ITERATIVE ESTIMATION AND DECODING FOR CHANNELS WITH MEMORY

by

Joseph Hyukjoon Kang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering: Systems)
in The University of Michigan
1999

Doctoral Committee:
Professor Wayne E. Stark, Chair
Professor Pinaki Mazumder
Assistant Professor Kimberly M. Wasserman
Associate Professor Kim A. Winick

ABSTRACT

ITERATIVE ESTIMATION AND DECODING FOR CHANNELS WITH MEMORY

by

Joseph Hyukjoon Kang

Chair: Professor Wayne E. Stark

In this thesis, iterative estimation and decoding techniques are considered for channels with memory. Much of the work is focused on the design and performance of turbo codes, a code with an iterative decoding algorithm, in frequency-hopped spread spectrum (FH-SS) systems where the data rate is slower than the hop rate. In such a system, multiple bits are transmitted over each hop. If the channel conditions can be assumed to be static over the duration of a hop, then estimation techniques can be utilized to estimate unknown channel parameters. In particular, iterative channel estimation schemes are considered in conjunction with an iterative decoding algorithm for the turbo code. Channels considered include the partial-band interference channel, the Rayleigh fading channel, and the channel with both partial-band interference and Rayleigh fading. In addition to the systems with FH-SS, a more general channel

with memory, the Gilbert-Elliot burst channel model, is considered. The channel estimation scheme paired with the powerful error correction capabilities of turbo codes are shown to yield significant performance improvements with respect to other well-known coding schemes.

To my parents, siblings, wife, and Allison.

ACKNOWLEDGEMENTS

First, I would like to thank God, maker of heaven and earth. Nothing in this world occurs outside His will. Thus, when I say that I have been blessed with a wonderful family, caring friends, and a beautiful, loving wife, I acknowledge that it is He who has provided for me. I pray that the knowledge and skills which I have acquired over the years will be used for His glory.

Next, I would like to thank Professor Wayne Stark, my thesis advisor, for his support and guidance in the field of communications. I came to Michigan as a graduate student intending to specialize in two-dimensional signal processing. Professor Stark, however, taught my first classes in communications and ignited my interest in the field. His expertise in digital communications has been and will always be a tremendous asset to me.

I would also like to thank the other members of my committee, Professor Winick, Professor Wasserman, and Professor Mazumder for their help in making this thesis as complete as possible. This thesis would not have been possible without their instruction and assistance over the years.

I would also like to acknowledge the support which makes this work possible. First, I would like to thank the Department of Electrical Engineering at the University of

Michigan for the fellowship which supported my studies during my initial year. Next, I would like to thank the Army Research Office and the Defense Advanced Research Projects Agency (DARPA) for their support in subsequent years. The Army Research Office provided support under grants DAAH04-96-1-0110, DAAH04-96-1-0377, and DAAH04-95-1-0246; DARPA provided support under grant SRA-588510732.

Finally, I would like to thank my family and friends who have both challenged and supported me throughout the years. To my parents, I thank you for all the love and sacrifices you have made for me. As a young elementary boy, I remember promising that I would take my studies all the way and receive a Ph.D. I am glad that I kept the promise. To my sister and brother, thanks for putting up with me over all the years and continuing to love me for who I am. To my parents-in-law and Eugene, thanks for the love and support over the years. Finally, I can take care of Susan the way I always wanted. To my friends at Presby, I appreciate the fellowship we have shared over the years, whether it was for trivial pursuit, WBC, or just hanging out. One day, we will beat the undergrads! To my friends of Room 4427, thanks for the daily lunches, night baseball, tolerating all my processes, and the memories of IM football, softball, and basketball. Before Michigan, I was fortunate to be on teams which rarely lost. Thanks to Mike's passing, John's batting, Troy's fielding, and Paul's shooting, I have learned the humility of defeat. To my high school and college friends, thanks for keeping in touch over the years. I'm coming back East, so get ready. Finally, I would like to thank Susan and the little one, Allison. You are my source of inspiration, my foundation for courage, my understanding of love. Please, just stay the way you are and we will grow happily old together.

TABLE OF CONTENTS

LIST OF TABLES

Table

LIST OF FIGURES

LIST OF APPENDICES

APPENDIX

CHAPTER 1

Introduction

1.1 The Goal of Communications

A communication system provides the framework which allows two users to both transmit and receive information over a channel. If the users wish to communicate without requirements on location, one system of interest is wireless communications. In general, the channel distorts the transmitted information in such a way that the received signal is different from the transmitted one. Depending on the type and degree of such distortions, the transmitted information may be received incorrectly. The goal of the communications engineer is to design a system which improves the reliability of such wireless transmissions.

One method of improving the error performance of communication between two users is to increase the power of the transmitted signal. In such cases, the receiver can more easily determine the content of the communicated message. However, there can be substantial costs which motivate why other methods of improving communications

are usually considered. One cost is the resulting interference to other users within a system, for the case of a multiuser system. While the intended target receives the message more easily, the amplified message may interfere with the communications between other users. Another cost is that in the case of the mobile user, expending too much power on signal transmission will result in shorter battery life.

An alternative way to mitigate the effects of channel distortion is to use error control coding. Essentially, by introducing structured redundancy in the data stream, the receiver can reduce its susceptibility to channel distortion. Hence, to achieve the same performance criteria, error control coding can provide several decibels (dB) of signal-to-noise ratio (SNR) gain over uncoded systems. It has been said that each decibel saved is worth one million dollars. In some cases, the gains of error control coding over the uncoded case can exceed 30-40 decibels.

Error control coding is a relatively young discipline. Its roots are often traced to Shannon, who proved in his seminal 1948 paper [1] that data communication with arbitrarily low error rates can be achieved over a noisy channel by properly encoding the information. The astonishing result of Shannon's work is that such limits can be obtained without sacrificing the rate of information transmission. While Shannon's paper demonstrates the existence of such error control codes, it does not specify how to build them. As a result, a major portion of the work in wireless communications focuses on developing codes that achieve performance near the Shannon limit with reasonable complexity.

One form of error control coding is the recently developed turbo codes. First published in 1993, turbo codes have been shown to yield performance near the Shannon

limit. While turbo codes are not the first codes to achieve results near this fundamental limit, turbo codes are exciting because they achieve such performance with reasonable complexity. For such reasons, turbo codes are the focus of this thesis.

While the advantages of error control coding were described above, it is important to understand the framework of a wireless communication system in order to gauge the importance of coding. In the next section, the building blocks of a generic communication system is discussed.

1.2 Building Blocks of a Communication System

The transmission of information over a wireless channel incurs distortion which alters the transmitted signal in such a way that the information may be received incorrectly. A well designed communications system can significantly improve the reliability of such transmissions. A typical communications system is shown in Figure 1.1.

The information source dictates what information is to be communicated. The output of the information source is transformed by the source encoder to a binary sequence, $\mathbf{d}$, which would ideally represent the information in as few binary digits as possible. The output of the source encoder, called the information sequence, is then passed to the channel encoder. The channel encoder introduces controlled redundancy into the data in such a manner that errors caused by the channel can be corrected. Especially useful for channels which exhibits bursts of errors, the goal of the channel interleaver is to permute the coded bits such that a manner that the channel appears

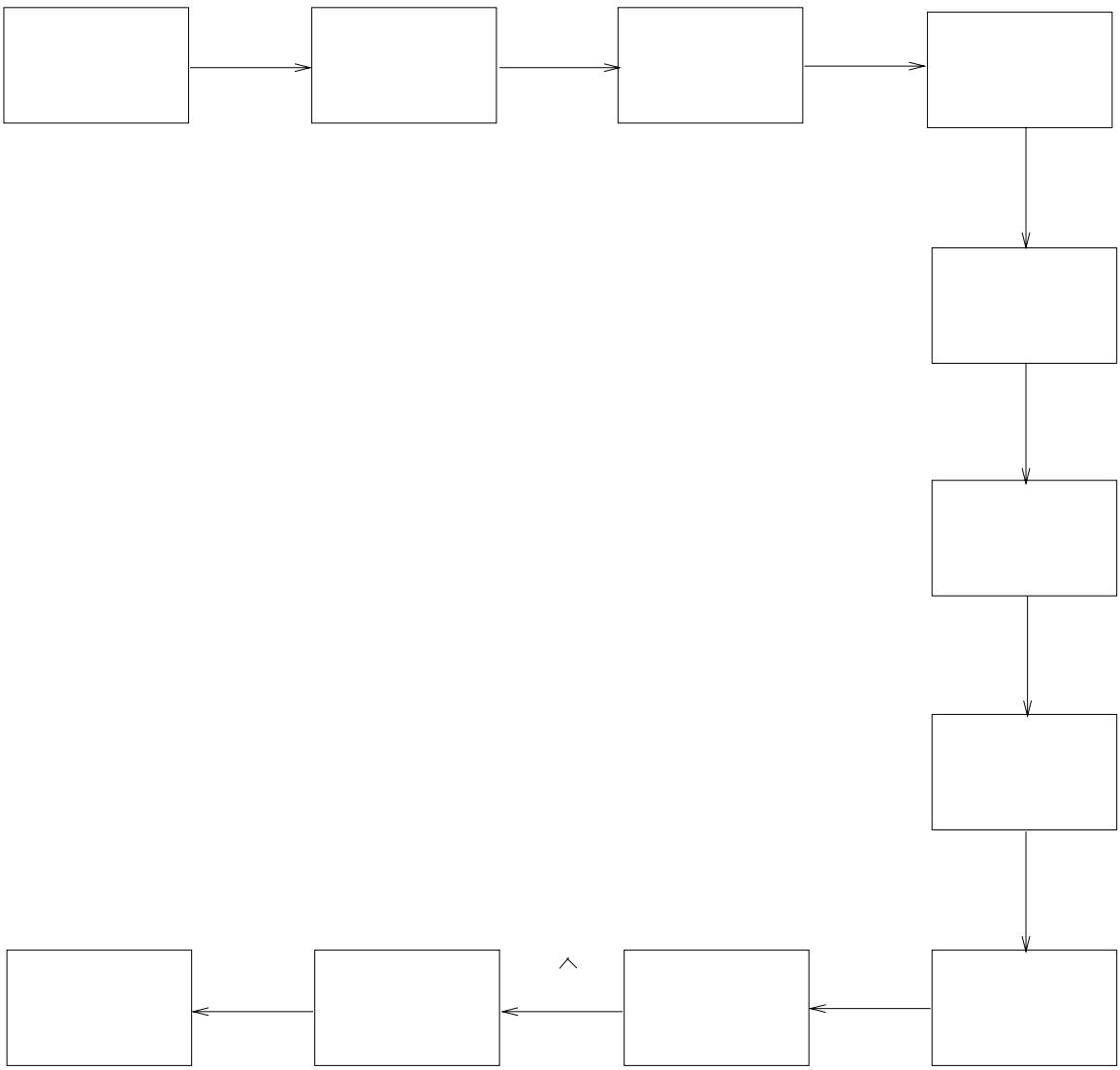

Figure 1.1: Block Diagram of a Generic Communication System

memoryless. The modulator maps each output symbol of the channel encoder to a set of analog waveforms which are suitable to be transmitted over the channel.

The channel is the physical medium through which the signals are transmitted. While wireless channels might include free space, other channels of interest include telephone lines, fiber optic cables, and magnetic tapes. The primary feature of the channel is that the transmitted signal may be corrupted by factors such as thermal noise, multipath fading, and partial-band jamming. Additive white Gaussian noise (AWGN) is often used to model thermal noise and is especially pertinent for deep space communications. Multipath fading occurs for communications over a terrain which contains objects between the transmitter and receiver. These objects reflect the transmitted signal, causing multiple paths with different phases and delays. The differing phases cause the incoming signals to add constructively or destructively, thus resulting in signal attenuation at the receiver. Finally, jammers describe the group of hostile communicators which attempt to disrupt the communications of targeted users by transmitting a signal over the same communication range.

The demodulator processes the channel output and provides an estimate of the modulated signal. The channel deinterleaver inverses the permutation operation of the channel interleaver and restores the coded data to its original order. The channel decoder reverses the operations performed by the channel encoder and provides an estimate of the data sequence, $\hat{\mathbf{d}}$. The source decoder transforms the estimated sequence $\hat{\mathbf{d}}$ into the most likely source sequence. Because of channel distortion, the received source sequence may be different from the original one.

Because the focus of this thesis is channel coding, the source encoder will not be

considered here. Instead, it will be grouped with the information source and together be considered as a digital source which produces a binary information stream which needs to be communicated.

Having described a generic type of communications, a specific type of communication system used for both commercial and military applications is discussed in the next section. In addition, the need for error control coding in this communication system is motivated.

1.3 Spread Spectrum Communications and Error Control Coding

One communication system of interest is spread spectrum communication. A spread spectrum system can be characterized as a system whose transmitted signal spectrum is much larger than the bandwidth of the information being transmitted. One form of spread spectrum is known as frequency-hopped spread spectrum (FH-SS). Suppose that the data rate is R bits per second and the transmitted power is P Watts. In FH-SS systems, the transmission bandwidth, W Hertz, is divided into q nonoverlapping frequency slots. After the signal is modulated to an intermediate frequency, a frequency hopper pseudorandomly changes the center frequencies of the carrier. In other words, at each point in time, information is transmitted over just one of the q frequency slots, where q may be very large. This is shown in Figure 1.2 where the vertical axis reflects the q frequency slots, the horizontal axis reflects the

transmission time with T_h as the hop duration, and the gray blocks reflect the slots where information was transmitted.

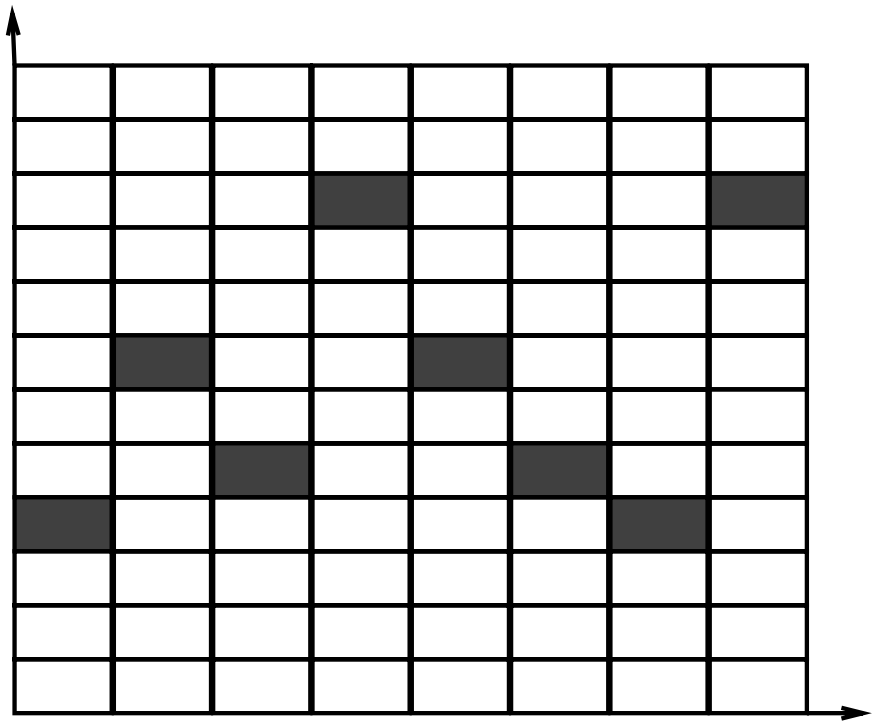

Figure 1.2: Frequency Hopping Pattern

There are many advantages to this seemingly wasteful allocation of bandwidth. First, such a system provides resistance to hostile interferers known as jammers whose primary goal is to interrupt communications. Essentially, by spreading the transmitted signal over a large bandwidth, the jammer is similarly forced to spread its power over a large bandwidth in order to guarantee that some hops are jammed. Another advantage is that spectrum-spectrum modulated signals have a lower probability of intercept (LPI) and detection (LPD). For instance, in the case of FH-SS, a larger frequency band must be monitored. Yet another advantage is the ability to allow several

users to simultaneously transmit over a common channel with a minimum amount of mutual interference (i.e. multiple access communications). One final advantage is that spreading the signal over a wide range of frequencies reduces the vulnerability to narrow band interference such as frequency-selective multipath fading channels.

In spite of the many gains associated with spread spectrum systems, the performance of these systems may not be sufficient. First consider the system with partial-band interference. Assuming that the jammer has fixed total power, J Watts, the jammer can either jam the entire bandwidth, W, with double-sided power spectral density, $\frac{J}{2W}$, or it can jam a fraction of the bandwidth, ρW, with double-sided power spectral density, $\frac{J}{2\rho W}$. An example of a partial-band jammer is shown in Figure 1.3 for $\rho = 4/q$.

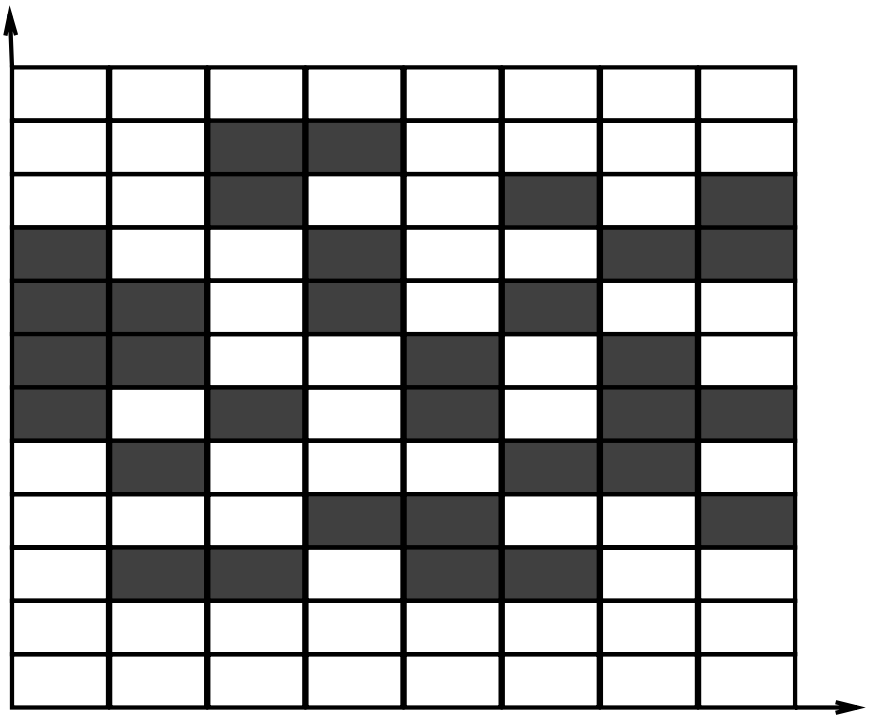

Figure 1.3: Partial Band Jammer

If the jammer chooses to jam the entire bandwidth, $\rho = 1$, it can be sure to corrupt every hop. However, if W is very large, the power spectral density of the jammer, $\frac{J}{2W}$, may be small and its effect on the transmitted bits may be minimal. If the jammer jams some fraction of the bandwidth, ρ, then it can have larger effect on the hops that are jammed, meanwhile having no effect on the unjammed hops. The intelligent jammer is one which chooses ρ to maximize the error rate of the system. The worst-case performance from such a jammer is often achieved by a partial-band jammer (especially at high E_b/N_0). Shown in Figure 1.4 is the bit error rate (BER) for full-band Gaussian noise jamming and worst-case partial-band Gaussian noise jamming [31] without thermal noise. The energy per bit is a function of the transmitted power and the data rate, $E_b = P/R$, and the double-sided power spectral density of the jammer is expressed as $\frac{N_J}{2} = \frac{J}{2W}$. The results assume that the modulation is binary frequency shift keying (BFSK) and the signal is noncoherently detected.

As shown in the figure, worst-case partial-band jamming is effective in significantly degrading performance. At BERs of 10^{-5}, the difference in performance is over 30 dB. Because the performance of worst-case partial-band jamming decreases inverse linearly for sufficiently large E_b/N_J, this difference will increase for increasingly stringent BER requirements. Hence, the performance of spread spectrum alone may be significantly degraded if worst-case partial-band jamming is applied.

Next, consider the performance of the Rayleigh fading channel in broadband Gaussian noise with double-sided power spectral density, $N_0/2$. The BER performance is shown in Figure 1.5 for the case where the signals are BFSK-modulated and the receiver applies noncoherent detection.

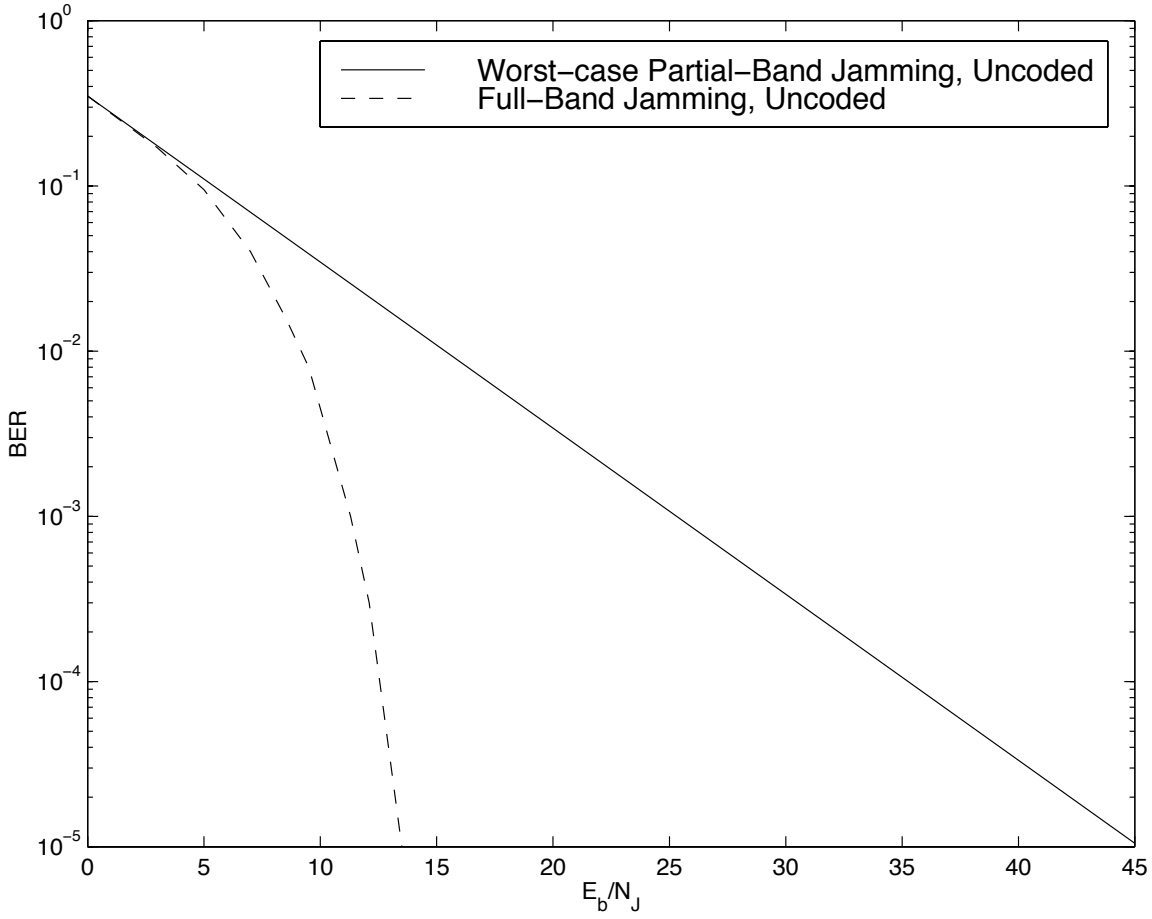

Figure 1.4: Performance Comparison of Full-Band Jamming and Worst-Case Partial-Band Jamming in Noncoherent BFSK

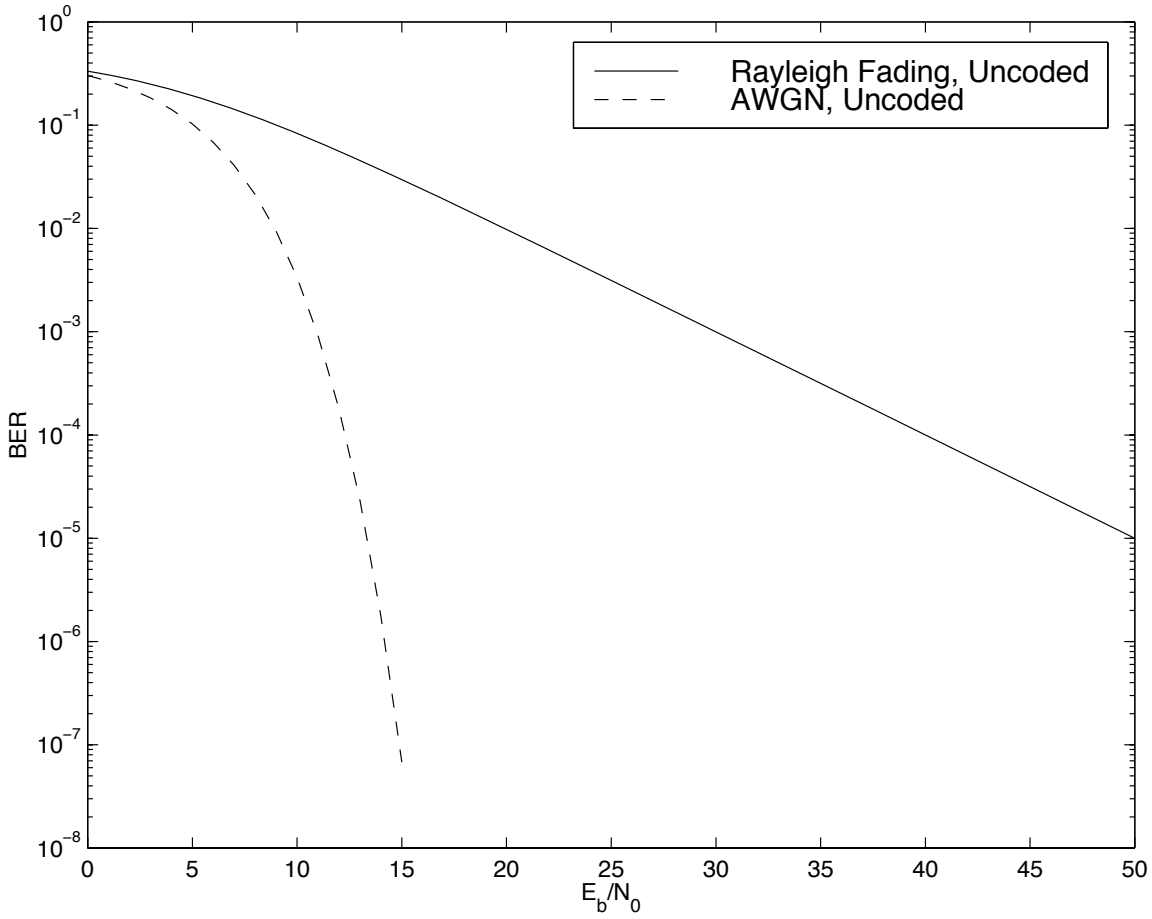

Figure 1.5: Performance Comparison of AWGN and Rayleigh Fading in Noncoherent BFSK

Similar to the previous result, the presence of fading leads to serious degradations in performance, with the Rayleigh faded result decreasing inverse linearly for large SNRs. At BERs of 10^{-5}, the degradation is approximately 35 dB with respect to the AWGN channel.

As mentioned in Section 1.2, there is a way to recover the performance losses associated with partial-band interference and fading. By introducing structured redundancy in the data, more errors introduced by the channel can be corrected. This procedure known as error control or error correction codes has been shown to mitigate the effects of channel distortions caused by partial-band interference and fading. As an example, consider a simple error control code called the repetition code. In this code, the information bits are repeated L times, resulting in a rate $1/L$ code. If an interleaver is used to guarantee that each of the L coded bits are transmitted over different frequencies and the receiver is assumed to know with certainty which hops were jammed, then an upper bound [31] to the performance of this system is shown in Figure 1.6.

For the repetition code with $L = 10$, the gain with respect to the uncoded case for worst-case jamming is nearly 30 dB at BERs of 10^{-5}. The most surprising result of Figure 1.6 is that such large gains can be accomplished with a very simple code. If a more powerful, but perhaps more complex code had been applied, the gains could be even greater.

Also plotted in Figure 1.6 is the Shannon limit which is the fundamental lower limit on the SNR for error-free communications by properly encoding the data. This limit is a function of code rate. The value plotted in Figure 1.6 represents the minimum E_b/N_0

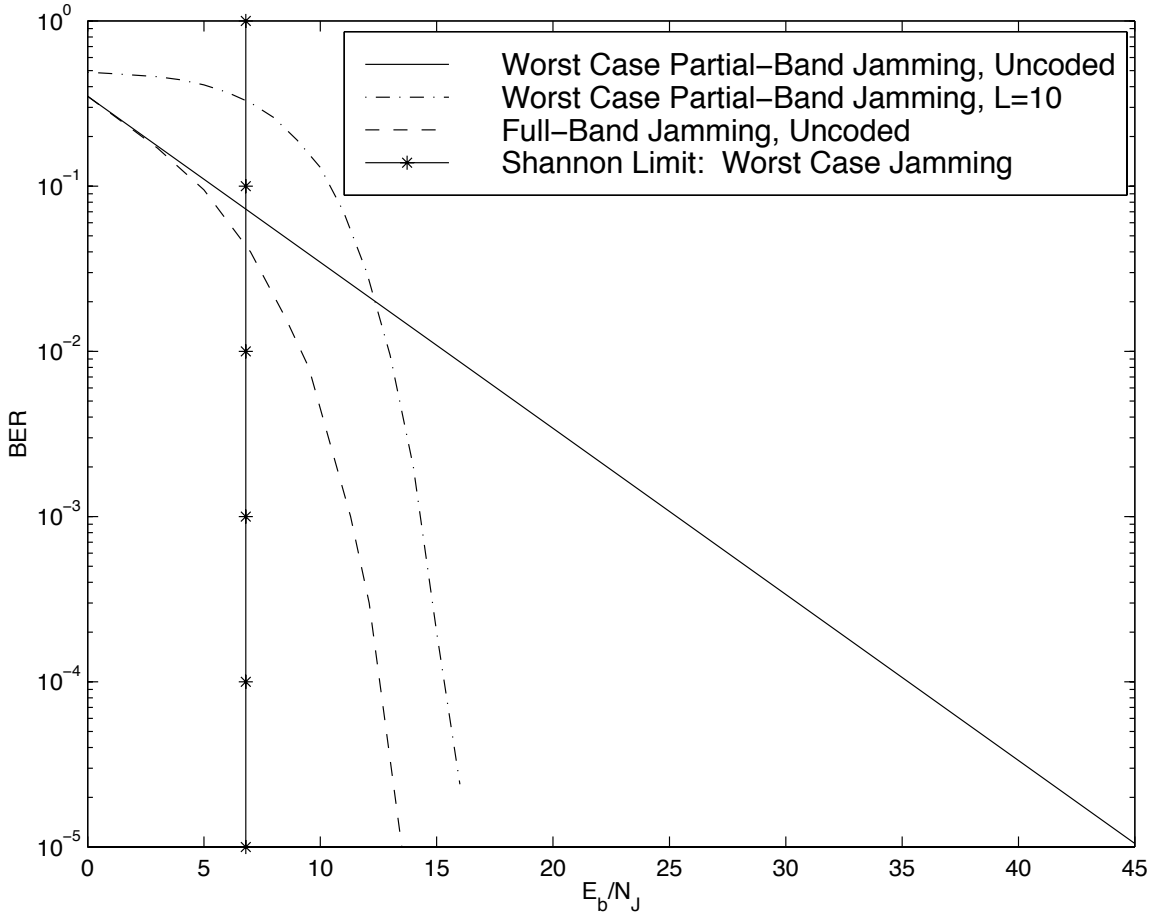

Figure 1.6: Gain of Error Control Codes for FH-SS with Partial-Band Interference

required to achieve error-free communications across all code rates. This minimum value is 6.71 dB and it occurs for a code rate of approximately 0.5. For this result, it is assumed that the decoder knows with certainty which hops have been jammed. If a code which achieved performance near the Shannon limit was considered, the gain over the repetition code with $L = 10$ would be an additional 10 dB.

Shown in Figure 1.7 are the analytical results for a repetition code in the Rayleigh fading channel. In the results, it is assumed that each coded bit is sent over a different hop and that the instantaneous fade amplitudes are independent from hop to hop. The instantaneous fade amplitudes are not assumed to be known. By using a simple repetition code of length $L = 10$, about 32 dB is gained at a BER of 10^{-5}. Also plotted is the Shannon limit, which indicates the minimum E_b/N_0 needed to achieve error-free performance when side information regarding the instantaneous fade amplitudes is unavailable. Again, this limit is shown at its minimum value across all code rates. This minimum value is about 8.1 dB and occurs at a code rate of approximately 0.22. If a more powerful code which achieves performance near the Shannon limit were applied, another 10 dB could be gained.

Thus far, the need for error control codes to improve bit error performance has been described. Even the use of a simple code like the repetition code was effective in significantly improving the performance when communicating over the partial-band interference and fading channels. In the next section, an error control code which yields performance near the fundamental limit is introduced.

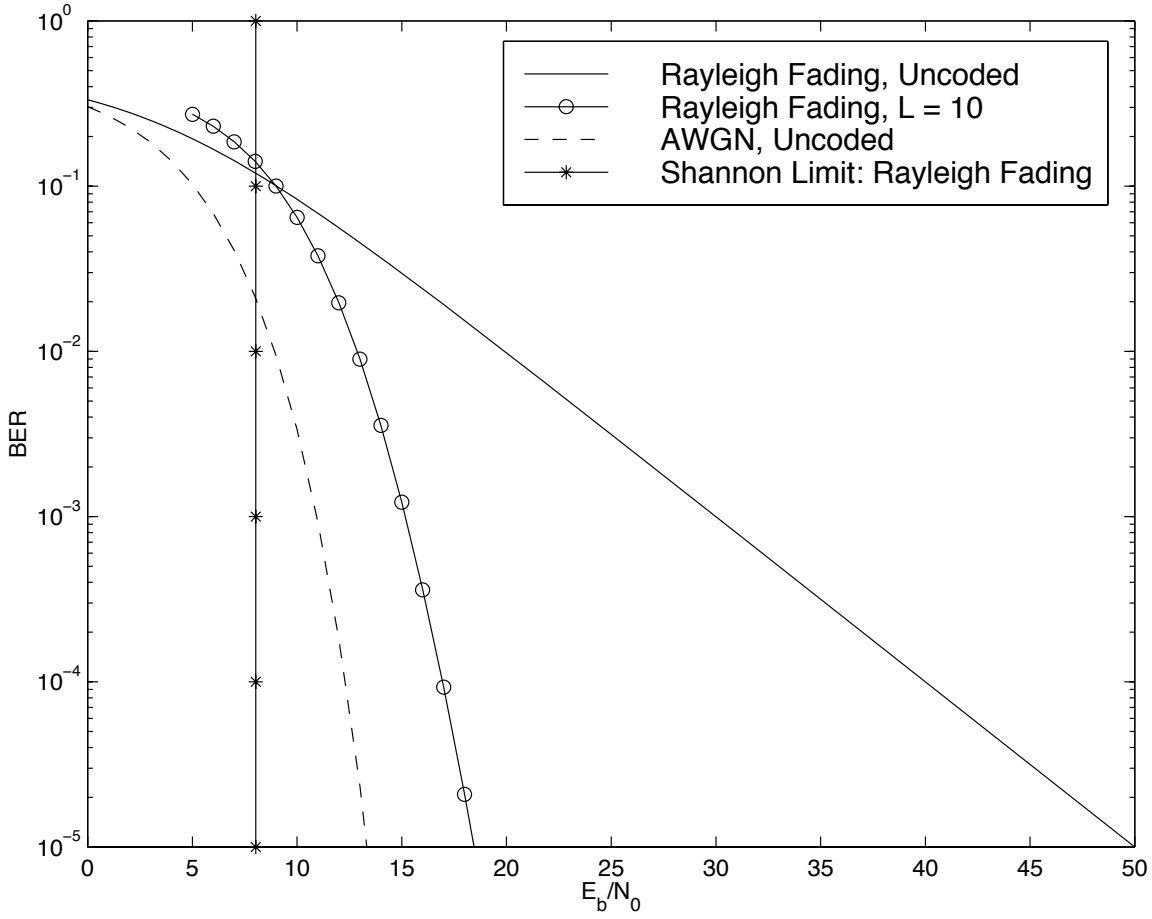

Figure 1.7: Gain of Error Control Codes for the Rayleigh Fading Channel

1.4 Turbo Codes: Near Shannon Limit Coding

Turbo codes are a recently developed channel coding scheme that has been shown to achieve data communication at signal-to-noise ratios close to the Shannon limit. The results published in the inaugural paper by Berrou et al [8] were so good that they were met with much skepticism by the coding community. Since then, however, these results have been reproduced and even improved [9]. Consequently, much of the present research is focused on applying turbo codes to different systems.

The most surprising element of turbo coding success is its simplicity. The encoder is formed using a parallel concatenation of two or more component encoders. If the input block has N information bits, the encoded bit stream is made up of the uncoded information bits and the parity bits of the component encoders. The key element of the encoder is the use of an interleaver which permutes the information sequence and then uses this as the input to the second component encoder. In general, this permutation allows low weight outputs of the first component encoder to result in high weight outputs of the second component encoder. Thus, the combination of the encoders might contain favorable distance properties, even if each component encoder does not.

It is well known that a randomly chosen code of sufficiently large block length is capable of approaching channel capacity [10]. In general, however, the complexity of maximum likelihood decoding such a code increases exponentially with block length up to the point where decoding becomes physically unrealizable. The encoder mimics random codes by making use of a large random interleaver. While turbo coding

performance also improves for increasing interleaver lengths, the decoding complexity grows only linearly, making the decoding of large block lengths possible. Note that a turbo decoder does not perform maximum likelihood decoding directly, but attempts to achieve maximum likelihood decoding in an iterative way. The original turbo decoder [8] used two decoders which each use the maximum *a posteriori* (MAP) algorithm. The MAP decoder is described in Chapter 2 and is also describedin the Appendix. There are other less complex algorithms that can be used in place of the MAP algorithm for each decoder such as SOVA and Max-Log-MAP [11]. However, because these other algorithms are suboptimal, they reduce the complexity of decoding at the cost of performance. Hence, for the purposes of this thesis, we will consider the turbo decoder where each component decoder uses the MAP algorithm to calculate *a posteriori* likelihood estimates for each bit.

The potential of turbo codes can be best exemplified by its successful application to deep space communications. Using MAP decoders, turbo codes (16 state constituent codes, overall rate 1/2) were shown to outperform the concatenated code of the Voyager, Galileo, and Cassini missions. The gain, for instance, over the Voyager code (rate 1/2, constraint length 7 convolutional code concatenated with a (255,223) Reed-Solomon code) is approximately 1.5 dB at a BER of 10^{-5}. The large coding gains of turbo codes exhibited over the complex deep space codes suggest that turbo codes should be considered for other wireless communication systems. The primary difference between the two codes, however, is that the turbo code has greater decoding complexity.

Thus, we arrive at the primary disadvantage of using turbo codes with the MAP

algorithm. Decoding complexity for turbo codes is proportional to the block length, the number of decoding iterations, and the constraint length of the constituent codes. The MAP decoder is approximately four times more complex than the Viterbi algorithm and this must be iterated several times.

It is well-known that turbo code performance generally increases with interleaver or block length [12]. In fact, Berrou *et al* showed a bit error rate of 10^{-5} at 0.7 dB, but needed to use 18 decoding iterations and a block length of $65,536$ bits. The large amount of computation required for turbo decoding explains why some of the current research is focused on reduced complexity decoders [11] [13] [14]. By also taking into consideration the processing delay inherent with large interleavers, it is easy to see why turbo codes which use the MAP algorithm may be unsuitable for voice communications [15].

In packet data communications, the use of error correction codes plays a key role in achieving low packet error rates. When transmitting speech, large processing delay is unacceptable and higher error rates are tolerable. In data communication, low error rates are more important and delay at the decoder is more acceptable. Therefore, it would appear that turbo codes are possible candidates for packet data communications. In much of this work, turbo codes are considered in a packet data communications system.

The motivation for research on turbo codes is clear. As a recently developed field which yields extremely good performance, it is important to investigate the performance of turbo codes in different systems. However, in this thesis, the iterative estimation of typically unknown channel parameters is also considered. Many coding

schemes require knowledge of the channel. While it is convenient to assume in the analysis that such information is perfectly available, often times this is an impractical assumption. In the next section, the importance of an iterative channel estimation scheme paired with the iterative decoding scheme of the turbo decoder is motivated.

1.5 Iterative Decoding and Iterative Channel Estimation

Channel coding schemes have generally been designed to increase the reliability of information transmission when the errors are statistically independent. However, there exists many channels such as multipath fading which exhibit bursts of errors. A common method for dealing with these bursts is to interleave the information in such a manner that the channel appears memoryless. Thus if interleaving is applied to a burst channel, a code devised for independent errors can be applied. Note, however, that such a scheme does not make use of the information inherent in the memory. Because multiple bits have been transmitted over similar channel conditions, it may be useful to estimate the channel state and use this information in the decoder.

While the gains associated with channel estimation are clear, the desire for iterative channel estimation must be motivated. Consider a receiver which makes bit decisions based on the log likelihood ratio

$$L = \log \frac{P(d_k = 1|\mathbf{y})}{P(d_k = 0|\mathbf{y})} \tag{1.1}$$

where d_k are the data bits and $\mathbf{y}$ represent the respective channel outputs.

If the *a posteriori* information bit probabilities, $p(d_k|\mathbf{y})$ (or bit estimates), can be improved, then the error performance will also be improved. One way to improve the bit estimates is to generate reliable information of the channel. For instance, consider the fading channel with diversity. If information regarding the instantaneous fade amplitudes are available to the receiver, then maximum ratio combining (MRC) is optimal. Without any information, the decoder may perform equal gain combining (EGC). It is well known [32] that MRC can yield large performance gains over EGC. Reliable knowledge of the channel improves the bit estimates and this leads to better performance. In a similar manner, channel estimates can be improved if reliable information of the data bits are available.

Turbo codes use an iterative decoding scheme where after each iteration, the bit error rate typically improves. In other words, because the turbo decoder makes bit decisions based on a log likelihood ratio like (1.1), the *a posteriori* bit probabilities or bit estimates also improve. For such reasons, it might be useful to estimate the channel after each iteration and run the next decoding iteration using the new channel estimates. In this way, better performance and faster convergence might be obtained. In this thesis, the joint task of iterative channel estimation and iterative decoding are considered for channels with memory.

1.6 Thesis Outline

The outline for the remainder of this thesis is as follows. In Chapter 2, turbo codes are reviewed, beginning with a description of the encoding and decoding operations. In addition, a survey of recent work on turbo codes is presented, including important design issues and describing how performance near the Shannon limit is attained.

In Chapter 3, the turbo codes are investigated in frequency-hop spread spectrum with partial-band interference. In addition, full-band thermal noise is present. The channel model is that of a partial-band jammer in which a fraction of the frequency band is jammed and the remaining fraction is unjammed. This chapter focuses on the implementation and performance of a modified turbo decoder for this model. The perfect knowledge regarding which hops are jammed is referred to as channel state information. Cases of known or unknown channel state and variable number of bits per hop are considered. The approach is to modify the calculation of branch transition probabilities inherent in the original turbo decoder. For the cases with no side information and multiple bits per hop, channel state estimates are iteratively calculated. Analytical bounds are derived and simulation is performed for coherent and noncoherent demodulation. The performance of turbo codes is compared with other well known codes such as the convolutional code, the Reed-Solomon code, and the concatenated code comprised of a convolutional inner code and Reed-Solomon outer code.

In Chapter 4, the performance of turbo codes is investigated in a frequency-hopped spread spectrum system with full-band thermal noise and Rayleigh fading. For cases

where the data rate exceeds the hopping rate (i.e. there exists multiple bits per hop) and there exists no side information about the fading amplitudes, the approach is to iteratively estimate the fading levels. Simulation is performed for coherent and noncoherent reception, variable number of bits per hop, and cases where fading side information is available or unavailable to the decoder. It is shown that iterative channel estimation performed in conjunction with iterative decoding can improve the overall decoding performance. Finally, the performance of a FH-SS system using standard fading assumptions is compared to the performance of a measured fading channel. Due to differences in assumptions, the measured fading channel yields performance several decibels higher than the ideal fading channel.

In Chapter 5, the performance of turbo codes are investigated in a frequency hopped spread spectrum system with partial-band interference, full-band thermal noise, and Rayleigh fading. Two types of channel side information are considered: knowledge that the transmitted bit is jammed or unjammed and knowledge of the instantaneous fading amplitudes. For the case with multiple bits per hop and no side information, the approach is to exploit the channel memory by computing channel state estimates for unknown parameters. Simulation is performed for coherent and noncoherent reception, variable number of bits per hop, and cases where side information is available or unavailable to the decoder.

In Chapter 6, turbo codes are investigated in a generic channel with memory, the Gilbert-Elliot burst channel model. The Gilbert-Elliot channel is a two state hidden Markov model where one state represents a bad channel state which typically has high error probabilities and the other state represents a good channel state which has

low error probabilities. For the burst channel model, turbo code calculations require knowledge of the hidden Markov state. As in previous chapters, the approach is to estimates unknown channel state parameters and use these in the turbo decoder. For cases with unknown channel state, the calculation of state estimates requires knowledge of the hidden Markov model (HMM) transition probabilities. When these probabilities are unknown, the Baum-Welch re-estimation procedure is used.

In Chapter 7, the robustness of turbo codes in different channels is considered. In previous chapters, it was assumed that certain parameters of the channel were known. For instance, in Chapter 3, it is assumed that the power spectral densities of the noise and the fractional bandwidth of jamming are known perfectly. In this chapter, the performance of turbo codes in the presence of mismatched channel parameters is investigated.

Finally, conclusions are drawn and topics for future work are discussed in Chapter 8.

CHAPTER 2

Turbo Codes

The emergence of turbo codes has had a strong effect on how coding theorists approach the design of codes. Traditionally, the major design objective for an error control code was to find maximize the minimum distance, d_{min}, amongst codes with comparable complexity. Yet despite having relatively low d_{min}, turbo codes have been shown to achieve bit error rates (BER) of 10^{-5} at SNRs smaller than codes with higher d_{min}.

The importance of minimum distance can be explained by applying the union bound to the AWGN channel. For the AWGN channel with double-sided power spectral density, $N_0/2$, the union bound on the word error probability for an (n, k) block code is

$$P_w \leq \sum_{d=d_{min}}^{n} A_d \, Q(\sqrt{2\, d\, R\, E_b/N_0}) \tag{2.1}$$

where $R = k/n$ is the rate of the code, E_b is the received energy per bit, and A_d is

the weight enumerator of the code.

Because $Q(x) \leq e^{-x^2/2}$ diminishes quickly for increasing values of x, (2.1) is dominated by the value of d_{min} when $E_s/N_0 = R\,E_b/N_0$ is large. If more than one code possesses the same value of d_{min}, then the code with smaller $A_{d_{min}}$ is chosen.

In the low SNR regime, however, weights greater than d_{min} may provide a strong contribution to the error rates because $Q(\cdot)$ falls off more slowly. In particular, if the values of A_d grow quickly enough in d, codewords with weight greater than d_{min} may contribute more to the error rate than lower weight codewords. For bit error rates, this effect is compounded if higher weight codewords have higher information weight. Hence, in the low SNR region, d_{min} may no longer dominate code performance. In other words for low SNRs, a code with smaller d_{min} may actually yield better performance because its distance profile beyond d_{min} is important to code performance.

The above discussion is not new. It is well known in coding theory that d_{min} dominates performance at high SNRs, but is not nearly as important for low SNRs. The difference is that most codes of interest yield their desired performance in the high SNR domain where d_{min} plays a key role. Those codes that do yield good performance in the low SNR region have prohibitively high decoding complexity [33]. Meanwhile, turbo codes with their relatively low d_{min} have been shown to yield good performance at low SNRs. In fact, turbo codes achieve performance near the Shannon limit and they do so with reasonable decoding complexity. In this chapter, turbo codes are described in greater detail, giving insights regarding their performance at low SNRs. The encoder and decoder are described in full detail. In addition, some design guidelines are presented to achieve the performance described in [8].

2.1 Turbo Encoder

The encoder is a parallel concatenation of two or more systematic constituent codes separated by interleavers. Systematic describes a code which takes a vector of data bits $\mathbf{d}$ as input to its encoder and then includes $\mathbf{d}$ among its encoder outputs. The advantage of using systematic codes for the component codes is that the effective rate of the code can be reduced if the systematic information is transmitted just once. The structure of a generalized turbo encoder is shown in Figure 2.1. The encoder takes as input the data sequence $\mathbf{d}$ of length N and then produces $M+1$ streams, each of length N: the information bits $\mathbf{d}$ and the parity bits of the M component encoders $\mathbf{p_1}$, $\mathbf{p_2}$, ..., $\mathbf{p_M}$. The overall rate of the code is $\frac{1}{M+1}$.

