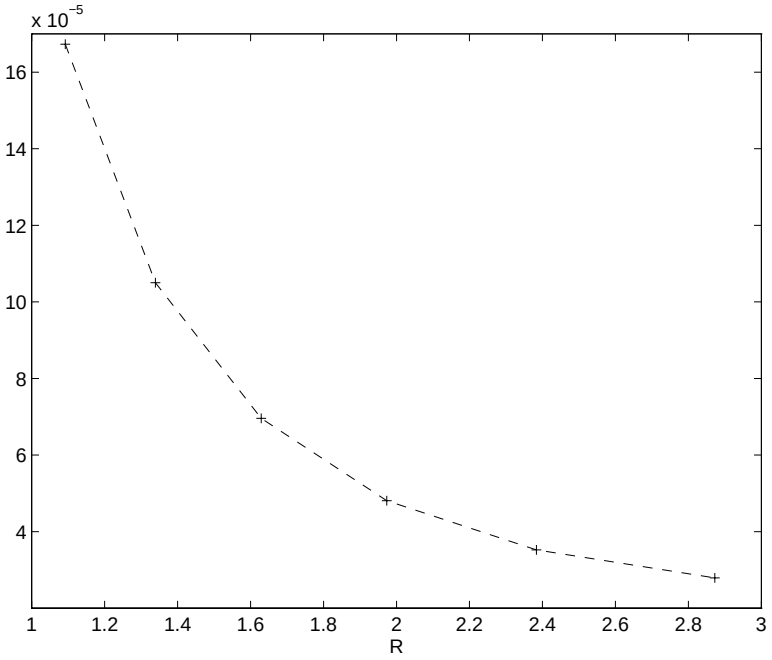

Figure 8.6: $P(\epsilon)$ *vs* $R = log_2 m$ for BPSK, $\frac{E_s}{N_o} = 9$dB.

where $\phi = \cos^{-1}(\frac{1-\tau_\epsilon}{\sqrt{(1-2\tau_\epsilon+2\tau_\epsilon^2)}})$.

We then apply these values to the remaining steps of the algorithm in Section 4.3, namely steps **B1** to **E1**. This leads to the $P(\epsilon) - m$ curve of Figure 8.7. This curve shows that, with $m \geq 8$, nearly all the performance to be had is attained.

The above two curves indicate the following. First, they point to the fact that our receiver structure can be realized with only a few parallel branches. They also indicate that, for MPSK, as constellation size grows, we require a larger number of branches in our receiver.

The Space $\tilde{\Upsilon} = \{\tau^1, ..., \tau^m\}$

In this subsection, we use the algorithm of Section 4.4 and create the exact size and elements of the discrete space $\tilde{\Upsilon} = \{\tau^1, ..., \tau^m\}$.

We begin by considering the steps at the heart of this algorithm, namely the

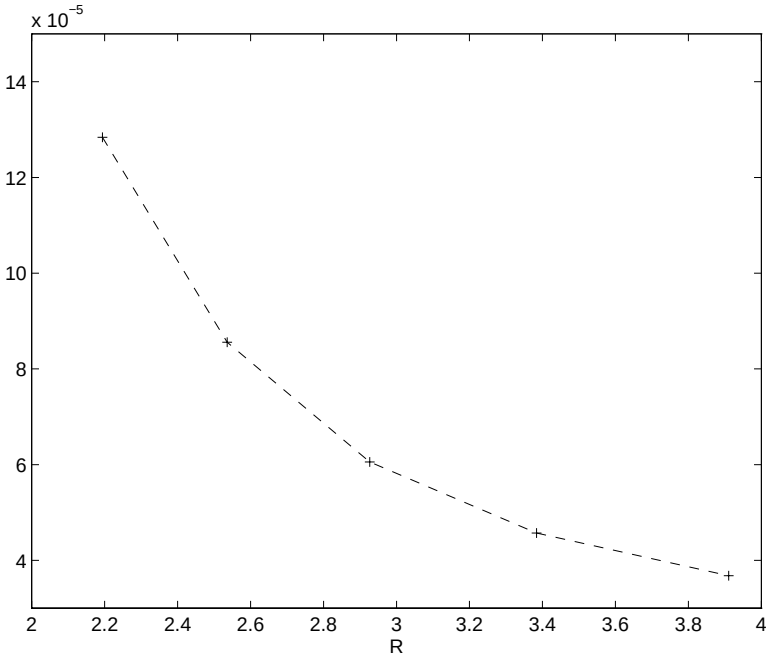

Figure 8.7: $P(\epsilon)$ *vs* $R = log_2 m$ for QPSK, $\frac{E_s}{N_o} = 12$dB.

iterative steps **B2** and **C2**, with stopping criteria **D2**. These steps establish the values $\{\tau^1, ..., \tau^m\}$ at any m.

Step **B2** generates the nearest neighbour cells R_j. In the case at hand, these can be generated by the simple rule of equation (4.36).

Step **C2** creates τ^j values by using the R_j of **B2** and a centroid computation. This centroid computation needs the value $P(\epsilon|\tau_\epsilon)$. We use the BPSK and QPSK approximate values for $P(\epsilon|\tau_\epsilon)$, namely equations (8.17) and (8.18).

The algorithm applies the above **B2** and **C2** steps iteratively, with stopping criteria **D2**, to create τ^j values. However, in the case at hand, the results of this iterative process can be established analytically. Specifically, using arguments matching those in [66, p.183], we find $\tilde{\Upsilon} = \{\frac{1}{2m}T, \frac{3}{2m}T, ..., \frac{2m-1}{2m}T\}$, i.e., $\tau^j = \frac{2j-1}{2m}T$.

