Codes and Iterative Receivers for Wireless Communication Systems

Andrew Porter Worthen

COMMUNICATIONS & SIGNAL PROCESSING LABORATORY
Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, MI 48109-2122

May, 2001

Technical Report No. 327

ABSTRACT

Codes and Iterative Receivers for Wireless Communication Systems

by

Andrew Porter Worthen

Chair: Wayne E. Stark

Future wireless communication systems will need sophisticated techniques to operate reliably under typical radio propagation and interference conditions. Emerging multi-media applications demand high data rates with low delay, and limited battery capacity makes it essential to communicate extremely efficiently. Optimum receivers for unknown, time-varying channels encountered by wireless communication systems must jointly estimate the channel properties and decode the transmitted data. This is typically too computationally complex to implement. We propose a unified framework based on factor graphs for designing iterative receivers which approximate joint channel estimation and decoding using iterative algorithms. Once the appropriate graphical model for the receiver has been constructed, deriving the details of the algorithm is quite straightforward. In order to handle the problems introduced by continuous variables, we suggest canonical distributions, a general approach to simplifying the algorithms that allows their key features to be retained. Several receiver design examples illustrate our approach for common channel models.

Given the possibility of approximating joint channel estimation and decoding using iterative receivers, we would like to estimate the potential performance of the techniques in various situations. For channels with block memory, which are relatively easy to analyze and model a variety of frequency-hopping spread spectrum systems, we obtain bounds based on the channel reliability function, or error exponent. Both theoretical and numerical results confirm that the bounds behave qualitatively very much like iterative receivers with low-density parity check codes. Our results provide insight into the behavior of high-performance systems that use iterative receivers, especially considering the effects of channel memory on performance.

To Julie Johnston and my parents, Richard and Martha Worthen.

ACKNOWLEDGEMENTS

It is always difficult to remember to thank everyone. At risk of leaving out many people who have contributed to my success, I would like to acknowledge a few of my benefactors.

I begin with my Ph.D. advisor, Prof. Wayne Stark. I have benefited tremendously from his laid-back advice and patience. His way of thinking about problems has been essential to developing the ideas in the dissertation.

I would like to thank my doctoral committee, Profs. Achilleas Anastasopoulos, Kevin Compton, and Kim Winick, for their careful reading of the dissertation and helpful comments. Prof. Compton's suggestions have greatly simplified some of the proofs in Chapter 5. I am also grateful to Prof. Demosthenis Teneketzis, who served on my thesis proposal committee, for being available whenever I needed advice.

My colleagues in the wireless communication lab, instructors, classmates, and students have all been fundamental to my graduate school experience. I would especially like to thank them for listening to my sometimes lengthy ideas.

A National Science Foundation Graduate Research Fellowship and an Armed Forces Communication and Electronics Association (AFCEA) Fellowship allowed me the flexibility to pursue topics that I believe are of both theoretical and practical importance. I learned many valuable lessons because of this freedom. Additional support, especially computing resources, came from a Department of Defense Research & Engineering (DDR&E) Multidisciplinary University Research Initiative (MURI) on "Low-Energy Electronics Design for Mobile Platforms" managed by the Army Research Office (ARO) under grant DAAH04-96-1-0377.

Finally, I would like to thank my family, especially my parents, Richard and Martha Worthen, and my fiancé, Julie Johnston, for their unwavering support.

TABLE OF CONTENTS

DEDICATION ii

ACKNOWLEDGEMENTS iii

LIST OF FIGURES vi

LIST OF APPENDICES viii

CHAPTERS

1 Wireless Communication: Unknown, Time-Varying Channels and Techniques 1
- 1.1 The Communication Problem 2
- 1.2 High Performance Codes 5
- 1.3 Wireless Communication Channels 8
- 1.4 Conventional Wisdom – Eliminate Memory 11
- 1.5 Optimum Receivers: Joint Channel Estimation and Decoding 13
- 1.6 Effective Diversity vs. Channel Estimation 13
- 1.7 Overview 14

2 Factor Graphs and the Sum-Product Algorithm 16
- 2.1 Factor Graphs 17
- 2.2 The Sum-Product Algorithm 19
- 2.3 Low-Density Parity Check Codes 22
- 2.4 Efficient Updates and Complexity 24

3 Unified Design of Iterative Receivers 26
- 3.1 Factoring the Objective Function $G(\underline{x}, \underline{u})$ 27
- 3.2 Canonical Distributions 33
- 3.3 Discussion 35

4 Case Studies in Receiver Design 39
- 4.1 Two-State Jamming Channel with Block Memory 39
  - 4.1.1 Receiver Design 40
  - 4.1.2 Simulation Results 42

4.2 Two-State Jamming Channel with Markov Memory 46
4.2.1 Receiver Design 48
4.2.2 Simulation Results 49
4.3 Coherent Rayleigh Fading Channel with Block Memory . . . 51
4.3.1 Receiver Design 51
4.3.2 Simulation Results 54
4.4 Non-Coherent Flat Rayleigh Fading Channel with Block Memory 57
4.4.1 Receiver Design 57
4.4.2 Simulation Results 65
4.5 General Discussion 67

5 Coding Theory for Channels with Block Memory 69
5.1 General Bounds and Properties 71
5.1.1 Channel Classes 73
5.1.2 Properties of the Reliability Function 75
5.1.3 An Alternate Approach to the Optimal Hop Length . 87
5.2 Approaches for Wireless Channels 88
5.2.1 Coherent Jamming and Fading Channels 90
5.2.2 Non-Coherent Fading Channels 97

6 Iterative Reception: Solutions and Challenges 104

APPENDICES 106

BIBLIOGRAPHY 114

LIST OF FIGURES

Figure

1.1 System model 2
1.2 Turbo code encoder 6
1.3 Turbo code decoder 7
2.1 Example factor graph 18
2.2 Factor graph for the example low-density parity check code 23
2.3 Factor graph for an LDPC code on a memoryless channel 24
3.1 Factor graph fragment from (3.7) for a channel with no ISI 28
3.2 Factor graph fragments for a channel with ISI 29
3.3 Factor graph fragment from (3.14) for a channel with block memory . 30
3.4 Factor graph fragment from (3.17) for a channel with Markov memory 31
3.5 Factor graph fragment from (3.19) for a channel with 2nd order Markov memory 32
3.6 Factor graph fragment from (3.20) for a channel with 2nd order Markov memory 32
3.7 Factor graph fragment for a turbo code 33
3.8 Factor graph for the multi-path fading channel 37
3.9 Factor graph for the multi-path fading channel using output estimates instead of coefficient estimates 38
4.1 Factor graph for the two-state jamming channel with block memory . 41
4.2 Factor graph for alternate receivers for the two-state jamming channel 43
4.3 Receiver performance comparison for the two-state jamming channel . 43
4.4 Required SJNR E_b/N_J for the two-state jamming channel 44
4.5 Channel estimation error dynamics for the two-state jamming channel 45
4.6 Bit errors vs. iterations for the two-state jamming channel 46
4.7 Factor graph for the two-state jamming channel with Markov memory 48
4.8 Performance vs. memory μ for the two-state jamming channel with Markov memory 49
4.9 Performance vs. memory length $2/(1-\mu)$ or m for two-state jamming channels 50
4.10 Performance vs. hop length for coherent Rayleigh fading 55

4.11 Channel estimation error for coherent Rayleigh fading 56
4.12 Factor graph for the non-coherent Rayleigh fading channel with block memory 59
4.13 Comparison of canonical distributions for the phase message 62
4.14 Comparison of receivers for the non-coherent Rayleigh fading channel 66
4.15 Performance vs. hop length for the non-coherent Rayleigh fading channel 66
5.1 Sketch of $E_r(R, m)$ vs. R for fixed m 73
5.2 Example diversity dominated channel 74
5.3 Sketch illustrating Theorem 5.3 82
5.4 Reliability function $E_r(R, m)$ vs. rate R for the two-state jamming channel 91
5.5 Reliability function $E_r(R, m)$ vs. SJNR E_b/N_J for two-state jamming channels 92
5.6 Required SJNR E_b/N_J vs. hop length for the two-state jamming channel 93
5.7 Performance from bounds and simulation for the two-state jamming channel 94
5.8 Objective function for partial-band jamming game 95
5.9 Required bit energy E_b vs. hop length for the two-state jamming channel 95
5.10 Reliability function $E_r(R, m)$ vs. average SNR E_b/N_0 for the coherent Rayleigh fading channel 96
5.11 Required average SNR E_b/N_0 for the coherent Rayleigh fading channel 97
5.12 Reliability function $E_r(R, m)$ vs. rate R for the flip channel 98
5.13 Required SNR E_b/N_0 for the flip channel 99
5.14 Required average SNR E_b/N_0 for the Rayleigh fading flip channel . . 102
A.1 Factor graph for two-state jamming channel with block memory . . . 108
A.2 Factor graph for two-state jamming channel with Markov memory . . 109
A.3 Factor graph for the Rayleigh fading channel with block memory . . . 110
A.4 Factor graph for the non-coherent Rayleigh fading channel with block memory 111

LIST OF APPENDICES

APPENDIX

A Update Formulas for Example Receivers 107

A.1 Two-State Jamming Channel with Block Memory 107

A.2 Two-State Jamming Channel with Markov Memory 108

A.3 Coherent Rayleigh Fading Channel with Block Memory . . . 109

A.4 Non-Coherent Rayleigh Fading Channel with Block Memory 111

CHAPTER 1

Wireless Communication: Unknown, Time-Varying Channels and Techniques

Future wireless communication systems will require substantial improvements over current technologies. Emerging multi-media applications demand data rates at least ten times larger than current voice networks, and still have strict delay requirements. Limited battery capacity and the ever increasing number of users mean that communication must be extremely efficient with respect to both power consumption and bandwidth. One way to get the best performance from a given physical channel is to use as much of the channel structure as possible. Simple, elegant approaches that capture the fundamental features of the wireless channel will be essential for creating ubiquitous connectivity. To realize the next generation of high-performance wireless devices, we will need effective, flexible design techniques for iterative algorithms and convenient analytical tools for system modeling. Our work addresses both of these problems by proposing a unified framework for designing receivers for channels with memory, and demonstrating that a well-known bounding technique can be applied to examine the performance of these systems.

This chapter begins with a very brief survey of relevant background material from digital communication and coding theory. An excellent introductory reference is [55]. We then describe some specific models of interest, motivate our contributions, and outline some core themes.

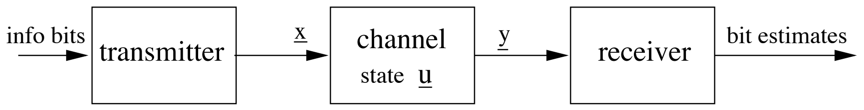


Figure 1.1: System model

1.1 The Communication Problem

The fundamental problem of communication theory is to transmit a message, usually in binary form, from one place to another over a channel which corrupts the message in various ways. We model a communication system as consisting of three components as shown in Fig. 1.1. We design the transmitter and receiver, and the channel is dictated by the application. The transmitter encodes a block of K information bits into a block of N encoded symbols $\underline{x}$ from the channel input alphabet $\mathcal{X}$. The channel performs some random operations on $\underline{x}$ to produce the channel output sequence $\underline{y}$, taken from the channel output alphabet $\mathcal{Y}$ and related to $\underline{x}$ according to the channel transition probability density function $\mathrm{p}(\underline{y}|\underline{x})$. The transition probability function completely defines the channel. The receiver's task is to reconstruct the message with the smallest possible probability of error given only the channel output $\underline{y}$ and knowledge of the system model.

The transmitter uses an error control code to introduce structured redundancy into the data which will allow the receiver to recover the message from the corrupted channel output. Formally, an error control code is a set $\mathcal{C} \subset \mathcal{X}^N$ of 2^K vectors called codewords selected from the possible channel input sequences of length N. The transmitter uses a one-to-one mapping called the encoder between the messages and the codewords to select the vector $\underline{x} \in \mathcal{C}$ to transmit across the channel. If the encoder is such that each codeword contains its corresponding K information bits in some fixed set of positions, then we say that the encoder is systematic. The code rate R, a measure of transmission efficiency, is K/N bits/channel use. The weight, or Hamming weight, of a codeword is equal to the number of non-zero elements in the codeword. The Hamming distance between two codewords is the Hamming weight of their element-by-element difference.