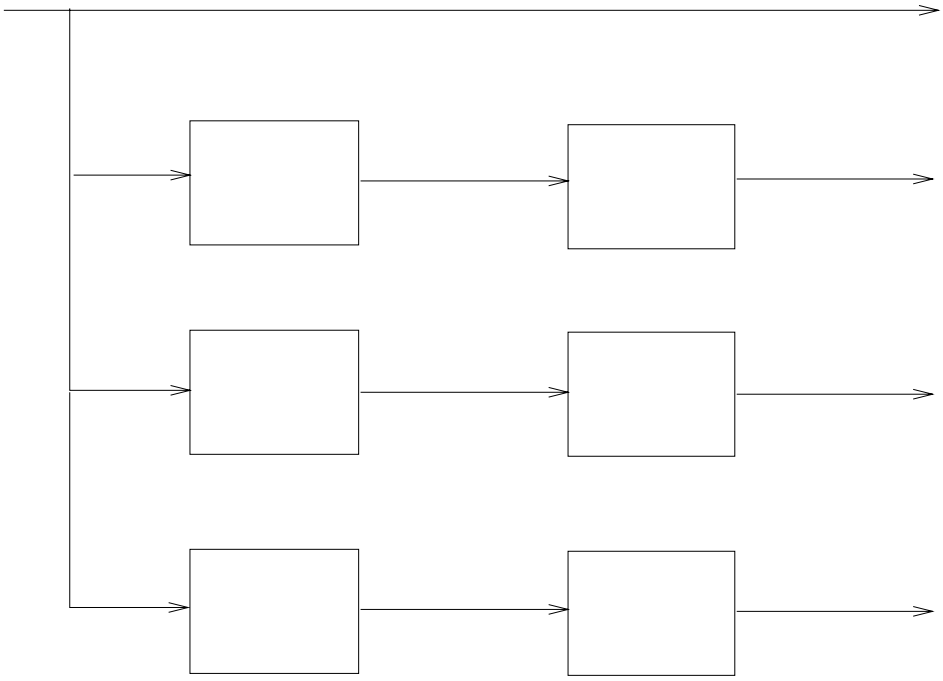

Figure 2.1: Block Diagram of the Turbo Encoder

If $M = 2$ and the constituent codes are constraint length K convolutional codes, a helical interleaver can be used to guarantee trellis termination [27]. The helical interleaver (also described in the Appendix) essentially constrains the trellises of both component codes to end in the same state, allowing a $K - 1$ bit tail to drive both codes to the all-zeros state.

In general, the constituent codes are recursive systematic convolutional (RSC) codes. The advantage of using such codes as opposed to conventional nonsystematic convolutional (NSC) codes and block codes can be explained by the so-called interleaver gain which was derived for the AWGN channel. If N is the size of the interleaver, then the interleaver gain was shown to be $N^{1-w_{min}}$ [12] where w_{min} is defined as the minimum weight input sequence of infinite length which results in a finite weight error event. For NSC and block codes, $w_{min} = 1$, whereas for the RSC code, $w_{min} = 2$. Hence, no interleaver gain exists for the NSC and block codes, whereas the interleaver gain for the RSC code goes as N^{-1}. The interpretation of this result is that for sufficiently large SNRs, the bit error performance of a turbo code with length N_1 will be a factor of N_1/N_2 better than a turbo code with length N_2 where $N_1 > N_2$. Hence, a length 10000 turbo code will perform approximately an order of magnitude better than a length 1000 turbo code.

There exists a more intuitive explanation for the relationship between the recursive systematic codes and the interleaver. If the component codes are assumed to be systematic, then the output codeword weight can be expressed as $i = w + z$, where i is the encoded weight, w is the weight of the systematic information, and z is the parity weight. Clearly, for systematic encoders, the input weight, w, is correlated

with the output weight, i. More importantly (because low weight codewords typically determine performance), low output weights are correlated with low input weights.

First, consider the parallel concatenated code which uses RSC codes as the constituent codes. For the RSC encoder, low weight input sequences can lead to high output weight sequences. For example, the input sequence with weight one and infinite length leads to an infinite weight output sequence. Hence, for $i = w+z$ and $w = 1$, the output codeword weight, i, is high despite a small weight contribution from the systematic bits because z is high. Of particular importance are input patterns which terminate the RSC code. These sequences are important, because self-terminating patterns often lead to low parity weights. If the self-terminating patterns also have low weight, then because the encoder is systematic, the resulting codeword will have low codeword weights which are undesirable. For the RSC code, however, these self-terminating patterns are generally specific. Thus, if an RSC code with input $\mathbf{d}$ yields a low weight output, then the same RSC code with interleaved $\mathbf{d}$ as input generally will not. As a result, even if each individual code is weak, the parallel concatenation of the codes separated by interleavers is quite powerful. The length of the interleaver is an important parameter for performance because the longer the length of the interleaver, the less chance of permuting to an input sequence which terminates the encoder.

Consider the example shown in Figure 2.2 which exhibits a parallel concatenated code consisting of RSC codes The minimum distance for each component code in Figure 2.2 is 5, 2, and 2, respectively. Thus, the worst case minimum distance for the concatenated code is 9. For each component code, the input sequence which yields

the minimum distance is (00...01110...00). Note, however, that the presence of the interleaver makes it unlikely that all three constituent codes will have an input of form (00...01110...00) when $\mathbf{d}$ is a weight three sequence. Hence, the worst case minimum distance will rarely be achieved.

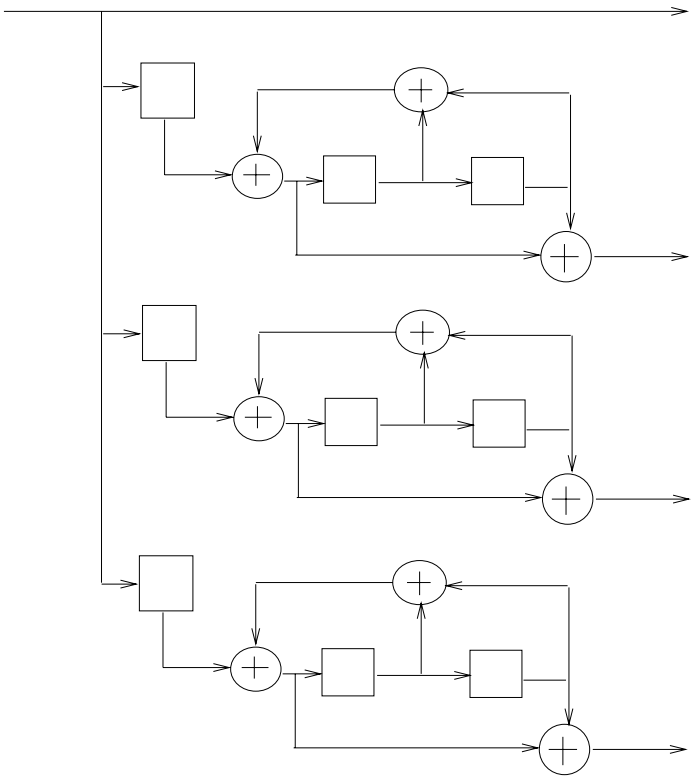

Figure 2.2: Example of a Parallel Concatenated Code Consisting of RSC Codes

Next, consider the concatenated code which uses NSC codes as the constituent codes. For NSC codes, low weight input sequences tend to be self-terminating. Hence, parity weights, z, with low weights often result from low weight input sequences, w. Thus for a systematic NSC code, low codeword weights, $i = w + z$, are strongly correlated with low weight inputs. For the parallel concatenated encoder consisting of NSC codes, the interleaved output will have low weight if $\mathbf{d}$ has low weight. Hence,

the encoded bits for each of the component codes will typically have low error weight. Consider the example shown in Figure 2.3 which exhibits a parallel concatenated encoder composed of NSC codes.

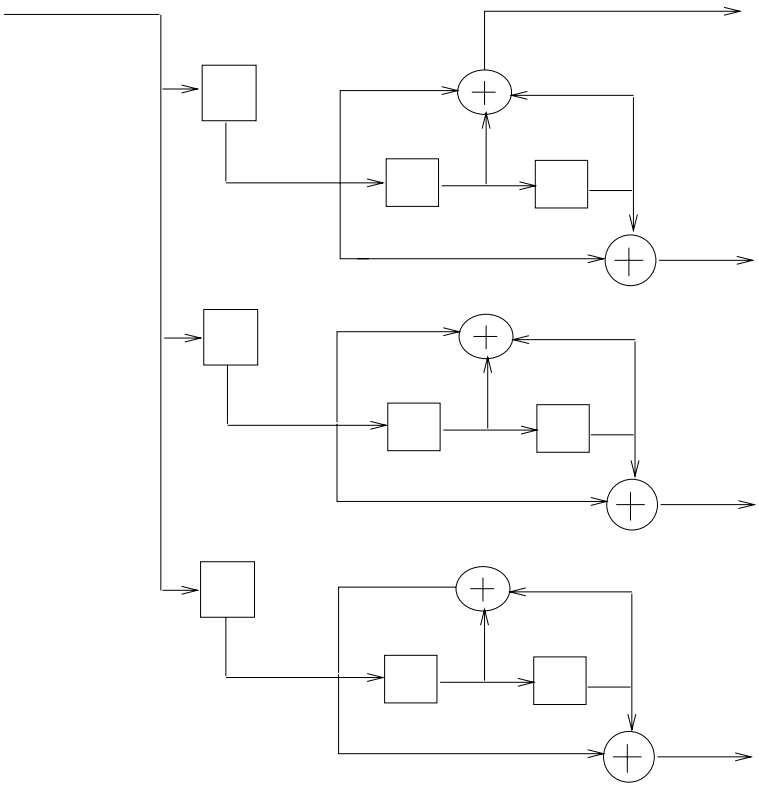

Figure 2.3: Example of a Parallel Concatenated Code Consisting of NSC Codes

Similar to the RSC codes in Figure 2.2, the minimum distance of each component NSC code in Figure 2.3 is 5, 2, and 2, respectively. The input sequence which yields the minimum distance for each component code is the weight one sequence (00...010...00). Each component encoder is preceded by an interleaver which for the RSC case, broke up input sequences which yielded low weight. However, for the weight one input sequence, the output of each interleaver output will also be the weight one sequence. Thus, the worst-case minimum distance is always achieved when $\mathbf{d}$ is a

weight one sequence.

To summarize the results of this section, the turbo code which uses RSC codes as its constituent codes yields favorable distance properties at least with respect to the encoder which uses NSC codes as its constituent codes. As the interleaver size increases, the turbo code with RSC codes yields performance which improves as N^{-1}. In the next section, the structure and computations associated with turbo decoding algorithm is described.

2.2 Turbo Decoder

Because the optimal maximum a posteriori (MAP) decoder is too computationally complex, the turbo decoder provides a suboptimal alternative which provides near-optimal performance with significantly reduced decoding complexity. Consider the rate one-third turbo code with two constituent codes (i.e. $M = 2$). Rather than using a single MAP decoder to compute likelihood estimates based on all three coded bitstreams, the turbo decoder uses two MAP decoders which are matched to the two component encoders (i.e. each MAP decoder uses the systematic bitstream and one of the two parity bitstreams to compute likelihood estimates). In this way, each MAP decoder needs to consider only one trellis structure rather than a concatenation of trellises.

The turbo decoding algorithm is essentially as follows. Each MAP decoder computes a soft output which is the log likelihood ratio of *a posteriori* information bit probabilities. This likelihood is then used to initialize the *a priori* probabilities in

the following MAP decoder after which a new likelihood ratio is computed. This procedure is iterated several times, generally converging to some low bit error rate.

The derivation of this algorithm has been well documented in previous papers [8][20][21] and is also included in the Appendix. For notational purposes, the algorithm is briefly reviewed here.

Let S_k be the state of the first encoder at time k let $y_{0,k}$, $y_{1,k}$, and $y_{2,k}$ represent the channel outputs respective to d_k, $p_{1,k}$, and $p_{2,k}$. Furthermore, let L_k be the log likelihood ratio (LLR) of the *a posteriori* probabilities for the first MAP decoder which makes calculations based on $\mathbf{y_0}$ and $\mathbf{y_1}$. Then as shown in [8],

$$\begin{aligned} L_k &= \log \frac{P(d_k = 1|\mathbf{y_0}, \mathbf{y_1})}{P(d_k = 0|\mathbf{y_0}, \mathbf{y_1})} && (2.2) \\ &= \log \frac{\sum_m \sum_{m'} \gamma_1(y_{0,k}, y_{1,k}, m', m)\alpha_{k-1}(m')\beta_k(m)}{\sum_m \sum_{m'} \gamma_0(y_{0,k}, y_{1,k}, m', m)\alpha_{k-1}(m')\beta_k(m)} && (2.3) \end{aligned}$$

where the forward and backward recursions can be expressed as

$$\alpha_k(m) = \frac{\sum_{m'} \sum_{i=0}^{1} \gamma_i(y_{0,k}, y_{1,k}, m', m)\alpha_{k-1}(m')}{\sum_m \sum_{m'} \sum_{i=0}^{1} \gamma_i(y_{0,k}, y_{1,k}, m', m)\alpha_{k-1}(m')} \quad (2.4)$$

$$\beta_k(m) = \frac{\sum_{m'} \sum_{i=0}^{1} \gamma_i(y_{0,k+1}, y_{1,k+1}, m', m)\beta_{k+1}(m')}{\sum_m \sum_{m'} \sum_{i=0}^{1} \gamma_i(y_{0,k+1}, y_{1,k+1}, m', m)\alpha_k(m')}. \quad (2.5)$$

The branch transition probabilities used by the MAP algorithm are calculated as

$$\begin{aligned} \gamma_i(y_{0,k}, y_{1,k}, m', m) &= p(y_{0,k}|d_k = i, S_k = m, S_{k-1} = m') \cdot \\ & \quad p(y_{1,k}|d_k = i, S_k = m, S_{k-1} = m') \cdot \\ & \quad P(S_k = m|d_k = i, S_{k-1} = m') \cdot P(d_k = i|S_{k-1} = m') \quad (2.6) \end{aligned}$$

where $P(S_k = m|d_k = i, S_{k-1} = m') = 1$ if bit i is associated with the given state transition and equals 0 if it is not.

The branch transition probabilities are of particular interest. The branch transition probability essentially measures the probability of channel outputs given a particular state transition along a branch in the encoder trellis. The computation of branch transition probabilities depends on the channel, so they play a key role in the design of the turbo decoder for FH-SS.

Note that the *a priori* information probability, $P(d_k = i|S_k = m) = P(d_k)$, in (2.6) is independent of the states S_k and S_{k-1}. In addition, because the encoder is systematic, the transition probability, $p(y_{0,k}|d_k = i, S_k = m, S_{k-1} = m')$, in (2.6) is also independent of the states S_k and S_{k-1}. Hence, (2.3) can be factored as

$$L_k = L'_k + L_{0,k} + L_k^{(1)} \tag{2.7}$$

where as shown below, L'_k represents the *a priori* information, $L_{0,k}$ represents the contribution corresponding to $y_{0,k}$, and $L_k^{(1)}$ represents the contribution from $\mathbf{y_1}$.

$$L'_k = \log \frac{P(d_k = 1)}{P(d_k = 0)} \tag{2.8}$$

$$L_{0,k} = \log \frac{p(y_{0,k}|d_k = 1)}{p(y_{0,k}|d_k = 0)} \tag{2.9}$$

$$L_k^{(1)} = \log \frac{\sum_m \sum_{m'} \gamma'_1(y_{1,k}, m', m)\alpha_{k-1}(m')\beta_k(m)}{\sum_m \sum_{m'} \gamma'_0(y_{1,k}, m', m)\alpha_{k-1}(m')\beta_k(m)} \tag{2.10}$$

where $\gamma'_i(y_{1,k}, m', m) = p(y_{1,k}|d_k = i, S_k = m, S_{k-1} = m') \cdot P(S_k = m|d_k = i, S_{k-1} =$

m').

$L_k^{(1)}$ is termed the *extrinsic* portion of the log likelihood ratio. It is used to initialize the *a priori* data probabilities for the second MAP (MAP2) decoder. The concept of extrinsic information is important in that it prevents information introduced by one of the component decoders to be passed back to that component decoder. Note that while this iterative approach does not yield the maximum likelihood solution, it attempts to do so in an iterative way. Let $L^{(1)}$ and $L^{(2)}$ correspond to the extrinsic information generated by the MAP1 and MAP2 decoders, respectively. For equally likely transmitted data, final bit decisions for d_k are based on

$$\tilde{L}_k \quad = \quad L_{0,k} + L_k^{(1)} + L_k^{(2)}. \tag{2.11}$$

The decoding algorithm is as follows. Initially, the data is assumed to be equally likely and $L'_k = 0$. The output of MAP1 is $L_{0,k} + L_k^{(1)}$. For MAP2, we let $L'_k = L_k^{(1)}$ (i.e. we let the extrinsic information generated by MAP1 become the *a priori* information for MAP2 calculations). The output of MAP2 is $\tilde{L}_k$. Similarly on the next iteration, we let $L'_k = L_k^{(2)}$. This process is iterated and eventually converges to some low bit error rate (BER), where final decisions for d_k use (2.11). The structure of the turbo decoder is shown in Figure 2.4.

Note that because the log likelihood ratio is the ratio of *a posteriori* information bit probabilities, the turbo decoding process can be viewed as the iterative improvement of *a posteriori* information bit probabilities. The turbo decoder uses the *a posteriori* probabilities generated by the previous MAP decoder as *a priori* information. This

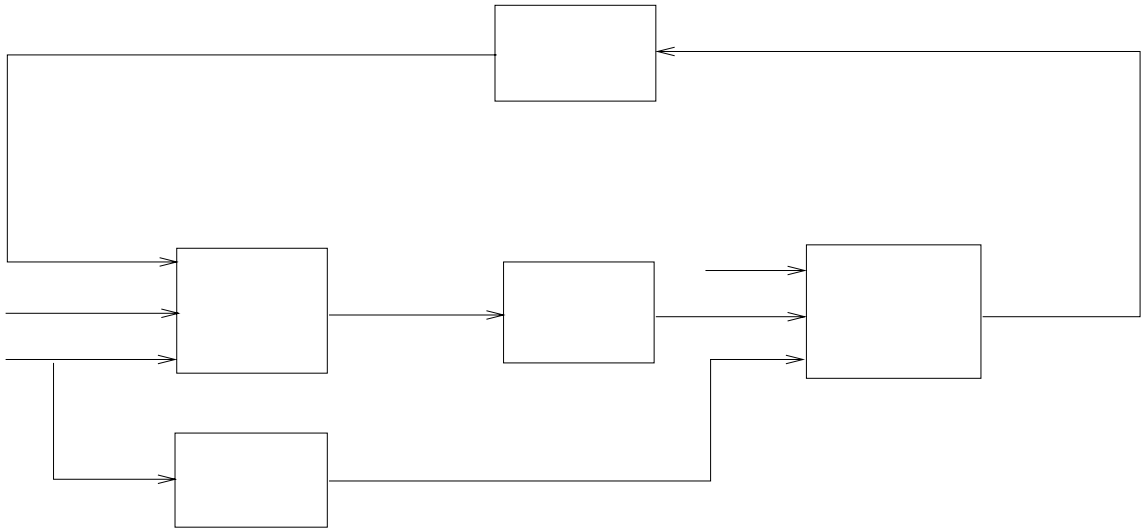

Figure 2.4: Block Diagram of the Turbo Decoder

idea will be applied in subsequent chapters to the iterative estimation techniques used for channels with memory.

CHAPTER 3

Turbo Codes in FH-SS with Partial-Band Interference

3.1 Introduction

In this chapter, turbo codes are considered in a frequency-hopped spread spectrum system with partial-band interference. In addition, an iterative channel estimation scheme is implemented to take advantage of the channel memory. The channel under consideration has memory if the frequency hopper changes frequencies at a faster rate than the rate the jammer changes its signal structure. In this case, it can be assumed that an entire hop is either completely jammed or unjammed. If multiple bits are transmitted over static channel conditions, it may be useful to estimate the channel state and use this information in the decoder. In this chapter, an iterative channel estimation technique paired with the powerful error correction capabilities of the turbo code will be shown to yield large performance gains over other well-known codes and decoding algorithms.

In previous chapters, it was stated that due to interleaver delay and high decoding complexity, turbo codes may be better suited for packet data communications. One packet data network of interest is slow-frequency hop radios. There has been considerable interest in enhancing slow-frequency hop radios so that they can be integrated into a packet radio network [16][17]. One such enhancement would be improved error correction coding. The use of the Reed-Solomon code and the concatenated code consisting of a Reed-Solomon outer code and a convolutional inner code have been considered in frequency-hopped spread spectrum in partial-band interference [16] [18].

Turbo codes were first considered in a frequency-hop spread spectrum system in [19] where performance was analyzed versus spectral efficiency. The model, however, did not include partial-band interference or memory. In this chapter, the performance of turbo codes is investigated in a frequency-hop spread spectrum system with partial-band interference and memory.

This chapter is organized as follows. In Section 3.2, the system model is presented, including details of the FH-SS model for the cases of coherent and noncoherent reception. In Section 3.3, the modifications to the turbo decoder necessary for FH-SS are described. Analytical performance bounds are derived in Section 3.4. In Section 3.5, simulation results are presented and compared to the performance of other well-known coding techniques. In Section 3.6, the numerical results of the analytical bounds are discussed. Finally, conclusions are drawn in Section 3.7.

3.2 System Model

In this section, the system model is described. In particular, the transmitter model is detailed, including descriptions of the encoder, relevant interleavers, and modulation schemes. In addition, the channel model is presented.

3.2.1 Transmitter

The encoder is formed by concatenating two constituent codes in parallel and separating the codes by a helical interleaver. As in the original work by Berrou *et al* [8], the constituent codes are recursive systematic convolutional codes. The helical interleaver is used in the encoder to guarantee trellis termination (see Section 2.1 and the Appendix). The encoder takes as input the data sequence of length N, $\mathbf{d} = d_1, ..., d_k, ..., d_N$, where $d_k \in \{0, 1\}$, and then produces three streams: the data sequence $\mathbf{d}$, the parity sequence $\mathbf{p_1}$ of the first component encoder with input $\mathbf{d}$, and the parity sequence $\mathbf{p_2}$ of the second component encoder with interleaved $\mathbf{d}$ as input.

The channel encoded bits are then passed to a channel interleaver. Described in Section 3.3.2, the channel interleaver is instrumental in breaking up the memory for channels which exhibit bursts of errors. The modulation considered is binary phase shift keying (BPSK) for coherent reception and binary frequency shift keying (BFSK) for noncoherent reception. The resultant signal is frequency hopped. The hopping patterns of the FH-SS system are modeled as sequences of independent random variables uniformly distributed over the allowable frequency range. Let R_c be the rate of the code, R_b be the data rate, and R_h be the hopping rate. If the coded data rate,

R_b/R_c, exceeds the hopping rate, R_h, then there exists $h = \frac{R_b}{R_c R_h}$ bits per hop, where $\frac{R_b}{R_c R_h} > 1$. A block diagram of the transmitter is shown in Figure 3.1.

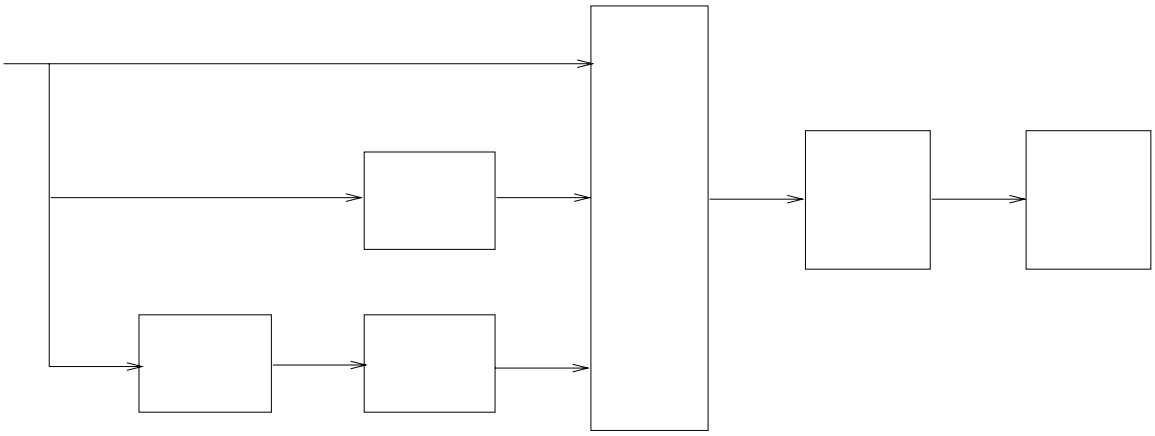

Figure 3.1: Block Diagram of the Transmitter

3.2.2 Channel Model

The model we assume for the channel consists of a jammer that evenly distributes its power over a fraction ρ of the frequency range. Thus, transmission occurs over a channel that includes full-band thermal noise with double-sided power spectral density $\frac{N_0}{2}$ and partial band interference with double-sided power spectral density $\frac{N_J}{2\rho}$ which covers a fraction ρ of the band. As a result, there are essentially two channel states: jammed and unjammed. The probability of hopping to a jammed state is ρ and the probability of hopping to an unjammed state is $1-\rho$. It is assumed that the jammer stays on for the entire duration of the hop if it is jammed at all. Let $(y_{0,k}, y_{1,k}, y_{2,k})$ be the outputs of the channel and let $z_{i,k}$ represent the channel state for $y_{i,k}$. Jammed and unjammed states correspond to $z_{i,k} = 1$ and $z_{i,k} = 0$, respectively. For coherent reception, the model for the matched filter outputs is

$$y_{i,k} = \sqrt{E}\, c_{i,k} + \eta_{i,k}, \qquad i = 0, 1, 2; \quad k = 1, ..., N \tag{3.1}$$

where $(c_{0,k}, c_{1,k}, c_{2,k}) = ((-1)^{d_k}, (-1)^{p_{1,k}}, (-1)^{p_{2,k}})$ maps the values of the binary encoded bits to $\{-1, +1\}$, and the noise, $\eta_{i,k} \sim N(0, \frac{N_0}{2} + z_{i,k} \cdot \frac{N_J}{2\rho})$, is assumed to be a zero mean Gaussian random variable with variance depending on whether the state is jammed. If the state is jammed, the variance is $\frac{N_0}{2} + \frac{N_J}{2\rho}$; if the state is not jammed, the variance is $\frac{N_0}{2}$.

Next, consider the case of noncoherent reception. Let $c_{i,k} \in \{-1, +1\}$ denote the coded bit (input to the BFSK modulator). Shown in the equations below, the FSK matched filter outputs can be represented as

$$x_{i,k}^{(c,+1)} = \sqrt{E}\, \delta_{c_{i,k},+1} \cos(\theta_{i,k}) + \eta_{i,k}^{(c,+1)} \tag{3.2}$$

$$x_{i,k}^{(s,+1)} = \sqrt{E}\, \delta_{c_{i,k},+1} \sin(\theta_{i,k}) + \eta_{i,k}^{(s,+1)} \tag{3.3}$$

$$x_{i,k}^{(c,-1)} = \sqrt{E}\, \delta_{c_{i,k},-1} \cos(\theta_{i,k}) + \eta_{i,k}^{(c,-1)} \tag{3.4}$$

$$x_{i,k}^{(s,-1)} = \sqrt{E}\, \delta_{c_{i,k},-1} \sin(\theta_{i,k}) + \eta_{i,k}^{(s,-1)} \tag{3.5}$$

where $\delta_{a,b} = 1$ if $a = b$ and $\delta_{a,b} = 0$ otherwise; the unknown phase $\theta_{i,k}$ is a uniform random variable spanning $(0, 2\pi)$; and $\eta_{i,k}^{(c,+1)}$, $\eta_{i,k}^{(s,+1)}$, $\eta_{i,k}^{(c,-1)}$, and $\eta_{i,k}^{(s,-1)}$ are independent Gaussian random variables with zero mean and conditional variance $\frac{N_0}{2} + z_{i,k} \frac{N_J}{2\rho}$.

Having described the channel, the necessary modifications to the original turbo decoder (described in Chapter 2) are presented in the next section.

3.3 Turbo Decoder for FH-SS

The turbo decoding algorithm is dependent on what information is available to the turbo decoder. The cases where knowledge of the channel state (i.e. jammed or unjammed) is either available or unavailable to the decoder are examined. The case of known channel state will be referred to as "side information". In addition, the cases of independent and identically distributed (IID) transmission (i.e. one bit per hop) and transmission over a channel with memory (i.e. h bits per hop) are considered. It is assumed for all cases that the model order of the hidden Markov model is known.

For all cases in this chapter, the power spectral densities of the channel noise, $\frac{N_0}{2}$ and $\frac{N_J}{2\rho}$, are assumed to be known to the decoder. These values are necessary to compute the branch transition probabilities in (2.6). Note, however, that exact SNR values are not necessary to achieve good decoding performance. In [41] and as well by our own investigation (Chapter 7), the performance of turbo codes is shown to be insensitive to SNR mismatch. Thus, low complexity SNR estimation algorithms can be employed to achieve similar performance to the case where the SNR is exactly known.

3.3.1 FH-SS without Memory

In this section, the case of one bit per hop is considered. If the channel state is unknown, then the modified turbo decoder for IID FH-SS needs to employ the appropriate branch transition probabilities. More specifically, (2.6) is calculated using

$$p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m') = \tag{3.6}$$

$$p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m', z_{j,k} = 1) \cdot \rho +$$

$$p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m', z_{j,k} = 0) \cdot (1 - \rho).$$

If the channel state is known, the side information is treated as information received by the decoder. Thus, for branch transition probabilities computations, the joint conditional probability of the channel output, $y_{j,k}$, and the appropriate channel state, $z_{j,k}$, are calculated.

$$p(y_{j,k}, z_{j,k} = z|d_k = i, S_k = m, S_{k-1} = m') =$$

$$p(y_{j,k}|z_{j,k} = z, d_k = i, S_k = m, S_{k-1} = m')p(z_{j,k} = z) \tag{3.7}$$

Note that for coherent reception,

$$p(y_{j,k}|z_{j,k} = z, d_k = i, S_k = m, S_{k-1} = m') = \frac{1}{\sqrt{2\pi\sigma_z^2}} \exp\left[-\frac{(y_{j,k} - c_{j,k})^2}{2\sigma_z^2}\right] \tag{3.8}$$

where $\sigma_0^2 = \frac{N_0}{2}$, $\sigma_1^2 = \frac{N_0}{2} + \frac{N_J}{2\rho}$, and the coded bit, $c_{j,k}$, is determined by d_k, and the trellis state transition, $S_k = m$ and $S_{k-1} = m'$.

For noncoherent reception, consider a square law receiver which uses the FSK channel outputs (3.2) - (3.5) to compute

$$y_{j,k}^{(+1)} = (x_{j,k}^{(c,+1)})^2 + (x_{j,k}^{(s,+1)})^2 \tag{3.9}$$

$$y_{j,k}^{(-1)} = (x_{j,k}^{(c,-1)})^2 + (x_{j,k}^{(s,-1)})^2. \tag{3.10}$$

From (3.9) and (3.10), the branch transition probabilities can be computed as

$$p(y_{j,k}|c_{j,k} = c, z_{j,k}) = p(y_{j,k}^{(+1)}, y_{j,k}^{(-1)}|c_{j,k} = c, z_{j,k}) \tag{3.11}$$

$$= p(y_{j,k}^{(+1)}|c_{j,k} = c, z_{j,k})\, p(y_{j,k}^{(-1)}|c_{j,k} = c, z_{j,k}) \tag{3.12}$$

where

$$p(y_{j,k}^{(l)}|c_{j,k} = c, z_{j,k} = z) = \begin{cases} \frac{1}{2\sigma_z^2} \exp[-\frac{y_{j,k}^{(l)}+E}{2\sigma_z^2}] I_0(\sqrt{y_{j,k}^{(l)}\, E}/\sigma_z^2) & \text{if } c = l \\ \\ \frac{1}{2\sigma_z^2} \exp[-\frac{y_{j,k}^{(l)}}{2\sigma_z^2}] & \text{if } c \neq l. \end{cases} \tag{3.13}$$

Having described the modifications for the one bit per hop case, the more interesting case with multiple bits per hop will be discussed in the next section.

3.3.2 FH-SS with Memory

In this section, the modification to the turbo decoder for the case of multiple bits per hop is considered. First, the design of the hopping structure is described. The hopping structure details which coded bits are transmitted over each hop. Hence, the channel interleaver should be employed to meet the design requirements of the

desired hopping structure. The hopping structure is important for cases with memory because unlike the IID case where jammed hops affect single bits, jammed hops now affect multiple bits. For instance, if $c_{0,k}$, $c_{1,k}$, and $c_{2,k}$ are the coded bits which correspond to information bit d_k, then it would be beneficial to send these bits over separate hops. If they were sent over the same hop and that hop was jammed, it would be difficult to decode the information bit correctly. Because the convolutional encoders display memory (i.e. $c_{1,k}$ is dependent on $c_{0,k}$, $c_{0,k-1}$, $c_{0,k-2}$, and so on), it makes sense for similar reasons to separate consecutive coded bits by as much as possible. Using $L = N/h$ where h is the number of bits per hop and N is the number of information bits per packet, the structure of the hopper is shown in Table 3.1.

HOP 1	$c_{0,1}$	$c_{1,L+1}$	$c_{2,2L+1}$	$c_{0,3L+1}$	$c_{1,4L+1}$	...
.	.	.	.	.	.	...
.	.	.	.	.	.	...
HOP L	$c_{0,L}$	$c_{1,2L}$	$c_{2,3L}$	$c_{0,4L}$	$c_{1,5L}$	...
HOP $L+1$	$c_{1,1}$	$c_{2,L+1}$	$c_{0,2L+1}$	$c_{1,3L+1}$	$c_{2,4L+1}$	...
.	.	.	.	.	.	...
.	.	.	.	.	.	...
HOP $2L$	$c_{1,L}$	$c_{2,2L}$	$c_{0,3L}$	$c_{1,4L}$	$c_{2,5L}$	...
HOP $2L+1$	$c_{2,1}$	$c_{0,L+1}$	$c_{1,2L+1}$	$c_{2,3L+1}$	$c_{0,4L+1}$	...
.	.	.	.	.	.	...
.	.	.	.	.	.	...
HOP $3L$	$c_{2,L}$	$c_{0,2L}$	$c_{1,3L}$	$c_{2,4L}$	$c_{0,5L}$	...

Table 3.1: Structure of Frequency Hopper

For cases with multiple bits per hop and no side information, we attempt to compensate for the lack of side information by generating estimates of the channel. The approach is to calculate *a posteriori* probabilities for each channel state, $p(z_k|\mathbf{y_0}, \mathbf{y_j})$ for $j = 1, 2$, and send this information in addition to $p(d_k|\mathbf{y_0}, \mathbf{y_j})$ between decoders.

Thus, information bit estimates and channel state estimates can be iteratively improved. The use of channel estimates to calculate branch transition probabilities should lead to improved error rates.

Note that the structure of the hopper shown in Table 3.1 allows channel estimates to be calculated in a manner similar to the way information bit estimates are calculated in the original turbo decoder. For instance, each MAP decoder in the original turbo decoder calculates *a posteriori* probabilities of the information bits given two of the three received observation vectors. This information is passed to the next MAP decoder which uses this information as *a priori* information bit probabilities. Similarly, by using the structure of the hopper in Table 3.1, two-thirds of the relevant observation sequence is available to each MAP decoder so that *a posteriori* probabilities for each hop can be calculated. This information can be passed between MAP decoders and be used as *a priori* channel state probabilities.

$\mathbf{R}$ is defined as the vector of received channel outputs that is available to the MAP decoder, $\mathbf{R_k}$ as the subset of $\mathbf{R}$ that has been received over a given hop with state z_k, and $\tilde{\mathbf{R}}_\mathbf{k}$ as the subset of $\mathbf{R}$ that has not been received over the hop with state z_k. Thus, $\mathbf{R} = \mathbf{R_k} \cup \tilde{\mathbf{R}}_\mathbf{k}$ where $\mathbf{R_k} = (\mathbf{R_{k,1}}, ..., \mathbf{R_{k,h}})$. The calculation of state estimates is shown below.

$$p(z_k = z|\mathbf{R}) = \frac{p(\mathbf{R}|z_k = z) \cdot p(z_k = z)}{p(\mathbf{R})} \tag{3.14}$$

$$= \frac{p(\mathbf{R_k}|z_k = z) \cdot p(\tilde{\mathbf{R}}_\mathbf{k}|z_k = z) \cdot p(z_k = z)}{p(\mathbf{R})} \tag{3.15}$$

$$= p(\mathbf{R}_k|z_k = z) \cdot p(z_k = z) \cdot \frac{p(\tilde{\mathbf{R}}_\mathbf{k})}{p(\mathbf{R})} \tag{3.16}$$

$$= \quad p(\mathbf{R_k}|z_k = z) \cdot p(z_k = z) \cdot K \tag{3.17}$$

$$\approx \quad p(\mathbf{R_k}|z_k = z) \cdot \tilde{p}(z_k = z|\mathbf{R_k}) \cdot K \tag{3.18}$$

where K is a normalizing factor chosen to make the probability density function sum to 1 and $\tilde{p}(z_k = z|\mathbf{R_k})$ is the channel estimate provided by the previous MAP decoder.

In the above equation $\tilde{p}(z_k = z|\mathbf{R_k})$ is used in place of $p(z_k = z)$ to take advantage of the state estimate provided by the previous MAP decoder. In order to compute each state estimate, the conditional joint probabilities of $\mathbf{R_k}$ in (14) must be computed. This can be calculated by performing total probability on $p(\mathbf{R_k}|z_k = z)$ over the respective coded bits.

$$p(\mathbf{R_k}|z_k = z) = \sum_{\mathbf{c_k}} p(\mathbf{R_k}|\mathbf{c_k} = \mathbf{c}, z_k = z)\, p(\mathbf{c_k} = \mathbf{c}) \tag{3.19}$$

where $\mathbf{c_k}$ represents the vector of coded bits respective to $\mathbf{R_k}$.

This will require *a priori* probability knowledge of the coded bits. But, the MAP decoders are already sending log likelihood ratios that give $p(d_k = d|\mathbf{R})$. Using this information, $p(c_{j,k} = c) \approx p(c_{j,k} = c|\mathbf{R})$ can be calculated. Thus, as the MAP decoders refine their estimates of $p(d_k|\mathbf{R})$, estimates of $p(z_k|\mathbf{R})$ are also getting more refined.

Note that as h, the number of bits per hop, increases, the complexity of directly computing $p(\mathbf{R_k}|z_k = z) = p(R_{k,1}, ..., R_{k,h}|z_k = z)$ rises exponentially. To overcome this problem, the following recursion was developed.

1. Initial Case

$$p(z_k = z|R_{k,1}) = \frac{p(R_{k,1}|z_k = z) \cdot p(z_k = z)}{p(R_{k,1})} \quad (3.20)$$

$$= \sum_{c=0}^{1} p(R_{k,1}|z_k = z, c_{k,1} = c)p(c_{k,1} = c) \cdot \frac{p(z_k = z)}{p(R_{k,1})} \quad (3.21)$$

$$\approx \sum_{c=0}^{1} p(R_{k,1}|z_k = z, c_{k,1} = c)p(c_{k,1} = c) \cdot \frac{\tilde{p}(z_k = z|\mathbf{R_k})}{p(R_{k,1})} \quad (3.22)$$

where $\tilde{p}(z_k = z|\mathbf{R_k})$ is the channel state estimate of the previous MAP decoder.

2. Recursion

$$p(z_k = z|R_{k,1}, R_{k,2}, ..., R_{k,i-1}, R_{k,i})$$

$$= \frac{p(R_{k,i}|z_k = z, R_{k,1}, ..., R_{k,i-1}) \cdot p(z_k = z|R_{k,1}, ..., R_{k,i-1})}{p(R_{k,i}|R_{k,1}, ..., R_{k,i-1})} \quad (3.23)$$

$$\approx \frac{p(R_{k,i}|z_k = z) \cdot p(z_k = z|R_{k,1}, ..., R_{k,i-1})}{p(R_{k,i}|R_{k,1}, ..., R_{k,i-1})} \quad (3.24)$$

Note that (3.24) is an approximation since $R_{k,i}$ is lightly correlated with $R_{k,i-1}$, ..., $R_{k,1}$.

Once the state estimates have been computed, they are ready to be used in a turbo decoder. State estimates are used in a manner analogous to the way that a turbo decoder uses information bit estimates. When there is no SI, the *a priori* state probabilities are replaced by the *a posteriori* state probabilities for branch transition probability calculations. As in the IID case, the appropriate *a priori* probability is used for cases with side information. Thus, for the MAP1 decoder with $j = 0, 1$

$$
\begin{aligned}
& p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m') \\
= \;& p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m', z_{j,k} = 1) \cdot p(z_{j,k} = 1) + \\
& p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m', z_{j,k} = 0) \cdot p(z_{j,k} = 0) \qquad (3.25) \\
\approx \;& p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m', z_{j,k} = 1) \cdot p(z_{j,k} = 1|\mathbf{y_0}, \mathbf{y_2})) + \\
& p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m', z_{j,k} = 0) \cdot p(z_{j,k} = 0|\mathbf{y_0}, \mathbf{y_2})) \qquad (3.26)
\end{aligned}
$$

when there is no side information and

$$
\begin{aligned}
& p(y_{j,k}, z_{j,k} = z|d_k = i, S_k = m, S_{k-1} = m') \\
= \;& p(y_{j,k}|z_{j,k} = z, d_k = i, S_k = m, S_{k-1} = m') \cdot p(z_{j,k} = z) \qquad (3.27)
\end{aligned}
$$

when there is side information.

3.4 Analytical Bounds

It is often impractical to generate simulation results for extremely low BERs. As a result, bounds are often calculated. First, the bounds for the case of coherent reception are discussed.

3.4.1 Coherent Reception

Turbo codes are linear, so without loss of generality, it will be assumed that the all-zeros codeword was transmitted. For the case of coherent reception, the union bound is used to form an analytical expression for the probability of error. Note that the union bound applies to the optimal decoder, while the MAP iterations of the turbo decoder are suboptimal. If A_d is the weight enumerator of the code and $P_2(d)$ is the pairwise error probability between the all-zeros codeword and a codeword of weight d assuming maximum likelihood decoding, the union bound for an (n, k) block code is

$$P_{word} \leq \sum_{d=d_{min}}^{n} A_d \, P_2(d). \tag{3.28}$$

The only known way to exactly calculate A_d is via an exhaustive search involving all possible input sequences. One solution is to calculate an average upper bound by computing an average weight function over all possible interleaving schemes [26]. The average weight function in [26] was calculated as

$$\overline{A}_d = \sum_{i=1}^{k} \begin{pmatrix} k \\ i \end{pmatrix} p(d|i) \tag{3.29}$$

where $p(d|i)$ is the probability that an interleaving scheme maps an input of weight i to produce a codeword of total weight d and $\begin{pmatrix} k \\ i \end{pmatrix}$ is the number of input frames

with weight i. Thus,

$$\overline{P}_{word} \leq \sum_{d=d_{min}}^{n} \sum_{i=1}^{k} \begin{pmatrix} k \\ i \end{pmatrix} p(d|i)\, P_2(d). \tag{3.30}$$

The average upper bound to the word error probability, $\overline{P}_{word}$ is computed by summing over the weights of the input sequences, i, and the weights of the resultant codewords, d. A lower bound on the bit error probability can be computed in a similar fashion by noting that for each summed value, there can be no greater than i bit errors out of the k total information bits.

$$\overline{P}_{bit} \leq \sum_{d=d_{min}}^{n} \sum_{i=1}^{k} \frac{i}{k} \begin{pmatrix} k \\ i \end{pmatrix} p(d|i)\, P_2(d). \tag{3.31}$$

An algorithm for calculating $p(d|i)$ is given in [26] and is described in the Appendix. Thus, in order to compute this bound, we need only calculate $P_2(d)$. Over a channel with both full-band thermal noise and partial-band jamming noise, the pairwise error probability can be calculated by conditioning on the number of jammed symbols.

$$P_2(d) = \sum_{l=0}^{d} P(\text{error} \mid E_l) \cdot P(E_l) \tag{3.32}$$

where E_l is the event that l symbols are jammed and

$$P(E_l) = \begin{pmatrix} d \\ l \end{pmatrix} \rho^l (1-\rho)^{d-l}. \tag{3.33}$$

The fundamental approach to calculating pairwise error probabilities is to compute log likelihood ratios. For the case where the decoder has no side information, this is difficult since the receiver does not know which bits have been jammed. Thus, for analytical purposes, the suboptimal decoder that makes bit decisions based on the sum of the channel outputs will be considered. For this suboptimal decoder,

$$P(\text{error} \mid E_l) = Q\left(\sqrt{\frac{E_b R d}{\frac{N_0}{2} + \frac{l}{d}\frac{N_J}{2\rho}}}\right). \tag{3.34}$$

If the decoder has side information, log likelihood ratios are calculated in the normal way and then calculate the probability of error.

$$P(\text{error} \mid E_l) = Q\left(\sqrt{E_b R \left(\frac{l}{\frac{N_0}{2} + \frac{N_J}{2\rho}} + \frac{d-l}{\frac{N_0}{2}}\right)}\right) \tag{3.35}$$

3.4.2 Noncoherent Reception

For the case of noncoherent reception, the approach is similar. Instead of invoking the union bound, however, the Union-Bhattacharrya Bound is used, where D is the Bhattacharrya parameter and $P_2(d) \leq D^d$. Thus,

$$P_{word} \leq \sum_{d=d_{min}}^{n} A_d \, P_2(d) \tag{3.36}$$

$$\leq \sum_{d=d_{min}}^{n} A_d \, D^d \tag{3.37}$$

and

$$\overline{P}_{word} \leq \sum_{d=d_{min}}^{n} \sum_{i=1}^{k} \begin{pmatrix} k \\ i \end{pmatrix} p(d|i)\, D^d. \tag{3.38}$$

It was shown in [23] that with noncoherent reception, optimal decoding with side information leads to

$$D = \begin{cases} [\int_0^\infty u e^{-u^2/2} I_0^{1/2}(u\sqrt{2E_s/N_J})du]^2 & E_s/N_J < 2.871 \\ \frac{1.424}{E_s/N_J} & E_s/N_J \geq 2.871. \end{cases} \tag{3.39}$$

Square-law combining is a suboptimal method of decoding in AWGN, but has been shown to have an approximate performance loss of 0.14 dB for reasonable SNRs [23]. Because square-law combining is suboptimal, an upper bound on its performance will also be an upper bound to the performance of optimal decoding. The Bhattacharrya parameter for square-law combining in worst case jamming is ($\Gamma = E_s/N_J$)

$$D = \begin{cases} [\frac{1}{1-\lambda^2} \exp\{-\frac{\lambda\Gamma}{(1+\lambda)}\}] & \Gamma < 3 \\ \frac{4e^{-1}}{\Gamma} & \Gamma \geq 3 \end{cases} \tag{3.40}$$

$$\lambda = \frac{\sqrt{(2+\Gamma)^2 + 8\Gamma} - (2+\Gamma)}{4}. \tag{3.41}$$

3.5 Simulation Results

In this section, the simulation results are presented for the cases of coherent and noncoherent reception, multiple bits per hop, and cases with and without side information.

3.5.1 Coherent Reception

In this section, the simulation results are described for cases with coherent detection. In the simulations, the component encoders are rate $\frac{1}{2}$ recursive systematic convolutional encoders with memory 4 and octal generators $(37, 21)$. The packet size is 1640 bits and the number of turbo code iterations is 10. The SNR of the full-band thermal noise, E_b/N_0, is set to 20 dB. Cases with memory are simulated using 3 and 45 bits per hop (BPH).

Figure 3.2 shows the plot of E_b/N_J needed to achieve a packet error rate (PER) of 10^{-2} as a function of ρ. As would be expected, cases with side information (SI) performed better than their counterparts with no SI (NSI). The SI and NSI curves only meet when $\rho = 1.0$. In this case, all states are jammed, so side information provides no additional information.