The remainder of the algorithm, steps **A2** and **E2**, establish the exact value of m, using, as the starting value for m, the m values generated in the previous subsection. Applying steps **A2** and **E2**, we found an exact value for m of $m = 8$ or $m = 16$,

depending on the SNR, $h\{\cdot\}$ function, and $P(\epsilon|\tau_\epsilon)$ equation used in **A2** and **E2**. For $m = 8$, $\tilde{\Upsilon}$ corresponds to $\tilde{\Upsilon} = \{\frac{1}{16}T, \frac{3}{16}T, ..., \frac{15}{16}T\}$, i.e., $\tau^j = \frac{2j-1}{16}T$. Similarly, for $m = 16$, $\tilde{\Upsilon}$ corresponds to $\tilde{\Upsilon} = \{\frac{1}{32}T, \frac{3}{32}T, ..., \frac{31}{32}T\}$, i.e., $\tau^j = \frac{2j-1}{32}T$.

8.3 Performance and Complexity

This section presents the performance and complexity of the receiver introduced in Section 8.2, the application of our general receiver to the case at hand.

8.3.1 Performance

We begin by presenting performance. This performance is generated, by computer simulation, using a_i's corresponding to 8-PSK symbols.

Figures 8.8 to 8.13 present the performance of our proposed receiver using $P(\epsilon)$ vs $\frac{E_s}{N_o}$ plots. Specifically, Figures 8.8 to 8.11 show the performance of our receiver plotted against coherent performance. Figures 8.12 and 8.13 plot our receiver's performance against both coherent performance and the performance of a receiver based on [39]; this latter receiver employs a widely-used digital-feedforward timing offset tracking, based on the DFT, and will henceforth be referred to as the DFT receiver.

Figures 8.8 and 8.9 show the performance of our receiver when the raised cosine waveform $h_s(t)$ demonstrates a roll-off of $\alpha = 0.4$ (a practical value). These figures show that both $m = 8$ and $m = 16$ achieve performances very close to coherent, regardless of whether $N = 10$ or $N = 100$.

Figures 8.10 and 8.11 show the performance of our receiver when the raised cosine waveform $h_s(t)$ represents a *sinc* waveform, i.e., $\alpha = 0$ (a bandwidth efficient value). In this case, we observe that, $m = 8$ is insufficient to achieve a near coherent performance; however, $m = 16$ is able to achieve a performance close to coherent regardless of whether $N = 10$ or $N = 100$; only about a 0.5 dB degradation is experienced in either case.

Finally, Figures 8.12 and 8.13 demonstrate the performance of our receiver along with the coherent performance and the performance of the DFT receiver. First, Figure 8.12 shows performance results for $\alpha = 0.4$. Here, we see that the performance of our receiver, with $m = 8$ or $m = 16$, and with $N = 10$, is very close to coherent; meanwhile, the performance of the DFT receiver with $N = 100$ is close to coherent, but, at $N = 10$, the performance of the DFT receiver has degraded to the point where reliable data detection is not attainable. Figure 8.13 shows that, for $\alpha = 0$, our receiver is able to achieve a near coherent performance, while the DFT receiver is not able to achieve reliable data detection.

8.3.2 Complexity

In this subsection, we present the complexity of our receiver. This complexity is measured in terms of computations per decoded symbol.

The receiver we introduce consists of two main computational components: the universal set of demodulators and the CDU. The universal set of demodulators carry out, for each decoded symbol, m discrete time filterings, as well as m symbol-by-symbol demodulations. Meanwhile, the CDU carries out m additions and $\frac{m-1}{N}$ of a comparison for each decoded symbol. Hence, the total complexity is more or less that of m discrete time filterings and m symbol-by-symbol demodulations.

This complexity can be reduced whenever N, the number of symbols experiencing a constant τ_i, is large, e.g., $N > 10$. This reduced complexity is achieved by introducing a slight modification to the receiver implementation. We first describe the modification to the receiver implementation, and we then present the reduced complexity of the modified implementation.

Whenever τ_i is constant over N symbols, and $N > 10$, we can simplify our receiver as follows. First, over the first $K = 10$ symbols in the block of N symbols, the simplified receiver implements the proposed receiver with $N = K = 10$. This results in an output which consists of the $K = 10$ symbols, all selected from the j^{th} branch, where the value of j is determined by the CDU. Next, for the remaining $N - K$ symbols, the simplified receiver simply applies the j^{th} branch in the universal set of

demodulators; it does not carry out the operation of the remaining $m-1$ branches, nor the operation of the CDU. This is because, when considering the first 10 symbols of the block, the simplified receiver has established which branch is best.

The complexity of the modified receiver implementation, in operations per decoded symbol, is: the universal set of demodulators contribute $\frac{m \cdot K + 1 \cdot (N-K)}{N}$ discrete time filterings and an equal number of symbol-by-symbol demodulations per decoded symbol; the CDU introduces $\frac{m \cdot K}{N}$ additions and $\frac{m-1}{N}$ comparisons. Hence, the total complexity is in the order of $\frac{m \cdot K + 1 \cdot (N-K)}{N}$ discrete time filterings and symbol-by-symbol demodulations per decoded symbol. The value of m is either 8 or 16, depending on the application. For example, with $\alpha = 0.4$ (in which case we can use $m = 8$) and $N = 100$, the total complexity is in the order of 1.7 discrete time filterings and demodulations per decoded symbol; with $N = 500$, this complexity drops to 1.14 discrete time filterings and demodulations per decoded symbol.

8.3.3 Advantages of the Parallel Receiver

Our receiver demonstrates many benefits when compared to the popular receiver scheme of [39], which we call the DFT receiver.

First, there are many gains to be had in terms of performance. Whenever the block of symbols experiencing a constant τ_i is small (e.g. $N = 10$), our receiver easily outperforms the DFT receiver. In fact, to the best of our knowledge, no receiver other than our own is able to achieve reliable data detection when τ_i changes every small burst duration. Furthermore, whenever the roll-off factor α is small, our receiver easily outperforms the DFT receiver. With smaller roll-off factors becoming more and more common in mobile communications (to achieve higher bandwidth efficiency), this performance gain is an important one.

