Iterative Noncoherent Detection of Differentially Encoded M-PSK

by

Ian D. Marsland

B.Sc.Eng. (Honours), Queen's University, 1987

M.A.Sc., The University of British Columbia, 1994

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY (PH.D.)

in

THE FACULTY OF GRADUATE STUDIES

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

We accept this thesis as conforming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

June 1999

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Electrical & Computer Engineering

The University of British Columbia
Vancouver, Canada

Date 24 June 1999

DE-6 (2/88)

Abstract

The combination of convolutional encoding, interleaving and differentially encoded M-ary phase shift keying (M-PSK) forms an attractive transmission scheme because it combines the benefits of error correction with the simplicity of noncoherent detection. The use of an interleaver, which rearranges the bits of the code word prior to modulation, has the potential to greatly enhance the error correcting ability of the code, provided a suitable decoder structure is used. In this thesis an iterative decoder structure, similar to the one used for decoding "turbo" codes, is proposed for use with this transmission scheme. The resulting technique, named iterative noncoherent detection, is applied to three different, well-known channel models; namely, the additive white Gaussian noise channel, the Rayleigh frequency-flat fading channel, and the Rayleigh frequency-selective fading channel. For each channel model novel soft-output decoders are derived, and the system performance, in terms of the bit error rate, is investigated by means of computer simulation. In all three applications, the proposed system works very well, not only providing performance that is significantly better than traditional non-iterative techniques, but also much better than anticipated.

Table of Contents

List of Tables

List of Figures

Glossary of Mathematical Symbols

Discrete-time signals

$\underline{a}$	-	message word
$\underline{b}$	-	word of data symbols
$\underline{c}$	-	code word
$\underline{d}$	-	word of differentially encoded symbols
$\underline{v}$	-	word of transmitted M-PSK symbols
$\underline{r}$	-	word of received samples
a_n	-	message symbol
b_n	-	data symbol
c_n	-	code symbol
d_n	-	differentially encoded symbol
g_n	-	Gray-mapped symbol
r_n	-	received sample
v_n	-	transmitted M-PSK symbol
w_n	-	AWGN sample
h_n	-	discrete-time fading process sample (flat fading)
$h_{n,k}$	-	discrete-time channel impulse response (selective fading)
$a_n^{(k)}$	-	transmitted message bit
$\hat{a}_n^{(k)}$	-	received message bit
$c_n^{(k)}$	-	code bit

Continuous-time signals

$h(t)$	-	lowpass fading process (flat fading)
$h(\tau;t)$	-	time-variant lowpass channel impulse response (selective fading)
$r(t)$	-	received lowpass signal
$r_c(t)$	-	received bandpass signal
$r_o(t)$	-	received demodulated signal
$v(t)$	-	transmitted lowpass signal
$v_c(t)$	-	transmitted bandpass signal
$w_c(t)$	-	bandpass additive white Gaussian noise

Autocorrelation functions

$\phi_{w_c}(\Delta t)$	- autocorrelation function of AWGN
$\phi_F(\Delta t)$	- autocorrelation function of fading process (flat fading)
$\phi_M(\tau)$	- multipath intensity profile (selective fading)
$\phi_S(\tau_1, \tau_2; \Delta t)$	- cross-correlation function (selective fading)
$\phi_m^{(W)}$	- discrete-time autocorrelation function of AWGN
$\phi_m^{(F)}$	- autocorrelation function of discrete-time fading
$\phi_{k,l;m}^{(S)}$	- discrete-time cross-correlation function (selective fading)
$\lambda_{k,l}$	- cross-correlation coefficients (selective fading)

Alphabets

$\mathcal{A}$	- message symbol alphabet
$\mathcal{C}$	- code symbol alphabet
$\mathcal{M}$	- data symbol alphabet
$\mathcal{S}_{\mathrm{CE}}$	- state space of convolutional code
$\mathcal{S}_{\mathrm{DE}}$	- state space of differential encoder
$\mathcal{S}_{\mathrm{Z}}$	- state space for MDD decoder
$\mathcal{S}_{\mathrm{L}}$	- state space of frequency-selective fading decoder

Mathematical operators and special functions

$\oplus$	- modulo-M addition
$*$	- complex conjugate (as superscript)
T	- vector or matrix transpose (as superscript)
H	- vector or matrix conjugate transpose (as superscript)
$\mathbf{Re}\{\bullet\}$	- real part of $\bullet$
$\mathbf{Im}\{\bullet\}$	- imaginary part of $\bullet$
$\mathbf{E}[\bullet]$	- expected value of $\bullet$
$\Pr\{\bullet\}$	- probability distribution
$f(\bullet)$	- probability density function (pdf)
$\delta(t)$	- Dirac delta function
δ_n	- Kronecker delta
$I_0(\bullet)$	- modified Bessel function of order zero
$J_0(\bullet)$	- Bessel function of the first kind of order zero

Miscellaneous system parameters

k_c	-	number of message bits per message symbol
m_c	-	number of bits per data symbol
n_c	-	number of code bits per code symbol
L	-	intersymbol interference length (selective fading)
L_c	-	constraint length of convolutional code
M	-	modulation order
N	-	number of transmitted data symbols per word
N_a	-	number of message symbols per word
N_c	-	number of code symbols per word
N_s	-	number of convolutional encoder states
Z	-	number of differential detectors (observation window size)
R_c	-	overall code rate of concatenated code
$\mathcal{E}_s$	-	energy per transmitted M-PSK symbol
$\mathcal{E}_b$	-	transmitted energy per message bit
$h_T(t)$	-	impulse response of transmit filter
$h_R(t)$	-	impulse response of receive filter
$h_{TR}(t)$	-	combined impulse response of receive and transmit filters
T	-	inverse of the symbol transmission rate (the symbol duration)
β	-	raised-cosine filter rolloff parameter
N_w	-	number of transmitted words (for simulation)
f_c	-	carrier frequency
Δf_c	-	carrier frequency offset
$\mathcal{N}_0$	-	signal-sided noise power spectral density
ϕ_c	-	carrier phase error (AWGN channel)
B_d	-	Doppler spread (fading channels)

Lookup functions

$\mathrm{GM}[\bullet]$	-	Gray map
$\mathrm{IL}[\bullet]$	-	interleaver mapping
$\mathrm{IL}^{-1}[\bullet]$	-	inverse interleaver mapping
$\mathrm{SG_{CE}}[\bullet,\bullet]$	-	symbol generation matrix of convolutional code
$\mathrm{SG_{DE}}[\bullet,\bullet]$	-	symbol generation matrix of differential encoder
$\mathrm{SG_L}[\bullet,\bullet]$	-	symbol generation matrix of frequency selective decoder
$\mathrm{SG_Z}[\bullet,\bullet]$	-	symbol generation matrix of MDD decoder
$\mathrm{ST_{CE}}[\bullet,\bullet]$	-	state transition matrix of convolutional code
$\mathrm{ST_{DE}}[\bullet,\bullet]$	-	state transition matrix of differential encoder
$\mathrm{ST_L}[\bullet,\bullet]$	-	state transition matrix of frequency selective decoder
$\mathrm{ST_Z}[\bullet,\bullet]$	-	state transition matrix of MDD decoder

Probability distributions for iterative decoding

$P_{n,b}^{(I_b)}$ - estimate of *a priori* data symbol probability
$P_{n,b}^{(O_b)}$ - estimate of *a posteriori* data symbol probability
$P_{n,b}^{(E_b)}$ - *a posteriori* data symbol extrinsic information
$P_{n,b}^{(S_c)}$ - *a posteriori* code symbol systematic information
$P_{n,b}^{(O_c)}$ - estimate of *a posteriori* code symbol probability
$P_{n,b}^{(E_c)}$ - *a posteriori* code symbol extrinsic information
$P_{n,b}^{(O_a)}$ - estimate of *a posteriori* message symbol probability

Acknowledgements

Many people were involved in the preparation of this thesis, without whose assistance it would not have been completed. I would like thank the members of my thesis examination committee and the external examiner for their time and effort spent reviewing it. In addition, I would like to thank Prof. Shlomo Shamai for his comments on some of my research for the AWGN channel, and Prof. Samir Kallel for engaging me in insightful discussions regarding the frequency-selective fading channel and other related topics. I am indebted to Jessica Dill, Cris Dunbar, and Mohamed Cherif for their generous hospitality while I was preparing the thesis. I am extremely grateful of my research supervisor, Prof. Takis Mathiopoulos, for his encouragement, support, and continued optimism during the course of my studies. His assistance in all aspects are immeasurable. I wish to thank my parents for their assistance and encouragement during my studies, and for showing me that it is, in fact, possible to get a Ph.D. while retaining one's sanity. I wish to apologize to my daughter, Kathryn, for being absent or otherwise distracted for so much of her life until now, and thank her for being such a well behaved little person. I am forever indebted to Kelly Harrison, for providing me with the opportunity to complete this thesis by keeping me healthy and fed for the last five years, and for the patience she showed by refraining from throttling me, at least until I had finished. She carried an unfair burden during my studies, and her contribution to this thesis is as great as my own.

This research was supported in part by a Natural Science and Engineering Research Council (NSERC) Post Graduate Award, NSERC OPG-443212, a BC Advanced Systems Institute (ASI) Fellowship, and a Killam Faculty Research Fellowship.

Chapter 1

Introduction

IN 1993, Berrou, Glavieux, and Thitimajshima presented a novel, powerful, class of error correcting codes, named "turbo" codes, and an efficient iterative algorithm for decoding them [1]. Although these codes yield impressive performance, and their nature has been the subject of considerable additional research (see, for example, [2–11]) it is perhaps their decoding algorithm that is the more significant contribution of the original paper. In fact, this algorithm has since been applied to numerous other problems, including decoding of serial concatenated convolutional codes (e.g. [12–14]), intersymbol-interference cancellation (e.g. [15,16]), code-division multiple access (CDMA) systems (e.g. [17–19]), systems using automatic repeat request (ARQ) protocols for error control (e.g. [20]), and joint source-channel decoding (e.g. [21, 22]). In this thesis the application of iterative decoding to noncoherent detection of M-ary phase shift keying (M-PSK) signals is proposed and investigated.

1.1 Motivation

In communication systems where information is conveyed by modulating the phase of a carrier wave, such as those employing M-PSK, care must be taken to ensure that accurate carrier phase synchronization is maintained. The transmitter for an M-PSK system selects one of M possible phases to represent the data transmitted during any particular symbol interval. Ideally, the receiver could determine the transmitted data merely by observing

the carrier's phase during the symbol interval. This would be accomplished by first demodulating the received signal with a locally derived carrier reference signal, passing the demodulated signal through a matched filter to minimize the effects of background noise, which adversely affects many communication channels, and then sampling the filter output. Finally a phase discriminator would use the sample to determine the actual data. Unfortunately, the presence of a carrier phase error (offset) obscures the transmitted data. Such an error arises unavoidably from a variety of sources, including imperfect knowledge of the propagation delay of the transmitted signal. Without accurate phase synchronization to compensate for the error, reliable communication will be hindered if not prohibited. Techniques for addressing this issue involve either *coherent* or *noncoherent* demodulation, depending on the point at which the phase error is removed. With coherent demodulation the phase error is removed during demodulation, whereas with noncoherent demodulation it is removed afterwards.

When coherent demodulation is used, the phase error is estimated in the continuous-time domain prior to demodulation, and is removed by demodulating the received signal with a carrier reference signal which has the same phase error as the carrier. One possible implementation involves transmitting a pilot tone (an unmodulated carrier) along with the data-bearing signal (see, for example, [23]). Since the pilot tone will suffer from the same phase error as the carrier but conveys no data it can be used directly as the carrier reference signal for demodulation. Alternatively, the reference signal can be generated locally at the receiver, using the same phase error as found in the pilot tone. Although both of these approaches are effective for cancelling the phase error in the carrier, the additional power and bandwidth required to transmit the pilot tone are undesirable drawbacks. A more favourable approach is to use a phase tracking loop to lock directly onto the phase error of the carrier and use this when generating the carrier reference signal. As the carrier phase

also contains the transmitted data, this approach is somewhat less reliable than using a pilot tone and suffers from phase lock slips and long phase acquisition time [24].

An alternative to coherent demodulation is noncoherent demodulation, where the phase error is estimated and removed after the received signal has been demodulated, filtered and sampled. The phase error is still present in the samples, and is removed using discrete-time processing. That is, the phase error is estimated only from the received samples and not from the continuous-time received signal. One method for assisting this task is to transmit pilot symbols (symbols with known phase) along with the data-bearing symbols (see, for example, [25]). Although the phase error is often time-variant, it usually varies fairly slowly relative to the symbol duration, so the phase error can be found by observing the pilot symbols and then removed from the data-bearing symbols. The additional power and time required to transmit the pilot symbols are drawbacks to this approach. A more commonly used method for facilitating noncoherent demodulation is to use *differentially encoded* M-PSK instead of the *absolutely encoded* M-PSK described above. With differentially encoded M-PSK, data is conveyed as the *change* of the carrier phase over two consecutive symbol intervals [24]. Under the assumption that the phase errors affecting two consecutive symbols do not differ significantly, the receiver can determine the transmitted data by observing the change in carrier phase over the two intervals. This simple approach, known as differential detection, has the advantage that the requirements on the analogue circuitry is much lower than for coherent demodulation, and is more robust without the need for carrier phase acquisition. However, because the phase error is implicitly estimated using only the previous sample, any background noise greatly impairs performance, resulting in a substantial *noncoherence penalty* over coherent demodulation[1]. To overcome this penalty, the use of

[1] Note that coherent demodulation can still be used with differentially encoded M-PSK, and an optimal receiver for these signals was recently presented in [26].

a multiple differential detector (MDD) receiver structure has been suggested [27–34]. The hardware structure of MDD receivers consists of a combination of more than one distinct differential detector, with elements of time delay equal to progressively increasing multiples of the symbol duration[2]. With this approach the previous Z samples, where Z is the number of distinct differential detectors, are used to estimate the phase error in the current sample. As such, the estimate is more accurate and the noncoherence penalty is reduced compared to when only a single differential detector is used.

Techniques for noncoherent detection of differentially encoded M-PSK are investigated in this thesis for three different, widely accepted, channel models; namely, the additive white Gaussian noise (AWGN) channel, the Rayleigh frequency-flat fading channel, and the Rayleigh frequency-selective fading channel. The first is a simple, standard channel model, in which the received signal is affected only by background noise and a static carrier phase error. The other two models, both associated with wireless communication systems, are applicable when *multipath interference* is present. Resulting from signal scattering and reflection, multipath interference occurs when the transmitted signal simultaneously follows many different paths to the receiver, arriving along each path with different attenuations and propagation delays [24, 35, 36]. Because of slight differences in propagation delays, it is possible that the multiple received versions of the modulated carrier wave used for transmission combine in a destructive fashion, cancelling one another out, yielding a combined received signal that is very weak compared to the background noise. This phenomenon, known as *multipath fading*, has severe consequences on the system's ability to communicate reliably. The distinction between frequency-flat and frequency-selective fading is a function of the *multipath spread* of the channel, which is the difference in the propagation delays

[2] In other papers on the same subject, notably [30], the same structure is referred to as multiple-symbol differential detection (MSDD) receivers.

of the shortest and longest transmission paths. Frequency-flat (or frequency-nonselective) fading occurs when the multipath spread is fairly short, much less than the symbol duration. With this channel model, the received signal strength fluctuates over time, and the received signal is affected by background noise and a time-variant carrier phase error (random FM). Frequency-selective fading occurs when the multipath spread is longer, so that a significant portion of the energy transmitted during one symbol interval is delayed sufficiently so as to interfere with information transmitted in subsequent intervals. To overcome the resulting intersymbol-interference (ISI), an equalization scheme must be used at the receiver. Unlike the static ISI found in wireline systems, however, the ISI in frequency-selective fading is time-variant, and therefore much more difficult to deal with.

For reliable communication of signals transmitted over any of these channels, channel estimation must be performed by the receiver. For the AWGN channel only the static carrier phase error must be estimated. For frequency-flat fading both the dynamic phase error and signal amplitude[3] must be estimated and tracked. In addition, for frequency-selective fading the ISI must also be estimated and tracked. When coherent detection is used, the channel is estimated in the continuous-time domain based on the analogue received signal, whereas with noncoherent detection the channel is estimated in the discrete-time domain based on the received samples.

Although accurate channel estimation is necessary for reliable communication over these channels, the presence of the background noise, which, particularly in wireless communication systems, can be rather insidious, means that some form of error control scheme is required to achieve acceptable system performance. Often this will be done with the use of an error correcting code, such as a convolutional code. Strictly controlled redundancy, as

[3] Although simple receivers for M-PSK signals transmitted over a frequency-flat fading channel can be designed without the need to estimate the signal amplitude, the sophisticated receiver proposed in this thesis requires that the amplitude be estimated.

specified by the code, is inserted into the data prior to transmission. An error correcting decoder at the receiver exploits this redundancy in an attempt to determine the data which was transmitted during those instances when the noise was particularly strong. The need for error correction is even more pronounced when the received signal is affected by multipath fading. As the wireless transmission environment is constantly changing, particularly if a user is mobile, over time the channel will experience periods of signal fading, during which the transmitted data will be lost. The use of an error correcting code can be useful in recovering data lost during fades, but to be effective the error correcting code must be able to spread the redundancy sufficiently widely so that it is unlikely that all information regarding a particular message symbol will be lost. Because of their simplicity and effectiveness, the use of a convolutional code in conjunction with interleaving is often suggested as a means for addressing the fading issue. The convolutional encoder adds redundancy and the interleaver, which merely rearranges the order of the symbols produced by the encoder, spreads it. A reverse interleaver at the receiver breaks up the faded symbols, which occur consecutively, so the decoder for the convolutional code can correct the errors caused by the fade.

The use of convolutional encoding and interleaving in conjunction with differentially encoded M-PSK is attractive for a variety of reasons. Differential encoding facilitates noncoherent detection, reducing the need for the analogue circuitry to estimate the channel and allowing for rapid adaptation to time-varying channels. Convolutional encoding allows the receiver to perform error correction, providing protection against the background noise. When used with interleaving, convolutional encoding is also effective against fading. Furthermore, the use of interleaving in this fashion permits iterative decoding.

As mentioned, iterative decoding has already proved to be useful for a variety of applications. The concept of iterative decoding is perhaps best described in the context of decoding

serial concatenated convolutional codes (SCCC's) [12]. These codes are characterized by an outer and an inner convolutional code, separated by an interleaver. To be effective, the inner code must be a recursive convolutional code, whereas the outer code may be either recursive or nonrecursive[4] The SCCC's can be decoded effectively using the iterative decoding structure presented in [13], which consists of an soft-output inner decoder for decoding the inner convolutional code, followed by a deinterleaver and a soft-output outer decoder for decoding the outer convolutional code. Any errors that the inner decoder is unable to correct will hopefully be corrected by the outer decoder. However, the real strength of these codes are better realized when soft-output reliability information produced by the outer decoder is fed back to the inner decoder. The inner decoder then decodes the inner code once again, but instead of basing its decisions only on the received observations, it uses knowledge of the probability distribution of the data entering the inner encoder, as provided by the outer decoder, to assist the decoding process, thereby increasing the accuracy of the results. By repeating the decoding process in an iterative fashion, the probability of error is drastically reduced, making SCCC's an attractive error control strategy.

1.2 Research Contributions of the Thesis

In this thesis the idea of using iterative decoding to assist noncoherent detection of data that is convolutionally encoded, interleaved, and transmitted with differentially encoded M-PSK is proposed and explored. The use of a decoding structure similar to the one described above for SCCC's is suggested. However, the inner decoder is substantially different because, instead of decoding an inner convolutional code, it must jointly reverse the effects of the differential encoding as well as the channel. To execute this task it is necessary that the inner decoder also perform channel estimation. Its implementation must be geared specifically for

[4] This is in contrast to turbo codes, where both component codes must be recursive.

the channel model.

For each of the three channel models a different, novel soft-output inner decoder is proposed. The decoders for the first two channel models are based on the MDD concept used in [29, 33], while the decoder for the frequency-selective channel makes use of Kalman filtering, similar to the approach taken by Dai and Schwedyk [37]. However, unlike the hard-output decoders presented in [29,33,37], the decoders presented here are all soft-output devices. A hard-output decoder selects, out of a set of alternatives, that alternative which it determines to be the most likely correct. A soft-output decoder, on the other hand, makes no such decision, but instead provides some measure of its confidence in each of the possible alternatives.

The soft-output decoders presented in this thesis are all implemented with the APP algorithm[5] [38, 39]. This algorithm is designed to calculate the *a posteriori* probability (APP) distribution of the input to a finite-state machine (FSM) based on noisy observations of the output. More specifically, suppose a sequence of independent input symbols are used to drive a FSM through a sequence of states. Each input symbol causes a state transition, and with each transition the FSM produces one output symbol. The APP algorithm calculates the APP distribution of each input symbol conditioned on noisy observations of all the output symbols. In doing so, the algorithm makes use of the relationship between the input and output symbols as defined by the FSM, and the relationship between the output symbols and the observations in the form of the likelihood function of each observation conditioned on the corresponding output symbol[6]. Further details, and a description of the actual APP algorithm, are provided in Appendix A. By appropriately defining the FSM model, and providing appropriately calculated branch metrics, this algorithm can be used

[5] This algorithm is also known as the MAP, Bahl, Bahl *et al.*, or BCJR algorithm.

[6] The values of these likelihood functions are also known as the branch metrics.

by the soft-output decoders presented here.

Although the focus of this thesis is on noncoherent detection, appropriate soft-output inner decoders for use with coherent detection of differentially encoded M-PSK are also presented, one for each of the three channel models. As mentioned, when noncoherent detection is used, channel estimation must be performed by the inner decoder. To measure the effectiveness of the channel estimation, it is useful to compare the performance with an ideal coherent system, where perfect knowledge of the channel is assumed to be provided to the decoder as side information. When the inner decoders for coherent detection are used in the iterative decoder structure it is then possible to determine the noncoherence penalty – the degradation in performance due to the use of noncoherent instead of coherent detection for the same received signal. In addition to the noncoherence penalty, the bit-error-rate (BER) performance of the proposed coding/modulation scheme and iterative decoding technique is investigated under a wide variety of operating conditions, for both coherent and noncoherent detection. These performance evaluation results are collected by means of extensive computer simulation.

In summary, the primary research contributions of this thesis are as follows:

1. The use of iterative decoding to assist with noncoherent detection of differentially encoded M-PSK signals is proposed.

2. Three novel soft-output decoders for noncoherent detection of differentially encoded M-PSK signals are derived for the following widely-accepted channel models:

   (a) The additive white Gaussian noise (AWGN) channel,

   (b) Rayleigh frequency-flat fading, and

   (c) Rayleigh frequency-selective fading.

3. The performance of the proposed technique for the three channel models, for both coherent and noncoherent detection, is investigated by means of computer simulation.

The results show that the proposed iterative noncoherent detection is very effective for all three channel models, with performance that is significantly better than traditional non-iterative receivers.

1.3 Organization of the Thesis

The thesis is composed of six chapters, including this introduction. Chapter 2 contains a description of the communication system model. Chapters 3, 4, and 5 are dedicated to the three channel models under consideration; namely, the AWGN channel, frequency-flat fading, and frequency selective fading, respectively. Each of these chapters contains a derivation of the inner decoders, for both coherent and noncoherent detection, along with a presentation of performance evaluation results. Conclusions and topics for future research are discussed in Chapter 6.

Chapter 2

Communication System Model

DIGITAL communication systems are generally composed of a transmitter, a transmission medium (the channel) and a receiver, and a typical model is shown in Fig. 2.1. The transmitter consists of an encoder which converts a digital message, $\underline{a}$, originating from a data source, into a sequence of signal-space symbols, $\underline{v}$, and a signal generator which converts these symbols into an analogue waveform, $v_c(t)$, that can be transmitted across the channel. As this signal propagates over the channel it is inevitably distorted, so the received signal, $r_c(t)$, differs from the transmitted signal. At the receiver, a detector demodulates, filters and samples the received signal, producing a set of received samples, $\underline{r}$. A decoder uses these samples in an attempt to determine the transmitted message. Although the decision made by the decoder, $\hat{\underline{a}}$, cannot be guaranteed to be identical to the transmitted message, a well-designed system will ensure that its decisions are the ones most likely to be correct.

Because the communication system model is specific to the particular channel model under consideration, three different system models are used in this thesis. However, because these three systems share a great deal in common, it is better to consider them as a single system model, with three possible variations, one for each channel. In particular, the encoder, signal generator, detector, and most of the decoder are common to all three system models, while the channel and part of the decoder (the inner decoder) are specific to the channel model. The common system model is described in this chapter, while descriptions

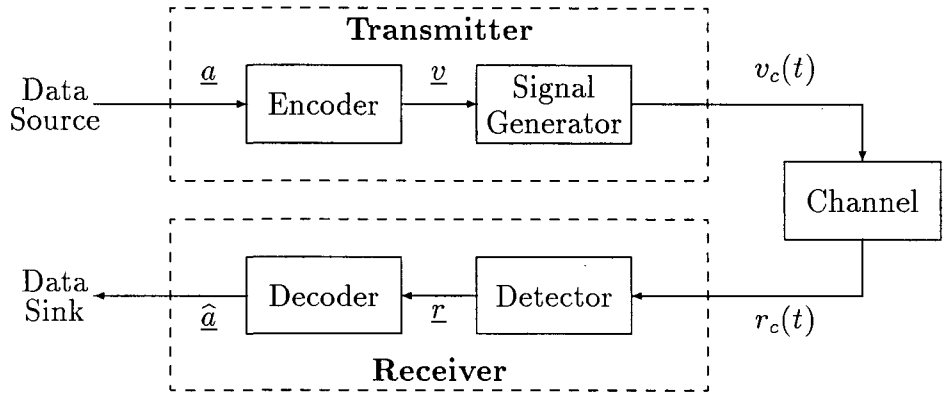


Figure 2.1: Block diagram of the communication system model.

of the channel-specific aspects are deferred to the individual chapters dedicated to each of the channel models.

The encoder, which performs the discrete-time operations of the transmitter, including convolutional encoding, interleaving and differential encoding, is described in Section 2.1. The signal generator, which produces the modulated carrier wave, is described in Section 2.2. Section 2.3 contains a very brief discussion on the channel, but most of the details are provided in later chapters. The noncoherent detector, which consists of a demodulator, a receive filter, and a symbol-rate signal sampler, is described in Section 2.4. Section 2.5 contains a description of the decoder and the iterative decoding process. Since the inner decoder must be tailored specifically to the desired channel model, different inner decoders must be used. Derivations of the inner decoders are deferred to Chapters 3, 4, and 5 for the AWGN channel, the frequency-flat fading channel, and the frequency-selective fading channel, respectively.

2.1 Encoder

As mentioned, the transmitter can be described by two components: an encoder which performs the discrete-time operations, and a signal generator which performs digital-to-analogue signal conversion. For the proposed communication system, the encoder, as shown in Fig. 2.2, is the serial concatenation of a convolutional encoder, an interleaver, and a differential encoder, followed by a signal mapper. Convolutional encoding allows for error correction at the receiver by adding redundancy to the transmitted data, the interleaver rearranges the symbols of the code word in a predetermined fashion to provide protection against error bursts due to fading, and differential encoding is used to facilitate noncoherent detection. The signal mapper produces the discrete phases that are used by the signal generator when modulating the carrier wave.

The encoder operates on finite-length message words which are transmitted and received independently of the others. Each message word, denoted by $\underline{a}$, consists of N_a symbols conveying k_c bits each. The nth message symbol is a_n, and the kth bit of the nth symbol is $a_n^{(k)}$. For the purpose of decoder design, each message symbol is modelled as a random variable drawn independently from the set $\mathcal{A} = \{0, 1, \ldots, 2^{k_c} - 1\}$ with equal *a priori* probabilities $\Pr\{a_n = a\} = 1/2^{k_c}$ for all $a \in \mathcal{A}$.

The message word is encoded with a rate k_c/n_c convolutional code with a constraint length of L_c symbols. The convolutional encoder (CE) is a finite-state machine with N_s

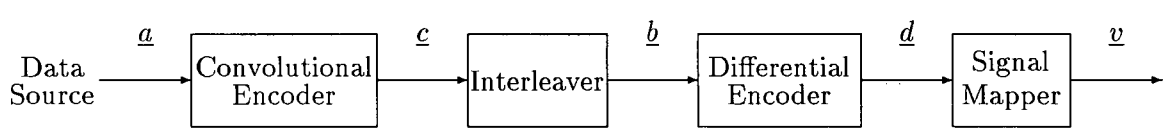


Figure 2.2: Block diagram of the encoder.

states given by the state space $\mathcal{S}_{\mathrm{CE}} = \{0, 1, 2, \ldots, N_s - 1\}$. The encoder is initialized to the zero state prior to encoding the word, and changes state as each message symbol is fed in. With each state transition, the convolutional encoder produces one code symbol of n_c bits, taken from the set $\mathcal{C} = \{0, 1, \ldots, 2^{n_c} - 1\}$. After the entire message word has been encoded an additional $L_c - 1$ zero symbols are fed in to drive the encoder back to the zero state, so the entire code word, $\underline{c}$, consists of $N_c = N_a + L_c - 1$ code symbols. Although usually defined in terms of its generator, a convolutional code can also be characterized in terms of its state transition matrix, $\mathrm{ST}_{\mathrm{CE}}[\bullet, \bullet]$, and symbol generation matrix, $\mathrm{SG}_{\mathrm{CE}}[\bullet, \bullet]$. If, at any time, the encoder is in state $s \in \mathcal{S}_{\mathrm{CE}}$, then, in response to message symbol $a \in \mathcal{A}$, the encoder would emit code symbol $\mathrm{SG}_{\mathrm{CE}}[s, a] \in \mathcal{C}$ and advance to state $\mathrm{ST}_{\mathrm{CE}}[s, a] \in \mathcal{S}_{\mathrm{CE}}$. As an example, for the convolutional encoder shown in Fig. 2.3 with generator $(23, 35)_8$, which is used for most of the simulations used for this thesis, the state transition and symbol generation matrices are as shown in Table 2.1, and the following parameters apply: $N_s = 16$, $L_c = 5$, $k_c = 1$ and $n_c = 2$.

The resulting code word is passed through an interleaver, which reorders the code symbols producing an interleaved word. The code symbols are written into an array in the order in which they are produced, but read out in some different, predetermined, order. The nth symbol of the interleaved word, denoted by b_n, is given by $b_n \triangleq c_{\mathrm{IL}[n]}$, where $\mathrm{IL}[\bullet]$ denotes the interleaver mapping function. In other communication systems employing interleaving and iterative decoding, such as those in [1, 13, 15], the choice of $\mathrm{IL}[\bullet]$ is a relevant system parameter which can have a significant impact on the bit error rate, and it is expected that the same will be the case for this application. For an interleaver size of N symbols there are a total of $N!$ different possible interleaver mappings. Some, such as the identity mapping ($\mathrm{IL}[n] = n$) are known to give poor performance, while one (or possibly more) will give optimal performance over the set of possible mappings. No attempt is made in this thesis

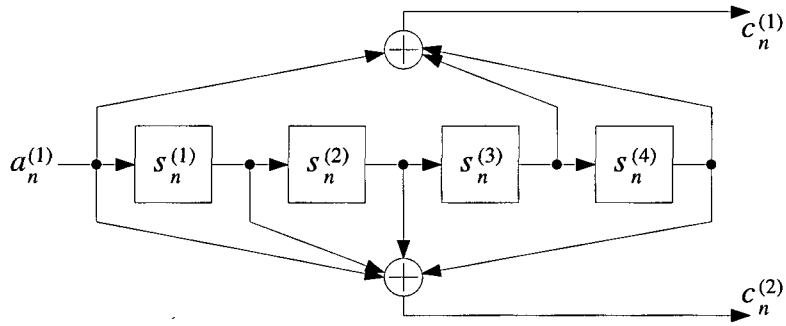


Figure 2.3: Block diagram of the 16-state, rate 1/2, convolutional encoder with generator $(23, 35)_8$.

$\mathrm{ST}_{\mathrm{CE}}[s, a]$		a	
		0	1
s	0000	0000	1000
	0001	0000	1000
	0010	0001	1001
	0011	0001	1001
	0100	0010	1010
	0101	0010	1010
	0110	0011	1011
	0111	0011	1011
	1000	0100	1100
	1001	0100	1100
	1010	0101	1101
	1011	0101	1101
	1100	0110	1110
	1101	0110	1110
	1110	0111	1111
	1111	0111	1111

(a)

$\mathrm{SG}_{\mathrm{CE}}[s, a]$		a	
		0	1
s	0000	00	11
	0001	11	00
	0010	10	01
	0011	01	10
	0100	01	10
	0101	10	01
	0110	11	00
	0111	00	11
	1000	01	10
	1001	10	01
	1010	11	00
	1011	00	11
	1100	00	11
	1101	11	00
	1110	10	01
	1111	01	10

(b)

Table 2.1: State transition (a) and symbol generation (b) matrices for the convolutional encoder in Fig. 2.3.

to find an optimal interleaver because this is an enormous task since the optimal interleaver depends not only on its size, but on the convolutional code, channel model, and inner decoder. Instead, to illustrate the effect of the choice of interleaver on system performance, three different classes of interleaver structures are investigated.

The first is a traditional block interleaver, whereby symbols are written along the rows of an $N_R \times N_C$ matrix, and read down the columns. The second is a non-uniform interleaver (NUIL) which, as described in [40], uses a simple heuristic algorithm to rearrange the symbols in a pseudo-random order. The third is an interleaver selected randomly from the set of possible mappings. The random interleaver is selected only once so the same interleaver is with all transmitted words, as opposed to randomly selecting a different interleaver with each transmitted block. Of course, care must be taken to ensure that a poor interleaver choice, such as the identity mapping, is not made. The first two types have the advantage of being structured in the sense that they may allow for simple implementation in hardware, while the third is unstructured. It is anticipated that the first type, block interleaving, will exhibit poor performance since this is the case when block interleaving is used in other iterative decoding applications, such as those in [1,13,15].

With the symbol-by-symbol interleaving described above, each code symbol remains intact, with only the symbol order changed. An alternative approach is to use bit-by-bit interleaving, where each interleaver element is a single bit as opposed to an entire symbol. Using this approach the code symbols are unpacked and stored in an array of $N_c n_c$ bits. These bits are then read out in a different order so that the individual code symbols are broken up. The interleaved bits are then regrouped into symbols of m_c bits so there is a total of $N = N_c n_c / m_c$ symbols. The parameter m_c is chosen to equal $\log_2 M$, where M is the number of different carrier phases used to transmitted data. When symbol interleaving is used it is necessary that $m_c = n_c$ (and $N = N_c$), since code symbols are not broken up.

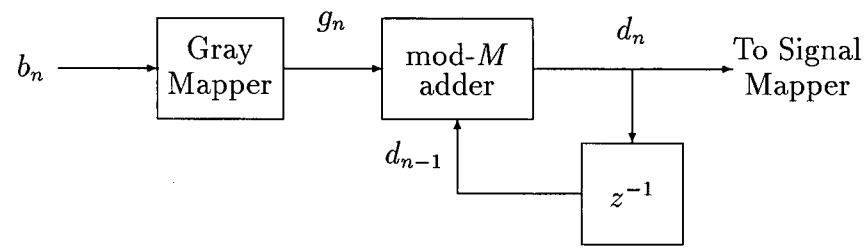


Figure 2.4: Digital M-ary differential encoder with Gray mapping, where z^{-1} denotes a delay element of one symbol.

Regardless of whether bit or symbol interleaving is used, the interleaver output is a word of symbols denoted by $\underline{b}$, with $b_n \in \mathcal{M} = \{0, 1, \ldots, M-1\}$ for all $n \in \{1, 2, \ldots, N\}$. These symbols are referred to as *data symbols*, to distinguish them from the message and code symbols.

As differentially encoded M-PSK is used for transmission, the data symbols are differentially encoded digitally prior to conversion to appropriate phases. This is accomplished with the structure shown in Fig. 2.4, which consists of a Gray mapper, a modulo-M adder and a delay element of one m_c-bit symbol. Gray mapping is generally used in conjunction with M-PSK to help reduce the probability of a bit error The interleaved symbols are Gray mapped to $g_n \triangleq \mathrm{GM}[b_n]$, where $\mathrm{GM}[\bullet]$ denotes the Gray Map, which is shown in Table 2.2a for quarternary-PSK (QPSK) (i.e. $M = 4$, $m_c = 2$) and Table 2.2b for 8-PSK ($M = 8$,

b	GM $[b]$
00	00
01	01
10	11
11	10

(a)

b	GM $[b]$	b	GM $[b]$
000	000	100	110
001	001	101	111
010	011	110	101
011	010	111	100

(b)

Table 2.2: Gray Mapping for (a) QPSK and (b) 8-PSK.

$m_c = 3$). Although Gray mapping is generally used, in some cases, such as when trellis-coded modulation (TCM) -type codes are used with symbol interleaving, the identity mapping is used with $\mathrm{GM}[b] = b$. The Gray mapped symbols are then differentially encoded, and the nth differentially encoded symbol, denoted by d_n, is given by

$$\begin{aligned} d_n &\triangleq d_{n-1} \oplus g_n \\ &= d_{n-1} \oplus \mathrm{GM}[b_n] \ , \end{aligned} \tag{2.1}$$

where $\oplus$ denotes modulo-M addition. The initial contents of the delay element are $d_0 = 0$. The first symbol, d_0, is transmitted along with the differentially encoded symbols, and is used as a reference symbol by the receiver. It is evident that differential encoding can be viewed as a form of recursive convolutional encoding, with a coding rate of unity[1] and a memory length of one symbol. The state of the differential encoder prior to encoding the nth data symbol is $s_n \triangleq d_{n-1}$, and the state space, $\mathcal{S}_{\mathrm{DE}}$, consists of M states, so $\mathcal{S}_{\mathrm{DE}} = \mathcal{M}$. Since the differentially encoded symbols are expressed recursively with Eq. (2.1), the symbol generation matrix for the differential encoder is $\mathrm{SG}_{\mathrm{DE}}[s, b] = s \oplus \mathrm{GM}[b]$, and since the encoder state at time $n+1$ is the same as the differentially encoded symbol produced at time n, the state transition matrix is $\mathrm{ST}_{\mathrm{DE}}[s, b] = s \oplus \mathrm{GM}[b]$.

The combination of the convolutional encoder, interleaver, and differential encoder can be viewed as a single block encoder with message word $\underline{a}$ producing code word $\underline{d}$. Each message word of $k_c N_a$ bits is encoded into a word of $m_c(N+1)$ bits, so the overall code rate is

$$R_c = \frac{k_c N_a}{m_c(N+1)} = \frac{k_c N_a}{m_c N + m_c} = \frac{k_c N_a}{n_c(N_a + L_c - 1) + m_c} \ . \tag{2.2}$$

Unlike many other block codes, however, this block code can be decoded with complexity that grows only linearly with the block length. Furthermore, by using differential encoding,

[1] To be precise, the coding rate is $N/(N+1)$ because of the reference symbol, but for large N the difference is negligible.

noncoherent detection can be employed at the receiver.

The final step of the encoder, and the last discrete-time operation of the transmitter, is the conversion of the differentially encoded symbols into discrete phases which will be used by the signal generator. Since M-PSK is used for transmission, the differentially encoded symbols are mapped to points in the M-PSK signal constellation using natural (direct) mapping. The transmitted symbols are

$$v_n \triangleq \sqrt{\mathcal{E}_s} \exp\left\{ j \frac{2\pi}{M} d_n \right\} , \tag{2.3}$$

where $\mathcal{E}_s$ is the transmitted energy per symbol. The transmitted energy per message bit is $\mathcal{E}_b = \mathcal{E}_s/(m_c R_c)$. The complex-valued quantities $\underline{v}$ are passed to the signal generator.

2.2 Signal Generator

The transmitted symbols are used to generate an analogue signal which is transmitted over the channel. This signal can be considered to be the result of passing an impulse stream through a pulse shaping filter and using the result to modulate a carrier wave.

The impulse stream based on the transmitted symbols, $\underline{v}$, is

$$v_i(t) \triangleq \sum_{n=0}^{N} v_n \delta(t - nT) \tag{2.4}$$

where T is the symbol duration (or, more precisely, the inverse of the symbol rate) and $\delta(\bullet)$ is the Dirac delta function. The impulse stream is passed through a pulse shaping transmit filter which limits the bandwidth of the transmitted signal. The output of the filter is the lowpass signal

$$\begin{aligned} v(t) &\triangleq \int_{-\infty}^{\infty} v_i(\tau) h_T(t - \tau)\, d\tau \\ &= \sum_{n=0}^{N} v_n h_T(t - nT) , \end{aligned} \tag{2.5}$$

where $h_T(t)$ is the filter's impulse response (the pulse shape).

For the proposed communication system the impulse response of the transmit filter must fulfill Nyquist's criterion for no intersymbol interference when used with a corresponding matched filter at the receiver [24]. Furthermore, two additional commonly used constraints on the pulse shape are imposed to simplify the analysis in the following chapters. One is that the filter neither attenuates nor amplifies the signal, and the other is that the filtering does not introduce any delay (i.e. $h_T(t)$ is centered about $t = 0$). Of course, this last requirement means the filter is non-causal, which is impossible in practice but convenient for analysis. Note that the latter two requirements are not strictly necessary, and the analysis could easily be extended to support the cases where they are not imposed. However, these extensions are inconsequential to the subject matter of the thesis.

The requirements on the pulse shape imply that $h_T(t)$ has the property that, for any integer n,

$$\int_{-\infty}^{\infty} h_T(nT + \alpha) h_T^*(\alpha)\, d\alpha = \delta_n \ , \tag{2.6}$$

where $*$ denotes complex conjugate and δ_n is the Kronecker delta, which equals one if $n = 0$ and zero if $n \neq 0$. Two potential pulse shapes possessing the required properties are the rectangular pulse with impulse response

$$h_T(t) = \begin{cases} 1/\sqrt{T} \ , \text{for } -T/2 \leq t < T/2 \\ 0 \ , \quad \text{elsewhere} \end{cases} \ , \tag{2.7}$$

and the root raised-cosine filter impulse response, with

$$h_T(t) = \frac{\sqrt{T}}{1 - \left(\frac{4\beta t}{T}\right)^2} \left[\frac{\sin\left(2\pi \left[\frac{1-\beta}{2T}\right] t\right)}{\pi t} + \frac{\cos\left(2\pi \left[\frac{1+\beta}{2T}\right] t\right)}{\frac{\pi T}{4\beta}} \right] \ , \tag{2.8}$$

where β is the rolloff parameter, which governs the excess bandwidth of the transmitted signal. While the first pulse shape has the benefit of simple implementation, the second has the advantage that the bandwidth of the transmitted signal is strictly limited to $(1+\beta)/(2T)$ Hz.

The output of the pulse shaping filter is used to modulate the carrier, producing the transmitted bandpass signal

$$v_c(t) \triangleq \mathbf{Re}\left\{v(t)\sqrt{2}\exp\left\{j2\pi f_c t\right\}\right\} , \tag{2.9}$$

where f_c is the carrier frequency and $\mathbf{Re}\{\bullet\}$ is the real part of $\bullet$. By substituting Eq. (2.5) for $v(t)$ and Eq. (2.3) for v_n, Eq. (2.9) can also be expressed as

$$v_c(t) = \sqrt{\mathcal{E}_s}\sum_{n=0}^{N} h_T(t-nT)\sqrt{2}\cos\left(2\pi f_c t + \theta_n\right) , \tag{2.10}$$

which better illustrates how the carrier phase, $\theta_n \triangleq (2\pi/M)d_n$, changes with each differentially encoded symbol, d_n. That is, the data is conveyed only in the phase of the carrier wave. However, the form of Eq. (2.9) is better suited for mathematical analysis of the communication system.

2.3 Communication Channel

As the transmitted signal propagates over the communication channel it is inevitably corrupted, so the received signal, $r_c(t)$, differs from the transmitted one, $v_c(t)$. The precise nature of the distortion is a function of the channel model under consideration. Therefore, discussion of the channel is deferred to the individual chapters dedicated to each model. Regardless of which of the three channel model is used, however, the received signal is processed in the same manner by the noncoherent detector, which is described in the next section.

2.4 Noncoherent Detector

The first stage of the receiver is the detector, which is followed by the decoder. The detector, as shown in Fig. 2.5, consists of a demodulator to convert the received signal to the

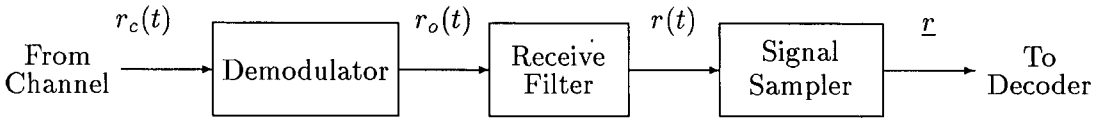


Figure 2.5: Block diagram of the detector.

baseband, a receive filter which is matched to the transmit filter to maximize the received signal-to-noise ratio (SNR), and a symbol-rate signal sampler.

The received bandpass signal is demodulated using a locally generated carrier reference signal with the same frequency as used by the transmitter. Without loss of generality, when noncoherent detection is used it is assumed that the initial phase of the reference signal is equal to zero. No attempt is made to determine the carrier phase error prior to demodulation. The demodulated signal is

$$r_o(t) \triangleq r_c(t)\sqrt{2}\exp\{-j2\pi f_c t\} \tag{2.11}$$

where $\sqrt{2}\exp\{-j2\pi f_c t\}$ is the carrier reference signal. The demodulated signal is passed through a receive filter, with impulse response $h_R(t)$, which is matched to the transmitted pulse shape, so $h_R(t) = h_T^*(-t)$. The received lowpass signal is

$$r(t) \triangleq \int_{-\infty}^{\infty} r_o(\alpha)h_R(t-\alpha)\,d\alpha\ . \tag{2.12}$$

The filtered signal is sampled at the symbol rate, and it is assumed for sampling purposes that perfect symbol synchronization is maintained, possibly by using one of the approaches outlined in [24, Ch. 6]. That is, perfect knowledge of the correct sampling instances is assumed, and the nth received sample is

$$r_n \triangleq r(nT)\ . \tag{2.13}$$

The block of $N+1$ received samples, $\underline{r}$, are passed to the decoder for processing.

2.5 *Iterative Decoder*

The purpose of the decoder is to determine the transmitted message word based on the detector output. Essentially, the decoder must reverse the function of the encoder while also taking into consideration the effects of the channel on the transmitted symbols. Given that the possibility of a decision error is unavoidable, an optimal decoder is one that maximizes the probability that correct decisions are made. Two different criteria for optimality are apparent: the decoder can either select that message word which was most likely transmitted given the received samples, or, for each message bit, the decoder can select that value which was most likely to have been transmitted. The first criterion minimizes the probability of a word error (the word error rate) while the second minimizes the probability of a bit error (the bit error rate). Although simultaneously meeting both criteria is generally not possible, a decoder that is optimal for one of the criteria is usually effective with regards to the other. The decoders presented in this thesis are designed in an attempt to fulfill the second criterion, namely minimizing the bit error rate (BER). Although sub-optimal, because they do not actually meet this criterion, these decoders are nonetheless very effective, as shown by the the performance evaluation results presented in the next three chapters. In this section the iterative decoding process, which is common to all three channel models, is described. Part of the decoder, the inner decoder, must be designed specifically for the channel model. Descriptions of the inner decoders for the various channel models under consideration are deferred to the next three chapters.

To minimize the probability of a bit error, the decoder should use the *maximum a posteriori* probability (MAP) decision rule. Consider the kth bit of the nth message symbol. If the probability that this bit is equal to a zero exceeds the probability that it is a one, given the received samples, the decoder should choose $\widehat{a}_n^{(k)} = 0$ as its output, and choose

$\widehat{a}_n^{(k)} = 1$ otherwise. That is, choose $\widehat{a}_n^{(k)} = 0$ if

$$\Pr\left\{a_n^{(k)} = 0 \,\middle|\, \underline{r}\right\} > \Pr\left\{a_n^{(k)} = 1 \,\middle|\, \underline{r}\right\} , \tag{2.14}$$

and choose $\widehat{a}_n^{(k)} = 1$ otherwise, where $\Pr\left\{a_n^{(k)} = i \,\middle|\, \underline{r}\right\}$ is the *a posteriori* probability (APP) that the kth bit of the nth message symbol is $i \in \{0,1\}$ given that the received samples are $\underline{r}$. To implement this decision rule, an optimal decoder should therefore compute the APP's of the message bits, $\Pr\left\{a_n^{(k)} = i \,\middle|\, \underline{r}\right\}$ for all $i \in \{0,1\}$, $n \in \{1,2,\ldots,N_a\}$, and $k \in \{1,2,\ldots,k_c\}$, based on the received samples. In doing so, the decoder must jointly take into consideration the convolutional encoding, the differential encoding, the phase mapping, and the channel.

In practice, unfortunately, exact computation of the APP's is computationally prohibitive for interleaved systems. However, because interleaving reduces the statistical dependency between proximous data symbols, it is possible to separate the decoding process into two stages: an inner decoder which reverses the differential encoding while taking into consideration the channel, and an outer decoder which decodes the convolutional code. Although splitting the decoder in this fashion is inherently suboptimal, much of the performance loss can be recovered by using iterative decoding.

The proposed decoder, shown in Fig. 2.6, consists of the inner and outer decoders connected in an iterative decoding structure. Although similar structures have been suggested for other applications involving iterative decoding, particularly decoding of serial concatenated convolutional codes [13], the implementation of the inner decoder is quite different as it must perform a different task. Nonetheless, the overall iterative decoding process described here is very similar, if not identical, to other iterative decoders (see, for example, [1,12,15,17,20,21]).