The receiver must estimate the transmitted message, which is equivalent, given knowledge of the encoder, to determining the transmitted codeword with minimum probability of making an error. It can be shown that the optimal decision rule is to maximize the *a posteriori* probability of the transmitted vector

$$\hat{\underline{x}} = \arg\max_{\underline{x}} \mathrm{p}(\underline{x}|\underline{y}) \tag{1.1}$$

where $\hat{\underline{x}}$ is the estimate of the transmitted codeword and $\underline{y}$ is the observed channel output. This is the maximum *a posteriori* (MAP) decoding rule. Using Bayes' rule we can change this to

$$\hat{\underline{x}} = \arg\max_{\underline{x}} \mathrm{p}(\underline{y}|\underline{x})\,\mathrm{p}(\underline{x}) \tag{1.2}$$

where the normalizing factor $1/\,\mathrm{p}(\underline{y})$ has been dropped because it does not affect the maximization. Note that this depends on the prior probability of the channel input vectors $\underline{x}$ which we often take to be uniform over the codewords. This gives the maximum likelihood (ML) rule

$$\hat{\underline{x}} = \arg\max_{\underline{x}\in\mathcal{C}} \mathrm{p}(\underline{y}|\underline{x}) \tag{1.3}$$

which is often used for convenience even when the codewords are not known to be equally likely. Note that for more than 30 or 40 information bits, the number of possible codewords 2^K is extremely large so a brute force implementation of the maximum likelihood decoder is rarely possible. The probability of error is the probability that the codeword selected by the receiver is not the codeword that was transmitted. We can also consider failures, where the decoder is unable to select a viable codeword. Typically, failures are preferable to errors because then the receiver can take appropriate action such as requesting a retransmission of the corrupted data.

We could also minimize the probability that a bit in the received message is in error, instead of minimizing the message error probability. In this case, the optimal decision rule for the i^{th} bit x_i is

$$\begin{aligned} \hat{x}_i &= \arg\max_{x_i} \mathrm{p}(x_i|\underline{y}) \\ &= \arg\max_{\xi} \sum_{\underline{x}:x_i=\xi} \mathrm{p}(\underline{x}|\underline{y}) \end{aligned} \tag{1.4}$$

This symbol-by-symbol MAP rule is slightly different from the sequence MAP rule described above and, of course, has better performance when the bit error rate is considered. It may not have a smaller probability of block error than the sequence MAP rule because the sequence MAP rule achieves the minimum possible block error rate.

An important class of codes are linear block codes, which are defined by

$$\underline{x} \in \mathcal{C} \Leftrightarrow \mathbf{H}\,\underline{x} = 0 \tag{1.5}$$

where $\mathbf{H}$ is a matrix called the parity check matrix of the code, and the operations are carried out over the appropriate finite field. For binary codes, all operations are performed modulo 2. All linear block codes contain the all zeros word and have the property that the sum of any two codewords is also a codeword. Although these codes have significant structure, efficient decoding algorithms only are known for a relatively small number of specific constructions. The rate of a linear block code is equal to the rank of the parity check matrix divided by the block length of the code. Thus, if the parity check matrix has full rank, its size is $N - K$ rows by N columns.

Thus, the central problem of communication theory is reduced to building a code $\mathcal{C}$ that achieves very small error probabilities using an implementable receiver. In 1948, Claude Shannon showed that there exist codes such that for rates less than the channel capacity C reliable communication is possible, i.e. for sufficiently long codewords, the probability of error for an optimal receiver can be made as small as desired [58]. Clearly, by simply transmitting the same message over and over again, we can achieve reliable communication, but only in the limit as the rate goes to 0. The surprise is that C is usually much greater than 0. Unfortunately, Shannon's proof provides little insight into how to find codes that operate close to the capacity using implementable receivers. We can also use the bound to find the worst channel conditions that allow reliable communication at a particular transmission rate.

An important channel model is the discrete-time additive white Gaussian noise (AWGN) channel. The output values y_i are related to the input values x_i according to

$$y_i = x_i + n_i \tag{1.6}$$

where n_i is a sequence of independent Gaussian random variables with mean 0 and variance $N_0/2$. We often characterize this channel by its signal to noise ratio (SNR). The signal energy per symbol is $E_s = E\{x_i^2\}$. However, in order to fairly compare systems operating at different rates R in bits per channel use, we will define the SNR as E_b/N_0 where $E_b = E_s/R$ is the energy per information bit. It is important to note the distinction between the SNR E_b/N_0, and the "channel SNR" E_s/N_0. The SNR is usually expressed in decibels (dB), $10 \log_{10} E_b/N_0$. We will often compare performance in terms of the difference in SNR, expressed in dB, required for systems achieving the same error probability. Almost all channels have an AWGN component because of thermal noise in the electronics at the front-end of the receiver.

1.2 High Performance Codes

The best practical codes known for large block lengths are turbo codes and low-density parity check (LDPC) codes. The receiver algorithms described later are applicable to both of these. Turbo codes, first described in [10], consist of a concatenation, or combination, of two recursive systematic convolutional codes (RSCC's), called the component codes. The encoder, shown in Fig. 1.2, takes a block of information bits and produces three blocks of encoded bits. The first of these is a copy of the information bits, the systematic part. The second is a collection of parity check bits which are generated by the first recursive convolutional encoder. The third block is the sequence of parity check bits generated by the second recursive convolutional encoder, which operates on an interleaved copy of the data bits. The interleaver, labeled π in the figure, simply reorders the bits so that the second component encoder sees the same data but in a different order. The recursive convolutional encoders contain shift registers where modified data is shifted in from the left and a linear combination of the shift register elements is produced as the output. The encoders are recursive because the data is combined with elements from the shift register before being shifted in. Note that all additions are modulo 2. Turbo codes, as described here, are often called parallel concatenated convolutional codes (PCCC's) where "parallel" refers to the fact that the encoders operate on the same data, instead of one operating on

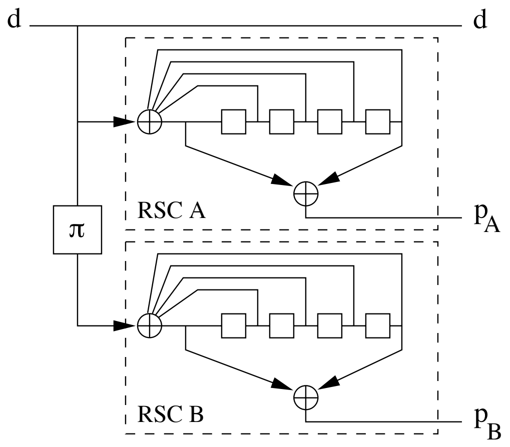


Figure 1.2: Turbo code encoder. d is the information sequence, p_A is the sequence of parity checks from the first component encoder, and p_B is the sequence of parity checks from the second component encoder.

the other's output. Numerous variations including serial concatenated convolutional codes (SCCC's), where the second encoder does operate on the first encoder's output, have been proposed. A good survey of turbo codes is given in [26].

A key feature of turbo codes is the existence of an effective, sub-optimal iterative decoding algorithm. The decoder block diagram is shown in Fig. 1.3. The data and the parity check bits from the first component encoder are decoded considering just the first component code. Estimated bit likelihoods for the information bits are then interleaved and used as prior probabilities by the second decoder, which uses the interleaved received data and the parity check bits from the second component encoder to decode the second component code. The estimated bit likelihoods from this second decoder are then deinterleaved (to get them back in the original order) and passed back to the first decoder as new prior probabilities. The process is repeated several times and the final information bit decisions are taken from the second component code. The component decoders are based on the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [9] for bit-wise MAP decoding of convolutional codes. The decoder must be slightly modified so that the incoming prior probabilities are not included in the outgoing bit likelihoods; otherwise, the resulting positive feedback makes the algorithm ineffective. Although no comprehensive analysis of the turbo decoding algorithm is available, numerous simulation studies have shown that it gives excellent

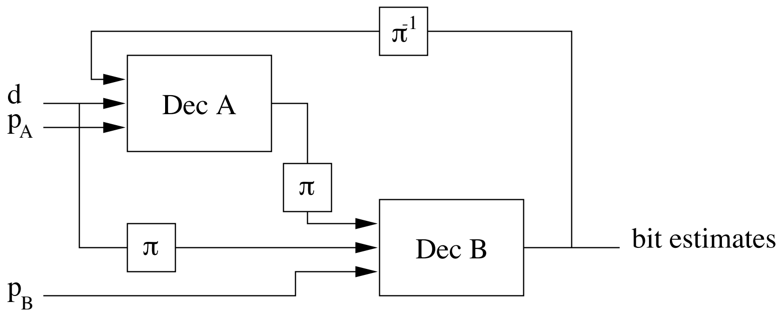


Figure 1.3: Turbo code decoder. d is the sequence of channel outputs corresponding to the systematic bits, p_A corresponds to the parity checks from encoder A, and p_B corresponds to the parity checks from encoder B. π matches the encoder interleaver, while π^{-1} is the deinterleaver for π.

performance.

Low-density parity check codes, invented by Gallager [17, 18] in 1962, are linear block codes with very sparse parity check matrices. Each column or row of the parity check matrix contains at most a small fixed number of ones, regardless of the block length of the code. An LDPC code is regular if the Hamming weights of its rows are the same and the Hamming weights of its columns are the same, i.e. if it has the same number of non-zero elements in each row and the same number of non-zero elements in each column. Irregular LDPC codes may have varying row and column weights, but the weights are still bounded independent of the block length. Gallager proposed an approximate iterative decoding algorithm based on the idea that each row is essentially an even parity check code, which, because of the low row weight, involves a small number of bits. Because this algorithm is best described using graphical models, we will delay the details until Chapter 2.

Probably due to computational complexity concerns, LDPC codes received very little attention until the mid-1990's, when MacKay and Neal [42] found that for long block lengths, on the order of 1000 or more, LDPC codes have extraordinarily good performance, very close to that of turbo codes. Expander codes [59] are a class of low-density parity check codes with provably good performance for a simple iterative decoding algorithm. Recently developed asymptotic analysis techniques [57] for iterative decoding of LDPC codes have yielded irregular LDPC codes with performance very close the channel capacity. For the erasure channel, code constructions and linear complexity decoders are known that operate reliably arbitrarily close to the channel capacity for asymptotically large block lengths [41]. On the AWGN channel, codes and decoders have been found that operate within 0.1 dB of the capacity bound [56].

1.3 Wireless Communication Channels

We will be most interested in radio-frequency communication channels. In order to understand the phenomena encountered in these channels, we need to delve a bit more deeply into how the discrete-time channels that we introduced above relate to real signals. A modulator is used to convert the discrete-time symbols into real continuous-time signals that will propagate well as electromagnetic waves. A demodulator converts the received electromagnetic signals back into discrete-time samples, which may be continuous or discrete-valued. Under some conditions, this process can done without losing any useful information. Typically, the modulator takes T seconds to transmit each symbol in sequence, where T is called the symbol duration and the time slot used is called the symbol interval. For discrete symbol alphabets, the complex baseband signal in each symbol interval $x(t)$ is selected from a collection of approximately low-pass, complex-valued, continuous-time signals $s_i(t)$, defined on the interval $t \in [0, T)$. Here, we almost always consider binary phase shift keying (BPSK) where there are two signals $g(t)$ and $-g(t)$ and we take $g(t)$ to be real. A common choice is the rectangular pulse, $g(t) = 1$ for $0 \leq t < T$ and $g(t) = 0$. Usually, we map a bit $x_i \in \{0, 1\}$ to the signal set $\{g(t), -g(t)\}$ as $x_i \to (-1)^{x_i} g(t)$. Because low-pass signals do not propagate well in space, and in order to make more efficient use of the radio spectrum, we shift our baseband signal in frequency, which produces an approximately band-pass signal

$$x'(t) = \Re\{x(t)e^{j2\pi f_c t}\} \tag{1.7}$$

where f_c is the carrier frequency in hertz and $\Re\{\cdot\}$ indicates that we take the real part. At the demodulator, we receive some corrupted version $y'(t)$ of the transmitted bandpass signal. We convert it back to baseband by

$$y(t) = y'(t)e^{-j2\pi f_c t} - \text{double frequency terms}, \tag{1.8}$$

pass $y(t)$ through a filter matched to the transmitted pulse $g(t)$, and sample its output at the end of each symbol interval to recover the discrete-time channel output. Variations on this scheme are required for some situations, but the basic idea of obtaining a discrete-time decision statistic from the continuous received signal still applies.