We can also compare receivers in terms of complexity. The simplified implementation of our receiver demonstrates a complexity matching that of coherent detection over the last $N - 10$ symbols in each burst. It is only over the first $K = 10$ symbols that our receiver demonstrates a slightly increased complexity. As a result, the complexity of our scheme is very low, and, for large N, it can be notably lower than that

of the DFT receiver.

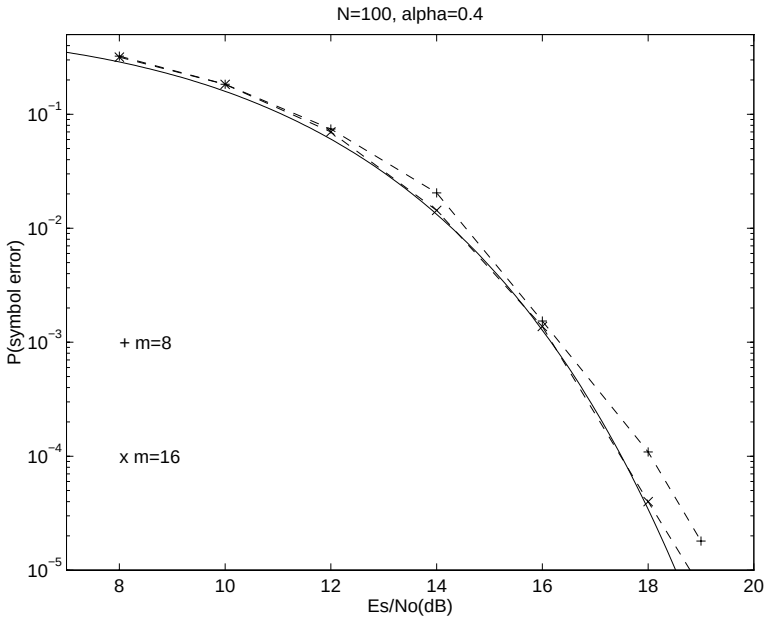

Figure 8.8: $P(\epsilon) - \frac{E_s}{N_o}$ for 8-PSK, $\alpha = 0.4$, $N = 100$.

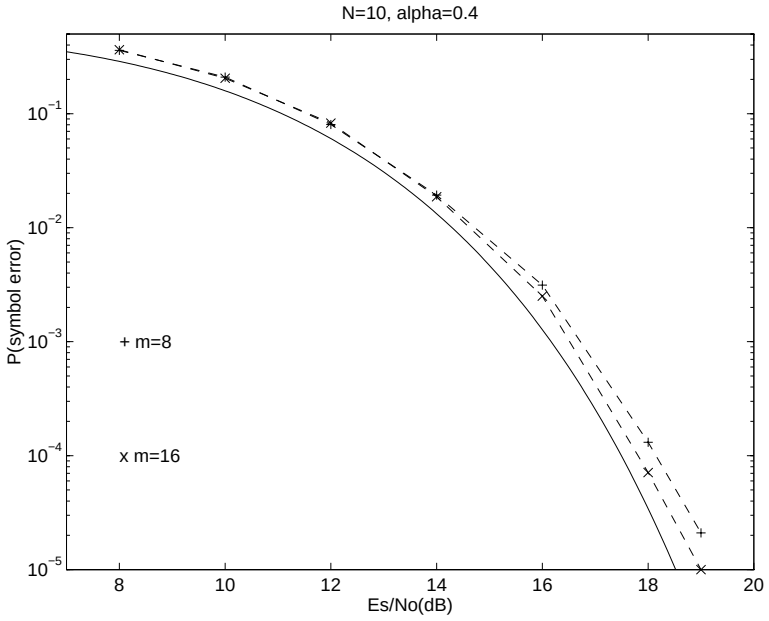

Figure 8.9: $P(\epsilon) - \frac{E_s}{N_o}$ for 8-PSK, $\alpha = 0.4$, $N = 10$.

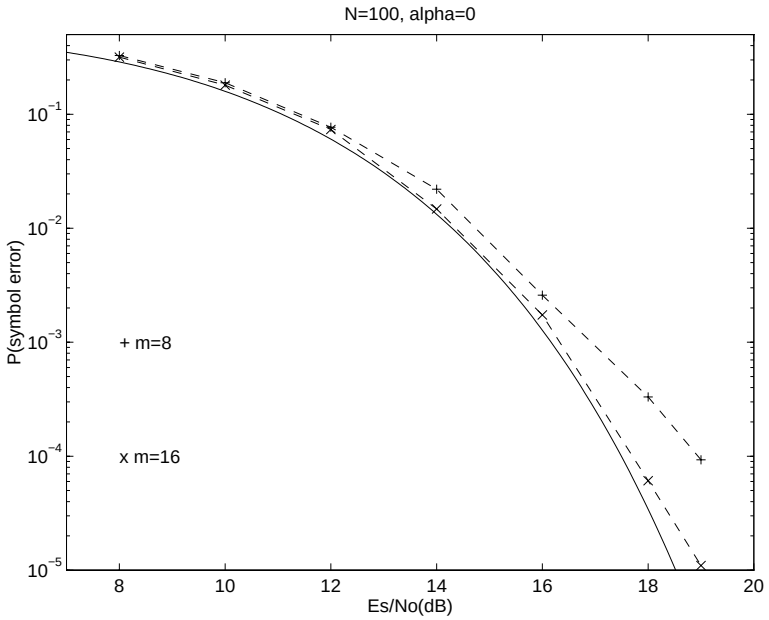

Figure 8.10: $P(\epsilon) - \frac{E_s}{N_o}$ for 8-PSK, $\alpha = 0.0$, $N = 100$.