Using the received samples, the inner decoder attempts to compute the *a posteriori* data symbol probabilities, $\Pr\left\{b_n = b \,\middle|\, \underline{r}\right\}$, for all $b \in \mathcal{M}$ and $n \in \{1,2,\ldots,N\}$. These APP's

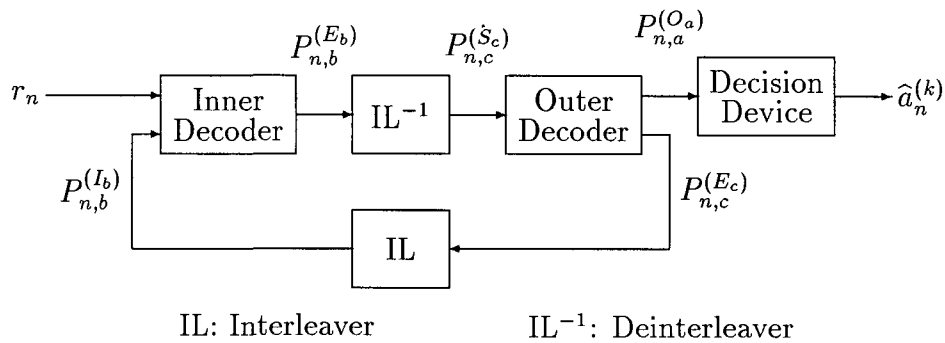


Figure 2.6: Block diagram of the iterative decoder structure.

reflect the likelihood of each of the M possible different values for b_n being transmitted given that the specific samples, $\underline{r}$, were received. To calculate the APP's, one of the algorithms given in the following chapters is used, depending on the channel model. These algorithms all take into consideration the relationship between $\underline{b}$ and $\underline{r}$, including the differential encoding, phase mapping, signal generation, the channel, and the detector. For simplicity, the algorithms all assume that the data symbols are statistically independent, although not necessarily identically distributed. However, since the data symbols are actually the output of a convolutional encoder, this assumption is not valid. Although interleaving the output of the convolutional encoder does reduce the statistical dependence between data symbols in the same vicinity, the inner decoder is nonetheless capably only of producing estimates of the APP's, denoted by $P_{n,b}^{(O_b)} \cong \Pr\left\{b_n = b \,\middle|\, \underline{r}\right\}$. For proper use by the outer decoder, the actual output of the inner decoder are these estimates with the *a priori* information removed by division. This so-called *extrinsic* information,

$$P_{n,b}^{(E_b)} = \frac{P_{n,b}^{(O_b)}}{\Pr\{b_n = b\}} \tag{2.15}$$

is deinterleaved and passed to the outer decoder.

Using the inverse of the mapping used by the interleaver in the encoder, the deinterleaver

keeps the M extrinsic information quantities associated with each data symbol together as a single unit, changing only the order of the units. The deinterleaved extrinsic information[2],

$$P_{n,c}^{(S_c)} = P_{\mathrm{IL}^{-1}[n],c}^{(E_b)} \tag{2.16}$$

is used by the outer decoder, where it is referred to as *intrinsic* information. The intrinsic information reflects the likelihood of the different values for each code symbol to have been transmitted based on the received samples. It is information gained about $\underline{c}$ solely from observation of the channel, as extracted by the detector and inner decoder. The constraints on $\underline{c}$ introduced by the convolutional encoder, in the form of statistical dependency, have not been exploited in determining the intrinsic information. These constraints are taken into consideration by the outer decoder.

The outer decoder decodes the convolutional code to produce estimates of the *a posteriori* message symbol probabilities, $P_{n,a}^{(O_a)} \cong \Pr\left\{a_n = a \,\middle|\, \underline{r}\right\}$ for all $a \in \mathcal{A}$, for each n. It is implemented with the APP algorithm developed by Bahl, Cocke, Jelinek, and Raviv [39] for soft-output decoding of convolutional codes. For convenience and notational continuity, a description of this algorithm is included as Appendix A of this thesis. For the outer decoder the algorithm is implemented based on the FSM model for the convolutional encoder described in Section 2.1. In particular, the state space $\mathcal{S}_{\mathrm{CE}}$, the state transition matrix, $\mathrm{ST}_{\mathrm{CE}}[\bullet,\bullet]$, and the symbol generation matrix, $\mathrm{SG}_{\mathrm{CE}}[\bullet,\bullet]$, are used to implement the algorithm. Referring to Appendix A, the APP algorithm uses two input sources, the *a priori* probabilities, $P_{\bullet,\bullet}^{(I_u)}$, and the branch metrics, $\mu_\bullet(\bullet)$. The *a priori* probabilities are provided to the algorithm through $P_{n,a}^{(I_u)} = \Pr\{a_n = a\} = 1/2^{k_c}$ for $n \in \{1, 2, \ldots, N_a\}$, and, since the last $L_c - 1$ symbols to enter the encoder are zeros (i.e. the encoder is terminated to the zero

[2] Note that $\mathrm{IL}^{-1}[\bullet]$ is the inverse of the interleaver mapping, $\mathrm{IL}[\bullet]$, so $\mathrm{IL}^{-1}[\mathrm{IL}[n]] = n$.

state), the algorithm uses

$$P_{n,a}^{(I_u)} = \begin{cases} 1 \text{ , if } a = 0 \\ 0 \text{ , otherwise} \end{cases} \tag{2.17}$$

for $n \in \{N_a + 1, N_a + 2, \ldots, N_c\}$[3]. The forward recursion is initialized with

$$\alpha_1(s) = \Pr\{s_1 = s\} = \begin{cases} 1 \text{ , if } s = 0 \\ 0 \text{ , otherwise} \end{cases} \tag{2.18}$$

since the encoder is initialized to the zero state prior to encoding the message word. All of these parameters and inputs are as would normally be used for decoding convolutional codes. However, the intrinsic information from the inner decoder is used instead of the normal values for the branch metrics, so $\mu_n(c) = P_{n,c}^{(S_c)}$.

As noted, separating the decoding process into these two stages is suboptimal, and leads to system performance that is quite poor. By assuming that the data symbols are independent in the inner decoder, considerable information is ignored. To compensate, the outer decoder also computes estimates of the *a posteriori* code symbol probabilities $P_{n,c}^{(O_c)} \cong \Pr\left\{c_n = c \,\middle|\, \underline{r}\right\}$. As described in Appendix A, these values are also readily computed by the APP algorithm. After the intrinsic information has been removed by division, the resulting quantities,

$$P_{n,c}^{(E_c)} = \frac{P_{n,c}^{(O_c)}}{P_{n,c}^{(S_c)}} \text{ ,} \tag{2.19}$$

are interleaved and passed back to the inner decoder. The inner decoder repeats its operation, using the same received samples, $\underline{r}$, but using the interleaved quantities from the outer decoder,

$$P_{n,b}^{(I_b)} = P_{\mathrm{IL}[n],b}^{(E_c)} \text{ ,} \tag{2.20}$$

[3] Since the message symbols are equiprobable, it is actually more convenient to use $P_{n,a}^{(I_u)} = 1/2^{k_c}$ for all n, and initialize the reverse recursion with $\beta_{N+1}(s) = 1$ for $s = 0$ and $\beta_{N+1}(s) = 0$ otherwise. This change does not affect the output of the algorithm.

in place of the *a priori* probabilities, $\Pr\{b_n = b\}$. Although the inner decoder is designed without knowledge of the constraints imposed by the outer (convolutional) code, using feedback in this fashion is an effective substitute. Using this additional information, the inner decoder recomputes the APP's of the data symbols, passing the new extrinsic information back to the outer decoder. This process is repeated several times in an iterative fashion, with the reliability of the APP's hopefully improving with each iteration, recovering some of the loss arising from the faulty independence assumption. One can imagine the following discussion occurring between the inner decoder (ID) and the outer decoder (OD):

ID: Upon careful examination of the received samples, I have concluded that the data symbols are $\underline{b}_{(1)}$.

OD: But you have neglected the convolutional code in your examination. I believe the data symbols are $\underline{b}_{(2)}$.

ID: That's nice, but you are too far removed from the channel to fully appreciate what is going on. However, incorporating some of your suggestions, I find that $\underline{b}_{(3)}$ is quite likely.

OD: That's better, but how about $\underline{b}_{(4)}$?

This bartering continues until both decoders agree on a mutually acceptable value for the data symbols.

In practice, decoding terminates after some fixed number of iterations, and the outer decoder passes $P_{n,a}^{(O_a)}$ produced on the final iteration to a decision device where hard decisions are made. Using the assumption that after sufficient iterations $P_{n,a}^{(O_a)} = \Pr\left\{a_n = a \,\middle|\, \underline{r}\right\}$, the decision device computes the *a posteriori* message bit probabilities by summing the *a posteriori* message symbol probabilities over all symbols with a value of 0 for the kth bit, and by summing over all symbols with a value of one. That is, it computes

$$\Pr\left\{a_n^{(k)} = 0 \,\middle|\, \underline{r}\right\} = \sum_{a \,|\, a^{(k)}=0} \Pr\left\{a_n = a \,\middle|\, \underline{r}\right\} \tag{2.21}$$

and

$$\Pr\left\{a_n^{(k)} = 1 \,\middle|\, \underline{r}\right\} = \sum_{a \,|\, a^{(k)}=1} \Pr\left\{a_n = a \,\middle|\, \underline{r}\right\} , \tag{2.22}$$

where $a^{(k)} \in \{0,1\}$ is the value of the kth bit of a. If the value of the first summation exceeds that of the second, the decision device chooses $\hat{a}_n^{(k)} = 0$, and chooses $\hat{a}_n^{(k)} = 1$ otherwise. Of course, since $P_{n,a}^{(O_a)}$ is only an estimate of $\Pr\left\{a_n = a \,\middle|\, \underline{r}\right\}$, the decoder is not optimal.

Although the preceding description of iterative decoding is based on symbol-by-symbol interleaving, it is also relevant to when bit-by-bit interleaving is used, except for a few minor differences. When the interleaver in the encoder is designed to operate on a bit-by-bit basis, the interleaver and deinterleaver in the decoder must work on a bit level. This requires some minor modifications to the inner and outer decoders. After the inner decoder has computed its estimates of the data symbol APP's, $P_{n,b}^{(O_b)}$, it expands them into estimates of the data bit APP's using

$$P_{n,k,i}^{(O_b)} = \sum_{b \,|\, b^{(k)}=i} P_{n,b}^{(O_b)} \cong \Pr\left\{b_n^{(k)} = i \,\middle|\, \underline{r}\right\} , \qquad \forall\, i \in \{0,1\} . \tag{2.23}$$

The actual output of the inner decoder is the extrinsic information

$$P_{n,k,i}^{(E_b)} = \frac{P_{n,k,i}^{(O_b)}}{\Pr\left\{b_n^{(k)} = i\right\}} . \tag{2.24}$$

The extrinsic information is then deinterleaved, keeping each pair of APP's associated with each bit together as a unit. The outer decoder uses the deinterleaved extrinsic information, $P_{n,k,i}^{(S_c)}$, to construct the code symbol intrinsic information by

$$P_{n,c}^{(S_c)} = \prod_{k=1}^{n_c} P_{n,k,c^{(k)}}^{(S_c)} = P_{n,1,c^{(1)}}^{(S_c)} P_{n,2,c^{(2)}}^{(S_c)} \cdots P_{n,n_c,c^{(n_c)}}^{(S_c)} , \tag{2.25}$$

which is then used by the outer decoder in the same manner as if symbol-by-symbol interleaving were used. A similar process is used with the interleaver for the feedback. From the estimates of the code symbol APP's, $P_{n,c}^{(O_c)}$, the outer decoder produces estimates of the

code bit APP's,

$$P_{n,k,i}^{(O_c)} = \sum_{c \mid c^{(k)}=i} P_{n,c}^{(O_c)} \cong \Pr\left\{c_n^{(k)} = i \;\middle|\; \underline{r}\right\} , \tag{2.26}$$

then calculates

$$P_{n,k,i}^{(E_c)} = \frac{P_{n,k,i}^{(O_c)}}{P_{n,k,i}^{(S_c)}} \tag{2.27}$$

to pass back to the inner decoder through the interleaver. Like the deinterleaver, the interleaver keeps the pair of values for each bit together as a unit, and the interleaved values, $P_{n,k,i}^{(I_b)}$, are used to construct

$$P_{n,b}^{(I_b)} = \prod_{k=1}^{m_c} P_{n,k,b^{(k)}}^{(I_b)} = P_{n,1,b^{(1)}}^{(I_b)} P_{n,2,b^{(2)}}^{(I_b)} \cdots P_{n,m_c,b^{(m_c)}}^{(I_b)} , \tag{2.28}$$

to be used by the inner decoder in place of the *a priori* data symbol probabilities, $\Pr\{b_n = b\}$.

Chapter 3

Additive White Gaussian Noise Channel

THE first application of the proposed coding/modulation scheme and iterative decoder is for the additive white Gaussian noise (AWGN) channel, in which the received signal is affected only by background noise and a static carrier phase error. Details of the communication system model specific to this channel are presented in this chapter, along with results of an in-depth investigation into the performance of the proposed system. In Section 3.1 the AWGN channel model is presented. From this continuous-time channel model, a model of the equivalent discrete-time channel, encapsulating the signal generator, the continuous-time channel, and the detector, is derived in Section 3.2. Based on this model, which provides an expression for the received samples in terms of the transmitted symbols, the inner decoder is derived in detail in Section 3.3. Different implementations of the inner decoder are required, depending on whether coherent or noncoherent detection is used. Both make use of the APP algorithm, but differ in the finite-state machine (FSM) model and branch metrics used. In addition, for noncoherent detection the inner decoder makes use of the MDD concept [29]. Following the derivation of the inner decoders, the performance evaluation results are presented and discussed in Section 3.4. Comparisons between the coherent and noncoherent systems are made, along with comparisons with more traditional non-iterative techniques.

3.1 Continuous-Time Channel Model

With the AWGN channel the transmitted signal is corrupted by background noise and a carrier phase error. The received signal, $r_c(t)$, is modelled as

$$r_c(t) = \mathbf{Re}\left\{v(t)\sqrt{2}\exp\left\{j\left(2\pi f_c t + \phi_c\right)\right\}\right\} + w_c(t) \ , \tag{3.1}$$

where $v(t)$ is the transmitted lowpass signal, f_c is the carrier frequency, ϕ_c is the carrier phase error and $w_c(t)$ is the background noise. The transmitted lowpass signal is given by Eq. (2.5). The carrier frequency is assumed to remain constant and is known perfectly by the receiver. The phase error is modelled as a constant random variable uniformly distributed over $[0, 2\pi)$. The background noise is modelled as a white Gaussian random process, with zero mean and an autocorrelation function of

$$\phi_{w_c}(\Delta t) \triangleq \mathbf{E}\left[w_c(t-\Delta t)w_c(t)\right] = \frac{\mathcal{N}_0}{2}\delta(\Delta t) \ , \tag{3.2}$$

where $\mathcal{N}_0$ is the single-sided power spectral density of the noise.

Based on the model for the continuous-time AWGN channel, a model for the equivalent discrete-time channel can be derived.

3.2 Discrete-Time Channel Model

The combined effect of the signal generator, the continuous-time channel, and the detector on the transmitted symbols is encapsulated by a discrete-time channel model. As shown in Appendix B, for the AWGN channel the received samples can be modelled as

$$r_n = v_n e^{j\phi_c} + w_n \ , \tag{3.3}$$

where $\{v_n\}$ are the transmitted symbols, ϕ_c is the carrier phase error, and $\{w_n\}$ is the discrete-time noise process. The transmitted symbols are given by Eq. (2.3) and related to

the data symbols through Eq. (2.1). The carrier phase error, ϕ_c, is uniformly distributed over the interval $[0, 2\pi)$, and remains constant over the entire block of received samples[1]. The discrete-time noise process has a complex Gaussian distribution, with zero mean and an autocorrelation function of (see Appendix B)

$$\phi_m^{(W)} \triangleq \frac{1}{2}\mathbf{E}\left[w_{n-m}^* w_n\right] = \frac{\mathcal{N}_0}{2}\delta_m \ , \tag{3.4}$$

where $\mathcal{N}_0$ is the single-sided power spectral density of the continuous-time AWGN process.

Note that this discrete-time channel model is directly applicable only to the case when noncoherent detection is used. When coherent detection is used, analogue circuitry in the detector estimates the carrier phase error prior to demodulation. Typically, this estimate is used by the demodulator to remove the phase error. Alternatively, instead of removing the phase error the detector can provide its estimate to the decoder as side information. Assuming ideal coherent detection, therefore, the model for the discrete-time channel is the same as given above, except the phase error is deterministic and the decoder has perfect knowledge of its value.

The relationship between the transmitted symbols and the received samples, as given by Eq. (3.3) for either coherent or noncoherent detection, is used to guide the design of the inner decoder.

3.3 Inner Decoder

The purpose of the inner decoder is to calculate the APP's of the data symbols entering the differential encoder, based on the received samples. In calculating these APP's, $\Pr\{b_n = b|\underline{r}\}$, use is made of the relationship between the data symbols and the received samples, as given by Eqs. (2.1) and (2.3) in conjunction with the discrete-time channel

[1] Although the decoder is designed with the constant-phase assumption, it is nonetheless fairly insensitive to modest phase variations, as shown in Section 3.4.

model. If coherent detection is used, the carrier phase error is assumed to be provided to the inner decoder as side information, whereas if noncoherent detection is used the phase error must be estimated by the inner decoder. As a result, different implementations of the inner decoder are required, depending on whether coherent or noncoherent detection is used. For both cases, however, it is assumed in designing the inner decoder that the data symbols are independent, even though this is not the case. The inner decoder is considered optimal if it correctly computes the exact values of the APP's under this assumption, and suboptimal otherwise. As mentioned in Section 2.5, iterative decoding is used to offset some, if not all, of the performance degradation due to the faulty independence assumption. In this section the inner decoders are described, one for use with coherent detection followed by one for use with noncoherent detection.

3.3.1 Coherent Detection

If the carrier phase error is known, calculating the APP's is fairly straightforward, since the problem is similar to decoding a convolutional code. The APP algorithm described in Appendix A is an optimal choice for use with the inner decoder.

The algorithm is implemented based on the FSM model for the differential encoder given in Section 2.1. At time index n the differential encoder state is $s_n \triangleq d_{n-1}$, the input is b_n, the output is $d_n = \mathrm{SG}_{\mathrm{DE}}[s_n, b_n] = s_n \oplus \mathrm{GM}[b_n]$, and the encoder advances to state $s_{n+1} = \mathrm{ST}_{\mathrm{DE}}[s_n, b_n] = s_n \oplus \mathrm{GM}[b_n]$. The FSM model describes the relationship between between the data symbols, $\{b_n\}$, and the differentially encoded symbols, $\{d_n\}$. The relationship between the differentially encoded symbols and the received samples is given by

$$r_n = \sqrt{\mathcal{E}_s} \exp\left\{ j \frac{2\pi}{M} d_n \right\} e^{j\phi_c} + w_n \ , \tag{3.5}$$

which comes from Eq. (3.3) with Eq. (2.3) used for v_n. For use with the APP algorithm this relationship is better characterized by the likelihood function, which is the conditional

probability density function (pdf) for received sample r_n, given the corresponding differentially encoded symbol, d_n, and, since coherent detection is used, the phase error, ϕ_c. Since the noise has a complex Gaussian distribution, the likelihood function is

$$f\left(r_n \,\middle|\, d_n = d, \phi_c\right) = \frac{1}{\pi \mathcal{N}_0} \exp\left\{-\frac{1}{\mathcal{N}_0}\left|r_n - \sqrt{\mathcal{E}_s}\exp\left\{j\frac{2\pi}{M}d\right\}e^{j\phi_c}\right|^2\right\} , \tag{3.6}$$

for hypothesis $d \in \mathcal{M}$. Referring to the notation of Appendix A, the APP algorithm uses $\mu_n(d) = f\left(r_n \,\middle|\, d_n = d, \phi_c\right)$ for all $d \in \mathcal{M}$ and $n \in \{1, 2, \ldots, N\}$ for the branch metrics. For the first iteration of the iterative decoder the APP algorithm uses $P_{n,b}^{(I_u)} = \Pr\{b_n = b\} = 1/M$ for the *a priori* probabilities, but in subsequent iterations the information fed back from the outer decoder is used instead, as described in Section 2.5. Since the differential encoder starts in the zero state (i.e. $s_1 \triangleq d_0 = 0$), the forward recursion is initialized with

$$\alpha_1(s) = \Pr\{d_0 = s\} = \begin{cases} 1 \text{ , if } s = 0 \\ 0 \text{ , otherwise .} \end{cases} \tag{3.7}$$

Note that the first received sample, r_0, is not used by the algorithm because it carries no useful information. By using the APP algorithm with these parameters, the inner decoder is optimal in computing the APP's (under the assumption that the data symbols are independent).

It is worth pointing out that this same decoder has a more practical application. If coherent demodulation is achieved through the use of an M-ary phase tracking loop to derive the local carrier reference signal, an unknown phase ambiguity which is some integer multiple of $2\pi/M$ will result. Denote this phase ambiguity by $\phi_c' = \frac{2\pi}{M}k'$ for some $k' \in \mathcal{M}$, so the received samples are

$$\begin{aligned} r_n &= v_n e^{j\phi_c'} + w_n \\ &= \sqrt{\mathcal{E}_s}\exp\left\{j\frac{2\pi}{M}(d_n + k')\right\} + w_n \, . \end{aligned} \tag{3.8}$$

The discrete phase ambiguity is readily absorbed by the algorithm merely by redefining the states as $s_n \triangleq d_{n-1} \oplus k'$. The same state transition and symbol generation matrices are used, along with the same branch metrics. The only change required to the algorithm is that the forward recursion be initialized with $\alpha_0(s) = 1/M$ for all $s \in \mathcal{M}$ to reflect the phase ambiguity, and to begin decoding with r_0. For block sizes of practical interest this has negligible effect on performance vis-à-vis the non-ambiguous phase case.

3.3.2 Noncoherent Detection

When noncoherent detection is used, the inner decoder must estimate the carrier phase error while decoding the differential encoding. As a result, implementation of the inner decoder is much more involved than when coherent detection is used. In this subsection it is shown that by limiting the size of the observation window over which the phase error is estimated, it is possible to derive a decoder which has reasonable complexity. Use of the APP algorithm is proposed, with the FSM model and branch metrics presented in the following.

By calculating the APP's, the inner decoder is producing information about the data symbols, $\underline{b}$. As far as the decoder is concerned there are a total of M^N different hypothetical realizations of $\underline{b}$, any one of which could have been transmitted. For $i \in \{1, 2, \ldots, M^N\}$, let $\underline{\tilde{b}}(i) = \tilde{b}_1(i), \tilde{b}_2(i), \ldots, \tilde{b}_N(i)$ be the data symbols associated with the ith hypothesis, and let $\underline{\tilde{d}}(i)$ and $\underline{\tilde{v}}(i)$ be the corresponding differentially encoded symbols and transmitted symbols associated with the ith hypothesis, respectively. One method for computing the APP's is with

$$\Pr\left\{b_n = b \,\middle|\, \underline{r}\right\} = \sum_{i \mid \tilde{b}_n(i) = b} \Pr\left\{\underline{b} = \underline{\tilde{b}}(i) \,\middle|\, \underline{r}\right\} \tag{3.9}$$

where the summation is over all hypotheses with a value of b for the nth data symbol (i.e. $\tilde{b}_n(i) = b$). Although computation of $\Pr\left\{\underline{b} = \underline{\tilde{b}}(i) \,\middle|\, \underline{r}\right\}$ is not unduly difficult, it does need

to be computed for all M^N hypotheses. As this is infeasible for large N a less burdensome approach is desired. Some insight into such an approach can be found by further examination of Eq. (3.9).

Using the shorthand notation $\underline{x}_a^b = x_a, x_{a+1}, \ldots, x_b$, it is possible to rewrite Eq. (3.9) as

$$\begin{aligned}\Pr\left\{b_n = b \,\middle|\, \underline{r}\right\} &= \sum_{i \mid \widetilde{b}_n(i)=b} \Pr\left\{\underline{b} = \underline{\widetilde{b}}(i) \,\middle|\, \underline{r}_0^{n-1}, r_n, \underline{r}_{n+1}^N\right\} \qquad (3.10)\\ &= \sum_{i \mid \widetilde{b}_n(i)=b} \Pr\left\{\underline{b} = \underline{\widetilde{b}}(i) \,\middle|\, \underline{r}_0^{n-1}\right\} \frac{f\left(r_n \,\middle|\, \underline{b} = \underline{\widetilde{b}}(i), \underline{r}_0^{n-1}\right)}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} \frac{f\left(\underline{r}_{n+1}^N \,\middle|\, \underline{b} = \underline{\widetilde{b}}(i), \underline{r}_0^n\right)}{f\left(\underline{r}_{n+1}^N \,\middle|\, \underline{r}_0^n\right)},\end{aligned}$$

where, roughly speaking, from left to right, the three multiplied terms correspond to the contribution to the APP from the past, present, and future received samples, respectively. Note that the summation is still over M^{N-1} hypotheses. The key to reducing the number of hypotheses in the summation lies within the conditional pdf $f\left(r_n \,\middle|\, \underline{b} = \underline{\widetilde{b}}(i), \underline{r}_0^{n-1}\right)$. If it can be shown that this pdf does not depend on all of the N data symbols associated with the ith hypothesis, but instead only on a small subset, a considerable reduction in complexity can be achieved.

Since the following discussion is for the ith hypothesis, the notational dependence on i has been dropped for convenience. Furthermore, because of the one-to-one relationship between $\underline{b}$ and $\underline{v}$, an expression for the equivalent pdf $f\left(r_n \,\middle|\, \underline{v} = \underline{\widetilde{v}}, \underline{r}_0^{n-1}\right)$ is given first.

In Appendix C.1 it is shown that, for the received samples given by Eq. (3.3), the conditional pdf can be written as (see Eq. (C.15))

$$f\left(r_n \,\middle|\, \underline{v} = \underline{\widetilde{v}}, \underline{r}_0^{n-1}\right) = K_n' \frac{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|r_n + \widetilde{x}_{n-1}\widetilde{v}_n\right|\right)}{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|\widetilde{x}_{n-1}\widetilde{v}_{n-1}\right|\right)}, \qquad (3.11)$$

where $K_n' = \frac{1}{\pi\mathcal{N}_0}\exp\left\{-\frac{1}{\mathcal{N}_0}\left[|r_n|^2 + \mathcal{E}_s\right]\right\}$ is a scale factor that does not depend on the hypothesis, $I_0(\bullet)$ is the modified Bessel function of order zero, and

$$\widetilde{x}_{n-1} = \frac{1}{\mathcal{E}_s}\sum_{m=0}^{n-1} r_m \widetilde{v}_m^* \qquad (3.12)$$

is an estimate of $ne^{j\phi_c}$, conditioned on the hypothesis. By using $f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right)$ to compute the APP's, the carrier phase error is implicitly estimated in the discrete-time domain. This is in contrast to coherent systems, where the phase error is explicitly estimated in the continuous-time domain prior to demodulation.

The estimate of the phase error is based on all the previously received samples, $\underline{r}_0^{n-1}$, as well as $\underline{\tilde{v}}_0^{n-1}$ of the hypothesis. So although the conditional pdf does not depend on all of $\underline{\tilde{v}}$ but only on $\underline{\tilde{v}}_0^n$, there are still M^n different possible realizations of $\underline{\tilde{v}}_0^n$. To reduce the complexity of the decoder an approach is taken that is similar to the one used in the hard-output MDD receiver given in [29].

Although ideally one would like to make use of all the previously received samples when estimating the phase error, a considerable reduction in complexity can be realized by limiting the number of samples used to some small number, Z, with Z typically less than five. That is, the size of the observation window over which the phase error is estimated is limited to Z. Instead of using $\tilde{x}_{n-1}$ to estimate the phase error, the truncated estimator

$$\tilde{x}'_{n-1} \triangleq \sum_{m=n-Z}^{n-1} r_m \tilde{v}_m^* \tag{3.13}$$

is used[2]. The truncated estimator is also unbiased, but with a larger variance than if truncation is not performed. Note that truncating the number of samples is equivalent to making the approximation $f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right) \cong f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_{n-Z}^{n-1}\right)$, which is given by substituting Eq. (3.13) for $\tilde{x}'_{n-1}$ into

$$f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_{n-Z}^{n-1}\right) = K'_n \frac{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|r_n + \tilde{x}'_{n-1}\tilde{v}_n\right|\right)}{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|\tilde{x}'_{n-1}\tilde{v}_{n-1}\right|\right)} . \tag{3.14}$$

Although truncation does degrade system performance, by limiting the size of the observation window it does allow the estimator to better track slow variations in the phase error.

[2] To be precise, the lower limit of the summation is actually $m = n - Z'$, where $Z' = \min(n, Z)$, since there are fewer than Z previous samples when $n < Z$. For clarity, this minor detail is neglected in the following, and Z' should be used in place of Z. Alternatively, one can use $r_n = 0$ for all $n \notin \{0, 1, \ldots, N\}$.

The primary advantage with using the truncated estimator, however, is that it limits the number of hypotheses that need to be examined for each n. This is best shown by expressing Eq. (3.14) in terms of the hypothetical data symbols, $\underline{\tilde{b}}$. Since $\tilde{v}_n = \sqrt{\mathcal{E}_s}\exp\left\{j\frac{2\pi}{M}\tilde{d}_n\right\}$ and $\tilde{d}_n = \tilde{d}_{n-1} \oplus \mathrm{GM}\left[\tilde{b}_n\right]$, it is clear that $\tilde{v}_n = \tilde{v}_{n-1}\exp\left\{j\frac{2\pi}{M}\mathrm{GM}\left[\tilde{b}_n\right]\right\}$. By defining $\tilde{y}'_{n-1} \triangleq \tilde{x}'_{n-1}\tilde{v}_{n-1}$, the conditional pdf can be written in terms of the data symbols instead of the transmitted symbols as

$$f\left(r_n \,\middle|\, \underline{b} = \underline{\tilde{b}}, \underline{r}_{n-Z}^{n-1}\right) = K'_n \frac{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|r_n + \tilde{y}'_{n-1}\exp\left\{j\frac{2\pi}{M}\mathrm{GM}\left[\tilde{b}_n\right]\right\}\right|\right)}{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|\tilde{y}'_{n-1}\right|\right)} \,. \tag{3.15}$$

Furthermore, $\tilde{y}'_{n-1}$ can be written as

$$\tilde{y}'_{n-1} \triangleq \tilde{x}'_{n-1}\tilde{v}_{n-1} = \frac{1}{\mathcal{E}_s}\sum_{m=n-Z}^{n-1} r_m \tilde{v}_m^* \tilde{v}_{n-1} \,, \tag{3.16}$$

which, when expressed in terms of the differentially encoded symbols is

$$\begin{aligned} \tilde{y}'_{n-1} &= \frac{1}{\mathcal{E}_s}\sum_{m=n-Z}^{n-1} r_m \mathcal{E}_s \exp\left\{j\frac{2\pi}{M}\left(\tilde{d}_{n-1} - \tilde{d}_m\right)\right\} \\ &= \sum_{z=1}^{Z} r_{n-z}\exp\left\{j\frac{2\pi}{M}\left(\tilde{d}_{n-1} - \tilde{d}_{n-z}\right)\right\} \,. \end{aligned} \tag{3.17}$$

Expanding the recursive definition of $\tilde{d}_n$ yields

$$\begin{aligned} \tilde{y}'_{n-1} &= \sum_{z=1}^{Z} r_{n-z}\exp\left\{j\frac{2\pi}{M}\left(\left[\sum_{l=1}^{n-1}\mathrm{GM}\left[\tilde{b}_l\right]\right] - \left[\sum_{l=1}^{n-z}\mathrm{GM}\left[\tilde{b}_l\right]\right]\right)\right\} \\ &= \sum_{z=1}^{Z} r_{n-z}\exp\left\{j\frac{2\pi}{M}\sum_{l=n-z+1}^{n-1}\mathrm{GM}\left[\tilde{b}_l\right]\right\} \\ &= \sum_{z=1}^{Z} r_{n-z}\exp\left\{j\frac{2\pi}{M}\sum_{k=1}^{z-1}\mathrm{GM}\left[\tilde{b}_{n-k}\right]\right\} \,, \end{aligned} \tag{3.18}$$

where the value of the k-indexed summation is defined to equal zero when $z = 1$. Clearly $\tilde{y}'_{n-1}$ depends only on $\underline{r}_{n-Z}^{n-1}$ and $\underline{\tilde{b}}_{n-Z+1}^{n-1}$. Therefore,

$$f\left(r_n \,\middle|\, \underline{b} = \underline{\tilde{b}}, \underline{r}_{n-Z}^{n-1}\right) = f\left(r_n \,\middle|\, \underline{b}_{n-Z+1}^{n} = \underline{\tilde{b}}_{n-Z+1}^{n}, \underline{r}_{n-Z}^{n-1}\right) \,. \tag{3.19}$$

For the given received samples, the number of distinct values of this conditional pdf is limited to M^Z, regardless of how large n becomes.

Since the conditional pdf depends only on a limited number of the data symbols, trellis-based decoding is suggested, and a trivial FSM model for the trellis is readily constructed. Define the state at time index n as the previous $Z-1$ data symbols, so $s_n \triangleq \underline{b}_{n-Z+1}^{n-1}$, and the state space, $\mathcal{S}_{\mathrm{Z}}$, as the set of all M-ary (Z–1)-tuples. Hypothetical state $s \in \mathcal{S}_{\mathrm{Z}}$ can be described in terms of its elementary symbols as

$$s = \left(s^{(Z-1)}, s^{(Z-2)}, \ldots, s^{(2)}, s^{(1)}\right) , \tag{3.20}$$

where at time index n, $s^{(k)} \in \mathcal{M}$ corresponds to the hypothetical value of b_{n-k}. From each of the M^{Z-1} states there are M branches corresponding to the M possible values of b_n. The FSM model is defined by the state transition matrix

$$\mathrm{ST}_{\mathrm{Z}}\left[s, b\right] = \left(s^{(Z-2)}, \ldots, s^{(2)}, s^{(1)}, b\right) , \tag{3.21}$$

for all $s \in \mathcal{S}_{\mathrm{Z}}$, so

$$\mathrm{ST}_{\mathrm{Z}}\left[s_n, b_n\right] = \mathrm{ST}_{\mathrm{Z}}\left[\underline{b}_{n-Z+1}^{n-1}, b_n\right] = (b_{n-Z+2}, \ldots, b_{n-2}, b_{n-1}, b_n) = \underline{b}_{n-Z+2}^{n} = s_{n+1} . \tag{3.22}$$

The input alphabet is $\mathcal{M}$, and the output alphabet is the set of all M-ary Z-tuples, corresponding to $\underline{b}_{n-Z+1}^{n}$. The symbol generation matrix for the FSM is

$$\mathrm{SG}_{\mathrm{Z}}\left[s, b\right] = \left(s^{(Z-1)}, s^{(Z-2)}, \ldots, s^{(2)}, s^{(1)}, b\right) , \tag{3.23}$$

so

$$\mathrm{SG}_{\mathrm{Z}}\left[s_n, b_n\right] = \mathrm{SG}_{\mathrm{Z}}\left[\underline{b}_{n-Z+1}^{n-1}, b_n\right] = (b_{n-Z+1}, b_{n-Z+2}, \ldots, b_{n-2}, b_{n-1}, b_n) = \underline{b}_{n-Z+1}^{n} . \tag{3.24}$$

Associated with the branch corresponding to the hypothetical output symbol $\underline{\widetilde{b}}_{n-Z+1}^{n}$ is the branch metric

$$\begin{aligned} \mu_n(\underline{\widetilde{b}}_{n-Z+1}^{n}) &= f\left(r_n \,\middle|\, \underline{b}_{n-Z+1}^{n} = \underline{\widetilde{b}}_{n-Z+1}^{n}, \underline{r}_{n-Z}^{n-1}\right) \\ &= K_n' \frac{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|r_n + \widetilde{y}_{n-1}' \exp\left\{j\frac{2\pi}{M}\mathrm{GM}\left[\widetilde{b}_n\right]\right\}\right|\right)}{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|\widetilde{y}_{n-1}'\right|\right)} \end{aligned} \tag{3.25}$$

with

$$\tilde{y}'_{n-1} = \sum_{z=1}^{Z} r_{n-z} \exp\left\{ j\frac{2\pi}{M} \sum_{k=1}^{z-1} \mathrm{GM}\left[\tilde{b}_{n-k}\right] \right\} . \tag{3.26}$$

Note that this FSM model is artificial; it is not to be confused with the FSM model associated with the differential encoder.

Using this FSM model it is straightforward to calculated the APP's. Note that

$$\begin{aligned} \Pr\left\{b_n = b \,\middle|\, \underline{r}\right\} &= \sum_{s\in\mathcal{S}_Z} \Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}\right\} \\ &= \sum_{s\in\mathcal{S}_Z} \Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^{n-1}\right\} \times \frac{f\left(r_n \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right)}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} \\ &\qquad \times \frac{f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n}\right)}{f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{r}_0^{n}\right)} , \end{aligned} \tag{3.27}$$

where, as with Eq. (3.10), the three multiplied terms correspond to the contribution to the APP from the past, present, and future received samples, respectively. Each of these three terms can all be simplified somewhat. Since the received samples are causally dependent on the data symbols, $\underline{r}_0^{n-1}$ does not depend on b_n (and vice versa). Similarly, $s_n \triangleq \underline{b}_{n-Z+1}^{n-1}$ does not depend on b_n. Therefore, the first term can be simplified to

$$\Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^{n-1}\right\} = \Pr\left\{b_n = b\right\} \Pr\left\{s_n = s \,\middle|\, \underline{r}_0^{n-1}\right\} . \tag{3.28}$$

By making use of the definition of $\mathrm{SG_Z}[\bullet,\bullet]$, the numerator of the second term can be expressed as

$$\begin{aligned} f\left(r_n \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right) &= f\left(r_n \,\middle|\, \underline{b}_{n-Z+1}^{n-1} = s, b_n = b, \underline{r}_0^{n-1}\right) \\ &= f\left(r_n \,\middle|\, \underline{b}_{n-Z+1}^{n} = \mathrm{SG_Z}\left[s, b\right], \underline{r}_0^{n-1}\right) . \end{aligned} \tag{3.29}$$

To simplify this term, note that although r_n depends on $\underline{r}_0^{n-1}$ for estimating the phase error, truncating the estimator to the previous Z samples provides an acceptable approximation. By arguing that the phase error is estimated with sufficient accuracy using only $\underline{r}_{n-Z}^{n-1}$, then

r_n does not depend on $\underline{r}_0^{n-Z-1}$. Therefore

$$\begin{aligned} f\left(r_n \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right) &\cong f\left(r_n \,\middle|\, \underline{b}_{n-Z+1}^{n} = \mathrm{SG_Z}\left[s, b\right], \underline{r}_{n-Z}^{n-1}\right) \\ &= \mu_n(\mathrm{SG_Z}\left[s, b\right]) \ , \end{aligned} \tag{3.30}$$

which is given by Eq. (3.25). Using the same argument, the numerator of the third term can also be simplified. When truncation is used, it is implied that $\underline{r}_{n+1}^{N}$ does not depend on $\underline{r}_0^{n-Z}$, so

$$\begin{aligned} f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n}\right) &\cong f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_n = s, b_n = b, \underline{r}_{n-Z+1}^{n}\right) \\ &= f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{b}_{n-Z+1}^{n-1} = s, b_n = b, \underline{r}_{n-Z+1}^{n}\right) \ . \end{aligned} \tag{3.31}$$

This, in turn, does not depend on b_{n-Z+1}. Making use of the definition of $\mathrm{ST_Z}\left[\bullet, \bullet\right]$ yields

$$\begin{aligned} f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n}\right) &\cong f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{b}_{n-Z+2}^{n} = \mathrm{ST_Z}\left[s, b\right], \underline{r}_{n-Z+1}^{n}\right) \\ &= f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_{n+1} = \mathrm{ST_Z}\left[s, b\right], \underline{r}_{n-Z+1}^{n}\right) \\ &\cong f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_{n+1} = \mathrm{ST_Z}\left[s, b\right], \underline{r}_0^{n}\right) \ . \end{aligned} \tag{3.32}$$

By combining Eqs. (3.28), (3.30), and (3.32) into Eq. (3.27), the APP's can be expressed as

$$\begin{aligned} \Pr\left\{b_n = b \,\middle|\, \underline{r}\right\} \cong \sum_{s \in \mathcal{S}_Z} \Pr\left\{b_n = b\right\} \Pr\left\{s_n = s \,\middle|\, \underline{r}_0^{n-1}\right\} \frac{\mu_n(\mathrm{SG_Z}\left[s, b\right])}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} \\ \times \quad \frac{f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_{n+1} = \mathrm{ST_Z}\left[s, b\right], \underline{r}_0^{n}\right)}{f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{r}_0^{n}\right)} \ , \end{aligned} \tag{3.33}$$

where the approximation is due strictly to the inaccuracy introduced by limiting the size of the window over which the carrier phase error is estimated.

The expression for the APP's given in Eq. (3.33) is in a form suitable for evaluating with the APP algorithm. This is emphasized by defining

$$\Omega_n \triangleq f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right) \ , \tag{3.34}$$

$$\alpha_n(s) \triangleq \Pr\left\{s_n = s \,\middle|\, \underline{r}_0^{n-1}\right\} \ , \text{ and,} \tag{3.35}$$

$$\beta_{n+1}(s) \triangleq \frac{f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_{n+1} = s, \underline{r}_0^{n}\right)}{f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{r}_0^{n}\right)} \tag{3.36}$$

so the APP's are given by

$$\Pr\left\{b_n = b \,\middle|\, \underline{r}\right\} \cong \frac{1}{\Omega_n}\Pr\{b_n = b\} \sum_{s\in\mathcal{S}_Z} \alpha_n(s)\mu_n(\mathrm{SG_Z}[s,b])\beta_{n+1}(\mathrm{ST_Z}[s,b]) \,. \tag{3.37}$$

A comparison with Eq. (A.9) reveals that Eq. (3.37) has the same form as the core expression in the APP algorithm described in Appendix A. In fact, as shown below, it is easy to prove that the APP algorithm is capable of computing $\Pr\left\{b_n = b \,\middle|\, \underline{r}\right\}$, provided the error in using truncation is ignored. Note that for $s' \in \mathcal{S}_Z$

$$\begin{aligned}
\alpha_{n+1}(s') &\triangleq \Pr\{s_{n+1} = s' | \underline{r}_0^n\} \\
&= \sum_{s\in\mathcal{S}_Z}\sum_{b\in\mathcal{M}} \Pr\left\{s_n = s, b_n = b, s_{n+1} = s' \,\middle|\, \underline{r}_0^n\right\} \\
&= \sum_{s\in\mathcal{S}_Z}\sum_{b\in\mathcal{M}} \Pr\left\{s_n = s, b_n = b \,\middle|\, r_n, \underline{r}_0^{n-1}\right\} \Pr\left\{s_{n+1} = s' \,\middle|\, s_n = s, b_n = b, \underline{r}_0^n\right\} \\
&= \sum_{s\in\mathcal{S}_Z}\sum_{b\in\mathcal{M}} \Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^{n-1}\right\} \frac{f\left(r_n \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right)}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} \\
&\qquad\qquad \times \quad \Pr\left\{s_{n+1} = s' \,\middle|\, s_n = s, b_n = b\right\}
\end{aligned} \tag{3.38}$$

where

$$\Pr\left\{s_{n+1} = s' \,\middle|\, s_n = s, b_n = b\right\} = \begin{cases} 1 \,, \text{ if } s' = \mathrm{ST_Z}[s,b] \\ 0 \,, \text{ otherwise} \,. \end{cases} \tag{3.39}$$

Making use of Eqs. (3.28), (3.30), (3.34) and (3.35), and ignoring the error due to truncation, yields

$$\alpha_{n+1}(s') = \frac{1}{\Omega_n}\sum_{s\in\mathcal{S}}\sum_{b\in\mathcal{M}} \Pr\{b_n = b\}\, \alpha_n(s)\mu_n(\mathrm{SG_Z}[s,b])\Pr\left\{s_{n+1} = s' \,\middle|\, s_n = s, b_n = b\right\} \,, \tag{3.40}$$

which corresponds to the recursive expression for $\alpha_{n+1}(s')$ given by Eq. (A.5) in Appendix A. To calculate the correct values for $\alpha_n(s)$, the recursion must be initialized with

$$\alpha_1(s) = \begin{cases} 1 \,, \text{ if } s = 0 \\ 0 \,, \text{ otherwise} \,. \end{cases} \tag{3.41}$$

Similarly, the recursive expression for $\beta_n(s)$ is

$$\begin{aligned}\beta_n(s) &\triangleq \frac{f\left(\underline{r}_n^N \,\middle|\, s_n = s, \underline{r}_0^{n-1}\right)}{f\left(\underline{r}_n^N \,\middle|\, \underline{r}_0^{n-1}\right)} \\ &= \sum_{b\in\mathcal{M}} \frac{f\left(\underline{r}_n^N \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right) \Pr\left\{b_n = b \,\middle|\, s_n = s, \underline{r}_0^{n-1}\right\}}{f\left(\underline{r}_n^N \,\middle|\, \underline{r}_0^{n-1}\right)} \\ &= \sum_{b\in\mathcal{M}} \Pr\{b_n = b\} \frac{f\left(r_n \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right)}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} \frac{f\left(\underline{r}_{n+1}^N \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n}\right)}{f\left(\underline{r}_{n+1}^N \,\middle|\, \underline{r}_0^{n}\right)} .\end{aligned} \tag{3.42}$$

Using Eqs. (3.30), (3.32), (3.34), and (3.36) yields

$$\beta_n(s) = \frac{1}{\Omega_n} \sum_{b\in\mathcal{M}} \Pr\{b_n = b\} \mu_n(\mathrm{SG_Z}[s,b]) \beta_{n+1}(\mathrm{SG_Z}[s,b]) , \tag{3.43}$$

which corresponds to Eq. (A.8) in Appendix A.

Since Eqs. (3.37), (3.40) and (3.43) correspond to Eqs. (A.9), (A.5) and (A.8) in Appendix A, the APP algorithm can be used to calculate $\Pr\left\{b_n = b \,\middle|\, \underline{r}\right\}$. However, because the size of the observation window is limited, the branch metrics that are used differ from the ideal $f\left(r_n \,\middle|\, \underline{b}_{n-Z+1}^n = \underline{\tilde{b}}_{n-Z+1}^n, \underline{r}_0^{n-1}\right)$, and thus the output of the algorithm is only an approximation of the APP's.

In summary, the inner decoder is implemented with the APP algorithm based on a simple FSM model consisting of M^{Z-1} states, where Z is the number of previous samples used to estimate the carrier phase error. The state transition matrix, $\mathrm{ST_Z}[\bullet,\bullet]$, is defined by Eq. (3.21), and the symbol generation matrix, $\mathrm{SG_Z}[\bullet,\bullet]$, is defined by Eq. (3.23). The forward recursion is initialized with $\alpha_1(s) = 1$ if $s = 0$ and $\alpha_1(s) = 0$ otherwise. The inputs to the algorithm are the *a priori* probabilities $P_{n,b}^{(I_u)} = \Pr\{b_n = b\} = 1/M$, and the branch metrics given by Eq. (3.25)[3]. The output of the algorithm is $P_{n,b}^{(O_u)} \cong \Pr\left\{b_n = b \,\middle|\, \underline{r}\right\}$, so the inner decoder is suboptimal. Nonetheless, as the performance evaluation results in the following section indicate, the inaccuracy due to this suboptimality is not a grave concern.

[3] On the second and subsequent iterations, the values from the outer decoder are used for $P_{n,b}^{(I_u)}$, but the same values for the branch metrics are used for all iterations.

3.4 Simulation Results and Related Discussion

The performance of the proposed coherent and noncoherent systems was thoroughly investigated by means of computer simulation. Performance is compared with coherently-detected absolutely-encoded M-PSK and traditional differentially-detected differentially-encoded M-PSK. Various choices for the of interleaver mapping are considered, and bit-by-bit and symbol-by-symbol interleaving are compared. The effects of the interleaver size and the constraint length of the convolutional code are investigated. In addition, a slight carrier frequency offset was added and the performance of the noncoherent system was observed.

Unless otherwise indicated, the following system parameters were used for the simulations. A 16-state, rate 1/2, convolutional code with generator $(23, 35)_8$ is used, so $k_c = 1$, $n_c = 2$, and $L_c = 5$, and code words of length $N = 1024$ symbols are produced by the encoder (i.e. $N_a = 1020$). A non-uniform symbol-by-symbol interleaver is used, where the symbols are stored in a 32×32 matrix and read according to the technique described in [40, with $P(\xi) = \{5, 11, 7, 19, 23, 29, 13\}$]. The interleaved symbols are differentially encoded and transmitted using QPSK ($M = 4$). Gray mapping is used and implemented prior to differential encoding. Details regarding the simulations can be found in Appendix D.

3.4.1 Iterative Decoding Example

Before presenting detailed performance evaluation results, a simple example is given which helps illustrate the operation of the decoder, and highlights the differences with other, more traditional, non-iterative techniques. Consider a system in which a single message word of $N_a = 96$ bits is encoded with a 16-state, rate 1/2 convolutional code, interleaved, and transmitted using differentially encoded QPSK. A total of 101 symbols are transmitted (i.e. $N = 100$) across an AWGN channel with a signal-to-noise ratio (SNR) per bit of 10

dB, and received with coherent detection. The received samples can be plotted in a signal-space diagram, an example of which is shown in Fig. 3.1. When the SNR ($\mathcal{E}_b/\mathcal{N}_0$) is high, as in this case, the samples are typically tightly clustered around the points in the QPSK signal constellation. Different symbols are used in the diagram to indicate the value of the transmitted symbol corresponding to each received sample. Supposing that the values of the transmitted symbols were of interest (and ignoring the differential and convolutional encoding), an optimal decoder could determine them by finding the region in which the corresponding received samples fall. The boundaries of the decision regions are indicated by dashed lines in the figure. For the samples in this example the decoder would make correct decisions for all but one of the transmitted symbols. Samples corresponding to symbols where correct decisions are made are indicated by hollow symbols in the plot, whereas incorrect decisions are indicated by filled symbols.

When the SNR is lower than in this example, the received samples are more spread out, so more errors will occur. This is evident in the signal-space diagram shown in Fig. 3.2, which is for a SNR of 2 dB. Because of the noise, the samples are removed from the points in the QPSK signal constellation, which are indicated by the asterisks in the plot. Furthermore, several samples fall outside of the correct decision regions, so the decoder would make incorrect decisions about these symbols. In this example there are 26 errors, or roughly one quarter of the symbols would be decoded incorrectly.