An important issue in the demodulation process is how precisely we can synchronize the receiver. The signal is delayed during transmission by some unknown time, which results in an unknown phase factor $e^{j\theta}$ appearing in the baseband signal along with some uncertainty in the timing of the symbol intervals. The symbol timing can usually be recovered without difficulty. Recovery of the phase depends largely on the phase offset being stable for long periods of time. If we assume that the phase offset is exactly known, then the channel is coherent. Otherwise, the channel is non-coherent, and typically we assume that the phase offset θ is uniformly distributed over the interval $[0, 2\pi)$. Channel effects particular to wireless channels often make phase synchronization difficult.

The most important effect in wireless communication channels is fading due to multi-path propagation. When the transmitted signal reflects off of various objects in the environment, the reflections arrive with different delays. If the delays are multiples of the period of the carrier signal, then the signals add constructively, otherwise they may interfere destructively, drastically reducing the received signal energy. Different carrier frequencies traveling over the same paths experience different attenuations because their periods are different. The largest difference between the delays of different received signals is called the delay spread. If the delay spread is very small with respect to the symbol duration, then each symbol remains mostly confined to its own interval and we say that the channel is frequency non-selective. Otherwise, inter-symbol interference (ISI) occurs and we say that the channel is frequency-selective because the fading amplitude will depend significantly on frequency even within the bandwidth of the signal. When there is no dominant line-of-sight path, the fading amplitude is distributed approximately according to the Rayleigh distribution, and the fading is referred to as Rayleigh fading. As the environment changes, especially when the transmitter or receiver is moving, the fading amplitude also changes. This unknown time-varying path gain or transfer function (for frequency-selective fading) is one of the key challenges of wireless communication.

Fading is bad because when the channel is in a deep fade the received SNR is very low, and the signal is almost completely corrupted. We can recover from this using error-control codes if deep fades are relatively infrequent and short-lived. Thus, memory becomes an issue. If the channel is highly correlated over time, it will tend to stay bad for long periods of time making it impossible to communicate. Therefore, memory is generally considered bad. However, longer channel memory means that there are more observations from which to estimate the channel state and a better estimate of the channel state improves performance. It is quite well-known that memory usually increases the capacity. We treat this issue in more detail below.

Spread spectrum techniques avoid the problem of getting stuck in a deep fade by using a larger signal bandwidth so that when some parts of the band are deeply faded, others will be unfaded. Direct sequence spread spectrum uses the entire bandwidth at once, along with careful signal processing to recover as much of the signal as possible. Frequency-hopping spread spectrum manages the duration of deep fades by periodically changing, or hopping, the carrier frequency. This results in a new realization of the fading level. With error control coding across many hops, the bits lost during deeply faded hops can be recovered from those sent during unfaded hops.

Spread spectrum techniques have been popular for military applications because they are quite robust to hostile interference, or jamming. If the jammer does not know the carrier frequency sequence for a frequency-hopping spread spectrum system, it must spread its power over a wide bandwidth in hopes of interfering with as many hops as possible. The partial band jammer tries to jam just enough of the bandwidth, and therefore to interfere with enough hops, that decoding will be unsuccessful. Hops that fall in the jammed bandwidth experience high power additive Gaussian noise from the jammer, while those that fall outside the jammed bandwidth are affected only by background thermal noise. There is a trade-off between impacting many hops by jamming a large bandwidth, and achieving a high jamming power spectral density by jamming a small bandwidth. Direct sequence spread spectrum systems are also robust to jamming because the transmitted signal has a particular signature, and interference that does not match the signature can be largely filtered out by the receiver.

Spread spectrum can also be used to allow multiple users to share the same channel and bandwidth. To some extent, this makes up for the fact that it uses significantly more bandwidth than a regular signal at the same data rate. With frequency-hopping spread spectrum, users have different hopping patterns so they usually use different sections of the bandwidth. When two users do choose the same carrier frequency at the same time, this results in a collision and typically their mutual interference means that both of their messages are corrupted. Just as when a hop is lost to a deep fade, error control coding can be used to recover the corrupted bits. Direct sequence spread spectrum can also be used for multiple access by assigning users different signature sequences. Then a receiver listening to a particular user can treat the other users as undesirable interference and filter them out because they do not match the signature sequence of the desired user. Sophisticated multi-user detection techniques, which jointly estimate signals from several users, can be employed in both direct sequence and frequency-hopping systems to improve performance.

More conventional multiple access techniques include frequency-division multiple access (FDMA) and time-division multiple access (TDMA), which divide frequency and time respectively into exclusive slots and allocate these slots to different users. These both require strict coordination between the users to prevent conflicts. FDMA also has the disadvantage that if a user's signal is deeply faded in its assigned frequency slot, the user will be unable to communicate and the slot wasted.

1.4 Conventional Wisdom – Eliminate Memory

Until recently, most wireless communication systems have tried to eliminate memory so that the channel outputs processed by the decoder are independent. This was motivated largely by the fact that common codes are designed to correct random, independent errors, and receivers that use the channel memory are often difficult to implement. Interleaving across hops in frequency-hopping, time slots in TDMA, and time in direct sequence spread spectrum, has been used with the idea that successive bits in the codeword, because they are transmitted at different frequencies, or separated times, will experience roughly independent fading or interference. In block interleaving, we combine several codewords by transmitting the first bit of each codeword, followed by the second bit of each codeword and so on, so that bits from a particular codeword are separated by the number of codewords being interleaved. This is often implemented by reading the symbols into an array row-by-row, and out column-by-column. At the receiver, the bits from each codeword are extracted and the codeword is decoded independently of the other codewords. This makes the system robust to lost blocks of bits because they produce only one or two errors in each codeword, which are easily corrected. A classic coding scheme for frequency-hopping spread spectrum consists of Reed-Solomon codewords interleaved so that one symbol from each codeword is transmitted on each hop [60, 61].

Interleaving has two significant disadvantages. The first is that, unless we apply channel estimation before deinterleaving, the channel estimation capability created by the channel memory is lost. The second is that, for fixed total delay, using several codewords with small block length is inferior to using a single codeword from a much longer code. Note that the class of longer codes contains the class of codes which can be separated into independent component codewords, thus the best long code must be at least as good as the collection of interleaved short codewords. In generally, the additional connections possible in a single large codeword lead to improved performance. Part of the original motivation for interleaving several Reed-Solomon codewords was that very long Reed-Solomon codes are known to have poor performance. Also, the decoding complexity of Reed-Solomon codes runs as approximately $O(t^3)$ where t is the number of errors to be corrected. Reed-Solomon codes correct a fraction of errors roughly proportional to the code rate so the total number of errors that can be corrected is about the same for several interleaved short codes or one long code. However, the complexity grows linearly with the number of short codewords but as the cube of the length of the long code. Thus, complexity may have been a concern in designing the Reed-Solomon code system. Turbo codes and LDPC codes have complexity that grows linearly in the block length, and thus interleaving multiple codewords does not make sense for these codes.

1.5 Optimum Receivers: Joint Channel Estimation and Decoding

If interleaving over multiple codewords is not used to produce effectively independent output symbols, the receiver should use the channel memory to improve decoding. Unfortunately, the optimal receiver in this case needs to consider the entire output sequence as a unit and simplifications possible for decoding with independent channel outputs no longer apply. The decoder complexity becomes roughly exponential in the block length and thus it cannot be implemented for reasonable block lengths. However, the optimal decoder is easy to write—the MAP rule continues to apply—but the channel memory must be included in the conditional density. We have the same formula,

$$\hat{\underline{x}} = \arg\max_{\underline{x}} \mathrm{p}(\underline{x}|\underline{y}), \tag{1.9}$$

except that $\mathrm{p}(\underline{x}|\underline{y})$ may be quite complicated. Often, the channel memory can be modeled by a channel state that evolves over time,

$$\mathrm{p}(\underline{x}|\underline{y}) = \sum_{\underline{u}} \mathrm{p}(\underline{x}, \underline{u}|\underline{y}) \tag{1.10}$$

where $\underline{u}$ is the channel state sequence. Substituting in the optimum decision rule (1.9), we find that the best decision for the transmitted codeword is

$$\hat{\underline{x}} = \arg\max_{\underline{x}} \sum_{\underline{u}} \mathrm{p}(\underline{x}, \underline{u}|\underline{y}). \tag{1.11}$$

If we approximate the sum with its largest term, we get

$$\hat{\underline{x}} = \arg\max_{\underline{x}} \max_{\underline{u}} \mathrm{p}(\underline{x}, \underline{u}|\underline{y}). \tag{1.12}$$

which corresponds to jointly choosing the best estimates of the channel state sequence and the transmitted codeword. Note that both this expression and the optimal rule (1.11) are symmetric, so that interchanging $\underline{x}$ and $\underline{u}$ gives estimators for $\underline{u}$.

1.6 Effective Diversity vs. Channel Estimation

For fixed codeword length, channel memory affects performance in two ways. Longer channel memory leads to fewer independent, or approximately independent, realizations of the channel state during the codeword. We refer to the number of approximately independent realizations of the channel state as the effective diversity seen by the code. For frequency-hopping spread spectrum when the channel remains fixed throughout a hop and is independent from hop to hop, the effective diversity is the codeword length N divided by the hop length m, both in symbols, which is the number of hops per codeword. For other channels, it is not clear how to quantify the effective diversity, but clearly a similar measure exists. If the effective diversity is large, then, by the law of large numbers, the average channel behavior within a codeword is almost always close to the statistically expected channel behavior. If the code performs well under the expected channel behavior, then catastrophically bad channel realizations will be rare and the error probability will be low. On the other hand, if the effective diversity is small, there will be a significant chance of experiencing bad channel conditions, which will lead to poor performance. Shorter channel memory leads to greater effective diversity and therefore improves performance.

If the channel state is available to the receiver somehow, then effective diversity is the only memory effect and shorter memory is always better. However, in the much more common case when the channel state is not available to the receiver, we must estimate the channel state in addition to decoding the data. In general, longer memory allows more observations of the same or similar channel conditions and therefore improves channel estimation accuracy. Thus, we are faced with a trade-off between long channel memory for channel estimation accuracy and short memory to produce large effective diversity. This trade-off is an important theme throughout our work.

1.7 Overview

Motivated by the success of iterative decoding algorithms and the similarity of some channel models to simple codes, we consider iterative approaches to joint channel estimation and decoding. Various researchers [30, 65, 68] have also suggested that perhaps channel estimation could be added to the framework of graphical models for iterative decoding. Hagenauer [24] proposed that the sort of iterated processing of constituent codes used in turbo decoding could be extended to more complicated problems such as joint equalization and decoding and joint decoding of multiple users. Unfortunately, his approach does not provide the details of how the individual soft-input, soft-output (SISO) processing modules should be implemented. Anastasopoulos and Chugg [6, 7] derive adaptive SISO algorithms for decoding component codes in the presence of unknown channels.

After reviewing the factor graph approach to graphical modeling, and the sum-product iterative algorithm in Chapter 2, we propose a unified approach to designing iterative receivers in Chapter 3. Iterative receivers approximate joint channel estimation and decoding by using iterative algorithms on graphical models. Once the graphical model has been constructed, deriving the details of the algorithm is quite straightforward. We propose canonical distributions to address the problem of handling continuous variables, such as the channel state, in the sum-product algorithm. Our technique includes many previous approaches as special cases, depending on the canonical distributions chosen and other decisions made during the derivation. Receiver design examples for some common channels with LDPC codes are given in Chapter 4.

Encouraged by the performance of iterative receivers, we investigate the behavior of some bounds on ideal joint channel estimation and decoding for channels with block memory in Chapter 5. Of special interest is the effect of memory length on performance and the trade-off between effective diversity and channel estimation. Chapter 6 summarizes our results.

CHAPTER 2

Factor Graphs and the Sum-Product Algorithm

Factor graphs are a type of graphical model for multiplicative factorizations of functions. Simple algorithms based on the graphical model can be used to compute some important operations on the function. Under some conditions, the algorithms are exact, while more commonly they produce useful approximations to the desired results.

Factor graphs are closely related to a number of other graphical models used in computer science, statistics, and coding theory. Gallager's original work on low-density parity check (LDPC) codes in 1962 [17] was one of the first mentions of graphical models in coding. In 1981, Tanner [65] formalized design of codes using graphical models. Pearl [53] introduced belief propagation as an algorithm for statistical inference in an artificial intelligence context. In his dissertation, Wiberg [68,69] generalized Tanner's work and noted the connection between graphical models and turbo decoding. The connection between belief propagation and turbo decoding was also noted by others [37, 47]. Junction trees [49] and the generalized distributive law [4, 5] are alternate formulations and generalizations of the same basic class of algorithms. Markov random fields studied in statistics use similar concepts as well.