The plots in Figure 3.2 exhibit a tradeoff as the number of bits per hop increases. For any memory, no SI case, where channel states are iteratively estimated, performance cannot be better than that of the corresponding memory SI case. Because channel state estimates will improve if the number of bits per hop increases, we should be able to get arbitrarily close to the corresponding memory, SI result by increasing

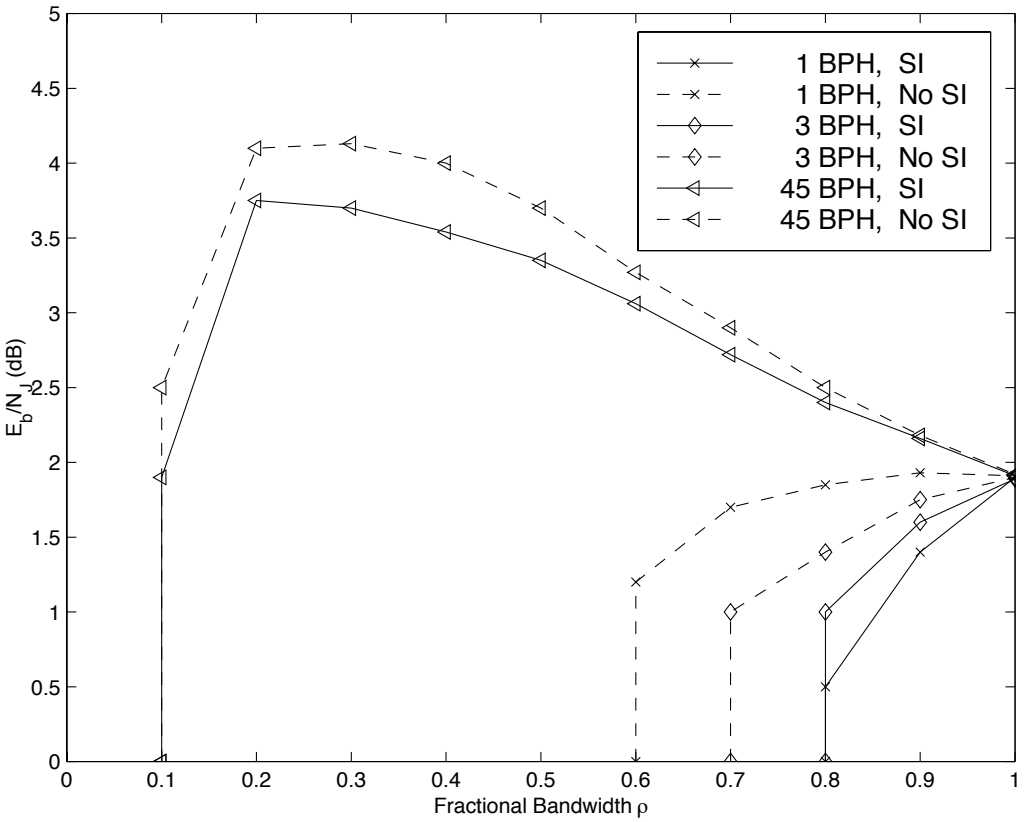

Figure 3.2: Performance of Turbo Codes in Coherent FH-SS with Partial-Band Interference

the memory. This is shown in Figure 3.2 by examining the performance differences between corresponding SI and no SI cases for variable bits per hop. For memory, no SI cases, reliable state information was computed. These state estimates provided useful information which in turn aided the decoding process.

The downside of increasing the memory can be seen by analyzing the SI cases in Figure 3.2. For all SI cases, knowledge of the channel state nulls out the advantage of more effective estimation. As a result, one might expect the performance of SI cases to be similar. However, as shown in Figure 3.2, this is clearly not the case. The performance of SI cases degrades as the memory increases. The reason for this trend in performance is that the number of independent hops per packet decreases as the number of bits per hop increases (for fixed packet length). Fewer independent hops means less hopping diversity and therefore reductions in performance. Thus, while increased memory leads to improved channel estimates in cases without SI, the best possible performance of no SI cases appears to effectively decrease.

In order to gauge these results, we refer to the application of convolutional codes to a FH-SS system with one bit per hop. Using a rate $1/3$, memory 4 convolutional code with maximal free distance, this result for PER $= 10^{-2}$ is shown in Figure 3.3. Performance of FH-SS systems is often measured by the worst case E_b/N_J across all ρ. Using this criterion, turbo codes show a gain of $3.1 - 3.5$ dB over convolutional codes, depending on whether SI is available to the decoder.

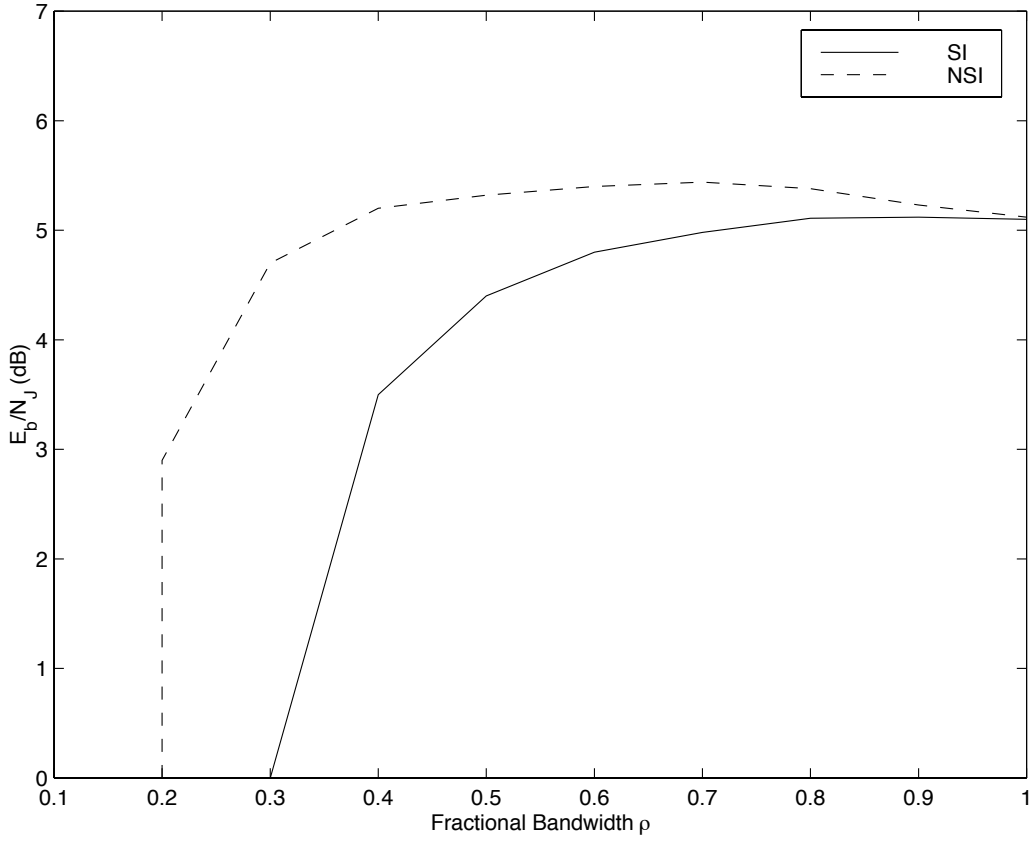

Figure 3.3: Performance of Convolutional Codes in FH-SS with Partial-Band Interference

3.5.2 Noncoherent Reception

In this section, simulation results for the case of noncoherent reception is presented. For these simulations, the component encoders are rate $\frac{1}{2}$ recursive systematic convolutional encoders with memory 4 and octal generators $(37, 21)$. The packet size is 1760 information bits and the number of decoder iterations is 5. A helical interleaver [27] is used to guarantee trellis termination (see Appendix). The SNR of the full-band thermal noise is set to 20 dB. Cases with memory are simulated using 20, 80, and 160 bits per hop (BPH).

Figure 3.4 shows the plot of minimum E_b/N_J needed to achieve a packet error rate (PER) of 10^{-3} as a function of ρ. The results are similar to the coherent case. The main result is the tradeoff which exists as the number of bits per hop increases. As the memory increases, the channel estimation improves because there are more channel outputs which can be used to estimate the channel. However, this improvement is offset because for fixed packet lengths, increasing memory results in fewer independent hops per packet. This tradeoff implies that for the NSI case with state estimation, there should be exist an optimal number of bits per hop that yields best performance. For $\rho \leq .7$ this value was found to be on the order of 20.

Due to inaccurate state estimation for large values of ρ, there is not a dwell interval length that is optimal for all values of ρ. To see this, first consider the results of the 1 bit per hop (IID) case in Figure 3.4. While it is expected that the SI case outperforms the NSI case without state estimation, it is interesting to compare these results with the IID NSI case when state estimation is performed. For low values of ρ, the NSI

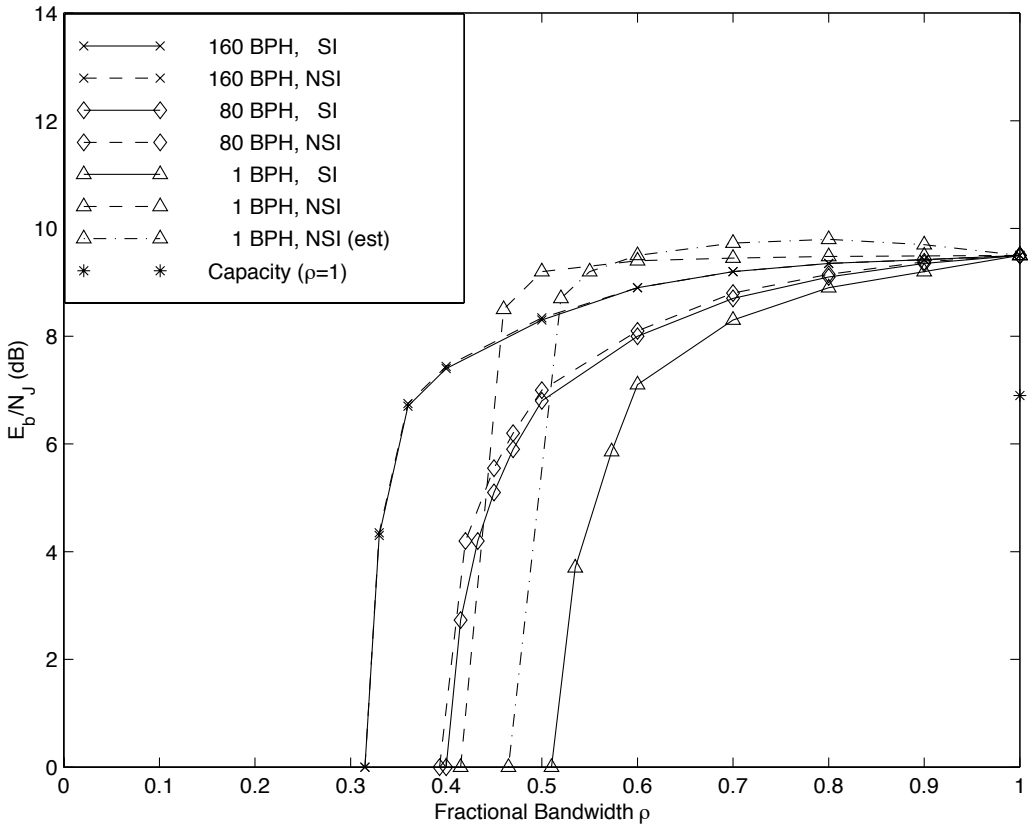

Figure 3.4: Performance of Turbo Codes in Noncoherent FH-SS with Partial-Band Interference

case with state estimation yields better results than the NSI case without estimation. Even with a single bit of observation, state estimation is valuable if the disparity between the SNRs of the estimated states ($\frac{N_0}{2}$ and $\frac{N_J}{2\rho}$) is large. For larger values of ρ, however, it becomes difficult to distinguish between the two states. Inaccurate state estimates lead to an improper weighting in branch transition probability calculations and thus, overall performance degrades.

Next, consider the results for the 20 bits per hop (BPH) case. In Figure 3.5, simulation results for four decoding cases are presented: SI, NSI with iterative state estimation, NSI with state estimates computed once before the decoder (i.e. prior to decoding, $p(z_k|\mathbf{R})$ is calculated; these probabilities are then used for all decoder branch transition probability calculations), and NSI with no state estimation. Note that for cases without SI, state estimation is beneficial for $\rho \leq .7$. At higher values of ρ, however, it becomes difficult to discern between the two channel states. Thus, channel state estimates become inaccurate and yield worse performance relative to the decoder which does not uses state estimation. However, because the NSI, no state estimation decoder shows a loss of less than 1 dB with respect to the SI case for $\rho > .7$, one solution is to use a hybrid decoder which decides whether or not to calculate state estimates based on a threshold at $\rho = .7$

In addition, note that when state estimates are computed, the 20 BPH case performs more poorly than even the IID case for $\rho > .7$. Clearly, more observations will lead to more reliable state estimates. However if these estimates are inaccurate, they adversely affect the decoder calculations for multiple bits. Thus for each value of ρ, state estimation is beneficial only when there is a sufficient number of observations

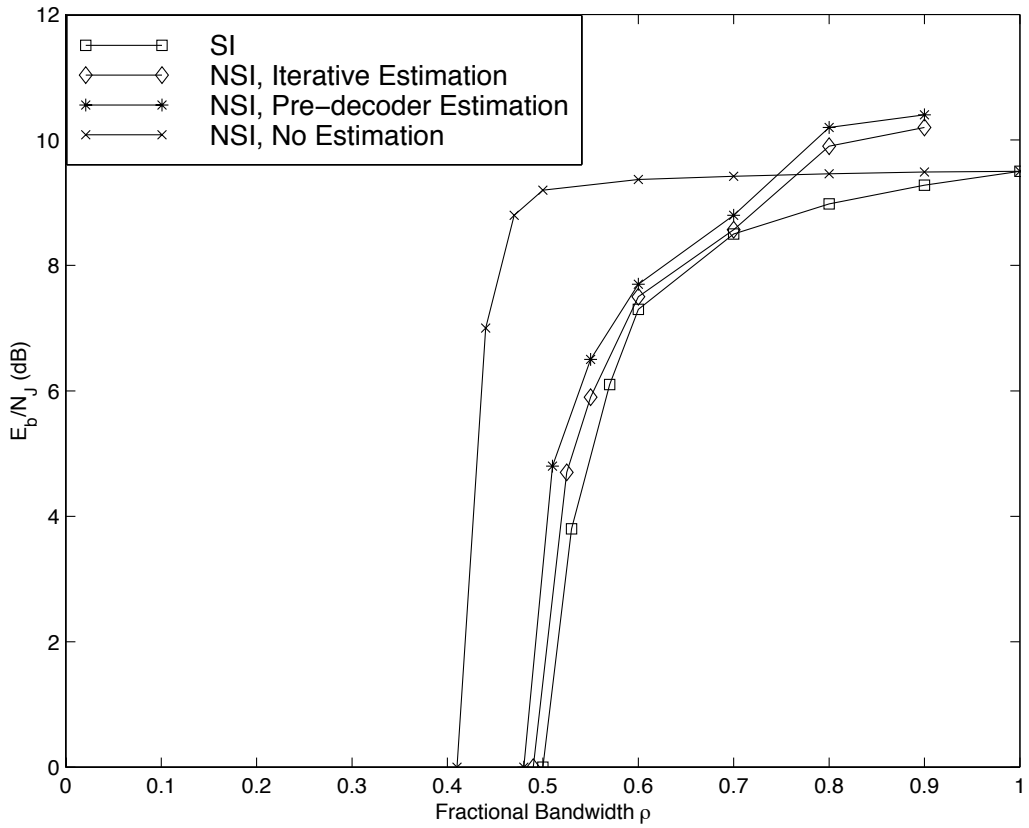

Figure 3.5: Performance of Turbo Codes in Noncoherent FH-SS with Partial-Band Interference: 20 Bits Per Hop

to guarantee a reliable estimate.

While the results obtained when using five turbo decoding iterations have been discussed, it is interesting to note the performance of the decoder after each iteration. The fast convergence of the turbo decoder is exhibited in Table 3.2 where the 160 BPH cases both use $\rho = .8$, $E_b/N_J = 9.5$ dB, the 1 BPH NSI case without state estimation uses $\rho = .7$, $E_b/N_J = 9.5$ dB, and the 1 BPH SI case uses $\rho = .7$, $E_b/N_J = 8.5$ dB.

PER After	**160 BPH, NSI**	**160 BPH, SI**	**1 BPH, NSI**	**1 BPH, SI**
1st iteration	$3.3e-01$	$2.8e-01$	$5.2e-01$	$4.6e-01$
2nd iteration	$2.2e-03$	$1.8e-03$	$2.6e-03$	$2.1e-03$
3rd iteration	$1.1e-03$	$1.1e-03$	$1.3e-03$	$1.2e-03$
4th iteration	$8.4e-04$	$8.5e-04$	$8.8e-04$	$8.6e-04$
5th iteration	$8.4e-04$	$8.4e-04$	$8.8e-04$	$8.6e-04$

Table 3.2: Performance of Turbo Codes By Iteration

The above results show that it is possible to bridge the gap between cases with and without side information by iteratively computing state estimates for a large number of bits per hop. However, the performance improvements may still be unsatisfactory. In the case of noncoherent reception, one possible improvement would be to perform phase estimation. If the phase during a hop is assumed to move very slowly and an orthogonal signal set is used, phase estimation can be performed using a method analogous to state estimation. In this manner, joint decoding and phase tracking can be achieved.

In order to gauge these results, we refer to the application of other coding methods to FH-SS systems. In particular, Pursley and Frank investigated the use of the Reed-Solomon code and the Reed-Solomon/convolutional concatenated code (RS-CC) in

[16]. For the Reed-Solomon code with no inner code, they used a $(32, 12)$ errors-only RS code over $GF(2^5)$ with noncoherent reception and 27 codewords per packet. Thus, the code had rate 3/8 and the total number of information bits per packet was 1620. There were 135 binary symbols per dwell period. For the concatenated code, they used a $(32, 18)$ RS code for the outer code and a rate 2/3, constraint length 6 convolutional code with erasure threshold $\gamma = 3$ for the inner code. In addition, the convolutional code used soft decisions. The dwell interval spanned 159 binary symbols and there were 20 codewords per packet. The overall rate of the code was 3/8 and the number of information bits per packet was 1800. Requiring a PER of 10^{-3}, these results are shown in Figure 3.6.

Because the turbo code system with 160 BPH matches the parameters shown in Figure 3.6 quite closely, this system will be used for comparison. Notice the large performance difference between the turbo code and the coding schemes of Figure 3.6. Comparing the worst-case performance of the turbo code system without side information with the RS and RS-CC systems, the turbo code system shows a gain of 6.9 and 6.3 dB, respectively. Note that to achieve worst-case performance for turbo codes, the jammer needs to jam the entire band ($\rho = 1$), while for RS and RS-CC codes, the jammer needs to jam only a small fraction of the band (about $\rho = .12$ and $\rho = .22$, respectively). Another performance measure of FH-SS systems is ρ^*, the minimum fractional jamming bandwidth required to induce any decoding errors. In this case, turbo codes save about .1 and .2 respectively.

While turbo codes seem to significantly outperform the RS and RS-CC codes, it is unfair to directly compare these results. First, the coding rates of the coding

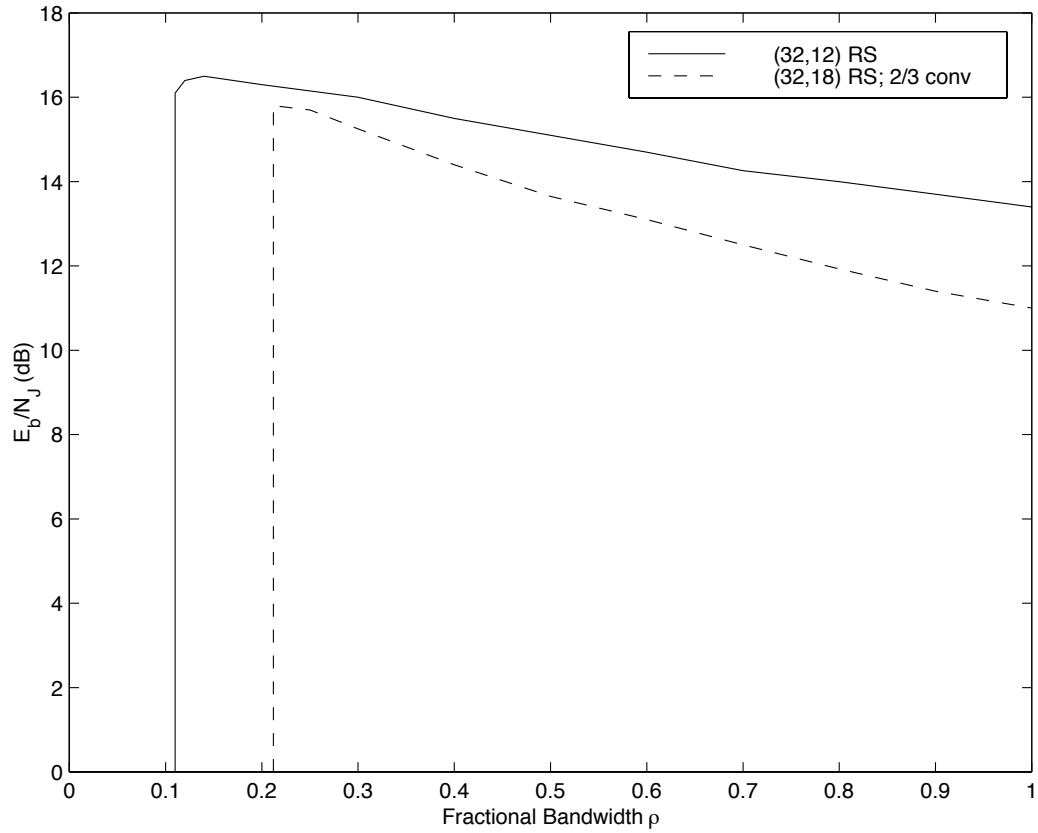

Figure 3.6: Performance of Other Codes in Noncoherent FH-SS with Partial-Band Interference

systems are different: 1/3 for the turbo code and 3/8 for the RS and RS-CC codes. However, this minor difference in code rate is not sufficient to explain the contrast in performance. Another difference is in the treatment of the memory case without side information. For turbo codes, the memory of the channel is exploited by estimating channel states and using this information in the branch transition probability calculations. For the RS-CC code, Pursley and Frank also use the memory to predict which hops have been jammed, but they do so in a very different way. The goal is to declare an erasure if the jamming in a dwell interval is sufficiently severe that many of the RS symbols are likely to be in error. The metric they use to estimate the channel is the Hamming distance between the binary code sequence chosen by the Viterbi decoder and the binary sequence that results from making hard decisions on the output of the demodulator. The resulting distance represents their estimate of the number of errors produced by the demodulator, γ. The different methods at which the turbo code and RS-CC systems treat the uncertainty of the channel state makes it difficult to compare the performance of the systems. The final difference between the coding systems is that the turbo decoder is more computationally complex. Even with the recursions of the MAP decoders, these calculations are iterated many times. Thus, we arrive at the familiar tradeoff between computational complexity and performance.

3.6 Numerical Results

3.6.1 Coherent Reception

The numerical results of the bounds are shown in Figures 3.7 and 3.8. The bounds are known to diverge for SNRs below 2 or 3 dB [26], a range close to the area of interest. Thus for coherent reception, we cannot concretely determine the precision of the bounds. Note, however, that for the case with SI, the general shape of the bounds conform to what is expected. At low SNRs, the jamming noise power is sufficiently high, so the code performs most poorly when $\rho = 1.0$. At high SNRs, the jamming power is so low that spreading it across more frequencies effectively makes the noise negligible. Thus, coding performance is best at $\rho = 1.0$ for high SNRs.

For the NSI case, a similar result is obtained at high SNRs. At low SNRs, however, the $\rho = 1.0$ case performs the best. There are two reasons these analytical results may not be accurate. First, recall that for analysis of the NSI case, a suboptimal decoder was used to calculate pairwise error probabilities. It is possible that the bound to the optimal decoder might have a different shape. A second explanation is that bounds are generally loose at low SNRs. Thus, the results of the bounds at low SNRs are unreliable.

Figures 3.9 and 3.10 show BER bounds for rate 1/3, memory 4 convolutional codes with maximal free distance. These were calculated by using standard techniques. The results show similar form to those of the turbo code, but yield a higher BER for a given E_b/N_J.

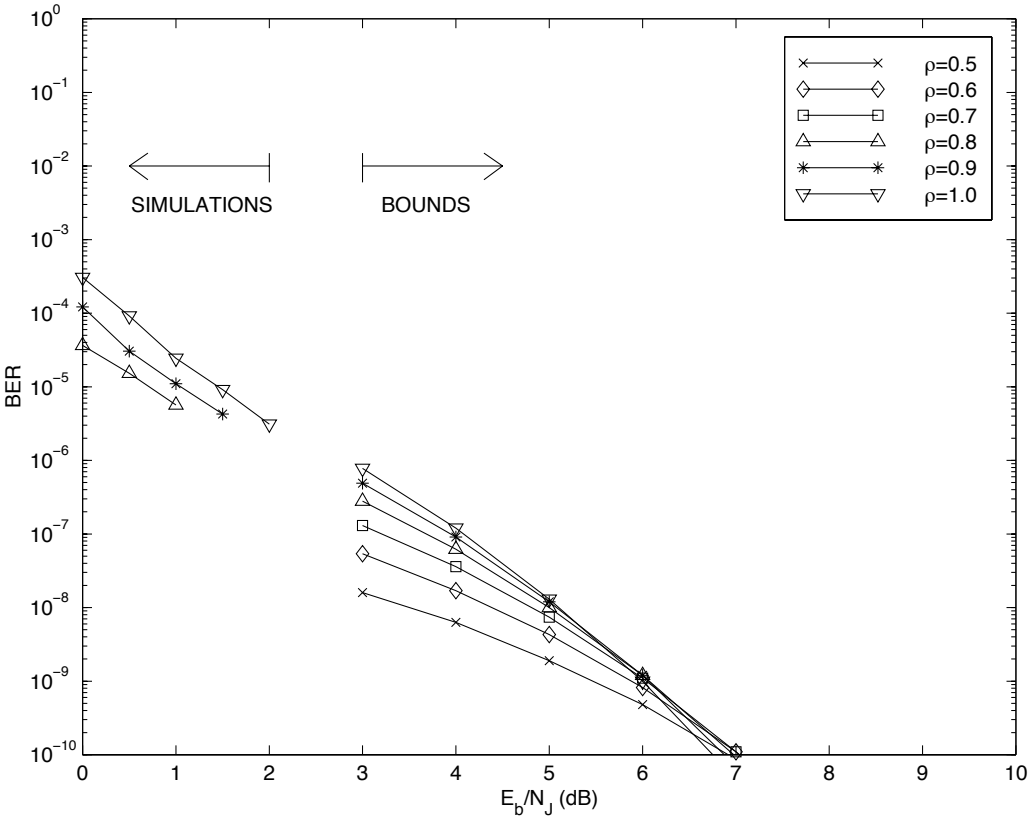

Figure 3.7: Bounds: Turbo Codes in Coherent FH-SS with Partial-Band Interference, With SI

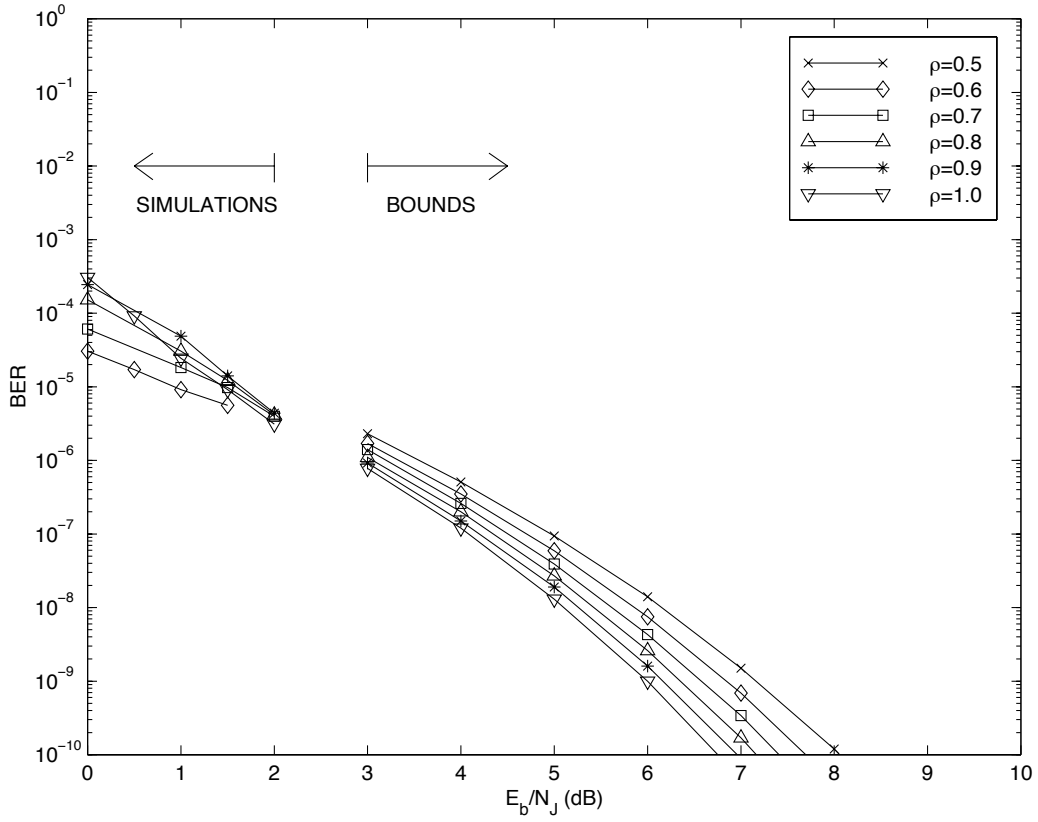

Figure 3.8: Bounds: Turbo Codes in Coherent FH-SS with Partial-Band Interference, No SI

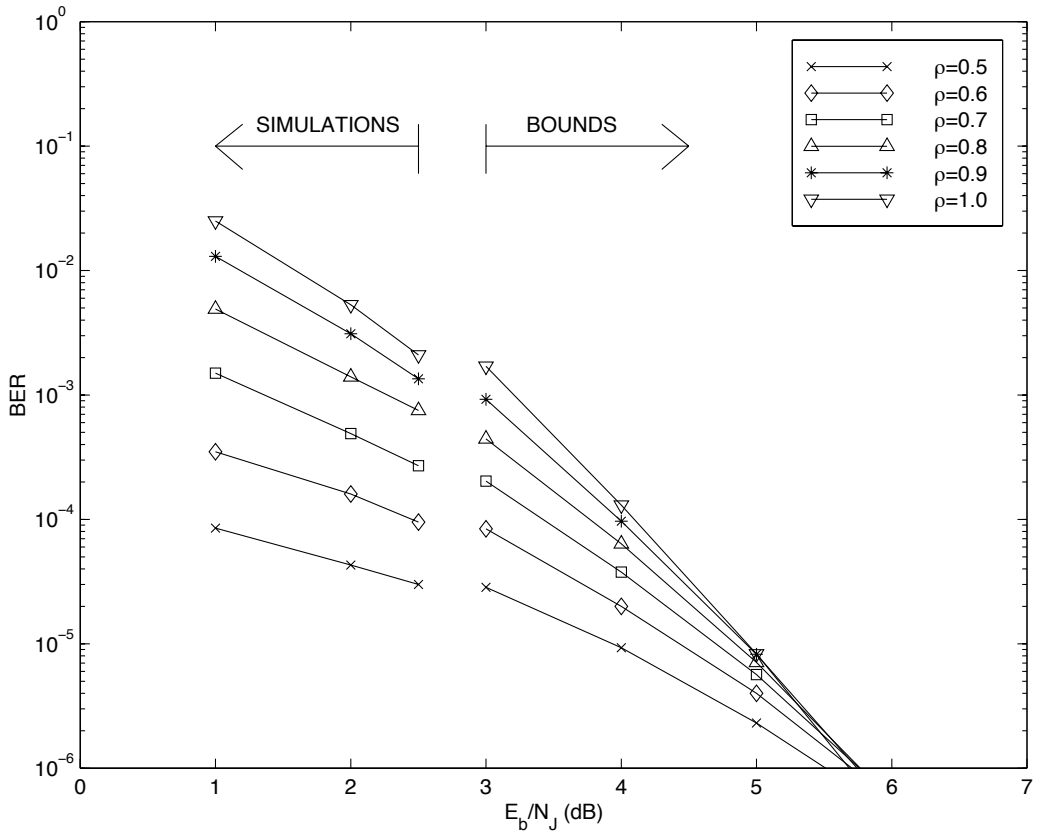

Figure 3.9: Bounds: Convolutional Codes in Coherent FH-SS with Partial-Band Interference, With SI

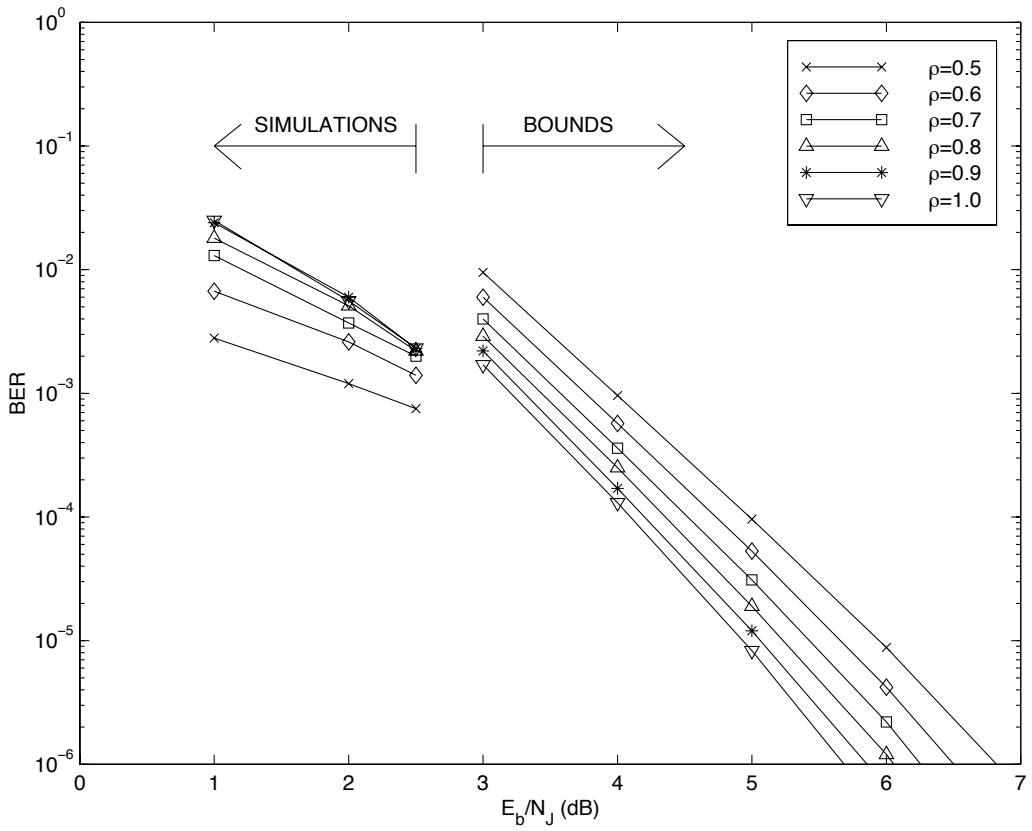

Figure 3.10: Bounds: Convolutional Codes in Coherent FH-SS with Partial-Band Interference, No SI

3.6.2 Noncoherent Reception

Figure 3.11 shows the numerical results for noncoherent reception when side information is available to the receiver. In addition, some simulation results are included for reference. Note that the Bhattacharrya parameter, D, was calculated assuming worst case jamming. As shown in Figure 3.11, the average upper bound calculated for noncoherent reception is tight for $\rho = 1.0$, the value which yields worst case jamming.

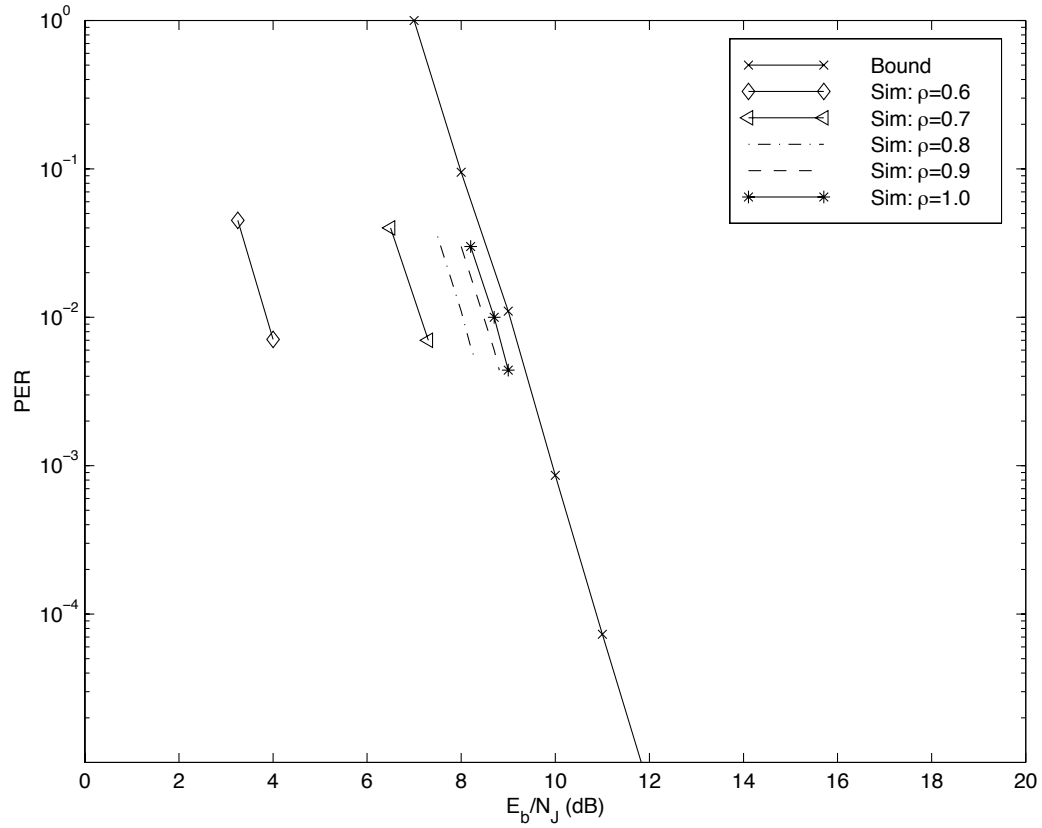

Figure 3.11: Bound: Turbo Codes in Noncoherent FH-SS with Partial-Band Interference, SI

3.7 Conclusion

In this chapter, the design and performance of turbo codes in frequency-hop spread spectrum systems with partial-band interference has been described. The approach was to adapt the calculation of branch transition probabilities inherent in the turbo decoder. In particular for the case with multiple bits per hop, the memory of the channel was exploited by calculating reliable channel state estimates and used these estimates in the turbo decoder. After presenting analytical bounds and simulation results, turbo codes were compared with other well known coding schemes and demonstrated the superior performance of turbo codes in frequency-hop spread spectrum systems with partial band interference. In the next chapter, a frequency-hop spread spectrum system is again considered, but with slow Rayleigh fading rather than partial-band interference.

CHAPTER 4

Turbo Codes in FH-SS with Rayleigh Fading

4.1 Introduction

In the previous chapter, turbo codes were considered in a frequency-hopped spread spectrum system with partial-band interference. An iterative channel estimation procedure was derived to determine which hops had been jammed and the iterative decoding scheme associated with turbo codes was used to mitigate the effects of jamming.

In this chapter, the performance of turbo codes is investigated in a slow Rayleigh fading channel. In particular, a frequency-hopped spread spectrum system with full-band thermal noise and Rayleigh fading is considered. For cases where the data rate exceeds the hopping rate (i.e. there exists multiple bits per hop) and there exists no side information about the fading amplitudes, the approach is to iteratively estimate the fading levels. If these estimates are reasonably accurate, they can be utilized by the turbo decoder in the calculation of bit likelihoods to improve overall performance.

Simulation is performed for coherent and noncoherent reception, variable number of bits per hop, and cases where fading side information is available or unavailable to the decoder. It is shown that iterative channel estimation performed in conjunction with iterative decoding can improve the overall decoding performance. Finally, the performance of a FH-SS system using standard fading assumptions is compared to the performance of a measured fading channel. Due to inaccuracies in the assumptions, the measured fading channel yields performance several decibels higher than the ideal fading channel.

The remainder of this chapter is organized as follows. In Section 4.2, the system model is presented, including descriptions of the transmitter, channel, and the original turbo decoder. In Section 4.3, the iterative decoding and iterative channel estimation schemes are presented for the given system model. Analytical bounds are described in Section 4.4. Simulation and analytical results are presented in Section 4.5. In Section 4.6, the methods are applied to a measured fading channel and the results are compared to those described in the previous section. The results of Section 4.6 are extended to a concatenated code consisting of a turbo code and a repetition code in Section 4.7. Finally, the work is summarized in Section 4.8.

4.2 System Model

4.2.1 Transmitter

The transmitter is exactly the same as described in the previous chapter (Section 3.2.1), thus it will not be repeated here.

4.2.2 Channel

Transmission occurs over a channel that includes full-band thermal noise and slow Rayleigh fading. The thermal noise has power spectral density $\frac{N_0}{2}$ and is assumed to take on a Gaussian distribution. The fading is assumed to be slow relative to the hopping rate such that the instantaneous fade amplitude is constant over each hop. In addition, it is assumed that the frequency separation between frequency slots is large relative to the coherence bandwidth. The coherence bandwidth is the smallest frequency separation such that the channel response is uncorrelated at that frequency separation. Thus, if the frequency separation between frequency slots is large relative to the coherence bandwidth, the fade amplitudes can be assumed to be independent between hops. Let $(y_{0,k}, y_{1,k}, y_{2,k})$ be the demodulator outputs that correspond to d_k, $p_{1,k}$, and $p_{2,k}$, respectively. For the case of coherent detection, the model of the modulating channel and demodulator is

$$y_{i,k} = \sqrt{E}\, a_{i,k}\, c_{i,k} + \eta_{i,k}, \qquad i = 0, 1, 2; \quad k = 1, ..., N \tag{4.1}$$

where the random variable $a_{i,k}$ represents the fade level for $y_{i,k}$ and takes on the normalized density, $f(a) = 2ae^{-a^2}$. In addition, $(c_{0,k}, c_{1,k}, c_{2,k}) = ((-1)^{d_k}, (-1)^{p_{1,k}}, (-1)^{p_{2,k}})$ maps the binary encoded bits to $\{-1, +1\}$, and $\eta_{i,k} \sim N(0, \frac{N_0}{2})$ represents the Gaussian thermal noise.

Next, the channel outputs are described for the case of noncoherent detection. Let $c_{i,k} \in \{-1, +1\}$ denote the coded bit input to the BFSK modulator. Then, the FSK matched filter outputs can be expressed as

$$x_{i,k}^{(c,+1)} = \sqrt{E}\, a_{i,k}\, \delta_{c_{i,k},+1} \cos(\theta_{i,k}) + \eta_{i,k}^{(c,+1)} \tag{4.2}$$

$$x_{i,k}^{(s,+1)} = \sqrt{E}\, a_{i,k}\, \delta_{c_{i,k},+1} \sin(\theta_{i,k}) + \eta_{i,k}^{(s,+1)} \tag{4.3}$$

$$x_{i,k}^{(c,-1)} = \sqrt{E}\, a_{i,k}\, \delta_{c_{i,k},-1} \cos(\theta_{i,k}) + \eta_{i,k}^{(c,-1)} \tag{4.4}$$

$$x_{i,k}^{(s,-1)} = \sqrt{E}\, a_{i,k}\, \delta_{c_{i,k},-1} \sin(\theta_{i,k}) + \eta_{i,k}^{(s,-1)} \tag{4.5}$$

where the unknown phase, $\theta_{i,k}$, is a uniform random variable from 0 to 2π; $\delta(x, y) = 1$ if $x = y$ and $\delta(x, y) = 0$ otherwise; and $\eta_{i,k}^{(c,+1)}$, $\eta_{i,k}^{(s,+1)}$, $\eta_{i,k}^{(c,-1)}$, and $\eta_{i,k}^{(s,-1)}$ are independent Gaussian random variables with zero mean and variance $\frac{N_0}{2}$.

4.3 Turbo Decoder for FH-SS

In this section, the modifications to the turbo decoder for the FH-SS system is described. The turbo decoding algorithm is dependent on what information is available to the turbo decoder. Perfect knowledge of the instantaneous fade amplitudes is referred to as fading side information (SI). The design and performance of turbo

codes is investigated when fading SI is available or unavailable to the decoder. Although fading SI is generally unavailable to a decoder, it is interesting to consider the design and performance of systems with fading SI. In particular, if a receiver tries to estimate the instantaneous fading amplitudes, its BER performance will be lower bounded by the performance of the system with fading SI. In addition to considering cases with and without fading SI, the cases of independent and identically distributed (IID) transmission (i.e. one bit per hop) and transmission over a channel with memory (i.e. h bits per hop) are considered.

4.3.1 FH-SS without Memory

In this section, FH-SS with one bit per hop is considered. The approach is to adapt the calculation of the branch transition probabilities in (2.6). The modifications to these calculations are dependent on the forms of SI available to the decoder. First consider the case of coherent detection with and without fading side information

Coherent Detection, No Fading Side Information

The first case considered is the one where no fading side information (NSI) is available to the turbo decoder. With just one bit per hop, the estimation of fade amplitudes would be unreliable. Hence, for this case, branch transition probability calculations (2.6) are simply averaged over the fading density.

$$\begin{aligned} & p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m') \\ &= \int_0^\infty p(y_{j,k}|a_{j,k} = a, d_k = i, S_k = m, S_{k-1} = m')\, p(a_{j,k} = a)\, da \end{aligned} \tag{4.6}$$

$$= \int_0^\infty \frac{1}{\sqrt{\pi\, N_0}} e^{-\frac{(y_{j,k}-\sqrt{E}\, a\, c_{j,k})^2}{N_0}}\, 2\, a\, e^{-a^2}\, da \tag{4.7}$$

$$= \frac{\sqrt{N_0}}{\sqrt{\pi}(E+N_0)}\, e^{-\frac{y^2}{N_0}} + \frac{2\sqrt{E}\, c_{j,k}\, y_{j,k}}{(E+N_0)^{3/2}}\, e^{-\frac{y^2}{E+N_0}}\, Q\left(-\sqrt{\frac{2E}{N_0}}\, \frac{c_{j,k}\, y_{j,k}}{\sqrt{E+N_0}}\right) \tag{4.8}$$

where the data bit, d_k, and the trellis state transition, $S_k = m$ and $S_{k-1} = m'$, determine the associated coded bit, $c_{j,k}$.

Coherent Detection, Fading Side Information

Next, the case with fading SI is considered. In this case, the channel outputs are the side information and the matched filter outputs. Hence, in computing (2.6), the following joint conditional probability of information received is used.

$$p(y_{j,k}, a_{j,k} | d_k = i, S_k = m, S_{k-1} = m')$$

$$= p(y_{j,k} | a_{j,k}, d_k = i, S_k = m, S_{k-1} = m')\, p(a_{j,k}) \tag{4.9}$$

$$= \frac{1}{\sqrt{\pi\, N_0}} e^{-\frac{(y_{j,k}-\sqrt{E} a_{j,k} c_{j,k})^2}{N_0}}\, 2\, a_{j,k}\, e^{-a_{j,k}^2} \tag{4.10}$$

Noncoherent Detection, No Fading Side Information

As in the case of coherent detection, fading estimation is not performed for the IID, NSI case. If the fading level and received phase are unknown (random), the signal portion of the BFSK receiver output is also Gaussian. Thus, if

$$y_{j,k}^{(+1)} = (x_{j,k}^{(c,+1)})^2 + (x_{j,k}^{(s,+1)})^2 \tag{4.11}$$

$$y_{j,k}^{(-1)} = (x_{j,k}^{(c,-1)})^2 + (x_{j,k}^{(s,-1)})^2 \tag{4.12}$$

then $y_{j,k}^{(+1)}$ and $y_{j,k}^{(-1)}$ are chi-square random variables with two degrees of freedom.

$$p(y_{j,k}^{(l)}|c_{j,k}=c) = \begin{cases} \frac{1}{N_0+1}\, e^{-\frac{y_{j,k}^{(l)}}{N_0+1}} & \text{if } c = l \\ \frac{1}{N_0}\, e^{-\frac{y_{j,k}^{(l)}}{N_0}} & \text{if } c \neq l \end{cases} \tag{4.13}$$

In addition, if $\mathbf{y_{j,k}} = \{y_{j,k}^{(+1)}, y_{j,k}^{(-1)}\}$, then

$$p(\mathbf{y_{j,k}}|c_{j,k}) = p(y_{j,k}^{(+1)}, y_{j,k}^{(-1)}|c_{j,k}) \tag{4.14}$$

$$= p(y_{j,k}^{(+1)}|c_{j,k})\, p(y_{j,k}^{(-1)}|c_{j,k}). \tag{4.15}$$

Thus, branch transition probabilities in (2.6) can be computed using (4.13) and (4.15).