The decoder described above is appropriate for absolutely-encoded QPSK, but now suppose that the values of the data symbols, as opposed to the transmitted symbols, are of interest. Based on the received samples, the inner decoder computes the APP's of each data symbol, using the algorithm described in Section 3.3.1. Ideally, these APP's, $\Pr\left\{b_n = b \,\middle|\, \underline{r}\right\}$, will be equal to one for the correct value of b and zero otherwise. In practice this does not occur because of the noise, but one can hope that the correct value at least has the largest

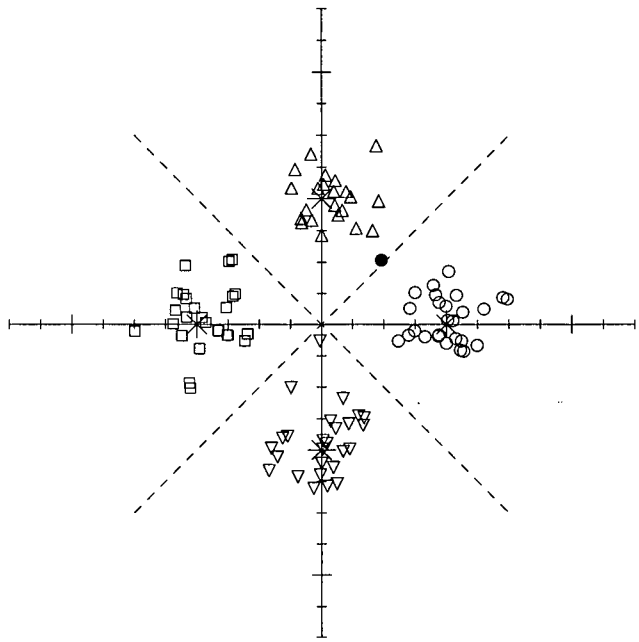


Figure 3.1: Signal-space diagram (SNR = 10 dB): showing 101 received samples, with coherent detection of QPSK. Errors are shown by filled symbols (only one error occurred).

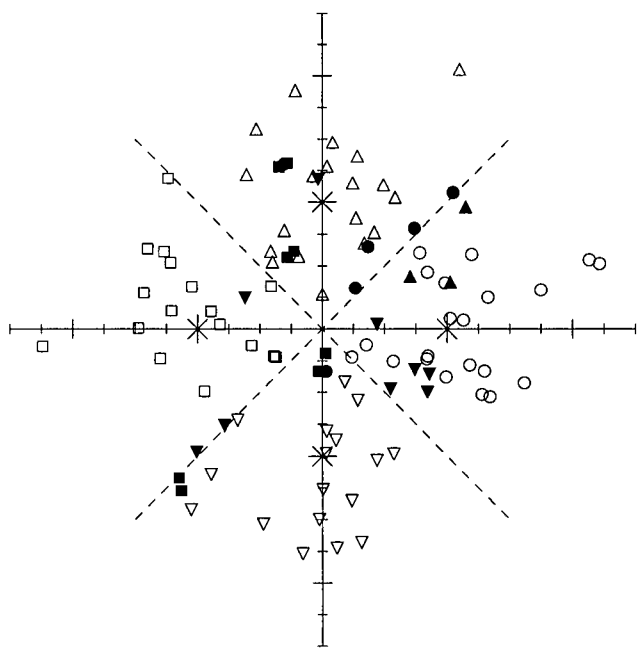


Figure 3.2: Signal-space diagram (SNR = 2 dB): showing 101 received samples, with coherent detection of QPSK. (26 errors).

probability. To determine each data symbol, a hard-decision decoder could find the value with the largest APP. Of greater importance when iterative decoding is used is the confidence of these decisions. For example, if all the APP's for one data symbol are roughly equal, then although one APP may be largest, the decoder cannot have much confidence in its decision. Because M APP's are computed for each data symbol, it is helpful to use the *a posteriori symbol mean*,

$$\mathbf{E}\left[\exp\left\{j\frac{2\pi}{M}\mathrm{GM}\left[b_n\right]\right\} \,\middle|\, \underline{r}\right] \triangleq \sum_{b\in\mathcal{M}} \exp\left\{j\frac{2\pi}{M}\mathrm{GM}\left[b\right]\right\} \Pr\left\{b_n = b \,\middle|\, \underline{r}\right\} , \tag{3.44}$$

to provide a graphical representation of the APP's. For the APP's computed from the received samples shown in Fig. 3.1 and Fig. 3.2, the symbol means are plotted in Fig. 3.3 and Fig. 3.4, respectively. Like the signal-space diagrams reflect the distribution of the received samples, these plots reflect the distribution of the APP's. Ideally, all the symbol means would fall at the points in the QPSK signal constellation, indicated by the asterisks. If hard decisions were to be made based on the APP's, a data symbol error would occur for each symbol mean that falls outside of the correct decision region. The distance between a symbol mean and the nearest point to it in the signal constellation reflects the confidence the decoder has in that decision. As can be seen by Fig. 3.3, when the SNR is high, the decoder is confident in its decisions, and only two decision errors are made in this example. On the other hand, when the SNR is low, Fig. 3.4 indicates that there are 36 data symbol errors. Typically, there are roughly twice as many decision errors for the data symbols as there are for the transmitted symbols. This illustrates a drawback to differential encoding, because any single transmission error can cause errors in two data symbols. Therefore, differential encoding is usually associated with a degradation is system performance, although it is useful for facilitating noncoherent detection.

For the case when the SNR is low, not only are there several errors in the output of the

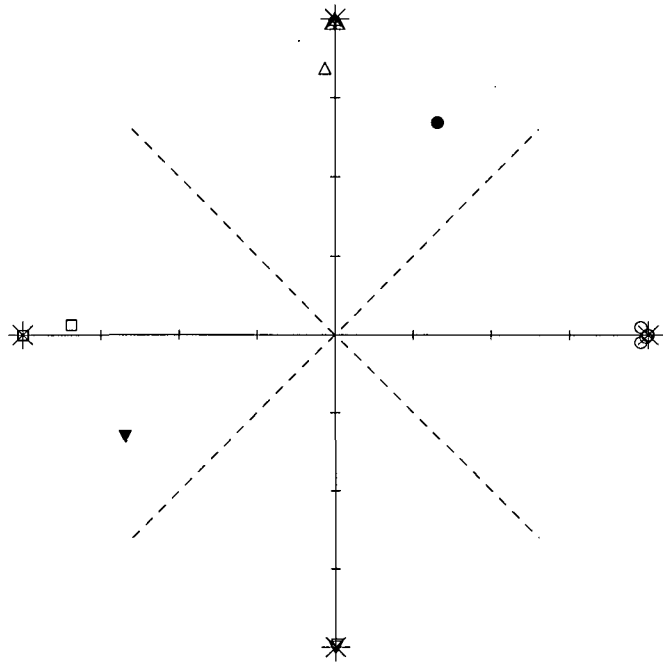


Figure 3.3: Symbol mean diagram (SNR = 10 dB): Symbol means for the data symbols based on the APP's produced by the inner decoder. Errors are shown as filled symbols (2 errors occurred).

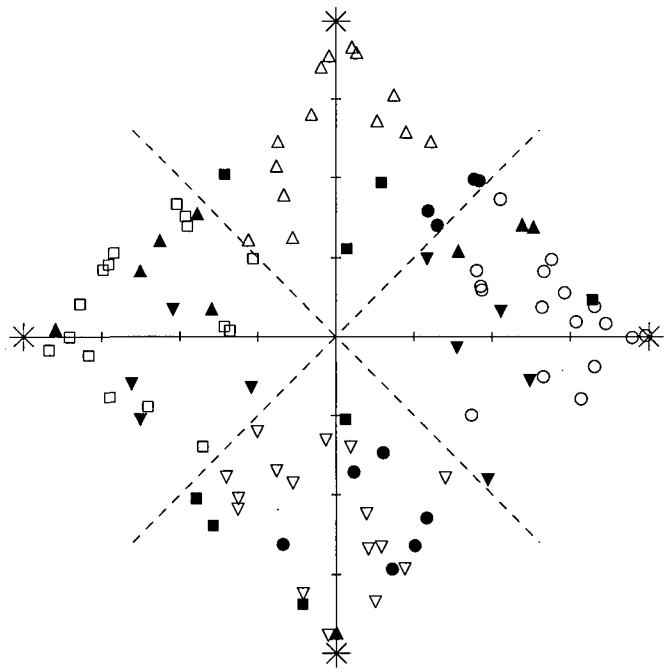


Figure 3.4: Symbol mean diagram (SNR = 2 dB): Symbol means for the data symbols based on the APP's produced by the inner decoder. (36 errors).

inner decoder, but there is also a great deal of uncertainty with regards to the values of the data symbols. Note, however, that the decoder tends to have little confidence in its incorrect decisions, and more confidence in its correct ones. This is important because it implies that the outer decoder will place less emphasis on the incorrect decisions when decoding the convolutional code. While decoding the convolutional code the outer decoder also produces information about the values of the data symbols. Based on this additional information, along with the received samples, the inner decoder can then recalculate the APP's. By using the iterative decoding process, it is possible to improve the reliability of the APP's for the data symbols. For the example above with SNR = 2 dB, this is illustrated in Figs. 3.5 - 3.8, which show the symbol means after the second to fifth iterations, respectively. With each iteration, the symbol means move closer to the points marked by the asterisks, thereby indicating an increase in the confidence the decoder has in its decisions. Furthermore, the number of errors also decreases, so the reliability of the APP's improves. After the fifth iteration there are no more data symbol errors, so the outer decoder is able to correctly decode the convolutional code. In fact, in this example the reliability of the APP's after the third iteration was sufficient for the outer decoder to correctly decode the convolutional code, but could not do so with confidence in its decisions.

In this example, the decoder was eventually able to correctly decode the message word because the reliability of the APP's increased with each iteration. However, at low SNR's such as used in this example, the decoder often fails, and when the decoder fails, it usually fails catastrophically. That is, when it fails, roughly half of the bits are usually in error. It is quite rare for the only a few bits to be in error after five iterations; there are either no errors or a lot of errors. More importantly, the bit error rate is dominated by the catastrophic error events. It is interesting to note that these events occur, not because the decoder is overly confident about its incorrect APP's, but because it is unable to achieve much confidence

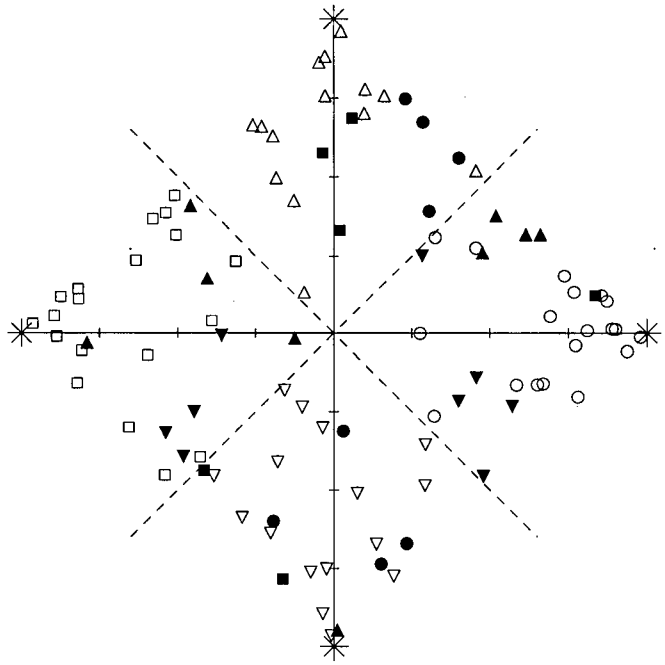


Figure 3.5: Symbol mean diagram after second iteration. (SNR = 2 dB, 32 errors).

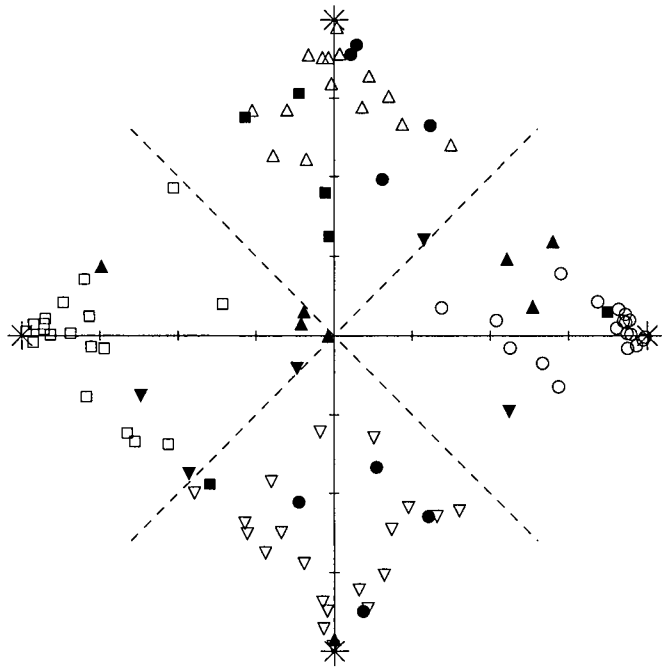


Figure 3.6: Symbol mean diagram after third iteration. (SNR = 2 dB, 27 errors).

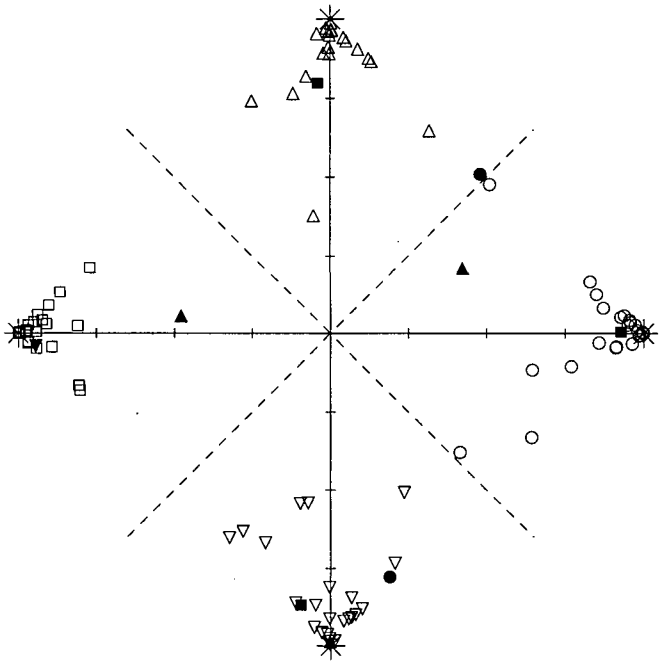


Figure 3.7: Symbol mean diagram after fourth iteration. (SNR = 2 dB, 9 errors).

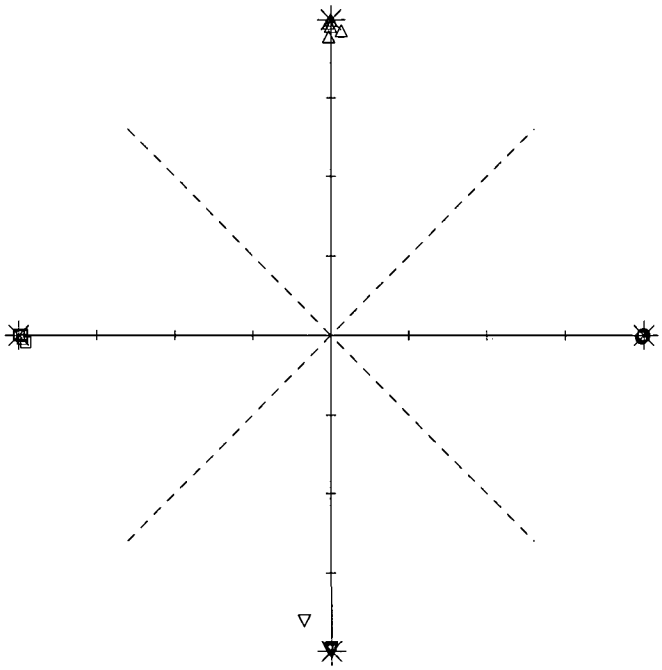


Figure 3.8: Symbol mean diagram after fifth iteration. (SNR = 2 dB, no errors).

at all. In Fig. 3.9 the symbol means are shown for a different randomly-simulated block of received samples, one for which the decoder fails. This is for the output of the inner decoder after the fifth iteration. Unlike Fig. 3.8, this figure shows that the inner decoder has been unable to significantly improve the reliability of the APP's. Note that the decoder does not actually converge to some steady-state, but actually thrashes about with each iteration. That is, after the sixth iteration the output of the inner decoder is quite different than after the fifth iteration, but no more reliable. The failure of the iterative decoder is even more evident by observing the APP's of the data symbols produced by the outer decoder after the fifth iteration, as shown in Fig. 3.10. The symbol means are clustered around the origin of the plot, indicating that the APP's produced by the outer decoder for each data

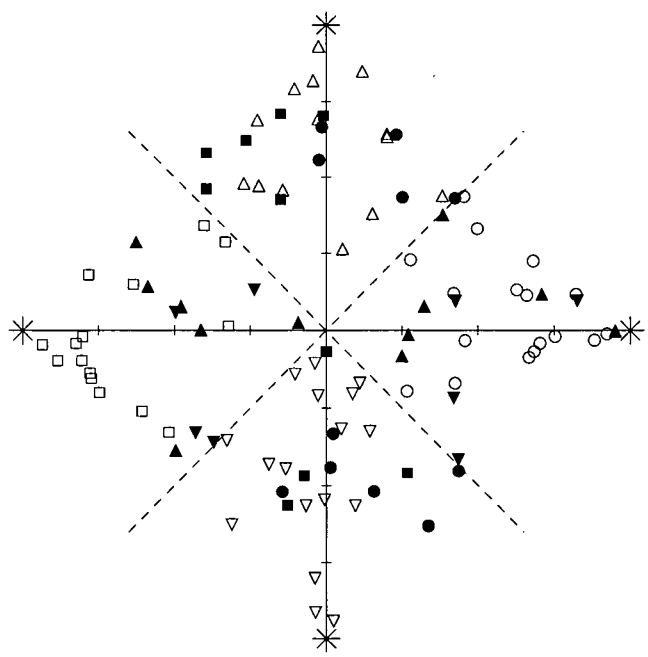


Figure 3.9: Symbol mean diagram (inner decoder): Symbol means for the data symbols based on the APP's produced by the inner decoder after the fifth iteration, for a block when the decoder fails.

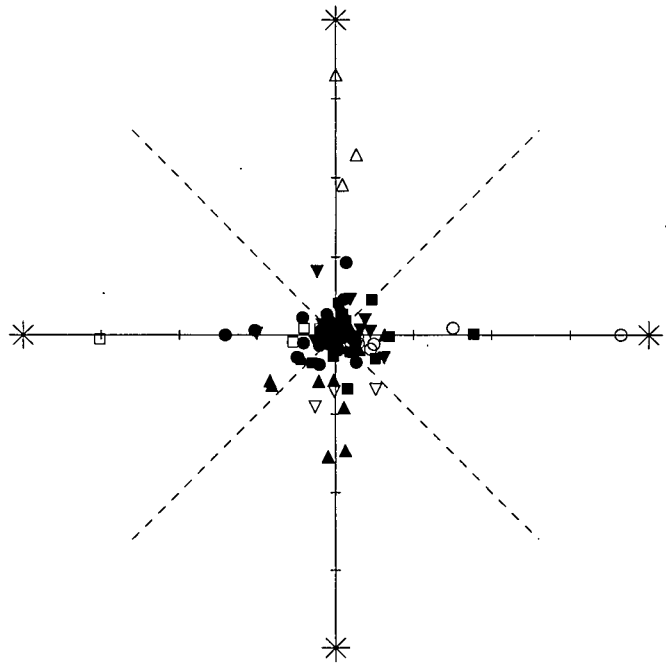


Figure 3.10: Symbol mean diagram (outer decoder): Symbol means for the data symbols based on the APP's produced by the outer decoder after the fifth iteration, for a block when the decoder fails

symbol are all roughly equal[4]. That is, the outer decoder has little confidence that any particular value for each data symbol is more likely than any other value. So, when the iterative decoder fails, it is not because the decoder is overly confident about its incorrect decisions, but rather that it is unable to make any reliable decisions at all. Note that the decoder does not converge to a steady-state situation in these cases, but instead thrashes about Although using more than five iterations may sometimes permit correct decoding, in many cases additional iterations make no difference, and the BER is still dominated by the catastrophic error events.

[4] For the previous example, where the decoder was successful, the symbol means for the APP's produced by the outer decoder were similar to the APP's produced by the inner decoder shown in Fig. 3.8.

3.4.2 Coherent Detection

To show how iterative decoding improves the bit error rate (BER), the system performance as a function of the SNR, $\mathcal{E}_b/\mathcal{N}_0$, is shown in Fig. 3.11, using a code word size of $N = 1024$ symbols. The performance after each of the first twenty iterations is shown, and it improves significantly with the first few iterations although gains appear to be marginal after about the sixth. Clearly iterative decoding provides an effective means for decoding the concatenation of a convolutional code with differentially-encoded QPSK.

As shown, the iterative decoding process is effective for decoding the concatenated code. However, it is also important to compare the performance with other coding schemes. In particular, a comparison with a system using the the same convolutional code but with absolutely-encoded QPSK will determine what degradation, if any, comes from using differential encoding. In the absence of convolutional coding, it is well known that systems

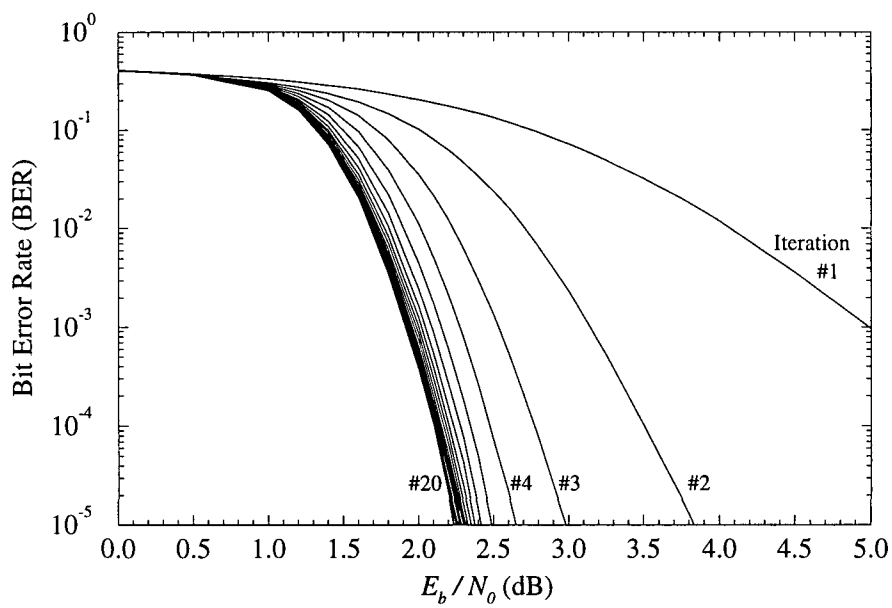


Figure 3.11: Coherent Demodulation: Performance of the concatenated code for the first twenty iterations.

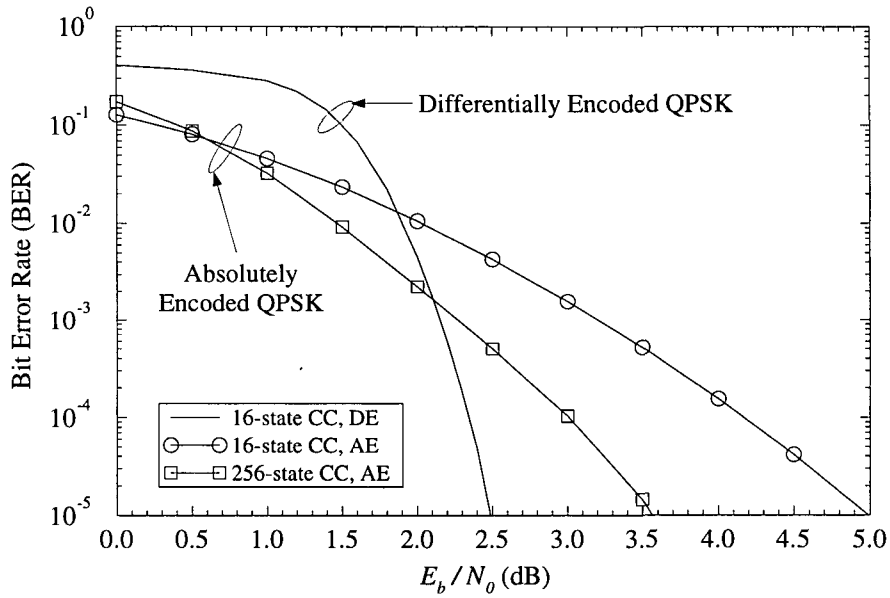


Figure 3.12: Coherent Demodulation (BER): BER performance of the concatenated code after the fifth iteration. Also shown is the performance of absolutely-encoded (AE) coherently-demodulated QPSK, with 16-state and 256-state convolutional codes.

using differentially-encoded QPSK have BER's that are roughly twice that of systems using absolutely-encoded QPSK [24]. The comparison when convolutional coding is used can be made from Fig. 3.12, where the performance of the iterative decoder after five iterations for differentially encoded QPSK is shown along with the performance of an optimal decoder for absolutely encoded QPSK. As can be seen, the differentially-encoded case actually outperforms the absolutely-encoded case, by about 2.5 dB at a BER of 10^{-5}. At first glance this result may seem quite surprising because differential encoding is typically associated with a degradation in system performance. Because the combination of convolutional encoding, interleaving, and differential encoding form a SCCC, this result is actually a very positive reflection on the power of serial concatenation. Even though one of the constituent codes (the differential encoding) has a coding rate of nearly one ($N/[N+1]$), so it introduces negligible redundancy, the concatenated code yields significantly better BER performance

over the stand-alone convolutional code. This is because serial concatenation of two component convolutional codes produces a concatenated code with greatly increased memory over the component codes individually. Of course, because iterative decoding is used with the concatenated code, but is not required (or effective) for the stand-alone convolutional code, there is a substantial difference in decoder complexity. To provide a more balanced comparison, the performance of a more powerful convolutional code, with 256-states and generator $(561, 753)_8$, is also shown in Fig. 3.12 for the absolutely-encoded case. The concatenated code can even outperform this more powerful code, by about 1.1 dB at a BER 10^{-5}.

For some communication systems the word error rate (WER) is a more important measure of system performance than the bit error rate. For example, in systems employing automatic repeat request (ARQ), words which are received with detected errors are retransmitted. If the word error rate is high the number of required retransmissions will also be large, reducing overall system throughput. As shown in Fig. 3.13, the concatenated code is effective in terms of the WER, so it is well suited for use with ARQ schemes. This fact is also highlighted by Fig. 3.14, which plots the BER divided by the WER as a function of the SNR. This indicates the average fraction of bits in a word that are in error when a word error occurs. As can be seen, when a word error occurs, typically a large number of bits are in error for the concatenated code, much more so that for the convolutional codes. This shows that when the decoder fails, it tends to fail drastically. As a result, when a large number of words are transmitted, most of the bit errors occur together in only a small number of words. This is advantageous for ARQ schemes because only a few retransmissions are required. Clearly, the concatenated code is surprising effective in terms of both the BER and the WER. Although more powerful codes, such as turbo codes, do exist, this code has the advantage that it is appropriate for use with noncoherent detection because of the differential encoding.

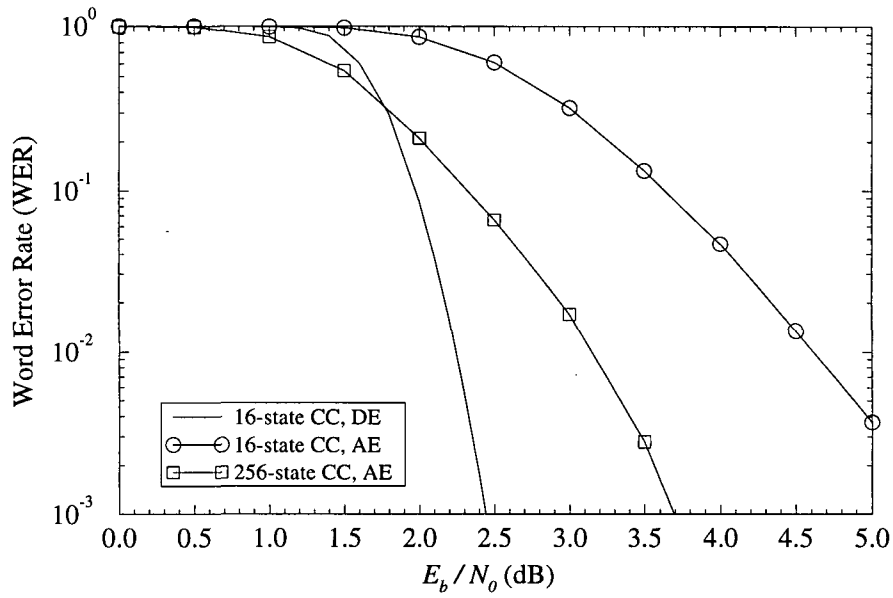


Figure 3.13: Coherent Demodulation (WER): WER performance of the concatenated code after the fifth iteration. Also shown is the performance of absolutely-encoded coherently-demodulated QPSK, with 16-state and 256-state convolutional codes.

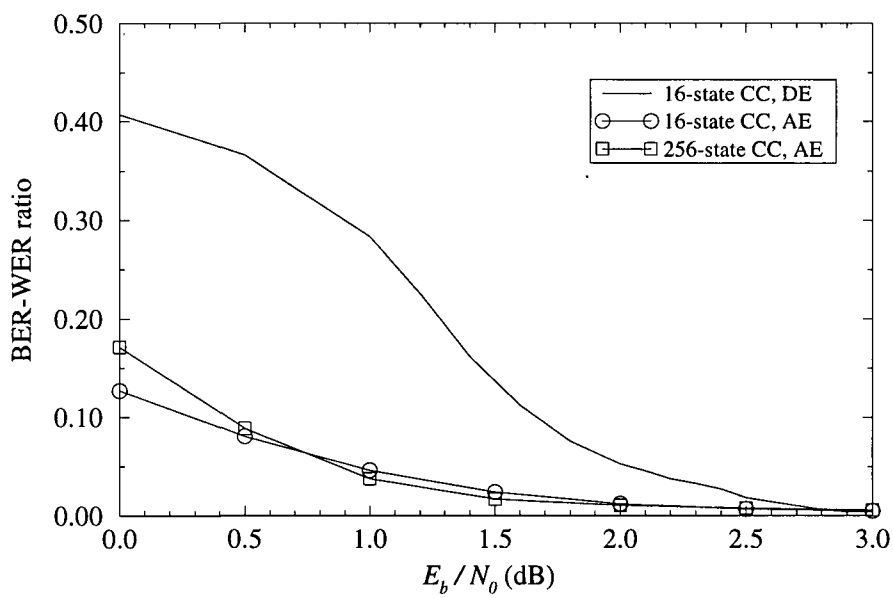


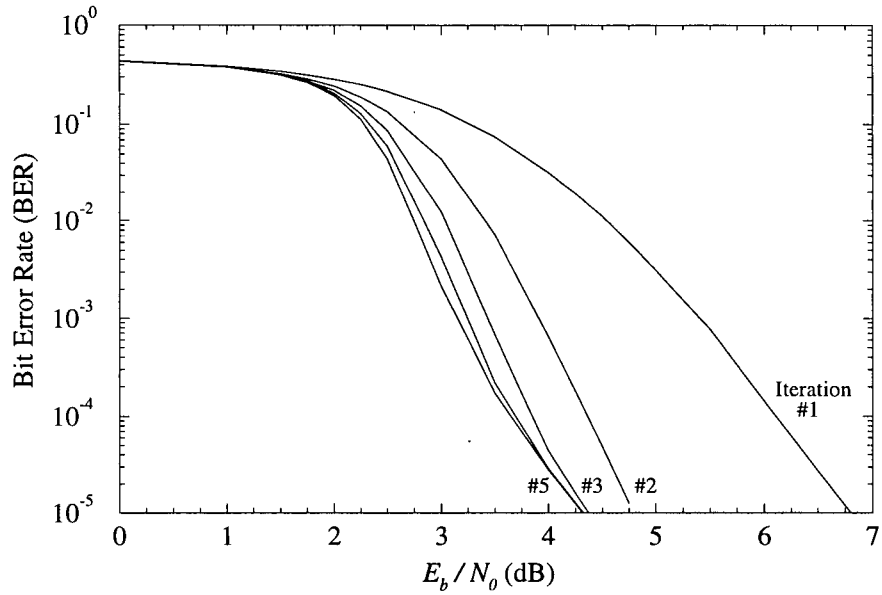
Figure 3.14: Coherent Demodulation (BER/WER ratio): The average fraction of bits in a word that are in error when a word error occurs. After five iterations.

3.4.3 Noncoherent Demodulation

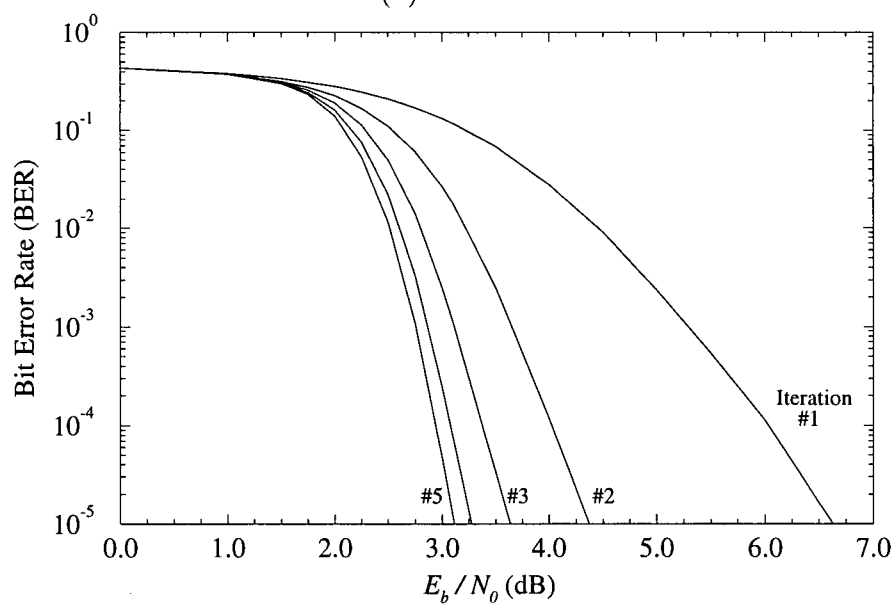
The same surprising performance found with coherent demodulation also extends to the noncoherent case when iterative decoding is used with the inner decoder described in Section 3.3.2. For the first five iterations, the performance of the $Z = 2$ (4-state) noncoherent decoder is shown in Fig. 3.15a. Although the performance improves with each iteration, it appears that there is little benefit from exceeding four or five iterations. Similar results hold for the $Z = 3$ (16-state) decoder, as shown in Fig. 3.15b. As shown in Fig. 3.16, for the $Z = 2$, 3, and 4 decoders after five iterations, increasing Z leads to performance gains as expected, and with $Z = 4$ (64 decoder states) there is only a 0.5 dB noncoherence penalty compared with coherent demodulation after the same number of iterations. However, the benefits of using this system are obvious if the results are compared to the performance of a traditional differential detector followed by a deinterleaver and a Viterbi decoder. When there is only a single differential detector (i.e. $Z = 1$) the inner decoder has only a single state and therefore cannot exploit the information fed back from the outer decoder. So, to provide a more fair comparison in terms of decoder complexity, the memory length of the convolutional code has been increased. It is clear that traditional differential detection, even with a powerful 256-state convolutional code, is not at all competitive, with performance inferior by more that 2 dB at a BER of 10^{-3}.

3.4.4 Effect of Interleaver Mapping

As mentioned in Section 2.1, I chose to consider three different types of interleaver mapping: *i*) the block interleaver, *ii*) the non-uniform interleaver, and *iii*) the random interleaver. The results presented above are all for the non-uniform interleaver. To investigate the effect of the interleaver on system performance, the coherent system was tested with 1000 different, randomly selected interleavers. Because of the excessive simulation time required



(a) $Z = 2$



(b) $Z = 3$

Figure 3.15: Noncoherent (NC) Demodulation: BER performance after each of the first five iterations. For (a) $Z = 2$, (b) $Z = 3$.

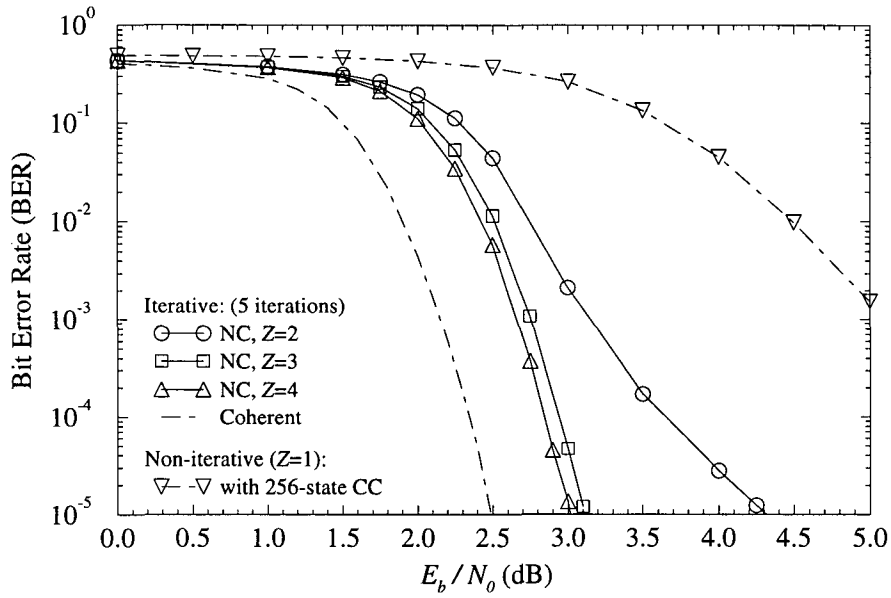


Figure 3.16: Noncoherent (NC) Demodulation: BER performance after five iterations with $Z = 2$, 3, and 4 differential detectors. Also shown is the performance with coherent demodulation after five iterations, and a 256-state convolutional code (CC) with a single differential detector followed by non-iterative Viterbi decoder.

to test so many interleavers, simulations were performed at only two SNR's, 2.15 dB and 2.35 dB. These values were chosen because they occur during the steep section of the performance curve, to better distinguish between effective and ineffective mappings. Fig. 3.17 shows the performance of the coherent QPSK system with the different interleavers, and a block size of $N = 1024$, after the fifth iteration. To generate this graph $N_w = 3000$ words (i.e. approximately 3 million bits) were transmitted for each interleaver. From this graph it appears that there is a large difference between the different interleavers, with the best interleaver giving a BER that is an order of magnitude better than the worst at a SNR of 2.35 dB. However, much of the difference can be attributed to statistical uncertainty because of the small number of transmitted words and the fact that bit errors do not occur independently. Increasing the number of transmitted words in the simulation would reduce

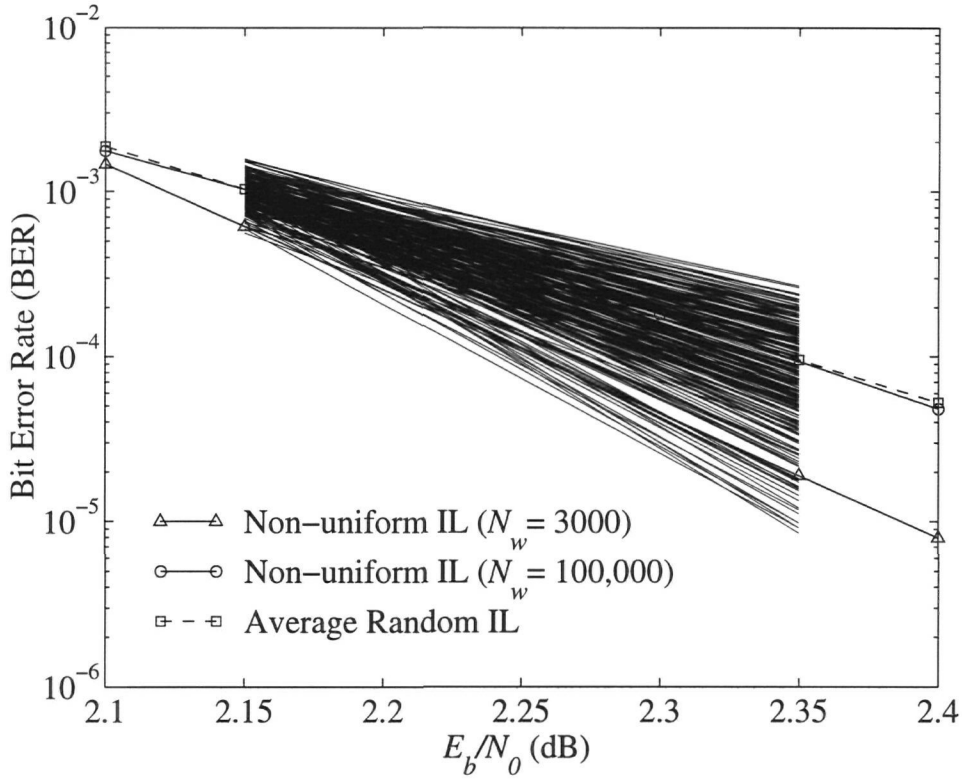


Figure 3.17: Random Interleavers: $N = 1024$, coherent QPSK, $N_p = 5$, $N_w = 3000$.

the spread between the different interleavers. Also shown in this graph is the performance of the non-uniform interleaver. As can be seen, the performance of the random interleavers is consistent with the non-uniform interleaver. To highlight the statistical uncertainty, the BER for the non-uniform interleaver is also shown for a simulation run of $N_w = 100,000$ transmitted words. The performance of the non-uniform interleaver when this many words are tested almost exactly matches the average performance of all the tested random interleavers. In addition, because of the steep drop in the BER curves at these SNR's, the difference between the random interleavers is negligible in terms of the SNR required to achieve a specific BER. If any of the interleavers were significantly better or worse than the others, their performance would have to fall outside of the range of this graph.

In Fig. 3.18 the performances of four random interleavers[5] are shown when $N_w = 10{,}000$ words are transmitted in the simulation. Also shown is the performance of the non-uniform interleaver. As can be seen, there is very little difference between the curves. The performances of a 32×32 block interleaver and the identity interleaver (i.e. without any interleaving) are also given in the figure. Without any interleaving, or with block interleaving, the concatenated code is not nearly as effective as when random or non-uniform interleaving is used[6]. Provided the permutation imposed by the interleaver is sufficiently diverse, the choice of interleaver mapping is not a major factor governing system performance. Although none of these interleavers are specifically selected to optimally minimize the BER, it seems

[5] These are the best and worse performers at the two SNR points used in Fig. 3.17.

[6] Of course, since both the block and identity interleavers are possible choices for the random interleaver, this statement is not entirely accurate. However, except for only a very few possible poor random selections, it appears that most of the $N!$ possible interleaver mappings yield very similar performance.

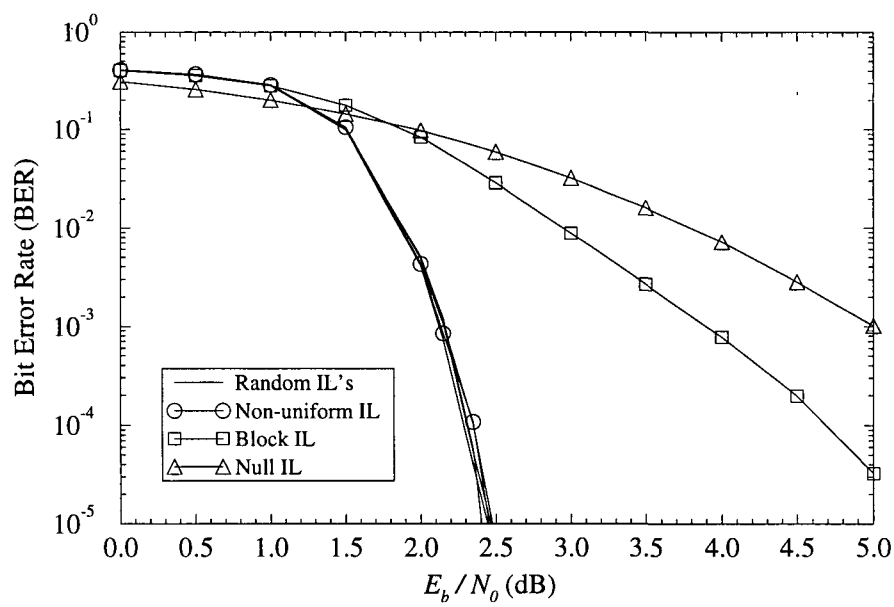


Figure 3.18: Random Interleavers (Coherent): Performance of four randomly selected interleaver mappings, along with the non-uniform interleaver, the 32×32 block interleaver, and the identity interleaver. $N = 1024$, coherent QPSK, $N_w = 10,000$.

unlikely that a carefully designed interleaver, or even an optimal one, will yield performance that is significantly different from these.

To a lesser extent, this observation is also valid when noncoherent detection is used, as shown by the results in Fig. 3.19 for $Z = 2$ and Fig. 3.20 for $Z = 3$. However, for the $Z = 2$ decoder the non-uniform interleaver appears to be slightly better than the random ones. It is not clear whether or not the difference is due to statistical uncertainty, but, from a practical point of view, it translates into only a 0.2 dB difference in SNR.

3.4.5 Bit vs. Symbol Interleaving

Aside from the choice of interleaver mapping, there is also the option of using bit-by-bit or symbol-by-symbol interleaving. When bit interleaving is used, the inner decoder must calculate the marginal APP's of each bit in the data symbols. In doing so, information regarding the correlation between different bits in the same symbol is discarded, theoretically leading to worse performance than symbol interleaving. To compare the differences, four bit interleavers of 2048 bits were randomly selected, and compared with the four random symbol interleavers of 1024 symbols used above[7]. Random interleaving was chosen for this simulation because the non-uniform interleaver used in this thesis is only defined for sizes that are even powers of two. Simulation results shown in Fig. 3.21 for coherent detection, and Fig. 3.22 and Fig. 3.23 for noncoherent detection with $Z = 2$ and $Z = 3$, respectively, confirm that symbol interleaving has a slight advantage over bit interleaving, but the difference is a modest 0.1 dB, except for the noncoherent $Z = 2$ case, where the difference is more pronounced.

[7] The message word length is the same for all cases (i.e. $N_a = 1020$).

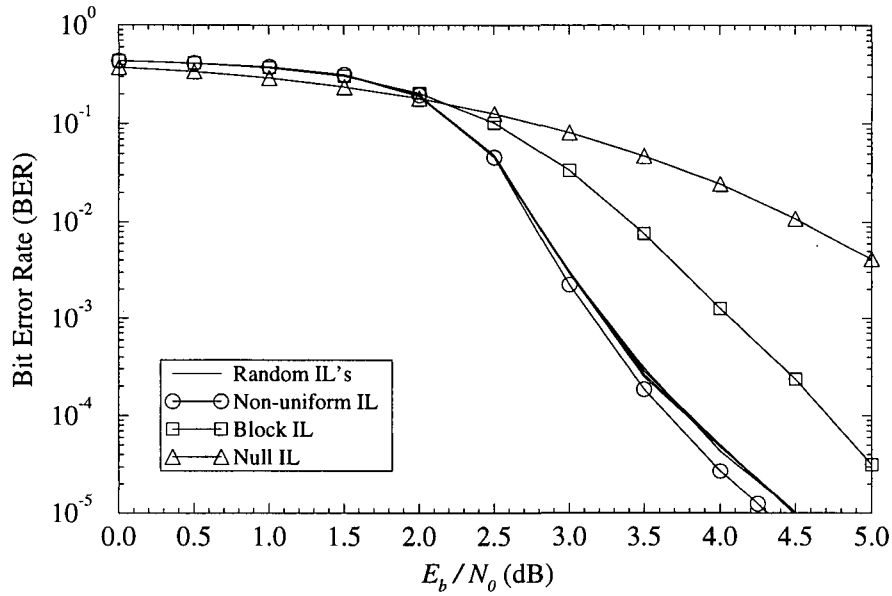


Figure 3.19: Random Interleavers (NC, $Z = 2$): Performance of four randomly selected interleaver mappings, along with the non-uniform interleaver, the 32×32 block interleaver, and the identity interleaver. $N = 1024$, noncoherent QPSK, $Z = 2$, $N_w = 10,000$.

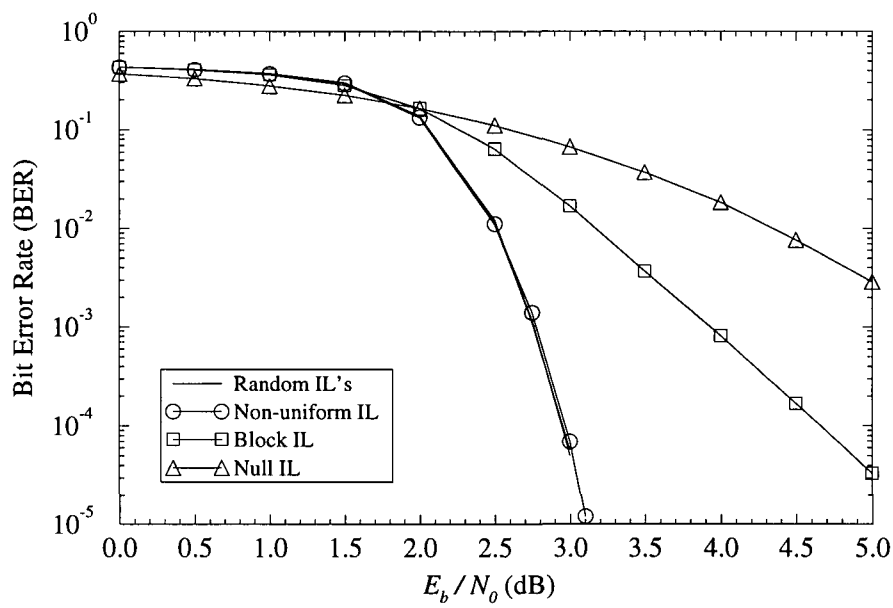


Figure 3.20: Random Interleavers (NC, $Z = 3$): Performance of various interleavers.