Factor graphs were proposed by Frey, Kschischang, Loeliger, and Wiberg [16, 38] to unify many of these ideas in a simple, general framework. One of their key contributions was the realization that the graphical models were related directly to a factorization of a global objective function. Once the factorization is available, the graph and its associated algorithms are immediate. They also show that a wide variety of algorithms from coding theory and statistical inference can be derived in the factor graph framework. In this chapter, the key concepts of factor graphs are described following their basic approach and notation, with minor variations for convenience in later chapters. Notation introduced here will be used throughout.

2.1 Factor Graphs

We begin with some notation. Let uppercase letters specify nodes in the factor graph. Lowercase letters will be used for values of variables and dummy variables in functions, so A is a node and a represents some value that the variable associated with A takes. Sets will typically be represented by script capital letters. If $\underline{v}$ is a vector of values associated with some variables, v_X is the value from the vector that is associated with variable node X. Similarly, $v_{\mathcal{S}}$ is the vector of values from $\underline{v}$ associated with the set of variables $\mathcal{S}$. One can think of this as a "slice" of $\underline{v}$ containing only the values for the variables in $\mathcal{S}$. $[\mathcal{S}]$ is the set of vectors representing all possible combinations of values of the variables in the set $\mathcal{S}$. For example, if $\mathcal{S} = \{X_1, X_2, X_3\}$ and $x_1 \in \mathcal{A}_{X_1}$, $x_2 \in \mathcal{A}_{X_2}$, and $x_3 \in \mathcal{A}_{X_3}$, then $[\mathcal{S}] = \mathcal{A}_{X_1} \times \mathcal{A}_{X_2} \times \mathcal{A}_{X_3}$. The $\mathrm{Ne}(\cdot)$ operator produces the set of neighbors of a node in the factor graph. The indicator function $\mathrm{I}\{$*statement*$\}$ is 1 if *statement* is true and 0 otherwise.

A factor graph is a graph representing a particular factorization of a function $g(\cdot)$ of many variables $x_1, x_2, \ldots, x_n$. Consider the factorization

$$g(x_1, x_2, \ldots, x_n) = \prod_j f_j(x_{\mathcal{Q}_j}) \tag{2.1}$$

where $x_{\mathcal{Q}_j}$ is a vector of elements from $x_1, x_2, \ldots, x_n$. The *factor graph* corresponding to the factorization consists of a *variable node* for each variable x_i, a *factor node* for each function f_j, and edges connecting each factor node to the variable nodes associated with its arguments. Thus, the factor graph is an undirected, bipartite graph where edges only connect variable nodes to factor nodes. In drawing factor graphs, we typically use filled circles for factor nodes and open circles for variable nodes.

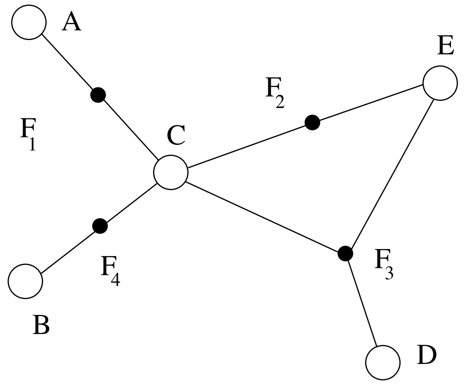


Figure 2.1: Example factor graph. $F_1 \triangleq f_1(a,c)$, $F_2 \triangleq f_2(c,e)$, $F_3 \triangleq f_3(c,d,e)$, and $F_4 \triangleq f_4(b,c)$.

For example, if

$$g(a,b,c,d,e) = f_1(a,c)f_2(c,e)f_3(c,d,e)f_4(b,c) \tag{2.2}$$

then the factor nodes correspond to $f_1(\cdot)$, $f_2(\cdot)$, $f_3(\cdot)$, and $f_4(\cdot)$ and the variable nodes correspond to a, b, c, d, and e. The factor graph is shown in Fig. 2.1 where we have indicated which functions the factor nodes represent in the caption. As described above, the variable nodes are labeled by their names in uppercase letters.

There are some simple operations that can be performed on a factor graph without changing its global function. Any variable nodes X and Y with alphabets $\mathcal{A}_X$ and $\mathcal{A}_Y$ can be combined into a new variable node $Z = (X,Y)$ whose alphabet is $\mathcal{A}_Z = \mathcal{A}_X \times \mathcal{A}_Y$ and whose neighbors are all the factor nodes that were connected to X or Y, $\mathrm{Ne}(Z) = \mathrm{Ne}(X) \cup \mathrm{Ne}(Y)$. The factor functions for the nodes $\mathrm{Ne}(Z)$ simply extract the components of Z that they need. Similarly, any factor nodes F_1 and F_2 with factor functions $f_1(\cdot)$ and $f_2(\cdot)$ can be combined into a new factor node F with factor function $f(\cdot) = f_1(\cdot)f_2(\cdot)$ and neighbors $\mathrm{Ne}(F) = \mathrm{Ne}(F_1) \cup \mathrm{Ne}(F_2)$. These operations are often helpful if the graph has undesirable cycles. We will see that although these operations can be used to modify the graph, they typically increase the complexity of the sum-product algorithm substantially.

Another common operation is to introduce additional *hidden variables* whose values do not interest us, but which somehow improve or simplify the graph structure. We usually draw these using double open circles. Formally, we are creating a new global function $g'(\cdot)$ which in addition to the variables of interest $x_1, x_2, \ldots, x_n$ also depends on some hidden variables $h_1, h_2, \ldots, h_k$. If

$$\sum_{\underline{h} \in [H_1, H_2, \ldots, H_k]} g'(\underline{x}, \underline{h}) = g(\underline{x}) \tag{2.3}$$

then we say that the graphs represent equivalent functions. A common example of this is when a function depends on the current variable and several past variables. If we add hidden variables to represent the vector of past values at each time, it turns out that the algorithm often performs better. Of course, these hidden vectors may have very large state spaces if there are many past variables or the past variables have large state spaces themselves.

We will often be considering particular sub-structures in factor graphs. In these cases, we draw the factor graph fragments with a dotted line to represent where edges would connect to the rest of the complete factor graph. Nodes that would be connected to other portions of the graph are shown with edges going to this dotted line. We will often reuse particular factorizations of functions, and thus particular factor graph fragments, in different derivations and it will sometimes be convenient to be able to look at each portion in isolation.

2.2 The Sum-Product Algorithm

We are often interested in computing the *marginal functions* $g_{x_i}(x_i)$ of the global function $g(\cdot)$ which are defined according to

$$g_{x_i}(x_i) = \sum_{\underline{x}^{(i)} \in [\{X_1, X_2, \ldots, X_n\} \setminus X_i]} g(\underline{x}^{(i)}, x_i) \tag{2.4}$$

where $\underline{x}^{(i)}$ contains all variables but x_i and the sum is over all possible values of the variables other than x_i, which takes the value given as the argument to $g_{x_i}(x_i)$. When $g(\cdot)$ is a probability mass function, then the marginal functions are the marginal probabilities of the variables. In our example, the marginal functions are

$$g_a(a) = \sum_{b,c,d,e} g(a,b,c,d,e),$$
$$g_b(b) = \sum_{a,c,d,e} g(a,b,c,d,e),$$
$$g_c(c) = \sum_{a,b,d,e} g(a,b,c,d,e),$$

and so on.

The sum-product algorithm is described as message passing on the graph. Every node stores a message "going to" each of its neighbors and can access a message "coming from" each of its neighbors. The message from node X to node Y is a function denoted $\mu_{X \to Y}(x)$ where we have assumed X is the variable node. Because of the graph structure, either X or Y (but not both) is a variable node. Beginning with all messages equal to 1, the algorithm updates the outgoing messages for each node according to a schedule that specifies the sequence in which the nodes are to be processed. We will say that the algorithm has completed when any node can be updated and its new outgoing messages will be the same as the old ones.

The new outgoing messages are computed in terms of the current incoming messages. For a variable node X_i and one of its neighboring factor nodes F_j, the general update is

$$\mu_{X_i \to F_j}(x_i) = \prod_{Q \in \mathrm{Ne}(X_i) \setminus F_j} \mu_{Q \to X_i}(x_i) \tag{2.5}$$

where $\mathrm{Ne}(X_i)$ is the set of neighbors of the node X_i in the graph, i.e. nodes corresponding to all the factor functions f_k for which x_i is an argument. $\mathrm{Ne}(X_i) \setminus F_j$ is the set of all neighbors of X_i except F_j. So the variable update simply multiplies incoming messages for all the edges adjacent to X_i except the one associated with the outgoing message being computed. If x_i only takes on a small finite set of values, we can compute and store the product for each value separately. The factor node update is a bit more involved. For a factor node F_j and one of its neighboring variable nodes X_i, the general update is

$$\mu_{F_j \to X_i}(x_i) = \sum_{\substack{\underline{z} \in [\mathrm{Ne}(F_j)]: \\ z_i = x_i}} f_j(\underline{z}) \prod_{Q \in \mathrm{Ne}(F_j) \setminus X_i} \mu_{Q \to F_j}(z_Q). \tag{2.6}$$

$[\mathrm{Ne}(F_j)]$ denotes the vector of all possible configurations, or combinations of values, of the variables adjacent to F_j. The condition restricts our attention to the set of configurations where the i^{th} component is x_i. The messages $\mu_{Q\to F_j}(\cdot)$ are evaluated at the corresponding values from the dummy vector of summation $\underline{z}$. When the state spaces of the variables are continuous-valued, we replace the sums with integrals and the updates remain valid.

Returning to our previous example, assume that the variables each have two possible values, so a can be a_0 or a_1, b can be b_0 or b_1 and so on. Then, for example, we can compute the update for the variable node C as

$$\mu_{C\to F_1}(c_0) = \mu_{F_2\to C}(c_0)\cdot\mu_{F_3\to C}(c_0)\cdot\mu_{F_4\to C}(c_0) \tag{2.7}$$

$$\mu_{C\to F_1}(c_1) = \mu_{F_2\to C}(c_1)\cdot\mu_{F_3\to C}(c_1)\cdot\mu_{F_4\to C}(c_1) \tag{2.8}$$

$$\mu_{C\to F_2}(c_0) = \mu_{F_1\to C}(c_0)\cdot\mu_{F_3\to C}(c_0)\cdot\mu_{F_4\to C}(c_0) \tag{2.9}$$

$$\mu_{C\to F_2}(c_1) = \mu_{F_1\to C}(c_1)\cdot\mu_{F_3\to C}(c_1)\cdot\mu_{F_4\to C}(c_1) \tag{2.10}$$

and so on. For the factor node F_1, we get

$$\mu_{F_1\to A}(a_0) = f_1(a_0,c_0)\mu_{C\to F_1}(c_0) + f_1(a_0,c_1)\mu_{C\to F_1}(c_1) \tag{2.11}$$

$$\mu_{F_1\to A}(a_1) = f_1(a_1,c_0)\mu_{C\to F_1}(c_0) + f_1(a_1,c_1)\mu_{C\to F_1}(c_1) \tag{2.12}$$

$$\mu_{F_1\to C}(c_0) = f_1(a_0,c_0)\mu_{A\to F_1}(a_0) + f_1(a_1,c_0)\mu_{A\to F_1}(a_1) \tag{2.13}$$

$$\mu_{F_1\to C}(c_1) = f_1(a_0,c_1)\mu_{A\to F_1}(a_0) + f_1(a_1,c_1)\mu_{A\to F_1}(a_1). \tag{2.14}$$

It is well-known [38] that when the graph is finite and has no cycles, i.e. it is a tree (or a forest), the sum-product algorithm completes in finitely many node updates and the marginal functions that it produces are exact. On the other hand, when the graph has cycles, the marginal functions are generally not exact, but it turns out that when we use them to make decisions, as in a decoder, the results are usually quite good.

This framework can be generalized by replacing the addition and multiplication operations throughout with other operations that have the same algebraic properties. This leads to cousins of the sum-product algorithm with different update complexity and performance on graphs with cycles. Note that all these algorithms are exact on trees for marginal functions taken with respect to the chosen sum operation. The min-sum algorithm, where multiplication is replaced with arithmetic addition and addition is replaced with minimization, is the most common example. The BCJR algorithm [9] and the Viterbi algorithm are related through this type of change of operations.