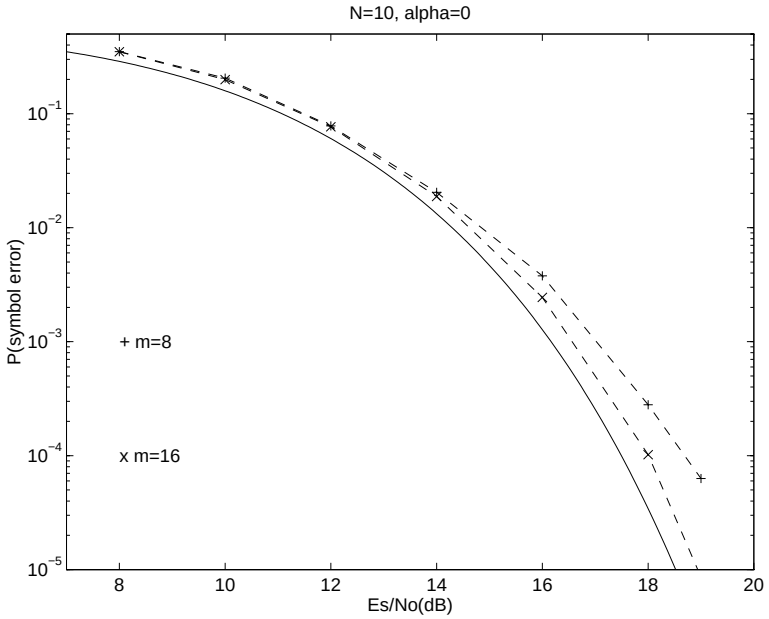

Figure 8.11: $P(\epsilon) - \frac{E_s}{N_o}$ for 8-PSK, $\alpha = 0.0$, $N = 10$.

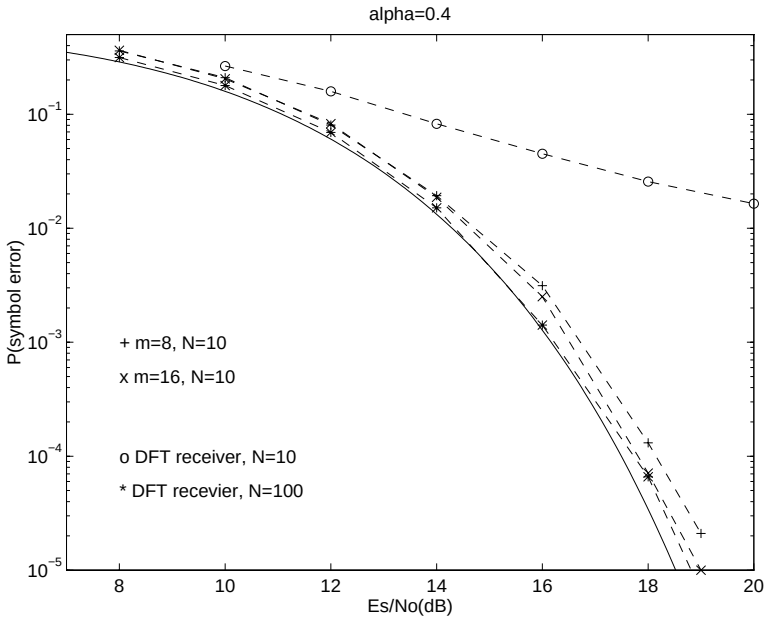

Figure 8.12: $P(\epsilon) - \frac{E_s}{N_o}$ for 8-PSK, $\alpha = 0.4$, with DFT receiver curve.

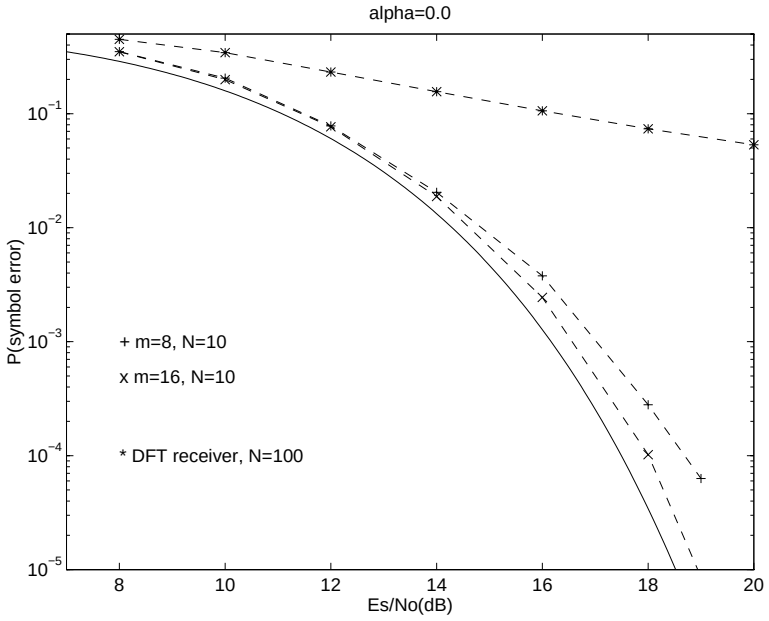

Figure 8.13: $P(\epsilon) - \frac{E_s}{N_o}$ for 8-PSK, $\alpha = 0.0$, with DFT receiver curve.

Part IV

Conclusions

Chapter 9

Conclusions and Future Work

This chapter concludes the thesis with a brief discussion of the work to date, highlighting the contributions, and an exploration of possible research for the future.

9.1 Discussion and Contributions

This thesis presents three novel equations for the detection of data in the presence of noise and nuisance parameters (which may be rapidly changing). These equations are derived by using joint MAP data detection and parameter estimation (a theoretically optimal starting point), and making one key approximation: the continuous nuisance parameter space is replaced by a carefully chosen m-element discrete space.