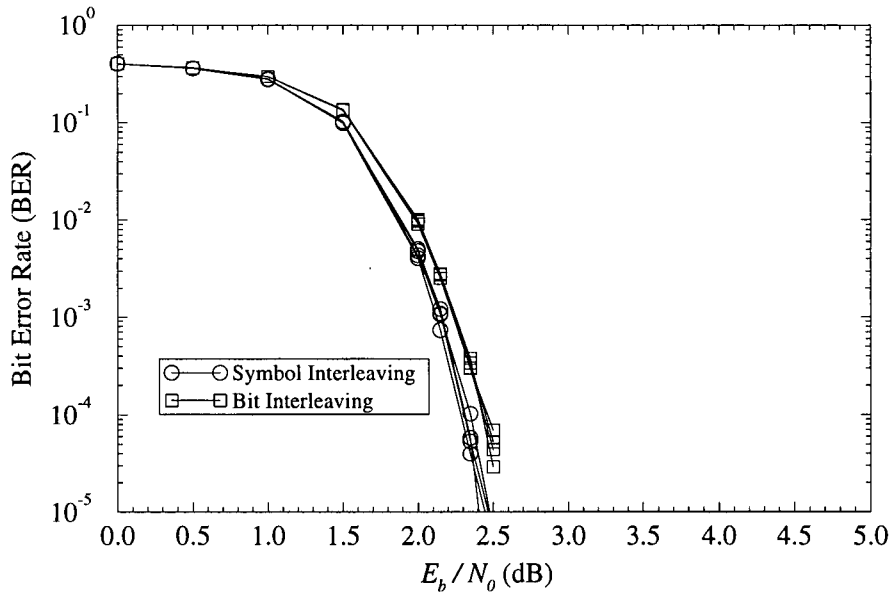


Figure 3.21: Bit vs. Symbol Interleaving (Coherent): Comparison of the performance of four randomly selected bit interleavers and four randomly selected symbol interleavers. $N = 1024$, coherent QPSK, $N_w = 10,000$.

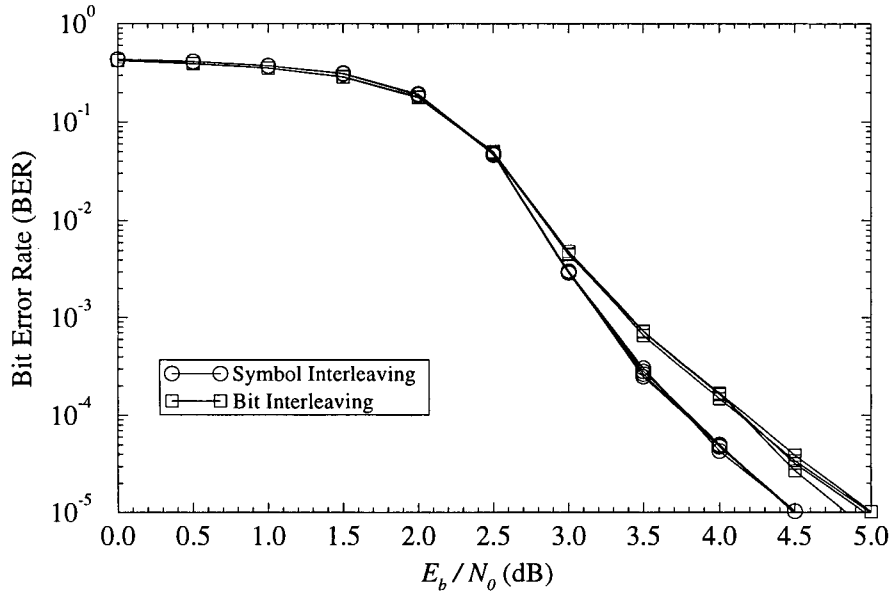


Figure 3.22: Bit vs. Symbol Interleaving (NC, $Z = 2$): Comparison of the performance of four randomly selected bit interleavers and four randomly selected symbol interleavers. $N = 1024$, noncoherent QPSK, $Z = 2$, $N_w = 10,000$.

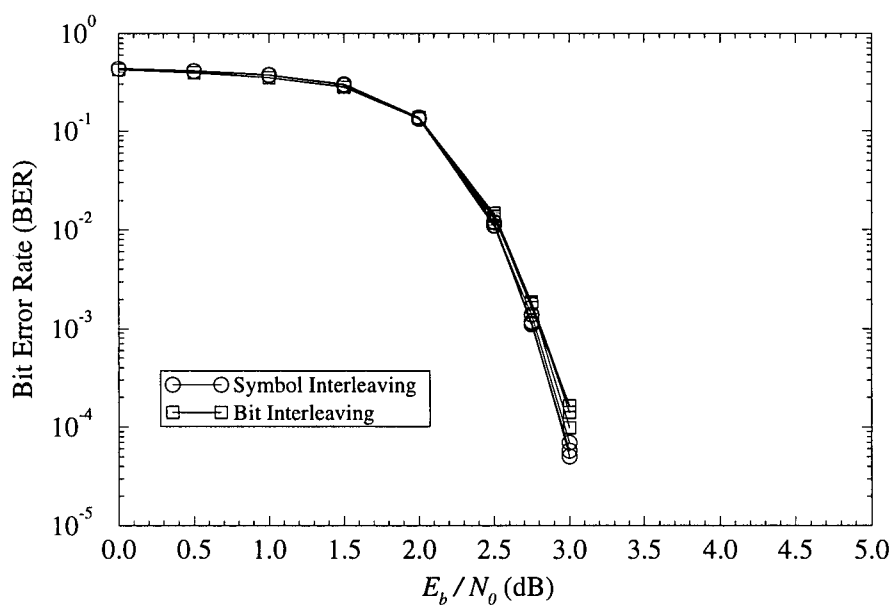


Figure 3.23: Bit vs. Symbol Interleaving (NC, $Z = 3$): Comparison of the performance of four randomly selected bit interleavers and four randomly selected symbol interleavers. $N = 1024$, noncoherent QPSK, $Z = 3$, $N_w = 10,000$.

3.4.6 Interleaver Size

Much more important than the interleaver mapping in governing system performance is the interleaver size, N. When the interleaver size is large, the encoder is better able to spread the redundancy introduced by the convolutional code, so errors that occur consecutively when decoding the differential encoding will be well separated when decoding the convolutional code. The advantage of larger (random) interleavers for coherent detection is shown in Fig. 3.24, with a 2 dB difference in performance between $N = 192$ and $N = 4096$. However, the advantages of larger interleavers is much less pronounced than for other interleaved codes, such as turbo-codes and SCCC's. This is because the inner code (the differential encoder) is by itself a fairly ineffective code with a short memory length, so there is less to be gained by a large interleaver than when the inner code is stronger as with regular SCCC's.

When noncoherent detection is used, there is still some advantage to using larger interleavers, as shown in Fig. 3.25 for $Z = 3$. However, the noncoherent decoder is not quite as effective at exploiting the larger interleavers as the coherent decoder. This is emphasized by the noncoherence penalty, which also increases with increased interleaver length, as shown in Table 3.1 (for a BER or 10^{-5}).

Interleaver Size	Noncoherence Penalty (dB)
192	0.4
512	0.5
1024	0.6
2048	0.8
4096	0.8

Table 3.1: Noncoherence penalty at BER $= 10^{-5}$ as a function of the interleaver size. Noncoherent QPSK, $Z = 3$.

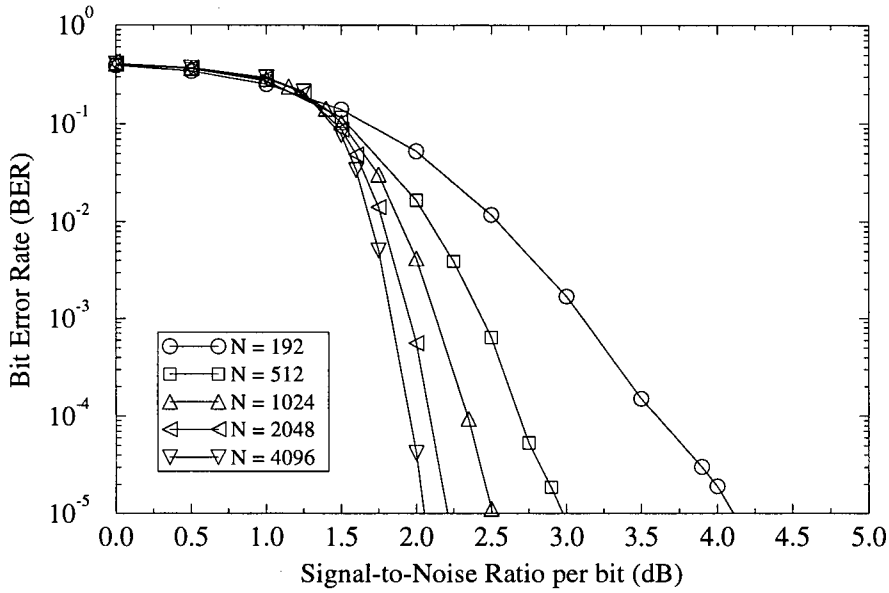


Figure 3.24: Effect of Word Length (Coherent): Comparison of the performance with different word lengths (N). Coherent QPSK, $N_w = 10,000$.

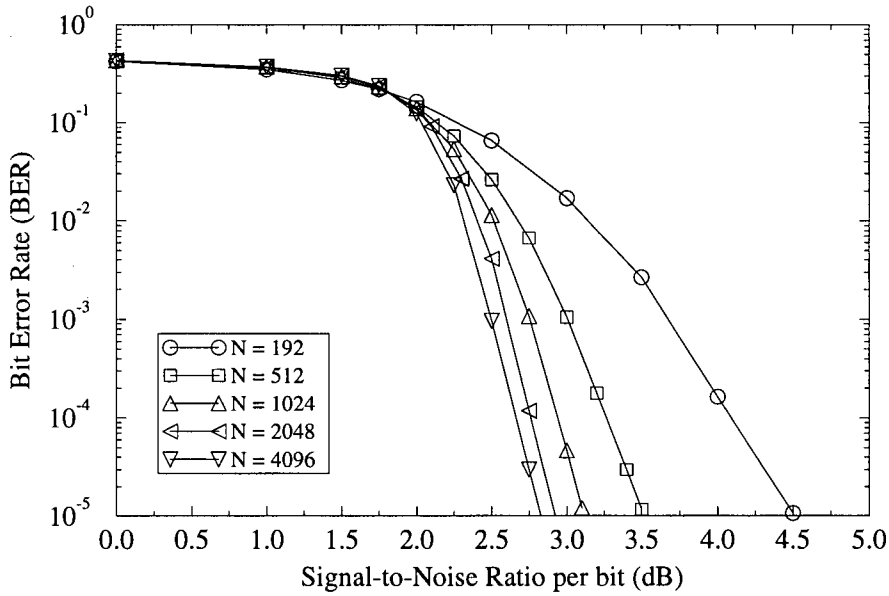


Figure 3.25: Effect of Word Length (NC, $Z = 3$): Comparison of the performance with different word lengths (N). Noncoherent QPSK, $Z = 3$, $N_w = 10,000$.

3.4.7 Constraint Length

For traditional convolutional codes, the constraint length is an important system parameter, with longer length codes giving better performance than shorter ones. In the concatenated code, however, the interleaver size is a much more important parameter, with the constraint length playing a secondary role. The performances of concatenated codes based on convolutional codes[8] with constraint lengths of $L_c = 3$, 4, 5, 6, 7, and 8 are shown in Fig. 3.26 after five iterations with the coherent detection. For BER's above 10^{-5} performance actually *deteriorates* with increasing constraint length when coherent demodulation is used. A conclusive explanation for this result is elusive, but is most likely due to an artifact of the sub-optimal iterative decoding algorithm, and not a property of the concatenated codes. Similar results hold with noncoherent demodulation, as shown in Fig. 3.27 for $Z = 3$, although the decoder performs poorly if L_c is too small (e.g. $L_c < 5$).

3.4.8 Carrier Frequency Offset

It is expected that the noncoherent receiver should be fairly insensitive to modest variations in the carrier phase error. To test this hypothesis, a slight carrier frequency offset (Δf_c) was added, making the received samples (see Eq. (3.3))

$$r_n = v_n \exp\left\{j\left[2\pi(\Delta f_c)Tn + \phi_c\right]\right\} + w_n \ , \tag{3.45}$$

where T is the symbol duration. For $Z = 2$, the results presented in Fig. 3.28 show that for frequency offsets up to $(\Delta f_c)T = 0.01$ there is very little performance degradation. The receiver is unable to compensate effectively for higher offsets, however, and the beginning of an error floor is visible in the curves for $(\Delta f_c)T = 0.02$ and 0.03. Similar results hold for $Z = 3$, as shown in Fig. 3.29.

[8] These codes have been taken from [24, Table 8-2-1], and are the largest-free-distance rate-1/2 codes for each constraint length.

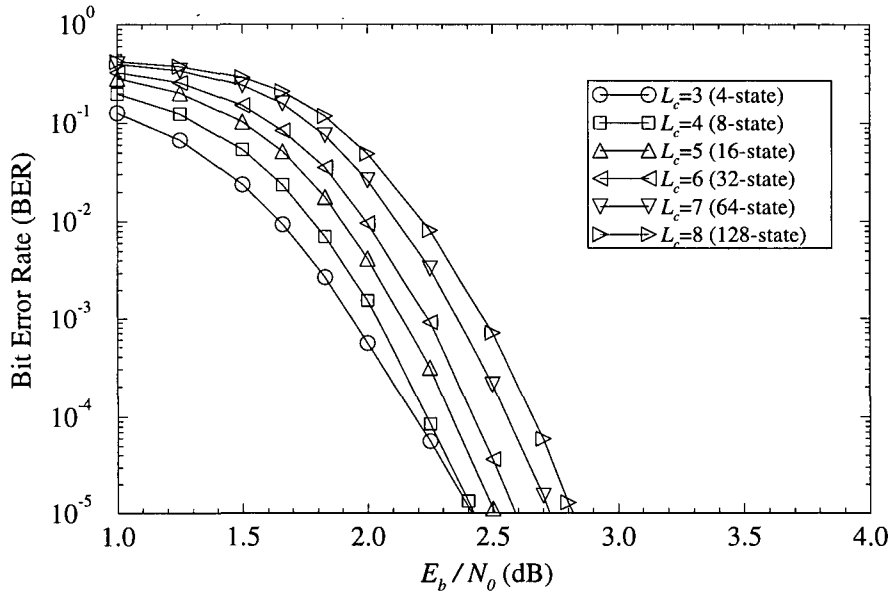


Figure 3.26: Constraint Length (Coherent): Performance of concatenated codes based on convolutional codes with various constraint lengths (L_c). Coherent QPSK, $N = 1024$, $N_p = 5$.

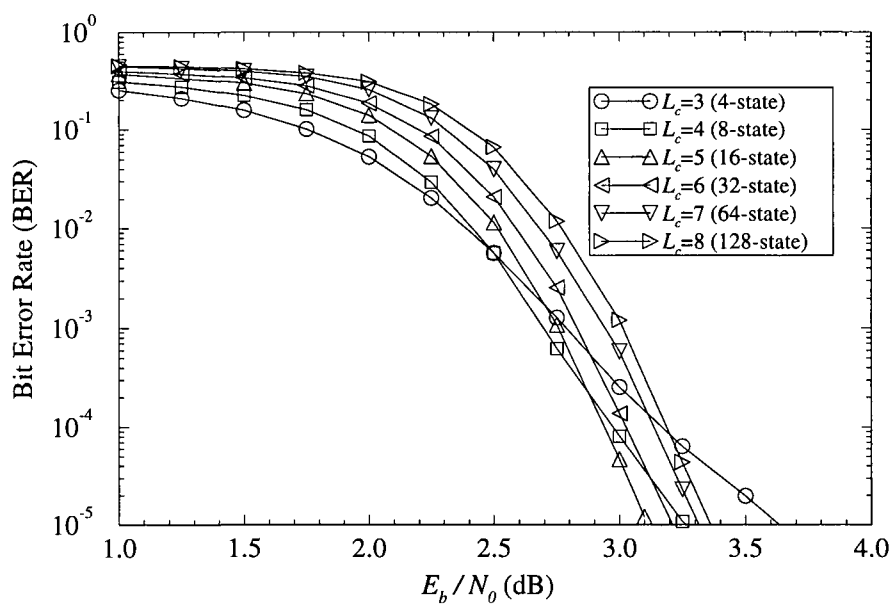


Figure 3.27: Constraint Length (NC, $Z = 3$): Performance of concatenated codes based on convolutional codes with various constraint lengths (L_c). Noncoherent QPSK, $Z = 3$, $N = 1024$, $N_p = 5$.

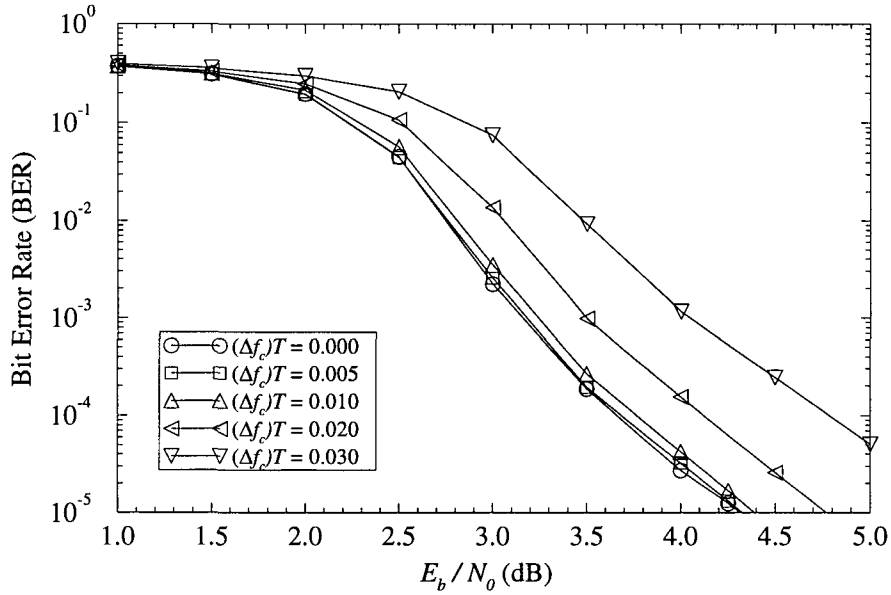


Figure 3.28: Frequency Offset (NC, $Z = 2$): Performance of noncoherent demodulation with various frequency offsets ($\Delta f_c T$).

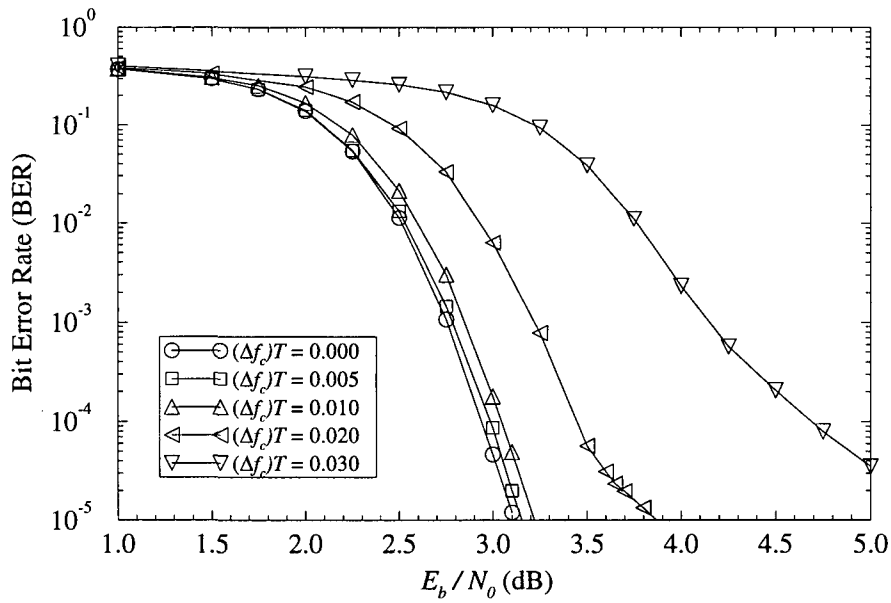


Figure 3.29: Frequency Offset (NC, $Z = 3$): Performance of noncoherent demodulation with various frequency offsets ($\Delta f_c T$).

3.4.9 Trellis Coded Modulation

Although the discussion to this point has focused on QPSK ($M = 4$), the same general results extend to higher-order modulation schemes, such as 8-PSK ($M = 8$). Both the coherent and noncoherent systems (with $Z = 2$ and $Z - 3$) were tested with the 16-state, rate-2/3, trellis-coded modulation (TCM) code taken from [41, Fig. 9], so $k_c = 2$, $n_c = 3$, and $L_c = 3$. Code words of length $N = 1024$ symbols were used with the same symbol-by-symbol interleaver described above, and natural mapping was used[9]. Note that the inner decoder for noncoherent detection uses 8 decoder states for $Z = 2$, and 64 decoder states for $Z = 3$. As can be seen in Fig. 3.30, performance with $M = 8$ shares the same characteristics as the $M = 4$ system, although in a less pronounced manner. Iterative decoding still works well, but the advantages over regular TCM with absolutely-encoded, coherently-demodulated, 8-PSK are not as large as with QPSK.

3.5 Conclusions

In this chapter two novel soft-output decoders, one for use with coherent detection and the other for noncoherent detection, of differentially encoded M-PSK signals transmitted over the AWGN channel have been presented. Their use within an iterative decoding structure has been proposed, and their performance evaluated by means of computer simulation. The most significant finding is that, for the AWGN channel, differentially encoded M-PSK yields better performance than absolutely encoded M-PSK, when both are used with a convolutional code and interleaving. This is contrary to the commonly-held belief that the use of differential encoding is not beneficial to BER performance in coherent systems.

It is also shown that the proposed noncoherent system is able to closely match the excel-

[9] Natural mapping is appropriate for use with TCM-type codes to properly reflect the set partitioning for which the code is designed.

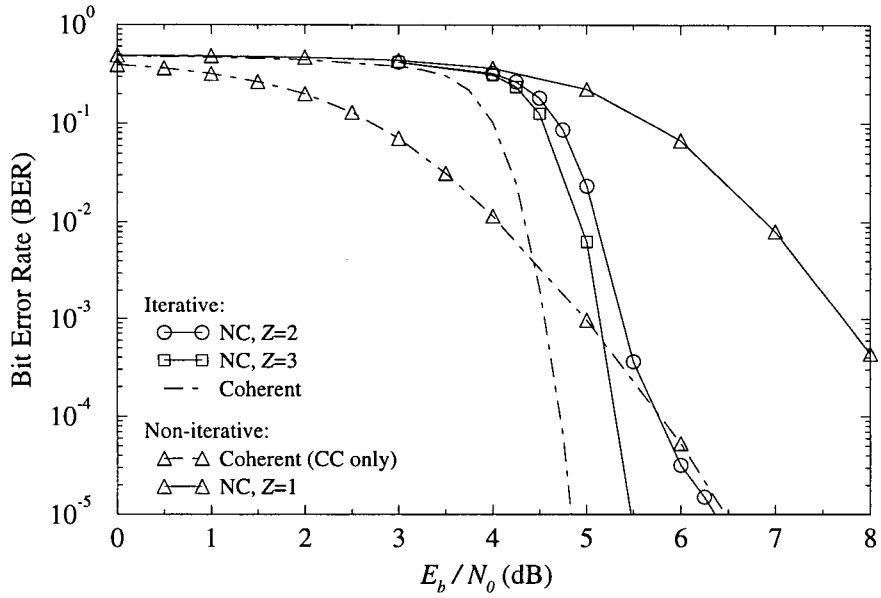


Figure 3.30: 8-PSK: Performance after five iterations of coherent and noncoherent demodulation when 8-PSK is used with a rate 2/3 TCM code.

lent performance of the coherent system, with only a slight noncoherence penalty provided that the number of differential detectors (Z) is sufficiently large. The choice of interleaver mapping is not a prime factor affecting system performance, provided a handful of poor choices (such as block and identity interleaving) are avoided. A large number of randomly-selected mappings where shown to yield very similar performance. There appears to be an advantage, albeit minor, to using symbol-by-symbol interleaving instead of bit-by-bit interleaving. As with other systems using iterative decoding, the size of the interleaver is one of the most important parameters, although its effect is less pronounced than with turbo-codes and SCCC's. The constraint length of the constituent convolutional code is a much less important parameter, and, surprisingly, performance actually deteriorates with increased constraint length. The proposed noncoherent system is very insensitive to slight frequency offsets (up to about 1% of the symbol rate).

Chapter 4

Rayleigh Frequency-Flat Fading

THE second application of the proposed coding/modulation scheme and iterative decoder, for the Rayleigh frequency-flat fading channel, is discussed in this chapter. This chapter has the same structure as Chapter 3, including a description of the continuous-time channel model in Section 4.1; a description of the corresponding discrete-time channel model in Section 4.2; derivations of the inner decoders, for coherent and noncoherent detection, in Section 4.3; and performance evaluation results in Section 4.4. Because of the similarities between the discrete-time channel models for the frequency-flat fading channel and the AWGN channel, the inner decoders presented here are very similar to the ones presented in Chapter 3. In particular, the APP algorithm is used, and, for noncoherent detection, the MDD concept is also used. The only difference between the decoders for the two channel models is in the branch metrics that are used by the APP algorithm.

4.1 Continuous-Time Channel Model

With the Rayleigh frequency-flat fading channel the received signal is modelled as

$$r_c(t) = \mathbf{Re}\left\{v(t)h(t)\sqrt{2}\exp\left\{j2\pi f_c t\right\}\right\} + w_c(t) \ , \tag{4.1}$$

where $v(t)$ is the transmitted lowpass signal, $w_c(t)$ is the background noise, with the same model as for the AWGN channel, and $h(t)$ is the continuous-time fading process. The

fading process is modelled as a zero-mean, wide-sense stationary, complex Gaussian random process. As such, at any time, t, its amplitude, $|h(t)|$, has a Rayleigh distribution and its phase is uniformly distributed over $[0, 2\pi)$. The phase of the fading process corresponds to the carrier phase error in the AWGN channel model. However, the fading process is time variant, making it much more difficult to estimate. Furthermore, because the amplitude of the fading process can be quite small for extended periods of time, the analogue circuitry used in coherent systems has a hard time tracking the phase.

The fading process is correlated over time, with an autocorrelation function of

$$\phi_F(\Delta t) \triangleq \frac{1}{2}\mathbf{E}\left[h^*(t-\Delta t)h(t)\right] \ . \tag{4.2}$$

One commonly used model for the correlation is the land-mobile fading model (Jakes' model) [35], with an autocorrelation function of

$$\phi_F(\Delta t) = \frac{1}{2}J_0(2\pi B_d \Delta t) \ , \tag{4.3}$$

where $J_0(\bullet)$ is the order zero Bessel function of the first kind, and B_d is the *Doppler spread* of the channel. The Doppler spread provides a measure of the rate at which the channel is changing. Other models for the autocorrelation function include the first-order Butterworth fading model[1] [42], with an autocorrelation function of

$$\phi_F(\Delta t) = \frac{1}{2}\exp\left\{-2\pi B_d|\Delta t|\right\} \ , \tag{4.4}$$

and the second-order Butterworth fading model [42], with

$$\phi_F(\Delta t) = \frac{1}{2}\exp\left\{-B'_d|\Delta t|\right\}\left[\cos\left(B'_d|\Delta t|\right) + \sin\left(B'_d|\Delta t|\right)\right] \tag{4.5}$$

where $B'_d = \frac{1}{\sqrt{2}}\pi B_d$.

Based on the model for the continuous-time channel, a model for the equivalent discrete-time channel can be derived.

[1] Also known as the exponential fading model.

4.2 Discrete-Time Channel Model

The combined effect of the signal generator, the continuous-time channel, and the detector on the transmitted symbols is encapsulated by a discrete-time channel model. As shown in Appendix B, for the Rayleigh frequency-flat fading channel the received samples can be modelled as

$$r_n = v_n h_n + w_n \ , \tag{4.6}$$

where $\{v_n\}$ are the transmitted symbols, $\{h_n\}$ is the discrete-time fading process, and $\{w_n\}$ is the discrete-time noise process. The transmitted symbols are given by Eq. (2.3) and related to the data symbols through Eq. (2.1). The fading process has a complex Gaussian distribution, with zero mean and an autocorrelation function of

$$\phi_m^{(F)} \triangleq \frac{1}{2}\mathbf{E}\left[h_{n-m}^* h_n\right] = \phi_F(mT) \tag{4.7}$$

where $\phi_F(\Delta T)$ is the autocorrelation function of the continuous-time fading process as described in Section 4.1. The noise process also has a complex Gaussian distribution, with zero mean and an autocorrelation function of

$$\phi_m^{(W)} \triangleq \frac{1}{2}\mathbf{E}\left[w_{n-m}^* w_n\right] = \frac{\mathcal{N}_0}{2}\delta_m \ . \tag{4.8}$$

Note that the noise process has the same properties as for the AWGN channel.

Although this channel model is derived for noncoherent detection, it can also be used with coherent detection. When coherent detection is used it is assumed that the detector perfectly tracks both the phase and amplitude of the fading process, and provides this information to the decoder. In this case the model for use with coherent detection is the same as for noncoherent detection, except that $\{h_n\}$ is perfectly known.

The relationship between the transmitted symbols and the received samples, as given by Eq. (4.6) for either coherent or noncoherent detection, is used to guide the design of the

inner decoder.

4.3 Inner Decoder

In this section the inner decoders for both coherent and noncoherent detection are presented. Because the discrete-time channel model described above is very similar to the model for the AWGN channel, it is not surprising that very similar inner decoders are used for both channel models. In fact, the only difference is in the branch metrics used by the APP algorithm, since it is necessary to consider the time-variant fading process instead of the static carrier phase error.

4.3.1 Coherent Detection

When coherent detection is used, the discrete-time fading process is provided to the inner decoder as side information. That is, $\underline{h} = h_0, h_1, h_2, \ldots, h_N$ is perfectly known. In this case calculating the APP's, $\Pr\left\{b_n = b \,\middle|\, \underline{r}\right\}$ is straightforward. The relationship between the differentially encoded symbols and the received samples is given by

$$r_n = \sqrt{\mathcal{E}_s} \exp\left\{ j\frac{2\pi}{M} d_n \right\} h_n + w_n \ , \tag{4.9}$$

which comes from Eq. (4.6) with Eq. (2.3) used for v_n. The pdf of the nth received sample conditioned on d_n and h_n is

$$f\left(r_n \,\middle|\, d_n = d, h_n\right) = \frac{1}{\pi \mathcal{N}_0} \exp\left\{ -\frac{1}{\mathcal{N}_0} \left| r_n - \sqrt{\mathcal{E}_s} \exp\left\{ j\frac{2\pi}{M} d \right\} h_n \right|^2 \right\} \ . \tag{4.10}$$

because the noise sample, w_n, has a complex Gaussian distribution. This equation is similar to the equivalent one for the AWGN channel, Eq. (3.6), except that the sample of the time-variant fading process, h_n, is used in place of the carrier phase error, ϕ_c. As with the AWGN channel, the inner decoder for the frequency-flat fading channel can be implemented with the

APP algorithm, based on the FSM model for the differential encoder. The branch metrics are $\mu_n(d) = f\left(r_n \,\middle|\, d_n = d, h_n\right)$ given by Eq. (4.10), but otherwise the APP algorithm is implemented in the same manner as for the AWGN channel.

4.3.2 Noncoherent Detection

The inner decoder to be used with noncoherent detection for the Rayleigh frequency-flat fading channel is very similar to the one for the AWGN channel. However, instead of estimating the static carrier phase error, the decoder must track the time-variant fading process. Therefore, different branch metrics are used, but both decoders use the APP algorithm. Like the decoder for the AWGN channel, this decoder is based on limiting the size of the observation window over which the fading process is estimated when calculating the branch metrics.

The pdf of the nth received sample conditioned on the transmitted symbols and the previous samples is $f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right)$. In Appendix C.2 it is shown that, for the received samples given by Eq. (4.6), this conditional pdf can be expressed as

$$f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right) = \frac{1}{2\pi\sigma_n^2} \exp\left\{-\frac{1}{2\sigma_n^2}\left|r_n - \tilde{v}_n \hat{\tilde{h}}_n\right|^2\right\} , \tag{4.11}$$

where

$$\hat{\tilde{h}}_n \triangleq \frac{1}{\mathcal{E}_s} \sum_{m=0}^{n-1} c_{n,m}^* r_m \tilde{v}_m^* \tag{4.12}$$

is the nth-order linear minimum mean squared error (MMSE) predictor of h_n, based on the previous samples, $\underline{r}_0^{n-1}$, and conditioned on the hypothetical transmitted symbols. For any

n, the prediction coefficients $\{c_{n,m}\}$ are the solution to the Wiener-Hopf equation,

$$\begin{bmatrix} \mathcal{E}_s\phi_0^{(F)} + \frac{\mathcal{N}_0}{2} & \mathcal{E}_s\phi_1^{(F)} & \cdots & \mathcal{E}_s\phi_{n-1}^{(F)} \\ \mathcal{E}_s\phi_1^{(F)} & \mathcal{E}_s\phi_0^{(F)} + \frac{\mathcal{N}_0}{2} & \cdots & \mathcal{E}_s\phi_{n-2}^{(F)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{E}_s\phi_{n-1}^{(F)} & \mathcal{E}_s\phi_{n-2}^{(F)} & \cdots & \mathcal{E}_s\phi_0^{(F)} + \frac{\mathcal{N}_0}{2} \end{bmatrix} \begin{bmatrix} c_{n,0} \\ c_{n,1} \\ \vdots \\ c_{n,n-1} \end{bmatrix} = \begin{bmatrix} \mathcal{E}_s\phi_n^{(F)} \\ \mathcal{E}_s\phi_{n-1}^{(F)} \\ \vdots \\ \mathcal{E}_s\phi_1^{(F)} \end{bmatrix} , \tag{4.13}$$

and the variance is

$$\sigma_n^2 = \frac{\mathcal{N}_0}{2} + \mathcal{E}_s \left(\phi_0^{(F)} - \sum_{m=0}^{n-1} c_{n,m}\phi_{n-m}^{(F)} \right) . \tag{4.14}$$

As with the AWGN channel, a considerable reduction in decoder complexity can be realized by limiting the number of samples used to predict the fading to some small number, Z, again with Z typically less than five. Instead of using the nth-order predictor, $\hat{\tilde{h}}_n$, the Zth-order predictor of h_n,

$$\hat{\tilde{h}}_n' \triangleq \frac{1}{\mathcal{E}_s} \sum_{m=n-Z}^{n-1} c_{Z,m-(n-Z)}^* r_m \tilde{v}_m^* , \tag{4.15}$$

is used, where the coefficients $\{c_{Z,0}, c_{Z,1}, \ldots, c_{Z,Z-1}\}$ are calculated by solving the Wiener-Hopf equation with Z in place of n. Note that limiting the number of samples used for the predictor is equivalent to making the approximation

$$f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right) \cong f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_{n-Z}^{n-1}\right) , \tag{4.16}$$

which is given by substituting Eq. (4.15) for $\hat{\tilde{h}}_n'$ into

$$f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_{n-Z}^{n-1}\right) = \frac{1}{2\pi\sigma_Z^2} \exp\left\{ -\frac{1}{2\sigma_Z^2} \left| r_n - \tilde{v}_n \hat{\tilde{h}}_n' \right|^2 \right\} , \tag{4.17}$$

where the variance is

$$\sigma_Z^2 = \frac{\mathcal{N}_0}{2} + \mathcal{E}_s \left(\phi_0^{(F)} - \sum_{m=n-Z}^{n-1} c_{Z,m-(n-Z)}\phi_{n-m}^{(F)} \right) . \tag{4.18}$$

Alternatively, by defining $P_z \triangleq c_{Z,Z-z}$, the predictor can be written as

$$\begin{aligned}\widehat{\widetilde{h}}'_n &\triangleq \frac{1}{\mathcal{E}_s}\sum_{m=n-Z}^{n-1} P^*_{n-m} r_m \widetilde{v}^*_m \\ &= \frac{1}{\mathcal{E}_s}\sum_{z=1}^{Z} P^*_z r_{n-z}\widetilde{v}^*_{n-z} \ ,\end{aligned} \tag{4.19}$$

and the variance as

$$\sigma_Z^2 = \frac{\mathcal{N}_0}{2} + \mathcal{E}_s\left(\phi_0^{(F)} - \sum_{z=1}^{Z} P_z \phi_z^{(F)}\right) \ . \tag{4.20}$$

The prediction coefficients are the solution to

$$\begin{bmatrix} \mathcal{E}_s\phi_0^{(F)} + \frac{\mathcal{N}_0}{2} & \mathcal{E}_s\phi_1^{(F)} & \cdots & \mathcal{E}_s\phi_{Z-1}^{(F)} \\ \mathcal{E}_s\phi_1^{(F)} & \mathcal{E}_s\phi_0^{(F)} + \frac{\mathcal{N}_0}{2} & \cdots & \mathcal{E}_s\phi_{Z-2}^{(F)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{E}_s\phi_{Z-1}^{(F)} & \mathcal{E}_s\phi_{Z-2}^{(F)} & \cdots & \mathcal{E}_s\phi_0^{(F)} + \frac{\mathcal{N}_0}{2} \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_Z \end{bmatrix} = \begin{bmatrix} \mathcal{E}_s\phi_1^{(F)} \\ \mathcal{E}_s\phi_2^{(F)} \\ \vdots \\ \mathcal{E}_s\phi_Z^{(F)} \end{bmatrix} . \tag{4.21}$$

The conditional pdf of Eq. (4.17) can also be expressed in term for the hypothetical data symbols, $\underline{\widetilde{b}}$, instead of the transmitted symbols, $\underline{\widetilde{v}}$. Since $\widetilde{v}_n = \widetilde{v}_{n-1}\exp\left\{j\frac{2\pi}{M}\mathrm{GM}\left[\widetilde{b}_n\right]\right\}$, by defining $\widetilde{y}'_n \triangleq \widehat{\widetilde{h}}'_n \widetilde{v}_{n-1}$, the conditional pdf can be written as

$$f\left(r_n \,\middle|\, \underline{b} = \underline{\widetilde{b}}, \underline{r}_{n-Z}^{n-1}\right) = \frac{1}{2\pi\sigma_Z^2}\exp\left\{-\frac{1}{2\sigma_Z^2}\left|r_n - \widetilde{y}_n \exp\left\{j\frac{2\pi}{M}\mathrm{GM}\left[bt_n\right]\right\}\right|^2\right\} \ . \tag{4.22}$$

Furthermore, $\widetilde{y}'_n$ can be expressed as

$$\widetilde{y}'_n \triangleq \widehat{\widetilde{h}}'_n \widetilde{v}_{n-1} = \frac{1}{\mathcal{E}_s}\sum_{z=1}^{Z} P^*_z r_{n-z}\widetilde{v}^*_{n-z}\widetilde{v}_{n-1} \ , \tag{4.23}$$

which, when expressed in terms of the differentially encoded symbols is

$$\widetilde{y}'_n = \sum_{z=1}^{Z} P^*_z r_{n-z}\exp\left\{j\frac{2\pi}{M}\left(\widetilde{d}_{n-1} - \widetilde{d}_{n-z}\right)\right\} \ . \tag{4.24}$$

Expanding the recursive definition of $\widetilde{d}_n$ given by Eq. (2.1) yields

$$\widetilde{y}'_n = \sum_{z=1}^{Z} P^*_z r_{n-z}\exp\left\{j\frac{2\pi}{M}\sum_{k=1}^{z-1}\mathrm{GM}\left[\widetilde{b}_{n-k}\right]\right\} \ , \tag{4.25}$$

where the value of the k-indexed summation is defined to equal zero when $z = 1$. Clearly $\tilde{y}'_n$ depends only on $\underline{r}_{n-Z}^{n-1}$ and $\underline{\tilde{b}}_{n-Z+1}^{n-1}$. Therefore,

$$f\left(r_n \,\middle|\, \underline{b} = \underline{\tilde{b}}, \underline{r}_{n-Z}^{n-1}\right) = f\left(r_n \,\middle|\, \underline{b}_{n-Z+1}^{n} = \underline{\tilde{b}}_{n-Z+1}^{n}, \underline{r}_{n-Z}^{n-1}\right) \ . \tag{4.26}$$

For the given received samples, the number of distinct values of this conditional pdf is limited to M^Z, corresponding to all the possible hypothetical realizations of $\underline{\tilde{b}}_{n-Z+1}^{n}$.

Since the conditional pdf depends only on a limited number of the data symbols, trellis-based decoding is suggested. Following the same arguments used for the AWGN channel, it is readily shown that the inner decoder for use with noncoherent detection can be implemented with the APP algorithm. The algorithm is based on the same FSM model as used with the AWGN channel, with M^{Z-1} states and with the state transition matrix, $\mathrm{ST_Z}\,[\bullet, \bullet]$, given by Eq. (3.21), and the symbol generation matrix, $\mathrm{SG_Z}\,[\bullet, \bullet]$, given by Eq. (3.23). However, the branch metrics are

$$\mu_n(\underline{\tilde{b}}_{n-Z+1}^{n}) = \frac{1}{2\pi\sigma_Z^2} \exp\left\{ -\frac{1}{2\sigma_Z^2} \left| r_n - \sum_{z=1}^{Z} P_z^* r_{n-z} \exp\left\{ j\frac{2\pi}{M} \sum_{k=0}^{z-1} \mathrm{GM}\,[bt_{n-k}] \right\} \right|^2 \right\} \ , \tag{4.27}$$

where $\{P_z\}$ and σ_Z^2 are specific to the fading model but independent of the transmitted data. $\{P_z\}$ is given by Eq. (4.21) and σ_Z^2 is given by Eq. (4.20), where it is assumed that the statistical nature of the channel is known for computing these parameters. In all other regards the implementation of the inner decoder is identical for both channel models.

4.4 Simulation Results and Related Discussion

The performance of the proposed coherent and noncoherent systems was extensively investigated by means of computer simulation. Performance is compared with coherently-demodulated absolutely-encoded M-PSK and traditional differentially-detected differentially-encoded M-PSK. Because decoder performance is very much a function of the fading

rate of the channel, as specified by the B_dT product, three different fading rates are considered: *i*) slow fading with $B_dT = 0.001$, *ii*) moderate fading with $B_dT = 0.01$, and *iii*) fast fading with $B_dT = 0.1$. The differences between bit-by-bit and symbol-by-symbol interleaving are investigated, and the effects of the interleaver size and the constraint length of the convolutional code are examined.

Unless otherwise indicated, the following system parameters were used for the simulations. A 16-state, rate 1/2, convolutional code with generator $(23, 35)_8$ is used, so $k_c = 1$, $n_c = 2$, and $L_c = 5$, and code words of length $N_c = 1024$ symbols are produced by the encoder. A random bit-by-bit interleaver is used, where the code bits are rearranged in a randomly selected fashion. The interleaved bits are grouped into quarternary symbols which are Gray mapped and differentially encoded prior to transmission using QPSK ($M = 4$). Root raised-cosine filtering is used at the transmitter and receiver, with a rolloff parameter of $\beta = 0.2$. The land-mobile fading model is used for the channel. Additional details regarding the simulations can be found in Appendix D.

4.4.1 Coherent Detection

To provide a guideline to measure the performance of noncoherent detection against, the coherent detector was first examined. In the following, the performance of the novel coherent system is investigated for the three fading rates, and is compared with that of two traditional coherent detection systems, both using absolutely-encoded QPSK. One uses the same 16-state convolutional code used in the concatenated code, and the other uses a more powerful, 256-state convolutional code, with generator $(561, 753)_8$. To support bit interleaving with the traditional systems, the coherently detected samples are first rotated by $-\pi/4$ radians, producing

$$z_n = r_n e^{-j\frac{\pi}{4}} , \tag{4.28}$$

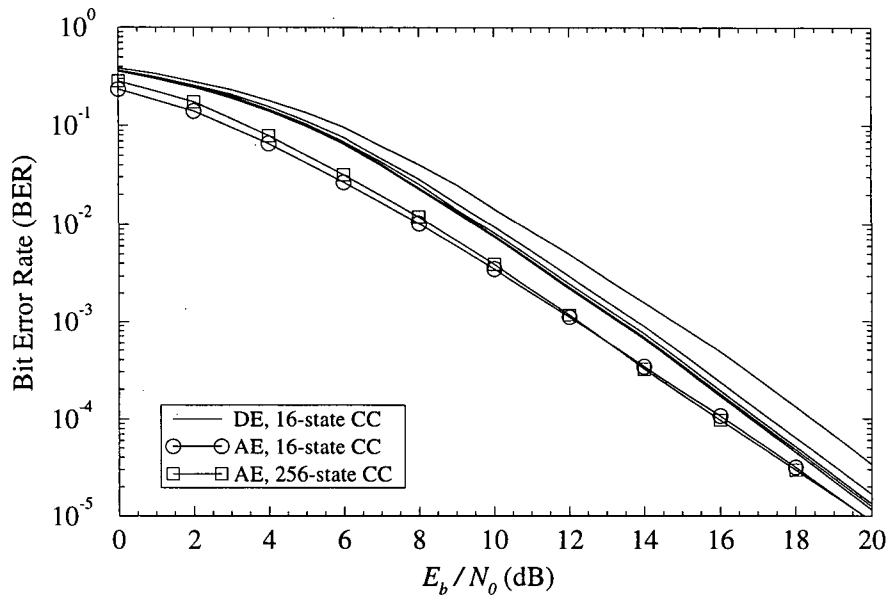


Figure 4.1: Coherent Demodulation ($B_dT = 0.001$): BER performance of the concatenated code after each of the first five iterations. Also shown is the performance of absolutely-encoded (AE) coherently-demodulated QPSK, with 16-state and 256-state convolutional codes (CC).

so that bit $b_n^{(0)}$ is reflected only the real part of z_n, and bit $b_n^{(1)}$ is reflected only in the imaginary part[2]. The real and imaginary parts of z_n are then separated and passed serially into the deinterleaver, which operates on samples[3] and uses the inverse mapping of the bit interleaver in the encoder. A Viterbi decoder uses the deinterleaved samples to decode the convolutional code.

For slow fading, the BER performance of the iterative decoder with coherent detection, with the inner decoder implemented as described in Section 4.3.1, is shown in Fig. 4.1, for each of the first five iterations. As can be seen, there is very little improvement in

[2] This is equivalent to using the $\pi/4$-shift QPSK signal constellation where the transmitted phases are taken from the set $\{\pi/4, 3\pi/4, 5\pi/4, 7\pi/4\}$ instead of the regular QPSK signal constellation described in Section 2.1.

[3] This is unlike the deinterleaver for the iterative decoder, which operates on sets of probabilities.

performance after the second iteration, although there is a 1.6 dB reduction in the SNR required to achieve a BER of 10^{-4} between the first and fifth iterations. However, both absolutely-encoded systems are capable of even better performance, with an advantage of 0.7 dB and 0.8 dB for the 16-state CC and the 256-state CC, respectively, at a BER of 10^{-4}, as compared to the fifth iteration of the concatenated system. Unlike for the AWGN channel, it is not advantageous to use differential encoding for the slow fading channel. When the fading rate is slow, a single fade can encompass a large number of symbol transmissions, so the BER is dominated by the few words which are completely lost due to a long fade. Since no coding scheme, no matter how powerful, is able to determine the transmitted data when nothing has been received, the performances of all three codes shown in Fig. 4.1 are fairly similar. The reason why the concatenated code is worse than the other two is that differential encoding inherently doubles the number of "errors" at the output of the inner decoder, increasing the burden on the decoder for the convolutional code. When a significant portion of the word is severely attenuated, the iterative decoder is unable to generate much confidence in its interim decisions, so there is very little gain with each iteration. More specifically, for most received words the fading is not severe, so the decoder is able to reduce, then eliminate, the errors by iterating. For a small fraction of words, however, the decoder is unable to make any progress, so a large number of errors remains. The effects of these words dominate the BER. As most of the received words are correctable by the iterative decoder, the concatenated code is much more attractive when the WER is of interest, as shown in Fig. 4.2. Differential encoding provides a 2.2 dB advantage for a WER of 10^{-3}, as compared to the same convolutional code with absolute encoding, and matches the performance of the more-complex 256-state convolutional code. This shows that the bit errors in the concatenated code are concentrated in only a few erroneously decoded words.

As the fading rate increases to a more moderate rate (e.g., $B_dT = 0.01$), the duration

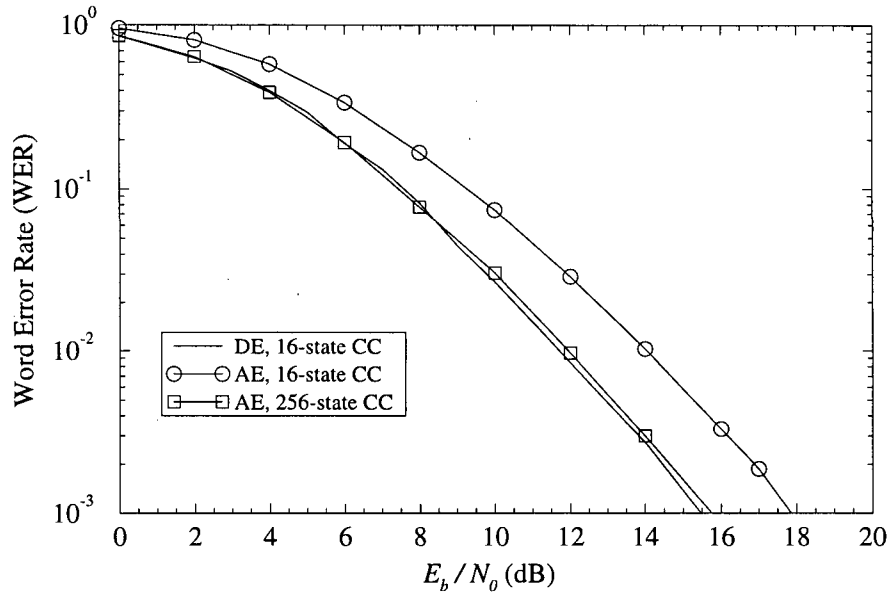


Figure 4.2: Coherent Demodulation ($B_dT = 0.001$): WER performance of the concatenated code after the fifth iteration. Also shown is the performance of absolutely-encoded (AE) coherently-demodulated QPSK, with 16-state and 256-state convolutional codes (CC).

of the fades decreases, so the likelihood of losing an entire block decreases. As a result, the difference between the performances of the codes become more pronounced. In addition, the iterative decoder is able to improve its confidence with each iteration, as shown in Fig. 4.3. At this fading rate of $B_dT = 0.01$, the results show that differential encoding improves the performance of the convolutional code by about 2 dB after five iterations at a BER of 10^{-5}, although the concatenated code is unable to match the performance of the 256-state convolutional code.