2.3 Low-Density Parity Check Codes

In Chapter 1, we defined a low-density parity check (LDPC) code as a linear block code whose parity check matrix $\mathbf{H}$ is low-density in that it has a very small number of ones in each row and column. The membership function for a low-density parity check code is $\mathrm{I}\{\mathbf{H}\,\underline{x} = 0\}$ which can be factored row by row as

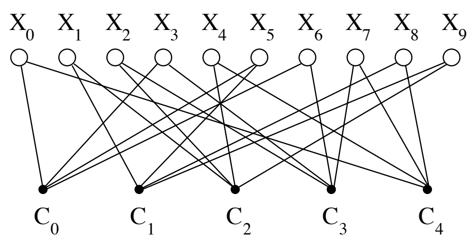
$$\mathrm{I}\{\mathbf{H}\,\underline{x} = 0\} = \prod_{j=0}^{r-1} \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\} \tag{2.15}$$

where $\underline{h}_j$ is the j^{th} row of the parity check matrix for $\mathcal{C}$, and r is the number of rows in the parity check matrix. Each of these row constraints $\mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\}$ is an even parity code involving a very small subset of the elements of $\underline{x}$. For example, consider the regular block length 10, rate 1/2 low-density parity check code with column weight 2 and row weight 4 described by the following parity check matrix

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \tag{2.16}$$

We can draw a factor graph for the factorization (2.15) by connecting each even-parity constraint to the bits that it checks. In our example, the first constraint checks the first, fourth, sixth, and seventh bits, so its factor node is connected to those variable nodes in the factor graph. The complete factor graph is shown in Fig. 2.2.

We build an iterative decoder graph for this code on a memoryless channel by attaching factor nodes T_i for the functions $\mathrm{p}(y_i|x_i)$ to the variable nodes x_i and applying the sum-product algorithm (see Fig. 2.3). The variables y_i do not appear in



X_0 X_1 X_2 X_3 X_4 X_5 X_6 X_7 X_8 X_9
C_0 C_1 C_2 C_3 C_4

Figure 2.2: Factor graph for the example low-density parity check code. $C_j \triangleq \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\}$.

the factor graph because they are known parameters observed by the receiver. The node updates are easy to write using the general update equations described above. The message $\mu_{T_i \to X_i}(x_i) = \mathrm{p}(y_i|x_i)$ and does not need to be updated. For the variable node X_0, we have

$$\mu_{X_0 \to C_0}(x_0) = \mathrm{p}(y_0|x_0) \cdot \mu_{C_4 \to X_0}(x_0) \tag{2.17}$$

$$\mu_{X_0 \to C_4}(x_0) = \mathrm{p}(y_0|x_0) \cdot \mu_{C_0 \to X_0}(x_0) \tag{2.18}$$

and similarly for the other variable nodes. For the constraint node C_0, we have

$$\mu_{C_0 \to X_0}(x_0) = \sum_{[x_3,x_5,x_6] \in \{0,1\}^3} \mathrm{I}\left\{x_0 + x_3 + x_5 + x_6 = 0\right\} \cdot \mu_{X_3 \to C_0}(x_3)\, \mu_{X_5 \to C_0}(x_5)\, \mu_{X_6 \to C_0}(x_6) \tag{2.19}$$

$$\mu_{C_0 \to X_3}(x_3) = \sum_{[x_0,x_5,x_6] \in \{0,1\}^3} \mathrm{I}\left\{x_0 + x_3 + x_5 + x_6 = 0\right\} \cdot \mu_{X_0 \to C_0}(x_0)\, \mu_{X_5 \to C_0}(x_5)\, \mu_{X_6 \to C_0}(x_6) \tag{2.20}$$

$$\mu_{C_0 \to X_5}(x_5) = \sum_{[x_0,x_3,x_6] \in \{0,1\}^3} \mathrm{I}\left\{x_0 + x_3 + x_5 + x_6 = 0\right\} \cdot \mu_{X_0 \to C_0}(x_0)\, \mu_{X_3 \to C_0}(x_3)\, \mu_{X_6 \to C_0}(x_6) \tag{2.21}$$

$$\mu_{C_0 \to X_6}(x_6) = \sum_{[x_0,x_3,x_5] \in \{0,1\}^3} \mathrm{I}\left\{x_0 + x_3 + x_5 + x_6 = 0\right\} \cdot \mu_{X_0 \to C_0}(x_0)\, \mu_{X_3 \to C_0}(x_3)\, \mu_{X_5 \to C_0}(x_5) \tag{2.22}$$

where addition of variables is performed modulo 2. Similar formulas hold for the other constraint nodes. Note that these updates can be rewritten as, for example,

$$\mu_{C_0 \to X_0}(x_0) = \sum_{[x_3,x_5,x_6] \in \mathcal{Z}(x_0)} \mu_{X_3 \to C_0}(x_3)\, \mu_{X_5 \to C_0}(x_5)\, \mu_{X_6 \to C_0}(x_6) \tag{2.23}$$

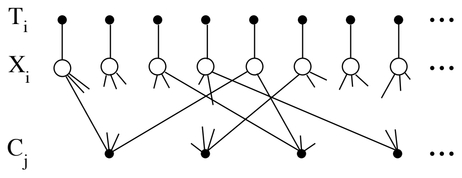


Figure 2.3: Factor graph for an LDPC code on a memoryless channel. $C_j \triangleq \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\}$, $T_i \triangleq \mathrm{p}(y_i|x_i)$.

where

$$\mathcal{Z}(x_0) = \left\{[x_3, x_5, x_6] \in \{0,1\}^3 : x_0 + x_3 + x_5 + x_6 = 0\right\}$$

so we sum over the values that give even parity overall.

2.4 Efficient Updates and Complexity

The sum-product algorithm updates are of manageable complexity if the node degrees, or numbers of neighbors, are small. Furthermore, the updates can often be simplified by various tricks.

As written above, the update for a variable node X_i with a finite state space and degree L requires $L(L-2)$ multiplications for each of its possible values. Inspired by the BCJR algorithm [9], an efficient recursion to compute the outgoing messages for all the neighbors of a variable node X_i places an arbitrary order on the elements of $\mathrm{Ne}(X_i)$ and computes the forward and backward partial products α_k and β_k. These are defined according to

$$\alpha_k = \prod_{j<k} \mu_{Q_j \to X_i}(x_i) \tag{2.24}$$

$$\beta_k = \prod_{j>k} \mu_{Q_j \to X_i}(x_i) \tag{2.25}$$

where $\alpha_0 = 1$ and $\beta_{L-1} = 1$, and can be computed according to the recursions

$$\alpha_k = \alpha_{k-1} \cdot \mu_{Q_{k-1} \to X_i}(x_i) \tag{2.26}$$

$$\beta_k = \beta_{k+1} \cdot \mu_{Q_{k+1} \to X_i}(x_i). \tag{2.27}$$

This requires $2(L-1)$ multiplications for each value of x_i. The outgoing messages for all $j = 0, 1, \ldots, L-1$ can be computed in an additional L multiplications

$$\mu_{X_i \to Q_j}(x_i) = \alpha_j \cdot \beta_j \tag{2.28}$$

for a total cost of $3L-2$ multiplications for each value of x_i.

The update for a factor node with degree L and common neighbor alphabet size M, if computed for each neighbor individually, requires $(L-1) \times M^{L-1}$ multiplications, M^{L-1} evaluations of the factor function, and $M^{L-1}-1$ additions for each neighbor, for each value. In the sequel, we will find that our factor nodes either have small degrees or the factor functions have properties that allow us to greatly simplify the updates. A neat trick is to find a small cycle-free factor graph that represents the factor function. If this is done carefully, applying the sum-product algorithm sometimes yields very low-complexity updates, which, because the small factor graph is a tree, are exact. For the low-density parity check code constraint node update, we can form a trellis representation of the even parity code, and then use the BCJR algorithm to perform the node update with effort linear in the node degree. This is equivalent to introducing a hidden state that stores the parity of the bits already processed. We start and end in the even parity state and the sum from (2.3) is over all the paths in between.

CHAPTER 3

Unified Design of Iterative Receivers

In this chapter, we describe a unified approach to designing iterative receivers for a wide range of channel models with memory. Beginning with basic principles of optimal reception, we develop graphical models that lead directly to iterative algorithms for joint channel estimation and decoding.

Recall our overall system model. The information bits are encoded into a codeword $\underline{x}$ which passes through the channel with state vector $\underline{u}$. The channel output $\underline{y}$ is processed by the iterative receiver, which does not know $\underline{u}$, to produce the message estimates. We assume that the encoder is systematic so that the message estimate is a subset of the codeword bit estimates. The channel is specified by its transition probability $\mathrm{p}(\underline{y}|\underline{x}, \underline{u})$ conditioned on the state and the prior probability for the channel state $\mathrm{p}(\underline{u})$ which we assume is independent of the channel input $\underline{x}$. As described above, we can either perform optimal sequence detection or optimal symbol detection. For iterative receivers, we choose to consider minimum symbol error probability decisions. Thus, the optimal decision rule for symbol x_i is

$$\hat{x}_i = \arg\max_{x_i} \mathrm{p}(x_i|\underline{y}). \tag{3.1}$$

Rewrite this as

$$\begin{aligned} \hat{x}_i &= \arg\max_{\xi} \sum_{\underline{x}: x_i = \xi} \mathrm{p}(\underline{x}|\underline{y}) \\ &= \arg\max_{\xi} \sum_{\underline{x}: x_i = \xi} \mathrm{p}(\underline{y}|\underline{x})\, \mathrm{p}(\underline{x}) \end{aligned} \tag{3.2}$$

using Bayes' rule and noting that $\mathrm{p}(\underline{y})$ does not depend on $\underline{x}$ or ξ. Using the channel state,

$$\hat{x}_i = \arg\max_{\xi} \sum_{\underline{x}:x_i=\xi} \int\limits_{\mathcal{U}} \mathrm{p}(\underline{y}|\underline{x},\underline{u})\,\mathrm{p}(\underline{u})\,\mathrm{p}(\underline{x})\;d\underline{u}. \tag{3.3}$$

Since we assume that the codewords are equally likely,

$$\mathrm{p}(\underline{x}) = \frac{1}{|\mathcal{C}|}\,\mathrm{I}\left\{\underline{x} \in \mathcal{C}\right\}. \tag{3.4}$$

Because $1/|\mathcal{C}|$ does not depend on $\underline{x}$, we find the decision rule

$$\hat{x}_i = \arg\max_{\xi} \sum_{\underline{x}:x_i=\xi} \int\limits_{\mathcal{U}} \mathrm{p}(\underline{y}|\underline{x},\underline{u})\,\mathrm{p}(\underline{u})\,\mathrm{I}\left\{\underline{x} \in \mathcal{C}\right\}\;d\underline{u}. \tag{3.5}$$

Conveniently, $\hat{x}_i$ is the maximizing value of the marginal that the sum-product algorithm computes. The global function that the factor graph should represent is

$$G(\underline{x},\underline{u}) = \mathrm{p}(\underline{y}|\underline{x},\underline{u})\,\mathrm{p}(\underline{u})\,\mathrm{I}\left\{\underline{x} \in \mathcal{C}\right\} \tag{3.6}$$

where we treat $\underline{y}$ as a parameter because it is known to the decoder. Thus, our goal is to find a factorization of $G(\underline{x},\underline{u})$ that leads to a realizable and hopefully also effective decoder. Once the factorization has been chosen, the process is essentially deductive–very few conceptual problems need to be solved. We draw the factor graph, apply the sum-product algorithm, and then use canonical distributions to obtain realizable updates. At each point, different choices lead to receivers of varying complexity and performance.

The rest of this chapter explains general techniques for factoring the objective function and choosing canonical distributions. It also includes a discussion of how these iterative receivers relate to other proposed algorithms and gives some tricks for finding useful receiver designs. The next chapter includes several specific receiver design examples with simulation results comparing the performance of various techniques.