This thesis then introduces a corresponding, novel receiver structure. This receiver structure can implement any one of the three new data detection equations (equations for the detection of data in the presence of nuisance parameters). The structure of our receiver corresponds to implementing a set of demodulators in parallel, and following this by a computation and decision unit, or CDU for short. The parallel nature of this receiver structure facilitates its real time implementation.

Next, this thesis presents a proof showing that it is theoretically possible to apply the novel receiver structure to almost any case of data detection in the presence

of nuisance parameters. Specifically, the proof we provide shows that our receiver structure can almost always be applied to achieve a performance arbitrarily close to that of joint MAP detection and estimation (which, in turn, is usually very close to the performance of optimal MAP data detection).

This thesis then tackles the issue of the number of parallel demodulators employed by the receiver structure, an issue that may restrict the practical applicability of the receiver. We introduce a study of the number of parallel demodulators the receiver employs *versus* the performance of the receiver structure. This study is based on rate distortion theory, a branch of information theory usually reserved for data compression. As a result of this study, we achieve an algorithm that can generate a bound on the smallest number of parallel demodulators that can be used to attain a stated performance of the receiver structure. Applying this algorithm, to cases wherein the nuisance parameters correspond to a phase offset or a timing offset, we find that only a few parallel demodulators (around 10) are required by our receiver structure to achieve a performance very close to that attained with $m \to \infty$ (i.e., very close to that of joint MAP detection and estimation). Hence, our receiver is not only theoretically possible, but also practically realizable in many cases of current interest, sometimes being realized at a low complexity.

This thesis then provides an algorithm establishing the nuisance parameter values assumed at each of the demodulators in the parallel structure. This algorithm, based on the Generalized Lloyd Algorithm, generates values that optimize the receiver's performance while allowing it to maintain a low complexity.

This thesis next details several important, practical applications of our receiver structure. These relate to data detection (of both coded and uncoded modulations) in the presence AWGN and either a rapidly changing phase offset or a rapidly changing timing offset. These examples relate to communication environments such as mobile communications, burst-mode communications, and frequency-hopping communications.

The first application we examine is the detection of MPSK in the presence of AWGN and a phase offset constant over N symbols, where N is small (e.g., $N = 3$). Here, we find that our receiver structure demonstrates a low complexity, and achieves

a performance matching theoretically optimal bounds. Compared to the well-known MSDD, our receiver's performance matches that of MSDD, while maintaining a far lower complexity.

We next examine the application to data detection of MPSK in the presence of AWGN and a phase offset constant over 2 symbol intervals, but not necessarily constant over a longer duration. In this case, our proposed receiver structure easily outperforms DPSK (by about 1.5 dB) as well as MSDD (the performance gain increases as phase change becomes more rapid). Furthermore, the complexity of our receiver is in the order of eight (8) times that of a symbol-by-symbol PSK demodulator, a low complexity.

We also consider data detection of *coded* MPSK in the presence of both noise (AWGN) and a phase offset constant over N symbols (N being small, e.g. $N = 20$). Here, we find that our receiver structure corresponds to implementing four (4) coherent-like VA's in parallel, a moderate complexity. Furthermore, the performance of our receiver is very close to coherent, even with phase constant over as few as 10 symbol intervals. When compared to receivers in recent literature, we find that no receiver to date has demonstrated a performance even close to coherent, with a phase this rapidly changing, while still maintaining a realizable complexity.

Finally, we consider data detection in the presence of noise and a timing offset constant over only a received burst of symbols. Here, our receiver structure demonstrates a low to moderate complexity, and displays many performance gains. Specifically, when compared to receivers in the current literature, our receiver structure demonstrates substantial gains whenever the burst of received signals with constant timing offset is short, or whenever the roll-off factor tends toward zero (i.e., whenever the bandwidth-efficient, *sinc*-like pulse shapes are transmitted).

9.2 Future Work

This section presents possible avenues for future research. This future work can be divided into two categories: work on the novel receiver structure itself, and work on

its applications.

9.2.1 The Novel Receiver Structure

There remains some work to be done regarding the *complexity* of our receiver structure. First, whenever noise samples are not independent (and a whitening filter is not applied), the complexity of our receiver structure, as presented in this thesis, is very large, with the number of parallel demodulators corresponding to the number of possible discrete nuisance parameter *sequences*. It is hoped that future work will show that a more realizable complexity can be attained by our receiver structure in these cases. Furthermore, in cases when several nuisance parameters are contained in the received signal, it may sometimes happen that the number of parallel demodulators required by our receiver structure will be prohibitively large. Herein lies another area for future work.

Future research can also be carried out regarding the *performance* of the novel receiver structure. We have determined that the performance of our receiver structure can almost always be made arbitrarily close to that of joint MAP detection and estimation. However, to better establish the performance of our receiver, we want to compare it to that of theoretically optimal data detection (namely MAP data detection). In Chapter 2, we provide an important practical example illustrating that joint MAP detection and estimation (and hence our receiver) performs effectively the same operation as optimal data detection. We point the reader to the results in [62, p.291] for further evidence of the closeness of joint MAP detection and estimation (and hence our receiver) and optimal data detection. However, the treatment in [62, p.291] is not a complete treatment of this issue, and hence a further exploration is a topic for future research.

9.2.2 Applications

Some further work can be carried out on the receiver structure applications we have introduced to date. First, consider our second application, namely the detection of

data in the presence of a phase offset that is constant over as few as two symbol intervals. Here, we can explore the use of different phase models in the receiver structure; i.e., we have assumed a rather general phase model to achieve a widely applicable receiver, but, if the specific phase model for a particular application is known, then using this may enhance performance.