When the fading rate is fast (e.g., $B_dT = 0.1$), fades affect only a few consecutive symbols. Although fading will affect some symbols in nearly every transmitted block, it is very unlikely that an entire block will be affected. Therefore, the random-error correcting capability of the code is the dominant factor governing system performance. At this fading

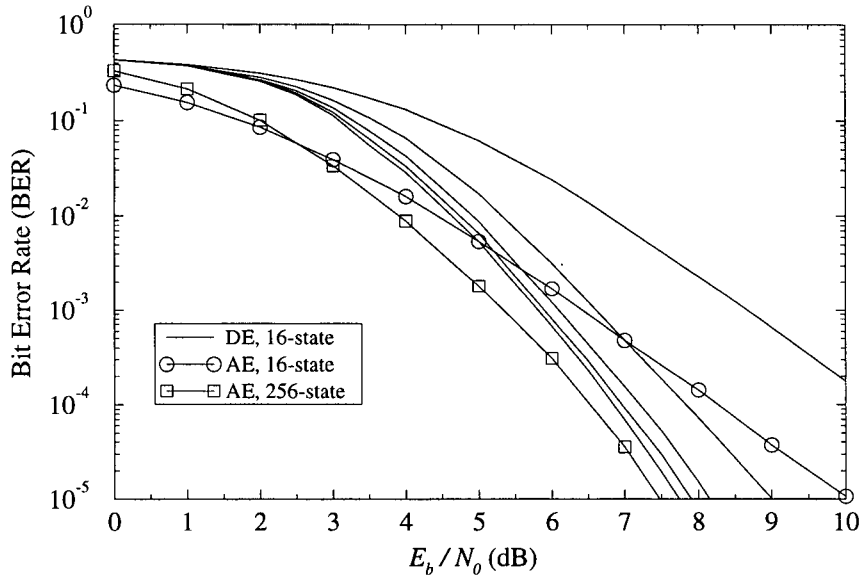


Figure 4.3: Coherent Demodulation ($B_dT = 0.01$): BER performance of the concatenated code after each of the first five iterations. Also shown is the performance of absolutely-encoded (AE) coherently-demodulated QPSK, with 16-state and 256-state convolutional codes (CC).

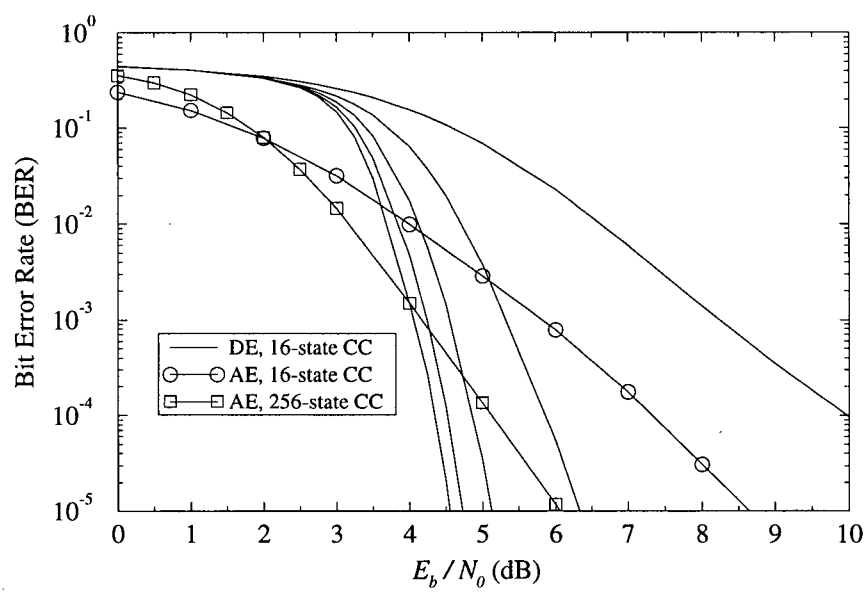


Figure 4.4: Coherent Demodulation ($B_dT = 0.1$): BER performance of the concatenated code after each of the first five iterations. Also shown is the performance of absolutely-encoded (AE) coherently-demodulated QPSK, with 16-state and 256-state convolutional codes (CC).

rate the results presented in Fig. 4.4 show that iterative decoding is capable of significantly improving the performance, by over 7 dB at a BER of 10^{-5} between the first and fifth iterations, and, unlike with slow fading, it appears there is room for further, albeit minor, improvement with additional iterations. It is also evident that differential encoding improves the performance of the convolutional code by about 4 dB, and the concatenated code can even outperform the 256-state convolutional code, by about 1.5 dB. Clearly, the advantages of using the concatenated code are more apparent with faster fading rates.

4.4.2 Noncoherent Detection

As shown above, the concatenated code is fairly effective for the fading channel with coherent detection, although it performs much better in faster fading. The primary motivation for using the concatenated code, however, is to facilitate noncoherent detection. Noncoherent detection alleviates the need for analogue circuitry to track the fading process, a task which can be very unreliable in fading environments. In the following, the performance of the novel noncoherent system is compared with that of the coherent system to measure the noncoherence penalty, and with two traditional differential detection systems, both of which also use differentially-encoded QPSK and bit interleaving. One uses the same 16-state convolutional code used in the concatenated code, and the other uses the 256-state convolutional code used in the comparisons above. Both use a single differential detector, and, to support bit interleaving with the traditional systems, the received samples are differentially detected and rotated by by $-\pi/4$ radians, producing

$$z_n = r_n r_{n-1}^* e^{-j\frac{\pi}{4}} , \tag{4.29}$$

so that bit $b_n^{(0)}$ is reflected only the real part of z_n, and bit $b_n^{(1)}$ is reflected only in the imaginary part[4]. The real and imaginary parts of z_n are then be separated and passed serially into the deinterleaver, which uses the inverse mapping of the bit interleaver in the encoder. A Viterbi decoder uses the deinterleaved samples to decode the convolutional code.

The inner decoder was implemented as described in Section 4.3.2 with $Z = 2$ differential detectors, and incorporated in the iterative decoding structure. For a moderate fading rate of $B_dT = 0.01$, the BER performance of the noncoherent decoder after each of the first five iterations is shown in Fig. 4.5. The performance does improve with each iteration, and although there is a 2 dB improvement at a BER of 10^{-5} between the first and second iterations, the gains with additional iterations are negligible. Because only two previous

[4] In practice, there will be some cross-talk between the real and imaginary parts. This is an inherent problem with differential detection.

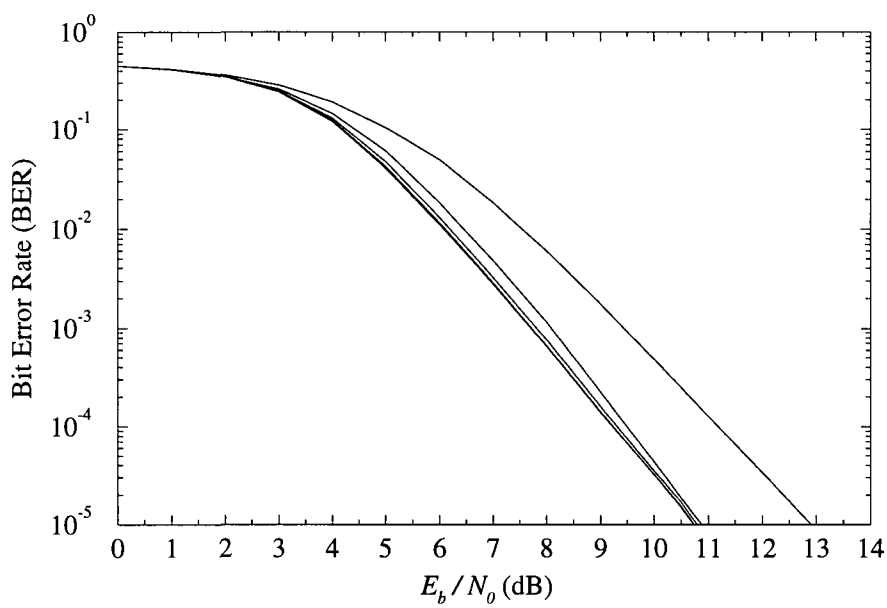


Figure 4.5: Noncoherent Demodulation ($Z = 2$): BER performance after each of the first five iterations of the noncoherent decoder with $Z = 2$ differential detectors at a fading rate of $B_dT = 0.01$.

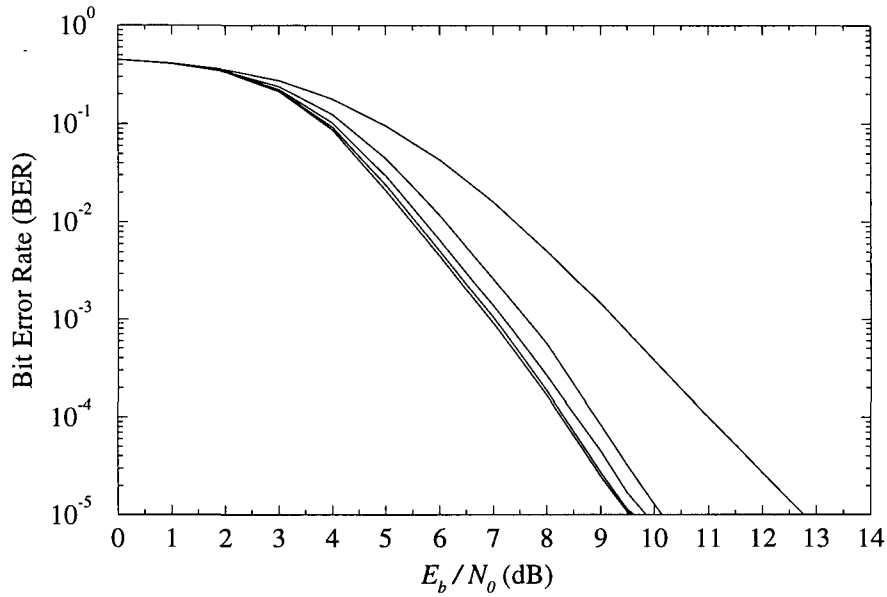


Figure 4.6: Noncoherent Demodulation ($Z = 3$): BER performance after each of the first five iterations of the noncoherent decoder with $Z = 3$ differential detectors at a fading rate of $B_dT = 0.01$.

samples are used to estimate the time-variant fading process, the estimates are not particularly reliable, so after only a small number of iterations the system performance is limited by the inaccuracy of the channel estimation. Increasing the number of differential detectors to $Z = 3$ alleviates this problem somewhat, as shown by Fig. 4.6, with a 2.6 dB improvement between the first and second iterations, and an addition 0.6 dB improvement between the second and fifth iterations.

For comparison, the BER performances for the $Z = 2$, $Z = 3$, and $Z = 4$ MDD decoders after the fifth iteration are shown in Fig. 4.7. As can be seen, increasing the number of differential detectors leads to a improvement in performance. At the BER of 10^{-5} there is a 1.2 dB improvement when Z is increased from two to three, and an additional 0.6 dB when Z is increased of four. These gains are because the increased size of the observation window allows for better tracking of the fading. However, even with $Z = 4$ there remains

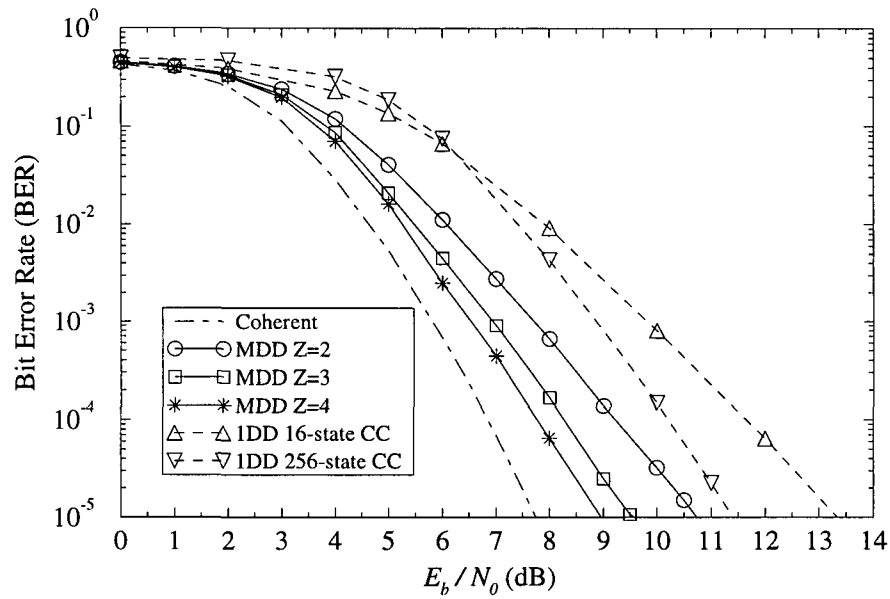


Figure 4.7: Noncoherent Demodulation ($B_dT = 0.01$): BER performance in moderate-rate fading of the noncoherent iterative MDD decoders after the fifth iteration, using $Z = 2$, 3, and 4 differential detectors. Also shown is the performance of the coherent iterative decoder after the fifth iteration, and traditional noncoherent single differential detectors (1DD) with 16-state and 256-state convolutional codes.

a 1.2 dB noncoherent penalty as compared to the iterative coherent decoder, which is also shown in Fig. 4.7. It appears that further increases of Z will lead to only very modest improvements in performance. Although ideally one would like to use a noncoherent decoder that can match the performance of an optimal coherent receiver, a more fair comparison is with other, traditional, noncoherent receivers. The performance of a single differential detector with the same 16-state convolutional code is also shown in Fig. 4.7. By using the $Z = 3$ iterative decoder there is almost a 4 dB gain in performance when compared to the traditional, non-iterative, single differential detector. From this point of view, the advantages of iterative decoding are obvious. Even when the comparison is made between decoders of similar complexity, the iterative decoder is advantageous. The performance of a single differential detector with a 256-state convolutional code is also shown in Fig. 4.7. In

this case, the iterative decoder outperforms the traditional one by over 2 dB.

At slower fading rates, such as $B_dT = 0.001$, the same observations of the iterative decoder performance apply, although in a less pronounced manner. The BER performance of the $Z = 2$ and $Z = 3$ MDD decoders after the fifth iteration are shown in Fig. 4.8. At a BER of 10^{-4} the $Z = 3$ decoder outperforms the $Z = 2$ decoder by 0.5 dB, but has a noncoherence penalty of 0.7 dB. The noncoherence penalty is lower for the slower fading rate[5] because it is easier to track the fading process when it is varying only slowly. The iterative decoder also outperforms the traditional systems, and there is a 2 dB difference between the $Z = 3$ MDD decoder and the 256-state code with only single differential detection.

At faster fading rates, such as $B_dT = 0.1$, the relationships between these various

[5] For the $Z = 3$ the noncoherence penalty is 1.8 dB when $B_dT = 0.01$.

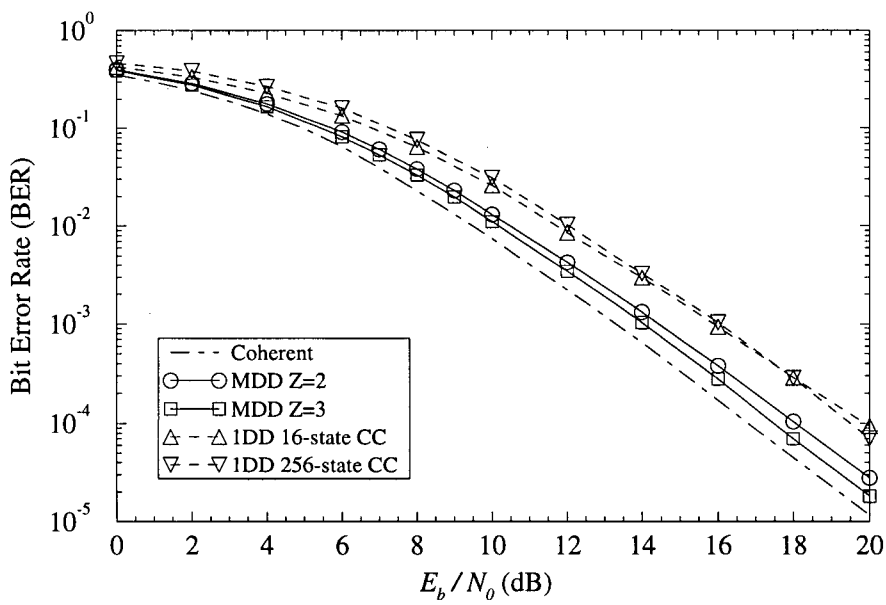


Figure 4.8: Noncoherent Demodulation ($B_dT = 0.001$): BER performance in slow fading of the noncoherent iterative MDD decoders after the fifth iteration, using $Z = 2$ and 3 differential detectors. Also shown is the performance of the coherent iterative decoder after the fifth iteration, and traditional noncoherent single differential detectors (1DD) with 16-state and 256-state convolutional codes (CC).

systems are much more pronounced, as shown by Fig. 4.9. There is a 2.5 dB difference between the $Z = 2$ and $Z = 3$ MDD decoders, but a 6 dB noncoherence penalty. The fading process is varying at such a fast rate that the noncoherent decoders have a difficult time tracking it, explaining the large noncoherence penalty. However, the traditional systems, which only have a single differential detector to track the fading, fare much worse. In the performance of both the 16-state CC and the 256-state CC, the beginning of an error floor are evident in Fig. 4.9.

The results given here for noncoherent detection illustrate that the proposed iterative MDD decoder is well-suited for the fading channel, giving performance much better than traditional single differential detection. The remainder of this section is devoted to an investigation of the effects of various system parameters on the performance.

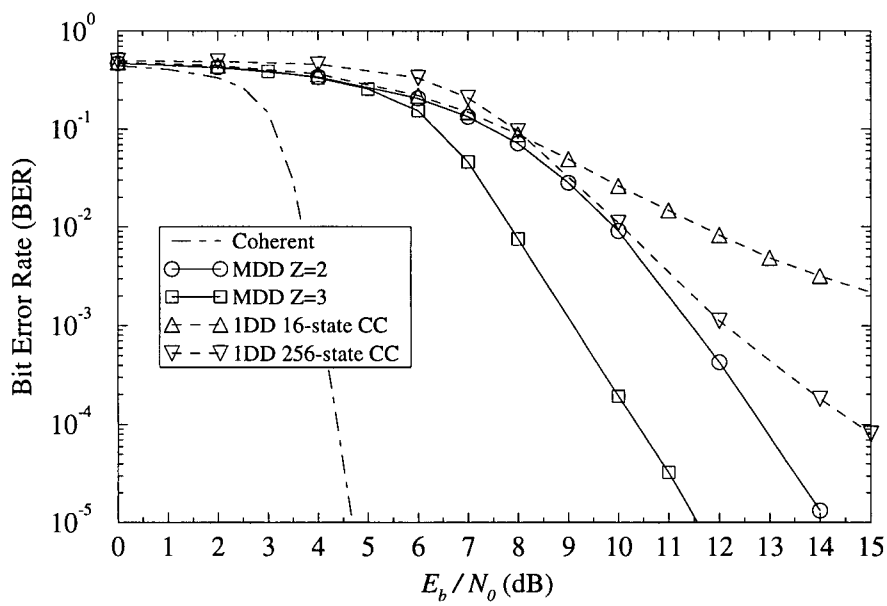


Figure 4.9: Noncoherent Demodulation ($B_dT = 0.1$): BER performance in fast fading of the noncoherent iterative MDD decoders after the fifth iteration, using $Z = 2$ and 3 differential detectors. Also shown is the performance of the coherent iterative decoder after the fifth iteration, and traditional noncoherent single differential detectors (1DD) with 16-state and 256-state convolutional codes (CC).

4.4.3 Bit vs. Symbol Interleaving

When fading affects the communication channel, the amplitudes of some of the transmitted symbols are attenuated, so some symbols will be lost in the fade. When this happens, all the bits for those symbols are lost. When symbol-by-symbol interleaving is used, this means that all the bits for a code symbol could be lost, hindering the performance of the decoder for the convolutional code (the outer decoder). However, when bit-by-bit interleaving is used, the bits in each transmitted symbol come from different code symbols, so it is less likely that all the bits for any particular code symbol will be lost. Therefore, for the fading channel, there is some advantage to using bit interleaving instead of symbol interleaving. On the other hand, as was shown in Section 3.4, the iterative decoder structure works better when symbol interleaving is used. As a result, the question of whether bit or symbol interleaving is the better choice for the fading channel is important. Fig. 4.10 shows the performance of both the coherent and noncoherent ($Z = 3$) iterative decoders, using both bit and symbol interleaving, at a fading rate of $B_dT = 0.01$. The results indicate that, for the coherent decoder, symbol interleaving provides performance that is slightly better than bit interleaving. That is, when coherent detection is used, the benefits of symbol interleaving for the iterative decoding channel outweigh the adverse affect for losing entire code symbols due to fades. However, when noncoherent detection is used, where less efficient use of iterative decoding is possible, this is not the case, and bit interleaving provides a sizable advantage (over 1 dB at a BER of 10^{-4}). For this reason bit interleaving has been used for the simulation results presented in this chapter.

4.4.4 Interleaver Size

As with the AWGN channel, the size of the interleaver is an important system parameter. With a larger interleaver the error correcting capabilities of the concatenated code improves.

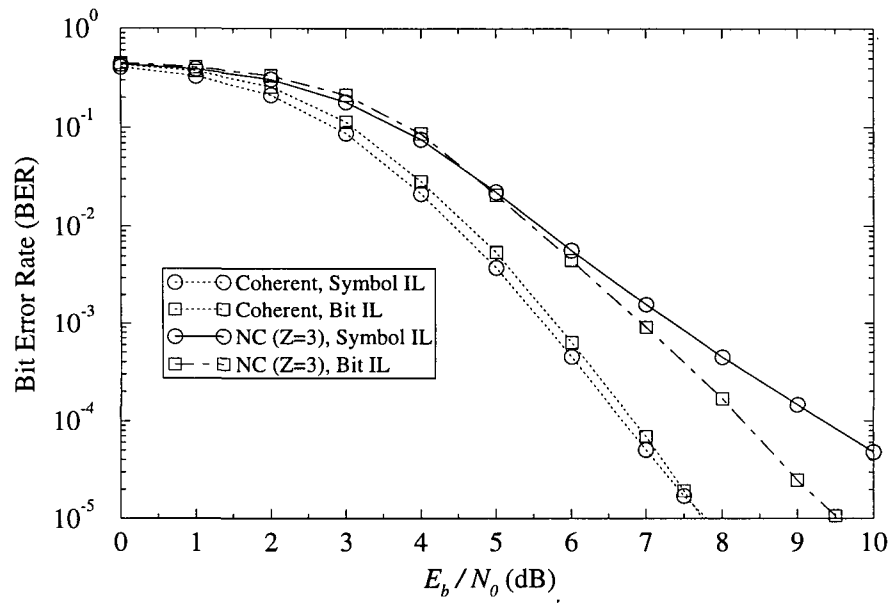


Figure 4.10: Bit vs. Symbol Interleaving ($B_dT = 0.01$): BER performance of the coherent and noncoherent ($Z = 3$) iterative decoders after the fifth iteration, using both bit and symbol interleaving. $B_dT = 0.01$.

In the fading channel, a large interleaver size also allows for wider spreading of symbols lost in fades, and reduces the likelihood that the entire transmitted word is lost in a single fade. The advantage of larger interleavers for coherent detection is shown in Fig. 4.11, with almost 11 dB difference in performance between $N = 256$ and $N = 4096$. When noncoherent detection is used, the differences in performance of different interleaver sizes is less pronounced, but there is still a great advantage to using larger sizes, as shown in Fig. 4.12 for $Z = 3$ when the fading rate is $B_dT = 0.01$. However, the noncoherent decoder is not quite as effective at exploiting the larger sizes as the coherent decoder. This is emphasized by the noncoherence penalty, which also increases with increased interleaver size, as shown in Table 4.1 (for a BER or 10^{-5}).

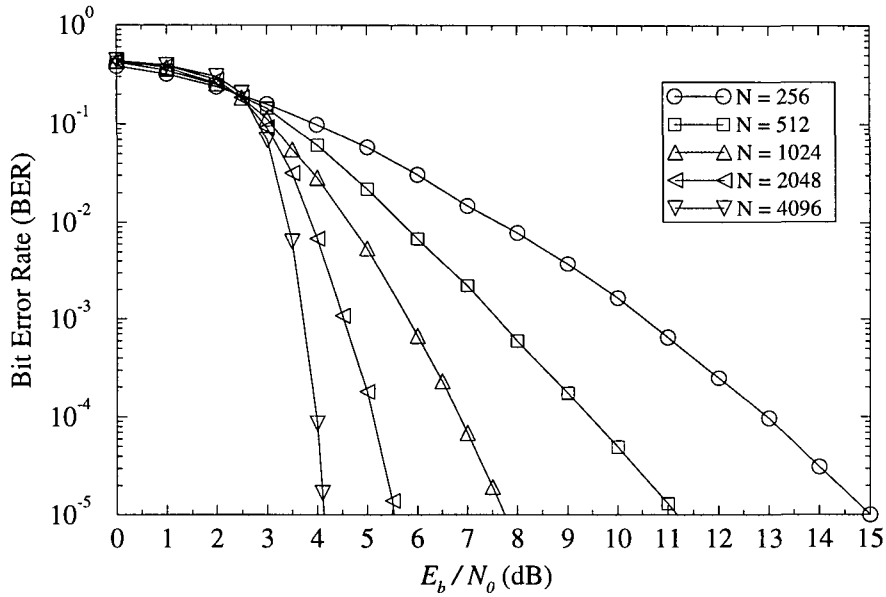


Figure 4.11: Interleaver Size (Coherent): Comparison of the performance with different word lengths (N). Coherent QPSK, $B_dT = 0.01$.

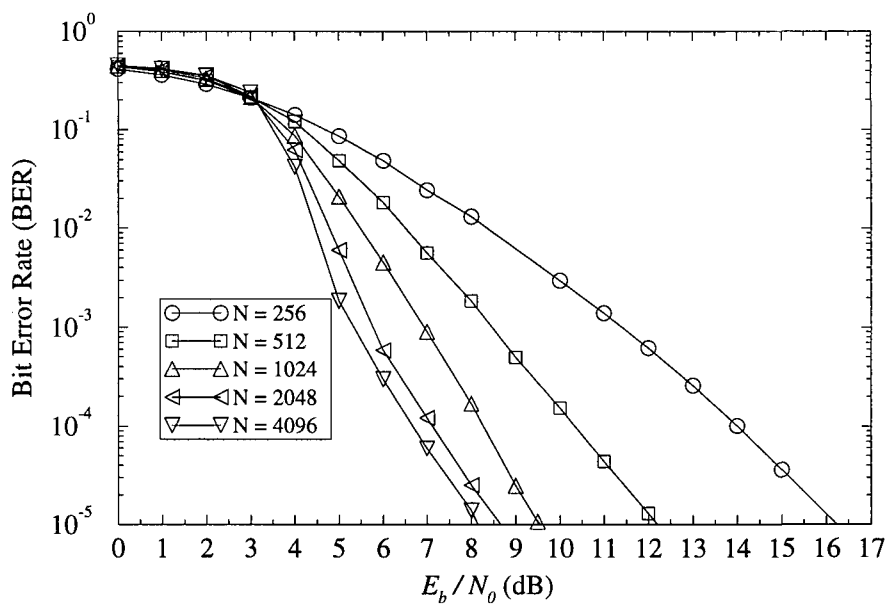


Figure 4.12: Interleaver Size Length (NC, $Z = 3$): Comparison of the performance with different word lengths (N). Noncoherent QPSK, $Z = 3$, $B_dT = 0.01$.

Word Length (N)	Noncoherence Penalty (dB)
256	1.2
512	1.0
1024	1.8
2048	3.2
4096	4.1

Table 4.1: Noncoherence penalty at BER $= 10^{-5}$ as a function of the interleaver size. Noncoherent QPSK, $Z = 3$, $B_dT = 0.01$.

4.4.5 Constraint Length

Both the coherent and noncoherent systems were tested using different convolutional codes of varying constraint lengths. When coherent detection is used there is very little difference in the performance of each of the codes, as the results in Fig. 4.13 indicate. These curves are for concatenated codes based on convolutional codes[6] with constraint lengths of $L_c = 3$, 4, 5, 6, and 7, after five iterations with the coherent detection and at a fading rate of $B_dT = 0.01$. As is the case with the AWGN channel, using shorter constraint length convolutional codes actually gives better performance when coherent detection is used. This is not the case when noncoherent detection is used, as indicated by the results shown in Fig. 4.14. There is a clear advantage in terms of performance by using longer length codes over shorter ones, with the length $L_c = 7$ code outperforming the $L_c = 3$ code by about 3 dB. Longer codes are better able to compensate for inaccurate channel estimation resulting from noncoherent detection, so the noncoherence penalty, and the BER, is reduced. When the $L_c = 7$ convolutional code is used, the noncoherence penalty is only 0.6 dB at a BER of 10^{-5}.

[6] These codes have been taken from [24, Table 8-2-1], and are the largest-free-distance rate-1/2 codes for each constraint length.

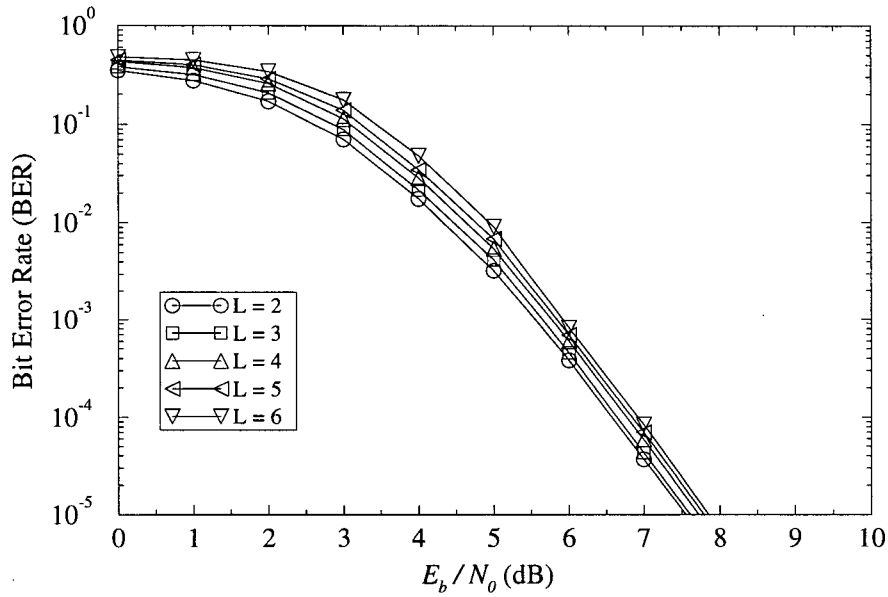


Figure 4.13: Constraint Length (Coherent): Performance of concatenated codes based on convolutional codes with various constraint lengths (L_c). Coherent QPSK, $N = 1024$, $N_p = 5$, $B_dT = 0.01$.

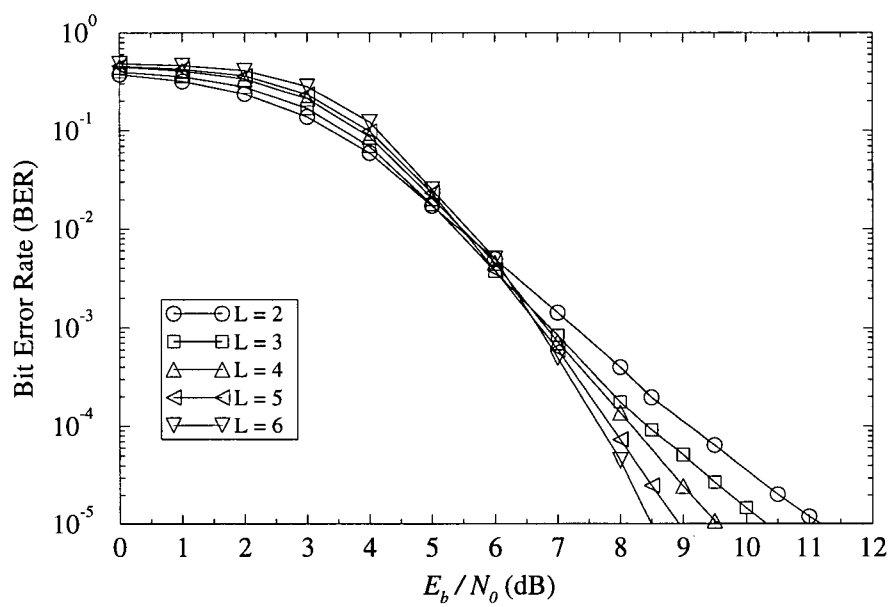


Figure 4.14: Constraint Length (NC, $Z = 3$): Performance of concatenated codes based on convolutional codes with various constraint lengths (L_c). Noncoherent QPSK, $Z = 3$, $N = 1024$, $N_p = 5$, $B_dT = 0.01$.

4.4.6 Conclusions

In this chapter two novel soft-output decoders of differentially encoded M-PSK signals transmitted over the Rayleigh frequency-flat fading channel have been presented. One is for use with coherent detection and the other for noncoherent detection. Their use within an iterative decoding structure has been proposed, and their performance evaluated by means of computer simulation.

It is shown that the proposed noncoherent system is able to substantially outperform a more traditional single differential detector, with performance improving as the number of previously received samples used to track the fading (Z) is increased. Although the noncoherent detector is effective in a slow fading environment, its performance relative to the traditional technique is much greater in fast fading. It has been shown that bit-by-bit interleaving gives better performance than symbol-by-symbol interleaving when fading is present. As with the AWGN channel, the size of the interleaver is one of the most important parameters, with performance improving with increased interleaver size. Increasing the constraint length of the convolutional code also leads to improved performance when noncoherent detection is used, although this is not the case for the coherent system.

Chapter 5

Rayleigh Frequency-Selective Fading

THE third application of the proposed coding/modulation scheme and iterative decoder, for the Rayleigh frequency-selective fading channel, is discussed in this chapter. This chapter has the same structure as the previous two, including a description of the continuous-time channel model in Section 5.1; a description of the corresponding discrete-time channel model in Section 5.2; derivations of the inner decoders, for coherent and non-coherent detection, in Section 5.3; and performance evaluation results in Section 5.4.

5.1 Continuous-Time Channel Model

With the Rayleigh frequency-selective fading channel the received signal is modelled as

$$r_c(t) = \mathbf{Re}\left\{\int_{-\infty}^{\infty} v(t-\tau)h(\tau;t)\, d\tau\, \sqrt{2}\exp\{j2\pi f_c t\}\right\} + w_c(t)\ , \tag{5.1}$$

where $v(t)$ is the transmitted lowpass signal, $w_c(t)$ is the background noise, with the same model as for the AWGN channel, and $h(\tau;t)$ is the time-variant complex lowpass equivalent channel impulse response, which corresponds to the response of the channel at time t to an impulse applied at time $t-\tau$. For any fixed delay, τ, the impulse response is modelled as a zero-mean, wide-sense stationary, complex Gaussian random process. Thus, over τ, $h(\tau;t)$ represents a continuum of random processes. The "uncorrelated scattering" model is used, so the processes corresponding to two different delays are uncorrelated [24]. The

cross-correlation function for the time-variant impulse response is

$$\phi_S(\tau_1, \tau_2; \Delta t) \triangleq \frac{1}{2}\mathbf{E}\left[h(\tau_1; t + \Delta t)h^*(\tau_2; t)\right] = \phi_F(\Delta t)\phi_M(\tau_1)\delta(\tau_2 - \tau_1) \ , \tag{5.2}$$

where $\phi_F(\Delta t)$ is the time-spaced autocorrelation function and $\phi_M(\tau)$ is the *multipath intensity profile* of the impulse response. The autocorrelation function reflects the time-variant nature of the impulse response, and the models given in Chapter 4 for frequency-flat fading are also applicable to frequency-selective fading. The multipath intensity profile gives the average received signal power as a function of the delay, τ.

Based on this model for the continuous-time channel, the model for the equivalent discrete-time channel can be derived.

5.2 Discrete-Time Channel Model

The combined effects of the signal generator, the continuous-time channel, and the detector on the transmitted symbols is encapsulated by a discrete-time channel model. As shown in Appendix B, for the Rayleigh frequency-selective fading channel the received samples can be modelled as

$$r_n = \sum_{k=0}^{L-1} v_{n-k} h_{n,k} + w_n \ , \tag{5.3}$$

where $\{v_n\}$ are the transmitted symbols[1], $\{w_n\}$ is the discrete-time noise process, with the same properties as for the AWGN channel, and $\{h_{n,k}\}$ is the time-variant discrete-time channel impulse response. The random variable $h_{n,k}$ is the response of the channel at time index n to a discrete impulse applied at time index $n - k$. The delay spread of the discrete-time channel is limited to L symbols, so the sample received at time index n depends on the symbol transmitted at time index n as well as the previous $L - 1$ transmitted symbols. As a result, the channel causes intersymbol interference (ISI). For any delay, k, the sequence

[1] With v_n taken equal to zero for all $n \notin \{0, 1, \ldots, N\}$.

$h_{0,k}, h_{1,k}, \ldots, h_{N,k}$ is modelled as a zero-mean, stationary complex Gaussian discrete-time random process, which is the same model as used for the discrete-time fading process of the Rayleigh frequency-flat fading channel. Although the uncorrelated scattering assumption is used in the continuous-time channel model, the filtering used by the detector introduces correlation in the discrete-time channel impulse response. That is, the processes for any two delays are correlated. The cross-correlation function for the discrete-time channel impulse response is

$$\phi_{k,l;m}^{(S)} \triangleq \frac{1}{2}\mathbf{E}\left[h_{n+m,k}h_{n,l}^{*}\right] = \phi_{m}^{(F)}\lambda_{k,l} \ , \tag{5.4}$$

where $\phi_m^{(F)}$ reflects the time-variant nature of the impulse response, and

$$\lambda_{k,l} \triangleq \int_{-\infty}^{\infty} \phi_M(\tau)h_{TR}(kT-\tau)h_{TR}^{*}(lT-\tau)\, d\tau \tag{5.5}$$

reflects the correlation between the different delays. In Eq. (5.5), $\phi_M(\tau)$ is the multipath intensity profile of the continuous-time channel impulse response and $h_{TR}(t)$, defined by Eq. (B.11), is the combined impulse response of the transmit and receive filters. The design of the inner decoder is based on knowledge of $\phi_m^{(F)}$ and $\{\lambda_{k,l}\}$, so these are considered system parameters.

Although this channel model is derived for noncoherent detection, it can also be used with coherent detection. When coherent detection is used it is assumed that the detector perfectly tracks the continuous-time channel impulse response and provides the equivalent discrete-time channel impulse response to the decoder as side information. In this case the model for use with coherent detection is the same as for noncoherent detection, except that $\{h_{n,k}\}$ is perfectly known.

The relationship between the transmitted symbols and the received samples, as given by Eq. (5.3) for either coherent or noncoherent detection, is used to guide the design of the inner decoder.

5.3 Inner Decoder

In this section the inner decoders for both coherent and noncoherent detection are presented. Both make use of the APP algorithm, implemented with the same FSM model. The inner decoder for use with noncoherent demodulation is much more complex, however, reflecting the need to perform channel estimation.

5.3.1 Coherent Detection

As mentioned, when coherent detection is used, the discrete-time channel impulse response is provided to the decoder as side information. Because channel estimation is not required, the inner decoder only needs to address the differential encoding and the ISI introduced by the channel. Decoding ISI (for absolutely encoded signals) was one of the original applications of the APP algorithm [38]. Furthermore, this is the approach used by the "turbo-equalizer" for iterative decoding of static ISI with coherently detected, absolutely encoded M-PSK [15]. When differentially encoded M-PSK is used for transmission, the APP algorithm is still applicable, requiring only a minor modification to the FSM model.

The relationship between the data symbols and the received samples can be modelled as noisy observations of the output of a FSM. From Eq. (5.3), the observation at time index n is

$$r_n = \sum_{k=0}^{L-1} \sqrt{\mathcal{E}_s} \exp\left\{ j\frac{2\pi}{M} d_{n-k} \right\} h_{n,k} + w_n \ , \tag{5.6}$$

where $\{h_{n,k}\}$ are known. The input to the FSM at time index n is b_n and the output "symbol" is $\underline{d}_{n-L+1}^{n}$, which corresponds to the L transmitted symbols upon which r_n depends. The output depends not only on the input, but on the state, $s_n \triangleq \underline{d}_{n-L+1}^{n-1}$. The state space, $\mathcal{S}_{\mathrm{L}}$, is the set of all M-ary (L–1)-tuples. The state transition matrix from state

$s = (s^{(L-1)}, s^{(L-2)}, \ldots, s^{(1)})$ in response to input b is given by

$$\mathrm{ST_L}\,[s, b] \triangleq (s^{(L-2)}, \ldots, s^{(1)}, s^{(1)} \oplus \mathrm{GM}\,[b]) \tag{5.7}$$

so that

$$\mathrm{ST_L}\,[s_n, b_n] = \mathrm{ST_L}\left[\underline{d}_{n-L+1}^{n-1}, b_n\right] = (d_{n-L+2}, \ldots, d_{n-1}, d_{n-1} \oplus \mathrm{GM}\,[b_n]) = \underline{d}_{n-L+2}^{n} = s_{n+1}\ . \tag{5.8}$$

The symbol generation matrix is

$$\mathrm{SG_L}\,[s, b] \triangleq (s^{(L-1)}, s^{(L-2)}, \ldots, s^{(1)}, s^{(1)} \oplus \mathrm{GM}\,[b]) \tag{5.9}$$

so that

$$\mathrm{SG_L}\,[s_n, b_n] = \mathrm{SG_L}\left[\underline{d}_{n-L+1}^{n-1}, b_n\right] = (d_{n-L+1}, d_{n-L+2}, \ldots, d_{n-1}, d_{n-1} \oplus \mathrm{GM}\,[b_n]) = \underline{d}_{n-L+1}^{n}\ . \tag{5.10}$$

Although implementation of the APP algorithm with this FSM model is straightforward, as described in Appendix A, a more detailed derivation of the algorithm is presented here to exploit overlap with the inner decoder for noncoherent detection described in the following subsection.

The APP's can be expressed as

$$\Pr\left\{b_n = b \,\middle|\, \underline{r}, \{h_{n,k}\}\right\} = \sum_{s \in \mathcal{S}_\mathrm{L}} \Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}, \{h_{n,k}\}\right\}\ , \tag{5.11}$$

which can be manipulated to[2]

$$\begin{aligned}\Pr\left\{b_n = b \,\middle|\, \underline{r}\right\} &= \sum_{s \in \mathcal{S}_\mathrm{L}} \Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^{n-1}\right\} \frac{f\left(\underline{r}_n^N \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right)}{f\left(\underline{r}_n^N \,\middle|\, \underline{r}_0^{n-1}\right)} \\ &= \sum_{s \in \mathcal{S}_\mathrm{L}} \Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^{n-1}\right\} \frac{f\left(r_n \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right)}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} \\ &\qquad \times \frac{f\left(\underline{r}_{n+1}^N \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n}\right)}{f\left(\underline{r}_{n+1}^N \,\middle|\, \underline{r}_0^{n}\right)}\ .\end{aligned} \tag{5.12}$$

[2] For notational convenience, explicit reference to the conditioning on the discrete-time channel impulse response, $\{h_{n,k}\}$, which is provided as side information, has been dropped in the following.

Each of the three multiplied terms in this expression can be simplified somewhat. Since b_n and $s_n \triangleq \underline{d}_{n-L+1}^{n-1}$ are independent,

$$\begin{aligned}\Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^{n-1}\right\} &= \Pr\left\{b_n = b \,\middle|\, s_n = s, \underline{r}_0^{n-1}\right\} \Pr\left\{s_n = s \,\middle|\, \underline{r}_0^{n-1}\right\} \\ &= \Pr\{b_n = b\} \Pr\left\{s_n = s \,\middle|\, \underline{r}_0^{n-1}\right\} .\end{aligned} \tag{5.13}$$

Since $\underline{d}_{n-L+1}^{n}$ is uniquely specified by $\underline{d}_{n-L+1}^{n-1}$ and b_n,

$$\begin{aligned}f\left(r_n \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right) &= f\left(r_n \,\middle|\, \underline{d}_{n-L+1}^{n-1} = s, b_n = b, \underline{r}_0^{n-1}\right) \\ &= f\left(r_n \,\middle|\, \underline{d}_{n-L+1}^{n} = \mathrm{SG_L}\left[s, b\right], \underline{r}_0^{n-1}\right) ,\end{aligned} \tag{5.14}$$

where $\mathrm{SG_L}\left[\bullet, \bullet\right]$ is the symbol generation matrix. For the third term in Eq. (5.12),

$$\begin{aligned}f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n}\right) &= f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{d}_{n-L+1}^{n-1} = s, b_n = b, \underline{r}_0^{n}\right) \\ &= f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{d}_{n-L+1}^{n} = \mathrm{SG_L}\left[s, b\right], \underline{r}_0^{n}\right) .\end{aligned} \tag{5.15}$$

When coherent detection is used, $\underline{r}_{n+1}^{N}$ depends only on the data symbols $\underline{d}_{n-L+2}^{n}$, and not on $\underline{d}_0^{n-L+1}$. Therefore[3],

$$\begin{aligned}f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n}\right) &= f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{d}_{n-L+2}^{n} = \mathrm{ST_L}\left[s, b\right], \underline{r}_0^{n}\right) \\ &= f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_{n+1} = \mathrm{ST_L}\left[s, b\right], \underline{r}_0^{n}\right) .\end{aligned} \tag{5.16}$$

By substituting Eqs. (5.13), (5.14), and (5.16) into Eq. (5.12), the APP's can be expressed as

$$\begin{aligned}\Pr\left\{b_n = b \,\middle|\, \underline{r}\right\} = \sum_{s \in \mathcal{S}_L} \Pr\{b_n = b\} \Pr\left\{s_n = s \,\middle|\, \underline{r}_0^{n-1}\right\} &\frac{f\left(r_n \,\middle|\, \underline{d}_{n-L+1}^{n} = \mathrm{SG_L}\left[s, b\right], \underline{r}_0^{n-1}\right)}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} \\ &\times \frac{f\left(\underline{r}_{n+1}^{N} \,\middle|\, s_{n+1} = \mathrm{ST_L}\left[s, b\right], \underline{r}_0^{n}\right)}{f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{r}_0^{n}\right)} .\end{aligned} \tag{5.17}$$

---

[3] When coherent detection is used,

$$f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{d}_{n-L+1}^{n} = \tilde{\underline{d}}_{n-L+1}^{n}, \underline{r}_0^{n}\right) = f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{d}_{n-L+2}^{n} = \tilde{\underline{d}}_{n-L+2}^{n}, \underline{r}_0^{n}\right) .$$

As is discussed in the following subsection, when noncoherent detection is used, this equality does not hold.

By using the standard definitions for the APP algorithm as described in Appendix A, notably

$$\Omega_n \triangleq f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right) , \tag{5.18}$$

$$\alpha_n(s) \triangleq \Pr\left\{s_n = s \,\middle|\, \underline{r}_0^{n-1}\right\} , \tag{5.19}$$

$$\mu_n(\underline{\tilde{d}}_{n-L+1}^n) \triangleq f\left(r_n \,\middle|\, \underline{d}_{n-L+1}^n = \underline{\tilde{d}}_{n-L+1}^n, \underline{r}_0^{n-1}\right) , \tag{5.20}$$

and

$$\beta_{n+1}(s) \triangleq \frac{f\left(\underline{r}_{n+1}^N \,\middle|\, s_{n+1} = s, \underline{r}_0^n\right)}{f\left(\underline{r}_{n+1}^N \,\middle|\, \underline{r}_0^n\right)} , \tag{5.21}$$

Eq. (5.17) can be rewritten as

$$\Pr\left\{b_n = b \,\middle|\, \underline{r}\right\} = \frac{1}{\Omega_n}\Pr\left\{b_n = b\right\} \sum_{s\in\mathcal{S}_\mathrm{L}} \alpha_n(s)\mu_n(\mathrm{SG_L}\left[s,b\right])\beta_{n+1}(\mathrm{ST_L}\left[s,b\right]) . \tag{5.22}$$

The quantities $\alpha_n(\bullet)$ and $\beta_n(\bullet)$ are calculated recursively by the APP algorithm. Note that, for $s' \in \mathcal{S}_\mathrm{L}$,

$$\begin{aligned}
\alpha_{n+1}(s') &\triangleq \Pr\left\{s_{n+1} = s' \,\middle|\, \underline{r}_0^n\right\} \\
&= \sum_{s\in\mathcal{S}_\mathrm{L}} \sum_{b\in\mathcal{M}} \Pr\left\{s_n = s, b_n = b, s_{n+1} = s' \,\middle|\, \underline{r}_0^n\right\} \\
&= \sum_{s\in\mathcal{S}_\mathrm{L}} \sum_{b\in\mathcal{M}} \Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^n\right\} \Pr\left\{s_{n+1} = s' \,\middle|\, s_n = s, b_n = b, \underline{r}_0^n\right\} \\
&= \sum_{s\in\mathcal{S}_\mathrm{L}} \sum_{b\in\mathcal{M}} \Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^n\right\} \Pr\left\{s_{n+1} = s' \,\middle|\, s_{n+1} = \mathrm{ST_L}\left[s,b\right]\right\} ,
\end{aligned} \tag{5.23}$$

where the last line follows from the fact that, together, s_n and b_n specify s_{n+1}. Also, Bayes' rule gives

$$\Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^n\right\} = \Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^{n-1}\right\} \frac{f\left(r_n \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right)}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} . \tag{5.24}$$

Applying Eqs. (5.13) and (5.14) leads to

$$\Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^n\right\} = \Pr\left\{b_n = b\right\} \Pr\left\{s_n = s \,\middle|\, \underline{r}_0^{n-1}\right\} \frac{f\left(r_n \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right)}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} , \tag{5.25}$$

or, using the notation given above,

$$\Pr\left\{s_n = s, b_n = b \,\middle|\, \underline{r}_0^n\right\} = \Pr\left\{b_n = b\right\} \alpha_n(s) \frac{\mu_n(\mathrm{SG_L}\left[s, b\right])}{\Omega_n} \,. \tag{5.26}$$

Therefore Eq. (5.23) can be rewritten as

$$\begin{aligned} \alpha_{n+1}(s') = \frac{1}{\Omega_n} \sum_{s \in \mathcal{S}_\mathrm{L}} \sum_{b \in \mathcal{M}} \Pr\left\{b_n = b\right\} \alpha_n(s) \mu_n(\mathrm{SG_L}\left[s, b\right]) \\ \times \Pr\left\{s_{n+1} = s' \,\middle|\, s_{n+1} = \mathrm{ST_L}\left[s, b\right]\right\} \,. \end{aligned} \tag{5.27}$$

To calculate $\beta_n(\bullet)$, a reverse recursion can be used. For $s \in \mathcal{S}_\mathrm{L}$ note that

$$\begin{aligned} \beta_n(s) &\triangleq \frac{f\left(\underline{r}_n^N \,\middle|\, s_n = s, \underline{r}_0^{n-1}\right)}{f\left(\underline{r}_n^N \,\middle|\, \underline{r}_0^{n-1}\right)} \\ &= \sum_{b \in \mathcal{M}} \Pr\left\{b_n = b \,\middle|\, s_n = s, \underline{r}_0^{n-1}\right\} \frac{f\left(\underline{r}_n^N \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right)}{f\left(\underline{r}_n^N \,\middle|\, \underline{r}_0^{n-1}\right)} \\ &= \sum_{b \in \mathcal{M}} \Pr\left\{b_n = b\right\} \frac{f\left(r_n \,\middle|\, s_n = s, b_n = b, \underline{r}_0^{n-1}\right)}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} \\ &\qquad \times \frac{f\left(\underline{r}_{n+1}^N \,\middle|\, s_n = s, b_n = b, \underline{r}_0^n\right)}{f\left(\underline{r}_{n+1}^N \,\middle|\, \underline{r}_0^n\right)} \,. \end{aligned} \tag{5.28}$$

Using Eq. (5.16) and the notation given above,

$$\beta_n(s) = \frac{1}{\Omega_n} \sum_{b \in \mathcal{M}} \Pr\left\{b_n = b\right\} \mu_n(\mathrm{SG_L}\left[s, b\right]) \beta_{n+1}(\mathrm{ST_L}\left[s, b\right]) \,. \tag{5.29}$$

Eqs. (5.22), (5.27), and (5.29) provide a recursive algorithm for computing the APP's. When coherent detection is used the algorithm is optimal, provided that the branch metrics, $\mu_\bullet(\bullet)$, are given by

$$\begin{aligned} \mu_n(\underline{\tilde{d}}_{n-L+1}^n) &\triangleq f\left(r_n \,\middle|\, \underline{d}_{n-L+1}^n = \underline{\tilde{d}}_{n-L+1}^n, \{h_{n,k}\}\right) \\ &= \frac{1}{\pi \mathcal{N}_0} \exp\left\{-\frac{1}{\mathcal{N}_0} \left| r_n - \sum_{k=0}^{L-1} \sqrt{\mathcal{E}_s} \exp\left\{j \frac{2\pi}{M} \tilde{d}_{n-k}\right\} h_{n,k} \right|^2 \right\} \,, \end{aligned} \tag{5.30}$$

since the impulse response is known and the noise process has a Gaussian distribution. When noncoherent detection is used, calculation of the branch metrics is much more difficult, as described in the following.