3.1 Factoring the Objective Function $G(\underline{x},\underline{u})$

In order to obtain realizable algorithms, we must find a factorization of $G(\underline{x},\underline{u})$ that leads to sum-product node update operations that are realizable. Typically, this

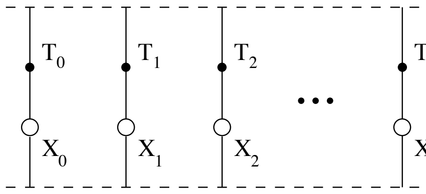


Figure 3.1: Factor graph fragment from (3.7) for a channel with no ISI. $T_i \triangleq \mathrm{p}(y_i|x_i, \underline{u})$

means that the degrees of the factor nodes in the factor graph must be small. Usually, we would like the overall algorithm complexity to be roughly $O(N)$ which requires that each node update have complexity bounded independent of N. Fortunately, useful factorizations are often easy to find. Since $G(\underline{x}, \underline{u})$ consists of three parts $\mathrm{p}(\underline{y}|\underline{x}, \underline{u})$, $\mathrm{p}(\underline{u})$, and $\mathrm{I}\{\underline{x} \in \mathcal{C}\}$, we will consider these parts one at a time.

Because we assume that the channel memory is encapsulated by the channel state, the channel likelihood $\mathrm{p}(\underline{y}|\underline{x}, \underline{u})$ factors easily. For channel models with no inter-symbol interference (ISI), the channel output is conditionally independent of all but the current input given the channel state

$$\mathrm{p}(\underline{y}|\underline{x}, \underline{u}) = \prod_{i=0}^{N-1} \mathrm{p}(y_i|x_i, \underline{u}). \tag{3.7}$$

This leads to the straightforward factor graph structure in Fig. 3.1.

If the channel has ISI of length L, the channel output is conditionally independent of all but the last L inputs given the channel state

$$\mathrm{p}(\underline{y}|\underline{x}, \underline{u}) = \prod_{i=0}^{N-1} \mathrm{p}(y_i|\underline{u}, x_i, x_{i-1}, \ldots, x_{i-L}) \tag{3.8}$$

where we take

$$\mathrm{p}(y_i|\underline{u}, x_i, x_{i-1}, \ldots, x_{i-L}) = \mathrm{p}(y_i|\underline{u}, x_i, x_{i-1}, \ldots, x_0)$$

if $i < L$. A factor graph fragment for this expression is displayed in Fig. 3.2a. Alternately, we could combine the L immediately past x_i values to create modulation state variables $h_i = [x_{i-1}, x_{i-2}, \ldots, x_{i-L}]$ representing the past channel inputs exactly as is done in the Viterbi algorithm for equalization. This leads to the factorization

$$\mathrm{p}(\underline{y}|\underline{x}, \underline{u}) = \mathrm{I}\{h_0 = 0\} \prod_{i=0}^{N-1} \mathrm{p}(y_i|\underline{u}, x_i, h_i) \cdot \mathrm{I}\{h_{i+1} = [x_i, \tilde{h}_i]\} \tag{3.9}$$

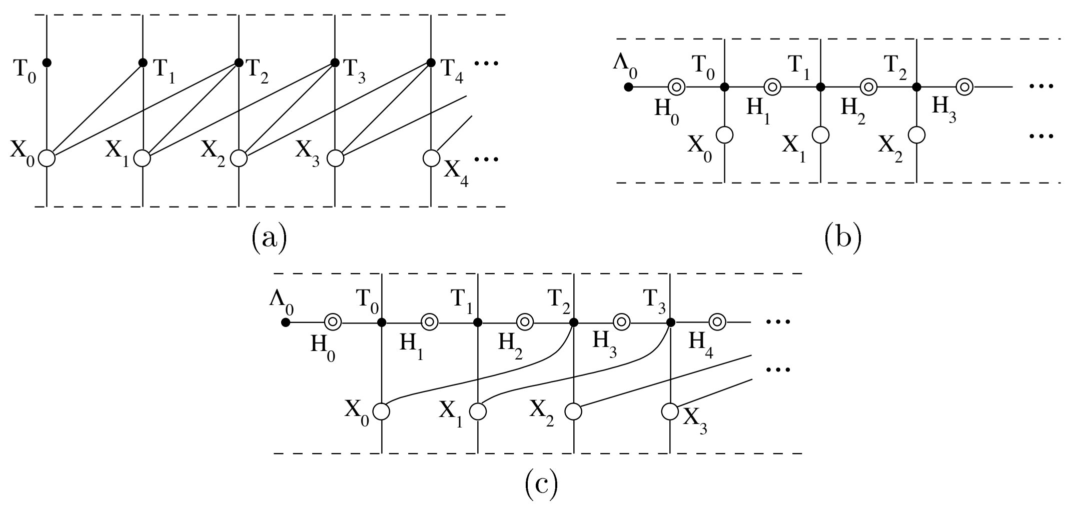


Figure 3.2: Factor graph fragments for a channel with ISI $L = 2$. (a) direct from (3.8) $T_i \triangleq \mathrm{p}(y_i|\underline{u}, x_i, x_{i-1}, \ldots, x_{i-L})$; (b) hidden states from (3.9) $T_i \triangleq \mathrm{p}(y_i|\underline{u}, x_i, h_i) \ \ \mathrm{I}\{h_{i+1} = [x_i, \tilde{h}_i]\}$; (c) reduced state from (3.10) $J = 2$ $T_i \triangleq \mathrm{p}(y_i|\underline{u}, x_i, h_i, x_{i-J}, \ldots, x_{i-L}) \ \mathrm{I}\{h_{i+1} = [x_i, \tilde{h}_i]\}$, $\Lambda_0 \triangleq \mathrm{I}\{h_0 = 0\}$.

where $\tilde{h}_i = [h_{i,0}, h_{i,1}, \ldots, h_{i,L-2}]$ is the vector containing the first $L-1$ elements of h_i (see Fig. 3.2b). The factor $\mathrm{I}\{h_{i+1} = [x_i, \tilde{h}_i]\}$ enforces the consistency of the state sequence. A hybrid, reduced state approach, reminiscent of decision-feedback sequence estimation [15], reduces the state dimension from $L+1$ to $J+1$, so $h_i = [x_{i-1}, x_{i-2}, \ldots, x_{i-J}]$, by only forcing state consistency for the $J+1$ immediate past values of x. This gives the factorization

$$\mathrm{p}(\underline{y}|\underline{x}, \underline{u}) = \mathrm{I}\{h_0 = 0\} \prod_{i=0}^{N-1} \mathrm{p}(y_i|\underline{u}, x_i, h_i, x_{i-J-1}, x_{i-J-2}, \ldots, x_{i-L}) \cdot \mathrm{I}\{h_{i+1} = [x_i, \tilde{h}_i]\} \tag{3.10}$$

(see Fig. 3.2c). A fundamental problem of (3.8), (3.9), and (3.10) is that the factor node updates have complexity exponential in L. Fortunately, previous work on reduced complexity equalization in iterative algorithms suggests that simplified updates using decision feedback work well [71].

Useful factorizations for the channel prior function $\mathrm{p}(\underline{u})$ exist for many common channel memory models. We will focus primarily on channels without ISI. Often, the channel functions $\mathrm{p}(y_i|x_i, \underline{u})$ do not actually depend on the whole channel state vector $\underline{u}$. Each symbol depends only on a relatively low-dimensional channel state $\underline{u}_i$ which

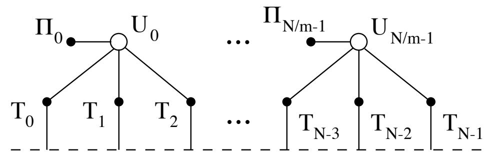


Figure 3.3: Factor graph fragment from (3.14) for a channel with block memory $m = 3$. $\Pi_k \triangleq \mathrm{p}(\underline{u}_k)$, $T_i \triangleq \mathrm{p}(y_i|x_i, \underline{u}_k)$.

could be considered to be a small sub-vector of the overall channel state sequence $\underline{u}$. In this case,

$$\mathrm{p}(y_i|x_i, \underline{u}) = \mathrm{p}(y_i|x_i, \underline{u}_i). \tag{3.11}$$

The time-variation and memory effects in the channel state are captured by the prior probability $\mathrm{p}(\underline{u})$.

For a block-memory channel, the channel state is fixed for each block of m symbols and then changes to a new independent value. Here the channel prior is just the product of the priors for the blocks,

$$\mathrm{p}(\underline{u}) = \prod_{k=0}^{\frac{N}{m}-1} \mathrm{p}(\underline{u}_{m\cdot k}) \cdot \mathrm{I}\left\{\underline{u}_{m\cdot k} = \underline{u}_{m\cdot k+1} = \cdots = \underline{u}_{m\cdot k+(m-1)}\right\} \tag{3.12}$$

where the indicator function reflects the fact that the state is fixed for each block of m symbols. Using this structure, we can reduce the number of $\underline{u}_i$ state vectors by having all the symbols in a block refer to the same state

$$\mathrm{p}(y_i|x_i, \underline{u}) = \mathrm{p}(y_i|x_i, \underline{u}_{\lfloor \frac{i}{m} \rfloor}) \tag{3.13}$$

where $\underline{u}_k$ is now the channel state for all m symbols in the kth block so $\underline{u}_{\lfloor \frac{i}{m} \rfloor}$ is the channel state when the ith symbol is sent. For this case, the channel prior factors as

$$\mathrm{p}(\underline{u}) = \prod_{k=0}^{\frac{N}{m}-1} \mathrm{p}(\underline{u}_k). \tag{3.14}$$

A factor graph fragment for block memory channels is shown in Fig. 3.3.

For a Markov-memory channel, the state sequence $\{\underline{u}_0, \underline{u}_1, \ldots, \underline{u}_{N-1}\}$ has the Markov property

$$\mathrm{p}(\underline{u}_j|\underline{u}_0, \ldots, \underline{u}_{j-1}) = \mathrm{p}(\underline{u}_j|\underline{u}_{j-1}). \tag{3.15}$$

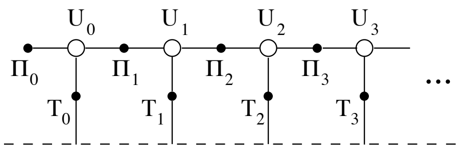


Figure 3.4: Factor graph fragment from (3.17) for a channel with Markov memory. $\Pi_k \triangleq \mathrm{p}(u_k|u_{k-1})$, $\Pi_0 \triangleq \mathrm{p}(u_0)$, $T_i \triangleq \mathrm{p}(y_i|x_i, u_i)$.

Any arbitrary $\mathrm{p}(\underline{u})$ can be factored according to

$$\begin{aligned}\mathrm{p}(\underline{u}) &= \mathrm{p}(\underline{u}_{N-1}|\underline{u}_{N-2}, \underline{u}_{N-3}, \ldots, \underline{u}_0) \cdot \mathrm{p}(\underline{u}_{N-2}, \ldots, \underline{u}_0) \\ &= \mathrm{p}(\underline{u}_0) \cdot \prod_{j=1}^{N-1} \mathrm{p}(\underline{u}_j|\underline{u}_{j-1}, \underline{u}_{j-2}, \ldots, \underline{u}_0).\end{aligned} \tag{3.16}$$

Using the Markov property (3.15), we get

$$\mathrm{p}(\underline{u}) = \mathrm{p}(\underline{u}_0) \cdot \prod_{j=1}^{N-1} \mathrm{p}(\underline{u}_j|\underline{u}_{j-1}). \tag{3.17}$$

A factor graph fragment for this channel is shown in Fig. 3.4.

Suppose that the channel state evolves according to a 2nd order Markov model. Some research indicates that accurate modeling of realistic fading channels requires a 2nd order model unless the fading is extremely slow [13]. In this case, we have the second-order Markov property

$$\mathrm{p}(\underline{u}_j|\underline{u}_{j-1}, \ldots, \underline{u}_0) = \mathrm{p}(\underline{u}_j|\underline{u}_{j-1}, \underline{u}_{j-2}). \tag{3.18}$$

There are two approaches to factoring $\mathrm{p}(\underline{u})$ in this case. If we substitute directly in (3.16), we get

$$\mathrm{p}(\underline{u}) = \mathrm{p}(\underline{u}_0)\,\mathrm{p}(\underline{u}_1|\underline{u}_0) \cdot \prod_{j=2}^{N-1} \mathrm{p}(\underline{u}_j|\underline{u}_{j-1}, \underline{u}_{j-2}). \tag{3.19}$$

The factor graph fragment for this case is shown in Fig. 3.5. We could also create a hidden state $h_i = [\underline{u}_{i-1}, \underline{u}_{i-2}]$ as we did for the ISI channel. With the hidden state,

$$\mathrm{p}(\underline{u}) = \mathrm{p}(h_0) \cdot \prod_{j=2}^{N-1} \mathrm{p}(\underline{u}_j|h_j)\,\mathrm{I}\left\{h_{j+1} = [\underline{u}_j, \tilde{h}_j]\right\}. \tag{3.20}$$

where $\mathrm{p}(h_0) = \mathrm{p}(u_0)\,\mathrm{p}(u_0|u_1)$. The factor graph using hidden states is displayed in Fig. 3.6.