Noncoherent Detection, Fading Side Information

If fading side information exists, then (4.2)-(4.5) assume a form similar to the noncoherent reception of BFSK modulated signals in AWGN with $\tilde{E} = a_k^2 E$. In this case, the branch transition probability can be calculated using a noncentral chi-square density.

$$p(y_{j,k}^{(l)}, a_{j,k}|c_{j,k}=c) = p(y_{j,k}^{(l)}|c_{j,k}, a_{j,k})\, p(a_{j,k}) \tag{4.16}$$

$$= \begin{cases} \frac{1}{N_0}\, e^{-\frac{y_{j,k}^{(l)}+\tilde{E}}{N_0}}\, I_0\left(\frac{2\sqrt{y_{j,k}^{(l)}\tilde{E}}}{N_0}\right)\, p(a_{j,k}) & \text{if } c = l \\ \frac{1}{N_0}\, e^{-\frac{y_{j,k}^{(l)}}{N_0}}\, p(a_{j,k}) & \text{if } c \neq l \end{cases} \tag{4.17}$$

Similar to the previous case, branch transition probabilities can be computed using a combination of (4.15) and (4.17).

4.3.2 FH-SS with Memory

In the previous section, estimates of the fading levels were not explicitly computed for the IID, NSI case since it was assumed that estimates based on just one channel output would be unreliable. In this section, the modification to the turbo decoder for FH-SS with h bits per hop, $h > 1$, is considered.

First, the design of the hopping structure is detailed. The hopping structure is important for cases with memory because unlike the IID case where deeply faded hops affect single bits, such hops now affect multiple bits. For instance, if $c_{0,k}$, $c_{1,k}$, and $c_{2,k}$ are the encoded bits corresponding to information bit d_k, then it would be beneficial to send these bits over separate hops. If they were sent over the same hop and that hop was deeply faded, it would be difficult to decode that information bit correctly. In addition, because the convolutional encoders display memory (i.e. $c_{1,k}$ is dependent on $c_{0,k}$, $c_{0,k-1}$, $c_{0,k-2}$, and so on), it makes sense for similar reasons to separate consecutively coded bits by as much as possible. Using $L = N/h$, the structure of the hopper which satisfies the above requirements is identical to the one described in Table 3.1. Thus, it will not be repeated here.

For cases with multiple bits per hop and no side information, the lack of side information is compensated by generating estimates of the channel. In Chapter 3, turbo

codes for FH-SS with partial-band interference was considered. Letting z_k represent the state of the channel (i.e. whether the channel is jammed or unjammed), the approach in Chapter 3 was was to calculate *a posteriori* probabilities for each jamming state, $p(z_k|\mathbf{y_0}, \mathbf{y_j})$ for $j = 1, 2$, and send this information in addition to $p(d_k|\mathbf{y_0}, \mathbf{y_j})$ between decoders. Thus, information bit estimates and jamming state estimates were iteratively improved. If it is assumed that the fading amplitude is constant over the entire hop, the analogous task of computing *a posteriori* probabilities for quantized values of fading amplitudes can be performed.

Note that the structure of the hopper shown in Table 3.1 allows channel estimates to be calculated in a manner similar to the way information bit estimates are calculated in the original turbo decoder. For instance, each MAP decoder in the original turbo decoder calculates *a posteriori* probabilities of the information bits given two of the three received observation vectors. This information is passed to the next MAP decoder which uses this information as *a priori* information bit probabilities. Similarly, by using the structure of the hopper in Table 3.1, two-thirds of the relevant observation sequence is available to each MAP decoder so that *a posteriori* probabilities for each hop can be calculated. This information can be passed between MAP decoders and be used as *a priori* channel state probabilities.

If the iterative fade estimates can be computed reliably, they can be used in the turbo decoder to improve performance. The turbo decoder computes likelihoods estimates based on the channel outputs, $y_{i,k} = \sqrt{E}\, a_{i,k}\, c_{i,k} + \eta_{i,k}$, which are in turn a function of the transmitted data and the channel conditions. If reliable information regarding the channel is known, the turbo decoder will be more successful in esti-

mating information bits. In a similar fashion, channel estimates can be improved if reliable estimates of the information bits are available. Thus, in addition to potential improvements in error performance, joint iterative decoding and channel estimation may also improve the convergence rate. Quicker decoder convergence is equivalent to fewer decoder iterations which may offset the complexity introduced by the iterative estimation scheme.

The general procedure is as follows. At each component decoder, the information bit estimates generated by the previous component decoder will be used to compute fading estimates. Next, the decoder will compute a new bit likelihood using these fading estimates and pass this information to the next component decoder. This process will be iterated several times. Having summarized the iterative decoding and estimation process, the details of computing fading estimates and using these estimates in the turbo decoding algorithm can be described.

The calculation of fading estimates is analogous to the manner in which jamming state estimates were computed in Chapter 3. Let $\mathbf{R}$ be the vector of $2N$ received channel outputs available to the MAP decoder and let $k = 1, ..., \frac{3N}{h}$ enumerate each hop. Furthermore, let $\mathbf{R} = \mathbf{R_k} \cup \tilde{\mathbf{R}}_\mathbf{k}$ where $\mathbf{R_k} = \{R_{k,i}\}_{i=1}^{h}$ are the channel outputs received over hop k and $\tilde{\mathbf{R}}_\mathbf{k} = \{\tilde{R}_{k,i}\}_{i=1}^{2N-h}$ are the channel outputs which are not received over hop k. Finally, let a_k be the fade amplitude for all information received hop k. Consider the quantization of the fading amplitudes into Q regions, $B_1, ..., B_Q$, and the corresponding Q output levels or centroids, $l_1, ..., l_Q$ where $l_i \in B_i$. The Lloyd-Max quantizer is utilized to obtain the quantized levels and regions. Then, the calculation of *a posteriori* fading probabilities is as follows for $i = 1, ..., Q$.

$$p(a_k \in B_i|\mathbf{R}) = \frac{p(\mathbf{R}|a_k \in B_i) \cdot p(a_k \in B_i)}{p(\mathbf{R})} \quad (4.18)$$

$$= p(\mathbf{R_k}|a_k \in B_i) \cdot p(a_k \in B_i) \cdot \frac{p(\tilde{\mathbf{R}}_\mathbf{k})}{p(\mathbf{R})} \quad (4.19)$$

$$= p(\mathbf{R_k}|a_k \in B_i) \cdot p(a_k \in B_i) \cdot K \quad (4.20)$$

$$\approx p(\mathbf{R_k}|a_k \in B_i) \cdot \tilde{p}(a_k \in B_i|\mathbf{R_k}) \cdot K \quad (4.21)$$

where K is a normalizing factor chosen to make the probability density function sum to 1 and $\tilde{p}(a_k \in B_i|\mathbf{R_k})$ is the fading estimate based on the previous MAP decoder's information bit estimates.

Note that as h, the number of bits per hop, increases, the complexity of directly computing $p(\mathbf{R_k}|a_k \in B_i)$ rises exponentially as a function of the coded bit sequence $\mathbf{c_k}$.

$$p(\mathbf{R_k}|a_k \in B_i) = p(R_{k,1}, ..., R_{k,h}|a_k \in B_i) \quad (4.22)$$

$$= \sum_{c_{k,1},..,c_{k,h}} p(R_{k,1}, ..., R_{k,h}|a_k \in B_i, c_{k,1}, .., c_{k,h})\, p(c_{k,1}, .., c_{k,h}) \quad (4.23)$$

To overcome this problem, the following approximation was developed.

1. Initialize

$$p(a_k \in B_i|R_{k,1}) = \frac{p(R_{k,1}|a_k \in B_i)\, p(a_k \in B_i)}{p(R_{k,1})} \quad (4.24)$$

$$= \sum_{c \in \{-1,+1\}} \frac{p(R_{k,1}|a_k \in B_i, c_{k,1} = c)\, p(c_{k,1} = c)\, p(a_k \in B_i)}{p(R_{k,1})} \quad (4.25)$$

$$\approx \sum_{c\in\{-1,+1\}} \frac{p(R_{k,1}|a_k \in B_i, c_{k,1}=c)p(c_{k,1}=c)\tilde{p}(a_k \in B_i|\mathbf{R_k})}{p(R_{k,1})} \quad (4.26)$$

2. Recursion

$$p(a_k \in B_i|R_{k,1}, R_{k,2}, ..., R_{k,i-1}, R_{k,i})$$

$$= \frac{p(R_{k,i}|a_k \in B_i, R_{k,1}, ..., R_{k,i-1})\, p(a_k \in B_i|R_{k,1}, ..., R_{k,i-1})}{p(R_{k,i}|R_{k,1}, ..., R_{k,i-1})} \quad (4.27)$$

$$\approx \frac{p(R_{k,i}|a_k \in B_i)\, p(a_k \in B_i|R_{k,1}, ..., R_{k,i-1})}{p(R_{k,i}|R_{k,1}, ..., R_{k,i-1})} \quad (4.28)$$

where (4.28) is approximate because memory in the encoder causes the matched filter outputs, $R_{k,i}$, to be correlated. Note, however, that the hopping structure separates consecutive bits by $3L+1$, thus reducing the correlation between bits received over a given hop.

Once the fading estimates have been computed, they can be used in the turbo decoder. When there is no SI, the *a priori* probabilities of the fade levels are replaced by the respective *a posteriori* probabilities for branch transition probability calculations. As in the IID case, the appropriate *a priori* probabilities are used for cases with SI.

Fading NSI

First consider the case without fading side information. The fade amplitudes are quantized to Q regions, $B_1, ..., B_Q$ and then fade estimates are computed. These estimates are used by the turbo decoder in the calculation of branch transition prob-

abilities. Note that the *apriori* probabilities of the fade levels are replaced by the estimates generated by the previous MAP decoder. The branch transition probability calculation for the MAP1 decoder is shown below.

$$
\begin{aligned}
& p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m') \\
&= \sum_{m=1}^{Q} p(y_{j,k}|a_{j,k} \in B_m, d_k = i, S_k = m, S_{k-1} = m')\, p(a_{j,k} \in B_m) \qquad (4.29) \\
&\approx \sum_{m=1}^{Q} p(y_{j,k}|a_{j,k} \in B_m, d_k = i, S_k = m, S_{k-1} = m')\, p(a_{j,k} \in B_m|\mathbf{y_1}, \mathbf{y_3}) \qquad (4.30)
\end{aligned}
$$

Fading SI

For the case where fading side information is available, branch transition probabilities are calculated using the same equations as the IID case, (4.10) for coherent reception, and both (4.15) and (4.17) for noncoherent reception.

$$
\begin{aligned}
& p(y_{j,k}, a_{j,k} = a|d_k = i, S_k = m, S_{k-1} = m') = \\
& \qquad p(y_{j,k}|a_{j,k} = a, d_k = i, S_k = m, S_{k-1} = m')\, p(a_{j,k} = a) \qquad (4.31)
\end{aligned}
$$

4.4 Analytical Bounds

It is often impractical to achieve simulated results for extremely low BERs. As a result, bounds are often calculated. Here, the union bound is invoked to obtain an upper bound on the probability of error.

Turbo codes are linear, so without loss of generality, it will be assumed that the all-zeros codeword was transmitted. Let A_d be the weight enumerator of the code and let $P_2(d)$ be the pairwise error probability between the all-zeros codeword and a codeword of weight d assuming maximum likelihood (ML) decoding. Note that this bound applies to the optimal ML decoder while the turbo decoder which attempts to achieve ML decoding with iterative MAP decoding is suboptimal. The bound for an (n, k) block code is

$$P_{word} \leq \sum_{d=d_{min}}^{n} A_d \, P_2(d). \tag{4.32}$$

The only known way to exactly calculate A_d is via an exhaustive search involving all possible input sequences. One solution is to calculate an average upper bound by computing an average weight function over all possible interleaving schemes [26]. The average weight function in [26] was computed as

$$\overline{A}_d = \sum_{i=1}^{k} \begin{pmatrix} k \\ i \end{pmatrix} p(d|i) \tag{4.33}$$

where $p(d|i)$ is the probability that an interleaving scheme maps an input of weight i to produce a codeword of total weight d and $\begin{pmatrix} k \\ i \end{pmatrix}$ is the number of input sequences with length k and weight i. An algorithm for calculating $p(d|i)$ was described in [26].

Thus,

$$\overline{P}_{word} \leq \sum_{d=d_{min}}^{n} \sum_{i=1}^{k} \begin{pmatrix} k \\ i \end{pmatrix} p(d|i)\, P_2(d). \tag{4.34}$$

The average upper bound to the word error probability, $\overline{P}_{word}$, is computed by summing over the weights of the input sequences, i, and the weights of the resultant codewords, d. A lower bound on the bit error probability can be computed in a similar fashion by noting that for each summed value, there can be no greater than i bit errors out of the k total information bits.

$$\overline{P}_{bit} \leq \sum_{d=d_{min}}^{n} \sum_{i=1}^{k} \frac{i}{k} \begin{pmatrix} k \\ i \end{pmatrix} p(d|i) P_2(d). \tag{4.35}$$

In order to compute the bounds in (4.34) and (4.35), the calculation of pairwise error probabilities must be described. For the case of coherent detection with fading side information, the pairwise error probability conditioned on each of the independent fading amplitudes is calculated below assuming the ML decoder, maximum ratio combining, is employed.

$$P_2(d|a_1, a_2, ..., a_d) = Q\left(\sqrt{\frac{2E_s}{N_0} \sum_{k=1}^{d} a_k^2}\right). \tag{4.36}$$

By applying the upper bound $Q(\sqrt{x+y}) \leq \frac{1}{2} Q(\sqrt{x}) e^{-y/2}$ and averaging over the respective independent fading distributions, the following upper bound [35] on the pairwise error probability can be obtained.

$$P_2(d) = \int_{a_1} \cdots \int_{a_d} p_{\mathbf{A}}(a_1, ..., a_d)\, Q\left(\sqrt{\frac{2\,E_s}{N_0}\sum_{k=1}^{d} a_k^2}\right) da_1 ... da_d \tag{4.37}$$

$$\leq \frac{1}{2}\left(1 - \left[\frac{\Gamma}{1+\Gamma}\right]^{1/2}\right)\left(\frac{1}{1+\Gamma}\right)^{d-1} \tag{4.38}$$

where $\Gamma = \frac{E_s}{N_0}$.

For the case of coherent detection without fading side information, it is difficult to analyze the exact pairwise error probability due to the complexity of (4.8). As a result, two upper bounds on the pairwise error probability are considered. The first bound that is considered was derived by Hagenauer [36]. Consider the approximation of the conditional probability density function (pdf) found in (4.8) by a Gaussian pdf.

$$p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m') \approx \frac{1}{\sqrt{\pi N_0}} \exp\left(y_{j,k} - \sqrt{E_s}E[a]c_k\right)^2/N_0 \tag{4.39}$$

where $E[a]$ is the expectation of each independent fading amplitude with normalized Rayleigh density. Then the following upper bound on the pairwise error probability can be calculated [36].

$$P_2(d) \leq e^{d(\beta/\Gamma)}\left[\frac{\sqrt{1+\frac{2}{\beta}}-1}{\sqrt{1+\frac{2}{\beta}}+1}\right]^d \tag{4.40}$$

where $\beta = \sqrt{\Gamma^2 - 1}$.

The second bound that is considered is the mismatch Chernoff bound [37]. Consider a discrete memoryless channel with input alphabet X, output alphabet Y, and

true transition probabilities $p(y|x)$ where $x \in X$ and $y \in Y$. In addition, consider a block code of length N which consists of M codewords $\mathbf{x_1}$, ..., $\mathbf{x_M}$. The maximum likelihood receiver computes the following decisions regions

$$\Lambda_m = \{\mathbf{y} : p(\mathbf{y}|\mathbf{x_m}) > p(\mathbf{y}|\mathbf{x_{m'}}) \quad \text{for all } m' \neq m\}, \qquad m = 1, ..., M \tag{4.41}$$

and chooses codeword $\mathbf{x_m}$ if $\mathbf{y} \in \Lambda_m$.

There are many instances when the true transition probabilities are not used in the receiver. Conventional receivers are often designed for the ideal linear channel with additive white Gaussian noise, although the actual channel may have nonlinearities and other forms of interference. Another reason for considering mismatched channel statistics is because there are occasions when statistics are difficult to track, perhaps because they are quickly time varying. In any case, assume that the decoder uses $p^*(y|x)$ as the transition probabilities instead of $p(y|x)$. For such cases, the mismatch Chernoff bound can be used to upper bound the pairwise error probability.

Consider a block code of length N which consists of M codewords $\mathbf{x_1}$, ..., $\mathbf{x_M}$. Based on the mismatched channel statistics the following decision regions can be computed.

$$\Lambda_m = \{\mathbf{y} : p^*(\mathbf{y}|\mathbf{x_m}) > p^*(\mathbf{y}|\mathbf{x_{m'}}) \quad \text{for all } m' \neq m\}, \qquad m = 1, ..., M \tag{4.42}$$

In the above equation, the codeword $\mathbf{x_m}$ is chosen if $\mathbf{y} \in \Lambda_m$. Now assume without loss of generality that codeword $\mathbf{x_1}$ is transmitted. If $\mathbf{y} \in \Lambda_m$, then

$$1 \leq \left[\frac{p^*(\mathbf{y}|\mathbf{x_m})}{p^*(\mathbf{y}|\mathbf{x_1})}\right]^{\lambda}, \qquad \lambda \geq 0 \tag{4.43}$$

and the probability of error can be upper bounded as

$$\begin{aligned} P_{E_1} &= \sum_{m=2}^{M} \sum_{\mathbf{y}\in\Lambda_m} p(\mathbf{y}|\mathbf{x_1}) && (4.44)\\ &\leq \sum_{m=2}^{M} \sum_{\mathbf{y}\in\Lambda_m} p(\mathbf{y}|\mathbf{x_1}) \left[\frac{p^*(\mathbf{y}|\mathbf{x_m})}{p^*(\mathbf{y}|\mathbf{x_1})}\right]^{\lambda} && (4.45)\\ &\leq \sum_{m=2}^{M} \sum_{\mathbf{y}} p(\mathbf{y}|\mathbf{x_1}) \left[\frac{p^*(\mathbf{y}|\mathbf{x_m})}{p^*(\mathbf{y}|\mathbf{x_1})}\right]^{\lambda} && (4.46)\\ &= \sum_{m=2}^{M} \prod_{n=1}^{N} \sum_{y} p(y|x_{1n}) \left[\frac{p^*(y|x_{mn})}{p^*(y|x_{1n})}\right]^{\lambda} && (4.47) \end{aligned}$$

where (4.47) assumes a discrete memoryless channel (i.e. $p^*(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{N} p^*(y_i|x_i)$).

The overall error probability can be computed by minimizing over λ the error associated with each codeword, P_{E_i} for $i = 1, .., M$, and averaging the errors over all codewords. In this section, the pairwise error probability of coherent reception without fading side information is determined.

Assume that a binary repetition code of length d is employed and the codewords, $\{x_{1k} = 0\}_{k=1}^{d}$ and $\{x_{2k} = 1\}_{k=1}^{d}$, are transmitted with probability p_0 and p_1, respectively. The mismatched receiver considered here is the equal gain combiner. Thus, $\frac{p^*(y_k|x_{2k})}{p^*(y_k|x_{1k})} = e^y$ and $p(y_k|x_{lk})$ is given in (4.8). For $i \neq j$, the error probability of transmitting $\{x_{ik}\}_{k=1}^{d}$ is

$$P_{e,i} \leq \prod_{k=1}^{d} \int p(y_k|x_{ik}) \left[\frac{p^*(y_k|x_{jk})}{p^*(y_k|x_{ik})}\right]^{\lambda} dy_k \tag{4.48}$$

$$= \left\{\int p(y_k|x_{ik}) \left[\frac{p^*(y_k|x_{jk})}{p^*(y_k|x_{ik})}\right]^{\lambda} dy_k\right\}^{d}. \tag{4.49}$$

The bound in (4.49) is computed using numerical integration. The pairwise error probability is computed by minimizing (4.49) over $\lambda \geq 0$ and averaging the error probabilities over both codewords.

$$P_2(d) = \sum_{i=0}^{1} p_i \min_{\lambda} P_{e,i}. \tag{4.50}$$

For the case of noncoherent detection without fading side information, the Chernoff bound yields the well-known result [32]

$$P_2(d) \leq \left(\frac{4(1+\Gamma)}{(2+\Gamma)^2}\right)^{d}. \tag{4.51}$$

4.5 Simulation and Numerical Results

In the simulations, the component encoders are rate $\frac{1}{2}$ recursive systematic convolutional encoders with memory 4 and octal generators $(37, 21)$. Each packet has 1920 information bits and the number of turbo decoding iterations is 10. Cases with memory are simulated using 20 and 80 bits per hop (bph). The fading amplitudes were quantized to 8 levels ($Q = 8$). Simulation results for the cases with coherent

detection are shown in Figures 4.1, 4.2, and 4.3. Figure 4.1 contains the results for cases with 1 bph, Figure 4.2 contains the results for 20 bph, and Figure 4.3 contains the results for 80 bph.

First, consider the simulation results for cases with 1 bph. As expected, the case with fading side information performs better than the cases without fading side

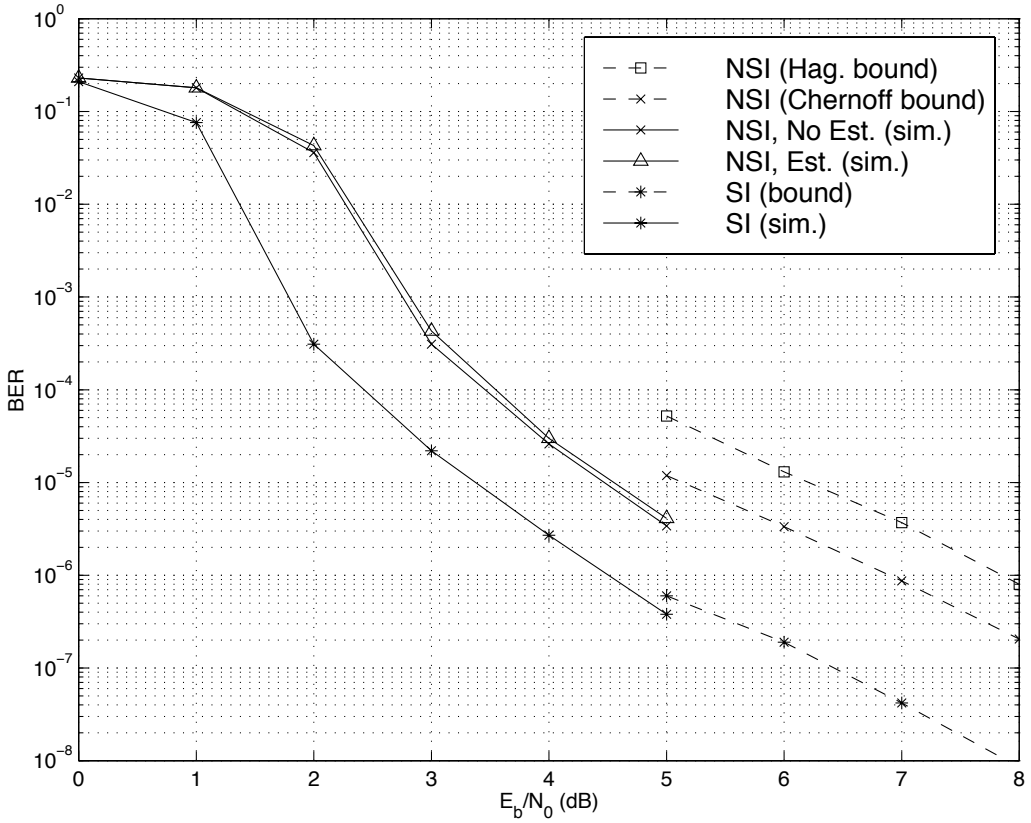

Figure 4.1: Performance of Turbo Codes in Coherent FH-SS with Rayleigh Fading: 1 BPH

information. The gain of side information is approximately 1 dB at a BER of 10^{-5}. For cases without fading side information, one of two methods were applied. The first method calculated branch transition probabilities directly, using (8). In the second method, fading estimates were computed using the iterative procedure outlined in Section 3.3, and these channel estimates were used in the iterative decoder. Note that

for 1 bph, the performance of these two methods are comparable. Fading estimates are inaccurate when based on just one bit of information and thus yield no performance gains. The performance, in fact, is slightly worse despite the additional complexity of iteratively computing fading estimates.

Also shown in Figure 4.1 are the analytical bounds for cases with fading side information and without fading side information (no channel estimation). The Chernoff NSI bound yields an improvement of approximately 1 dB over the Hagenauer NSI bound for BER $< 10^{-5}$. This gain, however, comes at greater complexity due to the minimization of the bound across different SNRs. The bound for fading SI is both tight and simple to compute.

Next, consider the plots with 20 bph shown in Figure 4.2. In this case, the gain of computing fade estimates is approximately 0.7 dB at a BER of 10^{-5} with respect to the NSI, no estimation case. In fact, the estimation procedure is virtually computing perfect side information, as it yields results just a couple tenths of a dB from the SI case. It is also interesting to note that fade estimation for 20 bph yields better performance than the estimation case for 1 bph. Increasing the number of bits per hop increases the reliability of the fading estimates, which in this comparison, improves the system performance. However, it remains to be seen whether increasing the channel memory yields better performance in general for cases without side information for a fixed block length.

Finally, consider the performance for cases with 80 bph shown in Figure 4.3. The SI case and the NSI case with fading estimation yield performance curves that are virtually indistinguishable. With 80 bits of information, fading estimates yield

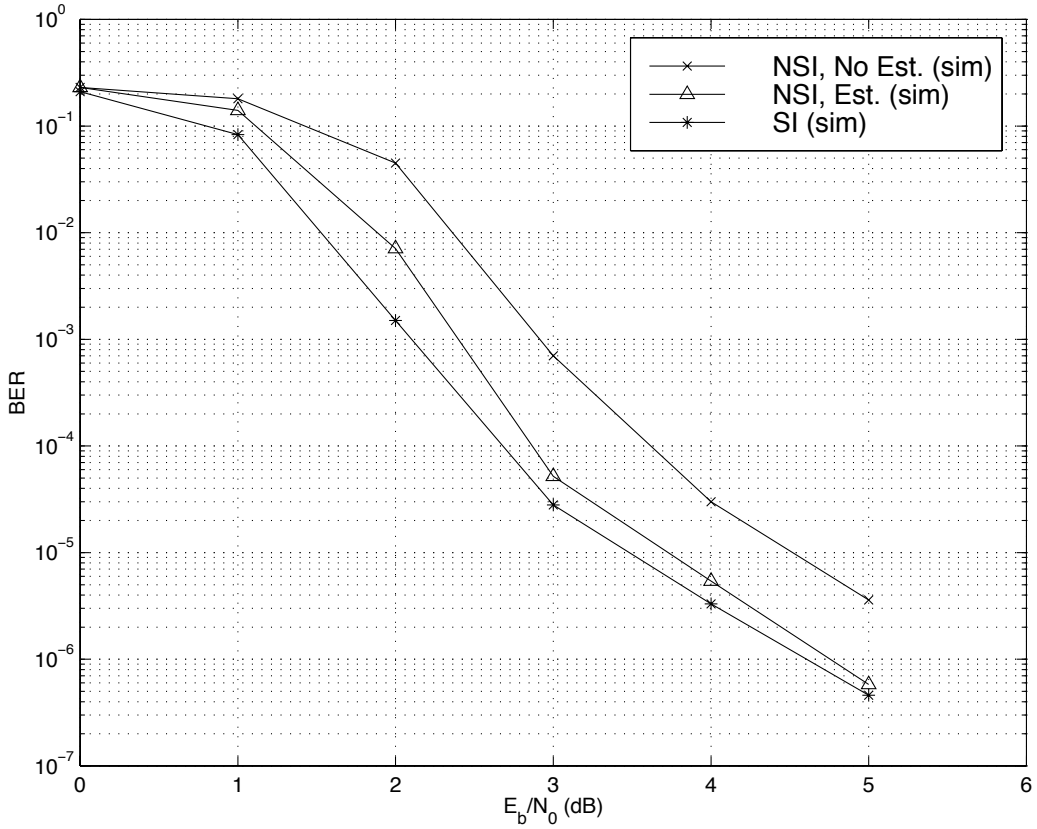

Figure 4.2: Performance of Turbo Codes in Coherent FH-SS with Rayleigh Fading: 20 BPH

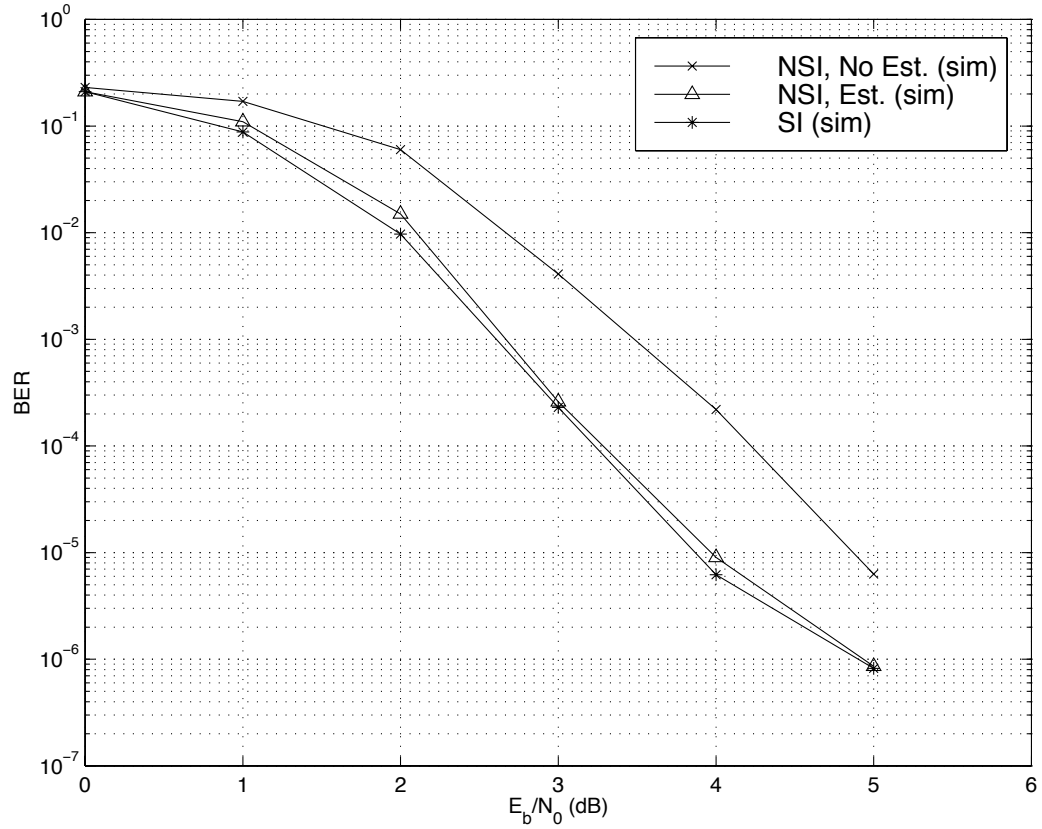

Figure 4.3: Performance of Turbo Codes in Coherent FH-SS with Rayleigh Fading: 80 BPH

nearly perfect SI. Note, however, that while fading estimation at 80 bph yields better performance than estimation at 1 bph, it yields worse performance in comparison with estimation at 20 bph.

The channel memory's beneficial and adverse effects on performance can be explained as follows. Increasing channel memory yields more accurate channel estimates since the estimates are based on more observations. Hence, NSI cases perform more closely to their respective SI cases as the channel memory grows. Increasing the channel memory, however, reduces the total number of independent hops used per packet if the packet size is assumed constant. Fewer independent hops per packet lead to a degradation in performance. This corresponds to the results shown in Figures 4.1, 4.2, and 4.3 where cases with SI yield progressively worse performance as the number of bits per hop increase. Hence, if the number of bits per hop is large such that the estimation process yields virtually perfect SI, increasing the channel memory further will lead to performance degradation. This tradeoff for NSI cases implies the existence of an optimal number of bits per hop to achieve best performance. For this case, this number was approximately 20.

The performance of turbo codes in FH-SS with noncoherent reception is shown in Figures 4.4, 4.5, and 4.6. The results are similar to that of the coherent case. As the number of bits per hop increase, the iterative estimation process performs better, bridging the gap with side information cases. However, as the number of bits per hop increase, BER performance deteriorates because there are fewer independent hops per packet. Here, 20 bits per hop yields the best performance among the performances of 1, 20, and 80 bits per hop.

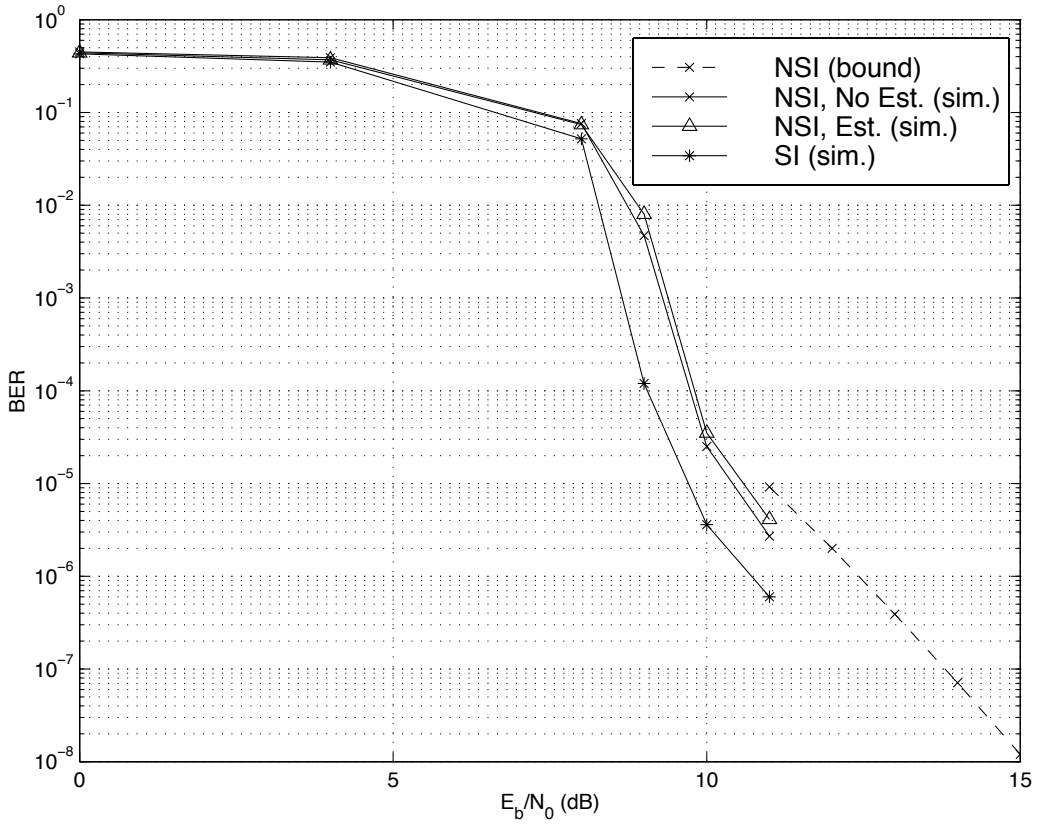

Figure 4.4: Performance of Turbo Codes in Noncoherent FH-SS with Rayleigh Fading: 1 BPH

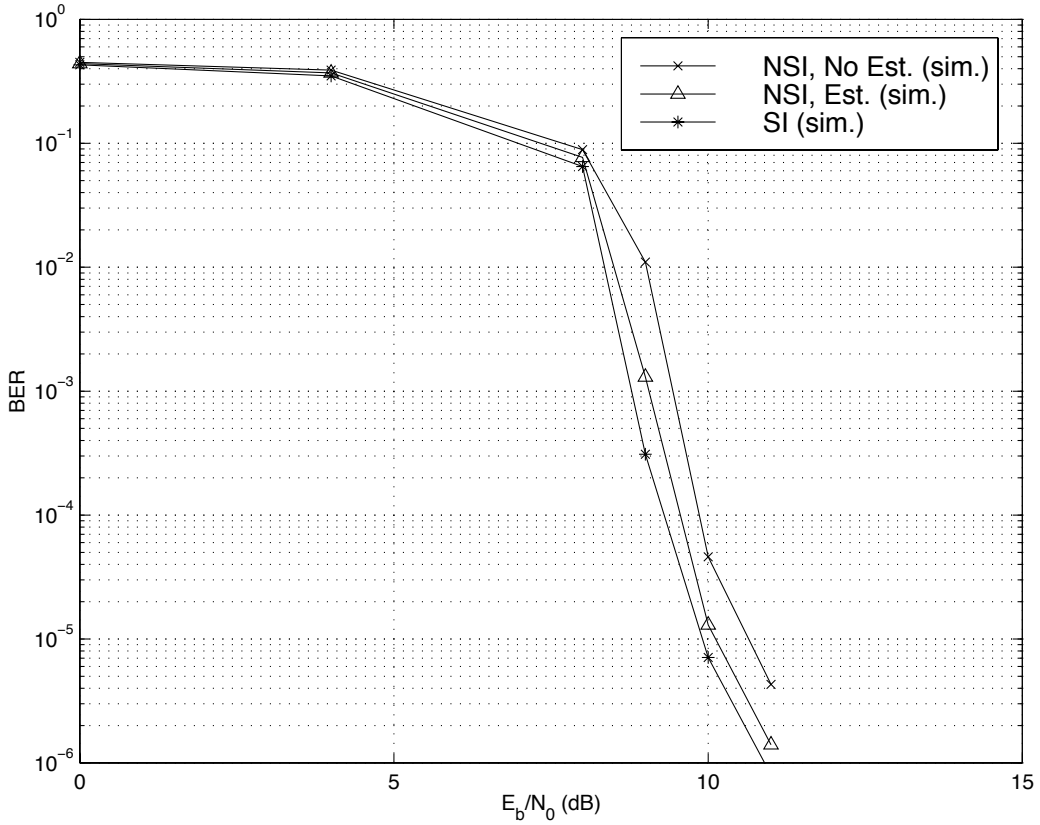

Figure 4.5: Performance of Turbo Codes in Noncoherent FH-SS with Rayleigh Fading: 20 BPH

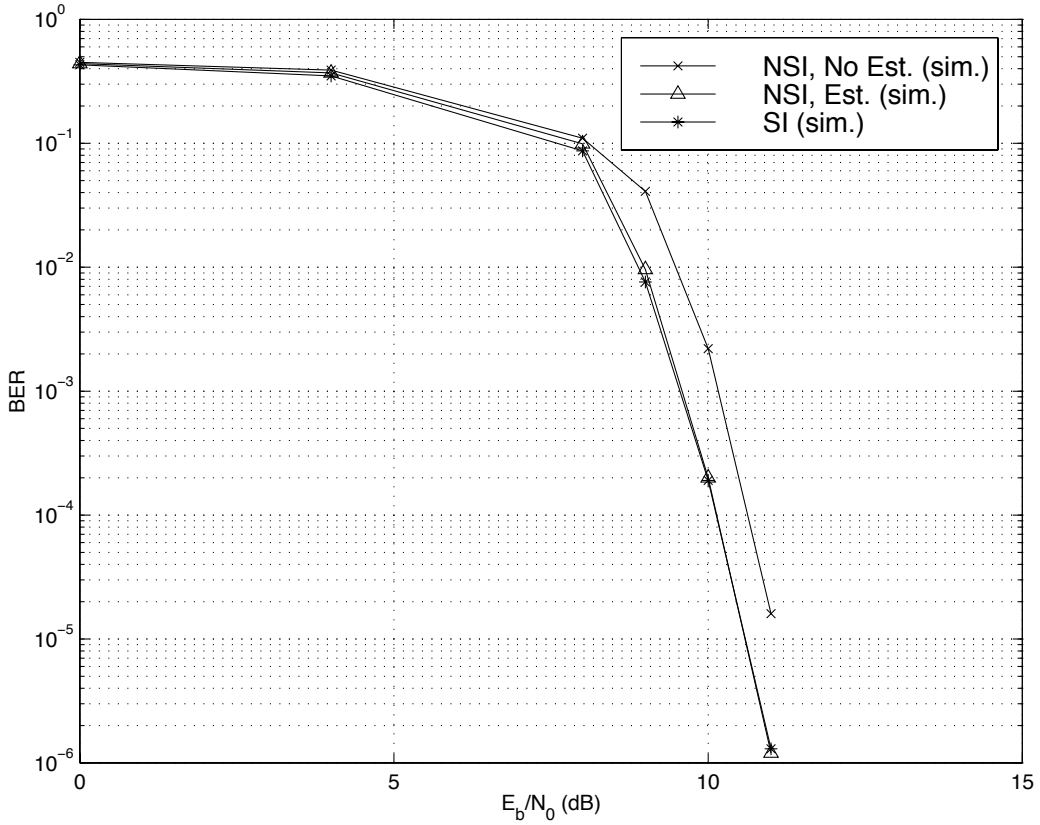

Figure 4.6: Performance of Turbo Codes in Noncoherent FH-SS with Rayleigh Fading: 80 BPH

4.6 Performance of Turbo Codes in FH-SS with Realistic Fading

In this section, the coded performance of frequency-hopped spread spectrum is considered in a measured fading channel. Often times, idealistic channel assumptions are made for analytical convenience. For instance, in previous sections, it was assumed that the fade amplitudes were constant over a hop and independent between hops. While results using such assumptions are important, they may not mimic realistic situations closely. For instance, there may exist time selectivity due to vehicle motion and frequency selectivity depending on the multipath fading profile. The existence of time selectivity means that the fade amplitudes will not be constant over a hop. Depending on the fading profile and the bandwidth of the FH-SS system, it may or may not be a good assumption to assume independence of fade amplitudes between hops. In this section, the performance of two measured channels is considered in order to gauge the relevance of channels which make idealistic fading assumptions. The section is concluded by comparing the measured channel performance of turbo codes to the performance of RS codes used in the Single Channel Ground and Airborne Radio System (SINCGARS) which is a military packet radio network.

4.6.1 System Model

The encoder is the same as the one described in Section 2.1 with $M = 2$, where a data sequence of length N is put into the encoder and for each information bit, three coded bits are produced. The encoded bits are then channel interleaved to

guarantee the hopping structure shown in Table 3.1. BPSK modulation is considered with coherent detection. The resultant signal is frequency hopped. It is assumed that the frequency hopper will choose each of q frequencies or subchannels with uniform probability. If R_b is the data rate, then the bandwidth of each subchannel is $W_{sc} = R_b/R_c$ where $R_c = 1/3$ is the rate of the code. The transmission bandwidth of the system is $W = q * W_{sc}$.

As mentioned above, two measured channels are considered. The first measured channel, called Pine Street (PS) is taken from an urban area and has 12 independent paths [38]. American Legion Drive (ALD) is the second measured channel and it is taken from a suburban area and has 5 independent paths [39]. The delay spreads of ALD and PS are $1.87\,\mu s$ and $2.53\,\mu s$, respectively. The average delay profiles are shown below in Tables 4.1 and 4.2 where the values are rounded to the nearest tenth.

Excess Delay (μs)	0.0	0.3	0.7	1.0	1.6	2.5	3.0	3.7	6.3	8.0	9.5	10.2
Amplitude	0.8	0.6	0.8	1.0	1.0	0.8	0.6	0.9	0.3	0.7	0.5	0.2

Table 4.1: Delay Profile for Pine Street

Excess Delay (μs)	0.0	1.1	4.2	4.8	5.4
Amplitude	1.0	0.8	0.4	0.3	0.3

Table 4.2: Delay Profile for American Legion Drive

The channel model, shown below, takes the standard form for coherent detection, but the fade amplitudes, are taken from the measured channel. Similar to before,

$(c_{0,k}, c_{1,k}, c_{2,k}) = ((-1)^{d_k}, (-1)^{p_{1,k}}, (-1)^{p_{2,k}})$.

$$y_{i,k} = \sqrt{E}\, a_{i,k}\, c_{i,k} + \eta_{i,k}, \qquad i = 0, 1, 2; \quad k = 1, ..., N \tag{4.52}$$

where $a_{i,k}$ is the measured fading amplitude and $\eta_{i,k}$ is IID with density $N(0, N_0/2)$.

If W_{sc} exceeds the hopping rate, R_h, there exist multiple bits per hop. The coherence time is defined as the separation in time for which the channel response is uncorrelated at that time separation. If the coherence time is larger than the duration of each hop, then the bits transmitted over a hop will be correlated (i.e. the fading amplitudes change slowly over a hop). If both of the above conditions are satisfied, fading estimation is employed using the recursion described in Section 4.3.2. Note that different from the previous application, the instantaneous fade amplitudes are not constant over each hop. However, because they are assumed to move slowly, the estimation technique can treat them as being constant without much loss in performance. Thus at the receiver, the tasks of joint channel estimation and turbo decoding will again be employed.

4.6.2 Simulation Results

The simulations for the realistic fading channels of ALD and PS used a transmission bandwidth of approximately 10 MHz with 62 subchannels and a data rate of 54000 bits per second. The packet size is 1920 information bits per packet and cases with 1, 20, and 80 bits per hop are considered. The carrier frequency is set to 38 MHz. For cases with no fading side information, the estimation technique uses 8

levels of quantization (i.e. Q = 8). Ten decoding iterations are considered.

In Figure 4.7, the simulation results for the Pine Street channel at a velocity of 30 meters per second is shown. In order to achieve a BER of 10^{-5} for cases with fading

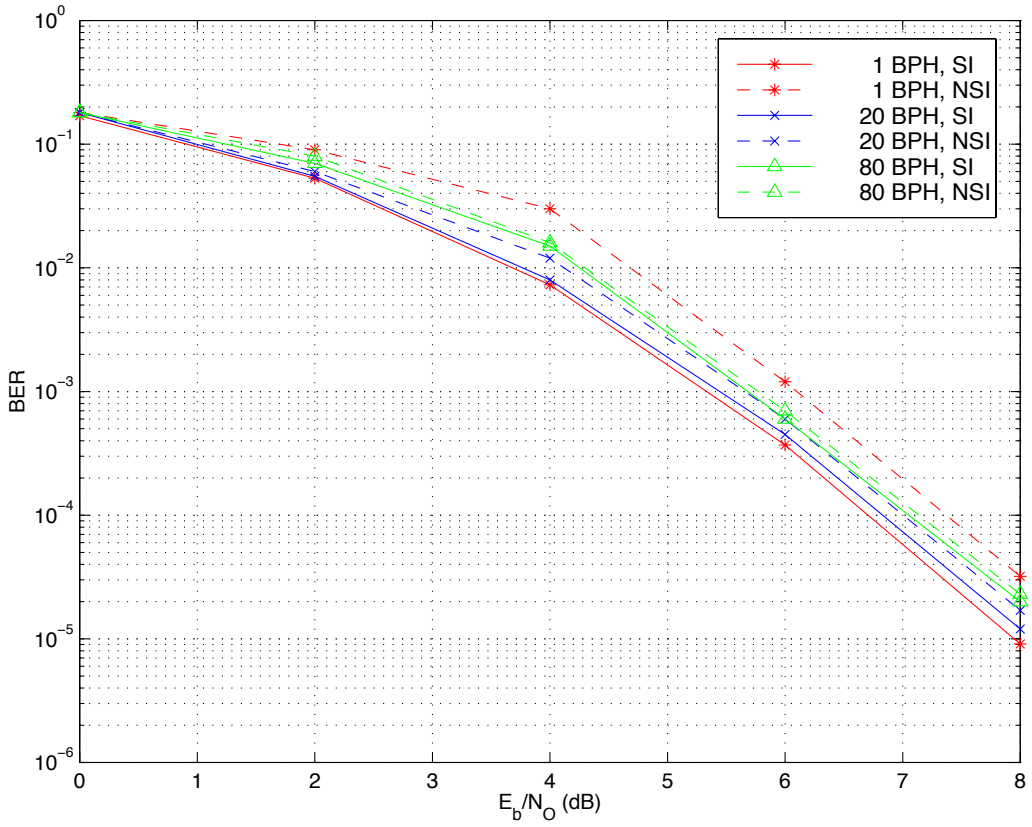

Figure 4.7: Performance of Turbo Codes in Coherent FH-SS with Measured Fading: Pine Street

side information, between 8 and 9 dB is necessary, depending on the number of bits per hop. In contrast, previous results which made "idealistic fading assumptions" (i.e. constant fade over each hop, independence between hops) needed $3-4$ dB to achieve a BER of 10^{-5} (see Figures 4.1, 4.2, 4.3). The root of the approximately 5 dB difference in performance is the level of frequency and time diversity in the channels. Previous results assumed independence between frequency slots, whereas for the measured channels, there is at times strong correlation between slots. Hence, the FH-SS system

with measured channels exhibits less frequency diversity. In addition, the idealistic fading cases assumed constant fades over the duration of a hop, but independence between hops. Hence, for these cases, the correlation time can be approximated by the hop duration, which for 1, 20, and 80 bits per hop are 6 μs, 123 μs, and 494 μs, respectively. In contrast, the coherence times of the Pine Street and American Legion Drive channels at 30 meters per second are both approximately 50 ms. Hence, the measured channels have considerably less time diversity as well.