5.3.2 Noncoherent Detection

When noncoherent detection is used, the inner decoder must estimate the channel impulse response while calculating the APP's. This is not an easy task since only $(N+1)$ received samples are available from which the $(N+1) \times L$ elements of $\{h_{n,k}\}$ must be estimated. Therefore the approach taken here is radically different from the approaches taken for the AWGN and frequency-flat fading channels. To reduce decoder complexity, the other two noncoherent decoders used the MDD concept. Limiting the size of the observation window over which the channel was estimated led to an automatic reduction in the number of decoder states, but this came at the expense of less than ideal channel estimation. In the approach taken here, all the previously received samples are used to estimate the channel, and a path pruning technique is used to limit the decoder complexity. In particular, Kalman filters are used to estimate the impulse response, but because the estimate is dependent on the hypothesis, only a few hypotheses are allowed to survive at any time.

The inner decoder for noncoherent detection is implemented using the APP algorithm as described in the previous subsection, using the same FSM model. The only difference is in the branch metrics that are used. To ensure accuracy when estimating the channel impulse response, it is desirable to make use of all the previously received samples. However, accurate channel estimation also requires knowledge of the transmitted symbols. In the following, one reasonable effective method is proposed for calculating the branch metrics when noncoherent detection is used.

The organization of this subsection is as follows. *Partial path metrics*, which are the likelihood of receiving sample r_n given all the previously received samples and conditioned on one particular hypothesis, $\underline{v} = \underline{\tilde{v}}$, are first described. If the channel impulse response can be represented by a state model, then a Kalman filter can be used to evaluate the partial path metrics. A description of the state model is followed by a presentation of

the governing equations of the Kalman filter. An ideal, albeit computationally infeasible, method of calculating the branch metrics from the partial path metrics is then given. This is followed by a discussion on *path pruning*, whereby the number of partial path metrics that need to be evaluated is curtailed, leading to a more practical algorithm for branch metric calculation.

Partial Path Metrics

Because the impulse response and the noise have Gaussian distributions, the distribution of sample r_n conditioned on the hypothesis $\underline{v} = \underline{\tilde{v}}$ and the previous received samples is also Gaussian. The conditional pdf is given by

$$f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right) = \frac{1}{2\pi\sigma_n^2} \exp\left\{-\frac{1}{2\sigma_n^2}\left|r_n - \overline{r}_n\right|^2\right\} , \tag{5.31}$$

where

$$\overline{r}_n \triangleq \mathbf{E}\left[r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right] \tag{5.32}$$

is the conditional mean of r_n and

$$\sigma_n^2 \triangleq \frac{1}{2}\mathbf{E}\left[\left|r_n - \overline{r}_n\right|^2 \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right] \tag{5.33}$$

is the conditional variance. From Eq. (5.3), the conditional mean can be expressed as

$$\overline{r}_n = \sum_{k=0}^{L-1} \tilde{v}_{n-k}\widehat{h}_{n,k} , \tag{5.34}$$

where $\widehat{h}_{n,k} = \mathbf{E}\left[h_{n,k} \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right]$ is an estimate of the impulse response at time index n. Thus, the conditional pdf can be rewritten as

$$f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right) = \frac{1}{2\pi\sigma_n^2} \exp\left\{-\frac{1}{2\sigma_n^2}\left|r_n - \sum_{k=0}^{L-1} \tilde{v}_{n-k}\widehat{h}_{n,k}\right|^2\right\} . \tag{5.35}$$

The pdf depends directly on $\underline{\tilde{v}}_{n-L+1}^{n}$, and also indirectly on $\underline{\tilde{v}}_0^{n-L}$ and $\underline{r}_0^{n-1}$ through the estimates and the variance.

The estimates could be found by solving the appropriate Wiener-Hopf equation, but since this involves inverting a $(n \times n)$ matrix, this approach is infeasible for large n. However, if the impulse response can be represented by a state model, a Kalman filter can be used to calculate the estimates, and the variance, in a less cumbersome manner.

State Model for the Impulse Response

Kalman filtering can be used for channel estimation only if the impulse response can be represented by, or at least well-approximated by, a state model. In practice, this is usually the case. Let

$$\underline{h}_n \triangleq [h_{n,0}\ h_{n,1}\ \cdots\ h_{n,L-1}]^T \tag{5.36}$$

be a column vector denoting the impulse response at time index n. With a state model of order ρ the impulse response is a function of the *state vector*, $\underline{x}_n$, of length ρL. The impulse response is related to the state vector by

$$\underline{h}_n = \underline{\underline{G}}\ \underline{x}_n\ , \tag{5.37}$$

where $\underline{\underline{G}}$ is the *linear connection matrix*, which is of size $(L \times \rho L)$. The state vector varies over time, driven by stationary white Gaussian noise, but the state at time index n is related to the state at time index $n-1$ by

$$\underline{x}_n = \underline{\underline{F}}\ \underline{x}_{n-1} + \underline{q}_n\ , \tag{5.38}$$

where $\underline{\underline{F}}$ is the *state transition matrix*[4] of size $(\rho L \times \rho L)$, and $\underline{q}_n$ is the driving noise vector (of length ρL) at time index n. Because the noise is white and stationary, its cross-correlation function has the form

$$\frac{1}{2}\mathbf{E}\left[\underline{q}_{n+m}\underline{q}_n^H\right] = \underline{\underline{Q}}\delta_m\ , \tag{5.39}$$

[4] This matrix is not to be confused with the state transition matrices of the finite-state machines used elsewhere in this thesis. In this case the number of states is not finite.

where the superscript H denotes conjugate transpose, and $\underline{\underline{Q}}$ is the $(\rho L \times \rho L)$ covariance matrix for $\underline{q}_n$. In general, the covariance matrix can be expressed as $\underline{\underline{Q}} = \underline{\underline{\Lambda}}' - \underline{\underline{F}}\,\underline{\underline{\Lambda}}'\,\underline{\underline{F}}^H$, for some matrix $\underline{\underline{\Lambda}}'$ [37].

The matrices $\underline{\underline{F}}$, $\underline{\underline{G}}$, and $\underline{\underline{\Lambda}}'$ are used to define the state model, and different impulse response models can be supported by selecting these matrices appropriately. With the cross-correlation function of the impulse response specified by Eq. (5.4), where $\phi_m^{(F)}$ takes one of the forms specified in Chapter 4, the state model is readily derived. When the first-order Butterworth (exponential) fading model is used to specify $\phi_m^{(F)}$, with

$$\phi_m^{(F)} = \frac{1}{2}\exp\left\{-2\pi B_d T|m|\right\} , \tag{5.40}$$

the state model is defined by

$$\underline{\underline{F}} = \exp\left\{-2\pi B_d T\right\}\underline{\underline{I}} , \qquad \underline{\underline{G}} = \underline{\underline{I}} , \qquad \text{and} \qquad \underline{\underline{\Lambda}}' = \frac{1}{2}\underline{\underline{\Lambda}} , \tag{5.41}$$

where $\underline{\underline{I}}$ is the $(L \times L)$ identity matrix, and

$$\underline{\underline{\Lambda}} \triangleq \begin{bmatrix} \lambda_{0,0} & \lambda_{0,1} & \cdots & \lambda_{0,L-1} \\ \lambda_{1,0} & \lambda_{1,1} & \cdots & \lambda_{1,L-1} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{L-1,0} & \lambda_{L-1,1} & \cdots & \lambda_{L-1,L-1} \end{bmatrix} , \tag{5.42}$$

with $\lambda_{k,l}$ given by Eq. (5.5). For the second-order Butterworth fading model, with

$$\phi_m^{(F)} = \frac{1}{2}\exp\left\{-B_d' T|m|\right\}\left[\cos\left(B_d' T|m|\right) + \sin\left(B_d' T|m|\right)\right] \tag{5.43}$$

where $B_d' = \frac{1}{\sqrt{2}}\pi B_d$, the state model is defined by

$$\underline{\underline{F}} = \exp\left\{-B_d' T\right\}\begin{bmatrix} \left(\cos B_d' T + \sin B_d' T\right)\underline{\underline{I}} & \left(\frac{1}{B_d'}\sin B_d' T\right)\underline{\underline{I}} \\ \left(-2B_d' \sin B_d' T\right)\underline{\underline{I}} & \left(\cos B_d' T - \sin B_d' T\right)\underline{\underline{I}} \end{bmatrix} , \tag{5.44}$$

$$\underline{\underline{G}} = \begin{bmatrix} \underline{\underline{I}} & \underline{\underline{0}} \end{bmatrix} , \tag{5.45}$$

and

$$\underline{\underline{\Lambda}}' = \begin{bmatrix} \frac{1}{2}\underline{\underline{\Lambda}} & \underline{\underline{0}} \\ \underline{\underline{0}} & (B_d')^2\,\underline{\underline{\Lambda}} \end{bmatrix}, \tag{5.46}$$

where $\underline{\underline{0}}$ is the $(L \times L)$ zero matrix. Note that the land-mobile fading model cannot be represented directly with a state model, but approximations of any order, ρ, can be found.

Kalman Filtering

A Kalman filter can be used to estimate the channel impulse response if the impulse response can be represented with a state model. But since the estimate is a function of the hypothesis, a separate Kalman filter is needed for each hypothesis. As each sample is received it is passed to the Kalman filter, and after sample r_{n-1} has been received, $\underline{\widehat{h}} \triangleq [\widehat{h}_{n,0}\ \widehat{h}_{n,1}\ \cdots\ \widehat{h}_{n,L-1}]^T$ and σ_n^2 are produced by the filter. As shown in [37], these are calculated with

$$\underline{\widehat{h}}_n = \underline{\underline{G}}\,\underline{\widehat{x}}_n \tag{5.47}$$

and

$$\sigma_n^2 = \frac{\mathcal{N}_0}{2} + \underline{\widetilde{v}}_n\,\underline{\underline{G}}\,\underline{\underline{\Gamma}}_n\,\underline{\underline{G}}^H\,\underline{\widetilde{v}}_n^H\,, \tag{5.48}$$

where $\underline{\widehat{x}}_n$ is an estimate of the state vector, $\underline{\underline{G}}$ is the linear connection matrix from the state model, $\underline{\underline{\Gamma}}_n$ is the internal *Kalman state matrix*, of size $(\rho L \times \rho L)$, and $\underline{\widetilde{v}}_n \triangleq [\widetilde{v}_n\,\widetilde{v}_{n-1}\,\cdots\,\widetilde{v}_{n-L+1}]$ is a row vector containing the last L hypothetical symbols. When sample r_n is received the filter is updated by first calculating the *Kalman gain vector*, $\underline{K}_n$, using

$$\underline{K}_n = \frac{1}{\sigma_n^2}\underline{\underline{F}}\,\underline{\underline{\Gamma}}_n\,\underline{\underline{G}}^H\,\underline{\widetilde{v}}_n^H\,, \tag{5.49}$$

where $\underline{\underline{F}}$ is the state transition matrix from the state model. The estimate of the state vector and the Kalman state matrix are then updated using

$$\underline{\widehat{x}}_{n+1} = \underline{\underline{F}}\,\underline{\widehat{x}}_n + \underline{K}_n\,(r_n - \overline{r}_n) \tag{5.50}$$

where $\overline{r}_n = \underline{\tilde{v}}_n \, \underline{\hat{h}}_n$, and

$$\underline{\underline{\Gamma}}_{n+1} = \underline{\underline{Q}} + \left(\underline{\underline{F}} - \underline{K}_n \, \underline{\tilde{v}}_n \, \underline{\underline{G}}\right) \underline{\underline{\Gamma}}_n \, \underline{\underline{F}}^H \, , \tag{5.51}$$

where $\underline{\underline{Q}}$ is the covariance matrix of the driving noise of the state model. The initial condition of the Kalman filter, before r_0 has been received, is

$$\underline{\hat{x}}_0 = [0 \; 0 \; \cdots \; 0]^T \qquad (\rho L \text{ zeros}) \tag{5.52}$$

and

$$\underline{\underline{\Gamma}}_0 = \underline{\underline{\Lambda}}' \tag{5.53}$$

where $\underline{\underline{\Lambda}}'$ is specified with the state model.

Using these equations, a Kalman filter is able to make use of all the previously received samples when estimating the impulse response. Provided that the impulse response is accurately represented by the state model, the estimates will be ideal in that they minimize the mean squared prediction error. However, the estimates also depend on the hypothesis $\underline{v}_0^{n-1} = \underline{\tilde{v}}_0^{n-1}$, or equivalently, on $\underline{d}_0^{n-1} = \underline{\tilde{d}}_0^{n-1}$. It is useful to define $p_n \triangleq \underline{d}_0^{n-1}$ as the initial path segment up to time index n. Also, define $\mathcal{P}_n$ as the set of all possible initial path segments up to time index n, so $\mathcal{P}_n$ is the set of all M-ary n-tuples. For the given received samples, $\underline{r}_0^{n-1}$, there are M^n different possible estimates of the impulse response, one for each hypothetical path segment in $\mathcal{P}_n$.

For any initial path segment $p \in \mathcal{P}_n$ and data symbol $b \in \mathcal{M}$, a Kalman filter can be used to calculate the partial path metric

$$\begin{aligned} \mu_n'(p,b) &\triangleq f\left(r_n \;\middle|\; p_n = p, b_n = b, \underline{r}_0^{n-1}\right) \\ &= f\left(r_n \;\middle|\; \underline{d}_0^n = \mathrm{PE}_n\left[p,b\right], \underline{r}_0^{n-1}\right) \, , \end{aligned} \tag{5.54}$$

where $\mathrm{PE}_n\left[p,b\right]$ is the *path extension function*, which specifies the path segment created by appending symbol b onto path p. For any $b \in \mathcal{M}$ and $p \in \mathcal{P}_n$, with $p = (p^{(0)}, p^{(1)}, \ldots, p^{(n-1)})$,

this function is defined as

$$\mathrm{PE}_n\left[p, b\right] \triangleq \left(p^{(0)}, p^{(1)}, \ldots, p^{(n-1)}, p^{(n-1)} \oplus \mathrm{GM}\left[b\right]\right) \tag{5.55}$$

so

$$\mathrm{PE}_n\left[p_n, b_n\right] = \mathrm{PE}_n\left[\underline{d}_0^{n-1}, b_n\right] = \left(d_0, d_1, \ldots, d_{n-1}, d_{n-1} \oplus \mathrm{GM}\left[b_n\right]\right) = \underline{d}_0^n = p_{n+1} \ . \tag{5.56}$$

Calculation of the Branch Metrics

One possible method of calculating the branch metrics from the partial path metrics involves summing over all possible initial path segments. Although computationally infeasible because of the large number of path segments, this approach is useful to illustrate how a sub-optimal, but tractable, solution can be found.

To calculate the branch metrics from the partial path metrics, note that

$$\begin{aligned} \mu_n(\mathrm{SG_L}\left[s, b\right]) &= f\left(r_n \;\middle|\; s_n = s, b_n = b, \underline{r}_0^{n-1}\right) \qquad (5.57) \\ &= \sum_{p \in \mathcal{P}_n(s)} f\left(r_n \middle| p_n = p, s_n = s, b_n = b, \underline{r}_0^{n-1}\right) \Pr\left\{p_n = p \middle| s_n = s, b_n = b, \underline{r}_0^{n-1}\right\} \ , \end{aligned}$$

where $\mathcal{P}_n(s)$ is the subset of $\mathcal{P}_n$ containing only those path segments that enter state s at time index n. Note that $\bigcup_{s \in \mathcal{S}_\mathrm{L}} \mathcal{P}_n(s) = \mathcal{P}_n$, and $\mathcal{P}_n(k) \bigcap \mathcal{P}_n(l) = \emptyset$ for any $k \neq l$, $k, l \in \mathcal{S}_\mathrm{L}$. This definition of $\mathcal{P}_n(s)$ implies that

$$f\left(r_n \;\middle|\; p_n = p, s_n = s, b_n = b, \underline{r}_0^{n-1}\right) = f\left(r_n \;\middle|\; p_n = p, b_n = b, \underline{r}_0^{n-1}\right) \tag{5.58}$$

for all $p \in \mathcal{P}_n(s)$. Furthermore, since $p_n \triangleq \underline{d}_0^{n-1}$ does not depend on b_n, Eq. (5.57) can be expressed as

$$\mu_n(\mathrm{SG_L}\left[s, b\right]) = \sum_{p \in \mathcal{P}_n(s)} f\left(r_n \;\middle|\; p_n = p, b_n = b, \underline{r}_0^{n-1}\right) \Pr\left\{p_n = p \;\middle|\; s_n = s, \underline{r}_0^{n-1}\right\} \ . \tag{5.59}$$

Using Bayes' rule gives

$$\mu_n(\mathrm{SG_L}\left[s, b\right]) = \sum_{p \in \mathcal{P}_n(s)} f\left(r_n \;\middle|\; p_n = p, b_n = b, \underline{r}_0^{n-1}\right) \frac{\Pr\left\{p_n = p, s_n = s \;\middle|\; \underline{r}_0^{n-1}\right\}}{\Pr\left\{s_n \;\middle|\; \underline{r}_0^{n-1}\right\}} \ , \tag{5.60}$$

and, since $\Pr\left\{p_n = p, s_n = s \,\middle|\, \underline{r}_0^{n-1}\right\} = \Pr\left\{p_n = p \,\middle|\, \underline{r}_0^{n-1}\right\}$ for all $p \in \mathcal{P}_n(s)$,

$$\mu_n(\mathrm{SG_L}\,[s,b]) = \sum_{p \in \mathcal{P}_n(s)} f\left(r_n \,\middle|\, p_n = p, b_n = b, \underline{r}_0^{n-1}\right) \frac{\Pr\left\{p_n = p \,\middle|\, \underline{r}_0^{n-1}\right\}}{\Pr\left\{s_n \,\middle|\, \underline{r}_0^{n-1}\right\}} . \tag{5.61}$$

By defining

$$\gamma_n(p) \triangleq \Pr\left\{p_n = p \,\middle|\, \underline{r}_0^{n-1}\right\} , \tag{5.62}$$

and using the definition of the partial path metric $\mu'_n(\bullet)$ given by Eq. (5.54), the branch metrics can be expressed as

$$\mu_n(\mathrm{SG_L}\,[s,b]) = \sum_{p \in \mathcal{P}_n(s)} \mu'_n(p,b) \frac{\gamma_n(p)}{\alpha_n(s)} , \tag{5.63}$$

where $\alpha_n(\bullet)$ is defined in Eq. (5.19). Calculation of $\gamma_n(\bullet)$ is straightforward and can be done during the forward recursion of the APP algorithm. Suppose that $\gamma_n(p)$ has been calculated for all $p \in \mathcal{P}_n$. Then, $\gamma_{n+1}(\mathrm{PE}_n\,[p,b])$ can be calculated for the extension of path p with each $b \in \mathcal{M}$. Note that

$$\begin{aligned}
\gamma_{n+1}(\mathrm{PE}_n\,[p,b]) &= \Pr\left\{p_n = p, b_n = b \,\middle|\, \underline{r}_0^{n}\right\} \\
&= \frac{f\left(r_n \,\middle|\, p_n = p, b_n = b, \underline{r}_0^{n-1}\right)}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} \Pr\left\{p_n = p, b_n = b \,\middle|\, \underline{r}_0^{n-1}\right\} \\
&= \frac{f\left(r_n \,\middle|\, p_n = p, b_n = b, \underline{r}_0^{n-1}\right)}{f\left(r_n \,\middle|\, \underline{r}_0^{n-1}\right)} \Pr\left\{b_n = b\right\} \Pr\left\{p_n = p \,\middle|\, \underline{r}_0^{n-1}\right\} \\
&= \frac{1}{\Omega_n} \Pr\left\{b_n = b\right\} \mu'_n(p,b)\gamma_n(p) ,
\end{aligned} \tag{5.64}$$

where Ω_n is defined in Eq. (5.18). In this manner $\gamma_{n+1}(p')$ can be calculated for all $p' \in \mathcal{P}_{n+1}$.

The obvious problem with using this approach to calculate the branch metrics is that the summation in Eq. (5.63) is over all path segments in $\mathcal{P}_n(s)$. Since there are M^{n-L+1} different possible path segments up time index n which terminate in state s, the number of terms in the summation grows exponentially with n. To limit the complexity of the decoder, some means of path pruning must be employed. That is, paths in $\mathcal{P}_n(s)$ which contribute little to the branch metrics should be discarded.

Path Pruning

To illustrate the proposed pruning technique used for this decoder, note that some path segments are unlikely to have been transmitted given the received samples. That is, $\gamma_n(p) \triangleq \Pr\left\{p_n = p \,\middle|\, \underline{r}_0^{n-1}\right\}$ will be quite small for some $p \in \mathcal{P}_n(s)$. These path segments will contribute little to the branch metrics. Conversely, for some $p \in \mathcal{P}_n(s)$, $\gamma_n(p)$ will be quite large, and these paths will dominate the calculation of the branch metrics. Let $\mathcal{P}'_n(s)$ be the subset of $\mathcal{P}_n(s)$ containing those path segments for which $\gamma_n(\bullet)$ is large. Then, from Eq. (5.63),

$$\mu_n(\mathrm{SG_L}\,[s,b]) \cong \sum_{p \in \mathcal{P}'_n(s)} \mu'_n(p,b) \frac{\gamma_n(p)}{\alpha_n(s)}\,, \tag{5.65}$$

The path segments in $\mathcal{P}'_n(s)$ are referred to as the survivors entering state s at time index n, and $\mathcal{P}'_n(s)$ is the survivor set for state s. Let P denote the number of survivors for each state at any time. Although keeping a large number of survivors is desirable to accurately determine the branch metrics, doing so requires considerable decoder complexity since one Kalman filter is required for each survivor.

Maintaining the survivor sets is fairly straightforward, and is done during the forward recursion of the APP algorithm. Let $\mathcal{P}'_n = \bigcup_{s \in \mathcal{S}_\mathrm{L}} \mathcal{P}'_n(s)$ contain all the survivor path segments at time index n for all states. Since there are M^{L-1} states in the state space, $\mathcal{S}_\mathrm{L}$, there are a total of PM^{L-1} survivors. Suppose, at time index n, that $\mathcal{P}'_n$ has been determined, and that $\gamma_n(p)$ has been calculated for all $p \in \mathcal{P}'_n$. Then, each of the PM^{L-1} paths in $\mathcal{P}'_n$ is extended along each branch $b \in \mathcal{M}$, and the corresponding value of $\gamma_{n+1}(\mathrm{PE}_n\,[p,b])$ is calculated using Eq. (5.64). Of the PM^L extended paths, a total of PM terminate in state $s \in \mathcal{S}_\mathrm{L}$ at time index $n+1$. Of these paths, only the P paths with the largest values of $\gamma_{n+1}(\bullet)$ survive, and the other $(P-1)M$ paths are pruned.

Summary

In summary, to calculate the APP's, the decoder uses the APP algorithm with the FSM model described in Section 5.3.1. A bank of Kalman filters is used, and, to limit the complexity of the decoder, path pruning is employed. At any time there are PM^{L-1} survivor paths, with P survivors terminating in each state. One Kalman filter is assigned to each survivor.

During the forward recursion at time index n it is assumed at the set of survivors, $\mathcal{P}'_n$, has been determined and $\gamma_n(p) \triangleq f\left(p_n = p \,\middle|\, \underline{r}_0^{n-1}\right)$ has been calculated for all $p \in \mathcal{P}'_n$. Furthermore, for each $p \in \mathcal{P}'_n$ one Kalman filter has calculated its Kalman state matrix, $\underline{\underline{\Gamma}}_n$, and its estimate of the state vector, $\underline{\hat{x}}_n$, for the hypothesis $\underline{d}_0^{n-1} = p$. For $n = 1, 2, \ldots, N$, the algorithm performs the following steps:

1. Each Kalman filter computes $\underline{\hat{h}}_n$, its estimate of the impulse response, using Eqs. (5.47).

2. For each $b \in \mathcal{M}$, each Kalman filter computes the conditional mean, $\overline{r}_n$, and variance, σ_n^2, using using Eqs. (5.34) and (5.48), respectively. Each filter then computes the partial path metric, $\mu'_n(p, b)$, using Eqs. (5.54) and (5.35).

3. The branch metrics are computed from $\gamma_n(\bullet)$ and the partial path metrics using Eq. (5.65). The branch metrics are saved for use in reverse recursion.

4. For all $s' \in \mathcal{S}_{\mathrm{L}}$, $\alpha_{n+1}(s')$ is computed using Eq. (5.27).

5. For all $p \in \mathcal{P}'_n$ and $b \in \mathcal{M}$, the algorithm computes $\gamma_{n+1}(\mathrm{PE}_n\,[p, b])$ using Eq. (5.64).

6. The P extended paths with the largest values of $\gamma_n(\bullet)$ entering each state are saved and stored in the survivor set $\mathcal{P}'_{n+1}$.

7. For each survivor at time index $n+1$, a Kalman filter is updated from the corresponding filter at time index n. This is accomplished by calculating $\underline{K}_n$ using Eq. (5.49), then $\underline{\hat{x}}_{n+1}$ using Eq. (5.50) and $\underline{\underline{\Gamma}}_{n+1}$ using Eq. (5.51).

Once the forward recursion completes, the reverse recursion begins, following the normal steps of the APP algorithm.

It must be pointed out that the proposed implementation of the inner decoder has a number of short-comings, so the decoder is suboptimal. As a result, there will be a noncoherence penalty. As mentioned in Section 5.3.1 the derivation of the APP algorithm is based on the assumption that

$$f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{d}_{n-L+1}^{n} = \underline{\tilde{d}}_{n-L+1}^{n}, \underline{r}_{0}^{n}\right) = f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{d}_{n-L+2}^{n} = \underline{\tilde{d}}_{n-L+2}^{n}, \underline{r}_{0}^{n}\right) . \tag{5.66}$$

which is valid for coherent detection, but not when noncoherent detection is used. This is because ideal channel estimation requires that all previously received samples be used to estimate the channel impulse response affecting the current sample. As a result, $\underline{r}_{n+1}^{N}$ depends not only on $\underline{d}_{n-L+2}^{n}$ through Eq. (5.3), but also on $\underline{d}_{0}^{n-L+1}$ through the channel estimation. However, by Bayes' rule,

$$\begin{aligned} f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{d}_{n-L+1}^{n} = \underline{\tilde{d}}_{n-L+1}^{n}, \underline{r}_{0}^{n}\right) = & \frac{\Pr\left\{d_{n-L+1} = \tilde{d}_{n-L+1} \,\middle|\, \underline{d}_{n-L+2}^{n} = \underline{\tilde{d}}_{n-L+1}^{n-1}, \underline{r}_{0}^{N}\right\}}{\Pr\left\{d_{n-L+1} = \tilde{d}_{n-L+1} \,\middle|\, \underline{d}_{n-L+2}^{n} = \underline{\tilde{d}}_{n-L+2}^{n}, \underline{r}_{0}^{n}\right\}} \\ & \times f\left(\underline{r}_{n+1}^{N} \,\middle|\, \underline{d}_{n-L+2}^{n} = \underline{\tilde{d}}_{n-L+2}^{n}, \underline{r}_{0}^{n}\right) . \end{aligned} \tag{5.67}$$

One can argue that $\underline{r}_{n+1}^{N}$ contributes little information regarding the value of d_{n-L+1} because of the time separation between when d_{n-L+1} is transmitted and when $\underline{r}_{n+1}^{N}$ is received[5]. Therefore

$$\Pr\left\{d_{n-L+1} = \tilde{d}_{n-L+1} \middle| \underline{d}_{n-L+2}^{n} = \underline{\tilde{d}}_{n-L+2}^{n}, \underline{r}_{0}^{N}\right\} \cong \Pr\left\{d_{n-L+1} = \tilde{d}_{n-L+1} \middle| \underline{d}_{n-L+2}^{n} = \underline{\tilde{d}}_{n-L+2}^{n}, \underline{r}_{0}^{n}\right\} \tag{5.68}$$

[5] Recall that a similar argument is used with the noncoherent decoders presented for the other two channel models.

and

$$f\left(\underline{r}_{n+1}^{N} \;\middle|\; \underline{d}_{n-L+1}^{n} = \underline{\tilde{d}}_{n-L+1}^{n}, \underline{r}_{0}^{n}\right) \cong f\left(\underline{r}_{n+1}^{N} \;\middle|\; \underline{d}_{n-L+2}^{n} = \underline{\tilde{d}}_{n-L+2}^{n}, \underline{r}_{0}^{n}\right) . \tag{5.69}$$

Another problem is that the method for maintaining the survivor set is somewhat weak since it supposes that if $\gamma_n(\underline{\tilde{d}}_0^{n-1})$ is small for some hypothesis $\underline{\tilde{d}}_0^{n-1}$, then $\gamma_m(\underline{\tilde{d}}_0^{m-1})$ will also be small for every continuation of $\underline{\tilde{d}}_0^{n-1}$ up to time index m, for all $m > n$. This is not necessarily the case as some hypotheses may initially appear unlikely, only to become more probable as more samples are received. A greater worry when path pruning is used is that there is a chance that the correct path (corresponding to the actually transmitted symbols) will be pruned. The Kalman filter state information for the correct path will then be irrecoverably lost. This could potentially have a serious impact on the performance of the noncoherent decoder.

5.4 Simulation Results and Related Discussion

The performance of the proposed coherent and noncoherent systems was investigated by means of computer simulation. Because of the high complexity of the decoder due to the Kalman filters, only BPSK is considered. Message words of $N_a = 508$ bits are convolutionally encoded with a 16-state, rate 1/2, convolutional code, with generator $(23, 35)_8$. The resulting code words of $N_c = 512$ symbols are interleaved with a randomly-selected bit-by-bit interleaver with a size of $N = 1024$ bits. Root raised-cosine filtering, with a rolloff parameter of $\beta = 0.5$, is used for pulse-shaping. Additional details regarding the simulations can be found in Appendix D.

5.4.1 Two-ray Exponential Fading

Initially, a simple, commonly-used, two-ray fading model is considered, with a multipath intensity profile of

$$\phi_M(\tau) = \frac{1}{2}\delta(\tau) + \frac{1}{2}\delta(\tau - T) \ . \tag{5.70}$$

With this profile, the inner decoder is implemented using

$$\begin{aligned} \lambda_{k,l} &\triangleq \int_{-\infty}^{\infty} \phi_M(\tau) h_{TR}(kT - \tau) h_{TR}^*(lT - \tau)\, d\tau \\ &= \frac{1}{2}\left[h_{TR}(kT)h_{TR}^*(lT) + h_{TR}(kT - T)h_{TR}^*(lT - T)\right] \\ &= \frac{1}{2}\left[\delta_k + \delta_{k-1}\right]\delta_{k-l} \ . \end{aligned} \tag{5.71}$$

The time variations of the impulse response are modelled with the first-order Butterworth (exponential) fading model, with

$$\phi_F(\Delta t) = \frac{1}{2}\exp\left\{-2\pi B_d|\Delta t|\right\} \ . \tag{5.72}$$

This fading model was selected because it permits the channel to be closely approximated by a first-order state model[6].

With the exponential fading model, the fading process varies relatively rapidly over time as compared to the land-mobile fading model used in the previous chapter. Computer simulation revealed that the average fade duration[7] for the exponential fading model with $B_dT = 0.001$ to be about 14.5 symbols. By contrast, the average fade duration for the land-mobile model with the same B_dT product was found to be about 350 symbols. Therefore, with exponential fading, a B_dT product of $B_dT = 0.001$ could be classified as moderately fast fading, and $B_dT = 10^{-5}$, which produces fades with an average length of 140 symbols, can be considered to be slow fading.

[6] Because of the Doppler spreading due to the fading process, the model is not exact. For low B_dT products (i.e. $B_dT < 0.05$), however, the Doppler spreading has only a very minor impact.

[7] The average fade duration is the average amount of time that elapses between when a frequency-flat fading process first drops below a certain threshold and when it next rises above that threshold. A threshold of a signal strength that is 3 dB below its mean value was used.

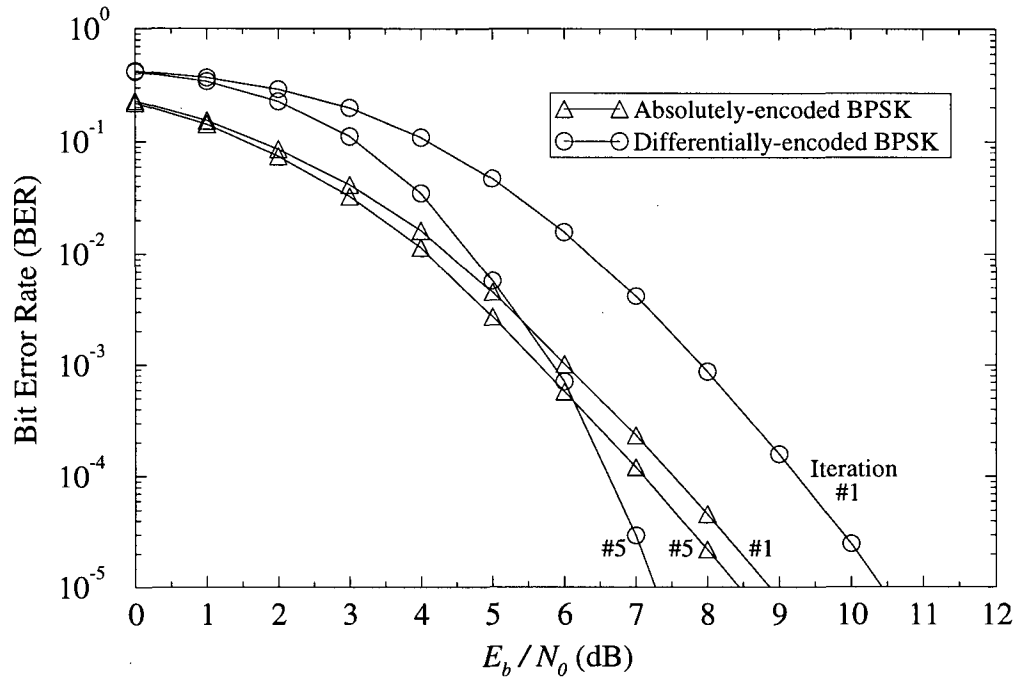


Figure 5.1: Coherent Detection: Comparison of BER performance of the 16-state convolutional code transmitted with differentially-encoded and absolutely-encoded BPSK. After the first and fifth iterations of the iterative decoders. For two-ray exponential fading with $B_dT = 0.001$.

For a fading rate of $B_dT = 0.001$, the BER performance of the proposed coherent system using differentially encoded BPSK after the first and fifth iterations of the decoder is shown in Fig. 5.1. As can be seen, there is 3 dB improvement in performance at a BER of 10^{-5}. For comparison, the performance of a coherent "turbo-equalizer" [15], after the first and fifth iterations, is also shown, using the same convolutional code but with absolutely-encoded BPSK. There is only a 1 dB performance gain for the absolutely-encoded case, and the differentially-encoded system provides performance that is over 1 dB better at a BER of 10^{-5} after five iterations. Once again, it is clear that using differential encoding can improve performance when coherent detection is used.

The BER performance of the proposed noncoherent system with $P = 1$ survivor per

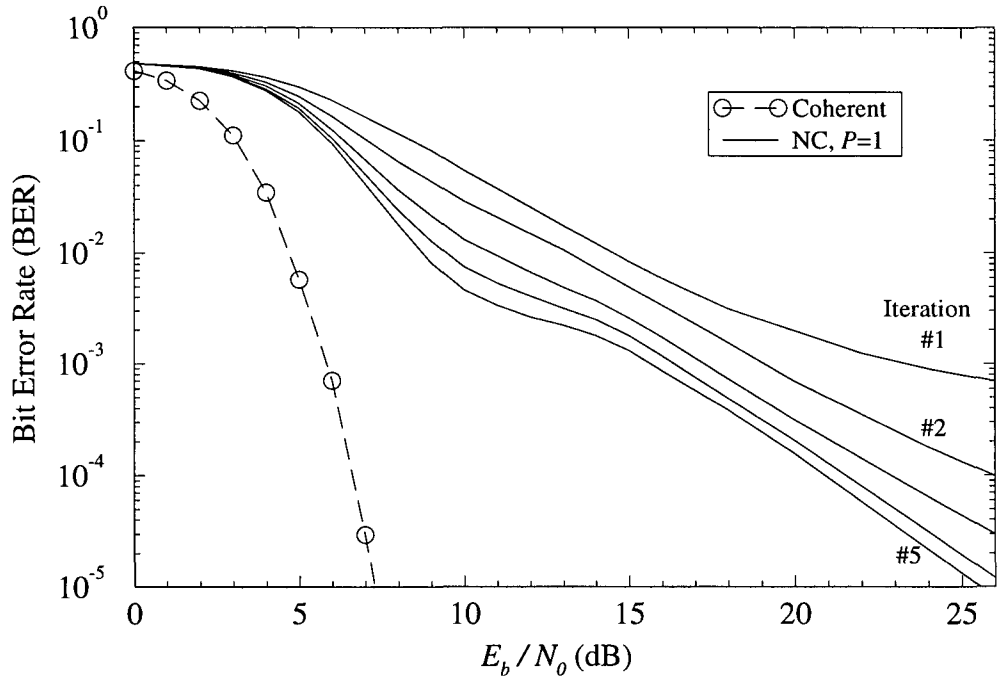


Figure 5.2: Noncoherent Detection: BER performance of the iterative decoder with noncoherent detection after each of the first five iterations. Also shown is the performance with coherent detection after the fifth iteration. For 2-ray exponential fading with $B_dT = 0.001$.

state is shown in Fig. 5.2, after each of the first five iterations. After the first iteration the performance is very poor, with a high error floor clearly visible. By using iterative decoding it is possible to lower the error floor below 10^{-5}, as well as reduce the SNR required to achieve a given BER (there is a 4 dB reduction in the SNR required for a BER of 10^{-4} between the second and fifth iterations). Nonetheless, there is a very substantial noncoherence penalty, as can be seen by making a comparison with the coherent system, the performance of which, after the fifth iteration, is also shown in Fig. 5.2. There is a noncoherence penalty of over 18 dB at a BER of 10^{-5}.

In an attempt to isolate the cause of the noncoherence penalty, an artificial decoder was devised based on the $P = 1$ inner decoder. However, the new decoder was modified

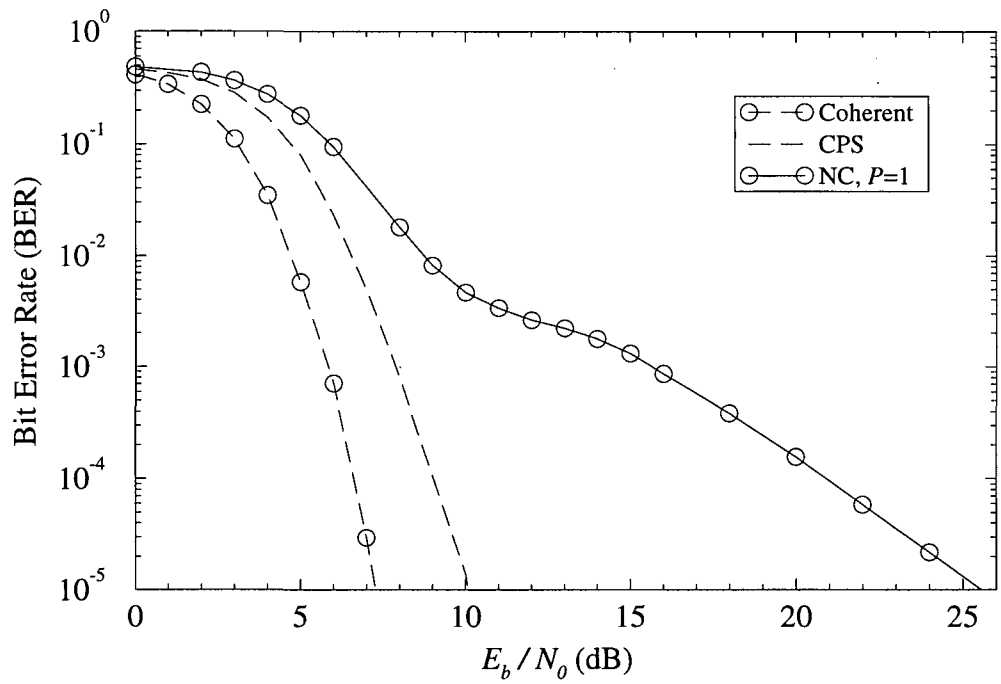


Figure 5.3: Noncoherent Detection: BER performance after the fifth iteration for the $P = 1$ noncoherent decoder, the coherent decoder, and the Correct Path Survives (CPS) decoder. For two-ray exponential fading with $B_dT = 0.001$.

to ensure that the path corresponding to the actually transmitted symbols is never pruned. More specifically, if the correct path is about to be pruned because another path entering the same state has a larger value of $\gamma_n(\bullet)$, the Kalman filter for the survivor path is updated from the Kalman filter for the path corresponding to the actual transmitted symbols[8]. This decoder, named the Correct Path Survives (CPS) decoder, is useful for distinguishing the effectiveness of the Kalman filters from the impact of pruning the correct path. In Fig. 5.3 the performance of the CPS decoder is shown, along with the performance of the coherent and noncoherent decoders. All results are for the fifth iteration. As can be seen, if the Kalman state information for the correct path is prevented from being discarded due to

[8] See step #7 in the summary given in the previous section.

path pruning, the noncoherence penalty is reduced to only 3 dB. To put it another way, the combined penalty of using only $P = 1$ path to calculate the branch metrics in Eq. (5.65) and using Kalman filtering to estimate the fading when the transmitted signal is known[9], is limited to 3 dB. The other 15 dB of the noncoherence penalty is due to the possibility that the Kalman state information for the correct path can be discarded. Analysis of the simulation results has shown that the BER of the noncoherent system is dominated by catastrophic error events. Typically, of the transmitted words that are decoded with one or more bits in error, over 50% have at least one quarter of the bits in error. Clearly, something is occurring within the noncoherent decoder to cause it to fail catastrophically. It seems likely that when the decoder prunes the correct path, a "false-lock" situation can arise, whereby the decoder becomes unreasonably convinced that its incorrect estimate of the impulse response is, in fact, highly reliable. Some specific numerical results help to support this observation. To generate the point corresponding to $\mathcal{E}_b/\mathcal{N}_0 = 13$ dB for the $P = 1$ noncoherent detector shown in Fig. 5.3, the transmission of $N_w = 100,000$ message words (50,800,000 message bits) was simulated. Out of these words, 898 words resulted in error events (at least one message bit was in error), and the correct path was pruned for 1393 message words. Therefore, pruning the correct path does not necessarily lead to an error event. However, all of the error events coincided with words in which the correct path was pruned. Although in general it cannot be claimed that error events are always caused by pruning the correct path, it seems that this is usually the cause at high SNR's.

One method to reduce the likelihood that the correct path is pruned is to increase the number of survivors per state, P. The results in Fig. 5.4 for $P = 1$, 2, 3, and 4 show that, as expected, increasing the number of survivors has a significant impact on the BER. In fact, at a BER of 10^{-5} there is almost a 10 dB gain by using $P = 4$ instead of $P = 1$. Nonetheless,

[9] and the knowledge of the transmitted signal is used strictly for the purpose of estimating the fading.

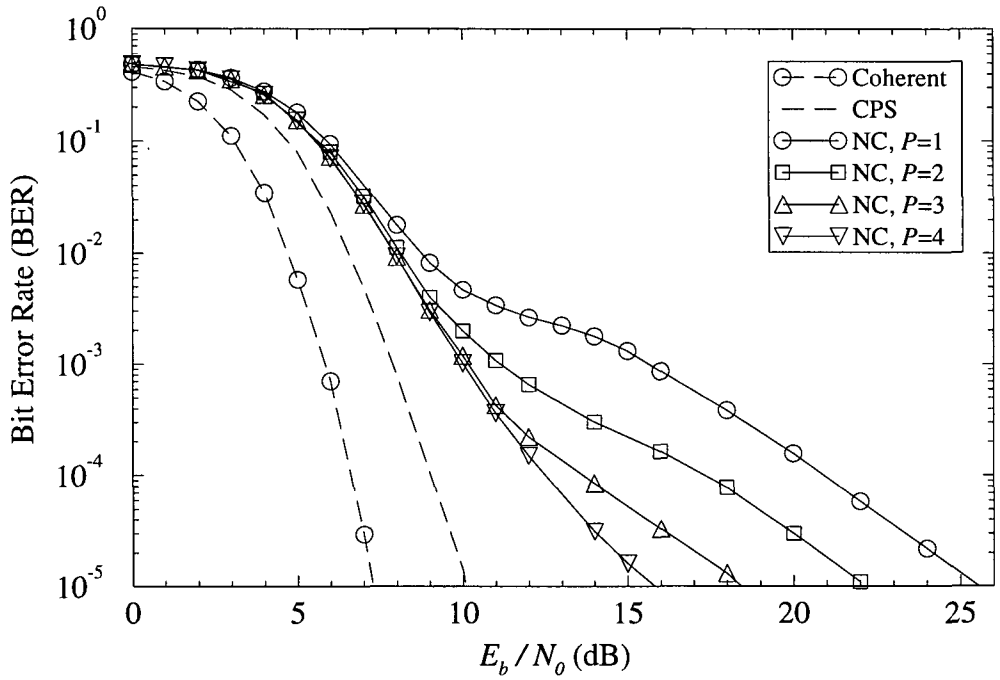


Figure 5.4: Noncoherent Detection: BER performance after the fifth iteration for the $P = 1$, 2, 3, and 4 noncoherent decoders, the coherent decoder, and the Correct Path Survives (CPS) decoder. For two-ray exponential fading with $B_d T = 0.001$.

the noncoherence penalty is still quite large (8 dB for $P = 4$ at a BER of 10^{-5}), with a 6 dB discrepancy between between the $P = 4$ noncoherent decoder and the CPS decoder. Judging by the results, especially at moderate BER's such as 10^{-3}, it appears that P would have to be much larger than four to significantly close the gap between the noncoherent and the CPS decoders. Because of the complexity of the Kalman filters, increasing P to such large numbers is computationally intensive.