Also, consider the third application, namely data detection of coded MPSK in the presence of a phase offset constant over N symbols. In the near future, we may try to update our receiver structure such it achieves near coherent performances even in cases of $N < 10$. We hope to achieve this by introducing different phase models into our receiver structure to account for a more rapidly changing phase.

In addition, in all four applications considered in the thesis, the nuisance parameters' first order statistics corresponded to uniform distributions; this resulted in discrete nuisance parameter spaces $\tilde{C}$ made up of uniformly distributed values. There exist some closely related applications of practical interest (e.g., partially coherent detection [76]) where the communication model is unchanged from its presentations in Part III, with the exception of the first order statistics of the nuisance parameter (which is non-uniform). Our receiver structure can be introduced to these applications, in which case we no longer expect uniformly-spaced values for $\tilde{C}$, but rather a set of values clustering around areas of high probability.

The greatest potential for future research lies in the discovery of entirely new applications for our receiver structure. Our rate distortion theory algorithm suggests that our receiver structure will demonstrate a reasonable number of parallel demodulators in cases of one or two nuisance parameters and independent noise samples, a situation of great practical interest in many communication environments. Furthermore, we know that our receiver's performance will correspond very closely to detection by joint MAP detection and estimation, which itself is usually very close to optimal data detection. Our four examples have shown that, indeed, our receiver structure can achieve both a low to moderate complexity and a near-coherent performance in cases of practical interest. Of course, the applications we have considered are far from complete. For instance, we have yet to consider data detection of a PSK signal sent over a fading channel. It is our intention to apply our receiver structure

to this situation, as well as many other applications of great practical interest.

Appendix A

List of Acronyms

AFC	automatic frequency control
AWGN	additive white Gaussian noise
BPSK	binary phase shift keying
CDU	computation and decision unit
DFT	discrete Fourier transform
DPSK	differential phase shift keying
EM	expectation maximization
GLA	generalized Lloyd algorithm
GU	genie unit
ISI	intersymbol interference
LHS	left hand side
MAP	maximum *a posteriori*
ML	maximum likelihood
MPSK	M-ary phase shift keying
MSDD	multiple symbol differential detection
MTCM	multiple trellis coded modulation
PLL	phase locked loop
PSK	phase shift keying
QPSK	quadrature phase shift keying
RHS	right hand side

SNR	signal-to-noise ratio
TCM	trellis coded modulation
TDMA	time division multiple access
VA	Viterbi algorithm
VCO	voltage controlled oscillator
8-PSK	8-ary phase shift keying

Bibliography

[1] L.E. Franks, "Carrier and bit synchronization in data communication - a tutorial review," *IEEE Trans. Commun.*, Vol. 28, pp.1107-1120, Aug. 1980.

[2] B. Sklar, *Digital Communications: Fundamentals and Applications.* Englewood Cliffs, NJ: Prentice-Hall, 1988.

[3] E.A. Lee and D.G. Messerschmitt, *Digital Communication.* Boston, MA: Kluwer Academic Publishers, 1988.

[4] J.J. Spilker, *Digital Communications by Satellite.* Englewood Cliffs, NJ: Prentice-Hall, 1977.

[5] F.D. Natali, "AFC tracking algorithms", *IEEE Trans. Commun.*, Vol. 32, pp. 935-947, Aug. 1984.

[6] J.G. Proakis, *Digital Communications.* New York, NY: McGraw-Hill, 1983.

[7] W.C. Lee, *Mobile Communications Engineering.* New York, NY: McGraw-Hill, 1982.

[8] J.G. Lawton, "Investigation of digital data communication systems," Rep. no. Ua-1420-S-1, Cornell Aeronautical Laboratory Inc., Buffalo, NY, Jan. 3, 1961.

[9] A.J. Viterbi and A.M. Viterbi,"Nonlinear estimation of PSK-modulated carrier phase with application to burst digital transmission,"*IEEE Trans. Info. Theory*, Vol. IT-29, pp. 543-551, July 1983.

[10] C.N. Georghiades and J.C.Han,"Optimum decoding of trellis-coded modulation in the presence of phase errors," presented at *1990 International Symposium on Information Theory (ISITA'90)*, Hawaii, USA, Nov.27-30, 1990.

[11] J. Qin, "Demodulation of binary PSK signals," presented at *ICC'93*, Geneva, Switzerland, May 23-26, 1993, pp.498-501.

[12] C.R. Nassar and M.R. Soleymani, "Joint sequence detection and carrier phase estimation using the EM algorithm," presented at *1994 Canadian Conference on Electrical and Computer Engineering*, Halifax, Nova Scotia, Canada, Sept 25-28, 1994, pp. 296-299.

[13] P.Y. Kam and P. Sinha, "Coherent detection of coded/uncoded MPSK sequences over the Gaussian channel with unknown carrier phase using a Viterbi decoder," presented at *Singapore ICCS/ISITA'92*, Singapore, pp.1312-1316.

[14] O. Macchi and L.L. Scharf, "A dynamic programming algorithm for phase estimation and data decoding on random phase channels," *IEEE Trans. Info. Theory*, Vol. 27, pp.581-595, Sept. 1981.

[15] P.Y. Kam and H.C. Ho, "Viterbi detection with simultaneous suboptimal maximum likelihood carrier phase estimation," *IEEE Trans. Commun.*, Vol. 36, pp.1327-1330, Dec.1988.

[16] P.E.K. Chow and D.H.S. Ko, "Improving DCPSK transmission by means of error control," *IEEE Trans. Commun.*, Vol. 19, pp.715-719, Oct. 1971.

[17] M. Kato and H. Inose, "Differentially coherent detection of phase modulated waves with error correction capability," *Electr. and Comm. in Japan*, Vol. 53, pp.54-61, Dec. 1970.

[18] M. Kato, H. Inose and S. Asano, "Differentially coherent detection scheme with error-correcting capability by means of decision patterns,"*Electr. and Commun. in Japan*, Vol. 54, pp.1-8, June 1971.