The simulation results for the American Legion Drive channel at a velocity of 30 meters per second are shown in Figure 4.8. In order to achieve a BER of 10^{-5}, approximately $14 - 15$ dB is needed for cases with fading side information. Hence,

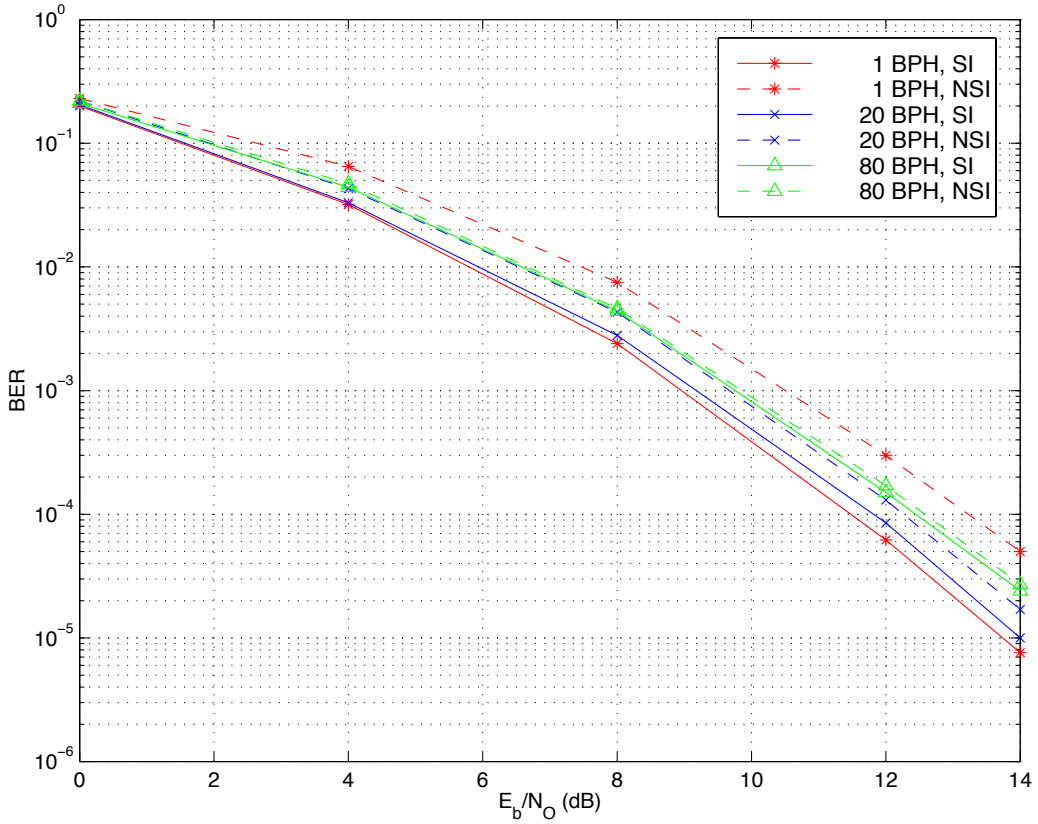

Figure 4.8: Performance of Turbo Codes in Coherent FH-SS with Measured Fading: American Legion Drive

even with respect to the Pine Street Channel, performance has degraded about 6 dB at BER of 10^{-5}. While the coherence time of ALD and PS are approximately the same at 50 ms, the coherence bandwidth for ALD is considerably larger than that of PS. The reason for this can be seen in the delay profiles of the two measured channels shown in Tables 4.1 and 4.2. ALD essentially has two dominant paths with amplitudes 1.0 and 0.8. If these two paths are deeply faded, the entire frequency spectrum is brought down. Thus for the ALD channel, "deep fades" not only occur with more regularity, but the extent of the fades is also broadened.

For both the PS and ALD channels, the tradeoff described for the idealistic channel is evident in Figures 4.7 and 4.8. While channel estimation for 80 bits per hop yielded performance virtually identical to that of the case with perfect SI, its performance is worse than that of the NSI, 20 bits per hop case since there are fewer hops per packet.

In the simulations for the PS and ALD measured channels, the bit errors were extremely bursty. In fact, the vast majority of the packets were corrected within a single iteration. This information can be exploited by adding a cyclic redundancy code (CRC) to detect after each iteration if the packet was decoded correctly. When a packet is decoded error-free, the decoder accepts the next packet for input. Thus, while slightly reducing the effective code rate, a CRC substantially reduces the decoder complexity by saving unnecessary decoder iterations.

While comparisons have been drawn between the performance of turbo codes in ideal and measured fading channels, it would be interesting to compare the performance of turbo codes to the performance of other codes in a measured fading channel. For this comparison, simulations were run over the measured channel called

San Diego Street. The transmission bandwidth of 10.375 MHz is composed of 332 frequency slots with 31.25 kHz per slot. The data rate is 9600 bits per second. The turbo-coded system is similar to the one described above, but instead uses a packet size of 1430 information bits and 130 coded bits per hop. Hence, there are a total of 33 frequency hops per packet. Its performance is compared to the performance of several RS codes.

First, a $(32, 12)$ errors-only Reed-Solomon (RS) code with 24 codewords per packet (i.e. 1440 information bits per packet) is considered. This rate 3/8 code is assumed to transmit 24 RS symbols per hop. Each RS symbol spans 5 bits, so there are 120 bits per hop and a total of 32 hops per packet. In addition, the packet is interleaved across codeword symbols so that each hop contains a symbol from each of the codewords. This code is referred to as RS-1. Table 4.1 shows the hopping structure for the RS-based codes where (i, j) represents the j^{th} symbol of the i^{th} RS codeword.

HOP 1	(1,1)	(2,1)	...	(23,1)	(24,1)
HOP 2	(1,2)	(2,2)	...	(23,2)	(24,2)
.	.	.	.	.	.
.	.	.	.	.	.
HOP 31	(1,31)	(2,31)	...	(23,31)	(24,31)
HOP 32	(1,32)	(2,32)	...	(23,32)	(24,32)

Table 4.3: Structure of Frequency Hopper for RS-Based Codes

The second RS code is a $(32, 12)$ errors and erasures RS code where each 5 bit symbol has a parity bit added. This rate 5/16 code also transmits 24 codewords per packet and 24 RS symbols per hop but now has 144 bits per hop. Two strategies of declaring erasures are considered.

- If the number of parity check errors within a hop exceeds some threshold, γ_T, then all the symbols within the hop are erased. Call this code RS-2. The RS-2 code is similar to the error control code used in SINCGARS, a frequency-hop packet radio network used in the military.

- Each RS symbol with a parity check error is erased. Call this code RS-3.

Figure 4.9 shows the simulation results for turbo and RS-based codes in San Diego Street. First, consider the simulation results of the turbo-coded systems. For 130 bits per hop, the performance of the turbo-coded systems is approximately 6 dB at a BER of 10^{-5}. The gain of side information at this BER is approximately 0.5 dB.

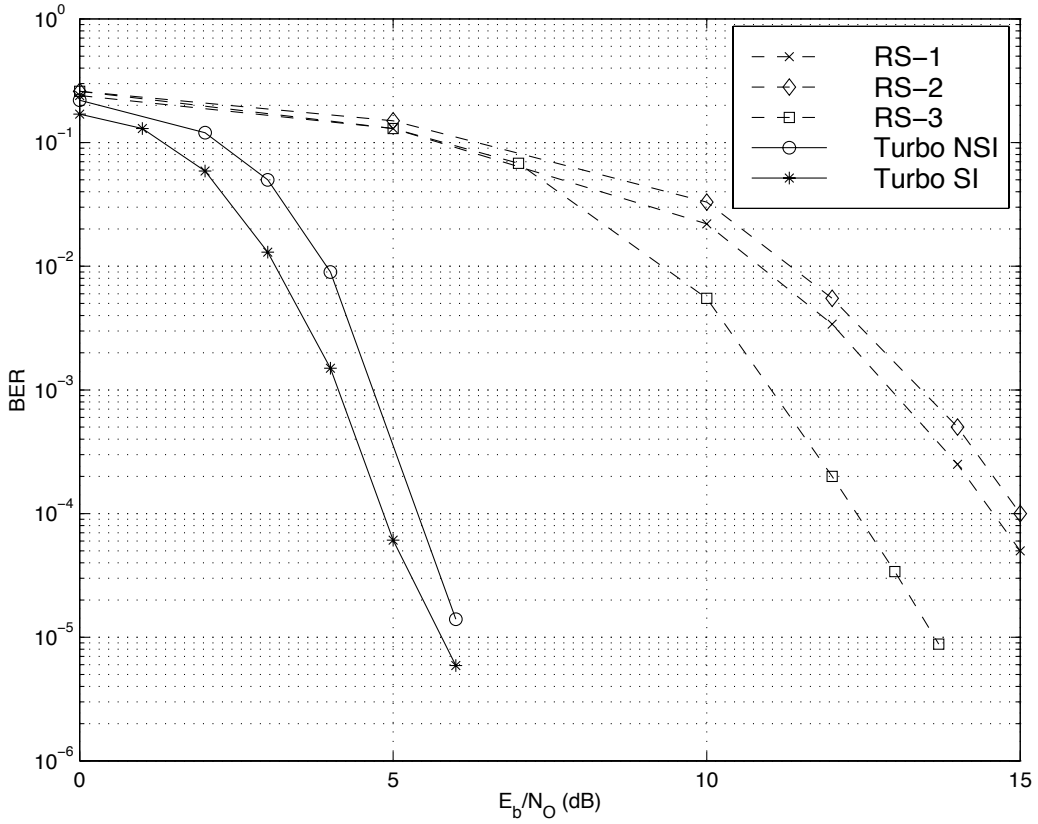

Figure 4.9: Performance of Turbo Codes and Reed-Solomon Codes in Coherent FH-SS with Measured Fading: San Diego St.

Next, compare the performance of the turbo code to the RS-based codes. At BERs of 10^{-5}, turbo codes show a gain of approximately 8 dB. It is interesting to note that the RS-2 code, an errors and erasures decoder, which erases entire hops performs worse than RS-1, the errors only decoder. Plotted in Figure 4.9 is the performance of the best threshold, γ_T, for RS-2. Erasing entire hops is an effective procedure if the channel memory is long. However, in this channel, deep fades do not span entire hop durations. Thus, the γ_T which yields best performance (typically some number around 20) is the one which guarantees that no hops are erased. In effect, RS-2 is an errors-only decoder which performs worse than RS-1 because it expends extra energy transmitting the parity check bits. The more effective errors and erasures decoder is RS-3 which shows a gain of approximately 2 dB with respect to RS-1 at a BER of 10^{-4}. This performance, however, is still far from the performance of the turbo-coded systems. Note that it is not fair to directly compare RS and turbo codes. The turbo decoding algorithm is fairly complex and requires knowledge of channel statistics. The RS code, on the other hand, is a hard decision code which is simpler to decode and does not require any knowledge of the channel.

4.7 Performance of a Concatenated Turbo Code in FH-SS with Realistic Fading

In this section, a concatenated code consisting of a turbo code and a repetition code are considered for a frequency-hopped spread spectrum system. For this system,

the performance of the concatenated turbo code is compared to the performance of a concatenated code consisting of a Reed-Solomon outer code and convolutional inner code.

4.7.1 System Model

The turbo encoder is the same as the one described in Section 2.1 with $M = 2$ where a data sequence of length N is put into the encoder and for each information bit, three coded bits are produced. The turbo-coded bits are then encoded using a length L repetition code. This can alternatively be viewed as a spreading process where each encoded bit is spread using an L chip sequence. These chips are passed through an interleaver which guarantees that each chip within an L chip sequence is transmitted over a different hop. BPSK modulation is considered with coherent detection. The resultant signal is frequency hopped. It is assumed that the frequency hopper will choose each of the q frequencies or subchannels with uniform probability. If R_b is the data rate and R_c is the rate of the code, then the bandwidth of each subchannel can be expressed as $W_{sc} = L * R_b / R_c$. The transmission bandwidth of the system is $W = q * R_c$. A block diagram of the transmitter is shown in Figure 4.10.

Two measured channels are considered: Pine Street and American Legion Drive. Both are described in greater detail in the previous section. The channel model shown below takes a standard form, but the fade amplitudes, are taken from the measured channel. Similar to before, $(c_{0,k}, c_{1,k}, c_{2,k}) = ((-1)^{d_k}, (-1)^{p_{1,k}}, (-1)^{p_{2,k}})$ and $\{y_{i,k,l}\}_{l=1}^{L}$

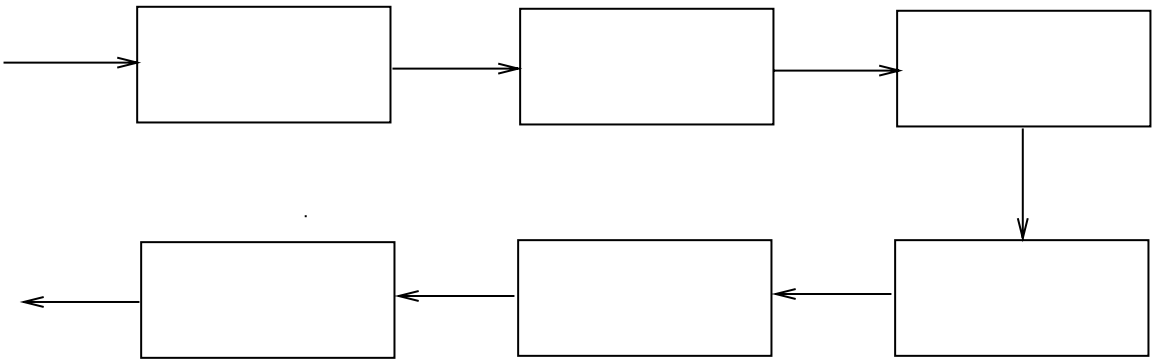

Figure 4.10: Block Diagram of the Transmitter for the FH-SS System

are the L chips corresponding to each coded bit $c_{i,k}$.

$$y_{i,k,l} = \sqrt{E}\, a_{i,k,l}\, c_{i,k} + \eta_{i,k,l}, \qquad i = 0,1,2; \quad k = 1,...,N; \quad l = 1,...,L \tag{4.53}$$

where $a_{i,k,l}$ is the fading amplitude and $\eta_{i,k,l}$ is i.i.d. with density $N(0, N_0/2)$.

Because the turbo decoding algorithm is dependent on what information is available to the decoder, multiple cases will again be considered. In the first case, it will be assumed that the fading amplitudes are perfectly known to the decoder. In the second case, such side information is assumed to be unavailable and thus fade estimates need to be computed.

First, consider the case where fading side information (SI) is available to the decoder. For the system with diversity and known fade levels, maximum ratio combining is optimal. If

$$x_{i,k} = \sum_{l=1}^{L} a_{i,k,l}\, y_{i,k,l} \qquad i = 0,1,2; \quad k = 1,...,N \tag{4.54}$$

then branch transition probabilities (2.6) can be computed using

$$p(x_{i,k}|\{a_{i,k,l}\}_{l=1}^{L}, d_k = i, S_k = m, S_{k-1} = m') = \frac{1}{\sqrt{2\pi\sigma_{x_{i,k}}^2}} e^{-\frac{(x_{i,k}-\mu_{x_{i,k}})^2}{2\sigma_{x_{i,k}}^2}} \tag{4.55}$$

where

$$\mu_{x_{i,k}} = \sqrt{E}\, c_{i,k} \sum_{l=1}^{L} a_{i,k,l}^2 \qquad i = 0, 1, 2; \quad k = 1, ..., N \tag{4.56}$$

$$\sigma_{x_{i,k}}^2 = \frac{N_0}{2} \sum_{l=1}^{L} a_{i,k,l}^2 \qquad i = 0, 1, 2; \quad k = 1, ..., N. \tag{4.57}$$

For the second case, there is no fading side information (NSI) available to the decoder. As before, the approach is to use the information inherent in the memory to compute estimates of the instantaneous fade amplitudes. If fade estimates are computed using the same procedure detailed in Section 4.3.2, maximum ratio combining can be performed using the estimated fading value, $\tilde{a}_{i,k,l}$.

$$\tilde{a}_{i,k,m} = \sum_{n=1}^{Q} l_n\, p(a_{i,k,m} \epsilon B_n | \mathbf{R}), \qquad i = 0, 1, 2; \quad k = 1, ..., N; \quad m = 1, ..., L. \tag{4.58}$$

4.7.2 Simulation Results

The simulations for the realistic fading channels of ALD and PS use a transmission bandwidth of approximately 10 MHz with 62 subchannels, data rate of 9600 bits per second, hopping rate of 9600 hops per second, and a spreading factor of $L = 5$. Thus, the chip rate is 144 Kchips per second, the chip duration is about 6.9 microseconds, and there are 15 chips per hop. Two velocities are considered: 30 meters/second

(m/s) and 0 m/s. The coherence time of the channels at 30 m/s is about 50 ms while at 0 m/s, the coherence time is infinity. The carrier frequency is set to 38 MHz. For cases with no fading side information, the estimation technique use 8 levels of quantization (i.e. $Q = 8$). The simulation results are shown in Figure 4.11 and Figure 4.12 for various levels of side information and different velocities (shown in meters per second). Also shown in the figures are simulation results for a FH-SS system which consists of a $(62, 24)$ RS outer code, a rate $1/2$, constraint length 5 convolutional code, and $(16, 5)$ biorthogonal modulation.

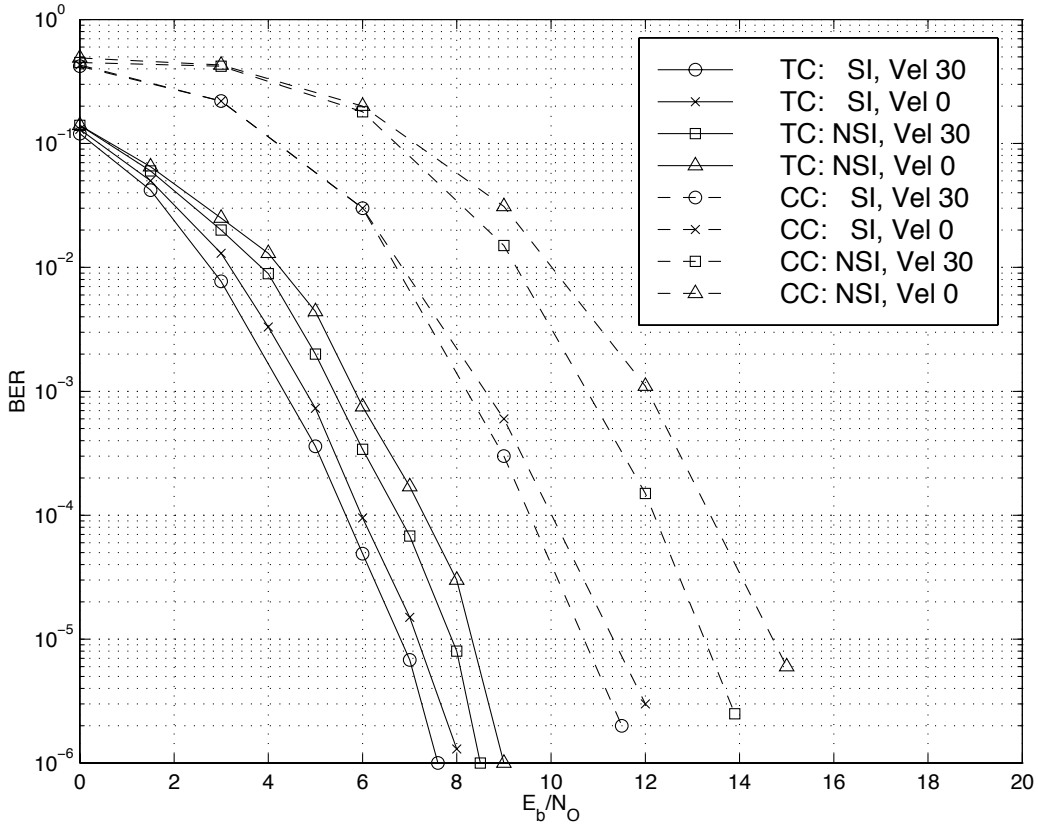

Figure 4.11: Comparison of Codes in FH-SS with Measured Fading: Pine Street

At BERs of 10^{-5}, the turbo code (TC) shows performance improvements ranging from $4 - 7$ dB for Pine Street and $3 - 4$ dB for American Legion Drive over the

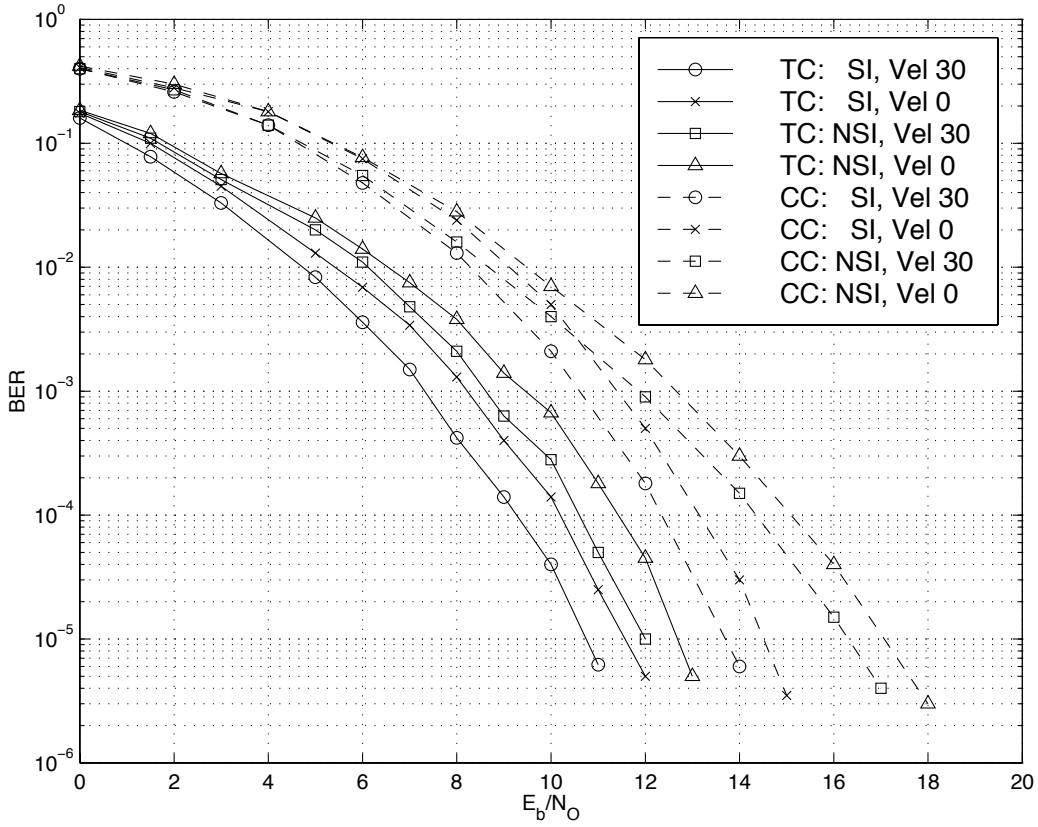

Figure 4.12: Comparison of Codes in FH-SS with Measured Fading: American Legion Drive

concatenated code (CC) consisting of the Reed-Solomon and convolutional codes. The gain was larger for cases without SI due to reliable channel estimates. Finally, note that the performance of an outer turbo code concatenated with an inner repetition code could be improved if a more powerful rate $1/L$ code was considered instead of the repetition code.

4.8 Conclusion

In this chapter, the performance of turbo codes in FH-SS with Rayleigh fading was investigated. The approach was to exploit the channel memory by calculating channel state estimates for unknown channel parameters. Simulation results confirmed that such an approach may be useful in mitigating the effects of fading for FH-SS. In addition, measured fading channels were considered. They were found to yield results considerably worse than that of "ideal" fading channels, which were defined to have constant fade amplitudes over a hop, but independent amplitudes between hops. Further investigation into the performance of channel coding in time and frequency selective channels is needed to properly assess the performance of real fading channels. Finally, turbo codes were shown to yield significant improvement over the codes used for SINCGARS. Such results explain why turbo codes are being considered for next generation military radios.

CHAPTER 5

Turbo Codes in FH-SS with Partial-Band Interference and Rayleigh Fading

5.1 Introduction

Iterative channel estimation paired with the iterative decoding algorithm associated with turbo codes have been shown to yield large performance gains over other commonly known error control codes and decoding algorithms. In Chapter 3, the effective use of jamming state estimates for channels with partial-band interference was demonstrated. In Chapter 4, fading estimates were shown to improve decoding performance. In this chapter, these techniques are applied to FH-SS with both partial-band jamming and slow Rayleigh fading.

The outline for this chapter is as follows. In Section 5.2, the system model will be discussed and in Section 5.3, the turbo decoder subject to this model will be described. Simulation results are presented in Section 5.4. Finally, a brief conclusion is made in Section 5.5.

5.2 System Model

5.2.1 Transmitter

Because the transmitter is exactly the same as the one described in Section 3.2.1, the description will not be repeated here.

5.2.2 Channel

The description of the jammer is analogous to Section 3.2.2. In addition to partial-band interference and full-band thermal noise, the channel has slow Rayleigh fading. The thermal noise has double-sided power spectral density $\frac{N_0}{2}$ and the partial-band interference has double-sided power spectral density $\frac{N_J}{2\rho}$ which covers a fraction ρ of the band. Both types of interference are each assumed to be Gaussian random variables. It is assumed that over each hop, the fading amplitude is constant and that between hops, the fading amplitudes are independent. In addition, each hop is assumed to be entirely jammed if it is jammed at all. For the case of coherent detection, the model for the demodulator outputs is

$$y_{i,k} = \sqrt{E}\, a_{i,k}\, c_{i,k} + \eta_{i,k}, \qquad i = 0, 1, 2; \quad k = 1, ..., N \tag{5.1}$$

where $a_{i,k}$ is a normalized Rayleigh random variable with density $f(a) = 2ae^{-a^2}$, $(c_{0,k}, c_{1,k}, c_{2,k}) = ((-1)^{d_k}, (-1)^{p_{1,k}}, (-1)^{p_{2,k}})$, and $\eta_{i,k} \sim N(0, \frac{N_0}{2} + z_{i,k} \cdot \frac{N_J}{2\rho})$.

For the case of noncoherent detection, the FSK outputs can be represented as the sum of the attenuated signal and Gaussian noise.

$$x_{i,k}^{(c,+1)} = \sqrt{E}\, a_{i,k}\, \delta_{c_{i,k},+1} \cos(\theta_{i,k}) + \eta_{i,k}^{(c,+1)} \tag{5.2}$$

$$x_{i,k}^{(s,+1)} = \sqrt{E}\, a_{i,k}\, \delta_{c_{i,k},+1} \sin(\theta_{i,k}) + \eta_{i,k}^{(s,+1)} \tag{5.3}$$

$$x_{i,k}^{(c,-1)} = \sqrt{E}\, a_{i,k}\, \delta_{c_{i,k},-1} \cos(\theta_{i,k}) + \eta_{i,k}^{(c,-1)} \tag{5.4}$$

$$x_{i,k}^{(s,-1)} = \sqrt{E}\, a_{i,k}\, \delta_{c_{i,k},-1} \sin(\theta_{i,k}) + \eta_{i,k}^{(s,-1)} \tag{5.5}$$

where $a_{i,k}$ is a normalized Rayleigh random variable with density $f(a) = 2ae^{-a^2}$, $c_{i,k}$ is the input to the FSK modulator, $\theta_{i,k}$ is a uniform random variable from 0 to 2π, $\delta_{a,b} = 1$ if $a = b$ and $\delta_{a,b} = 0$ otherwise, and $\eta_{i,k}^{(c,+1)}$, $\eta_{i,k}^{(s,+1)}$, $\eta_{i,k}^{(c,-1)}$, and $\eta_{i,k}^{(s,-1)}$ are Gaussian random variables with zero mean and variance, $\frac{N_0}{2} + z_{i,k}\frac{N_J}{2\rho}$, which depends on whether the state is jammed.

Note that if the phase and fading amplitudes are unknown (random), the signal portion of the FSK receiver output is Gaussian. Thus, if

$$y_{j,k}^{(+1)} = (x_{j,k}^{(c,+1)})^2 + (x_{j,k}^{(s,+1)})^2 \tag{5.6}$$

$$y_{j,k}^{(-1)} = (x_{j,k}^{(c,-1)})^2 + (x_{j,k}^{(s,-1)})^2 \tag{5.7}$$

then $y_{j,k}^{(+1)}$ and $y_{j,k}^{(-1)}$ are chi-square random variables with two degrees of freedom.

5.3 Turbo Decoder for FH-SS with Partial-Band Interference and Rayleigh Fading

The turbo decoding algorithm is dependent on what information is available to the turbo decoder. In this chapter, two forms of side information (SI) are considered. Jamming side information reveals perfect knowledge as to which bits have been jammed. Fading side information yields perfect knowledge of each fading amplitude. We examine cases where jamming and/or fading side information is available (SI) or unavailable (NSI). In addition, the cases of independent and identically distributed (IID) transmission (i.e. one bit per hop) and transmission over a channel with memory (i.e. h bits per hop) are considered.

5.3.1 FH-SS without Memory

In this section, the case of one bit per hop is considered. The approach is to adapt the calculation of the branch transition probabilities in (2.6). The modifications to these calculations is dependent on the forms of side information available to the decoder.

Jamming SI and Fading SI

First, consider the case where both jamming and fading side information are available to the turbo decoder. In this case, (2.6) is calculated using (5.8).

$$p(y_{j,k}, a_{j,k}, z_{j,k} | d_k = i, S_k = m, S_{k-1} = m')$$

$$= \quad p(y_{j,k}|a_{j,k}, z_{j,k}, d_k = i, S_k = m, S_{k-1} = m')\, p(a_{j,k})\, p(z_{j,k}) \tag{5.8}$$

For coherent detection,

$$p(y_{j,k}|a_{j,k}, z_{j,k}, d_k = i, S_k = m, S_{k-1} = m') = \frac{1}{\sqrt{2\pi\sigma^2_{z_{j,k}}}} e^{-\frac{(y_{j,k} - a_{j,k}c_{j,k})^2}{2\sigma^2_{z_{j,k}}}} \tag{5.9}$$

where $\sigma^2_{z_{j,k}} = \frac{N_0}{2} + z_{j,k}\,\frac{N_J}{2\rho}$.

For noncoherent detection,

$$p(y_{j,k}|c_{j,k}, z_{j,k}) \quad = \quad p(y^{(+1)}_{j,k}, y^{(-1)}_{j,k}|c_{j,k}, z_{j,k}) \tag{5.10}$$

$$= \quad p(y^{(+1)}_{j,k}|c_{j,k}, z_{j,k})\, p(y^{(-1)}_{j,k}|c_{j,k}, z_{j,k}) \tag{5.11}$$

and

$$p(y^{(l)}_{j,k}|c_{j,k}, z_{j,k} = z) = \begin{cases} \frac{1}{2\sigma_z^2} \exp[-\frac{y^{(l)}_{j,k} + \tilde{E}}{2\sigma_z^2}] I_0(\sqrt{y^{(l)}_{j,k}\tilde{E}}/\sigma_z^2) & \text{if } c_{j,k} = l \\ \\ \frac{1}{2\sigma_z^2} \exp[-\frac{y^{(l)}_{j,k}}{2\sigma_z^2}] & \text{if } c_{j,k} \neq l \end{cases} \tag{5.12}$$

where $\tilde{E} = a^2_{j,k}\, E$.

Jamming SI, Fading NSI

The second case is the one where jamming side information is available to the decoder, but fading side information is unavailable. For this case, the branch transition probability calculations (2.6) for coherent reception are calculated using

$$
\begin{aligned}
& p(y_{j,k}, z_{j,k} | d_k = i, S_k = m, S_{k-1} = m') \\
&= \int_0^\infty p(y_{j,k}, z_{j,k} | a_{j,k} = a, d_k = i, S_k = m, S_{k-1} = m') p(a)\, da && (5.13) \\
&= \int_0^\infty p(y_{j,k} | a_{j,k} = a, z_{j,k}, d_k = i, S_k = m, S_{k-1} = m')\, 2ae^{-a^2}\, p(z_{j,k})\, da && (5.14) \\
&= \int_0^\infty \frac{1}{\sqrt{2\pi\sigma^2_{z_{j,k}}}} e^{-\frac{1}{2\sigma^2_{z_{j,k}}}(y_{j,k} - a\, c_{j,k})^2}\, 2ae^{-a^2}\, p(z_{j,k})\, da && (5.15) \\
&= \frac{2p(z_{j,k})}{\lambda} e^{-\frac{y^2_{j,k}}{2\sigma^2_{z_{j,k}}}} \left(\frac{\sigma_{z_{j,k}}}{\sqrt{2\pi}} + \frac{y_{j,k} c_{j,k}}{\sqrt{\lambda}} e^{\frac{y^2_{j,k}}{2\lambda\sigma^2_{z_{j,k}}}}\, Q\left(-\frac{y_{j,k} c_{j,k}}{\sigma_{z_{j,k}} \sqrt{\lambda}} \right) \right) && (5.16)
\end{aligned}
$$

where $\lambda = 2\sigma^2_{z_{j,k}} + 1$.

For the case of noncoherent reception, $y^{(+1)}_{j,k}$ and $y^{(-1)}_{j,k}$ are chi-square random variables with two degrees of freedom (Section 5.2.2).

$$
p(y^{(l)}_{j,k} | c_{j,k}, z_{j,k}) = \begin{cases} \frac{1}{2(\sigma^2_{z_{j,k}} + \frac{1}{2})}\, e^{-\frac{y^{(l)}_{j,k}}{2(\sigma^2_{z_{j,k}} + \frac{1}{2})}} & \text{if } c_{j,k} = l \\ \frac{1}{2\sigma^2_{z_{j,k}}}\, e^{-\frac{y^{(l)}_{j,k}}{2\sigma^2_{z_{j,k}}}} & \text{if } c_{j,k} \neq l. \end{cases} \qquad (5.17)
$$

Again,

$$
\begin{aligned}
p(y_{j,k} | c_{j,k}, z_{j,k}) &= p(y^{(+1)}_{j,k}, y^{(-1)}_{j,k} | c_{j,k}, z_{j,k}) && (5.18) \\
&= p(y^{(+1)}_{j,k} | c_{j,k}, z_{j,k})\, p(y^{(-1)}_{j,k} | c_{j,k}, z_{j,k}). && (5.19)
\end{aligned}
$$

Jamming NSI, Fading SI

For the third case, jamming side information is assumed to be unavailable to the decoder, but fading side information is available. In this case, the calculation of branch transition probabilities, shown below, includes a mixture of Gaussians which uses (5.9) for coherent detection and uses (5.11) and (5.12) for noncoherent detection.

$$\begin{aligned} & p(y_{j,k}, a_{j,k} | d_k = i, S_k = m, S_{k-1} = m') \\ &= p(y_{j,k} | a_{j,k}, z_{j,k} = 1, d_k = i, S_k = m, S_{k-1} = m')\, p(a_{j,k})\, p(z_{j,k} = 1) + \\ & \quad p(y_{j,k} | a_{j,k}, z_{j,k} = 0, d_k = i, S_k = m, S_{k-1} = m')\, p(a_{j,k})\, p(z_{j,k} = 0) \end{aligned} \tag{5.20}$$

Jamming NSI, Fading NSI

The final case that is considered is the case where neither jamming nor fading side information is available to the turbo decoder. Branch transition probabilities are calculated using the equation below and a weighted sum of either (5.16) for coherent detection or (5.17) and (5.19) for noncoherent reception.

$$\begin{aligned} & p(y_{j,k} | d_k = i, S_k = m, S_{k-1} = m') \\ &= \int_0^\infty p(y_{j,k} | a_{j,k}, z_{j,k} = 1, d_k = i, S_k = m, S_{k-1} = m')\, p(a_{j,k})\, p(z_{j,k} = 1)\, da + \\ & \quad \int_0^\infty p(y_{j,k} | a_{j,k}, z_{j,k} = 0, d_k = i, S_k = m, S_{k-1} = m')\, p(a_{j,k})\, p(z_{j,k} = 0)\, da \end{aligned} \tag{5.21}$$

5.3.2 FH-SS with Memory

In this section, the modifications to the turbo decoder for the case of multiple bits per hop are discussed. Much of the setup is similar to the cases with memory in Chapters 3 and 4. For cases with multiple bits per hop and no side information, we attempt to compensate for the lack of side information by generating estimates of the channel. In this chapter, both $p(z_k|\mathbf{y_1}, \mathbf{y_j})$ and $p(z_k|\mathbf{y_1}, \mathbf{y_j})$ for $j = 1, 2$ will be computed and sent in addition to $p(d_k|\mathbf{y_1}, \mathbf{y_j})$ between decoders. Thus, information bit estimates and channel state estimates will be iteratively improved.

The calculation of jamming and fading estimates is exactly the same as described in Sections 3.3.2 and 4.3.2. Once both the fading and jamming estimates have been computed, they are ready to be used in the turbo decoder. Fading and jamming estimates are used in a manner analogous to the way that the turbo decoder uses information bit estimates. When there is no SI, the *a priori* probabilities are replaced by the respective *a posteriori* probabilities for branch transition probability calculations. As in the IID case, the appropriate *a priori* probabilities are used for cases with side information. Thus, for the MAP1 decoder, we can again consider the four cases with varying degrees of side information, using (5.9) for cases with coherent detection and using both (5.11) and (5.12) for cases with noncoherent detection.

Jamming SI, Fading SI

For the case with both jamming and fading side information, branch transition probabilities use the same equations as in the IID Case (5.8).

$$p(y_{j,k}, a_{j,k} = a, z_{j,k} = z | d_k = i, S_k = m, S_{k-1} = m') \tag{5.22}$$
$$= \quad p(y_{j,k} | d_k = i, S_k = m, S_{k-1} = m', a_{j,k} = a, z_{j,k} = z)\, p(a_{j,k} = a)\, p(z_{j,k} = z)$$

Jamming SI, Fading NSI

For the case with jamming side information, but no fading side information, the fading amplitudes are quantized to Q levels, $l_1, ..., l_Q$, and utilize the fading estimates provided by the previous MAP decoder.

$$p(y_{j,k}, z_{j,k} = z | d_k = i, S_k = m, S_{k-1} = m')$$
$$= \quad \sum_{t=1}^{Q} p(y_{j,k} | d_k = i, S_k = m, S_{k-1} = m', a_{j,k} = l_t, z_{j,k} = z) p(a_{j,k} = l_t) p(z_{j,k} = z) \tag{5.23}$$
$$\approx \quad \sum_{t=1}^{Q} p(y_{j,k} | d_k = i, S_k = m, S_{k-1} = m', a_{j,k} = l_t, z_k = z) p(a_{j,k} = l_t | \mathbf{y_1}, \mathbf{y_3}) p(z_{j,k} = z)$$
(5.24)

Jamming NSI, Fading SI

For the case where the decoder has fading side information, but no jamming side information, calculation of branch transition probabilities uses the jamming estimate calculated by the previous MAP decoder.

$$p(y_{j,k}, a_{j,k} = a | d_k = i, S_k = m, S_{k-1} = m')$$

$$= \sum_{z=0}^{1} p(y_{j,k} | d_k = i, S_k = m, S_{k-1} = m', a_{j,k} = a, z_{j,k} = z) p(a_{j,k} = a) p(z_{j,k} = z) \quad (5.25)$$

$$\approx \sum_{z=0}^{1} p(y_{j,k} | d_k = i, S_k = m, S_{k-1} = m', a_{j,k} = a, z_{j,k} = z) p(a_{j,k} = a) p(z_{j,k} = z | \mathbf{y_1}, \mathbf{y_3}) \quad (5.26)$$

Jamming NSI, Fading NSI

Finally, for the case where neither jamming nor fading side information is available, both the jamming and fading estimates calculated by the previous MAP decoder are used in branch transition probability computations.

$$p(y_{j,k} | d_k = i, S_k = m, S_{k-1} = m')$$

$$= \sum_{t=1}^{Q} \sum_{z=0}^{1} p(y_{j,k} | d_k = i, S_k = m, S_{k-1} = m', a_{j,k} = l_t, z_{j,k} = z) p(a_{j,k} = l_t) p(z_{j,k} = z) \quad (5.27)$$

$$\approx \sum_{t=1}^{Q} \sum_{z=0}^{1} p(y_{j,k} | d_k = i, S_k = m, S_{k-1} = m', a_{j,k} = l_t, z_{j,k} = z) \cdot p(a_{j,k} = l_t | \mathbf{y_1}, \mathbf{y_3}) p(z_{j,k} = z | \mathbf{y_1}, \mathbf{y_3}) \quad (5.28)$$

Note that in the IID (one bit per hop) case, state estimates were not computed. This would be impractical, since estimates would be based on only one bit of information. With memory, there are more bits of information, thus allowing for more reliable estimates.

5.4 Simulation Results

In this section, the simulation results are presented. The component encoders are rate $\frac{1}{2}$ recursive systematic convolutional encoders with memory 4 and octal generators $(37, 21)$. The packet size is 1760 information bits and the number of decoder iterations is 5. A helical interleaver [27] is used to guarantee trellis termination. The SNR of the full-band thermal noise, E_b/N_0, is set to 20 dB. Cases with memory are simulated using 160 bits per hop (BPH). The number of quantization levels, Q, for the fading amplitudes is either 4 or 8.

Figure 5.1 shows the simulation results for the coherent reception of turbo codes in frequency-hop spread spectrum with Rayleigh fading and partial-band interference. The plots show the minimum E_b/N_J required to achieve a packet error rate of 10^{-3} for different values of ρ. Note that "F NSI, J SI (Q=4)" translates to the case where there is no fading side information, there is jamming side information, and the fading amplitudes were quantized to 4 levels for the estimation procedure.

First, consider the case of one bit per hop. At $\rho = 1$, all hops are jammed, so the decoder which is assumed to know ρ essentially has jamming side information. Thus, it is not surprising that the performance curves for the F NSI, J NSI and F NSI, J SI cases meet at $\rho = 1$. Similar results holds for the F SI, J NSI and F SI, J SI cases. Note that at $\rho = 1$, the gain of having fading side information is about 1 dB.

Next, consider the results for the IID case when $\rho < 1$. While it is expected that the F SI, J SI case outperforms the F NSI, J NSI case, it is interesting to compare the importance of having either fading side information (i.e. F SI, J NSI) or jamming

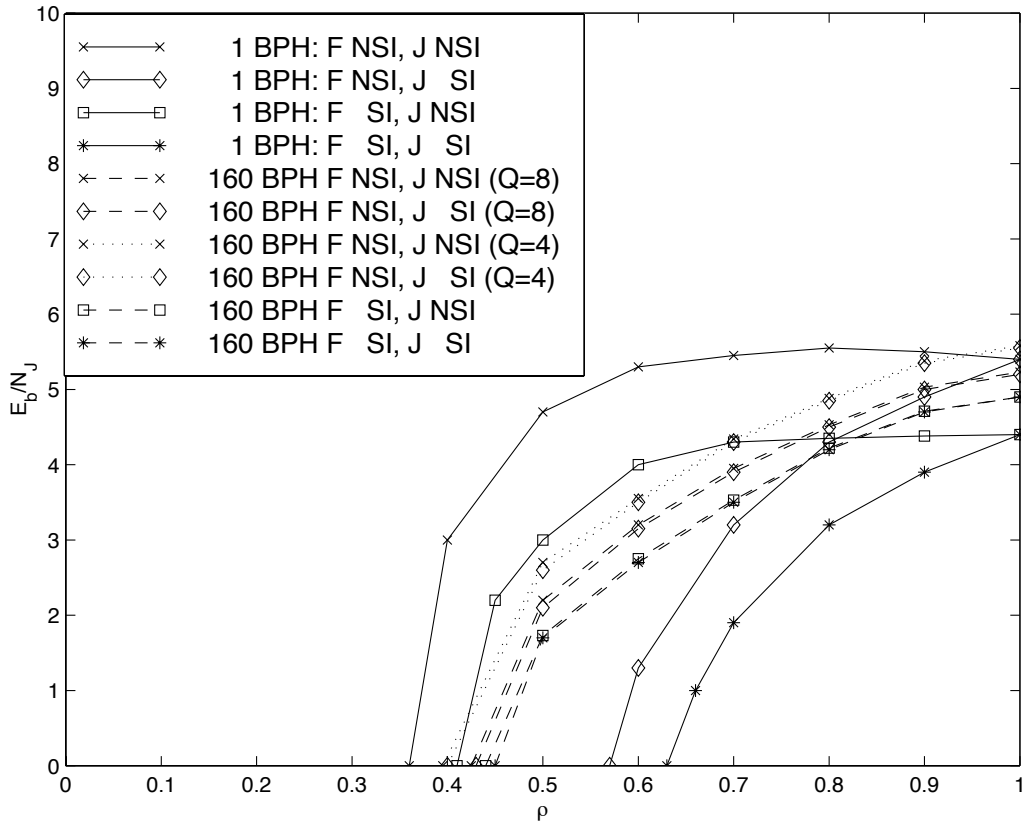

Figure 5.1: Performance of Turbo Codes in Coherent FH-SS with Rayleigh Fading and Partial-Band Interference

side information (i.e. F NSI, J SI) for each value of ρ. For ρ close to 1, the decoder essentially has jamming side information, so the F SI, J NSI outperforms the F NSI, J SI case in this region. Alternatively for decreasing values of ρ, $\frac{N_J}{2\rho}$ increases in magnitude. Thus, for small values of ρ, the effects of jamming begin to dominate the effects of fading. As a result, jamming side information is more important than fading side information for low values of ρ. Because, fading side information is more important for high values of ρ and jamming side information is more important for low values of ρ, the F NSI, J SI and F SI, J NSI curves should cross at some intermediate value of ρ. According to Figure 5.1, this occurs at approximately $\rho = .8$.

Next consider the case of 160 bits per hop. At $\rho = 1$, the F SI, J SI case for 160 bits per hop does not meet the F SI, J SI case for 1 bit per hop, a phenomena which occurred for FH-SS with partial-band interference without fading. In these simulations, the fade amplitude was assumed to be constant over the duration of each hop. For the case with 160 bits per hop, the duration of each fade is longer and consequently, the coherence time is larger. Seen in the previous chapter, this results in performance degradations. The loss of slow fading for 160 bits per hop is about 0.5 dB at $\rho = 1$ with respect to the IID case (for fading side information cases).

Next, consider the memory cases with $\rho < 1$. Due to reliable jamming state estimates, the difference between the performances of cases with and without jamming side information is virtually negligible. In addition, note that the calculation of quantized fading estimates yields promising results. The loss with respect to cases with fading side information is approximately 0.4 dB when there are eight quantized levels and 0.8 dB when there are four quantized levels.

Figure 5.2 shows the simulation results for the noncoherent reception of turbo codes in frequency-hop spread spectrum with Rayleigh fading and partial-band interference. The results are similar in form to the results yielded by coherent detection. For the IID case, there is a threshold at which fading side information becomes more

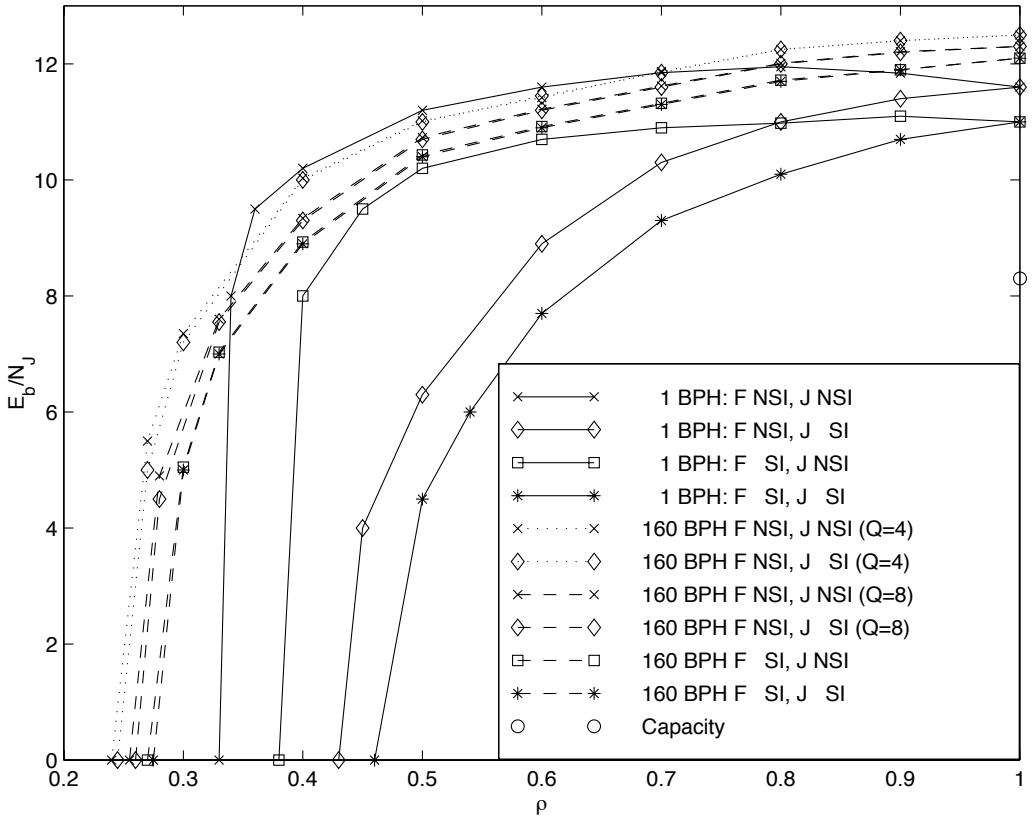

Figure 5.2: Performance of Turbo Codes in Noncoherent FH-SS with Rayleigh Fading and Partial-Band Interference

important than jamming side information. Similar to the case of coherent detection, this value is about $\rho = .8$. For the memory case, jamming state estimates can be accurately computed, explaining the small performance differences between cases with and without jamming side information. Finally, the the effective usage of quantized fade estimates is demonstrated. The loss with respect to the F SI case is about 0.5

dB for $Q = 8$ and 1.0 dB for $Q = 4$.