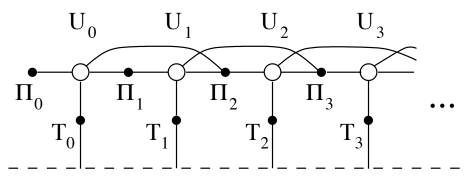


Figure 3.5: Factor graph fragment from (3.19) for a channel with 2nd order Markov memory. $\Pi_k \triangleq \mathrm{p}(\underline{u}_j|\underline{u}_{j-1},\underline{u}_{j-2})$, $T_i \triangleq \mathrm{p}(y_i|x_i\underline{u}_i)$.

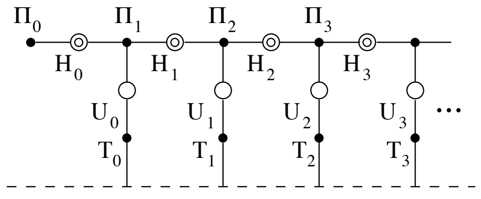


Figure 3.6: Factor graph fragment from (3.20) for a channel with 2nd order Markov memory. $\Pi_k \triangleq \mathrm{p}(\underline{u}_j|h_j)\,\mathrm{I}\left\{h_{j+1} = [\underline{u}_j, \tilde{h}_j]\right\}$, $T_i \triangleq \mathrm{p}(y_i|x_i\underline{u}_i)$.

The final portion of $G(\underline{x},\underline{u})$ to be considered is the code membership function $\mathrm{I}\{\underline{x} \in \mathcal{C}\}$. Some of the best known systems use codes with well-known graphical models and iterative decoding, including low-density parity check codes and turbo codes. Factor graph models for low-density parity codes were discussed in detail in section 2.3. For turbo codes, let $q_X(\cdot)$ and $t_X(\cdot)$ map the encoder state and the encoder input into the parity symbol and next state respectively for encoder X, so the membership function factors as

$$\mathrm{I}\{\underline{x} \in \mathcal{C}\} = \mathrm{I}\{h_{A,0} = 0\} \prod_{i=0}^{N/3-1} \mathrm{I}\{x_{A,i} = q_A(h_{A,i}, x_{D,i})\}\ \mathrm{I}\{h_{A,i+1} = t_A(h_{A,i}, x_{D,i})\} \cdot$$
$$\mathrm{I}\{h_{B,0} = 0\} \prod_{i=0}^{N/3-1} \mathrm{I}\{x_{B,i} = q_B(h_{B,i}, x_{D,\tilde{i}})\}\,\mathrm{I}\{h_{B,i+1} = t_B(h_{B,i}, x_{D,\tilde{i}})\} \quad (3.21)$$

where h_A and h_B are the state sequences for the two constituent encoders, x_A and x_B are the two parity check output sequences, x_D is the information bit sequence and $\tilde{i}$ indicates the i^{th} element of the permuted information bits. A turbo code factor graph fragment is shown in Fig. 3.7.

By replacing the parts of $G(\underline{x},\underline{u})$ with their factorizations, we find a complete factorization and thus, a complete factor graph for the objective function $G(\underline{x},\underline{u})$. We can apply the sum-product algorithm to this factor graph directly. Although the

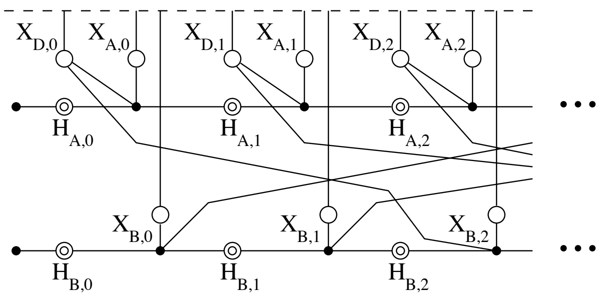


Figure 3.7: Factor graph fragment for a turbo code. The factor nodes in the A row represent $\mathrm{I}\{x_{A,i} = q_A(h_{A,i}, x_{D,i})\}$ $\mathrm{I}\{h_{A,i+1} = t_A(h_{A,i}, x_{D,i})\}$ and those in the B row represent $\mathrm{I}\{x_{B,i} = q_B(h_{B,i}, x_{D,\tilde{i}})\}$ $\mathrm{I}\{h_{B,i+1} = t_B(h_{B,i}, x_{D,\tilde{i}})\}$.

factor graph will almost always have loops, the resulting algorithm often still works well. In the factorizations given above, the maximum degree of any factor node is asymptotically less than m and N so the algorithm should scale with manageable complexity to long block lengths and even to long hop lengths. Although there are variable nodes with degree m, such as the state nodes for channels with block memory, these can be processed with complexity linear in m and therefore are not a concern. Because some factor node degrees depend on the ISI length L, long ISI can potentially require computationally costly node updates. Hybrid factorizations such as (3.10) and simplified updates, using decision feedback for example, allow us to manage complexity for these nodes as well.

3.2 Canonical Distributions

The update equations given by the sum-product algorithm often include messages that are functions of real-valued arguments. Manipulating these is difficult because their functional form rarely allows systematic simplification. Important exceptions include Kalman filtering and some other situations with Gaussian densities. In order to create manageable updates for cases where the update equations do not simplify, we suggest representing the message functions $\mu_{X \to Y}(\cdot)$ with parameterized *canonical distributions* (CD's). Instead of manipulating the actual functional form of the message, we instead compute the parameters of the canonical distribution. Typical approaches to channel estimation for continuous-valued channels in the "turbo" framework can often be expressed using our canonical distributions approach.

Consider a variable node V and one of its neighboring factor nodes F. There is a message $\mu_{V\to F}(v)$ from V to F and another $\mu_{F\to V}(v)$ from F to V. The canonical distributions approach is to replace *a priori* the nominal message $\mu_{V\to F}(v)$ calculated using the usual sum-product update equations with a parameterized function $\mu'_{V\to F}(v)$ called the canonical distribution for $\mu_{V\to F}(v)$. The functional form for the canonical distribution $\mu'_{V\to F}(v)$ is chosen so that the update for F is easy to compute in terms of the parameters for the canonical distribution. Thus, V passes the parameters of $\mu'_{V\to F}(v)$ to F which F uses to compute its outgoing messages with reasonable effort. Often, the nominal message $\mu_{F\to V}(v)$ has a simple parametric form that we can use directly, especially when the other incoming messages for F are approximated using canonical distributions. Otherwise, we select a canonical distribution $\mu'_{F\to V}(v)$ for $\mu_{F\to V}(v)$ as well. This process of approximating the nominal messages involving continuous variable nodes continues until all the messages can be represented with fixed parametric forms and all the updates can be computed using reasonable parameter computations. The goal is to prevent the functional forms of the messages from growing more complicated as the nodes are repeatedly updated. Although this sounds difficult, usually the updates involve at most a few continuous-valued variables. The task is made easier by the fact that the approximation is local; we need only compute the nominal messages and choose a simple function to represent their general behavior.

A common technique in the SISO algorithm approach is to estimate the channel using standard estimation techniques and then use the estimate as if it were true in the SISO algorithm. In our unified approach, this is equivalent to using a canonical distribution that is a delta function at the estimated value

$$\mu'_{X\to Y}(x) = \delta(x - \hat{x}) \tag{3.22}$$

where X is the variable node, Y is the factor node, and $\hat{x}$ is our estimate of the variable x. As we will see below, this approach is sometimes not very effective because it looses all reliability information about the estimate.

Another common choice is to discretize the message according to some fixed grid. This is essentially assuming that the value is quantized. There are two natural approaches to choosing a discretized canonical distribution. One approach considers the distribution to be a series of weighted delta functions

$$\mu'_{F\to V}(v) = \sum_{l=0}^{L-1} a_l \delta(v - \hat{v}_l) \tag{3.23}$$

where there are L possible levels, $\hat{v}_0, \hat{v}_1, \ldots, \hat{v}_{L-1}$, with weights $a_0, a_1, \ldots, a_{L-1}$. The weights a_l might be samples of the nominal message. Another approach supposes that the message function is piece-wise constant, i.e. its value is fixed over some intervals $(\tilde{v}_l, \tilde{v}_{l+1}]$. Let $r_l(v) = 1$ if $\tilde{v}_l < v \leq \tilde{v}_{l+1}$ and $r_l(v) = 0$ otherwise, for some fixed set of $\tilde{v}_l$, $l = 0, \ldots, L$. For a set of levels $a_0, a_1, \ldots, a_{L-1}$ corresponding to the intervals, we have

$$\mu'_{F\to V}(v) = \sum_{l=0}^{L-1} a_l r_l(v). \tag{3.24}$$

We could choose the levels to be samples of the nominal message function or we could take the average of the nominal message function over each interval,

$$a_l = \frac{1}{\tilde{v}_{l+1} - \tilde{v}_l} \int_{\tilde{v}_l}^{\tilde{v}_{l+1}} \mu_{F\to V}(v)\, dv. \tag{3.25}$$

Clearly, if the nominal message is continuous, the two approaches give the same weights a_l as the $\tilde{v}_l$ values get closer together.

The real power of canonical distributions is the possibility of choosing functions specifically matched to the form of the nominal message. For example, in the case of an unknown phase channel with phase-shift-keyed inputs, we know that the distribution will be multi-modal, with a peak corresponding to each possible input. As we will see below, selecting a canonical distribution that captures this multi-modal character is essential for realizing the performance benefits of iterative reception.

3.3 Discussion

Previously proposed algorithms can often be derived by choosing a specific factorization and canonical distributions. One technique for joint channel estimation and decoding applies to codes and channels that are easily represented by trellises, such as convolutional codes or the inner code of SCCC's, and channels with Markov memory. The channel is discretized and then a new super-trellis is generated whose states come from the product space of the channel states and the code states. Clearly, this can yield very high complexity if either of the state spaces is large. A simple variant of the BCJR algorithm operates on the super-trellis to produce soft-decisions for other component SISO algorithms. A receiver of this type for turbo codes in time-correlated Rayleigh fading is given in [35]. Approaches based on per-survivor processing [7] are related except that instead of operating on the super-trellis, they store a single channel estimate for each current stage in the code trellis. We can derive the super-trellis receiver in our framework by simply combining stage-by-stage the code state nodes and channel state nodes, and the code factor nodes and channel factor nodes in the factor graph. When we derive the sum-product updates for this new factor graph we arrive at the exact modified BCJR algorithm that the super-trellis receiver uses. An important disadvantage of the super-trellis and per-survivor processing-based receivers is that the code trellis and the channel trellis must align stage-by-stage. This eliminates the possibility of interleaving between the code and the channel, which is necessary because convolutional codes are extremely susceptible to burst errors produced by correlated channel fades.

It is also important to consider carefully how the (soft) channel estimation portion of the factor graph interacts with the received data and code. Consider a blind equalization problem where the observation is of the form

$$y_i = \sum_{l=0}^{L-1} u_l b_{i-l} + n_i \tag{3.26}$$

where u_l are the multi-path propagation coefficients, n_i is additive white Gaussian noise, and $b_i = (-1)^{x_i}$ for $i \geq 0$ and $b_i = 0$ for $i < 0$. We assume that the multi-path propagation u_l coefficients are unknown, but remain fixed for the entire codeword. Further suppose that they are zero-mean, independent complex Gaussian random variables with known variance. We can factor the likelihood function for this ISI

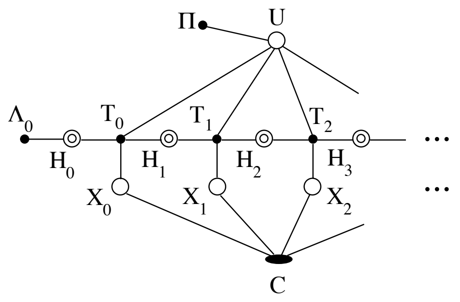


Figure 3.8: Factor graph for the multi-path fading channel. $C \triangleq \mathrm{I}\{\underline{x} \in \mathbb{C}\}$, $T_i \triangleq \mathrm{p}(y_i|\underline{u}, x_i, h_i)\ \mathrm{I}\{h_{i+1} = [x_i, \tilde{h}_i]\}$, $\Pi \triangleq \mathrm{p}(\underline{u})$, $\Lambda_0 \triangleq \mathrm{I}\{h_0 \text{ all zero}\}$.

channel as

$$G(\underline{x}, \underline{u}) = \mathrm{I}\{\underline{x} \in \mathbb{C}\} \cdot \mathrm{I}\{h_0 \text{ all zero}\} \cdot \prod_{i=0}^{N-1} \mathrm{p}(y_i|\underline{u}, x_i, h_i)\, \mathrm{I}\{h_{i+1} = [x_i, \tilde{h}_i]\} \cdot \mathrm{p}(u) \tag{3.27}$$

which gives the factor graph shown in Fig. 3.8.