[19] M. Kato and H. Inose, "Majority logic detection scheme of differentially phase-modulated waves,"*Electr. and Commun. in Japan*, Vol. 55, pp.25-33, Jan. 1972.

[20] D. Lombard and J.C. Imbeaux,"Multidifferential PSK-demodulation for TDMA transmission," presented at *1975 Int. Conf. Satellite Comm. Syst. Technology*, London, 1975.

[21] S. Samajima, K. Enomoto, and Y. Watanabe,"Differential PSK system with nonredundant error correction,"*IEEE J. Selec. Areas in Commun.*, Vol. 1, pp.74-81, Jan. 1983.

[22] D. Divsalar and M.K. Simon, "Multiple-symbol differential detection of MPSK," *IEEE Trans. Commun.*, Vol. 38, pp.300-308, Mar.1990.

[23] S.G. Wilson, J. Freebersyser, and C. Marshall,"Multi-symbol detection of M-DPSK," presented at GLOBECOM'89, Dallas, Texas, Nov. 27-30, 1989, pp.1692-1697.

[24] D. Makrakis and K. Feher,"Optimal noncoherent detection of PSK signals,"*Electronics Letters*, Vol.26, pp. 398-400, Mar.15, 1990.

[25] H. Leib and S. Pasupathy,"Optimal noncoherent block demodulation of differential phase shift keying (DPSK),"*Archiv fur Electronik und Ubertragungstechnik*, Vol. 45, pp.299-305, May 1991.

[26] H. Leib and S. Pasupathy, "The phase of vectors perturbed by Gaussian noise and differentially coherent receivers," *IEEE Trans. Info. Theory*, Vol. 34, pp.1491-1501, Nov. 1988.

[27] F. Edbauer, "Bit error rate of binary and quaternary DPSK symbols with multiple differential feedback detection," *IEEE Trans. Commun.*, Vol.40, pp. 457-460, March 1992.

[28] F. Adachi and M. Sawahashi, "Decision feedback multiple-symbol differential detection for M-ary DPSK,"*Electronics Letters*, Vol. 29, pp. 1385-1387, July 22, 1993.

[29] K.M. Mackenthun, "A fast algorithm for multiple-symbol differential detection of MPSK," *IEEE Trans. Commun.*, Vol. 42, pp. 1471-1474, April 1994.

[30] H. Leib and S. Pasupathy, "Effects of phase noise on trellis coded differentially coherent (TCDC) MPSK," presented at *GLOBECOM'86*, Houston, TX, Dec. 1-4, 1986, pp.1055-1059.

[31] H. Leib and S. Pasupathy, "Trellis-coded differentially coherent (TCDC) MPSK with carrier-phase-noise," *IEEE Trans. Commun.*, Vol. 39, pp.877-886, June 1991.

[32] H. Leib and S. Pasupathy, "Trellis-coded double differentially coherent (TCD^2C) MPSK with carrier phase noise," presented at *GLOBECOM'90*, San Diego, CA, Dec 2-5, 1990, pp.360-364.

[33] D. Divsalar, M.K. Simon, and M. Shahshahani, "The performance of trellis coded MDPSK with multiple symbol detection," *IEEE Trans. Commun.*, Vol. 38, pp.1391-1403, Sept.1990.

[34] D. Divsalar and M.K. Simon, "Multiple trellis coded modulation (MTCM)," *IEEE Trans. Commun.*, Vol. 36, pp. 410-419, April 1988.

[35] T.R. Giallorenzi and S.G. Wilson, "Noncoherent sequence demodulation for trellis coded M-DPSK," presented at MILCOM'91, McLean, VA, Nov. 4-7, 1991, pp. 1023-1027.

[36] G.K. Kaleh, "Joint carrier phase estimation and symbol decoding of trellis codes,"*European Transactions on Telecommunications and Related Technologies*, Vol. 4, pp. 157-163, Mar-Apr. 1993.

[37] Y. Miyake, M. Hagiwara, and M. Nakagawa, "Block demodulation for trellis coded modulation," *Trans. of the IEICE*, Vol. E73, pp. 1674-1680, Oct. 1990.

[38] F.M. Gardner, "A BPSK/QPSK timing-error detector for sampled receivers," *IEEE Trans. Commun.*, Vol. 34, pp 423-429, May 1986.

[39] M. Oerder and H. Meyr, "Digital filter and square timing recovery," *IEEE Trans. Commun.*, Vol. 36, pp 605-612, May 1988.

[40] L.P. Sabel and W.G. Cowley, "Block processing feedforward symbol timing estimator for digital modems," *Electronics Letters*, Vol. 30, pp. 1273-1274, Aug. 4, 1994.

[41] W.W. Harman, *Principles of the Statistical Theory of Communication*. New York, NY: McGraw-Hill, 1963.

[42] U. Madhow and M. B. Pursley, "A parallel systems approach to universal receivers," *IEEE Trans. Info. Theory*, Vol. 37, pp. 291-306, March 1991.

[43] U. Madhow, *A parallel systems approach to universal receivers*, Ph. D. thesis, University of Illinois at Urbana-Champaign, Urbana, Illinois, 1990.

[44] U. Madhow and M. B. Pursley, "On the design of universal receivers for nonselective rician-fading channels," *IEEE Trans. Commun.*, Vol. 42, pp. 2703-2712, Sept 1994.

[45] U. Madhow and M. B. Pursley, "Universal receivers with side information for the demodulators: an example for rician fading channels," *IEEE Trans. Commun.*, Vol. 42, pp. 2395-2405, July 1994.

[46] M. Alles and S. Pasupathy,"Parallel receivers and performance monitoring," presented at the *1992 Canadian Conference on Electrical and Computer Engineering*, Toronto, Ontario, Canada, Sept. 13-16, 1992, pp. WM2.23.1-WM2.23.4.