It is possible to slightly improve the performance further without additional increases to the number of survivors by modifying the algorithm used to determine the survivors. Instead of maintaining exactly P survivors per state, it is possible to keep an average of P_{ave} survivors per state, while keeping a minimum of only P_{min} survivors for any particular

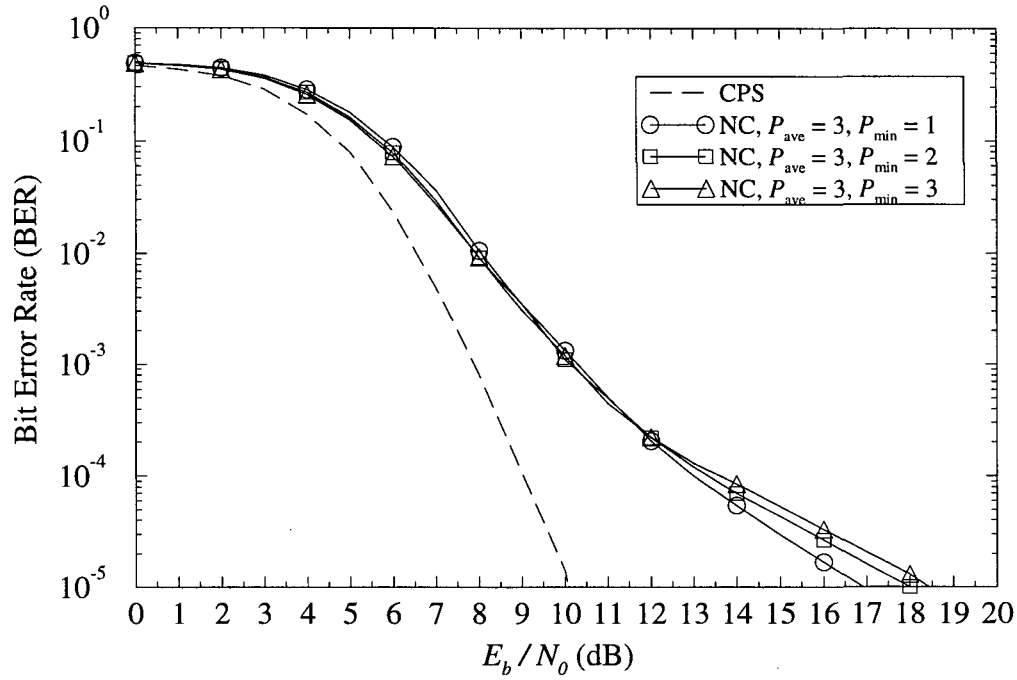


Figure 5.5: Noncoherent Detection: BER performance after the fifth iteration for the noncoherent decoders using the modified path pruning algorithm with $P_{\text{ave}} = 3$ and $P_{\text{min}} = 1$, 2, and 3. Also shown is the BER performance of the Correct Path Survives (CPS) decoder. For two-ray exponential fading with $B_d T = 0.001$.

state. The best P_{min} paths terminating in each state s are always maintained. Of the remaining paths, an additional $(P_{\text{ave}} - P_{\text{min}})M^{L-1}$ paths with the largest values of $\gamma_n(\bullet)$ are also maintained. With this modified algorithm, the total number of survivor paths is $P_{\text{ave}}M^{L-1}$. The algorithm proposed in Section 5.3.2 is supported with the algorithm by taking $P_{\text{ave}} = P_{\text{min}} = P$. With the modified algorithm using $P_{\text{ave}} = 3$ and $P_{\text{min}} = 1$, 2, and 3, the noncoherent decoders yield the results shown in Fig. 5.5. Although all three decoders use the same number of survivors ($3M^{L-1}$), and hence have the same complexity, there is a slight difference in performance. By not strictly controlling the number of survivors kept for each state, the likelihood that the correct path will be pruned is diminished. It is important to note, however, that taking $P_{\text{min}} = 0$, so the $P_{\text{ave}}M^{L-1}$ paths with the largest values of

$\gamma_n(\bullet)$ survive regardless of which state they terminate in, is entirely ineffective. At least one path must survive for each state, or else the corresponding value of $\alpha_n(\bullet)$ will be equal to zero, and the decoder will mistakenly believe that that state cannot occur.

5.4.2 Two-ray Slow Exponential Fading

When the fading rate is fast, the Kalman filters have difficultly tracking the rapidly changing channel impulse response. For a slower fading rate of $B_dT = 10^{-5}$, the performance of the proposed coherent and noncoherent systems (with $P = 1$, 2, and 3) after five iterations is shown in Fig. 5.6. Also shown is the performance of the CPS decoder and the "turbo-equalizer" for absolutely-encoded BPSK, both also after five iterations. In slow fading, the

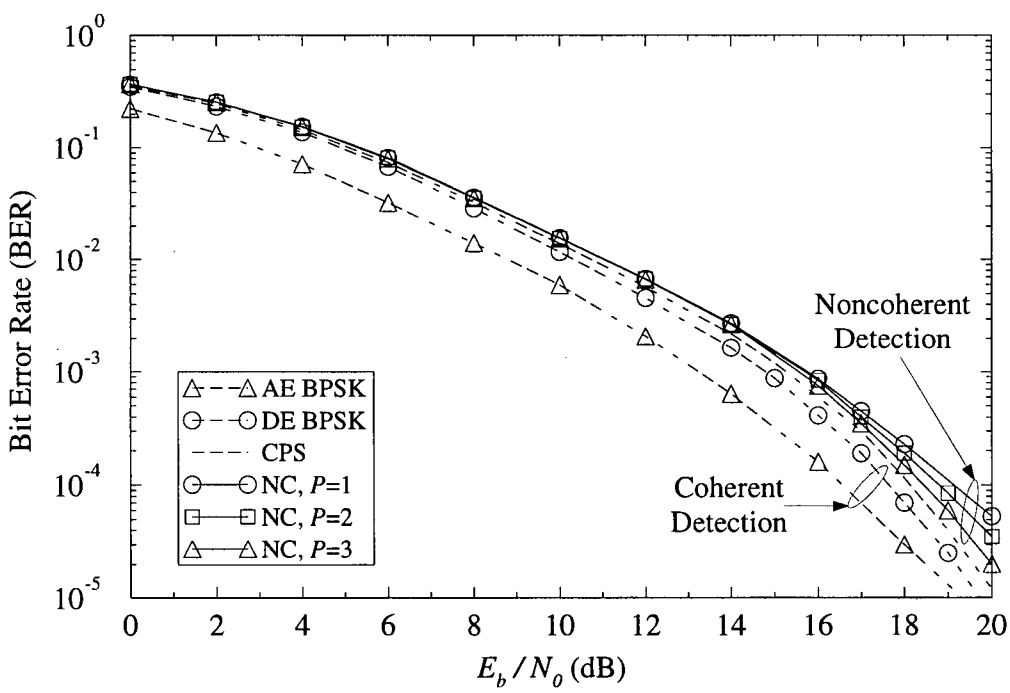


Figure 5.6: Slow Fading: BER performance after the fifth iteration for the coherent, CPS, and noncoherent decoders (with $P = 1$, 2, and 3). Also shown is the BER performance of the "turbo-equalizer" for absolutely-encoded BSPK. For two-ray exponential fading with $B_dT = 10^{-5}$.

Kalman filters are readily able to accurately estimate the channel impulse response, so there is only a 0.5 dB discrepancy between the coherent system and the CPS system. Furthermore, with $P = 3$, the noncoherent decoder is able to closely match the performance of the CPS decoder (with a difference of only 0.5 dB), and the noncoherent decoders with smaller values of P perform fair very well, with a noncoherence penalty of about 2 dB for the $P = 1$ decoder. It is important to note that the small noncoherence penalties are not due only to the accurate channel estimation, but are due in large part to the fact that the BER for all the systems is strongly governed by those instances where both rays of the channel impulse response are simultaneously subjected to long-lasting fades. When the received signal strength is weak, all of the decoders are ineffective at decoding the transmitted signal. In addition, there is a penalty of about 1 dB for using differentially encoded BPSK instead of absolutely encoded BPSK, a finding that is consistent with the slow frequency-flat fading case, as discussed in Chapter 4.

5.4.3 Two-ray Land-Mobile Fading

With the land-mobile fading model [35], the fading process cannot be modelled exactly with a state model. As a result, the ability of the Kalman filters to track the channel impulse response is diminished. This is evident in the results shown in Fig. 5.7, which is for land-mobile fading with $B_dT = 0.01$ (the average fade length is 35 symbols), and a first-order state model is used to model the channel impulse response for the noncoherent and CPS decoders.

When coherent detection is used the results are very good, with differentially encoded BPSK outperforming absolutely encoded BPSK by over 1 dB at a BER of 10^{-5}. The effect of the inexact channel modelling is apparent when the performance of the CPS decoder is considered, which is over 8 dB worse than the performance of the coherent decoder. By

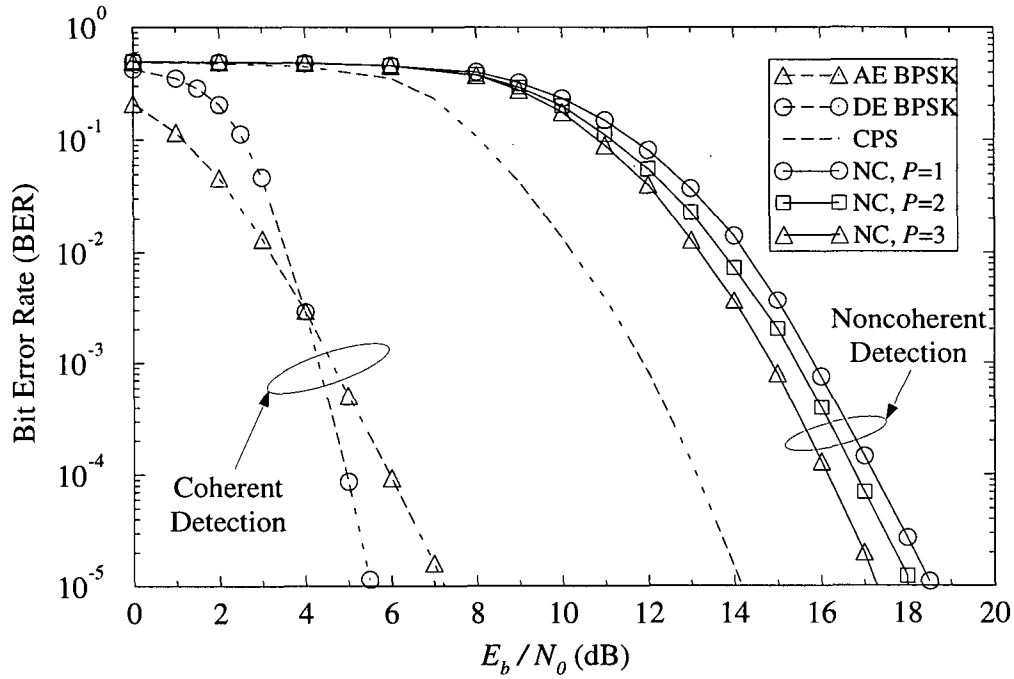


Figure 5.7: Land-Mobile Fading: BER performance after the fifth iteration for the coherent, CPS, and noncoherent decoders (with $P = 1$, 2, and 3). Also shown is the BER performance of the "turbo-equalizer" for absolutely-encoded BSPK. For two-ray land-mobile fading with $B_dT = 0.01$.

increasing the order of the state model it is possible to more accurately model the channel impulse response. The Kalman filters should then be more effective at channel estimation. However, doubling the order of the state model leads to a squaring of the complexity of the Kalman filters, so complexity limitations prohibit exploration of this hypothesis here. With the first-order model, the performance of the noncoherent decoders is well behaved, and somewhat less susceptible to the false-locks associated with the exponential fading. The noncoherence penalty at a BER of 10^{-5} is within the range of 12 - 13 dB.

5.4.4 *Continuous Delay-Spread Channels*

Although most of the performance evaluation results presented in this chapter are for the simple two-ray fading model, the inner decoder is designed to support continuous delay-spread channels. An example of such a channel is one with a multipath intensity profile specified by the Gaussian profile [37], with

$$\phi_M(\tau) = \begin{cases} 2\sqrt{\frac{\alpha}{\pi}}e^{-\alpha\tau^2} \text{ , for } 0 \leq \tau < \infty \\ 0 \text{ , } \quad \text{otherwise} \end{cases} , \tag{5.73}$$

where α is a parameter controlling the effective multipath delay spread. Larger values of α lead to longer delays spreads. With $\alpha = 0.6$ and root raised-cosine filtering with $\beta = 0.5$, the discrete-time channel impulse response has a delay spread that is effectively limited to three symbols ($L = 3$), which contain 98% of the transmitted signal energy. Although the uncorrelated scattering assumption is used for the continuous-time channel model, there is correlation between the delays of the discrete-time channel model.

The performance of the proposed coherent and noncoherent systems (with $P = 1$, 2, and 3) after five iterations is shown in Fig. 5.8 for exponential fading with $B_dT = 0.001$ and the Gaussian multipath intensity profile with $\alpha = 0.6$. Also shown in the performance of the CPS decoder, and the "turbo-equalizer" for absolutely-encoded BPSK, also after five iterations. Examination of the performance of the two coherent systems reveals that absolutely encoded BPSK is slightly better that differentially encoded BPSK (by about 0.5 dB at a BER of 10^{-5}). A comparison with the performance of the coherent systems for the two-ray fading model given in Fig. 5.1 shows that this channel is more severe than the two-ray model, with a performance difference of 3 dB for absolutely encoded BPSK and 4.7 dB for differentially encoded BPSK, both at a BER of 10^{-5}. Because of the correlation between the delay paths, it is more likely that the three delay paths will fade simultaneously than when the delay paths are uncorrelated, degrading system performance. On the other

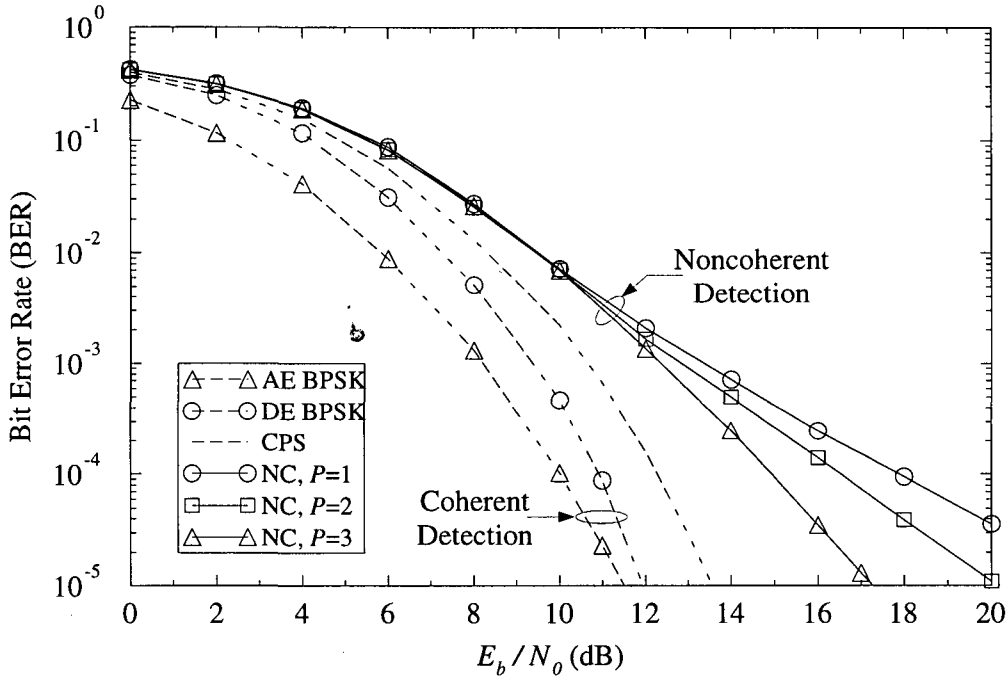


Figure 5.8: Continuous Delay-Spread Channel: BER performance after the fifth iteration for the coherent, CPS, and noncoherent decoders (with $P = 1$, 2, and 3). Also shown is the BER performance of the "turbo-equalizer" for absolutely-encoded BSPK. For exponential fading with $B_dT = 0.001$ with a Gaussian multipath intensity profile with $\alpha = 0.6$ ($L = 3$).

hand, the correlation between the delay paths is exploited by the Kalman filters to assist the channel estimation, so the noncoherence penalty is only 5 dB for the $P = 3$ decoder.

5.5 *Conclusions*

In this chapter the "turbo-equalizer" of [15] has been extended to support coherent detection of differentially encoded M-PSK signals, and its performance has been evaluated for a few different frequency-selective fading channel models. In addition, a novel soft-output decoder for noncoherent detection of differentially encoded M-PSK signals transmitted of frequency-selective fading channels has been presented, and its performance when used in

an iterative decoding structure has been examined by means of computer simulation.

It is shown that iterative noncoherent detection is a promising technique for reliable data communication in frequency-selective fading. However, there is a fairly significant noncoherence penalty, suggesting that there are likely better alternatives for the implementation of the inner decoder. In particular, the problem with "false-locks" needs to be addressed, as does the problem with modelling land-mobile fading. In the next chapter some topics for future research are suggested that may lead to reductions in the noncoherence penalty.

Chapter 6

Conclusions and Topics for Future Research

6.1 Conclusions

THE idea of using iterative decoding to improve the performance of systems using noncoherent detection of differentially encoded M-PSK signals has been proposed and explored in this thesis. Three different, well-known, channel models have been considered; namely, the AWGN channel, the Rayleigh frequency-flat fading channel, and the Rayleigh frequency-selective fading channel. For each of these three channel models, a novel soft-output decoder has been proposed that is capable of jointly estimating the channel and decoding the differential encoding. When convolutional encoding and interleaving are used at the transmitter, the novel soft-output decoders can be incorporated in an iterative decoding structure. The resulting decoding technique is referred to as *iterative noncoherent detection.*

The performance of iterative noncoherent detection has been investigated for the three channel models by means of extensive computer simulation. It has been shown that there is a clear advantage to using iterative decoding with noncoherent detection. Perhaps the most surprising finding is that in some situations noncoherently-detected differentially-encoded M-PSK can provide better performance than coherently-detected absolutely-encoded M-PSK, where both systems are using the same convolutional code for error correction. This is possible because the combination of convolutional encoding, interleaving, and differential

encoding generates a serial concatenated convolutional code, which is much more powerful than just a stand-alone convolutional code. As the concatenated code uses differential encoding it is also well suited for use with noncoherent detection. Iterative decoding is used to exploit the power of the code, and to allow the decoder to perform better channel estimation than when iterative decoding is not used. As a result, excellent BER performance can be achieved with noncoherent detection.

6.2 Topics for Future Research

Iterative noncoherent detection is a very promising idea, and there are a multitude of avenues for additional research in this area. In the following some topics of future research are suggested.

6.2.1 Theoretical Analysis

Theoretical performance analysis of the proposed systems would be very useful to help better explain the surprising performance. There are, however, a number of issues which must be addressed before useful expressions can be found. Even in the simplest case, coherent detection for the AWGN channel, there are many difficulties. Because the combination of convolutional encoding, interleaving, and differential encoding form a SCCC, the analytical tools developed in [14] for BER analysis of SCCC's could be applied. However, the results presented there are bounds on the performance of an optimal maximum likelihood sequence estimator (MLSE), as opposed to a bit-by-bit MAP decoder. Since typically there is not much difference in the BER performance of MLSE and MAP decoders, this issue is not very important. However, since the results are for an optimal decoder, the performance of the sub-optimal iterative decoder is not given. A means of theoretically determining the BER after any finite number of iterations would be more practical. In addition, the analysis makes use

of the *uniform interleaver* concept [43], and therefore provides an expression for the average performance over all of the $N!$ possible interleaver mappings instead of for any one particular mapping. Although it has been shown in this thesis that the choice of interleaver mapping does not greatly affect system performance, results for a specific interleaver mapping are preferable. Another problem that needs to be addressed is that many bounding techniques, such as the union bound, give bounds which are very loose at low signal-to-noise ratios, so care must be taken to use a bound that will be meaningful. This assumes, of course, that an exact expression for the BER cannot be found. Although it may be possible to find a useful bound on the performance when BPSK is used, when higher order modulation schemes are used (i.e. $M > 2$), the concatenated encoder is not linear. As such, the uniform error property does not hold (i.e. the probability of error depends on the transmitted message word). Most existing tools for theoretical analysis of communication system exploit the uniform error property, so this presents a particularly significant challenge.

If solutions to all of these problems can be found, then it should then be possible to extend the results for the AWGN to coherent detection for the other two channel models. Extending the results to noncoherent detection, on the other hand, will be more difficult.

6.2.2 Selection of the Code and Interleaver

Another topic of research involves the selection of the coding scheme. Determining the ideal choice of convolutional code of a given memory length and interleaver mapping would be useful, but the two must be chosen together as a pair. Replacing the convolutional code with either a turbo code or a SCCC is also well worth exploring, and, since iterative decoding is already being used, employing either of these types of codes will not greatly increase the decoder complexity. A better way to improve performance is likely by replacing the differential encoding with a more powerful code, but one that is still useful for when

noncoherent detection is used. This seems like a promising idea because it is known that for SCCC's it is better to use a more powerful inner code than a more powerful outer code [14].

6.2.3 Inner Decoder Complexity

The complexity of all the inner decoders are perhaps a little too great, and means for reducing the complexity should be investigated. One simple improvement involves implementing the APP algorithm in the log domain, using either the log-MAP or max-log-MAP (best path) algorithm. Other reductions in complexity come from examining the techniques used for channel estimation. For the AWGN channel and frequency-flat fading, the complexity of the decoder (the number of states) grows exponentially with the size of the observation window. It would be worthwhile to explore techniques for increasing the size of the observation window without increasing the number of decoder states. Care must be taken to avoid the false-lock problem found with the decoder for the frequency-selective fading channel.

6.2.4 Bandwidth Efficient Signalling

To improve the spectral efficiency of the communication system while still maintaining an acceptable BER, it is worthwhile to consider using quadrature amplitude modulation (QAM) instead of M-PSK. The inner decoders for noncoherent will have to be redesigned, but this will mostly only involve extending the designs already given here.

6.2.5 Improved Algorithms for Frequency-selective Fading

The proposed inner decoder for the frequency-selective fading with noncoherent detection yields a fairly substantial noncoherence penalty. It is not clear if this is merely an unavoidable problem with the channel conditions considered, but it seems likely that a different design for the inner decoder could be found that gives better performance. Kalman

filters, which use all of the previously received samples to estimate the impulse response, may be prone to "false-locks", so a less elaborate technique for channel estimation may, in fact, yield better performance when used in the iterative decoder. In particular, by limiting the size of the observation window over which the impulse response is estimated to some small number, Z (presumably with Z at least as large as the ISI length of the channel), it may be possible to estimate the channel with sufficient accuracy that the convolutional code will be effective enough to allow iterative decoding to succeed. By removing the need for Kalman filters, the decoder complexity will also be greatly reduced.

6.2.6 Additional Channel Models

Only three channel models have been considered in this thesis. A more generalized inner decoder should be designed for Ricean frequency-flat fading, where there is a direct line-of-sight component in addition to the Rayleigh frequency-flat fading. Both the AWGN channel and the Rayleigh frequency-flat fading channel would be supported by this model. Other amplitude distributions of the fading, such as Nakagami m-distribution, instead of the Rayleigh distribution, could also be considered. Other possibilities for the channel model include those with adjacent channel interference (ACI) and co-channel interference (CCI).

6.2.7 Frequency Hopping

In slow fading, the BER performance is limited by those instances when the channel is in a long fade. It should be possible to improve the performance by hopping to a different frequency band a few times during the transmission of each message word. Because signals at different, well-separated, frequencies tend to fade independently, it would then be unlikely that the whole message word would be lost in a fade. Because the noncoherent decoders presented here are able to adapt quickly to changes in the channel, without the need for

training sequences, they are well-suited for frequency hopped environments. It may be helpful to insert a differential encoding reference symbol prior to each hop, but this will not have a great impact on the overall code rate if the number of hops per word is small. The reference symbol may not even be necessary, but its absence will certainly have some effect on the BER.

In conclusion, because the results presented in this thesis are very encouraging, research of the topics suggested above will likely be a constructive endeavour.

Afterword

Although the material presented in this thesis is original work, it is by no means unique. During the course of my investigation into this subject, the concept of iterative noncoherent detection has been proposed independently by other researchers. In particular, credit is due to Peleg and Shamai, who published the first paper on iterative noncoherent detection [44]. The research in [44], and in [45], is for the AWGN channel with static carrier phase error. Some theoretical analysis for the system performance with coherent detection for the AWGN channel is presented in [46]. Iterative noncoherent detection for the AWGN channel with dynamic carrier phase error is investigated in [45,47,48]. In addition, Lodge and Gertsman explored iterative noncoherent detection for Rayleigh frequency-flat fading [49], and Hoeher and Lodge presented a system that is nearly identical to the one presented here for frequency-flat fading [50].

As far as my own research and publications are concerned, I began investigating turbo codes and iterative decoding in January 1996. In September of that year we submitted a paper to the IEEE Journal on Selected Areas of Communications, proposing a system which is similar to the one presented in Chapter 4 of this thesis, for Rayleigh frequency-flat fading, although a turbo code was used for error correction instead of a convolutional code (this paper was published in February 1998 [51], and the material was presented at a conference in December 1997 [52]). In September 1997 we proposed iterative noncoherent detection for Rayleigh frequency-selective fading [53] (Chapter 5 of this thesis). The research into the AWGN channel followed (Chapter 3), with the submission of a paper to the IEEE

Transactions on Communications in February 1998 [54], which was also presented at a conference in June 1998 [55].

Bibliography

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

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

[3] P. Robertson, "Improving decoder and code structure of parallel concatenated recursive systematic (turbo) codes," in *Proceedings of the IEEE International Conference on Universal Personal Communications (ICUPC'94)*, San Diego, Sept. 1994, pp. 1298–1303.

[4] A. S. Barbulescu and S. S. Pietrobon, "Interleaver design for turbo codes," *IEE Electronics Letters*, vol. 30, no. 25, pp. 2107–2108, Dec. 1994.

[5] S. Dolinar and D. Divsalar, "Weight distributions for turbo codes using random and nonrandom permutations," JPL TDA Progress Report 42-122, Jet Propulsion Laboratory, Pasadena, California, Aug. 1995.

[6] D. Divsalar, S. Dolinar, and F. Pollara, "Transfer function bounds on the performance of turbo codes," JPL TDA Progress Report 42-122, Jet Propulsion Laboratory, Pasadena, California, Aug. 1995.

[7] D. Divsalar, S. Dolinar, R. McEliece, and F. Pollara, "Performance analysis of turbo codes," in *Proceedings of the Military Communications Conference (MILCOM'95)*, San Diego, Nov. 1995, pp. 91–96.

[8] S. Benedetto and G. Montorsi, "Design guidelines of parallel concatenated convolutional codes," in *Proceedings of the IEEE Global Conference on Communications (GLOBECOM'95)*, Singapore, Nov. 1995, pp. 2273–2277.

[9] E. K. Hall and S. G. Wilson, "Design and performance analysis of turbo codes on Rayleigh fading channels," in *Proceedings of the Conference on Information Sciences and Systems (CISS'96)*, Princeton, New Jersey, Mar. 1996, pp. 43–48.

[10] *Proceedings of the International Symposium on Turbo Codes and Related Topics*, Brest, France, Sept. 1997.

[11] *IEEE Journal on Selected Areas in Communications*, vol. 16, Feb. 1998.

[12] S. Benedetto and G. Montorsi, "Serial concatenation of block and convolutional codes," *IEE Electronics Letters*, vol. 32, no. 10, pp. 887–888, May 1996.

[13] S. Benedetto and G. Montorsi, "Iterative decoding of serially concatenated convolutional codes," *IEE Electronics Letters*, vol. 32, no. 13, pp. 1186–1188, June 1996.

[14] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design, and iterative decoding," JPL TDA Progress Report 42-126, Jet Propulsion Laboratory, Pasadena, California, Aug. 1996.

[15] C. Douillard, M. Jezequel, C. Berrou, A. Picart, P. Didier, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," *European Transactions on Telecommunications*, vol. 6, pp. 507–511, Sept.-Oct. 1995.

[16] A. Glavieux, C. Laot, and J. Labat, "Turbo-equalization over a frequency selective channel," in *Proceedings of the International Symposium on Turbo Codes* [10], pp. 96–102.

[17] P. Jung, M. Nasshan, and J. Blanz, "Application of turbo-codes to a CDMA mobile radio system using joint detection and antenna diversity," in *Proceedings of the IEEE 44th Vehicular Technology Conference (VTC'94)*, Stockholm, June 1994, pp. 770–774.

[18] R. Herzog, A. Schmidbauer, and J. Hagenauer, "Iterative decoding and despreading improves CDMA-systems using M-ary orthogonal modulation and FEC," in *Proceedings of the IEEE International Conference on Communications (ICC'97)*, Montreal, June 1997.

[19] M. C. Reed, C. B. Schlegel, P. Alexander, and J. Asenstorfer, "Iterative multi-user detection for DS-CDMA with FEC," in *Proceedings of the International Symposium on Turbo Codes* [10], pp. 162–165.

[20] K. R. Narayanan and G. L. Stüber, "A novel ARQ technique using the turbo coding principle," *IEEE Communications Letters*, vol. 1, pp. 49–51, Mar. 1997.

[21] J. Garcia-Frias and J. D. Villasenor, "Joint source channel decoding of turbo codes," in *Proceedings of the International Symposium on Turbo Codes* [10], pp. 96–102.

[22] R. Link and S. Kallel, "Markov model aided decoding for DPCM image transmission," in *Proceedings of the 19th Biennial Symposium on Communications*, Kingston, Ontario, May 1998, pp. 174–178.

[23] F. Davarian, "Mobile digital communications via tone calibration," *IEEE Transactions on Vehicular Technology*, vol. 36, no. 5, pp. 55–61, May 1987.

[24] J. G. Proakis, *Digital Communications*, McGraw-Hill, New York, 3rd edition, 1995.

[25] J. K. Cavers, "An analysis of pilot symbol assisted modulation for Rayleigh fading channels," *IEEE Transactions on Vehicular Technology*, vol. 40, pp. 686–693, Nov. 1991.

[26] M. K. Simon and D. Divsalar, "On the optimality of classical coherent receivers of differentially encoded M-PSK," *IEEE Communications Letters*, vol. 1, pp. 67–70, May 1997.

[27] D. Makrakis and P. T. Mathiopoulos, "Trellis coded noncoherent QAM: A new bandwidth and power efficient scheme," in *Proceedings of the IEEE 39th Vehicular Technology Conference (VTC'89)*, San Francisco, May 1989, pp. 95–100.

[28] D. Makrakis and P. T. Mathiopoulos, "Optimal decoding in fading channels: A combined envelope, multiple differential and coherent approach," in *Proceedings of the IEEE Global Conference on Communications (GLOBECOM'89)*, Dallas, Nov. 1989, pp. 1551–1557.

[29] D. Makrakis and K. Feher, "Optimal noncoherent detection of PSK signals," *IEE Electronics Letters*, vol. 26, pp. 146–155, Mar. 1990.

[30] D. Divsalar and M. K. Simon, "Multiple-symbol differential detection of MPSK," *IEEE Transactions on Communications*, vol. 38, pp. 300–308, Mar. 1990.

[31] F. Edbauer, "Bit error rate of binary and quaternary DPSK signals with multiple differential feedback decision," *IEEE Transactions on Communications*, vol. 40, pp. 457–460, Mar. 1992.

[32] S. Samejima, K. Enomoto, and Y. Watabe, "Differential PSK system nonredundant error correction," *IEEE Journal on Selected Areas in Communications*, vol. 1, pp. 78–81, Jan. 1983.

[33] D. Makrakis, P. T. Mathiopoulos, and D. P. Bouras, "Optimal decoding of coded PSK and QAM signals in correlated fast fading channels and AWGN: A combined envelope, multiple differential and coherent approach," *IEEE Transactions on Communications*, vol. 42, pp. 63–75, Jan. 1994.

[34] D. P. Bouras, *Advanced Noncoherent Receivers for Mobile Fading Channels*, Ph.D. thesis, The University of British Columbia, Vancouver, BC, Canada, Apr. 1995.

[35] W. C. Jakes, *Microwave Mobile Communications*, John Wiley & Sons, New York, 1974.

[36] W. Y.-C. Lee, *Mobile Communications Engineering*, McGraw-Hill, New York, 1982.

[37] Q. Dai and E. Shwedyk, "Detection of bandlimited signals over frequency selective Rayleigh fading channels," *IEEE Transactions on Communications*, pp. 941–950, Feb./Mar./Apr. 1994.

[38] R. W. Chang and J. C. Hancock, "On receiver structures for channels having memory," *IEEE Transactions on Information Theory*, vol. 12, pp. 463–468, Oct. 1966.

[39] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, pp. 284–287, Mar. 1974.

[40] C. Berrou and A. Glavieux, "New optimal error correcting coding and decoding: Turbo codes," *IEEE Transactions on Communications*, vol. 44, pp. 1261–1271, Oct. 1996.

[41] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Transactions on Information Theory*, vol. 28, no. 1, pp. 55–67, Jan. 1982.

[42] L. J. Mason, "Error probability evaluation for systems employing differential detection in a Rician fast fading environment and Gaussian noise," *IEEE Transactions on Communications*, vol. 35, no. 1, pp. 39–46, Jan. 1987.

[43] S. Benedetto and G. Montorsi, "Average performance of parallel concatenated block codes," *IEE Electronics Letters*, vol. 31, no. 3, pp. 156–158, Feb. 1995.

[44] M. Peleg and S. Shamai (Shitz), "Iterative decoding of coded and interleaved noncoherent multiple symbol detected DPSK," *IEE Electronics Letters*, vol. 33, no. 12, pp. 1018–1020, June 1997.

[45] M. Peleg, S. Shamai (Shitz), and S. Galan, "On iterative decoding for coded noncoherent MPSK communications over block-noncoherent AWGN channel," in *International Conference on Telecommunications (ICT'98)* [57], pp. 109–114.

[46] M. Peleg, I. Sason, S. Shamai (Shitz), and A. Elia, "On interleaved, differentially encoded convolutional codes," CC Pub. #235, Dept. of Electrical Engineering, Technion, Haifa, Israel, Feb. 1998.

[47] S. Galán, M. Peleg, and S. Shamai (Shitz), "On iterative phase trellis based noncoherent detection of coded MPSK in a noisy phase regime," in *Proceedings of MELECON 98*, Tel Aviv, May 1998, pp. 834–838.

[48] M. Peleg, S. Galán, and S. Shamai (Shitz), "On iterative decoding for coded noncoherent MPSK communications over phase-noisy AWGN channel," CC Pub. #249, Dept. of Electrical Engineering, Technion, Haifa, Israel, June 1998.

[49] J. Lodge and M. Gertsman, "Joint detection and decoding by turbo processing for fading channel communications," in *Proceedings of the International Symposium on Turbo Codes* [10], pp. 88–95.

[50] P. Hoeher and J. Lodge, "Iterative decoding/demodulation of coded DPSK systems," in *Proceedings of the IEEE Global Conference on Communications (GLOBECOM'98)*, Syndey, Nov. 1998.

[51] I. D. Marsland and P. T. Mathiopoulos, "Multiple differential detection of parallel concatenated convolutional (turbo) codes in correlated fast Rayleigh fading," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 265–275, Feb. 1998.

[52] I. D. Marsland and P. T. Mathiopoulos, "Multiple differential detection of turbo codes in correlated fast Rayleigh fading," in *Proceedings of the IEEE International Conference on Electrical Circuits and Systems (ICECS'97)*, Cairo, Dec. 1997.

[53] I. D. Marsland, P. T. Mathiopoulos, and S. Kallel, "Noncoherent turbo-equalization for frequency selective Rayleigh fast fading channels," in *Proceedings of the International Symposium on Turbo Codes* [10], pp. 196–199.

[54] I. D. Marsland and P. T. Mathiopoulos, "Iterative noncoherent detection of convolutionally encoded signals," in *International Conference on Telecommunications (ICT'98)* [57], pp. 115–120.

[55] I. D. Marsland and P. T. Mathiopoulos, "Iterative noncoherent detections of convolutionally encoded signals," *IEEE Transactions on Communications*, Feb. 1998, submitted.

[56] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "A soft-input soft-output APP module for iterative decoding of concatenated codes," *IEEE Communications Letters*, vol. 1, pp. 22–24, Jan. 1997.

[57] *International Conference on Telecommunications (ICT'98)*, Porto Carras, Greece, June 1998.

Appendix A

The APP Algorithm

IN 1966, Chang and Hancock developed an algorithm for combating intersymbol interference by computing the *a posteriori* probability (APP) distribution of the transmitted data [38]. This same algorithm was later proposed by Bahl, Cocke, Jelinek and Raviv as a means for decoding convolutional codes [39]. This algorithm, known variously as the APP, MAP, Bahl, Bahl *et al.*, or BCJR algorithm, has even broader application beyond these two cases, including the uses described in this thesis. Although the following description is given primarily in the context of decoding convolutional codes, other applications are realized merely be redefining the meaning of the various system parameters and inputs. To isolate the implementation of the algorithm from its application, the concept of the soft-input soft-output (SISO) module [56] is used in the following description.

To illustrate the algorithm, consider the following system model. Let $\underline{u} = u_1, u_2, \ldots, u_N$ denote a sequence of N symbols, with each symbol drawn randomly and independently from some input alphabet, $\mathcal{U}$. These symbols are used to drive a finite-state machine, such as a convolutional encoder, through a sequence of states over some state-space, $\mathcal{S}$, with $\underline{s} = s_1, s_2, \ldots, s_{N+1}$ denoting the state sequence. From an initial state of s_1, a state transition occurs with each input symbol, and with each state transition the machine emits an output symbol taken from some output alphabet, $\mathcal{V}$. The state transitions and output symbols are causal and deterministic with respect to the input symbols, and are governed by

the state transition matrix, $\mathrm{ST}[\bullet,\bullet]$, and symbol generation matrix, $\mathrm{SG}[\bullet,\bullet]$, respectively. If, at any time, the current state is $s \in \mathcal{S}$, then, in response to input $u \in \mathcal{U}$, the machine emits symbol $\mathrm{SG}[s,u] \in \mathcal{V}$ and advances to state $\mathrm{ST}[s,u] \in \mathcal{S}$. The sequence of output symbols are denoted as $\underline{v} = v_1, v_2, \ldots, v_N$ with $v_n \in \mathcal{V}$. Noisy observations of the output symbols are available to the decoder, and are denoted by $\underline{r} = r_1, r_2, \ldots, r_N$. The noise affecting one observation is independent of the noise affecting the other observations. The primary purpose of the algorithm is to compute the *a posteriori* probability distribution of each of the input symbols based on the noisy observations of the output symbols.

The algorithm is based primarily on the finite-state machine (FSM) model, which includes the state space ($\mathcal{S}$), the state transition and symbol generation matrices ($\mathrm{ST}[\bullet,\bullet]$ and $\mathrm{SG}[\bullet,\bullet]$, respectively), and the input and output alphabets ($\mathcal{U}$ and $\mathcal{V}$). In addition to the FSM model, the algorithm needs to have knowledge of the sequence length (N) and the *a priori* probability distribution of the machine's initial state ($\Pr\{s_1 = s\}\ \forall\ s \in \mathcal{S}$). The algorithm also needs the *a priori* probability distribution of the input symbols ($\Pr\{u_n = u\}\ \forall\ u \in \mathcal{U},\ n \in \{1,2,\ldots,N\}$) and the branch metrics ($f\left(r_n \,\middle|\, v_n = v\right)\ \forall\ v \in \mathcal{V},\ n \in \{1,2,\ldots,N\}$). Based on these system parameters and branch metrics the algorithm calculates the APP's of the input symbols ($\Pr\left\{u_n = u \,\middle|\, \underline{r}\right\}$). The algorithm may also be configured to compute the APP's for the output symbols ($\Pr\left\{v_n = v \,\middle|\, \underline{r}\right\}$).

For the various applications of the APP algorithm in this thesis it is not always possible to use the correct values branch metrics, or desirable to use the correct values of the *a priori* probabilities, and estimates are used instead. To isolate the functionality of the APP algorithm from its inputs, the concept of the SISO module is useful. The SISO module has two input ports, labelled $P^{(I_u)}_{\bullet,\bullet}$ and $\mu_\bullet(\bullet)$. It is intended that the *a priori* probabilities are passed to the $P^{(I_u)}_{\bullet,\bullet}$ port as

$$P^{(I_u)}_{n,u} = \Pr\{u_n = u\} \tag{A.1}$$

and the branch metrics are passed to $\mu_\bullet(\bullet)$ as

$$\mu_n(v) = f\Big(r_n \;\Big|\; v_n = v\Big) \ . \tag{A.2}$$

The SISO module also has two output ports, labelled $P_{\bullet,\bullet}^{(O_u)}$ and $P_{\bullet,\bullet}^{(O_v)}$. If the correct values are supplied to the input ports then

$$P_{n,u}^{(O_u)} = \Pr\Big\{u_n = u \;\Big|\; \underline{r}\Big\} \tag{A.3}$$

and

$$P_{n,v}^{(O_v)} = \Pr\Big\{u_n = v \;\Big|\; \underline{r}\Big\} \ , \tag{A.4}$$

which are the APP's of the input and output symbols of the FSM, respectively.

The algorithm is implemented with two recursive procedures, one working forward over n, followed by one in reverse. In the forward recursion the algorithm calculates the intermediate quantities $\alpha_n(s) = \Pr\Big\{s_n = s \;\Big|\; \underline{r}_1^{n-1}\Big\}$, where $\underline{r}_1^{n-1}$ is shorthand for $r_1, r_2, \ldots, r_{n-1}$. The recursion is initialized with the *a priori* probability distribution of the initial state of the FSM, $\alpha_1(s) = \Pr\{s_1 = s\}$. Then, for $n = 1, 2, \ldots, N$, $\alpha_{n+1}(s')$ is computed for all $s' \in \mathcal{S}$, by

$$\alpha_{n+1}(s') = \frac{1}{\Omega_n} \sum_{s\in\mathcal{S}} \sum_{u\in\mathcal{U}} P_{n,u}^{(I_u)} \alpha_n(s) \mu_n(\mathrm{SG}\,[s,u]) \Pr\Big\{s_{n+1} = s' \;\Big|\; s_n = s, u_n = u\Big\} \ , \tag{A.5}$$

where

$$\Pr\Big\{s_{n+1} = s' \;\Big|\; s_n = s, u_n = u\Big\} = \begin{cases} 1 \ , \text{ if } s' = \mathrm{ST}\,[s,u] \\ 0 \ , \text{ otherwise} \ . \end{cases} \tag{A.6}$$

The quantity $\Omega_n = f\Big(r_n \;\Big|\; \underline{r}_1^{n-1}\Big)$ is a scale factor that does not need to be calculated provided that $\alpha_{n+1}(\bullet)$ is normalized so that $\sum_{s'\in\S} \alpha_{n+1}(s')$ equals one. The quantities $\alpha_n(\bullet)$ are stored for use in the reverse recursion.

Once the forward recursion has completed, the algorithm computes another set of intermediate quantities,

$$\beta_n(s) = \frac{f\left(\underline{r}_n^N \,\middle|\, s_n = s, \underline{r}_1^{n-1}\right)}{f\left(\underline{r}_n^N \,\middle|\, \underline{r}_1^{n-1}\right)} \tag{A.7}$$

from $\beta_{n+1}(s')$ in a reverse recursion. From an initial condition of $\beta_{N+1}(s) = 1$ for all $s \in \mathcal{S}$, the algorithm computes

$$\beta_n(s) = \frac{1}{\Omega_n} \sum_{u \in \mathcal{U}} P_{n,u}^{(I_u)} \mu_n(\mathrm{SG}\,[s,u]) \beta_{n+1}(\mathrm{ST}\,[s,u]) \tag{A.8}$$

for $n = N, N-1, \ldots, 2, 1$. During the reverse recursion the algorithm also computes the desired APP's. The APP's for the input symbols are given by

$$P_{n,u}^{(O_u)} = \frac{1}{\Omega_n} P_{n,u}^{(I_u)} \sum_{s \in \mathcal{S}} \alpha_n(s) \mu_n(\mathrm{SG}\,[s,u]) \beta_{n+1}(\mathrm{ST}\,[s,u]) \,, \tag{A.9}$$

and the APP's for the output symbols are given by

$$P_{n,v}^{(O_v)} = \frac{1}{\Omega_n} \mu_n(v) \sum_{s \in \mathcal{S}} \sum_{u \in \mathcal{U}} P_{n,u}^{(I_u)} \alpha_n(s) \beta_{n+1}(\mathrm{ST}\,[s,u]) \Pr\left\{v_n = v \,\middle|\, s_n = s, u_n = u\right\} \,, \tag{A.10}$$

where

$$\Pr\left\{v_n = v \,\middle|\, s_n = s, u_n = u\right\} = \begin{cases} 1 \,, \text{ if } v = \mathrm{SG}\,[s,u] \\ 0 \,, \text{ otherwise} \,. \end{cases} \tag{A.11}$$

By following this algorithm, the values stored in $P_{n,u}^{(O_u)}$ and $P_{n,v}^{(O_v)}$ will correctly reflect $\Pr\left\{u_n = u \,\middle|\, \underline{r}\right\}$ and $\Pr\left\{v_n = v \,\middle|\, \underline{r}\right\}$, provided that the correct values of $\Pr\{u_n = u\}$ and $f\left(r_n \,\middle|\, v_n = v\right)$ are supplied to the inputs $P_{n,u}^{(I_u)}$ and $P_{n,v}^{(I_v)}$.

Appendix B

Detector Analysis and Discrete-Time Channel Models

FOR each of the three channel models investigated in this thesis, as described in Sections 3.1, 4.1, and 5.1, it is necessary to derive mathematical models of the detector output. These models, which are used to guide the design of the inner decoders, provide expressions for the detector output, $\underline{r}$, in terms of the transmitted symbols, $\underline{v}$. They define abstract discrete-time channels which encapsulate the signal generator, the continuous-time channel, and the detector, as shown in Fig. B.1.

The received signal, as given by either Eq. (3.1), Eq. (4.1), or Eq. (5.1) depending on the channel model, can be expressed as

$$r_c(t) = \mathbf{Re}\left\{y_l(t)\sqrt{2}\exp\left\{j2\pi f_c t\right\}\right\} + w_c(t) \tag{B.1}$$

where $y_l(t)$ is the complex lowpass equivalent data-bearing portion of the received signal, $w_c(t)$ is the baseband AWGN signal, and f_c is the carrier frequency. The received lowpass data-bearing signal depends on the channel model and is given by

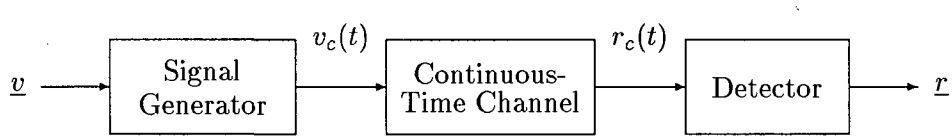


Figure B.1: Block diagram of the discrete-time channel.

$$y_l(t) \triangleq \begin{cases} v(t)e^{j\phi_c} , & \text{for the AWGN channel,} \quad \text{(B.2a)} \\ v(t)h(t) , & \text{for frequency-flat fading,} \quad \text{(B.2b)} \\ \int_{-\infty}^{\infty} v(t-\tau)h(\tau;t)\, d\tau , & \text{for frequency-selective fading,} \quad \text{(B.2c)} \end{cases}$$

where $v(t)$ is the transmitted lowpass signal given by Eq. (2.5), ϕ_c is the static carrier phase error of the AWGN channel, $h(t)$ is the frequency-flat fading process, and $h(\tau;t)$ is the frequency-selective fading channel lowpass impulse response.

By substituting Eq. (B.1) for $r_c(t)$ in Eq. (2.11), the demodulated received signal is

$$\begin{aligned} r_o(t) &\triangleq r_c(t)\sqrt{2}\exp\{-j2\pi f_c t\} \\ &= \Big[\mathbf{Re}\left\{y_l(t)\sqrt{2}\exp\{j2\pi f_c t\}\right\} + w_c(t)\Big]\sqrt{2}\exp\{-j2\pi f_c t\} \\ &= y_l(t) + y_l^*(t)\exp\{-j4\pi f_c t\} + w_o(t) , \end{aligned} \tag{B.3}$$

where

$$w_o(t) \triangleq w_c(t)\sqrt{2}\exp\{-j2\pi f_c t\} . \tag{B.4}$$

is the demodulated noise signal. The first term of Eq. (B.3) is the lowpass data-bearing portion of the received signal. The second term is a high-frequency signal, with data centered around a frequency of $2f_c$. This component will be removed by the lowpass receive filter and can be ignored. The third term in Eq. (B.3) is the demodulated noise.

An expression for the filtered signal, $r(t)$, can be found by substituting Eq. (B.3) for the demodulated signal in Eq. (2.12). Ignoring the high-frequency component, this yields

$$\begin{aligned} r(t) &\triangleq \int_{-\infty}^{\infty} r_o(t-\alpha)h_R(\alpha)\, d\alpha \\ &= \int_{-\infty}^{\infty} \Big[y_l(t-\alpha) + w_o(t-\alpha)\Big]h_R(\alpha)\, d\alpha \\ &= \int_{-\infty}^{\infty} y_l(t-\alpha)h_R(\alpha)\, d\alpha + w(t) , \end{aligned} \tag{B.5}$$

where

$$w(t) \triangleq \int_{-\infty}^{\infty} w_o(t-\alpha)h_R(\alpha)\, d\alpha \tag{B.6}$$

is the lowpass noise signal.

By substituting Eq. (B.5) for the filtered signal in Eq. (2.13), the received samples are given by

$$\begin{aligned} r_n &\triangleq r(nT) \\ &= \int_{-\infty}^{\infty} y_l(nT-\alpha)h_R(\alpha)\,d\alpha + w_n\ , \end{aligned} \tag{B.7}$$

where

$$w_n \triangleq w(nT) \tag{B.8}$$

is the sampled noise, and $\{w_n\}$ is the discrete-time noise process.

To express the received samples in terms of the transmitted symbols, the particular channel models are considered separately.