5.5 Conclusions

In this chapter, the design and performance of turbo codes for frequency-hop spread spectrum with partial-band interference and Rayleigh fading was investigated. For cases with memory, the jammer and fading amplitude was assumed to remain constant over the entire hop. In this case, channel estimation for both the fading level and jammer state was performed. Finally, simulation results were discussed. In the next chapter, we move away from frequency-hop spread spectrum systems and consider a more general type of channel, the Gilbert-Elliot burst channel.

CHAPTER 6

Turbo Codes for Burst Channels

6.1 Introduction

In previous chapters, the performance of iterative decoding and estimation were investigated in specific channels with memory. In this chapter, a more general channel with memory, the Gilbert-Elliot burst channel, is considered. The Gilbert-Elliot channel is a two state hidden Markov model where one state represents a bad channel state which typically has high error probabilities and the other state represents a good channel state which has low error probabilities. For the burst channel model, turbo code calculations require knowledge of the hidden Markov state. As in previous chapters, the approach is to estimates unknown channel state parameters and use these in the turbo decoder. This chapter is organized as follows. In Section 6.2, the system model is described. The iterative estimation and decoding scheme is presented in Section 6.3. The simulation results are described in Section 6.4. Finally, in Section 6.5, a brief summary of this work is presented.

6.2 System Model

6.2.1 Transmitter

The turbo encoder is formed by concatenating the constituent codes in parallel and then separating the codes by an interleaver [27]. The encoder takes as input the data sequence $d_k \epsilon \{0, 1\}$ and then produces three streams: the information bits d_k, the parity bits $p_{1,k}$ of the first component encoder with input d_k, and the parity bits $p_{2,k}$ of the second component encoder with interleaved d_k as input. BPSK modulation is considered with coherent demodulation.

6.2.2 Gilbert-Elliot Channel

The Gilbert-Elliot channel is a two state hidden Markov model (HMM) where one state represents a bad state which typically has high error probabilities and the other state is a good state which generally has low error probabilities. This model is shown below in Figure 6.1, where at time k, $z_k = 1$ represents the bad state and $z_k = 0$ represents the good state. The probability of moving from state $z_k = i$ to $z_{k+1} = j$ is denoted by p_{ij}.

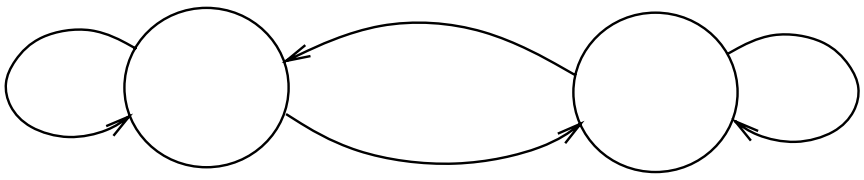

Figure 6.1: Hidden Markov Model of the Gilbert-Elliot Channel

If transmission occurs over the good state at time k, the noise is assumed to be additive white Gaussian noise (AWGN) with power spectral density $N_0/2$ where N_0 typically has low magnitude. Similarly, for transmission over the bad state, the noise is white Gaussian with power spectral density $N_1/2$ where $N_1 > N_0$. Let $(y_{1,k}, y_{2,k}, y_{3,k})$ be the channel outputs and let $(c_{1,k}, c_{2,k}, c_{3,k}) = ((-1)^{d_k}, (-1)^{p_{1,k}}, (-1)^{p_{2,k}})$. Then assuming coherent detection, the model for the channel outputs is

$$y_{i,k} = \sqrt{E}\, c_{i,k} + \eta^z_{i,k}, \qquad i = 0, 1, 2; \quad k = 1, ..., N; \quad z = 0, 1$$

where $\eta^z_{i,k} \sim N(0, N_z/2)$ assumes a zero mean Gaussian density with conditional variance N_z which depends on the channel state, $z_{i,k} = z$.

Note that this channel model is similar to the channel model of a FH-SS system with an on-off jammer. If the jammer is on, the signal is corrupted by jamming interference and thermal noise (bad state). If the jammer is off, the signal is corrupted by only thermal noise (good state). If the hopping rate is much faster than the rate at which the jammer changes its signal structure, then it is likely that most of the bits within a hop will be either jammed or unjammed. This could be modeled in the figure above by high values of p_{11} and p_{00}. Alternatively, the states of the channel model could represent whether an entire hop has been jammed or not. If ρ is the probability that a hop is jammed, then the transition probabilities could be set such that the steady state probability $\lim_{k\to\infty} p(z_k = 1) = \rho$.

6.3 Turbo Decoder for the Gilbert-Elliot Channel

The turbo decoding algorithm is dependent on what information is available to the turbo decoder. This chapter considers three cases: known channel state; unknown channel state but known HMM transition probabilities p_{ij}; and unknown channel state and unknown p_{ij}.

6.3.1 Known Channel State

If the state, $z_{i,k}$, is known, then the modification to the turbo decoder is straightforward. The decoder can simply use the relevant noise variance to calculate the branch transition probabilities. Thus, (2.6) can be calculated using

$$p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m', z_{j,k} = z) = \frac{1}{\sqrt{\pi N_z}} e^{-\frac{1}{N_z}(y_{j,k} - \sqrt{E}\, c_{j,k})^2} \tag{6.1}$$

where d_k and the trellis state transition $S_k = m$ and $S_{k-1} = m'$ determine the associated coded bit $c_{j,k}$.

6.3.2 Unknown Channel State, Known Transition Probabilities

If the channel state is unknown, but the transition probabilities are known, then (2.6) can be calculated by invoking total probability with respect to the channel state.

$$p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m')$$

$$
\begin{aligned}
&= p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m', z_{j,k} = 1) \cdot p(z_{j,k} = 1) + \\
&\quad p(y_{j,k}|d_k = i, S_k = m, S_{k-1} = m', z_{j,k} = 0) \cdot p(z_{j,k} = 0) \qquad (6.2) \\
&= \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{1}{2\sigma_1^2}(y_{j,k} - \sqrt{E}\, c_{i,k})^2} \cdot p(z_{j,k} = 1) + \\
&\quad \frac{1}{\sqrt{2\pi}\sigma_0} e^{-\frac{1}{2\sigma_0^2}(y_{j,k} - \sqrt{E}\, c_{j,k})^2} \cdot p(z_{j,k} = 0) \qquad (6.3)
\end{aligned}
$$

Note that $p(z_{j,k} = z)$ is not known. One possibility is to use the steady state probability $\lim_{k\to\infty} p(z_{j,k} = z)$. This can be obtained by solving $\mathbf{v} = \mathbf{v}P$, $\sum_l v_l = 1$, and setting $p(z_{j,k} = l) = v_l$.

Another possibility is to estimate the probability of being in each state given the received sequence and knowing the HMM transition probabilities. For $k = 1$,

$$
p(z_{i,1} = 0|y_{i,1}) = \frac{p(y_{i,1}|z_{i,1} = 0)\, p(z_{i,1} = 0)}{p(y_{i,1})} \qquad (6.4)
$$

where $p(z_{i,1} = 0)$ is set to the steady state probability v_0. For $k \geq 2$,

$$
\begin{aligned}
p(z_{i,k} = 0|y_{i,1}, ..., y_{i,k}) &= \frac{p(y_{i,k}|z_{i,k} = 0, y_{i,1}, .., y_{i,k-1})\, p(z_{i,k} = 0, y_{i,1}, .., y_{i,k-1})}{p(y_{i,1}, .., y_{i,k})} \qquad (6.5) \\
&\approx \frac{p(y_{i,k}|z_{i,k} = 0)\, p(z_{i,k} = 0, y_{i,1}, .., y_{i,k-1})}{p(y_{i,1}, .., y_{i,k})} \qquad (6.6)
\end{aligned}
$$

where (6.6) is approximate because encoder memory yields correlated channel outputs $\{y_{i,j}\}_{j=1}^{k}$. Furthermore, (6.6) is computed using

$$
\begin{aligned}
p(z_{i,k} = 0, y_{i,1}, ..., y_{i,k-1}) &= \sum_{z=0}^{1} p(z_{i,k} = 0, z_{i,k-1} = z, y_{i,1}, ..., y_{i,k-1}) \qquad (6.7) \\
&= \sum_{z=0}^{1} p(y_{i,k-1}|z_{i,k} = 0, z_{i,k-1} = z, y_{i,1}, ..., y_{i,k-2}) \cdot
\end{aligned}
$$

$$p(z_{i,k}=0, z_{i,k-1}=z, y_{i,1}, ..., y_{i,k-2}) \tag{6.8}$$

$$= \sum_{z=0}^{1} p(y_{i,k-1}|z_{i,k-1}=z) \cdot$$

$$p(z_{i,k}=0, z_{i,k-1}=z, y_{i,1}, ..., y_{i,k-2}) \tag{6.9}$$

and (6.9) is computed using

$$p(z_{i,k}=a, z_{i,k-1}=b, y_{i,1}, ..., y_{i,k-2})$$

$$= p(y_{i,1}, ..., y_{i,k-2}|z_{i,k}=a, z_{i,k-1}=b)\, p(z_{i,k}=a, z_{i,k-1}=b) \tag{6.10}$$

$$= p(y_{i,1}, ..., y_{i,k-2}|z_{i,k-1}=b)\, p(z_{i,k}=a|z_{i,k-1}=b)\, p(z_{i,k-1}=b) \tag{6.11}$$

$$= \frac{p(z_{i,k-1}=b, y_{i,1}, ..., y_{i,k-2})}{p(z_{i,k-1}=b)}\, p(z_{i,k}=a|z_{i,k-1}=b)\, p(z_{i,k-1}=b) \tag{6.12}$$

$$= p(z_{i,k-1}=b, y_{i,1}, ..., y_{i,k-2})\, p_{ba}. \tag{6.13}$$

Combining (6.9) and (6.13),

$$p(z_{i,k}=0, y_{i,1}, ..., y_{i,k-1}) = \sum_{z=0}^{1} p(y_{i,k-1}|z_{i,k-1}=z)\, p(z_{i,k-1}=z, y_{i,1}, ..., y_{i,k-2})\, p_{z0}. \tag{6.14}$$

Note that due to the recursive nature of (6.14), (6.6) can be computed efficiently.

6.3.3 Unknown Channel State, Unknown Transition Probabilities

If the HMM transition probabilities are unknown, it is necessary to estimate the transition probabilities of the chain. Once the transition probabilities have been adap-

tively estimated, the channel states can be estimated and then used by the turbo decoder, as seen above. Thus the problem is to find the HMM model which maximizes the probabilities of the observation sequence. The Baum-Welch re-estimation procedure yields an ML estimate of the HMM which is a locally optimal solution.

Before describing the Baum-Welch algorithm, it is necessary to detail the procedure of calculating the observation sequence probability given the HMM model [34].

Calculation of Observation Sequence Probability

Let the channel outputs be denoted as $\mathbf{y} = \{y_i\}_{i=1}^{N}$ and the state sequence corresponding to $\mathbf{y}$ be denoted as $\mathbf{z} = \{z_i\}_{i=1}^{N}$. Furthermore, let $\pi = \{\pi_i\}_{i=0}^{1}$ for $\pi_i = P(z_1 = i)$ represent the initial state distribution, let $A = \{p_{ij}\}$ for $i = 0, 1$ and $j = 0, 1$ be the set of state transition probabilities, and let $B = \{b_j(k)\}$ represent the probability distributions of the observation symbol where $b_j(k) = p(y_k|z_k = j)$. If the hidden Markov model is denoted by $\Lambda = (\pi, A, B)$, then the Baum-Welch procedure attempts to maximize $p_\Lambda(\mathbf{y})$, the probability of the observation sequence, $\mathbf{y}$, given the HMM model, Λ. But first, we need to know how to compute $p_\Lambda(\mathbf{y})$. Let $p_\Lambda(x)$ represent the probability density function of some random variable x given the HMM model, Λ.

$$
\begin{aligned}
p_\Lambda(\mathbf{y}) &= \sum_{z_1,..,z_N} p_\Lambda(\mathbf{y}, \mathbf{z}) && (6.15)\\
&= \sum_{z_1,..,z_N} p_\Lambda(\mathbf{y}|\mathbf{z}) \cdot p_\Lambda(\mathbf{z}) && (6.16)\\
&= \sum_{z_1,..,z_N} [\prod_{i=1}^{N} p_\Lambda(y_i|z_i)] \cdot [\pi_1 p_{z_1 z_2} p_{z_2 z_3} \cdots p_{z_{N-1} z_N}] && (6.17)
\end{aligned}
$$

$$= \sum_{z_1,..,z_N} [\prod_{i=1}^{N} b_{z_i}(i)] \cdot [\pi_1 p_{z_1 z_2} p_{z_2 z_3} \cdots p_{z_{N-1} z_N}] \tag{6.18}$$

where it is assumed that the observations are independent, $p(y_1, ..., y_k) = \prod_{i=1}^{k} p(y_i)$. Note that for encoders with memory (e.g. convolutional codes), this assumption is invalid. For such cases, (6.17) and (6.18) become approximations.

If there are m states, this direct computation would take on the order of $2N \cdot m^N$ calculations. This is computationally intensive as the complexity grows exponentially in N. As a result, a recursive procedure was established. If $g_i(j) = p_\Lambda(y_1, y_2, ..., y_i, z_i = j)$ represents the probability of the partial observation sequence, then the forward procedure can be computed as shown below. For the calculations, let $\mathbf{y_1^k} = \{y_i\}_{i=1}^{k}$ and assume that the observations are independent, $p(\mathbf{y_1^k}) = \prod_{i=1}^{k} p(y_i)$.

1. Initialization

$$g_1(i) = \pi_i b_i(1) \qquad 1 \leq i \leq m \tag{6.19}$$

2. Induction

$$\begin{aligned}
g_{k+1}(j) &= p_\Lambda(\mathbf{y_1^{k+1}}, z_{k+1} = j) && (6.20)\\
&= \sum_i p_\Lambda(\mathbf{y_1^k}, y_{k+1}, z_{k+1} = j, z_k = i) && (6.21)\\
&= \sum_i p_\Lambda(\mathbf{y_1^k} | y_{k+1}, z_{k+1} = j, z_k = i)\, p_\Lambda(y_{k+1}, z_{k+1} = j, z_k = i) && (6.22)\\
&= \sum_i p_\Lambda(\mathbf{y_1^k} | z_k = i)\, p_\Lambda(y_{k+1} | z_{k+1} = j, z_k = i) \cdot && (6.23)\\
&\qquad p_\Lambda(z_{k+1} = j | z_k = i)\, p_\Lambda(z_k = i)
\end{aligned}$$

$$= \sum_i p_\Lambda(\mathbf{y_1^k}, z_k = i)\, p_\Lambda(y_{k+1}|z_{k+1} = j)\, p_{ij} \tag{6.24}$$

$$= [\sum_{i=1}^{m} g_k(i) p_{ij}] b_j(k+1) \qquad 1 \le k \le N-1, \quad 1 \le j \le m \tag{6.25}$$

3. Termination

$$p_\Lambda(\mathbf{y}) = \sum_{i=1}^{N} p_\Lambda(\mathbf{y}, z_N = i) \tag{6.26}$$

$$= \sum_{i=1}^{N} g_N(i) \tag{6.27}$$

To summarize the steps, the first step initializes the forward probabilities by the initial state probability and probability of the first observation. The second step essentially considers all states from the previous time step and weights them by the associated state transition and observation probability. The termination step simply computes the desired $p_\Lambda(\mathbf{y})$ by summing over the states of the forward variable. For $h_i(j) = p_\Lambda(y_{i+1}, y_{i+2}, ..., y_N|z_i = j)$, the backward procedure takes a similar form [34].

1. Initialization

$$h_N(i) = 1 \qquad 1 \le i \le m \tag{6.28}$$

2. Induction

$$h_k(i) = \textstyle\sum_{j=1}^{m} h_{k+1}(j) p_{ij} b_j(k+1) \qquad 1 \le k \le N-1,\ 1 \le j \le m \tag{6.29}$$

We now have the tools to describe the Baum-Welch procedure.

Baum-Welch Re-estimation Procedure

If $\xi_k(i,j)$ is defined to be the probability of being in state i at time k and in state j at time $k+1$ given the model and observation sequence, then for the observation sequence, $\mathbf{y_1^N} = y_1, y_2, ..., , y_N$,

$$\begin{aligned}
\xi_k(i,j) &= p_\Lambda(z_k = i, z_{k+1} = j | \mathbf{y_1^N}) && (6.30)\\
&= \frac{p_\Lambda(z_k = i, z_{k+1} = j, \mathbf{y_1^N})}{p_\Lambda(\mathbf{y_1^N})} && (6.31)\\
&= \frac{p_\Lambda(z_k = i, z_{k+1} = j, \mathbf{y_1^N})}{\sum_i \sum_j p_\Lambda(z_k = i, z_{k+1} = j, \mathbf{y_1^N})} && (6.32)
\end{aligned}$$

where for independent observations, $p(\mathbf{y_1^k}) = \prod_{i=1}^{k} p(y_i)$,

$$\begin{aligned}
&p_\Lambda(z_k = i, z_{k+1} = j, \mathbf{y_1^N})\\
&\quad = p_\Lambda(\mathbf{y_1^k} | z_k = i, z_{k+1} = j, \mathbf{y_{k+1}^N})\, p_\Lambda(z_k = i, z_{k+1} = j, \mathbf{y_{k+1}^N}) && (6.33)\\
&\quad = p_\Lambda(\mathbf{y_1^k} | z_k = i)\, p_\Lambda(y_{k+1} | z_k = i, z_{k+1} = j, \mathbf{y_{k+2}^N})\, p_\Lambda(z_k = i, z_{k+1} = j, \mathbf{y_{k+2}^N}) && (6.34)\\
&\quad = g_k(i)\, p_\Lambda(y_{k+1} | z_{k+1} = j)\, p_\Lambda(\mathbf{y_{k+2}^N} | z_k = i, z_{k+1} = j)\, p_\Lambda(z_{k+1} = j | z_k = i) && (6.35)\\
&\quad = g_k(i)\, b_j(k+1)\, p_\Lambda(\mathbf{y_{k+2}^N} | z_{k+1} = j)\, p_{ij} && (6.36)\\
&\quad = g_k(i)\, b_j(k+1)\, h_{k+1}(j)\, p_{ij}. && (6.37)
\end{aligned}$$

Combining (6.32) and (6.37),

$$\xi_k(i,j) = \frac{g_k(i) p_{ij} b_j(k+1) h_{k+1}(j)}{\sum_{i=1}^{m} \sum_{j=1}^{m} g_k(i) p_{ij} b_j(k+1) h_{k+1}(j)} \qquad (6.38)$$

If $\gamma_k(i)$ is defined as the probability of being in state i at time k given the observation sequence and model, $\gamma_k(i)$ can be related to $\xi_k(i,j)$ by

$$\begin{aligned} \gamma_k(i) &= p_\Lambda(z_k = i|\mathbf{y_1^N}) && (6.39) \\ &= \sum_{j=1}^{m} \xi_k(i,j) && (6.40) \end{aligned}$$

If $\gamma_k(i)$ is summed over all k, the expected number of transitions made from state i can be determined. Similarly, summing $\xi_k(i,j)$ over all k gives the expected number of transitions from state i to j. Using these results, a set of re-estimation formulas can be made for $\overline{p}_{ij}$ and $\overline{b}_j(k)$.

$$\begin{aligned} \overline{p}_{ij} &= \frac{\text{expected number of transitions from state i to state j}}{\text{expected number of transitions from state i}} \\ &= \frac{\sum_{k=1}^{N-1} \xi_k(i,j)}{\sum_{k=1}^{N-1} \gamma_k(i)} && (6.41) \\ & && (6.42) \end{aligned}$$

$$\begin{aligned} \overline{b}_j(n) &= \frac{\text{expected number of times in state j and observing } y_n = y}{\text{expected number of times in state j}} \\ &= \frac{\sum_{k=1, y_n=y}^{N} \gamma_k(j)}{\sum_{k=1}^{N} \gamma_k(j)} && (6.43) \end{aligned}$$

Based on the above procedure, if $\overline{\Lambda}$ is iteratively used in place of Λ and the re-estimation calculations are repeated, the probability of $\mathbf{y}$ given the HMM model can

be improved until some limiting point is reached. The final result of this procedure is an ML estimate of the HMM. However, it should be pointed out that the algorithm only leads to a local maxima, where many local maxima might exist.

Note that at each iteration, the Baum-Welch procedure inherently calculates $p(z_i = j)$ through $\gamma_k(i)$. Hence, after the Baum-Welch algorithm converges, it is not necessary to recalculate the state estimates as we did before.

6.4 Simulation Results

In this section, simulation results are presented for the Gilbert-Elliot channel. For all simulations, the two component encoders were rate $\frac{1}{2}$ convolutional encoders with memory 4 and octal generators $(37, 21)$. Each block had 1920 information bits and a total of 8 turbo decoding iterations were used. In addition, the SNR of the good state, E_b/N_0, was set to 4 dB for all realizations.

Figure 6.2 shows the BER performance of the system when the channel state is known for different values of p_{ij}. This set of performance curves serve as a reference from which the following cases can be based. Note that the average received SNR is not the same for each of the systems in Figure 6.2. The average received SNR is $10 \log \frac{E_b}{v_0 N_0 + v_1 N_1}$ where v_i represents the steady state probability of being in state i. For the first system ($p_{00} = 0$, $p_{11} = 1.$), $v_0 = 0$ and $v_1 = 1$. For the last system ($p_{00} = .8$, $p_{11} = .2$), $v_0 = .8$ and $v_1 = .2$. For the middle three systems, $v_0 = v_1 = .5$. If we assume that $N_1 > N_0$, then for a given N_0 and N_1, the first system has the lowest average SNR and the second system has the highest average SNR. The average

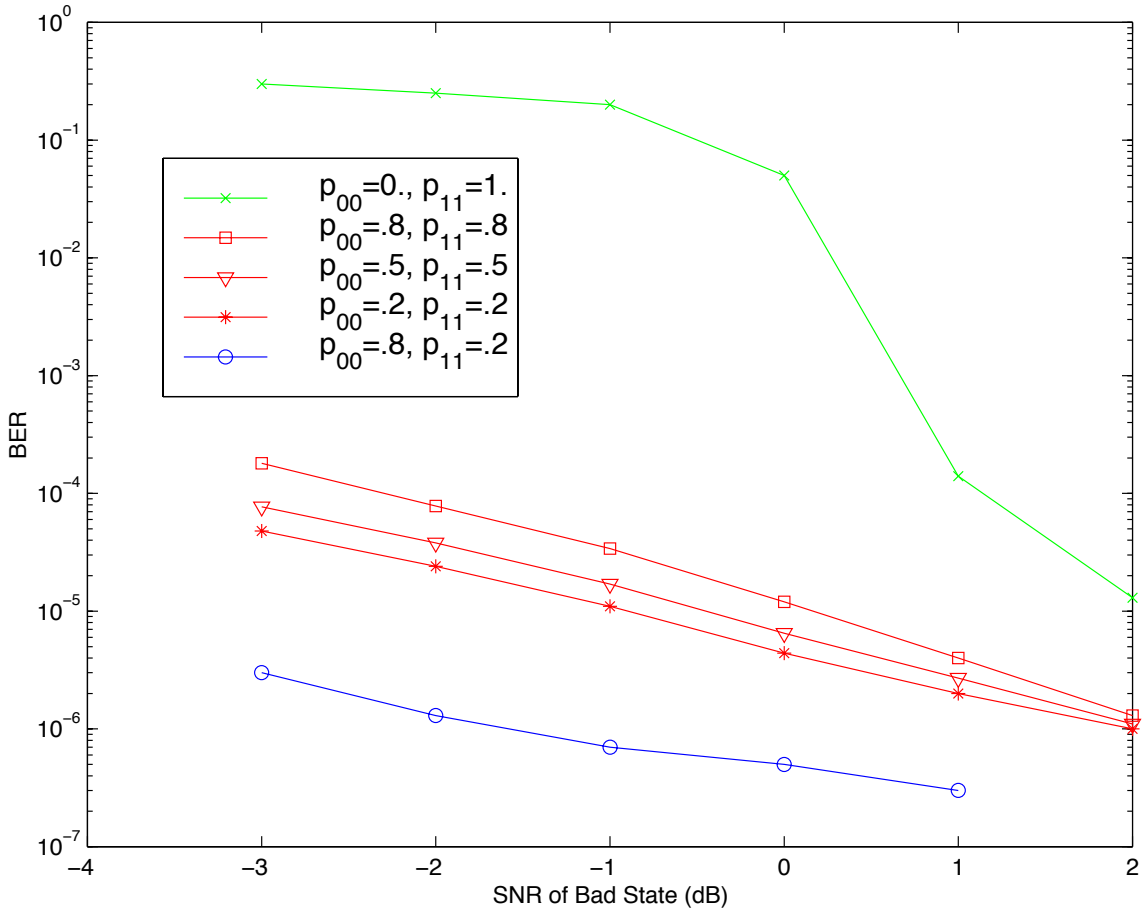

Figure 6.2: Performance of Turbo Codes in the Gilbert-Elliot Channel: Known Channel State

SNR of the middle three systems is the same and their average SNR is between those of the first and last systems. This explains the difference in performance between the three clusters of curves (grouped by v_0 and v_1). Among the curves with $v_0 = v_1 = .5$, as expected, performance degrades if the sequence of states is more bursty (i.e. higher values of p_{00} and p_{11}).

The simulation results for the case of known HMM transition probabilities but unknown state are shown in Figures 6.3 and 6.4. At low SNRs, the *a posteriori* state estimation method performs better than the steady state method as the decoder

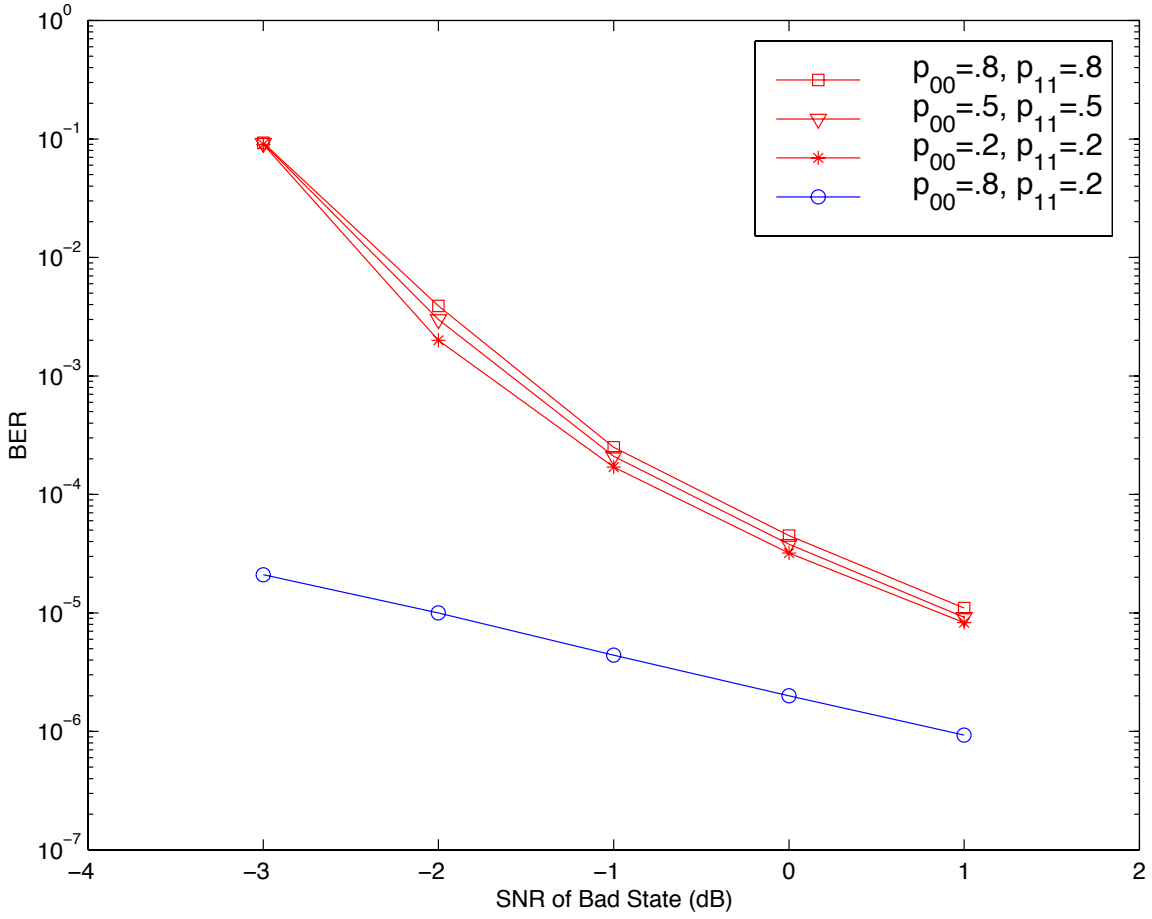

Figure 6.3: Performance of Turbo Codes in the Gilbert-Elliot Channel: Known p_{ij}, Steady State

successfully uses the information inherent in the memory of the channel. There is, however, still a large gap between these two cases and the known channel state case. With just one bit transmitted over each state, channel state estimates are far from reliable and this degrades decoding performance. As the SNR increases, the performance difference between the graphs of Figures 6.3 and 6.4 decreases as the channel estimation method has a more difficult time distinguishing between the two states. Note, however, that for high SNRS, the performance of both systems is comparable

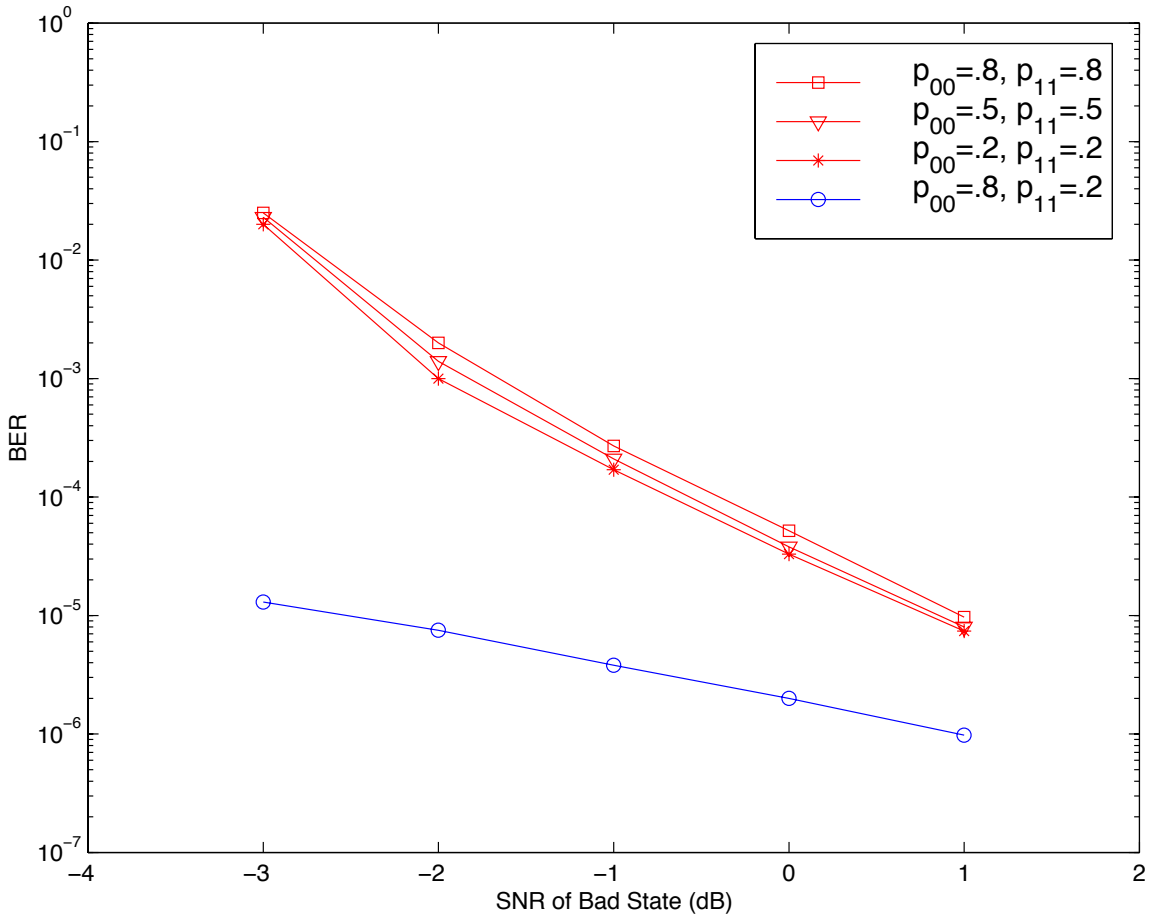

Figure 6.4: Performance of Turbo Codes in the Gilbert-Elliot Channel: Known p_{ij}, Channel Estimation

to that of the known channel state case. At high SNRs, the two state system essentially reduces to a one state, high SNR system. Thus, even though channel estimation

methods might lose accuracy at high SNRs, the performance is still good since precise state estimation is not necessary in high SNR regions.

The simulation results of the channel estimation scheme in Figure 6.4 showed that the benefit of the system which employed channel estimation was small since estimation was based on just one transmitted bit over each state. For practical purposes, this model is unrealistic because for reasonably high data rates, the channel normally does not exhibit large flunctuations from bit to bit. A more realistic model might have several bits transmitted over each state. Increasing the memory of the channel would lead to improved channel estimates. In Figure 6.5, the benefit of increased channel memory is demonstrated for $p_{00} = p_{11} = 0.8$.

Similar to previous chapters, increasing the memory reduces the effective block length of the code. This explains the loss in performance of the known state case when the channel model has 20 bits transmitted per state rather than 1 bit. The benefit of increasing channel memory is that channel estimation improves. At low SNRs, the channel estimation technique is more easily able to distinguish between the two hidden Markov states. Hence, relative to the steady state method, channel estimation yields the largest gains at low SNRs. At high SNRs, the channel estimation technique performs more and more poorly, even to the point where it yields worse performance than the steady state method. The dominating factor at high SNRs, however, is that the channel model reduces to a one state high SNR system. Thus, overall performance despite unreliable channel estimates is good at high SNRs. Note that while the channel estimation scheme may perform worse than the steady state method at high SNRs, the difference is performance is marginal. Hence, the channel

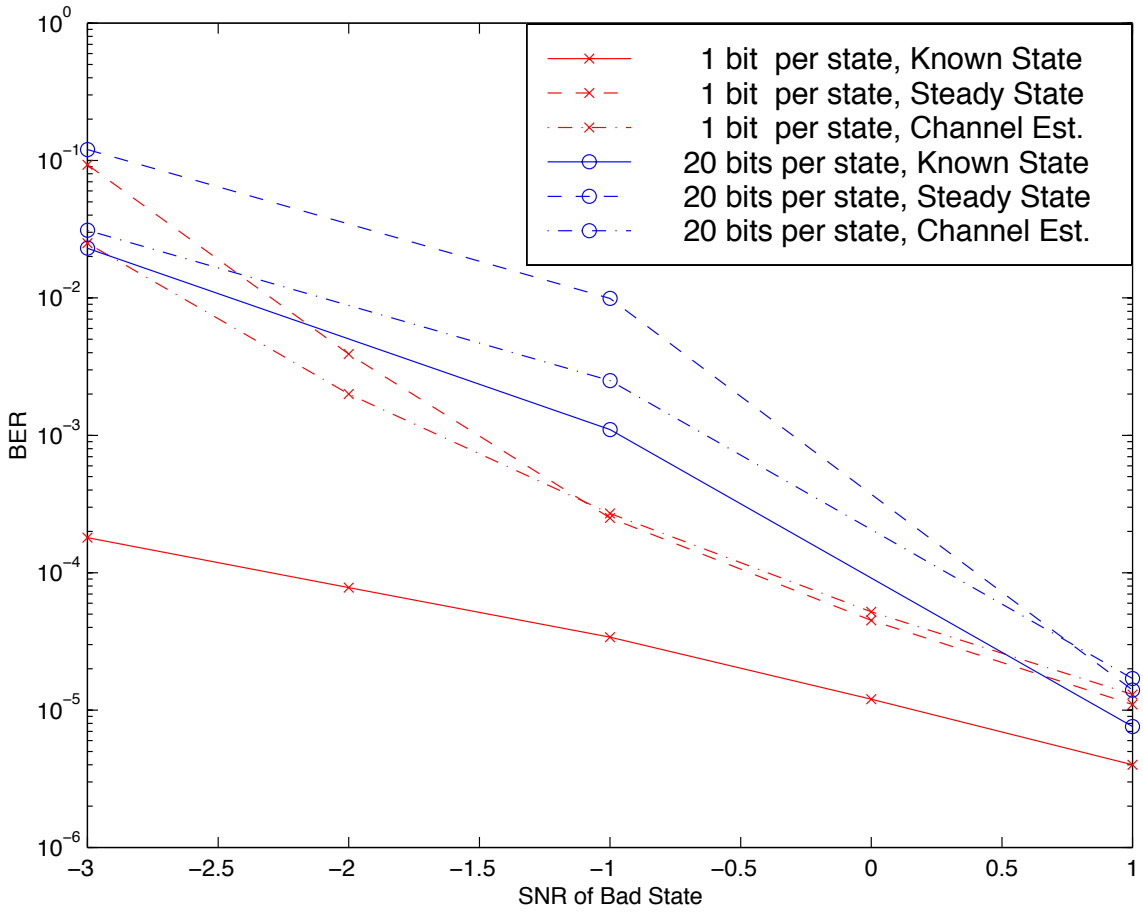

Figure 6.5: Performance of Turbo Codes in the Gilbert-Elliot Channel: Known p_{ij}, Channel Estimation, 20 Bits Transmitted Over Each State

estimation method is robust across all SNRs.

Figure 6.6 contains the simulation results for the case where neither channel state nor transition probability information is available to the decoder. Note that the performance is close to that of the known transition probability, *a posteriori* state estimation case. Comparing the plots of Figures 6.4 and 6.6, there is little difference

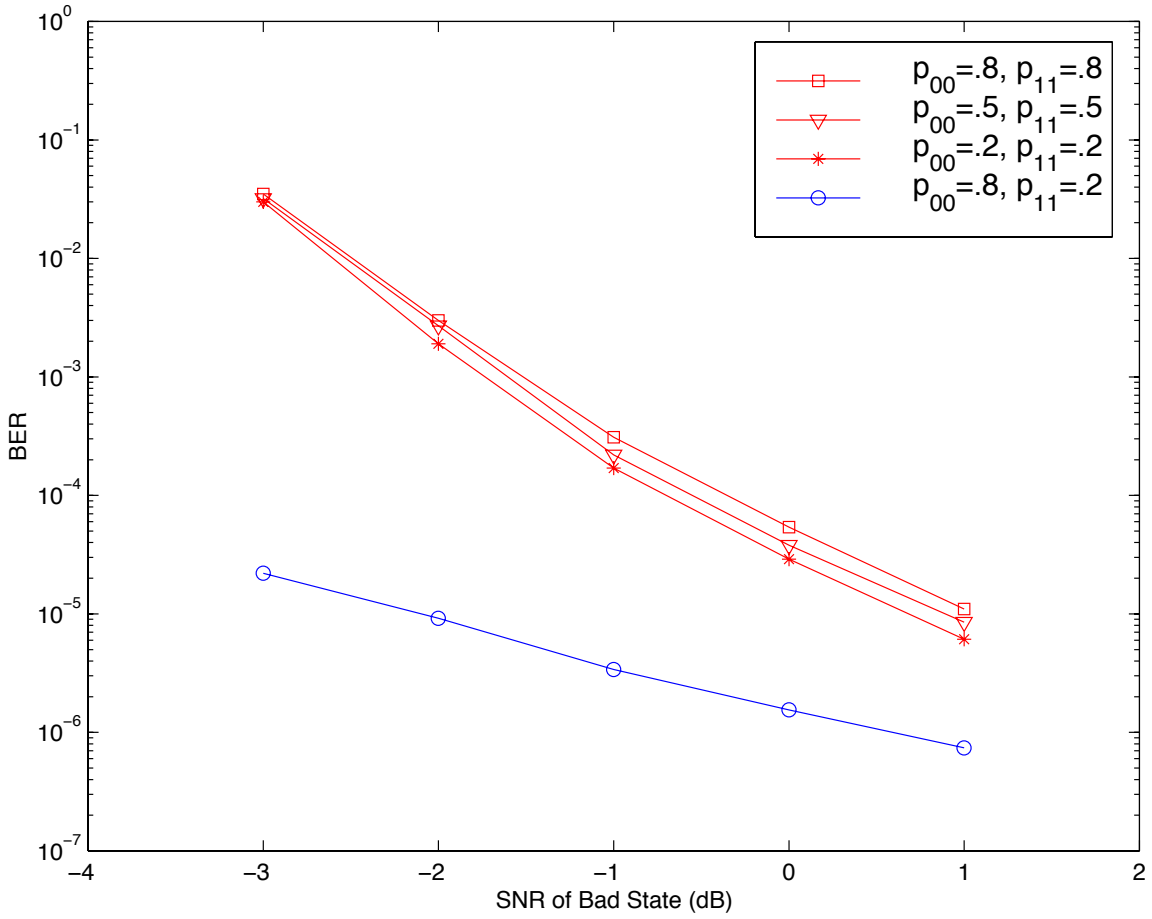

Figure 6.6: Performance of Turbo Codes in the Gilbert-Elliot Channel: Unknown State, Unknown p_{ij}

in performance particularly at low SNRs, because the Baum Welch algorithm is doing a good job at estimating channel states. Knowing the state does not give a major advantage at low SNRs since large magnitude noise makes tracking the information

bits difficult. At high SNRs, the Baum-Welch algorithm performs with less success, but the more dominating factor is that state estimation at high SNRs is less important.

6.5 Conclusion

The performance of turbo codes in the Gilbert-Elliot burst channel model was investigated with varying levels of side information. It was shown that for cases where the transition probabilities of the HMM are unknown, the performance is not seriously degraded if the Baum-Welch procedure is applied. In fact, the performance difference between cases where the HMM transition probabilities are known and unknown (but the channel state is unknown) is marginal. For the case where the channel state was unknown but the HMM transition probabilities are known, the gain of using *a posteriori* state estimates over use of steady state probabilities is marginal when just one bit has been transmitted over each state. The state estimates are inaccurate if based on just one observation. The benefit of increasing the channel memory to 20 bits transmitted over each HMM state is that channel state estimation is improved. However, increasing the channel memory can result in worse overall performance because the effective block length is reduced.

CHAPTER 7

Robustness of Turbo Decoding

In previous chapters, it was assumed that certain parameters of the channel were known, including the power spectral densities of the interference and the memory of the channel. Such channel knowledge is required not only for soft decision decoders, but also for the channel estimation schemes. When the channel is relatively static, estimation schemes can be utilized to estimate unknown channel parameters. However, such estimation schemes typically rely on some channel knowledge. For instance, the estimation technique of Chapter 3 which estimated the jamming state, z_k, required knowledge of the noise power spectral densities, $\frac{N_0}{2}$ and $\frac{N_J}{2\rho}$, but also the fractional jamming bandwidth, ρ, and the length of the channel memory. Mismatches in any of the assumed information can lead to severe degradation in overall performance. In this chapter, the robustness of turbo code performance is investigated when assumptions concerning channel knowledge are relaxed.

This chapter is organized as follows. In Section 7.1, the performance of turbo-coded systems are considered in the additive white Gaussian noise (AWGN) channel

with SNR mismatch. The Section 7.2, turbo codes will be considered in a FH-SS system with partial-band jamming and mismatched channel statistics. In Section 7.3, the work is summarized and conclusions are drawn.

7.1 SNR Mismatch in AWGN

As a baseline system, the performance of turbo codes in an additive white Gaussian noise (AWGN) channel with SNR mismatch is considered. Soft decision decoders typically require knowledge of the SNR. When the channel is relatively static (like the AWGN channel), there are many good SNR estimation techniques [28] which offer different levels of performance at the cost of complexity. However, such estimation schemes are sensitive to parameters such as the length of the transmitted packet. If turbo codes are insensitive to mismatched SNR which may arise from estimation errors, then they can be considered robust in the AWGN channel. In the event that turbo codes are determined to be sensitive to mismatched SNR, then an alternative scheme which is more robust to SNR mismatch needs to be considered.

First, consider the performance of turbo codes in AWGN with true power spectral density $N_0/2$ and mismatched statistics at the decoder. The turbo code in the simulation is a rate $1/3$ code with block length 1640. Figure 7.1 plots the BER performance of BPSK-modulated turbo codes for different true values of SNR versus several levels of SNR mismatch. For instance, the plot in the figure for SNR $= 3$ dB at an SNR offset of -2 dB shows the performance of turbo codes when the actual SNR is 3 dB, but the receiver decodes as if the SNR is 1 dB. For SNR < 0 dB, mismatches have

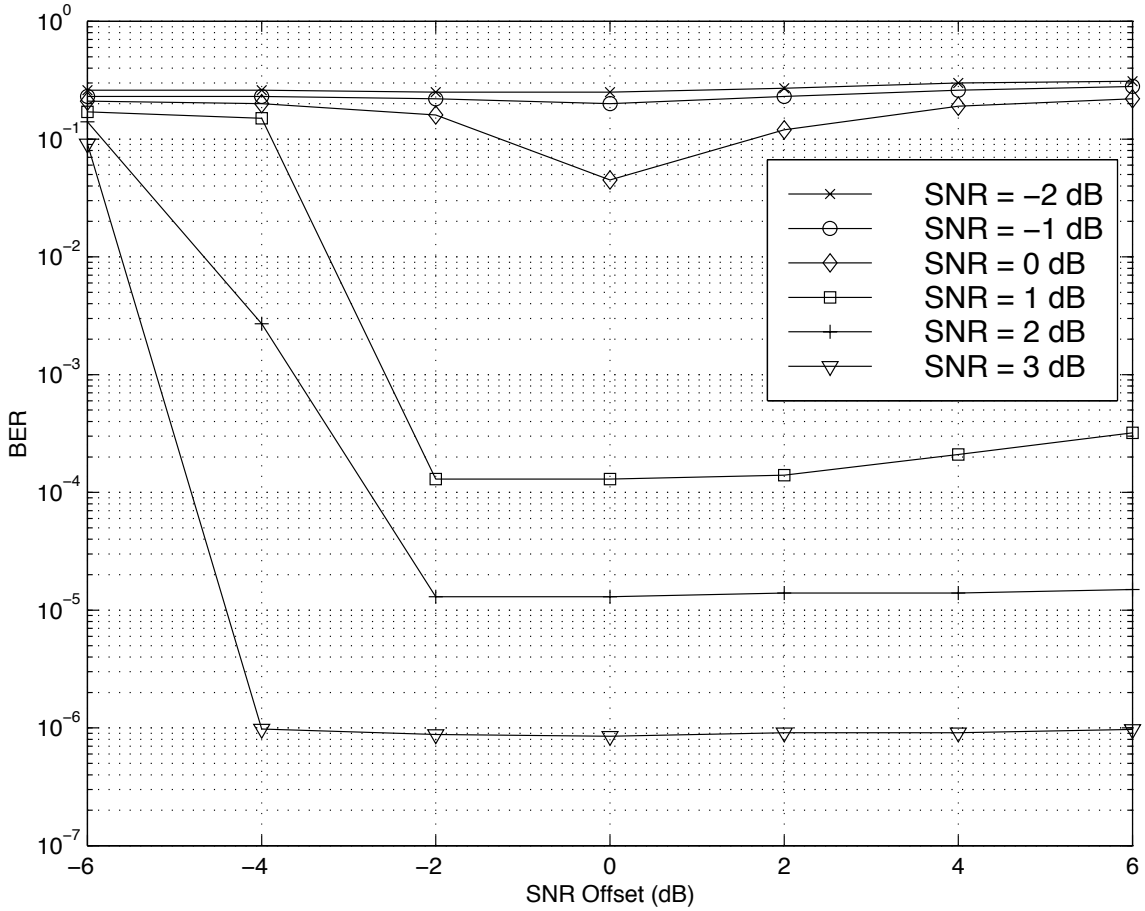

Figure 7.1: SNR Mismatch in Coherent AWGN

little effect since the performance is already very poor. For SNR ≥ 0 dB, the sensitivity to SNR offsets decreases as the SNR increases. For instance, the performance at SNR $= 2$ dB appears robust to offsets in the range of -2 dB to 6 dB. In comparison, the performance at SNR $= 3$ dB is robust to offsets from -4 dB to 6 dB. While the performance is virtually constant over these ranges, the number of iterations required to achieve such performance increases as the degree of mismatch increases. Note that the penalty of overestimating the SNR is much smaller than the penalty of underestimating the SNR. The reason for this is as follows. Overestimating the SNR implies that the estimated noise variance has smaller magnitude. Underestimating the SNR implies that the estimated noise variance has larger magnitude. This affects the turbo decoder which makes bit decisions based on the conditional Gaussian probability densities of the channel outputs. If the estimated noise variance is smaller than the actual noise variance, then the density will become sharper, placing greater differentiation between received bits. If the estimated noise variance is larger than the actual noise variance, then the resultant density will become more smooth, making it more difficult to distinguish between bits. This can be seen in Figure 7.2 for an exaggerated case. For $N_0/2 = 0.01$, the density values at $y_k = 1.00$ and $y_k = 1.22$ are 3.989 and 0.540, respectively. For $N_0/2 = 1.0$, the values at $y_k = 1.00$ and $y_k = 1.22$ are 0.399 and 0.391, respectively. The main point here is that if the variance is significantly overestimated (i.e. the SNR is significantly underestimated), the turbo decoder has a difficult time distinguishing between information bits because the density makes little distinction between the channel outputs.