[47] E. Zervas, J. Proakis, and V. Eyuboglu,"A quantized channel approach to blind equalization," presented at *ICC'92*, Chicago, Ill., June 15-18, 1992, pp. 1539-1543.

[48] C.R. Nassar and M.R. Soleymani,"The general unknown parameter receiver," presented at the *Canadian Conference on Electrical and Computer Engineering*, Montreal, Quebec, Canada, Sept. 5-8, 1995, pp. 237-240.

[49] C.R. Nassar and M.R. Soleymani,"A novel receiver structure for MPSK in the presence of rapidly changing phase," accepted for publication in Springer-Verlag's *Lecture Notes in Computer Science*.

[50] C.R. Nassar and M.R. Soleymani,"A novel receiver structure for data detection in the presence of rapidly changing nuisance parameters," submitted for publication in *IEEE Trans. Commun.*

[51] C.R. Nassar and M.R. Soleymani,"Optimal data detection of MPSK in the presence of unknown carrier phase at low complexity," presented at the 32^{nd} *Annual Allerton Conference on Communication, Control, and Computing*, Monticello, Ill., USA, Sept. 28-30, 1994, pp. 121-123.

[52] C.R. Nassar and M.R. Soleymani,"Data detection of MPSK in the presence of unknown carrier phase at low complexity," *Electronics Letters*, Vol. 31, pp. 945-947, June 8, 1995.

[53] C.R. Nassar and M.R. Soleymani,"Data detection of MPSK in the presence of rapidly changing, random carrier phase," presented at the *1995 Canadian Workshop on Information Theory*, Lac Delage, Quebec, pp. 19.

[54] C.R. Nassar and M.R. Soleymani,"Data detection of MPSK in the presence of rapidly changing carrier phase," *IEEE Trans. on Vehicular Technology*, Vol. 45, Aug. 1996, pp. 484-490.

[55] C.R. Nassar and M.R. Soleymani,"Data detection of coded PSK in the presence of unknown carrier phase," presented at the *IEEE International Symposium on Information Theory*, Whistler, B.C., Canada, Sept. 17-22, 1995, pp.421.

[56] C.R. Nassar and M.R. Soleymani,"A novel receiver for data detection of coded PSK in the presence of unknown carrier phase," to be submitted to *IEEE Trans. Commun.*

[57] C.R. Nassar and M.R. Soleymani,"A novel receiver structure for burst-mode data detection in the presence of timing offset," submitted for presentation at *ICC'97.*

[58] C.R. Nassar and M.R. Soleymani,"Burst-mode data detection in the presence of timing offset," to be submitted to *IEEE Trans. Commun.*

[59] H.L. Van Trees, *Detection, Estimation, and Modulation Theory: Part 1.* New York, NY: Wiley, 1968.

[60] T. Kailath, "A general likelihood-ratio formula for random signals in Gaussian noise," *IEEE Trans. Info. Theory*, Vol. 15, pp. 350-361, May 1969.

[61] W.C. Dam and D.P. Taylor, "An adaptive maximum likelihood receiver for correlated Rayleigh-fading channels," *IEEE Trans. Commun.*, Vol. 42, pp. 2684-2692, Sept. 1994.

[62] C. W. Helstrom, *Statistical Theory of Signal Detection*. New York, NY: Pergamon Press, 1968.

[63] W. Feller, *An introduction to probability theory and its applications: Volume II*. New York, NY: John Wiley and sons, 1971.

[64] C.E. Shannon, "A mathematical theory of communication," *Bell Sys. Tech. Journal*, Vol. 28, pp. 379-423, 623-656, 1948.

[65] T. Berger, *Rate Distortion Theory: A Mathematical Basis for Data Compression*. Englewood Cliffs, NJ: Prentice-Hall, 1971.

[66] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*. Boston, MA: Kluwer Academic Publishers, 1992.

[67] J.C. Chen, D. Lu, J.S. Sadowsky, and K. Yao, "On importance sampling in digital communications - part 1: fundamentals," *IEEE JSAC*, Vol. 11, pp. 289-299, April 1993.

[68] R.K. Bahr and J. A. Bucklew, "Quick simulation of detector error probabilities in the presence of memory and non-linearity," *IEEE Trans. Commun.*, Vol. 41, pp. 1610-1617, Nov. 1993.

[69] P.M. Hahn and M.C. Jeruchim,"Developments in the theory and application of importance sampling," *IEEE Trans. Commun.*, Vol. 35, pp. 706-714, July 1987.

[70] P.Y. Kam, K.Y. Seek, T.T. Tjhung, and P. Sinha, "Error probability of 2DPSK with phase noise," *IEEE Trans. Commun.*, Vol. 42, pp. 2366-2369, July 1994.

[71] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Trans. Info. Theory*, Vol. 28, pp.55-66, Jan. 1982.

[72] M. Oerder, "Rotationally invariant trellis codes for MPSK modulation," presented at *ICC'85*, Chicago, Ill., June 23-26, 1985, pp. 552-556.

[73] L-F Wei, "Rotationally invariant convolutional channel coding with expanded signal space — part 1: 180^o," *IEEE JSAC*, Vol. 2, pp.659-671, Sept. 1984.

[74] L-F Wei, "Rotationally invariant convolutional channel coding with expanded signal space — part 2: nonlinear codes," *IEEE JSAC*, Vol. 2, pp.672-686, Sept. 1984.

[75] G. Ungerboeck, "Codes for QPSK modulation with invariance under 90^o rotation," presented at the *Mobile Satellite Conference*, Geneva, Switzerland, May 3-5, 1988, pp. 277-282.

[76] M.K. Simon and D. Divsalar,"Multiple symbol partially coherent detection of MPSK," *IEEE Trans. Commun.*, Vol.42, pp.430-439, Feb. 1994.