B.1 The Additive White Gaussian Noise Channel

With the AWGN channel $y_l(t)$ is defined by Eq. (B.2a). In this case the received samples are

$$\begin{aligned} r_n &= \int_{-\infty}^{\infty} y_l(nT-\alpha)h_R(\alpha)\,d\alpha + w_n \\ &= \int_{-\infty}^{\infty} \Big[v(nT-\alpha)e^{j\phi_c}\Big]h_R(\alpha)\,d\alpha + w_n\ . \end{aligned} \tag{B.9}$$

Substituting Eq. (2.5) for the transmitted lowpass signal, $v(t)$, yields

$$\begin{aligned} r_n &= \int_{-\infty}^{\infty} \Big[\sum_{m=0}^{N} v_m h_T(nT-\alpha-mT)\Big]e^{j\phi_c}h_R(\alpha)\,d\alpha + w_n \\ &= \sum_{m=0}^{N} v_m e^{j\phi_c}\int_{-\infty}^{\infty} h_T\Big((n-m)T-\alpha\Big)h_R(\alpha)\,d\alpha + w_n \\ &= \sum_{m=0}^{N} v_m e^{j\phi_c} h_{TR}\Big((n-m)T\Big) + w_n\ , \end{aligned} \tag{B.10}$$

where

$$h_{TR}(t) \triangleq \int_{-\infty}^{\infty} h_T(t-\alpha)h_R(\alpha)\,d\alpha \tag{B.11}$$

is the combined impulse response of the transmit and receive filters. Since the receive filter is matched to the transmit filter, $h_R(t) = h_T^*(-t)$ and

$$\begin{aligned} h_{TR}(t) &= \int_{-\infty}^{\infty} h_T(t-\alpha)h_T^*(-\alpha)\, d\alpha \\ &= \int_{-\infty}^{\infty} h_T(t+\alpha)h_T^*(\alpha)\, d\alpha \ . \end{aligned} \tag{B.12}$$

The requirements on the transmit filter impulse response described in Section 2.2 imply that Eq. (2.6) holds, so

$$h_{TR}(nT) = \int_{-\infty}^{\infty} h_T(nT+\alpha)h_T^*(\alpha)\, d\alpha = \delta_n \ , \tag{B.13}$$

and the received samples are therefore

$$\begin{aligned} r_n &= \sum_{m=0}^{N} v_m e^{j\phi_c}\delta_{n-m} + w_n \\ &= v_n e^{j\phi_c} + w_n \ . \end{aligned} \tag{B.14}$$

Analysis of the noise is also straightforward. Since the baseband noise is Gaussian with zero mean, the demodulated noise, given by Eq. (B.4), has a complex Gaussian distribution with zero mean and an autocorrelation function of

$$\begin{aligned} \phi_{w_o}(\Delta t) &\triangleq \frac{1}{2}\mathbf{E}\left[w_o^*(t-\Delta t)w_o(t)\right] \\ &= \frac{1}{2}\mathbf{E}\left[\left(\begin{array}{c} w_c^*(t-\Delta t)\sqrt{2}\exp\{j2\pi f_c(t-\Delta t)\} \\ \times \quad w_c(t)\sqrt{2}\exp\{-j2\pi f_c t\} \end{array}\right)\right] \\ &= \mathbf{E}\left[w_c(t-\Delta t)w_c(t)\right]\exp\left\{-j2\pi f_c\Delta t\right\} \\ &= \phi_{w_c}(\Delta t)\exp\{-j2\pi f_c\Delta t\} \end{aligned} \tag{B.15}$$

where $\phi_{w_c}(\Delta t)$ is the autocorrelation function of the bandpass noise process, $w_c(t)$. Substituting Eq. (3.2) for $\phi_{w_c}(\Delta t)$ gives

$$\begin{aligned} \phi_{w_o}(\Delta t) &= \left[\frac{\mathcal{N}_0}{2}\,\delta(\Delta t)\right]\exp\{-j2\pi f_c\Delta t\} \\ &= \frac{\mathcal{N}_0}{2}\,\delta(\Delta t)\ . \end{aligned} \tag{B.16}$$

The filtered noise, given by Eq. (B.6), also has a complex Gaussian distribution, with zero mean and an autocorrelation function of

$$\begin{aligned}
\phi_w(\Delta t) &\triangleq \frac{1}{2}\mathbf{E}\left[w^*(t-\Delta t)w(t)\right] \\
&= \frac{1}{2}\mathbf{E}\left[\int_{-\infty}^{\infty} w_o^*(t-\Delta t-\alpha_1)h_R^*(\alpha_1)\,d\alpha_1 \int_{-\infty}^{\infty} w_o(t-\alpha_2)h_R(\alpha_2)\,d\alpha_2\right] \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{1}{2}\mathbf{E}\left[w_o^*(t-\Delta t-\alpha_1)w_o(t-\alpha_2)\right] h_R^*(\alpha_1)h_R(\alpha_2)\,d\alpha_1\,d\alpha_2 \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \phi_{w_o}(\Delta t+\alpha_1-\alpha_2)h_R^*(\alpha_1)h_R(\alpha_2)\,d\alpha_1\,d\alpha_2\ .
\end{aligned} \tag{B.17}$$

Substituting Eq. (B.16) for $\phi_{w_o}(\Delta t)$ yields

$$\begin{aligned}
\phi_w(\Delta t) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left[\frac{\mathcal{N}_0}{2}\delta(\Delta t+\alpha_1-\alpha_2)\right] h_R^*(\alpha_1)h_R(\alpha_2)\,d\alpha_1\,d\alpha_2 \\
&= \frac{\mathcal{N}_0}{2}\int_{-\infty}^{\infty} h_R^*(\alpha-\Delta t)h_R(\alpha)\,d\alpha\ .
\end{aligned} \tag{B.18}$$

Since $h_R(t) = h_T^*(-t)$,

$$\begin{aligned}
\phi_w(\Delta t) &= \frac{\mathcal{N}_0}{2}\int_{-\infty}^{\infty} h_T(\Delta t-\alpha)h_R(\alpha)\,d\alpha \\
&= \frac{\mathcal{N}_0}{2}h_{TR}(\Delta t)\ ,
\end{aligned} \tag{B.19}$$

where $h_{TR}(t)$ is defined in Eq. (B.11). The discrete-time noise process, given by Eq. (B.8), also has a complex Gaussian distribution, with zero mean and an autocorrelation function of

$$\begin{aligned}
\phi_m^{(W)} &\triangleq \frac{1}{2}\mathbf{E}\left[w_{n-m}^* w_n\right] \\
&= \frac{1}{2}\mathbf{E}\left[w^*\big((n-m)T\big)w\big(nT\big)\right] \\
&= \phi_w(mT)\ .
\end{aligned} \tag{B.20}$$

Substituting Eq. (B.19) for $\phi_w(\Delta t)$ yields

$$\phi_m^{(W)} = \frac{\mathcal{N}_0}{2}h_{TR}(mT) = \frac{\mathcal{N}_0}{2}\delta_m \tag{B.21}$$

because of Eq. (B.13). Therefore, the noise samples are uncorrelated (and also independent).

In summary, for the AWGN channel the detector output, $\underline{r}$, can be modelled in terms of the transmitted symbols, $\underline{v}$, by

$$r_n = v_n e^{j\phi_c} + w_n\ , \tag{B.22}$$

where ϕ_c is the carrier phase error and $\{w_n\}$ is the discrete-time noise process. The phase error is uniformly distributed over $[0, 2\pi)$ and remains constant over the entire block of received samples. The noise process has a complex Gaussian distribution with zero mean and an autocorrelation function of $\phi_m^{(W)} = \frac{N_0}{2}\delta_m$.

B.2 Rayleigh Frequency-Flat Fading

For the frequency-flat fading channel model $y_l(t)$ is given by Eq. (B.2b). Substituting this case into Eq. (B.7) yields received samples of

$$r_n = \int_{-\infty}^{\infty} [v(nT - \alpha)h(nT - \alpha)]\, h_R(\alpha)\, d\alpha + w_n \,. \tag{B.23}$$

By using Eq. (2.5) for $v(t)$, this can be expressed as

$$\begin{aligned} &= \int_{-\infty}^{\infty} \left[\sum_{m=0}^{N} v_m h_T(nT - \alpha - mT)\right] h(nT - \alpha) h_R(\alpha)\, d\alpha + w_n \\ &= \sum_{m=0}^{N} v_m \int_{-\infty}^{\infty} h_T\Big((n-m)T - \alpha\Big) h(nT - \alpha) h_R(\alpha)\, d\alpha + w_n \,. \end{aligned} \tag{B.24}$$

To simplify Eq. (B.24) it is convenient to note that the fading process, $h(t)$, tends to vary quite slowly relative to the symbol duration. Since $|h_R(t)|$ is essentially nonzero only for a short period around $t = 0$, it is safe to assume that $h(t)$ does not change significantly during this period. As such, $h(nT - \alpha) \cong h(nT)$ for small $|\alpha|$. This assumption is made only to simplify decoder design, and has not been used when testing the communication system. Taking the approximation as exact leads to

$$\begin{aligned} r_n &= \sum_{m=0}^{N} v_m h(nT) \int_{-\infty}^{\infty} h_T\Big((n-m)T - \alpha\Big) h_R(\alpha)\, d\alpha + w_n \\ &= \sum_{m=0}^{N} v_m h(nT) h_{TR}\Big((n-m)T\Big) + w_n \,, \end{aligned} \tag{B.25}$$

where $h_{TR}(t)$ is given by Eq. (B.11). By defining $h_n \triangleq h(nT)$ as the nth sample of the fading process and recalling that $h_{TR}(nT) = \delta_n$, the received samples can be modelled as

$$\begin{aligned} r_n &= \sum_{m=0}^{N} v_m h_n \delta_{n-m} + w_n \\ &= v_n h_n + w_n \ . \end{aligned} \tag{B.26}$$

The discrete-time noise process, $\{w_n\}$, has the same properties as for the AWGN channel model, and the discrete-time fading process, $\{h_n\}$, is modelled as a complex Gaussian random process, with zero mean and an autocorrelation function of

$$\phi_m^{(F)} \triangleq \frac{1}{2}\mathbf{E}\left[h_{n-m}^* h_n\right] = \frac{1}{2}\mathbf{E}\left[h^*\big((n-m)T\big)h\big(nT\big)\right] = \phi_F(mT) \tag{B.27}$$

where $\phi_F(mT)$ is the autocorrelation function of the continuous-time fading process, as described in Section 4.1.

B.3 Rayleigh Frequency-Selective Fading

For the frequency-selective fading channel model $y_l(t)$ is given by Eq. (B.2c). Substituting this case into Eq. (B.7) yields received samples of

$$\begin{aligned} r_n &= \int_{-\infty}^{\infty} y_l(nT-\alpha)h_R(\alpha)\ d\alpha \\ &= \int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty} v(nT-\alpha-\tau)h(\tau;nT-\alpha)\ d\tau\right]\ h_R(\alpha)\ d\alpha + w_n \ . \end{aligned} \tag{B.28}$$

Substituting Eq. (2.5) for the transmitted lowpass signal, $v(t)$, gives

$$\begin{aligned} r_n &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[\sum_{m=0}^{N} v_m h_T(nT-\alpha-\tau-mT)\right] h(\tau;nT-\alpha)h_R(\alpha)\ d\tau\ d\alpha + w_n \\ &= \sum_{m=0}^{N} v_m \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h_T\big((n-m)T-\alpha-\tau\big)h(\tau;nT-\alpha)h_R(\alpha)\ d\tau\ d\alpha + w_n \ . \end{aligned} \tag{B.29}$$

Making the same assumption regarding the slow variation of the fading as was made for the frequency-flat case implies $h(\tau;nT-\alpha) \cong h(\tau;nT)$ for small $|\alpha|$. Therefore,

$$\begin{aligned} r_n &\cong \sum_{m=0}^{N} v_m \int_{-\infty}^{\infty} h(\tau;nT)\int_{-\infty}^{\infty} h_T\big((n-m)T-\alpha-\tau\big)h_R(\alpha)\ d\alpha\ d\tau + w_n \\ &= \sum_{m=0}^{N} v_m \int_{-\infty}^{\infty} h(\tau;nT)h_{TR}\big((n-m)T-\tau\big)d\tau + w_n \ , \end{aligned} \tag{B.30}$$

where $h_{TR}(t)$ is given by Eq. (B.11). By defining

$$h_{n,k} \triangleq \int_{-\infty}^{\infty} h(\tau; nT) h_{TR}(kT - \tau)\, d\tau \tag{B.31}$$

the received samples can be expressed as

$$\begin{aligned} r_n &= \sum_{m=0}^{N} v_m h_{n,n-m} + w_n \\ &= \sum_{k=-\infty}^{\infty} v_{n-k} h_{n,k} + w_n \ , \end{aligned} \tag{B.32}$$

where v_n is taken equal to zero for $n \notin \{0, 1, \ldots, N\}$. The discrete-time noise process, $\{w_n\}$, has the same properties as for the AWGN channel. The random variable $h_{n,k}$ is the response of the discrete-time channel at time index n to an impulse applied at time index $n - k$. In practice, the channel has a finite effective delay spread, so $h_{n,k}$ is negligible for $k < 0$ and $k \geq L$, for some L. Therefore,

$$r_n = \sum_{k=0}^{L-1} v_{n-k} h_{n,k} + w_n \ , \tag{B.33}$$

where, for the purpose of modelling the channel for decoder design, the error due to the approximation is ignored. For any delay, k, the sequence $h_{0,k}, h_{1,k}, \ldots, h_{N,k}$ is modelled as a zero-mean, stationary, complex Gaussian discrete-time random process. The cross-correlation function for the discrete-time channel impulse response is

$$\begin{aligned} \phi_{k,l;m}^{(S)} &\triangleq \frac{1}{2}\mathbf{E}\left[h_{n+m,k} h_{n,l}^*\right] \\ &= \frac{1}{2}\mathbf{E}\left[\int_{-\infty}^{\infty} h\Big(\tau_1; (n+m)T\Big) h_{TR}(kT - \tau_1)\, d\tau_1 \int_{-\infty}^{\infty} h^*\Big(\tau_2; nT\Big) h_{TR}^*(lT - \tau_2)\, d\tau_2\right] \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{1}{2}\mathbf{E}\left[h\Big(\tau_1; (n+m)T\Big) h^*\Big(\tau_2; nT\Big)\right] \\ &\qquad \times\ h_{TR}(kT - \tau_1) h_{TR}^*(lT - \tau_2)\, d\tau_2\, d\tau_1 \ . \end{aligned} \tag{B.34}$$

Substituting Eq. (5.2) for the cross-correlation function of the continuous-time channel impulse response yields

$$\begin{aligned} \phi_{k,l;m}^{(S)} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left[\phi_F(mT)\phi_M(\tau_1)\delta(\tau_2 - \tau_1)\right] h_{TR}(kT - \tau_1) h_{TR}^*(lT - \tau_2)\, d\tau_2\, d\tau_1 \\ &= \phi_m^{(F)} \int_{-\infty}^{\infty} \phi_M(\tau) h_{TR}(kT - \tau) h_{TR}^*(lT - \tau)\, d\tau \ , \end{aligned} \tag{B.35}$$

where $\phi_m^{(F)} \triangleq \phi_F(mT)$, and $\phi_M(\tau)$ is the multipath intensity profile of the continuous-time channel.

Appendix C

Notes on the Conditional Probability Density Function

FOR each of the three channel models, this Appendix contains a derivation of the conditional pdf $f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right)$, manipulated into a form suitable for illustrating how channel estimation is implicitly performed through the use of these pdfs. Each channel model is considered separately.

C.1 The Additive White Gaussian Noise Channel

For the AWGN channel the received samples are modelled, as described in Section 3.2, as

$$r_n = v_n e^{j\phi_c} + w_n \ . \tag{C.1}$$

Since the noise sample, w_n, has a complex Gaussian distribution, with the real and imaginary parts independent and identically distributed, the pdf of r_n conditioned on the phase error, ϕ_c, and the hypothesis $\underline{v} = \underline{\tilde{v}}$, is

$$\begin{aligned} f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \phi_c\right) &= f\left(r_n \,\middle|\, v_n = \tilde{v}_n, \phi_c\right) \\ &= \frac{1}{\pi \mathcal{N}_0} \exp\left\{-\frac{1}{\mathcal{N}_0}\left|r_n - \tilde{v}_n e^{j\phi_c}\right|^2\right\} \ , \end{aligned} \tag{C.2}$$

and, because the noise samples are independent,

$$\begin{aligned} f\left(\underline{r}_0^n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \phi_c\right) &= \prod_{m=0}^{n} f\left(r_m \,\middle|\, \underline{v} = \underline{\tilde{v}}, \phi_c\right) \\ &= \left(\frac{1}{\pi \mathcal{N}_0}\right)^{n+1} \exp\left\{-\frac{1}{\mathcal{N}_0}\sum_{m=0}^{n}\left|r_m - \tilde{v}_m e^{j\phi_c}\right|^2\right\} \ . \end{aligned} \tag{C.3}$$

The dependence on ϕ_c can easily be removed with

$$f\left(\underline{r}_0^n \,\middle|\, \underline{v} = \underline{\tilde{v}}\right) = \frac{1}{2\pi}\int_0^{2\pi} f\left(\underline{r}_0^n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \phi_c\right)\, d\phi_c \tag{C.4}$$

since ϕ_c is uniformly distributed over $[0, 2\pi)$. By using the identity

$$I_0\left(2|\epsilon|\right) = \frac{1}{2\pi}\int_0^{2\pi} \exp\left\{\epsilon e^{-j\phi} + \epsilon^* e^{j\phi}\right\}\, d\phi \tag{C.5}$$

where $I_0(\bullet)$ is the modified Bessel function of order zero, Eq. (C.3) can be manipulated to

$$\begin{aligned}
f\left(\underline{r}_0^n \,\middle|\, \underline{v} = \underline{\tilde{v}}\right) &= \frac{1}{2\pi}\int_0^{2\pi}\left(\frac{1}{\pi\mathcal{N}_0}\right)^{n+1} \exp\left\{-\frac{1}{\mathcal{N}_0}\sum_{m=0}^{n}\left|r_m - \tilde{v}_m e^{j\phi_c}\right|^2\right\}\, d\phi_c \\
&= \left(\frac{1}{\pi\mathcal{N}_0}\right)^{n+1} \exp\left\{-\frac{1}{\mathcal{N}_0}\sum_{m=0}^{n}\left[|r_m|^2 + \left|\tilde{v}_m e^{j\phi_c}\right|^2\right]\right\} \\
&\qquad \times\frac{1}{2\pi}\int_0^{2\pi} \exp\left\{\frac{1}{\mathcal{N}_0}\sum_{m=0}^{n}\left[r_m\tilde{v}_m^* e^{-j\phi_c} + r_m^*\tilde{v}_m e^{j\phi_c}\right]\right\}\, d\phi_c \\
&= \left(\frac{1}{\pi\mathcal{N}_0}\right)^{n+1} \exp\left\{-\frac{1}{\mathcal{N}_0}\sum_{m=0}^{n}\left[|r_m|^2 + \mathcal{E}_s\right]\right\} \\
&\qquad \times\frac{1}{2\pi}\int_0^{2\pi} \exp\left\{\left(\frac{1}{\mathcal{N}_0}\sum_{m=0}^{n} r_m\tilde{v}_m^*\right) e^{-j\phi_c} + \left(\frac{1}{\mathcal{N}_0}\sum_{m=0}^{n} r_m\tilde{v}_m^*\right)^* e^{j\phi_c}\right\}\, d\phi_c \\
&= K_n I_0\left(\frac{2}{\mathcal{N}_0}\left|\sum_{m=0}^{n} r_m\tilde{v}_m^*\right|\right) ,
\end{aligned} \tag{C.6}$$

where $K_n \triangleq \left(\frac{1}{\pi\mathcal{N}_0}\right)^{n+1} \exp\left\{-\frac{1}{\mathcal{N}_0}\sum_{m=0}^{n}[|r_m|^2 + \mathcal{E}_s]\right\}$ is a quantity that does not depend on the hypothesis. By substituting

$$\tilde{x}_n \triangleq \frac{1}{\mathcal{E}_s}\sum_{m=0}^{n} r_m\tilde{v}_m^* \tag{C.7}$$

and multiplying the argument to the Bessel function by $\frac{1}{\sqrt{\mathcal{E}_s}}|\tilde{v}_n| = 1$, Eq. (C.6) becomes

$$f\left(\underline{r}_0^n \,\middle|\, \underline{v} = \underline{\tilde{v}}\right) = K_n I_0\left(\frac{2}{\mathcal{N}_0}\left|\mathcal{E}_s\tilde{x}_n\right|\frac{|\tilde{v}_n|}{\sqrt{\mathcal{E}_s}}\right) = K_n I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|\tilde{x}_n\tilde{v}_n\right|\right) . \tag{C.8}$$

From Eq. (C.8), the desired conditional pdf can by found by using Bayes' rule as

$$f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right) = \frac{f\left(\underline{r}_0^n \,\middle|\, \underline{v} = \underline{\tilde{v}}\right)}{f\left(\underline{r}_0^{n-1} \,\middle|\, \underline{v} = \underline{\tilde{v}}\right)} = K_n'\frac{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|\tilde{x}_n\tilde{v}_n\right|\right)}{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|\tilde{x}_{n-1}\tilde{v}_{n-1}\right|\right)} , \tag{C.9}$$

where

$$K_n' \triangleq \frac{K_n}{K_{n-1}} = \frac{1}{\pi\mathcal{N}_0}\exp\left\{-\frac{1}{\mathcal{N}_0}\left[|r_n|^2 + \mathcal{E}_s\right]\right\} . \tag{C.10}$$

The conditional pdf can be manipulated further by observing that

$$\begin{aligned}\tilde{x}_n\tilde{v}_n &= \frac{1}{\mathcal{E}_s}\left(r_n\tilde{v}_n^* + \sum_{m=0}^{n-1} r_m\tilde{v}_m^*\right)\tilde{v}_n \\ &= \frac{1}{\mathcal{E}_s}r_n\left|\tilde{v}_n\right|^2 + \left(\frac{1}{\mathcal{E}_s}\sum_{m=0}^{n-1} r_m\tilde{v}_m^*\right)\tilde{v}_n \\ &= r_n + \tilde{x}_{n-1}\tilde{v}_n\ ,\end{aligned} \tag{C.11}$$

so

$$f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right) = K_n' \frac{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|r_n + \tilde{x}_{n-1}\tilde{v}_n\right|\right)}{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|\tilde{x}_{n-1}\tilde{v}_{n-1}\right|\right)}\ . \tag{C.12}$$

The role played by $\tilde{x}_{n-1}$ in Eq. (C.12) is quite interesting. In fact, if the hypothesis is correct then $\frac{1}{n}\tilde{x}_{n-1} \cong e^{j\phi_c}$. This can be shown by noting that, from Eq. (C.1),

$$\mathbf{E}\left[r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \phi_c\right] = \mathbf{E}\left[r_n \,\middle|\, v_n = \tilde{v}_n, \phi_c\right] = \tilde{v}_n e^{j\phi_c} \tag{C.13}$$

since the noise has zero mean. Therefore,

$$\begin{aligned}\mathbf{E}\left[\frac{1}{n}\tilde{x}_{n-1} \,\middle|\, \underline{v} = \underline{\tilde{v}}, \phi_c\right] &= \frac{1}{n}\mathbf{E}\left[\frac{1}{\mathcal{E}_s}\sum_{m=0}^{n-1} r_m\tilde{v}_m^* \,\middle|\, \underline{v} = \underline{\tilde{v}}, \phi_c\right] \\ &= \frac{1}{n}\frac{1}{\mathcal{E}_s}\sum_{m=0}^{n-1}\mathbf{E}\left[r_m \,\middle|\, \underline{v} = \underline{\tilde{v}}, \phi_c\right]\tilde{v}_m^* \\ &= \frac{1}{n}\frac{1}{\mathcal{E}_s}\sum_{m=0}^{n-1}\tilde{v}_m e^{j\phi_c}\tilde{v}_m^* \\ &= \frac{1}{n}\sum_{m=0}^{n-1} e^{j\phi_c} = e^{j\phi_c}\ .\end{aligned} \tag{C.14}$$

Thus, $\frac{1}{n}\tilde{x}_{n-1}$ is an unbiased estimator of $e^{j\phi_c}$, based on all the previous samples, and conditioned on $\underline{v}$.

In summary, the conditional pdf is

$$f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right) = K_n' \frac{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|r_n + \tilde{x}_{n-1}\tilde{v}_n\right|\right)}{I_0\left(\frac{2\sqrt{\mathcal{E}_s}}{\mathcal{N}_0}\left|\tilde{x}_{n-1}\tilde{v}_{n-1}\right|\right)}\ . \tag{C.15}$$

where K_n' is a scale factor that does not depend on the hypothesis and

$$\tilde{x}_{n-1} = \frac{1}{\mathcal{E}_s}\sum_{m=0}^{n-1} r_m\tilde{v}_m^* \tag{C.16}$$

is used to estimate the carrier phase error.

C.2 Rayleigh Frequency-Flat Fading

For the Rayleigh frequency-flat fading channel a useful expression for the conditional pdf $f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right)$ was given in [28]. Following their approach, the expression is derived here using the complex variable notation of this thesis.

As described in Section 4.2, the received samples for the frequency-flat fading channel are modelled as

$$r_n = v_n h_n + w_n \ , \tag{C.17}$$

where $\{h_n\}$ is the fading process, which has a zero-mean complex Gaussian distribution, as does the noise process, $\{w_n\}$.

To find an expression for the conditional pdf $f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right)$, it is useful to first consider a change of variables. Define

$$\tilde{u}_m \triangleq \frac{1}{\sqrt{\mathcal{E}_s}} r_m \tilde{v}_m^* \tag{C.18}$$

for all $m \in \{0, 1, \ldots, n\}$. These zero-mean complex Gaussian random variables have covariances

$$\begin{aligned} \phi_{a,b}^{(U)} &\triangleq \frac{1}{2}\mathbf{E}\left[\tilde{u}_a \tilde{u}_b^* \,\middle|\, \underline{v} = \underline{\tilde{v}}\right] \\ &= \frac{1}{2}\mathbf{E}\left[\frac{1}{\sqrt{\mathcal{E}_s}} r_a \tilde{v}_a^* \frac{1}{\sqrt{\mathcal{E}_s}} r_b^* \tilde{v}_b \,\middle|\, \underline{v} = \underline{\tilde{v}}\right] \ . \end{aligned} \tag{C.19}$$

Substituting $r_n = v_n h_n + w_n$ from Eq. (C.17) yields

$$\begin{aligned} \phi_{a,b}^{(U)} &= \frac{1}{\mathcal{E}_s}\frac{1}{2}\mathbf{E}\left[\left(\tilde{v}_a h_a + w_a\right)\tilde{v}_a^*\left(\tilde{v}_b^* h_b^* + w_b^*\right)\tilde{v}_b \,\middle|\, \underline{v} = \underline{\tilde{v}}\right] \\ &= \frac{1}{\mathcal{E}_s}\frac{1}{2}\mathbf{E}\left[\left(\mathcal{E}_s h_a + w_a \tilde{v}_a^*\right)\left(\mathcal{E}_s h_b^* + w_b^* \tilde{v}_b\right) \,\middle|\, \underline{v} = \underline{\tilde{v}}\right] \\ &= \frac{1}{\mathcal{E}_s}\frac{1}{2}\mathbf{E}\left[\mathcal{E}_s^2 h_a h_b^* + \mathcal{E}_s h_a w_b^* \tilde{v}_b + \mathcal{E}_s h_b^* w_a \tilde{v}_a^* + w_a w_b^* \tilde{v}_a^* \tilde{v}_b \,\middle|\, \underline{v} = \underline{\tilde{v}}\right] \ . \end{aligned} \tag{C.20}$$

Since the noise and fading processes are independent, and each has zero mean,

$$\begin{aligned} \phi_{a,b}^{(U)} &= \mathcal{E}_s \frac{1}{2}\mathbf{E}\left[h_a h_b^*\right] + \frac{1}{\mathcal{E}_s}\frac{1}{2}\mathbf{E}\left[w_a w_b^*\right]\tilde{v}_a^* \tilde{v}_b \\ &= \mathcal{E}_s \phi_{a-b}^{(F)} + \frac{1}{\mathcal{E}_s}\phi_{a-b}^{(W)} \tilde{v}_a^* \tilde{v}_b \ , \end{aligned} \tag{C.21}$$

where $\phi_m^{(F)}$ and $\phi_m^{(W)}$ are the autocorrelation functions of the fading and noise processes, respectively. Using Eq. (4.8) for $\phi_m^{(W)}$ leads to

$$\begin{aligned}\phi_{a,b}^{(U)} &= \mathcal{E}_s \phi_{a-b}^{(F)} + \frac{1}{\mathcal{E}_s}\left(\frac{\mathcal{N}_0}{2}\delta_{a-b}\right)\tilde{v}_a^* \tilde{v}_b \\ &= \mathcal{E}_s \phi_{b-a}^{(F)} + \frac{\mathcal{N}_0}{2}\delta_{b-a} \ .\end{aligned} \tag{C.22}$$

Note that the covariance does not depend on the hypothesis, $\underline{\tilde{v}}$.

An expression for the conditional pdf $f\left(\tilde{u}_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{\tilde{u}}_0^{n-1}\right)$ can be found by applying the results of Appendix C.3. In that section a generic expression for the conditional pdf $f\left(z_n \,\middle|\, \underline{z}_0^{n-1}\right)$ is derived, where $\underline{z}_0^n$ is any sequence of $(n+1)$ zero-mean complex Gaussian random variables. The derivation is limited to the case where the real and imaginary parts of $\underline{z}_0^n$ are identically distributed, with the additional property that $\mathbf{E}\left[\mathbf{Re}\left\{z_a\right\}\mathbf{Im}\left\{z_b\right\}\right] = -\mathbf{E}\left[\mathbf{Im}\left\{z_a\right\}\mathbf{Re}\left\{z_b\right\}\right]$ for all $a, b \in \{0, 1, \ldots, n\}$. The random variables $\underline{\tilde{u}}_0^n$ conditioned on $\underline{v}$ possess these properties, the proof of which is straightforward but tedious, and has been omitted.

By using $\phi_{a,b}^{(U)}$ in place of $\phi_{a,b}^{(Z)}$ in Appendix C.3, the conditional pdf $f\left(\tilde{u}_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{\tilde{u}}_0^{n-1}\right)$ is given by Eq. (C.41) as

$$f\left(\tilde{u}_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{\tilde{u}}_0^{n-1}\right) = \frac{1}{2\pi\sigma_n^2}\exp\left\{-\frac{1}{2\sigma_n^2}\left|\tilde{u}_n - \sum_{m=0}^{n-1} c_{n,m}^* \tilde{u}_m\right|^2\right\} \tag{C.23}$$

where, from Eq. (C.42), $\{c_{n,0}, c_{n,1}, \ldots, c_{n,n-1}\}$ are the solution to

$$\begin{bmatrix} \mathcal{E}_s\phi_0^{(F)} + \frac{\mathcal{N}_0}{2} & \mathcal{E}_s\phi_1^{(F)} & \cdots & \mathcal{E}_s\phi_{n-1}^{(F)} \\ \mathcal{E}_s\phi_1^{(F)} & \mathcal{E}_s\phi_0^{(F)} + \frac{\mathcal{N}_0}{2} & \cdots & \mathcal{E}_s\phi_{n-2}^{(F)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{E}_s\phi_{n-1}^{(F)} & \mathcal{E}_s\phi_{n-2}^{(F)} & \cdots & \mathcal{E}_s\phi_0^{(F)} + \frac{\mathcal{N}_0}{2} \end{bmatrix} \begin{bmatrix} c_{n,0} \\ c_{n,1} \\ \vdots \\ c_{n,n-1} \end{bmatrix} = \begin{bmatrix} \mathcal{E}_s\phi_n^{(F)} \\ \mathcal{E}_s\phi_{n-1}^{(F)} \\ \vdots \\ \mathcal{E}_s\phi_1^{(F)} \end{bmatrix}, \tag{C.24}$$

and, from Eq. (C.43), the variance is

$$\sigma_n^2 = \mathcal{E}_s\phi_0^{(F)} + \frac{\mathcal{N}_0}{2} - \sum_{m=0}^{n-1} c_{n,m}\mathcal{E}_s\phi_{n-m}^{(F)} = \frac{\mathcal{N}_0}{2} + \mathcal{E}_s\left(\phi_0^{(F)} - \sum_{m=0}^{n-1} c_{n,m}\phi_{n-m}^{(F)}\right) \ . \tag{C.25}$$

The conditional pdf $f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right)$ can be found by substituting Eq. (C.18) for $\tilde{u}_n$ in Eq. (C.23), giving

$$f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right) = \frac{1}{2\pi\sigma_n^2} \exp\left\{-\frac{1}{2\sigma_n^2}\left|\frac{1}{\sqrt{\mathcal{E}_s}} r_n \tilde{v}_n^* - \sum_{m=0}^{n-1} c_{n,m}^* \frac{1}{\sqrt{\mathcal{E}_s}} r_n \tilde{v}_n^*\right|^2\right\} . \tag{C.26}$$

The benefits of making the change of variables before applying the results of Appendix C.3 are that, since $\phi_{a,b}^{(U)}$ does not depend on the hypothesis, $\underline{\tilde{v}}$, neither do the coefficients, $\{c_{n,m}\}$ or the variance, σ_n^2. This simplifies the evaluation of the pdf because the coefficients do not need to be calculated for each hypothesis.

The pdf can be further manipulated by defining

$$\hat{\tilde{h}}_n \triangleq \frac{1}{\mathcal{E}_s} \sum_{m=0}^{n-1} c_{n,m}^* r_m \tilde{v}_m^* \tag{C.27}$$

and multiplying the exponent by $\frac{1}{\mathcal{E}_s}\left|\tilde{v}_n\right|^2 = 1$, giving

$$\begin{aligned} f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}\right) &= \frac{1}{2\pi\sigma_n^2} \exp\left\{-\frac{1}{2\sigma_n^2}\left|\frac{1}{\sqrt{\mathcal{E}_s}} r_n \tilde{v}_n^* - \sqrt{\mathcal{E}_s}\hat{\tilde{h}}_n\right|^2 \left(\frac{|\tilde{v}_n|}{\sqrt{\mathcal{E}_s}}\right)^2\right\} \\ &= \frac{1}{2\pi\sigma_n^2} \exp\left\{-\frac{1}{2\sigma_n^2}\left|r_n - \tilde{v}_n \hat{\tilde{h}}_n\right|^2\right\} \end{aligned} \tag{C.28}$$

The role of $\hat{\tilde{h}}_n$ in Eq. (C.28) is quite interesting, as illustrated by a comparison with the corresponding pdf conditioned on the fading process as well. Since the noise is Gaussian and $r_n = v_n h_n + w_n$,

$$\begin{aligned} f\left(r_n \,\middle|\, \underline{v} = \underline{\tilde{v}}, \underline{r}_0^{n-1}, \underline{h}\right) &= f\left(r_n \,\middle|\, v_n = \tilde{v}_n, h_n\right) \\ &= \frac{1}{2\pi\left(\frac{\mathcal{N}_0}{2}\right)} \exp\left\{-\frac{1}{2\left(\frac{\mathcal{N}_0}{2}\right)}\left|r_n - \tilde{v}_n h_n\right|^2\right\} . \end{aligned} \tag{C.29}$$

The only differences between Eq. (C.28) and this equation are the $\hat{\tilde{h}}_n$ is used in place of h_n and the variance is σ_n^2 instead of $\mathcal{N}_0/2$. This highlights the fact that $\hat{\tilde{h}}_n$ is used as an estimate of h_n. In fact, $\hat{\tilde{h}}_n$ is the minimum mean squared error (MMSE) predictor of h_n based on the n prior observations, $\underline{r}_0^{n-1}$, conditioned on the hypothetical data symbols. The

mean squared prediction error is

$$\frac{1}{2}\mathbf{E}\left[\left|h_n - \hat{\tilde{h}}_n\right|^2\right] = \phi_0^{(F)} - \sum_{m=0}^{n-1} c_{n,m}\phi_{n-m}^{(F)} \ . \tag{C.30}$$

C.3 The Conditional Probability Distribution of Complex Gaussian Random Variables

Consider a sequence of $(n+1)$ zero-mean complex Gaussian random variables, denoted by $\underline{z}_0^n$, and suppose that the real and imaginary parts of $\underline{z}_0^n$ are identically distributed. Denote the covariance between any two of the random variables by $\phi_{a,b}^{(Z)} \triangleq \frac{1}{2}\mathbf{E}\left[z_a z_b^*\right]$. If $\underline{z}_m \triangleq [z_0\ z_1\ \cdots\ z_m]^T$ is column vector containing the first $(m+1)$ of these random variables, then the covariance matrix for $\underline{z}_0^n$ is given by

$$\underline{\underline{\Phi}}_n \triangleq \frac{1}{2}\mathbf{E}\left[\underline{z}_n \underline{z}_n^H\right] = \begin{bmatrix} \phi_{0,0}^{(Z)} & \cdots & \phi_{0,n-1}^{(Z)} & \phi_{0,n}^{(Z)} \\ \vdots & \ddots & \vdots & \vdots \\ \phi_{n-1,0}^{(Z)} & \cdots & \phi_{n-1,n-1}^{(Z)} & \phi_{n-1,n}^{(Z)} \\ \phi_{n,0}^{(Z)} & \cdots & \phi_{n,n-1}^{(Z)} & \phi_{n,n}^{(Z)} \end{bmatrix} , \tag{C.31}$$

where the superscript H denotes conjugate transpose. The multivariate complex Gaussian pdf is[1]

$$f(\underline{z}_0^n) = \frac{1}{(2\pi)^{n+1}\det \underline{\underline{\Phi}}_n} \exp\left\{-\frac{1}{2}\underline{z}_n^H \underline{\underline{\Phi}}_n^{-1} \underline{z}_n\right\} , \tag{C.32}$$

where $\underline{\underline{\Phi}}_n^{-1}$ and $\det \underline{\underline{\Phi}}_n$ are the inverse and determinant of matrix $\underline{\underline{\Phi}}_n$, respectively.

The Levinson-Durbin Algorithm provides a recursive means for calculating the inverse and determinant of the covariance matrix. Suppose $\underline{\underline{\Phi}}_{n-1}^{-1}$ and $\det \underline{\underline{\Phi}}_{n-1}$ have already been

[1] Strictly speaking, this pdf is valid only if, $\forall a, b \in \{0, 1, \ldots, n\}$, $\mathbf{E}[z_a] = 0$, $\mathbf{E}[\mathbf{Re}\{z_a\}\mathbf{Re}\{z_b\}] = \mathbf{E}[\mathbf{Im}\{z_a\}\mathbf{Im}\{z_b\}]$, and $\mathbf{E}[\mathbf{Re}\{z_a\}\mathbf{Im}\{z_b\}] = -\mathbf{E}[\mathbf{Im}\{z_a\}\mathbf{Re}\{z_b\}]$. These conditions are all met with the random variables to which this equation is applied in this thesis.

computed for the covariance matrix $\underline{\underline{\Phi}}_{n-1} \triangleq \frac{1}{2}\mathbf{E}\left[\underline{z}_{n-1}\underline{z}_{n-1}^H\right]$. Then

$$\underline{\underline{\Phi}}_n^{-1} = \begin{bmatrix} & & 0 \\ & \underline{\underline{\Phi}}_{n-1}^{-1} & \vdots \\ & & 0 \\ 0 & \cdots & 0\ 0 \end{bmatrix} + \frac{1}{\sigma_n^2} \begin{bmatrix} \\ \underline{c}_n \\ \\ -1 \end{bmatrix} \begin{bmatrix} \underline{c}_n^H & -1 \end{bmatrix} \tag{C.33}$$

and

$$\det \underline{\underline{\Phi}}_n = \sigma_n^2 \det \underline{\underline{\Phi}}_{n-1} \ , \tag{C.34}$$

where the coefficients $\underline{c}_n \triangleq [c_{n,0}\ c_{n,1}\ \ldots\ c_{n,n-1}]^T$ are given by

$$\begin{bmatrix} c_{n,0} \\ c_{n,1} \\ \vdots \\ c_{n,n-1} \end{bmatrix} = \underline{\underline{\Phi}}_{n-1}^{-1} \begin{bmatrix} \phi_{0,n} \\ \phi_{1,n} \\ \vdots \\ \phi_{n-1,n} \end{bmatrix} , \tag{C.35}$$

and

$$\sigma_n^2 = \phi_{n,n} - \sum_{m=0}^{n-1} c_{n,m}\phi_{n,m} \ . \tag{C.36}$$

To find an alternate expression for the joint pdf, $f(\underline{z}_0^n)$, observe that by substituting

Eq. (C.33) for $\underline{\underline{\Phi}}_n^{-1}$,

$$
\begin{aligned}
\underline{z}_n^H \underline{\underline{\Phi}}_n^{-1} \underline{z}_n &= \begin{bmatrix} \underline{z}_{n-1}^H & z_n^* \end{bmatrix} \underline{\underline{\Phi}}_n^{-1} \begin{bmatrix} \underline{z}_{n-1} \\ z_n \end{bmatrix} \\
&= \begin{bmatrix} \underline{z}_{n-1}^H z_n^* \end{bmatrix} \begin{bmatrix} & & 0 \\ & \underline{\underline{\Phi}}_{n-1}^{-1} & \vdots \\ & & 0 \\ 0 & \cdots & 0\,0 \end{bmatrix} \begin{bmatrix} \underline{z}_{n-1} \\ \\ z_n \end{bmatrix} + \frac{1}{\sigma_n^2} \begin{bmatrix} \underline{z}_{n-1}^H z_n^* \end{bmatrix} \begin{bmatrix} \underline{c}_n \\ -1 \end{bmatrix} \begin{bmatrix} \underline{c}_n^H & -1 \end{bmatrix} \begin{bmatrix} \underline{z}_{n-1} \\ z_n \end{bmatrix} \\
&= \underline{z}_{n-1}^H \underline{\underline{\Phi}}_{n-1}^{-1} \underline{z}_{n-1} + \frac{1}{\sigma_n^2} \left(\underline{z}_{n-1}^H \underline{c}_n - z_n^* \right) \left(\underline{c}_n^H \underline{z}_{n-1} - z_n \right) \\
&= \underline{z}_{n-1}^H \underline{\underline{\Phi}}_{n-1}^{-1} \underline{z}_{n-1} + \frac{1}{\sigma_n^2} \left| z_n - \underline{c}_n^H \underline{z}_{n-1} \right|^2 .
\end{aligned}
\tag{C.37}
$$

By substituting Eq. (C.37) for $\underline{z}_n^H \underline{\underline{\Phi}}_n^{-1} \underline{z}_n$ and Eq. (C.34) for $\det \underline{\underline{\Phi}}_n$ in Eq. (C.32), the joint pdf can be written as

$$
\begin{aligned}
f(\underline{z}_0^n) &= \frac{1}{(2\pi)^{n+1} \det \underline{\underline{\Phi}}_n} \exp\left\{ -\frac{1}{2} \underline{z}_n^H \underline{\underline{\Phi}}_n^{-1} \underline{z}_n \right\} \\
&= \frac{1}{(2\pi)^{n+1} \sigma_n^2 \det \underline{\underline{\Phi}}_{n-1}} \exp\left\{ -\frac{1}{2} \underline{z}_{n-1}^H \underline{\underline{\Phi}}_{n-1}^{-1} \underline{z}_{n-1} - \frac{1}{2\sigma_n^2} \left| z_n - \underline{c}_n^H \underline{z}_{n-1} \right|^2 \right\} .
\end{aligned}
\tag{C.38}
$$

By using Bayes' rule, the pdf of z_n conditioned on $\underline{z}_0^{n-1}$ is

$$
f\left(z_n \,\middle|\, \underline{z}_0^{n-1} \right) = \frac{f(\underline{z}_0^n)}{f\left(\underline{z}_0^{n-1} \right)} . \tag{C.39}
$$

Substituting Eq. (C.38) for the numerator and using Eq. (C.32) for the denominator, the conditional pdf can be expressed as

$$
\begin{aligned}
f\left(z_n \,\middle|\, \underline{z}_0^{n-1} \right) &= \frac{\frac{1}{(2\pi)^{n+1} \sigma_n^2 \det \underline{\underline{\Phi}}_{n-1}} \exp\left\{ -\frac{1}{2} \underline{z}_{n-1}^H \underline{\underline{\Phi}}_{n-1}^{-1} \underline{z}_{n-1} - \frac{1}{2\sigma_n^2} \left| z_n - \underline{c}_n^H \underline{z}_{n-1} \right|^2 \right\}}{\frac{1}{(2\pi)^n \det \underline{\underline{\Phi}}_{n-1}} \exp\left\{ -\frac{1}{2} \underline{z}_{n-1}^H \underline{\underline{\Phi}}_{n-1}^{-1} \underline{z}_{n-1} \right\}} \\
&= \frac{1}{2\pi\sigma_n^2} \exp\left\{ -\frac{1}{2\sigma_n^2} \left| z_n - \underline{c}_n^H \underline{z}_{n-1} \right|^2 \right\} .
\end{aligned}
\tag{C.40}
$$

In summary, the conditional pdf is given by

$$
f\left(z_n \,\middle|\, \underline{z}_0^{n-1} \right) = \frac{1}{2\pi\sigma_n^2} \exp\left\{ -\frac{1}{2\sigma_n^2} \left| z_n - \sum_{m=0}^{n-1} c_{n,m}^* z_m \right|^2 \right\} \tag{C.41}
$$

where $\{c_{n,m}\}$ are the solution to

$$\begin{bmatrix} \phi_{0,0}^{(Z)} & \phi_{0,1}^{(Z)} & \cdots & \phi_{0,n-1}^{(Z)} \\ \phi_{1,0}^{(Z)} & \phi_{1,1}^{(Z)} & \cdots & \phi_{1,n-1}^{(Z)} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{n-1,0}^{(Z)} & \phi_{n-1,1}^{(Z)} & \cdots & \phi_{n-1,n-1}^{(Z)} \end{bmatrix} \begin{bmatrix} c_{n,0} \\ c_{n,1} \\ \vdots \\ c_{n,n-1} \end{bmatrix} = \begin{bmatrix} \phi_{0,n}^{(Z)} \\ \phi_{1,n}^{(Z)} \\ \vdots \\ \phi_{n-1,n}^{(Z)} \end{bmatrix} , \tag{C.42}$$

$$\sigma_n^2 = \phi_{n,n}^{(Z)} - \sum_{m=0}^{n-1} c_{n,m} \phi_{n,m}^{(Z)} , \tag{C.43}$$

and

$$\phi_{a,b}^{(Z)} = \frac{1}{2} \mathbf{E} \left[z_a z_b^* \right] . \tag{C.44}$$

Appendix D

Notes on Computer Simulation

EXTENSIVE computer simulation was used to generate the bit error rate performance graphs presented in this thesis. In this appendix some comments on various aspects of the implementation and execution of the simulations are given. In general, analogue systems cannot be exactly replicated by digital systems such as a computer, so some approximations to the channel models must be made. The generation of the AWGN samples, the implementation of the transmit and receive filters, and the generation of the fading process are all inexactly represented by the simulator, although the approximations are all very minor. These approximations are discussed in the following. In addition, a description of the Monte Carlo technique used for estimating the bit and word error rates is discussed.

D.1 The Additive White Gaussian Noise

For the AWGN channel the discrete-time channel model given in Appendix B.1 is exact, so no approximations regarding the generation of the data-bearing signal are required. The complex AWGN samples are generated using the Box-Muller method. Each noise sample is generated with

$$w_n = \sqrt{-\mathcal{N}_0 \log X_1} \exp\{j2\pi X_2\} \ , \tag{D.1}$$

where X_1 and X_2 are samples of a random variable uniformly distributed over $[0, 1)$. These are generated using the UNIX function `drand48()`, which uses the linear congruential algo-

rithm for pseudo-randomly generating 48-bit integers. The 48 bits are used as the mantissa of a C language `double` type.

D.2 Transmit and Receive Filters

The root raised-cosine filters were implemented using fast fourier transforms (FFT's). To fulfill Nyquist's sampling criterion, two samples per symbol were used. The impulse response of the filter was truncated to $2M_T + 1$ samples, with $M_T = 65$. The block length used for the FFT, N', was found by rounding $2(N + 1) + 4M_T$ up to the nearest power of two, where N is the number of transmitted symbols per message word. This block size provides space for sufficient zero-padding to permit filtering the signal twice (once for the transmit filter and once for the receive filter), without causing any wrap-around distortion when circular convolution is performed.

The output of the transmit filter is generated by inserting a zero between every transmitted symbol, then padding the block with additional zeros up to the block size, N'. The FFT is then applied to block, and the result is multiplied by the FFT'd version of the truncated and zero-padding impulse response. The product is inverse FFT'd, yielding the samples of the transmitted signal. The fading is then applied to this signal. The receive filtering is applied to the result, and the signal is sampled at the symbol rate (i.e. every second sample is discarded).

D.3 Rayleigh frequency-flat fading

The frequency-flat fading process is generated with Jakes' method [35],

$$h(nT_s) = \sum_{i=0}^{N_0} A_i \cos\left(2\pi f_i n T_s + \theta_i\right) \ , \tag{D.2}$$

where θ_i is uniformly distributed over $[0, 2\pi)$ and remains constant for the entire block. The coefficients

$$A_i = \begin{cases} \sqrt{\frac{2}{2*N_0+1}}\,, & \text{for } i = 0 \\ \sqrt{\frac{4}{2*N_0+1}} \exp\left\{ j \frac{\pi i}{N_0+1} \right\}, & \text{for } i \in \{1, 2, \ldots, N_0\}\,, \end{cases} \tag{D.3}$$

and

$$f_i = B_d * \cos\left(\frac{\pi i}{2 * N_0 + 1}\right) \tag{D.4}$$

for $i \in \{0, 1, \ldots, N_0\}$. The number of terms in the summation, N_0, is chosen as $N_0 = 8$.

D.4 Error Rate Calculations

The error rates for a specific SNR are estimated by repeatedly transmitting messages words across the channel. Each message word is transmitted and received entirely independently of the others. The number of transmitted words is denoted by N_w. The estimated word error rate is found by dividing the number of words in which bit errors occurred by N_w. Because the word errors occur independently, the word error rate can be estimated accurately provided sufficient word errors are counted. At least 10 word errors are need before results have statistical significance, but many more word errors were counted in all the simulations that were performed.

The bit error rate is estimated by dividing the number of bit errors counted by the number of transmitted message bits ($N_a N_w$). Because the bit errors do not occur independently with coded systems, accurately estimating the BER is more problematic than estimating the WER. Statistical analysis of the experimental uncertainty would require knowledge of the probability distribution of the number of bit errors in a message word, and estimating this distribution requires considerably more experimentation than is needed to estimate the average number of bit errors in a word.

In the absence of statistical analysis, care has been made to ensure accuracy in the

results, following the basic principle of making the graphs look smooth. Any irregularities where either smoothed out by simulating additional transmissions, or verified by transmitting considerably more messages. The total number of transmitted messages was dependent on the word size, but for a word size of about one thousand message bits used for most of the simulations, the following techniques were typically use. For BER's above 10^{-2}, one thousand message words were transmitted (1 million bits). For BER's in the range of 10^{-5} to 10^{-2}, the same number of simulations were run for each SNR point, and the simulations were repeated until at least 20 messages words were received in error[1]. were received in error at the SNR point with a BER of just below 10^{-5}. This approach helps to generate smooth curves, because the same simulation data was used for each point. When comparisons were to be made between different curves which lay close to one another, the same number of simulations were done for both curves (the maximum of the two required for each curve individually). This helps to justify the superiority of one curve over the other, since the results were based on the same simulation data. Typically, at this block size between 10,000 and 200,000 simulations. This translates into roughly 10 million to 200 million transmitted bits. If the bit errors occurred independently, the 95confidence interval at a BER of 10^{-5} after 10,000 simulations would be $[0.81 \times 10^{-5}, 1.21 \times 10^{-5}]$. Because the bit error do not occur independently (and in fact are very burst-like for the concatenated code), the confidence intervals are not nearly this tight, even after 200,000 simulations. However, because of the steep drop-off in the performance of the concatenated code as the SNR increases, accurately estimating the BER at some SNR in the region of the drop-off is not necessary. This is because a large difference in the BER translates into a very small difference in the SNR. As a result, even though there may be some question as to the accuracy of the BER results,

[1] In some situations, notably for the frequency-selective fading channel, an even larger number of word errors was required before acceptable stability in the BER results was achieved. Typically, between 100 and 500 word errors were sufficient.

one can be confident that the performance in terms of SNR is sufficiently accurate for the comparisons and discussions in this thesis.