One possible method of improving the robustness of the system to SNR mismatch

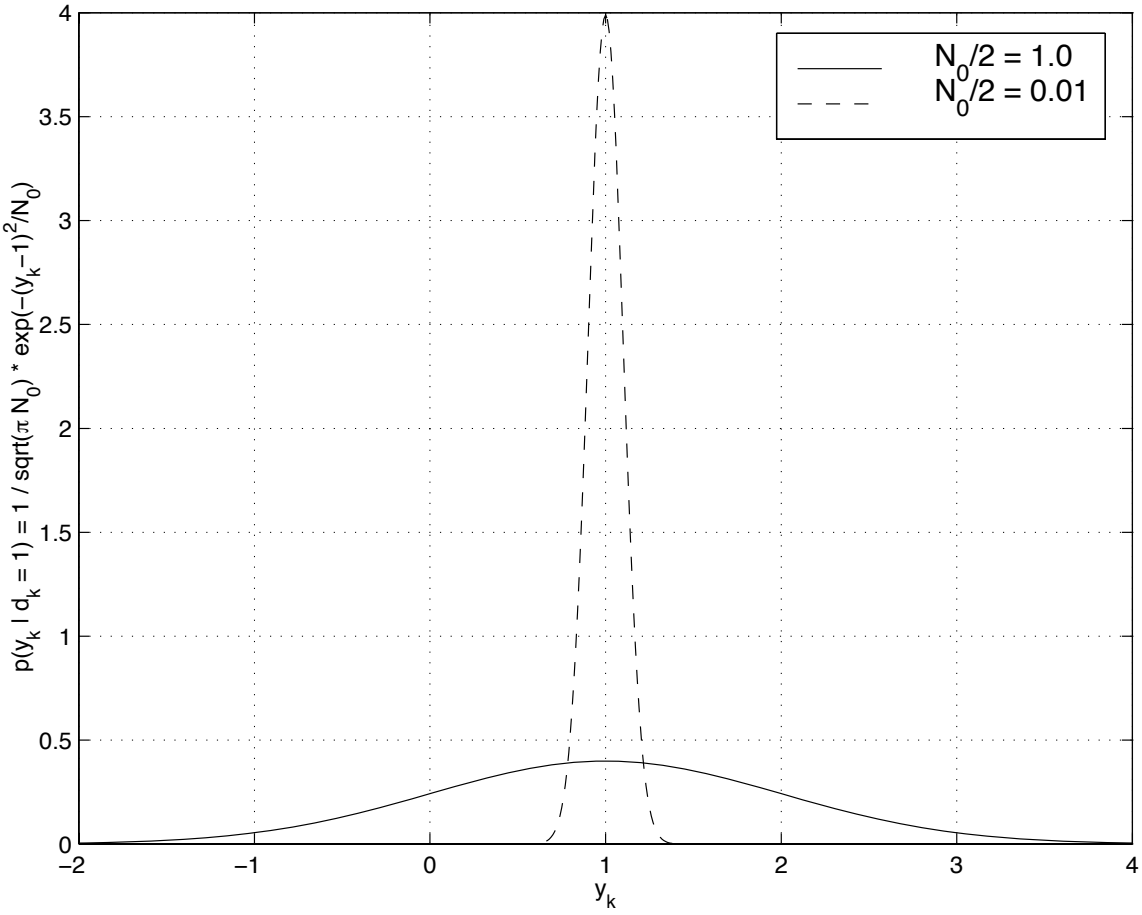

Figure 7.2: SNR Mismatch Effect on Gaussian Density

is to consider the density functions over several SNRs and then use the composite density function which averages over the SNRs. For simplicity, first consider the case where the SNR is assumed to be uniformly distributed over some range, A dB to B dB. Let γ represent the SNR in dB, $f_{\gamma}(g)$ be the probability density function of γ, and $F_{\gamma}(g)$ be the cumulative distribution function of γ. Then,

$$f_{\gamma}(g) = \frac{1}{B-A}, \qquad A < g \leq B \tag{7.1}$$

and

$$\begin{aligned} F_{\gamma}(g) &= \int_{A}^{g} f_{\gamma}(g')dg' && (7.2) \\ &= \frac{g}{B-A}, \qquad A < g \leq B. && (7.3) \end{aligned}$$

The Gaussian density function does not operate in the log domain, so the following variable transformation is performed, where $\gamma = 10\log_{10} E_b/N_0 = 10\log_{10} h$.

$$\begin{aligned} F_h(H) &= P(h \leq H) && (7.4) \\ &= P(10^{\gamma/10} \leq H) && (7.5) \\ &= P(\gamma \leq 10\log_{10} H) && (7.6) \\ &= F_{\gamma}(\frac{10\log_{10} H}{B-A}) && (7.7) \end{aligned}$$

Hence,

$$f_h(H) = \frac{\partial F_h(H)}{\partial H} \tag{7.8}$$

$$= \frac{10}{(B-A)\ln 10}\frac{1}{H}, \qquad 10^{A/10} < H \leq 10^{B/10}. \tag{7.9}$$

The conditional density function used in the rate R_c turbo code can then be calculated numerically as shown below for $E_s = 1$. For simulations, this density can be calculated offline for quantized values of $y_k - d_k$. In this way, the computation requirements for a real-time decoder can be significantly reduced, while slightly increasing the memory requirements.

$$p(y_k|d_k) = \int p(y_k|H, x_k)\, f_h(H) dH \tag{7.10}$$

$$= \int_{10^{A/10}}^{10^{B/10}} \frac{1}{\sqrt{\pi N_0}} \exp(-\frac{1}{N_0}(y_k - d_k)^2) \frac{10}{(B-A)\ln 10}\frac{1}{H}\,\mathrm{dH} \tag{7.11}$$

$$= \int_{10^{A/10}}^{10^{B/10}} \sqrt{\frac{R_c}{\pi H}} \exp(-H\, R_c(y_k - d_k)^2) \frac{10}{(B-A)\ln 10}\,\mathrm{dH} \tag{7.12}$$

The only parameters which need to be determined are A and B, which define the range of possible SNR values. For our simulations, we chose $A = 0$ and $B = 3$. There are two ways to justify this choice. First, we refer to Figure 7.3. Figure 7.3 shows the performance of SNR mismatched systems where the true SNR is displayed in the legend and these systems are plotted versus the SNR assumed in decoder calculations. Hence, Figure 7.3 is essentially a shifted version of Figure 7.1. The range of 0 to 3 dB was chosen because as shown in Figure 7.3, the performance of all systems is virtually constant when the decoder assumes an SNR in the range of 0 and 3 dB. The second

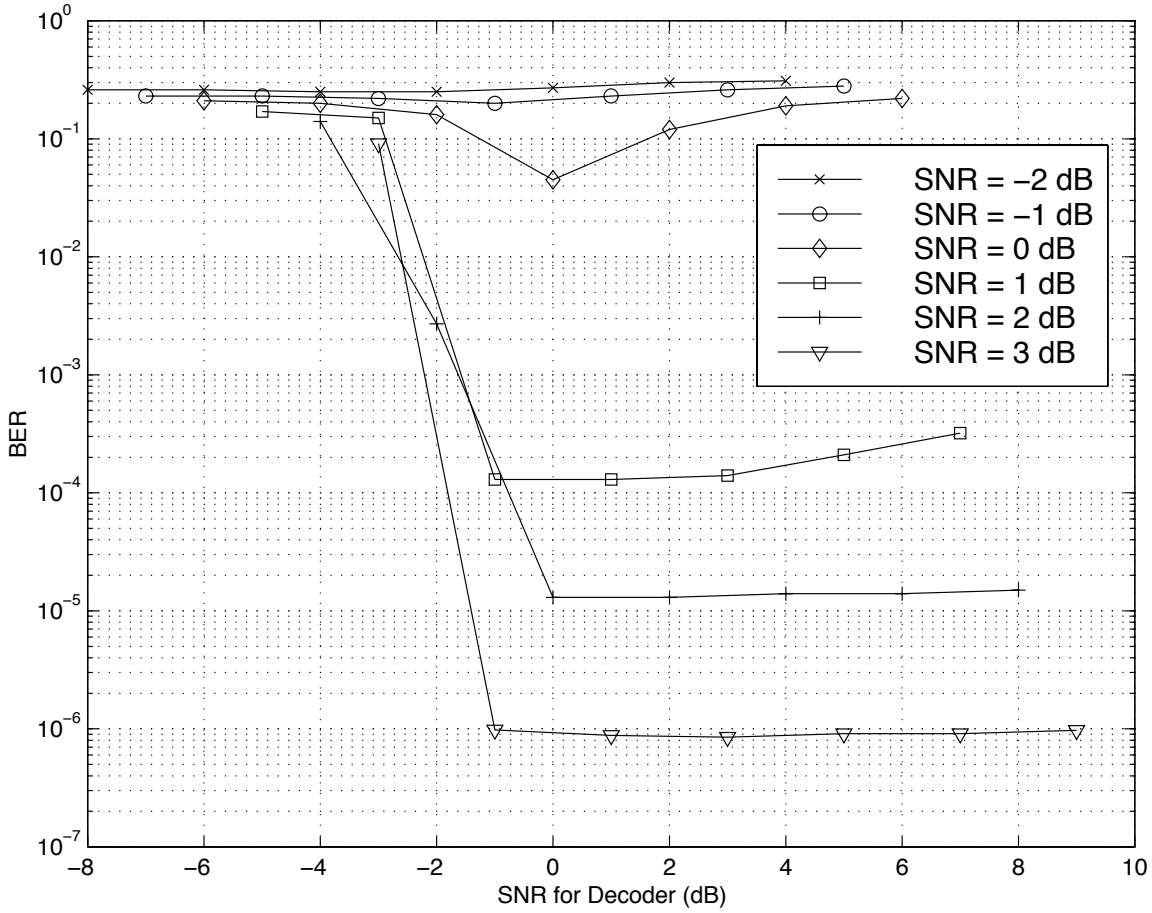

Figure 7.3: Mismatched Performance of True SNR versus Decoder SNR

reason is that the range of 0 to 3 dB appears to yield performance in the range of interest. Shown in Figure 7.1, the simulation results of systems even without any mismatch yield BER rates greater than $4.0e-02$ for SNR < 0. Such performance is worse than any reasonable system would require. For SNRs significantly greater than 3 dB, the performance without mismatch yields BERs much less than $1.0e-06$ which is perhaps better than most systems would require. If the upper range of the decoder SNR were set to 3 dB, the performance of systems with true SNRs up to 7 dB (i.e -4 dB offset) would probably be little affected given the trend shown in Figure 7.1 which showed that as the SNR increases, the BER performance becomes less sensitive to SNR overestimation. For such reasons, the SNR range of the decoder is assumed to be between 0 and 3 dB,

Figure 7.4 shows the simulation results of the robust decoder compared to the decoder with perfect knowledge of the channel SNR. As shown in the figure, there are performance losses for SNR < 0 dB (i.e. outside the "range" of the robust decoder), but these losses are small because the performance is generally poor in this range of SNRs. For SNR > 0 dB, however, there is little difference in performance. Hence, virtually identical performance can be obtained without expending any computation on SNR estimation techniques.

Similar results can be obtained for the case of noncoherent detection. Figure 7.5 shows the simulation results of various SNR systems versus the SNR offset of the decoder. Again, there are greater penalties for underestimating the SNR. Figure 7.6 compares the performance of the decoder which has perfect information of the channel SNR to a robust decoder which operates in the range of 7 to 9 dB. Again,

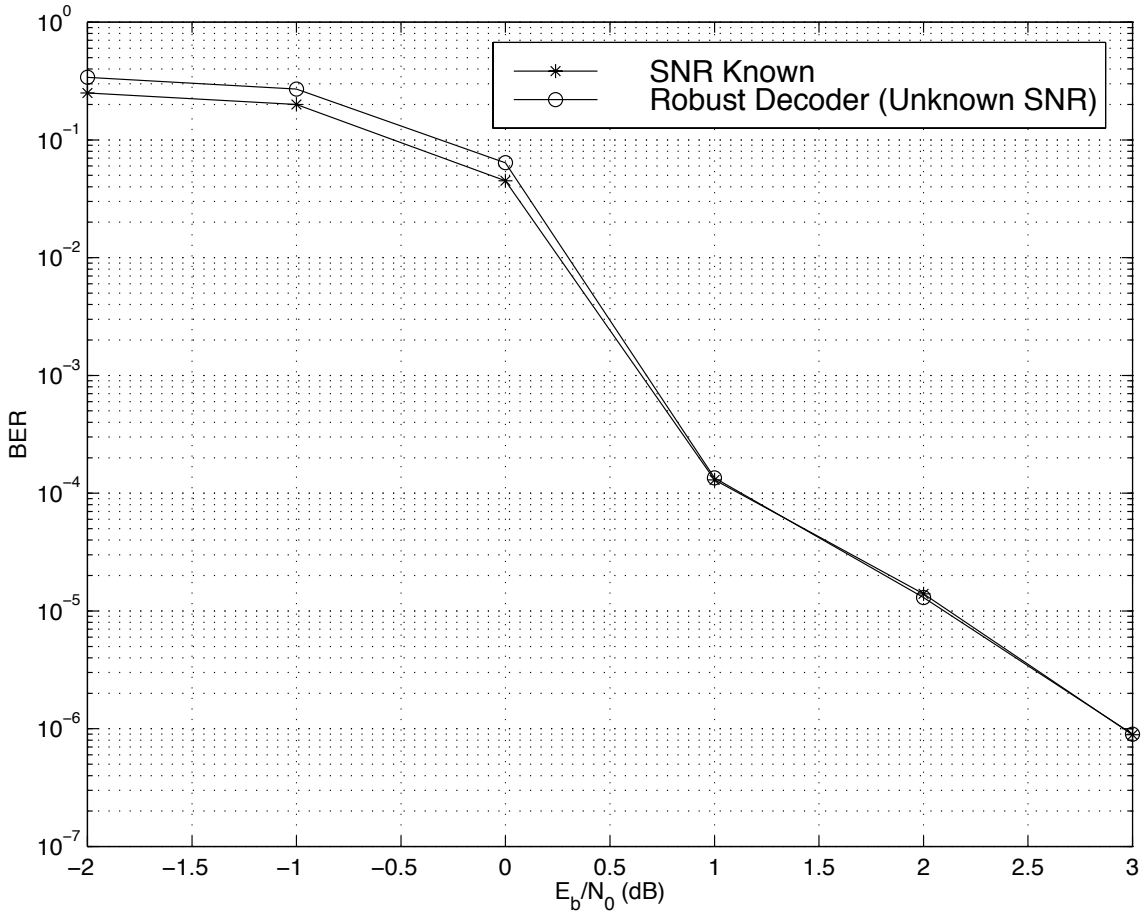

Figure 7.4: Performance of the Robust Decoder in Coherent AWGN

the performance of both systems are virtually identical.

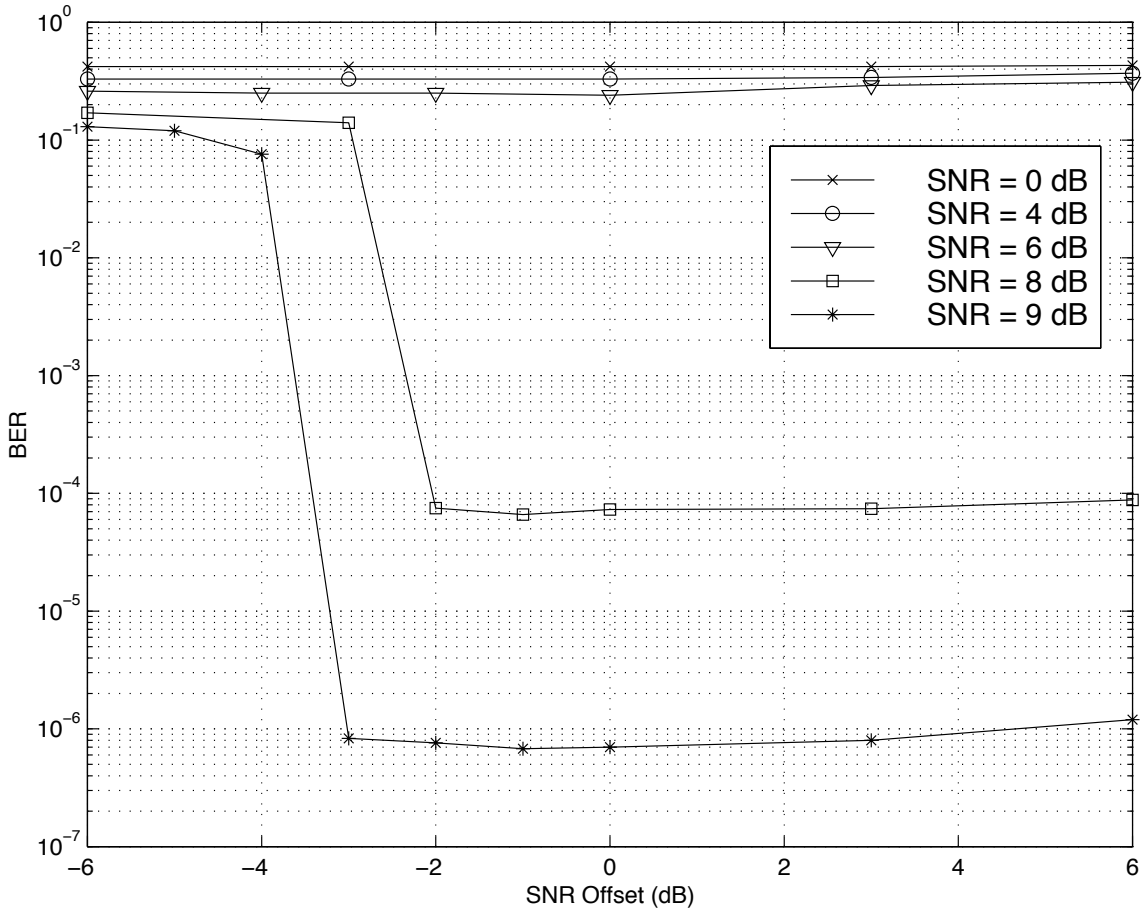

Figure 7.5: SNR Mismatch in Noncoherent AWGN

7.2 SNR Mismatch in Partial-Band Jamming

In this section, the robustness of turbo decoding is investigated in FH-SS with partial-band jamming and full-band thermal noise. In previous sections, it was assumed that the frequency hopper changed frequencies at a much greater rate than the jammer. Thus, the entire hop was assumed to be either completely jammed or unjammed. In addition, we considered the case where the data rate was faster

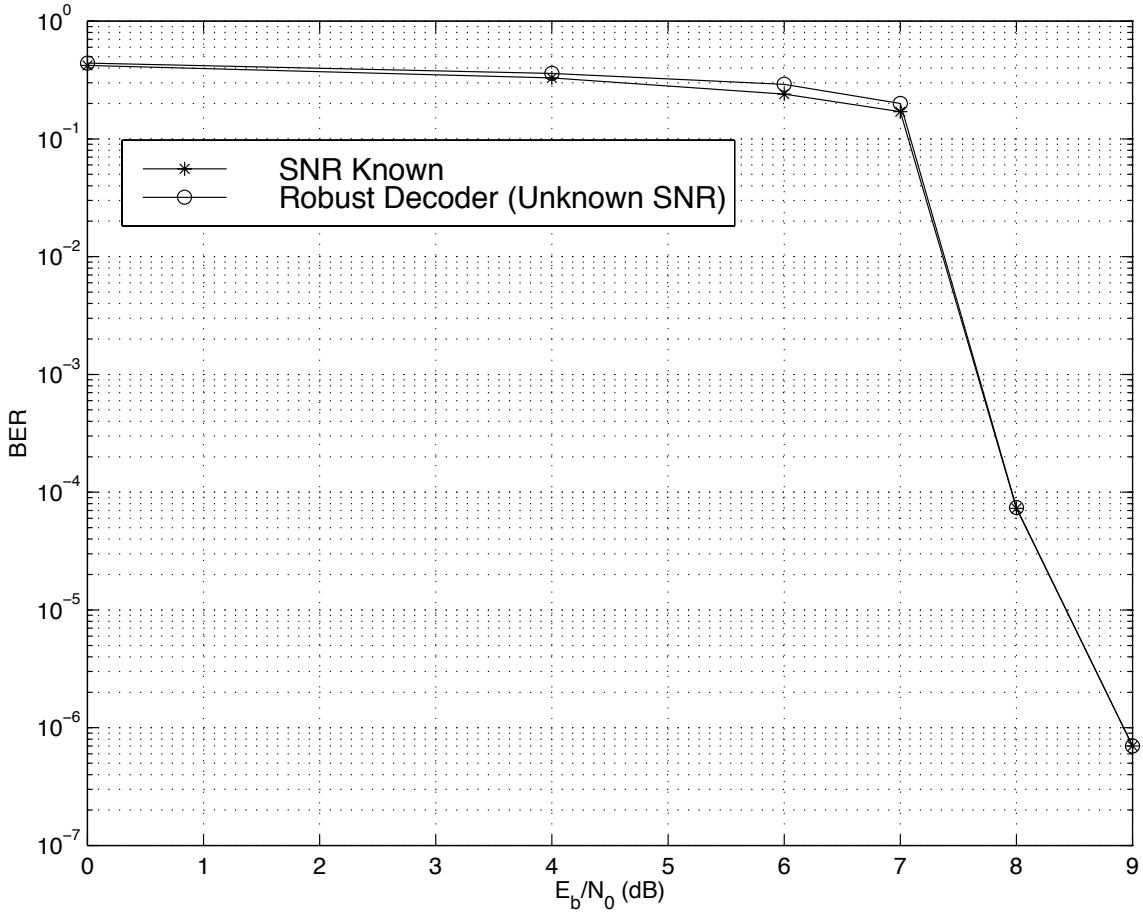

Figure 7.6: Performance of the Robust Decoder in Noncoherent AWGN

than the hopping rate. In such cases where multiple bits were transmitted over each hop, estimation techniques were employed to estimate which hops were jammed. At around 20 bits per hop, the receiver without perfect jamming side information (SI) was able to achieve performance virtually identical to the receiver with jamming SI due to effective estimation.

While the estimation procedures were highly effective in improving decoder performance, their success hinged on knowing several parameters of the channel among which were the power spectral densities of the jamming and thermal noise ($\frac{N_J}{2\rho}$ and $\frac{N_0}{2}$, respectively), and the fractional jamming bandwidth (ρ). If such information was corrupted in any manner, the performance of the estimation techniques would suffer.

There exists many situations in which channel estimation is impractical. The jammer may change frequencies very quickly which would yield few bits per hop from which estimation procedures could estimate channel parameters. The jammer may also change its fractional bandwidth or even introduce multiple levels of jamming. This would make it more difficult to estimate whether a hop was jammed and if so, what the appropriate jamming power was. The goal of this section is to investigate the robustness of turbo codes in FH-SS with partial-band interference with little to no knowledge of the channel. In our discussion, two receivers for noncoherent reception will be considered.

7.2.1 Square Law Combining Receiver

First, consider the square law combining receiver. The deleterious effects of jamming using square law combining for noncoherent reception have been studied in detail [5] [18] [22] [23] [16]. Particularly in [5], it was determined that reliable knowledge of the jamming channel state, z_k, can significantly improve the performance. This is particularly true for soft decision decoders which make bit decisions based on the conditional densities of the channel outputs. Consider the conditional densities shown in Figures 7.7, 7.8, and 7.9. These figures show the conditional densities of the noncoherent square law outputs, Y_{+1} and Y_{-1}, given $d_k = 1$ when the channel is known to be unjammed (Figure 7.7), jammed (Figure 7.8), and when the decoder has no knowledge of the channel state (Figure 7.9). In the final case, the decoder computes conditional densities which are averaged over each jamming state (3.7).

$$p(y_k|d_k = i) = p(y_k|d_k = i, z_k = 1)\,\rho + p(y_k|d_k = i, z_k = 0)\,(1 - \rho) \qquad (7.13)$$

In these figures, the jamming and thermal SNRs are $E_b/N_J = 9$ dB and $E_b/N_0 = 20$ dB, respectively, and the fractional jamming bandwidth is $\rho = 0.5$. Note that in all three cases, the channel parameters, N_0, N_J, and ρ are assumed to be known to the decoder.

Figure 7.9 shows that the conditional density averaged over the jamming state, z_k, is far different from each of those shown in Figures 7.7 and 7.8. Hence, even if N_0, N_J, and ρ are known to the decoder, performance is seriously degraded if z_k is unknown. Next, note the large differences between the densities of Figures 7.7 and

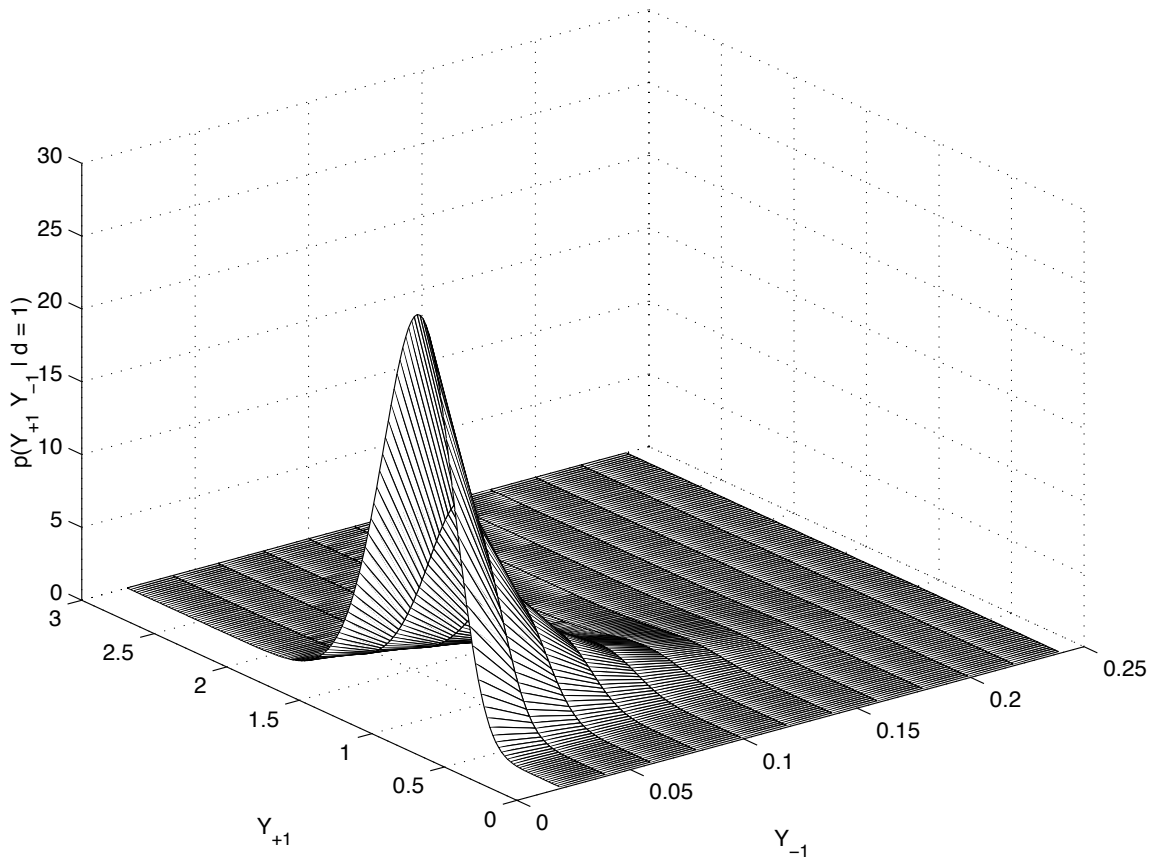

Figure 7.7: Conditional Density: Known $E_b/N_0 = 20$ dB, $E_b/N_J = 9$ dB, $\rho = 0.5$, and $z_k = 0$

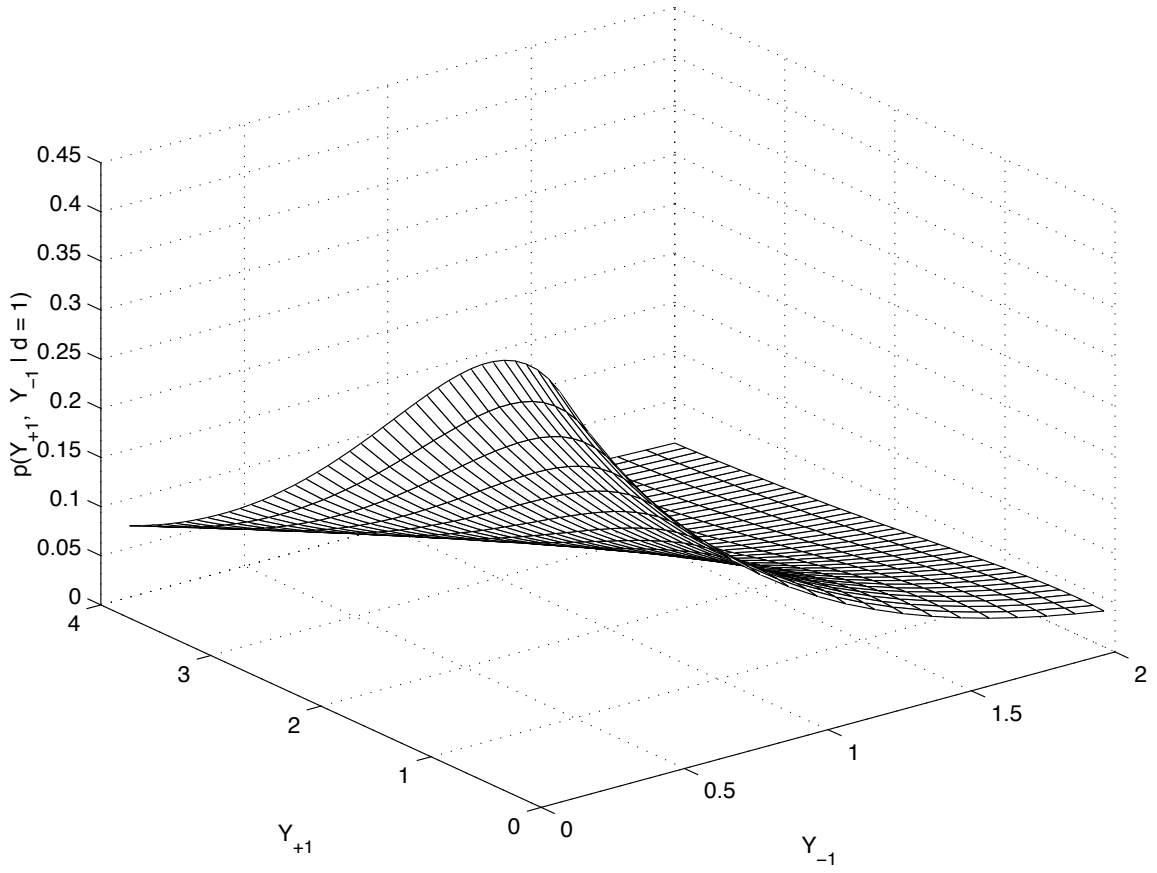

Figure 7.8: Conditional Density: Known $E_b/N_0 = 20$ dB, $E_b/N_J = 9$ dB, $\rho = 0.5$, and $z_k = 1$

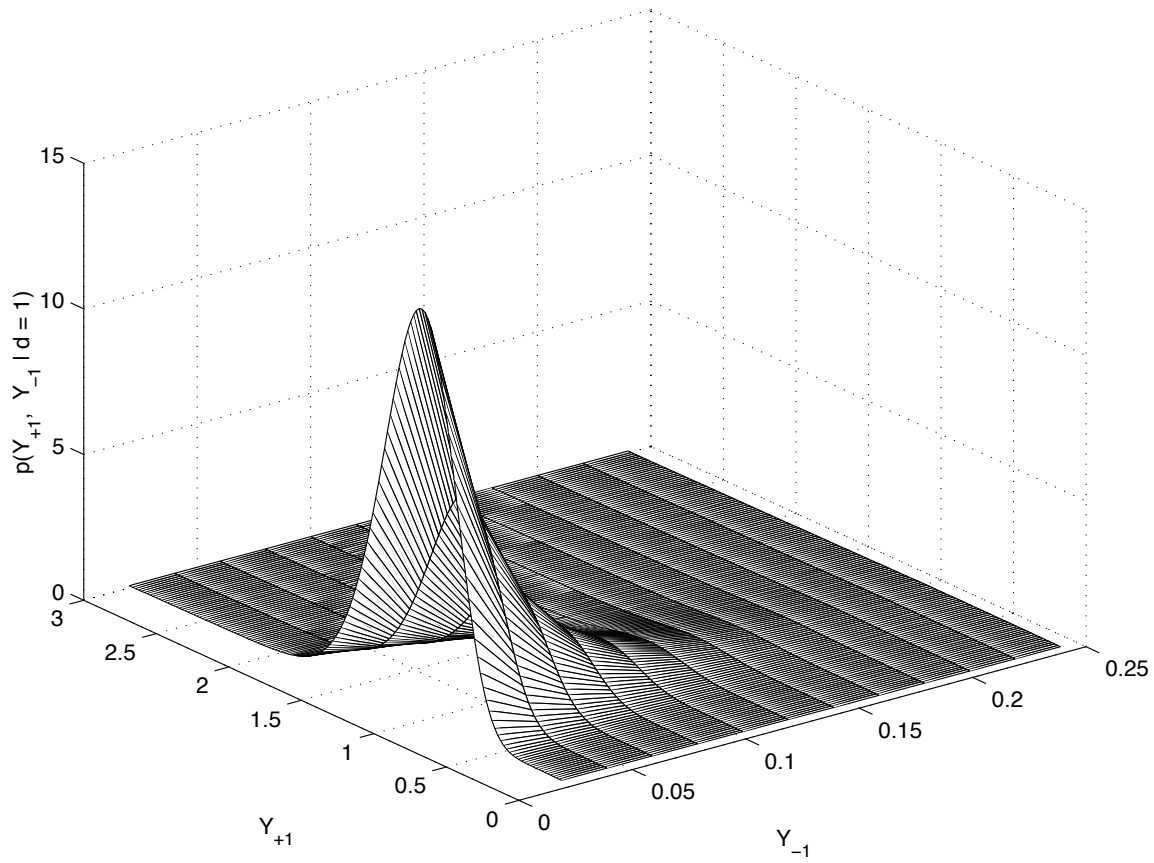

Figure 7.9: Conditional Density: Known $E_b/N_0 = 20$ dB, $E_b/N_J = 9$ dB, and $\rho = 0.5$; Unknown z_k

7.8. Unlike the AWGN case where the range of interest was small (between 0 and 3 dB), partial-band jamming makes it difficult for a robust decoder to encompass the large range of SNRs.

Given the results of the previous section, however, we would be amiss if we did not even consider the decoder which assumes a uniform distribution on the SNRs. Shown in Figure 7.10 are the simulation results for turbo codes in FH-SS with partial-band jamming when the packet length is 1760 information bits and there are 160 bits per hop. The BER performance of two systems are shown for different values of the fractional bandwidth, ρ. The first system is assumed to have knowledge of the fractional bandwidth, ρ, and the noise power spectral densities, $N_0/2$ and $N_J/(2\rho)$, but has no knowledge of z_k. Hence, the decoder uses (7.13) in its computations. The second system is assumed to have no knowledge of the channel. The decoder computes conditional densities which are averaged over a uniformly distributed SNR range. In Figure 7.10, this range is assumed to be 6 to 11 dB for similar reasons given in the previous section.

For $\rho = 0.9$, there is little difference in performance between the two systems. At high values of ρ, most of the hops are jammed and the power spectral density of the jammer, $N_J/(2\rho)$, does not cause a large SNR mismatch with the robust decoder which assumes the SNR to be in the range of 6 to 11 dB. At $\rho = 0.5$, the effects of the jammer become more obvious. The SNR mismatch increases as the interference of the channel is either $N_J/(2\rho) + N_0/2$ or $N_0/2$. These mismatches cause degradation in the performance of the robust decoder. Finally, the results for $\rho = 0.1$ are discussed. In Figure 7.10, the simulation results for the receiver which knows the power spectral

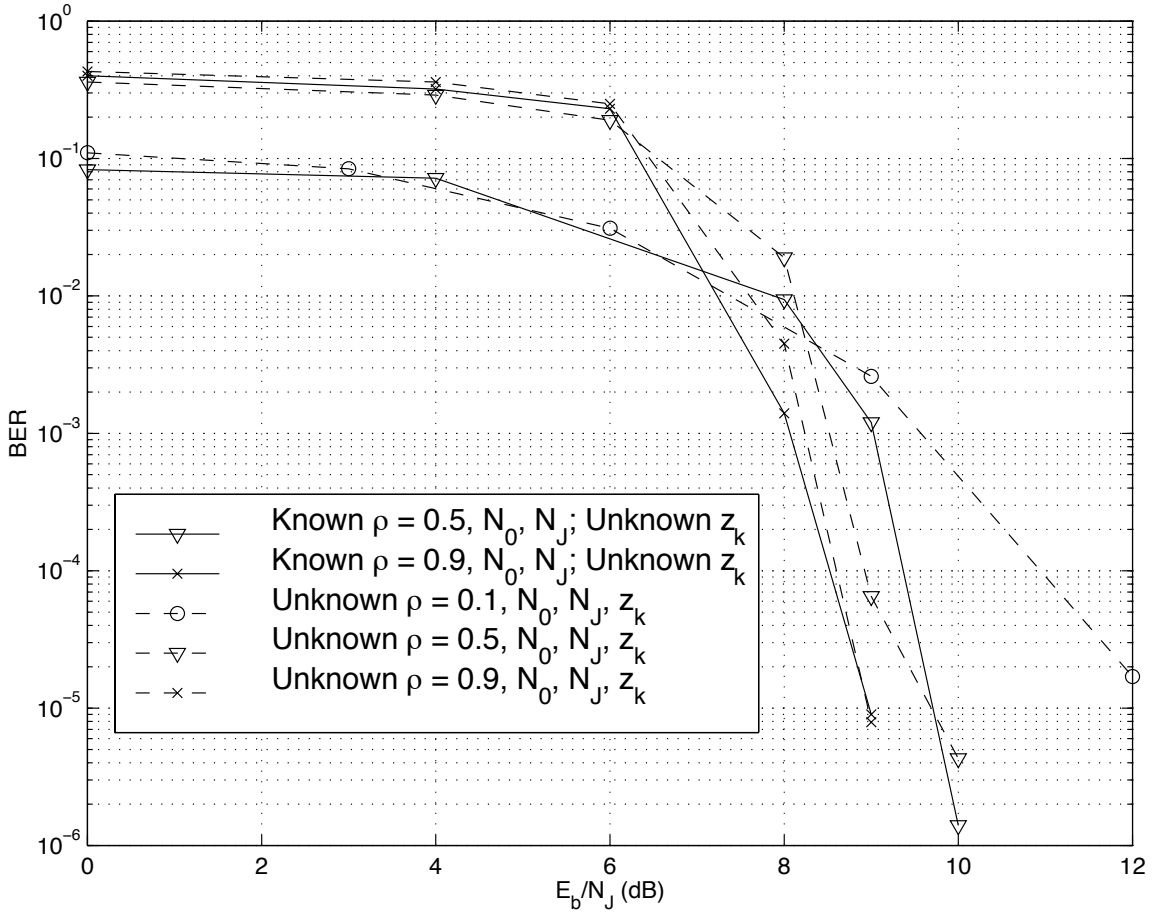

Figure 7.10: Performance Comparison of Known and Unknown Channel Parameters

densities and the fractional jamming bandwidth is not shown. The simulations for this receiver even at $E_b/NJ = -10$ dB yielded no detectable errors. In contrast, however, the decoder with no channel information requires more than 10 dB to achieve BER $< 10^{-4}$. In fact, at high SNRs, the BER performance gets worse and worse as the value of ρ decreases towards 0. At $\rho = 0.01$, 12 dB yielded BER $= 2.1e^{-03}$. This matches the analytical results [31] which reveal that as the SNR increases, the value of ρ which yields worst case jamming decreases towards zero. For any decoder design, it must be assumed that the jammer will be intelligent (i.e. the jammer will choose ρ to yield worst-case jamming). It is unlikely that BER $= 10^{-3}$ is a satisfactory result at 12 dB, so we consider another possible decoder.

The previous decoder only assumed to know the performance of turbo codes in noncoherent FH-SS. Based on this performance, a suitable SNR range was selected and SNRs within this range were assumed to be uniformly distributed. While this approach worked in the AWGN channel, this approach does not yield good performance in a channel with partial-band jamming due to the increased levels of SNR mismatch. If the level of SNR mismatch can be reduced, then as shown in Figure 7.10, performance can be improved significantly.

It was mentioned before that it may be unsuitable to assume that pertinent channel parameters such as ρ, $N_J/(2\rho)$, and $N_0/2$ are known because the jammer may change frequencies quickly, may vary its fractional jamming bandwidth, and may even vary the number of jamming levels. It may be possible, however, to estimate the average power of the interference, $\frac{N_J+N_0}{2}$. If the average noise powers are assumed to be known, then the level of mismatch will be reduced and perhaps the performance

will improve.

Figure 7.11 shows the simulation results of the decoder which uses the average interference, $\frac{N_J+N_0}{2}$, to decode all square law combined outputs. In the simulations, the packet length is 800 information bits and there are 80 bits per hop. As shown

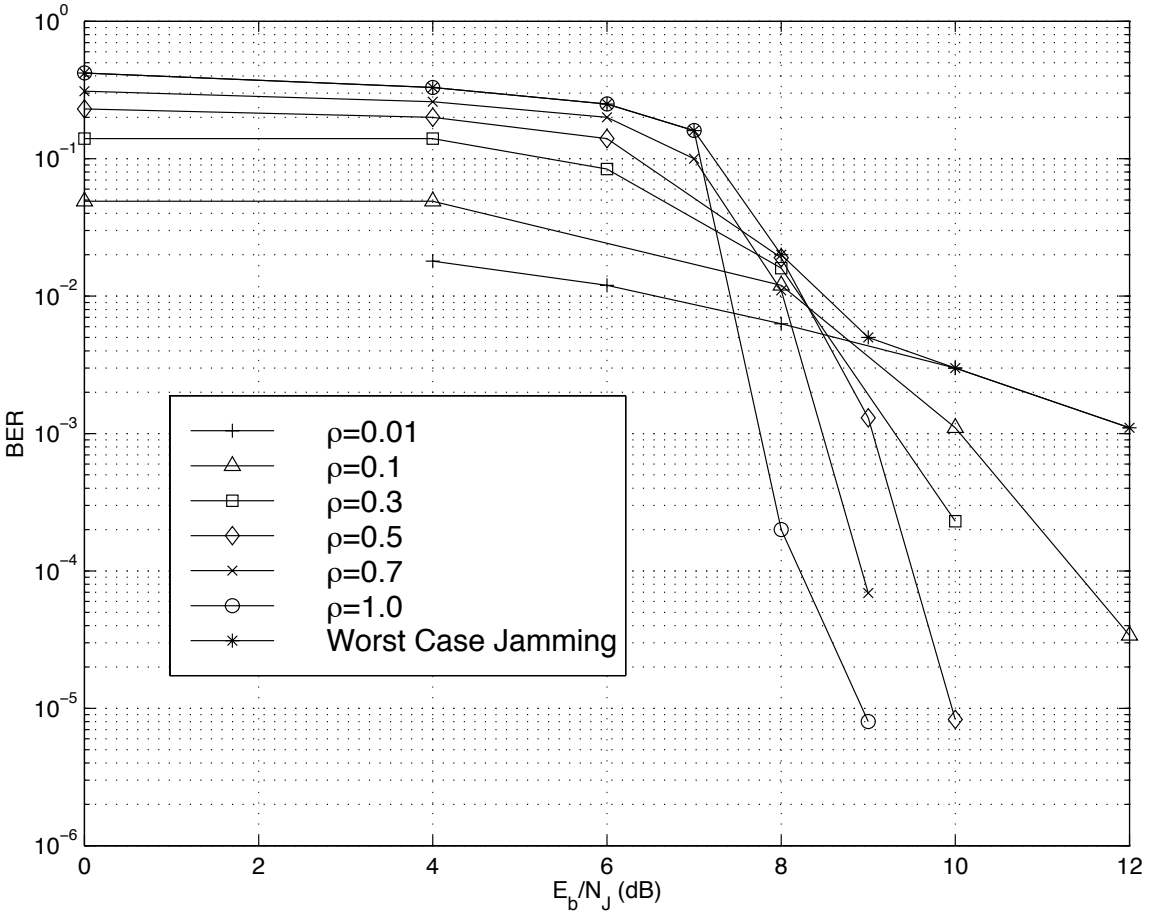

Figure 7.11: Performance of the Decoder which Uses Average Power to Decode Square Law Combined Channel Outputs

in the figure, the performance improvements are marginal. Especially at high SNRs, the worst case jamming performance is still dominated by small values of ρ.

Hence, the square law receiver does not yield good performance when limited channel knowledge is available. Next consider another receiver which has been developed

for channels with partial-band jamming.

7.2.2 Self-Normalizing Receiver

One receiver which has shown strong resistance to partial-band interference is the self-normalizing receiver [42] [43]. In essence, the channel outputs are normalized in order to minimize the deleterious effect of jammed bits. If Y_1, Y_2, ..., Y_M are the square law detected outputs corresponding to each of M modulated signals, then the self-normalized receiver uses the metric

$$S_i = \begin{cases} Y_i/\Omega & i = 1, ..., M-1 \\ \Omega & i = M \end{cases} \tag{7.14}$$

where $\Omega = \sum_{i=1}^{M} Y_i$.

The conditional densities of Y_i can be computed as in (3.13). Using the series representation of the modified zeroth order bessel function,

$$I_0(\sqrt{x}) = \sum_{k=0}^{\infty} \frac{(x/4)^k}{n!n!} \tag{7.15}$$

the joint density of the transformed variables in (7.14) can be obtained [43].

$$f_{S_1,..,S_{M-1}}(s_1, .., s_{M-1}) = e^{-\gamma} \sum_{k=0}^{\infty} \frac{(k+M-1)!}{n!n!} (\gamma\, s_1)^k \tag{7.16}$$

$$= (M-1)! e^{x-\gamma} L_{M-1}^0(-x)|_{x=\gamma s_1} \tag{7.17}$$

where $\gamma = \rho E_s/N_J$ and $L_n^{\alpha}(\cdot)$ are the Laguerre polynomials [44] which are defined as

$$L_n^{\alpha}(x) = e^x \frac{x^{-\alpha}}{n!} \frac{d^n}{dx^n}(e^{-x}x^{n+\alpha}) \qquad n = 0, 1, 2, .. \tag{7.18}$$

Hence, for $M = 2$, the density for the self-normalized metric is

$$f_{S_1}(s_1) = (1 + \gamma s_1)e^{x-\gamma}. \tag{7.19}$$

Note that for $M = 2$, the normalized metric produces only one random variable since one metric completely defines the other (i.e. $s_2 = 1 - s_1$).

For the case where $M = 2$, the Chernoff parameter can be computed as [43] where thermal noise is ignored and the partial-band jamming is assumed to dominate the performance.

$$D = \max_{0<\rho\leq 1} \min_{\lambda\geq 0} D(\lambda, \rho) \tag{7.20}$$

$$= \max_{0<\rho\leq 1} \min_{\lambda\geq 0} \left[(1-\rho)e^{-\lambda} + \frac{\rho}{(2\lambda-\gamma)^2}(2\lambda e^{\lambda-\gamma} + (\gamma^2 - 2\lambda - 2\lambda\gamma)e^{-\lambda})\right]^L \tag{7.21}$$

where L represents the diversity of the channel. As shown in (7.21) the Chernoff parameter is computed for worst-case jamming.

Figure 7.12 shows the analytical results of the self-normalizing (SN) and the square law combining (SLC) receivers with diversity L concatenated with a rate 1/3 turbo code of packet length 800 information bits. Note that these analytical results assume that there are no SNR mismatches.

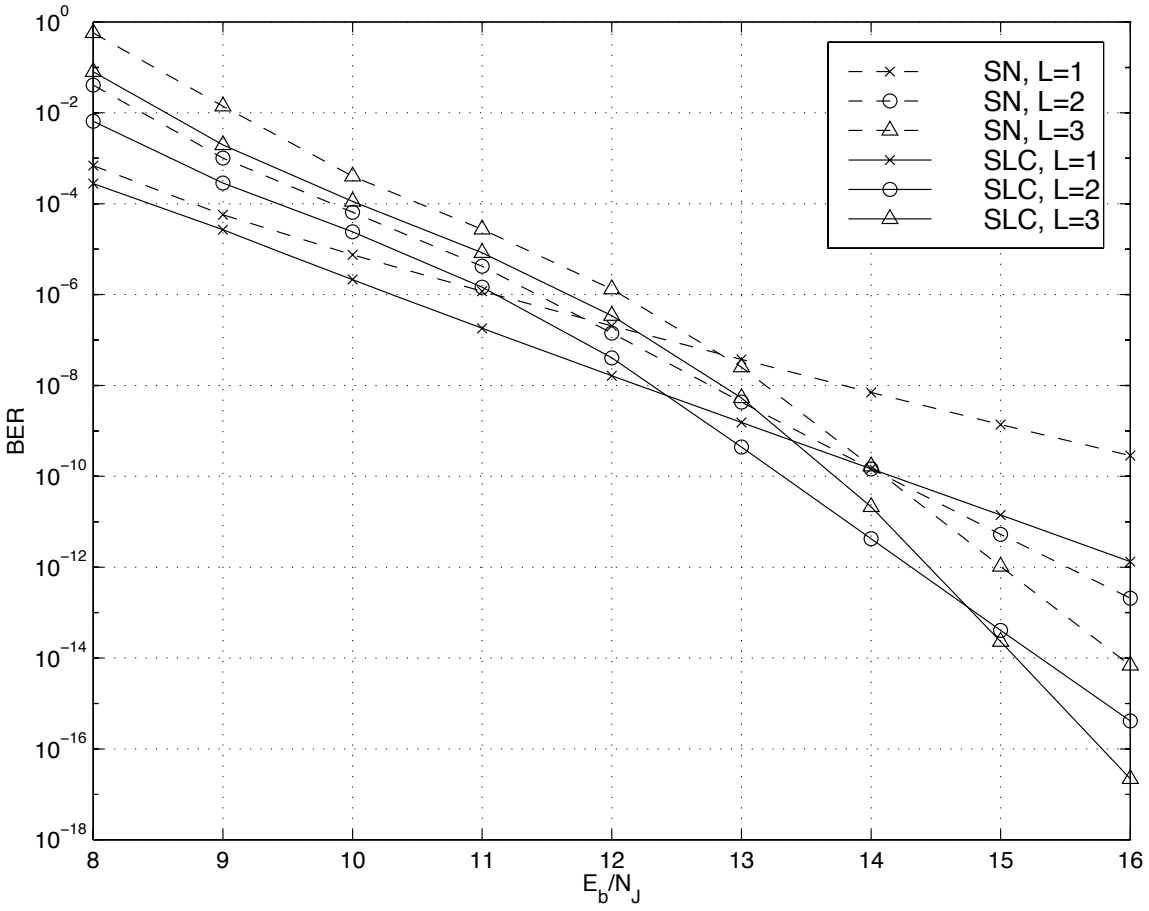

Figure 7.12: Analytical Results of the Self-Normalizing and Square Law Combining Receivers with Diversity L Concatenated with a Turbo Code ($E_b/N_0 = \infty$)

The self-normalizing receiver performs closest to the square law combining receiver at low E_b/N_J. This is the region where jamming has the largest effect on system performance. Hence, this is the region where the self-normalizing receiver is most effective. At high E_b/N_J, the performance of the self-normalizing receiver performs progressively worse relative to the square law receiver because in this region, the received signal consists of little noise. In this region, signal energies are essentially being normalized.

Next, consider the performance of diversity across low and high SNRS. At low E_b/N_J, $L = 1$ performs best. In this region, the worst case jamming typically occurs for $\rho = 1$ [31]. Hence, for low jamming SNR, the gain of diversity is very small, if any gain exists at all. However, the price of diversity for noncoherent reception is noncoherent combining loss. At high E_b/N_J, $L > 1$ gives best results. The increase in diversity could be mitigated by increasing the fractional jamming bandwidth. However, in this high SNR regime, the jammer has little power to spread, making it unable to combat increased diversity. Note that these results do not include thermal noise. If thermal noise existed in the system, then clearly there would come a point where increasing L would yield worse performance. The degradation incurred by the noncoherent combining loss would eventually fall under the thermal noise floor.

Figure 7.13 shows the simulation results of the self-normalizing receiver with $L = 1$ when the turbo decoder uses the average power, $\frac{N_J+N_0}{2}$, in its calculations. Similar to the results in Figure 7.11, the packet size spanned 800 information bits and the number of bits per hop were 80.

Shown in the figure, the self-normalized metric succeeds in removing the effects

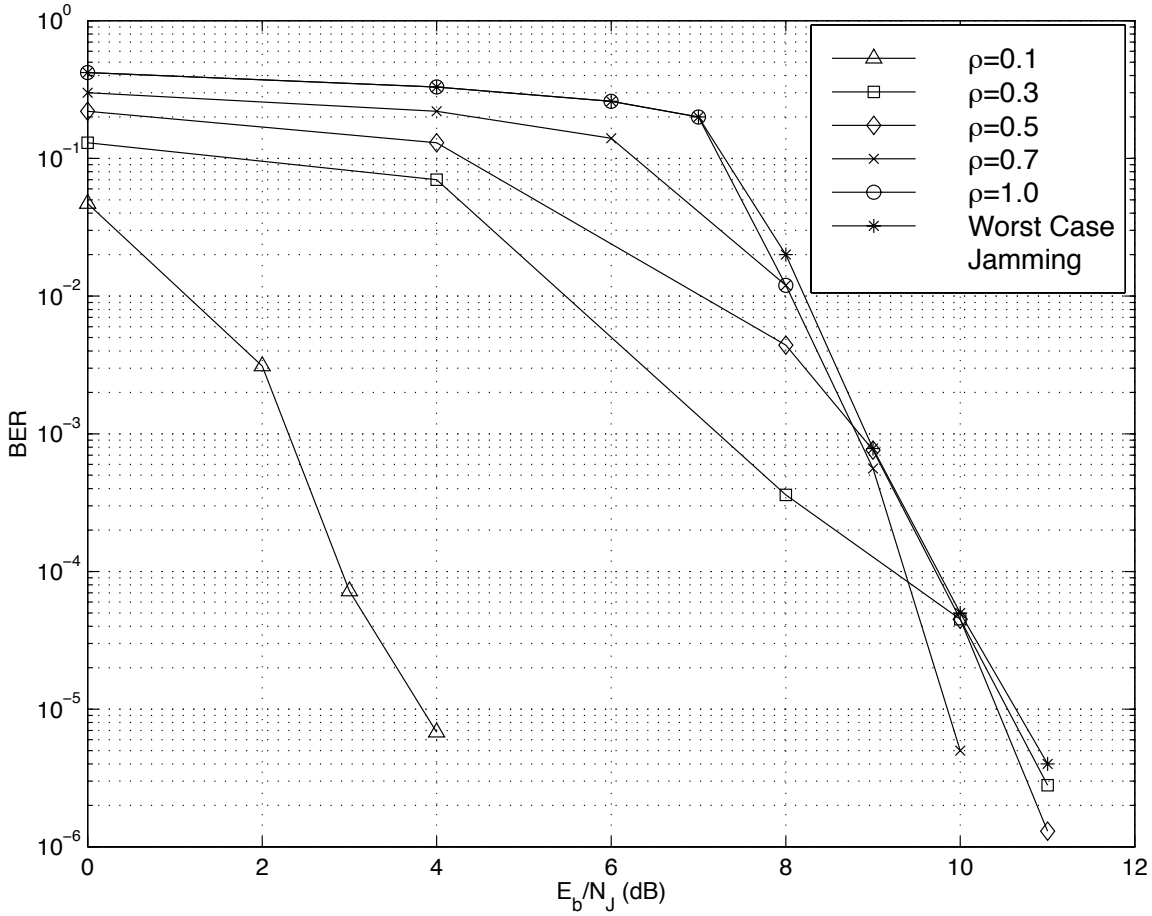

Figure 7.13: Performance of the Decoder which Uses Average Power to Decode Self-Normalized Channel Outputs

of narrow band jamming (i.e. $\rho \approx 0$) at high SNRs. Meanwhile, the worst-case jamming performance at low to mid-range SNRs is virtually unchanged. While these results show greater anti-jam capability relative to square-law detection, it is useful to compare the mismatched results to the case without any mismatch. These results for worst-case jamming are shown in Figure 7.14.

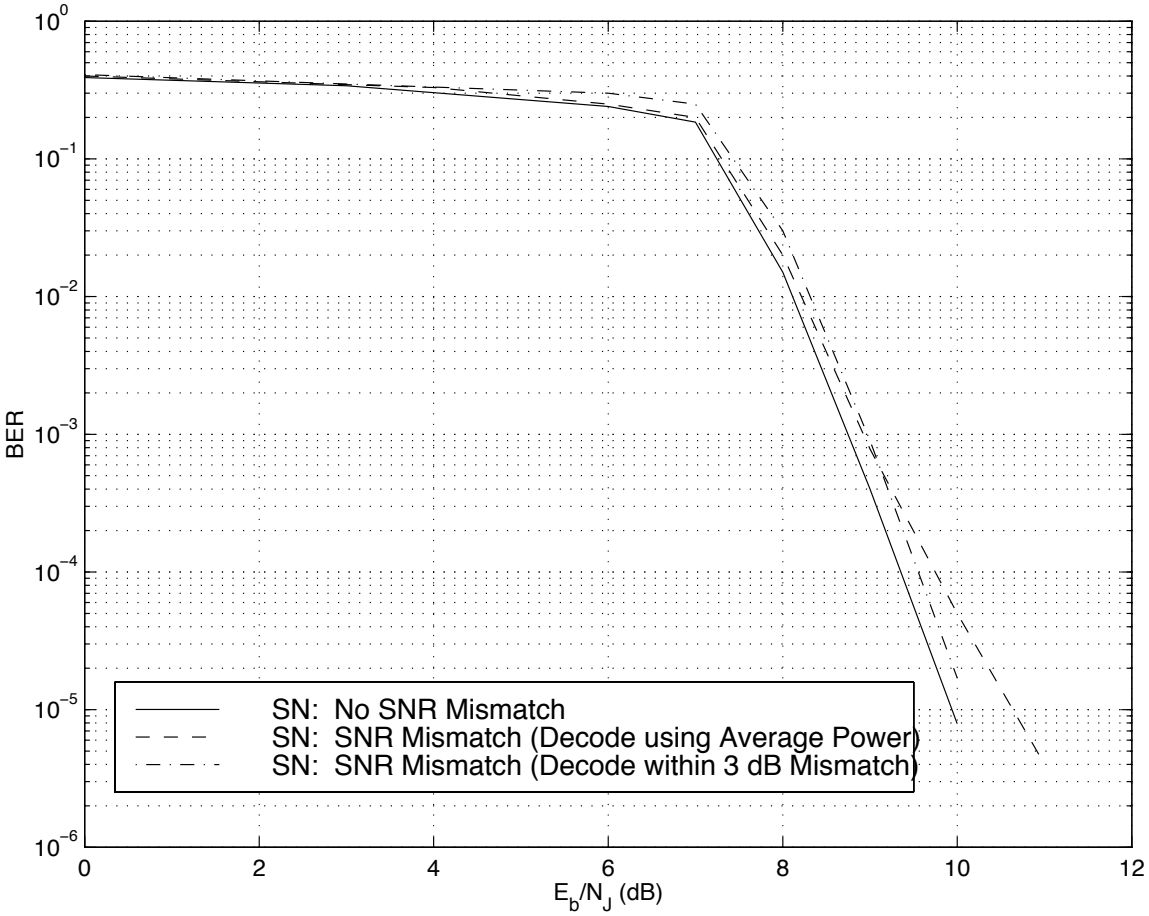

Figure 7.14: Worst Case Performance of the Self-Normalizing Receiver with and without SNR Mismatch ($0.1 \leq \rho \leq 1$)

For $E_b/N_J \leq 7$ dB, the plots are indistinguishable. In this region, worst-case jamming occurs for $\rho = 1.0$. Hence, the method without mismatch (i.e. uses $N_0/2 + N_J/(2\rho)$) and with mismatch (uses the average power $N_0/2 + N_J/2$) are

identical. As E_b/N_J increases, however, the fractional bandwidth which yields worst-case performance decreases towards zero. Hence, in the high SNR regime, as E_b/N_J increases, so does the mismatch for the decoder which does not know the instantaneous power spectral densities. While the loss at BER $= 10^{-5}$ is about 0.7 dB, this margin will steadily increase for higher SNRs.

Also shown in Figure 7.14 is the simulation result for the receiver which attempts to estimate the instantaneous power spectral density. Figure 7.14 plots the performance of the decoder which can estimate the power spectral densities ($\frac{N_0}{2} + \frac{N_J}{2\rho}$ for jammed states, $\frac{N_0}{2}$ for unjammed states) to within 3 dB. As shown in the figure, the performance improvement is greatest for high E_b/N_J. In particular, relative to the decoder which uses average power, the mismatch is constant across all E_b/N_J. Hence, the performance relative to the decoder without mismatch is virtually constant across E_b/N_J.

Finally, the results of the square law receiver shown in Figure 7.15 are compared to the self-normalizing receiver with similar mismatches in Figure 7.14. While the square law receiver with no SNR mismatch shows a 0.8 dB gain with respect to the self-normalizing receiver at BER $= 10^{-5}$, the gain diminishes to 0.4 dB for the decoder with a 3 dB mismatch. If only the average power can be determined, the square law receiver experiences significant loss with respect to the self-normalizing receiver at high E_b/N_J. Hence, the self-normalizing receiver is less sensitive to changes in channel information.

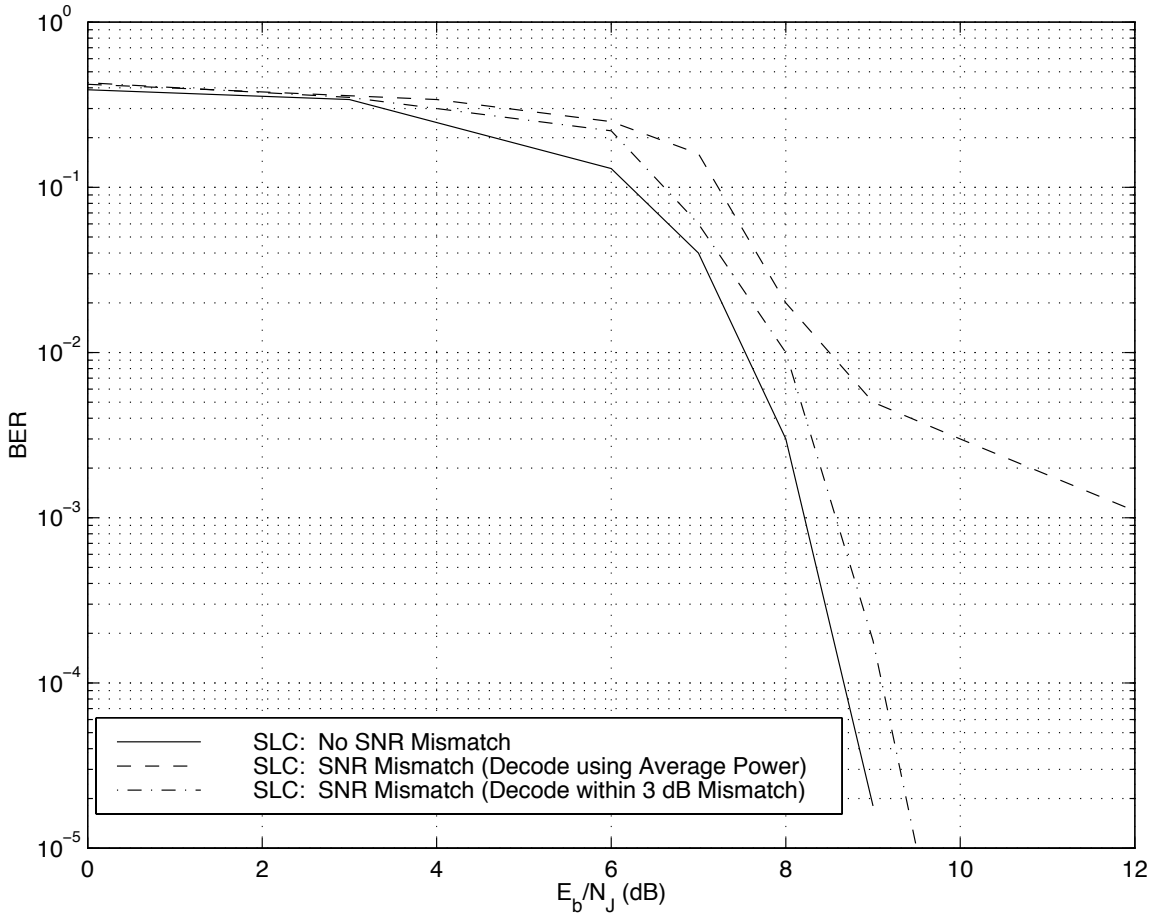

Figure 7.15: Worst Case Performance of the Square Law Receiver with and without SNR Mismatch ($0.1 \leq \rho \leq 1$)

7.3 Conclusion

In this chapter, the robustness of turbo codes in the AWGN channel and in FH-SS with partial-band interference was considered. In AWGN, a robust scheme was developed which requires no knowledge of channel statistics. Without introducing any complexity, its performance was virtually identical to the case where SNR knowledge was assumed. For the channel with partial-band interference, the self-normalizing receiver was shown to achieve superior performance over the square law receiver when channel knowledge is absent. In such cases, each receiver assumed the channel to be AWGN. In addition, for the case where SNR is known exactly, the performance of the self-normalizing receiver is within a decibel of the square law receiver.

CHAPTER 8

Conclusions and Future Work

The design and performance of turbo codes with joint iterative channel estimation have been considered for several channels with memory. The channels which were considered included the partial-band interference channel, the Rayleigh fading channel, and the channel with both partial-band interference and Rayleigh fading. For these channels, a frequency-hopped spread spectrum system was considered. In addition to the systems with FH-SS, a more general channel with memory, the Gilbert-Elliot burst channel model, was considered.

The approach was to take advantage of the channel memory by calculating estimates of certain channel parameters when side information was unavailable. For channel estimation computations, a recursive algorithm was developed to reduce the computational complexity incurred by this procedure. Simulation results revealed a performance tradeoff for varying lengths of the channel memory when fixed packet lengths were considered. The benefit of increasing channel memory was improved performance of the channel estimation schemes. It was shown that if the channel

memory was sufficiently long to permit a reliable estimate, performance virtually identical to the cases with side information could be obtained. For the partial-band jamming channel, 20 bits per hop provided sufficient memory for the channel estimation scheme to provide performance similar to the case with side information. Increasing the channel memory, however, also reduced the effective block length of the code and led to performance degradations.

In addition to simulation results, analytical results were calculated to upper bound the performance of turbo codes. These results are especially useful in the regions with high signal-to-noise ratios since it is computationally intensive to generate simulation results for extremely low bit error rates. The calculation of the bound requires the weight enumerator of the code and the pairwise error probability. Due to the presence of the interleaver in the encoder, it is difficult to compute the weight enumerator for a turbo code of block length N. As a result, an average upper bound which used a weight enumerator averaged over all possible interleaving schemes was considered. Once this average weight enumerator is calculated, knowledge of the pairwise error probabilities is required to compute the union bound. Where possible, the pairwise error probability is calculated. For other cases, the Bhattacharrya parameter or the Chernoff parameter was considered. These average upper bounds were shown to be tight especially at high signal-to-noise ratios.

Both simulation and analytical results were compared to other well-known coding schemes and decoding algorithms. For the frequency-hopped spread spectrum system with partial-band interference, the worst case jamming performance of turbo codes with noncoherent reception was 6 dB better than the worst case jamming per-

formance of a concatenated code consisting of a Reed-Solomon outer code and a convolutional inner code for packet error rates of 10^{-3}. For the frequency-hopped spread spectrum system with measured fading, the iterative estimation and decoding scheme yielded a gain of over 15 dB at a BER of 10^{-5} when compared to an errors and erasures Reed-Solomon decoder similar to the one used in current packet radio networks (SINCGARS).

Finally, the robustness of turbo codes was investigated in cases with mismatched channel parameters. It was shown in the additive white Gaussian noise channel that the sensitivity to signal-to-noise ratio mismatch decreases as the signal-to-noise ratio increases. In addition, the penalty of underestimating the signal-to-noise ratio is generally more severe than overestimating it. Despite the degradations which were shown to result from signal-to-noise ratio mismatch, a robust decoder with little additional complexity was exhibited to yield virtually identical performance to the decoder with perfect knowledge of the signal-to-noise ratio. This decoder computed branch transition probabilities averaged over a signal-to-noise ratio range of interest. For the channel with partial-band interference, the self-normalizing receiver was shown to achieve large performance gains over the square law receiver when statistics of the jammer were absent. In such cases, each receiver assumed the channel to be the additive white Gaussian noise channel.

While turbo codes were shown to be effective in reducing the signal-to-noise ratio required to achieve given performance requirements, the iterative MAP decoding algorithm requires considerable computational cost. Presumably, the receiver for a wireless system has finite battery life. Thus, before turbo codes can be integrated

into any wireless communications system, the computational complexity needs to be reduced while minimizing performance losses. The work in this thesis exemplifies the error correction power of turbo codes. Future research should investigate the application of lower complexity turbo decoders to wireless communications systems. A brief description of a few common reduced complexity decoders is presented in the Appendix. The Max-Log-MAP method is of particular interest. While significantly reducing the complexity of the decoder calculations, the loss with respect to the MAP decoder for a packet size of 1024 in AWGN is less than 0.5 dB [11].

Another area of future interest would be to apply similar channel estimation techniques to the case of noncoherent detection. Consider a frequency-hopped spread spectrum system where the phase is assumed to vary slowly over the duration of each hop. For this case, joint decoding and phase tracking could be performed. If the phase estimates could be computed reliability, significant performance improvements could be achieved.

APPENDICES

APPENDIX A

Helical Interleaver

First introduced by Berlekamp and Tong [29], helical interleavers have the function of terminating the trellis in the turbo encoder. Customary non-systematic encoders (NSC) can be driven to the all-zero state by a sequence of M zeros. Thus, a turbo encoder consisting of NSCs can be terminated by a sequence of M zeros. For recursive systematic codes (RSC), a sequence of M zeros generally does not drive the trellis of each component code to the all-zero state due to the existence of a feedback loop inherent in all RSCs. However, for each of the 2^M states, there exists an M bit code that will drive each encoder to the all-zero state. Unfortunately, at each point in time, the state of each component encoder is generally different due to the existence of the interleaver. Usage of a helical interleaver solves this problem by forcing both encoders to end in the same state, thus allowing a sequence of M bits to terminate both trellises [27].

The idea behind helical interleavers is as follows. If the state variables at a specified end time depend on the sum of bits found in disjoint partitions of the message stream,

then if an interleaver permutes bits within these partitions, the final encoder state remains unchanged.

Consider the following example which uses the four state RSC shown below.

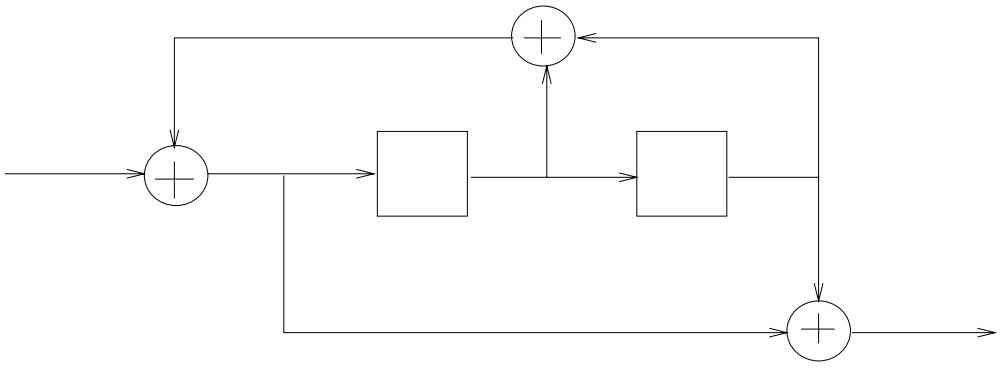

Figure A.1: Four State RSC

In addition, consider partitioning the data stream into three sequences.

$$\begin{aligned} \text{Sequence } 0 &= \{d_k | k \bmod (M+1) = 0\} \\ \text{Sequence } 1 &= \{d_k | k \bmod (M+1) = 1\} \\ \text{Sequence } 2 &= \{d_k | k \bmod (M+1) = 2\} \end{aligned}$$

The state sequence for this encoder is shown in Table A.1. If $N \bmod (M+1) = 0$, then

$$\begin{aligned} S_N^0 &= \text{Sequence } 1 + \text{Sequence } 2 \\ S_N^1 &= \text{Sequence } 0 + \text{Sequence } 1. \end{aligned}$$

If $N \bmod (M+1) = 1$, then

$$S_N^0 = \text{Sequence 2} + \text{Sequence 0}$$

$$S_N^1 = \text{Sequence 1} + \text{Sequence 2}.$$

If $N \bmod (M+1) = 2$, then

$$S_N^0 = \text{Sequence 0} + \text{Sequence 1}$$

$$S_N^1 = \text{Sequence 2} + \text{Sequence 0}.$$

$\mathbf{d_k}$	$\mathbf{S_k^0}$	$\mathbf{S_k^1}$
d_1	0	0
d_2	d_1	0
d_3	$d_1 + d_2$	d_1
d_4	$d_2 + d_3$	$d_1 + d_2$
d_5	$d_1 + d_3 + d_4$	$d_2 + d_3$
d_6	$d_1 + d_2 + d_4 + d_5$	$d_1 + d_3 + d_4$
d_7	$d_2 + d_3 + d_5 + d_6$	$d_1 + d_2 + d_4 + d_5$
d_8	$d_1 + d_3 + d_4 + d_6 + d_7$	$d_2 + d_3 + d_5 + d_6$
d_9	$d_1 + d_2 + d_4 + d_5 + d_7 + d_8$	$d_1 + d_3 + d_4 + d_6 + d_7$

Table A.1: Encoder State Sequence

As a result, any permutations within each sequence will yield the same final state. The structure for this helical interleaver with depth $M+1 = 3$ is shown in Table A.2. The output of this interleaver is

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

d_1	d_2	d_3
d_4	d_5	d_6
d_7	d_8	d_9
d_{10}	d_{11}	d_{12}
d_{13}	d_{14}	d_{15}
d_{16}	d_{17}	d_{18}
d_{19}	d_{20}	d_{21}

Table A.2: Interleaver structure

APPENDIX B

Derivation of Equations Used in the Turbo Decoder

This is a summary of the equations used in the turbo decoder. A more complete discussion can be found in [8][20].

First, the notation used in this section is described. The input to the encoder is the data sequence of length N, $\{d_k\}_{k=1}^N$. The outputs of the encoder are the data sequence, $\{d_k\}_{k=1}^N$ and the parity sequences, $\{p_{1,k}\}_{k=1}^N$ and $\{p_{2,k}\}_{k=1}^N$. Let $y_{0,k}$, $y_{1,k}$, and $y_{2,k}$ be the channel outputs corresponding to d_k, $p_{1,k}$, and $p_{2,k}$, respectively. In addition, define $R_1^N = \{R_1, ..., R_k, ...R_N\}$ where $R_k = (y_{0,k}, y_{1,k}, y_{2,k})$ and $k = 1, .., N$. If we define

$$\lambda_k^{(i)}(m) = P(d_k = i, S_k = m | R_1^N), \tag{B.1}$$

then

$$P(d_k = i|R_1^N) = \sum_m \lambda_k^{(i)}(m). \tag{B.2}$$

Thus, the *a posteriori* log likelihood ratio (LLR) is

$$\begin{aligned}
\Lambda(d_k) &= \log \frac{P(d_k = 1|R_1^N)}{P(d_k = 0|R_1^N)} && \text{(B.3)}\\
&= \log \frac{\sum_m P(d_k = 1, S_k = m|R_1^N)}{\sum_m P(d_k = 0, S_k = m|R_1^N)} && \text{(B.4)}\\
&= \log \frac{\sum_m \lambda_k^{(1)}(m)}{\sum_m \lambda_k^{(0)}(m)}. && \text{(B.5)}
\end{aligned}$$

The above equation can be computed by noting that events after time k are independent of R_1^k and d_k if S_k is known.

$$\begin{aligned}
\lambda_k^{(i)}(m) &= P(d_k = i, S_k = m|R_1^N) && \text{(B.6)}\\
&= \sum_{m'} \frac{P(d_k = i, S_k = m, S_{k-1} = m', R_1^{k-1}, R_k, R_{k+1}^N)}{P(R_1^N)} && \text{(B.7)}\\
&= \sum_{m'} \frac{P(R_{k+1}^N|d_k = i, S_k = m, S_{k-1} = m', R_1^{k-1}, R_k)}{P(R_1^N)} \cdot && \text{(B.8)}\\
&\qquad P(d_k = i, S_k = m, S_{k-1} = m', R_1^{k-1}, R_k)\\
&= \sum_{m'} \frac{P(R_{k+1}^N|S_k = m)\, P(d_k = i, S_k = m, R_k|S_{k-1} = m', R_1^{k-1})}{P(R_1^N)} \cdot && \text{(B.9)}\\
&\qquad P(S_{k-1} = m', R_1^{k-1})\\
&= \sum_{m'} \frac{P(R_{k+1}^N|S_k = m)\, P(d_k = i, S_k = m, R_k|S_{k-1} = m')}{P(R_1^N)} \cdot && \text{(B.10)}\\
&\qquad P(S_{k-1} = m'|R_1^{k-1})\, P(R_1^{k-1})
\end{aligned}$$

Combining (B.5) and (B.10),

$$\Lambda(d_k) = \log\left[\frac{\sum_m \sum_{m'} P(R_{k+1}^N|S_k=m)\,P(S_{k-1}=m'|R_1^{k-1})}{\sum_m \sum_{m'} P(R_{k+1}^N|S_k=m)\,P(S_{k-1}=m'|R_1^{k-1})}\cdot \frac{P(d_k=1,S_k=m,R_k|S_{k-1}=m')}{P(d_k=0,S_k=m,R_k|S_{k-1}=m')}\right] \quad \text{(B.11)}$$

$$= \log\frac{\sum_m \sum_{m'} \gamma_1(y_k,m',m)\alpha_{k-1}(m')\beta_k(m)}{\sum_m \sum_{m'} \gamma_0(y_k,m',m)\alpha_{k-1}(m')\beta_k(m)} \quad \text{(B.12)}$$

where

$$\alpha_k(m) = P(S_k=m|R_1^k) \quad \text{(B.13)}$$

$$\beta_k(m) = \frac{P(R_{k+1}^N|S_k=m)}{P(R_{k+1}^N|R_1^k)}. \quad \text{(B.14)}$$

Both α_k and β_k can be calculated via recursions described by [20].

$$\alpha_k(m) = \frac{\sum_{m'} \sum_{i=0}^{1} \gamma_i(y_k,m',m)\alpha_{k-1}(m')}{\sum_m \sum_{m'} \sum_{i=0}^{1} \gamma_i(y_k,m',m)\alpha_{k-1}(m')} \quad \text{(B.15)}$$

$$\beta_k(m) = \frac{\sum_{m'} \sum_{i=0}^{1} \gamma_i(y_{k+1},m',m)\beta_{k+1}(m')}{\sum_m \sum_{m'} \sum_{i=0}^{1} \gamma_i(y_{k+1},m',m)\alpha_k(m')} \quad \text{(B.16)}$$

where

$$\gamma_i(R_k,m',m) = P(d_k=i,R_k,S_k=m|S_{k-1}=m') \quad \text{(B.17)}$$

$$= p(R_k|d_k=i,S_k=m,S_{k-1}=m')\cdot \quad \text{(B.18)}$$

$$P(S_k=m|d_k=i,S_{k-1}=m')\cdot P(d_k=i|S_{k-1}=m') \quad \text{(B.19)}$$

APPENDIX C

Calculation of $p(d|i)$ for the Union Bound

In this section, the calculation of $p(d|i)$ which is essential to the computation of the union bound for turbo codes is described [26]. The state transition matrix details the input/output relationship of the encoder. A sample state transition matrix is shown below for a convolutional code with generator octals $(7, 5)$.

$$A(L, I, D) = \begin{pmatrix} L & LID & 0 & 0 \\ 0 & 0 & LD & LI \\ LID & L & 0 & 0 \\ 0 & 0 & LI & LD \end{pmatrix}$$

The rows index the states of the encoder before a transition while the columns index the states of the encoder after a transition. The monomial $L^l I^i D^d$ describes the input/output relationship for a specific transition. The variables i and d are equal to 0 or 1 depending on whether the corresponding input and output bits, respectively,

are 0 or 1 and l, which represents the length of the path is always equal to 1.

If we define $t(l,i,d)$ as the number of paths of length l, input weight i, and output weight d which start and end in the all-zero state (represented by 0^m), then we define the transfer function as follows.

$$T(L,I,D) = \sum_{l\geq 0}\sum_{i\geq 0}\sum_{d\geq 0} L^l I^i D^d t(l,i,d)$$

According to [30], $T(L,I,D)$ is the $(0^m, 0^m)$ entry in

$$(I + A(L,I,D) + A(L,I,D)^2 + A(L,I,D)^3 + ...)A(1,1,D)^m.$$

Thus,

$$T(L,I,D) = [(I-A)^{-1}A(1,1,D)^m]_{0^m,0^m}.$$

By omitting the termination factor $A(1,1,D)^m$, we can get the following approximation for the example state transition shown above.

$$T(L,I,D) \approx \frac{1 - LD - L^2D + L^3(D^2 - I^2)}{1 - L(1+D) + L^3(D + D^2 - I^2 - I^2D^3) - L^4(I^4D^2 - I^2D^4 + D^2 - I^2)}$$

This yields the recursion

$$\begin{aligned} t(l,i,d) &= t(l-1,i,d-1) + t(l-1,i,d) + \\ & t(l-3,i-2,d-3) + t(l-3,i-2,d) - t(l-3,i,d-2) - \\ & t(l-3,i,d-1) + t(l-4,i-4,d-2) - t(l-4,i-2,d-4) - \\ & t(l-4,i-2,d) + t(l-4,i,d-2) + \delta(l,i,d) - \delta(l-1,i,d-1) - \\ & \delta(l-2,i,d-1) + \delta(l-3,i,d-2) - \delta(l-3,i-2,d). \end{aligned}$$

By assuming a uniform interleaver, we can compute $p(d|i)$ for each code fragment.

$$p(d|i) = \frac{t(N,i,d)}{\sum_{d'} t(N,i,d')} = \frac{t(N,i,d)}{\begin{pmatrix} N \\ i \end{pmatrix}}$$

Finally, we can calculate $p(d|i)$ for the entire code by using

$$p(d = d_0 + d_1 + d_2|i) = p_0(d_0|i)p_1(d_1|i)p_2(d_2|i).$$

APPENDIX D

Reduced Complexity Decoders

One area of interest for future work is to consider reduced complexity decoders. Decoding complexity is one of the major disadvantages for turbo codes. Complexity is proportional to the block length, the number of decoding iterations, and the constraint length of the convolutional codes. Each MAP decoder has the approximate complexity of the Viterbi algorithm and this must be iterated several times. Thus, it is not surprising that some of the current research is focused on reducing the computational complexity of turbo decoders [11][13][14]. In this section, I will discuss the basis behind four reduced complexity decoders: Max-Log-Map, SOVA, Log-MAP, and Sliding Window BCJR.

D.1 Max-Log-MAP

Using the MAP algorithm to compute log likelihood ratios requires the calculation of the branch transition probability, γ, the forward recursion, α, and the backward

recursion, β (see Appendix B). Consider the natural logarithm of $\alpha_k(m)$.

$$\begin{aligned} \ln \alpha_k(S_k) &= \ln\Big(\sum_{S_{k-1}} \sum_i e^{\ln \gamma_i(y_k, S_k, S_{k-1}) + \ln \alpha_{k-1}(S_{k-1})}\Big) - \\ &\quad \ln\Big(\sum_{S_k} \sum_{S_{k-1}} \sum_i e^{\ln \gamma_i(y_k, S_k, S_{k-1}) + \ln \alpha_{k-1}(S_{k-1})}\Big) \end{aligned} \tag{D.1}$$

The calculation of α_k (and hence β_k and the log likelihood ratio, Λ_k) could be greatly simplified if the following approximation was used.

$$\ln(e^{\delta_1} + e^{\delta_2} + ... + e^{\delta_n}) \approx \max_{1 \le i \le n} \delta_i \tag{D.2}$$

$$\tag{D.3}$$

This approximation can be applied to the forward and backward recursions, $\alpha_k(S_k)$ and $\beta_k(S_k)$, and the log likelihood ratio, Λ_k, where $\hat{\alpha}_k(S_k) = \ln \alpha_k(S_k)$, $\hat{\beta}_k(S_k) = \ln \beta_k(S_k)$, and $\hat{\gamma}_i(y_k, S_k, S_{k-1}) = \ln \gamma_i(y_k, S_k, S_{k-1})$.

$$\begin{aligned} \hat{\alpha}_k(S_k) &\approx \max_{S_{k-1}, i}(\hat{\gamma}_i(y_k, S_k, S_{k-1}) + \hat{\alpha}_{k-1}(S_{k-1})) - \\ &\quad \max_{S_k, S_{k-1}, i}(\hat{\gamma}_i(y_k, S_k, S_{k-1}) + \hat{\alpha}_{k-1}(S_{k-1})) \end{aligned} \tag{D.4}$$

$$\begin{aligned} \hat{\beta}_k(S_k) &\approx \max_{S_{k+1}, i}(\hat{\gamma}_i(y_k, S_k, S_{k+1}) + \hat{\beta}_{k+1}(S_{k+1})) - \\ &\quad \max_{S_k, S_{k+1}, i}(\hat{\gamma}_i(y_k, S_k, S_{k+1}) + \hat{\alpha}_k(S_k)) \end{aligned} \tag{D.5}$$

$$\begin{aligned} \Lambda_k &\approx \max_{S_k, S_{k-1}} (\hat{\gamma}_1(y_k, S_k, S_{k-1}) + \hat{\alpha}_{k-1}(S_{k-1}) + \hat{\beta}_k(S_k)) - \\ &\quad \max_{S_k, S_{k-1}} (\hat{\gamma}_0(y_k, S_k, S_{k-1}) + \hat{\alpha}_{k-1}(S_{k-1}) + \hat{\beta}_k(S_k)) \end{aligned} \tag{D.6}$$

Despite significantly reducing the decoding complexity, this method yielded a

performance loss less than 0.5 dB with respect to the original MAP algorithm for packet length 1024 information bits, 8 decoding iterations, and component encoders with memory 4 over the AWGN channel [11].

D.2 Log-MAP

This method improves the performance of Max-Log-MAP by adding a correction term to the approximation used above. This can be accomplished by using the Jacobian logarithm [11].

$$\ln(e^{\delta_1} + e^{\delta_2}) = \max(\delta_1, \delta_2) + \ln(1 + e^{-|\delta_2 - \delta_1|}) \quad \text{(D.7)}$$

$$= \max(\delta_1, \delta_2) + f_c(|\delta_2 - \delta_1|) \quad \text{(D.8)}$$

In the above equation, f_c can be approximated (quantized) by using a pre-computed lookup table. While this may result in small performance losses, the advantage is that less storage is needed. The performance of this decoder is virtually identical to the MAP decoder.

D.3 SOVA

This method is similar to Max-Log-MAP, but only considers survivor paths (i.e. for a path to be considered, the competing path must join the path chosen by the Viterbi algorithm without being eliminated). Thus competing paths may not be the best ones.

D.4 Sliding Window BCJR

This method uses the original Bahl equations, but instead operates on a fixed memory span. Thus, decisions are forced with a given delay.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] C. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 379-423 and 623-656, July and October, 1948.

[2] J. Kang and W. Stark, "Turbo Codes for Coherent FH-SS with Partial-Band Interference," *Proc. MILCOM '97*, Nov. 1997.

[3] J. Kang and W. Stark, "Performance of Turbo-Coded FH-SS with Partial Band Interference," *Proc. ISIT '98*, August 1998.

[4] J. Kang and W. Stark, "Performance of Turbo-Coded FH-SS with Partial-Band Interference and Rayleigh Fading," *Proc. of the 1998 IEEE Military Communications Conference*, October 1998.

[5] J. Kang and W. Stark, "Turbo Codes for Noncoherent FH-SS With Partial Band Interference," *IEEE Trans. Communications*, November 1998.

[6] J. Kang, W. Stark, and A. Hero, "Turbo Codes for Fading and Burst Channels," *IEEE Conference Theory Mini Conference*, November 1998.

[7] J. Kang and W. Stark, "Iterative Estimation and Decoding for FH-SS with Slow Rayleigh Fading," Submitted to *IEEE Trans. Communications*, March 1999.

[8] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding," *Proc. ICC '93*, May 1993.

[9] C. Berrou, "Some Clinical Aspects of Turbo Codes," *Int. Symp. on Turbo Codes*, Sep. 1997.

[10] T. Cover and J. Thomas, *Elements of Information Theory*, Wiley-Interscience, 1991.

[11] P. Robertson, E. Villebrun, and P. Hoeher, "A Comparison of Optimal and Sub-optimal MAP Decoding Algorithms Operating in the Log Domain," *Proc. ICC '95*, June 1995.

[12] S. Benedetto and G. Montorsi, "Design of Parallel Concatenated Convolutional Codes," *IEEE Trans. Communications*, vol. 44, May 1996.

[13] C. Berrou, P. Adde, E. Angui, and S. Faudeil, "A Low Complexity Soft-Output Viterbi Decoder Architecture," *Proc. ICC '93*, May 1993.

[14] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Algorithm for Continuous Decoding of Turbo Codes," *Electronics Letters*, vol. 32, no. 4, Feb. 15, 1996.

[15] H. Koorapaty, Y. Wang, and K. Balachandran, "Performance of Turbo Codes with Short Frame Sizes," VTC '97, May 1997.

[16] C. Frank and M. Pursley, "Concatenated Coding for Frequency Hop Spread Spectrum with Partial-Band Interference," *IEEE Trans. Communications*, vol. 44, no. 3, March 1996.

[17] J. Mathis *et al*, "Final Design Plan: Sincgars Packet Switch Overlay,", *SRI Technical Report*, SRI Project no. 1244, July 1986.

[18] M. Pursley and W. Stark, "Performance of Reed-Solomon Coded Frequency-hop Spread-spectrum Communications in Partial-Band Interference," *IEEE Trans. Communications*, August 1985.

[19] U. Fiebig and P. Robertson, "Soft Decision Decoding in Fast Frequency Hopping Systems with Convolutional Codes and Turbo Codes," *Proc. ICC '96*, June 1996.

[20] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Trans. Information Theory*, vol. 20, March 1974.

[21] P. Robertson, "Illuminating the Structure of Code and Decoder for Parallel Concatenated Recursive Systematic (Turbo) Codes," *Proc. GLOBECOM'94*, Dec. 1994.

[22] W. Stark, "Coding for Frequency-Hopped Spread-spectrum Communications with Partial-band Interference - Part I: Capacity and cutoff Rate," *IEEE Trans. Communications*, Oct. 1985.

[23] W. Stark, "Coding for Frequency-hopped Spread-spectrum Communications with Partial-band interference - Part II: Coded Performance," *IEEE Trans. Communications*, Oct. 1985.

[24] E. Geraniotis and M. Pursley, "Error Probabilities for Slow Frequency-hopped Spread Spectrum Multiple-access Communications over Fading Channels," *IEEE Trans. Communications*, May 1982.

[25] D. Divsalar and F. Pollara, "Turbo Codes for Deep-Space Communications," *Telecom. and Data Aquisition Progress Report 42-120*, Jet Propulsion Laboratory, February 1995.

[26] D. Divsalar, S. Dolinar, F. Pollara, and R. McEliece, "Transfer Function Bounds on the Performance of Turbo Codes," *TDA Progress Report 42-122*, JPL, August 1995.

[27] A. Barbulescu and S. Pietrobon, "Terminating the Trellis of Turbo Codes in the Same State," *Electronics Letters*, vol. 31, Nov. 1995.

[28] D. Pauluzzi and N. Beaulieu, "A Comparison of SNR Estimation Techniques in the AWGN Channel," *Proc. of Pacific Rim Conference on Communications, Computers, and Signal Processing*, 1995.

[29] E. Berlekamp and P. Tong, "Improved Interleavers for Digital Communications," US Patent Number 4559625, Dec. 17, 1985.

[30] R. Stanley, *Enumerative Combinatorics*, Monterey, California, 1986.

[31] M. Simon, J. Omura, R. Scholtz, and B. Levitt, *Spread Spectrum Communications*, Computer Science Press, 1985.

[32] J. Proakis, *Digital Communications*, McGraw-Hill, 1995.

[33] S. Lin and D. Costello, *Error Control Coding: Fundamentals and Applications*, Prentice Hall, 1983.

[34] L. Rabiner and R. Schafer, *Digital Processing of Speech Signals*, Prentice Hall, 1978.

[35] E. Hall and S. Wilson, "Design and Analysis of Turbo Codes on Rayleigh Fading Channels," *IEEE Trans. Communications*, vol. 16, no. 2, February 1998, pp. 160-174.

[36] J. Hagenauer, "Viterbi Decoding of Convolutional Codes for Fading and Burst Channels," *Proc. Int. Zurich Seminar*, 1980.

[37] L. Biederman, J. Omura, and P. Jain, "Decoding with Approximate Channel Statistics for Band-Limited Nonlinear Satellite Channels," *IEEE Trans. Information Theory*, vol. IT-27, no. 6, November, 1981.

[38] D. Cox, "910 MHz Urban Mobile Radio Propagation: Multipath Characteristics in New York City," *IEEE Trans. Communications*, Nov. 1973.

[39] D. Cox, "Delay Doppler Characteristics of Multipath Propagation at 910 MHz in a Suburban Mobile Radio Environment," *IEEE Trans. Antennas Propagat*, Sept. 1972.

[40] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial Concatenation of Interleaved Codes: Performance Analysis, Design, and Iterative Decoding," *TDA Progress Report 42-126*, JPL, August 1996.

[41] T. Summers and S. Wilson, "SNR Mismatch and Online Estimation in Turbo Decoding," *IEEE Trans. Communications*, April 1998.

[42] L. Miller, J. Lee, and A. Kadrichu, "Probability of Error Analyses of a BFSK Frequency-Hopping System with Diversity Under Partial-Band Jamming Interference–Part III: Performance of a Square-Law Self-Normalizing Soft Decision Receiver," *IEEE Trans. Communications*, July 1986.

[43] K. Cheun and W. Stark, "Performance of Robust Metrics with Convolutional Coding and Diversity in FHSS Systems under Partial-Band Noise Jamming," *IEEE Trans. Communications*, January 1993.

[44] N. Lebedev, *Special Functions and Their Applications*, 1972.