The nominal messages $\mu_{T_i \to U}$ turn out to be Gaussian mixtures with 2^{L+1} terms corresponding to the 2^{L+1} possible values for $[x_i, h_i]$. Using the nominal messages directly, it turns out that the distribution for each coefficient after the first iteration has symmetry representing a 180° phase ambiguity due to the unknown data. By comparison, a blind adaptive equalizer, like the constant modulus algorithm, can use the shift properties of the data sequence to determine the channel $\underline{u}$ up to a single overall phase factor. The factor graph technique seems to loose this because the messages coming from the T nodes are treated as independent when actually messages from successive T nodes are related through their mutually adjacent hidden state.

If we estimate the conditional channel outputs given the current and past inputs $v(x_i, x_{i-1}, \ldots, x_{i-L})$ as originally suggested in [8] instead of the channel response $\underline{u}$, we can reduce the ambiguity to a single common phase factor. In our approach, this leads to the following factorization, shown in Fig. 3.9,

$$G(\underline{x}, \underline{v}) = \mathrm{I}\{\underline{x} \in \mathbb{C}\} \cdot \mathrm{I}\{h_0 \text{ all zero}\} \cdot \prod_{i=0}^{N-1} \mathrm{p}(y_i|x_i, h_i, \underline{v})\, \mathrm{I}\{h_{i+1} = [x_i, \tilde{h}_i]\} \tag{3.28}$$

where each element of $\underline{v}$ is associated with a particular value $[x_i, h_i]$ or equivalently a particular edge in the trellis. We assume no prior information is available about $\underline{v}$,

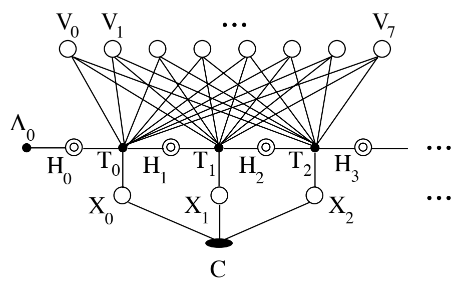


Figure 3.9: Factor graph for the multi-path fading channel using output estimates instead of coefficient estimates. $C \triangleq \mathrm{I}\{\underline{x} \in \mathbb{C}\}$, $T_i \triangleq \mathrm{p}(y_i|x_i, h_i, \underline{v})\ \mathrm{I}\{h_{i+1} = [x_i, \tilde{h}_i]\}$, $\Lambda_0 \triangleq \mathrm{I}\{h_0 \text{ all zero}\}$.

and note that, for given x_i and h_i, $\mathrm{p}(y_i|x_i, h_i, \underline{v})$ only depends on one element of $\underline{v}$. The algorithm proposed for blind equalization and turbo decoding in [21] uses this structure with an estimate canonical distribution for the conditional channel outputs combined with a channel estimator based on the Baum-Welch algorithm. It is also possible to further constrain the $\underline{v}$ to correspond to a valid combination of outputs for the channel model (3.26) as mentioned in [8].

Another interesting result from this analysis is the notion that the Baum-Welch algorithm, which is a realization of the expectation-maximization (EM) algorithm [14], can be derived in the factor graph framework. We believe that further study of the connection between the EM algorithm and factor graph-based iterative receivers might provide insight into the behavior of iterative receivers.

CHAPTER 4

Case Studies in Receiver Design

This chapter gives detailed receiver design examples for a number of specific systems. Usually, we present the design, perhaps with several variations, and then use simulation results to show how the design tradeoffs affect the performance. We also compare our designs to existing algorithms and briefly discuss code design issues for the specific receivers described.

4.1 Two-State Jamming Channel with Block Memory

This channel model typically arises when frequency-hopping spread spectrum is used in a partial band jamming environment. The jammer uses a fixed total power which it transmits as white Gaussian noise over a fraction ρ_J of our communication band. Because the carrier frequency is changing for each hop, some hops fall in jammed portions of the bandwidth while others are unjammed. We assume that each hop is either entirely jammed or entirely unjammed and that the hopping pattern is random and independent from the jammed bands. Thus, the probability of a hop being jammed is ρ_J. Further, we assume that each time we return to the same carrier frequency, there is a new, independent jamming state for that carrier. Let the channel state for the k^{th} hop be $u_k \in \{G, B\}$ where G is the "good" or unjammed state and B is the "bad" or jammed state. As usual, the channel input at each time is $x_i \in \{0, 1\}$.

The channel output y_i at time i is then

$$y_i = \sqrt{E_s} \cdot (-1)^{x_i} + \sigma(u_{\lfloor \frac{i}{m} \rfloor}) n_i \tag{4.1}$$

where m is the hop length, $\sqrt{E_s}$ is the transmitted signal energy per code symbol, $\sigma^2(u_k)$ is the noise variance in state u_k and n_i is a zero-mean, unit-variance Gaussian random variable. The noise variance $\sigma^2(u_k)$ is $N_0/2$ for $u_k = G$ and $N_0/2 + N_J/(2\rho_J)$ for $u_k = B$. We parameterize the jammer power by the full-band equivalent jammer power spectral density $N_J/2$ in order to take into account the fact that the jammer has a fixed total power.

Some closely related channels could be treated with minor adaptation. In the frequency-hopping multiple access case, collisions, which occur when different users use the same carrier frequency for a particular hop, could be modeled as partial-band jamming if we assume that the interfering user's signal is Gaussian noise. A single user in a time division multiple access (TDMA) system with pulsed jamming also can be modeled using a channel whose noise variance changes block by block.

4.1.1 Receiver Design

Consider our generic system when the code is a low-density parity check code and the channel is the two-state jamming channel. To design an iterative receiver, we begin by factoring the likelihood function $G(\underline{x}, \underline{u})$ from (3.6). Using the factorization techniques described in Chapter 3 for no-ISI channels with block memory and low-density parity check codes, we find

$$G(\underline{x}, \underline{u}) = \prod_{j=0}^{r-1} \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\} \cdot \prod_{i=0}^{N-1} \mathrm{p}(y_i | x_i, u_{\lfloor \frac{i}{m} \rfloor}) \cdot \prod_{k=0}^{\frac{N}{m}-1} \mathrm{p}(\underline{u}_k) \tag{4.2}$$

where $\underline{h}_j$ is the j^{th} row of the parity check matrix. From the channel model,

$$\mathrm{p}(y_i | x_i, u_{\lfloor \frac{i}{m} \rfloor}) = \frac{1}{\sqrt{2\pi\sigma^2(u_{\lfloor \frac{i}{m} \rfloor})}} \exp\left(-\frac{(y_i - \sqrt{E_s}(-1)^{x_i})^2}{2\sigma^2(u_{\lfloor \frac{i}{m} \rfloor})}\right) \tag{4.3}$$

and

$$\mathrm{p}(u_k) = \begin{cases} 1 - \rho_J & \text{if } u_k = G \\ \rho_J & \text{if } u_k = B \end{cases} \tag{4.4}$$

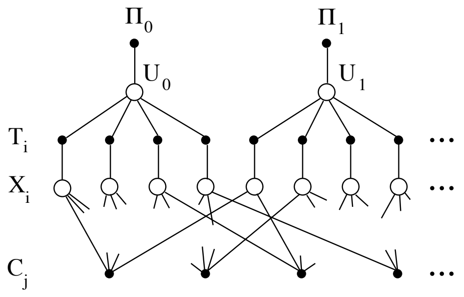


Figure 4.1: Factor graph for the two-state jamming channel with block memory and LDPC code $m = 4$. $T_i \triangleq \mathrm{p}(y_i|x_i, u_{\lfloor \frac{i}{m} \rfloor})$, $\Pi_k \triangleq \mathrm{p}(u_k)$, $C_j \triangleq \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\}$.

The factor graph is shown in Fig. 4.1. The T_i node updates are easy to realize directly and the Π_k node messages do not need to updated at all. As described in Chapter 2, there is also an efficient way to update the C_j nodes.

A practical schedule begins by setting the messages for the Π nodes, then updates groups of nodes repeatedly in the order: T, U, T, X, C, X. After each round of updates, the receiver checks to see if the bit-wise decisions for the X variables form a codeword. If they do, the algorithm stops and outputs the systematic bits of the codeword; otherwise it continues until some maximum number of iterations has been reached at which point it declares a failure and outputs the estimates for the bits. A possible variation would be to repeat the bit and code constraint updates X, C, X, more than once before adjusting the channel state estimate (T, U, T). Although this does not seem to make much difference here, in other cases it sometimes improves the performance even when the total number of code constraint updates is fixed [40].

Kang and Stark [31] have derived a similar receiver using more traditional methods for turbo codes on the frequency-hopping channel with partial band jamming. They divide the channel state estimation into separate components for the two component codes, but the overall effect is the same as the receiver above with an alternate update schedule. Their decoder equivalently updates the first component code, updates all the T and U nodes using that new information, decodes the second component codes, updates all the T and U nodes, and so on. We prefer to update all the code elements before updating the channel states.

4.1.2 Simulation Results

In order to evaluate the performance of the proposed receiver, we will compare it to perfect channel state information (CSI) and prior-only decoders. The perfect CSI decoder uses the objective function

$$G(\underline{x}) = \mathrm{p}(\underline{y}|\underline{x}, \underline{u})\, \mathrm{p}(\underline{x}) \tag{4.5}$$

factored as

$$G(\underline{x}) = \prod_{i=0}^{N-1} \mathrm{p}(y_i|x_i, u_i) \cdot \prod_{j=0}^{r-1} \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\} \tag{4.6}$$

where $\underline{h}_j$ is the j^{th} row of the parity check matrix for the code $\mathcal{C}$, and u_i is the known channel state for the i^{th} symbol. $\mathrm{p}(y_i|x_i, u_i)$ is as given in (4.3). The prior-only receiver ignores the fact that the symbols in each hop all have the same channel state. It uses the objective function

$$G(\underline{x}) = \mathrm{p}(\underline{y}|\underline{x})\, \mathrm{p}(\underline{x}) \tag{4.7}$$

factored as

$$G(\underline{x}) = \prod_{i=0}^{N-1} \mathrm{p}(y_i|x_i) \cdot \prod_{j=0}^{r-1} \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\} \tag{4.8}$$

where $\mathrm{p}(y_i|x_i)$ is the transition density based on the prior of the channel state

$$\begin{aligned} \mathrm{p}(y_i|x_i) &= \sum_{u_i \in \mathcal{U}} \mathrm{p}(y_i|x_i, u_i)\, \mathrm{p}(u_i) \\ &= (1-\rho_J)\, \mathrm{p}(y_i|x_i, G) + \rho_J\, \mathrm{p}(y_i|x_i, B). \end{aligned} \tag{4.9}$$

These decoders both have the factor graph shown in Fig. 4.2 except that the functions for the T_i nodes depend on what information is available to the receiver. For both cases, we adopt a natural LDPC decoder update schedule, where the T nodes are updated once and then the X and C nodes are alternately updated until the X decisions form a codeword or the maximum number of iterations is reached.

For hop length $m = 16$, the performance of the proposed receiver is essentially identical to that of the perfect CSI decoder indicating that the proposed receiver realizes most of the gain available for channel estimation. For hop length $m = 8$, Fig.

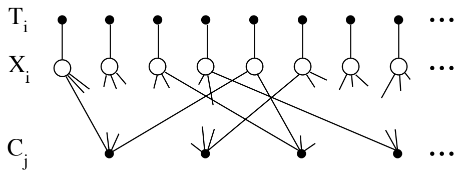


Figure 4.2: Factor graph for alternate receivers for the two-state jamming channel with block memory. $C_j \triangleq \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\}$; CSI $T_i \triangleq \mathrm{p}(y_i|x_i, u_i)$; prior-only $T_i \triangleq \mathrm{p}(y_i|x_i)$.

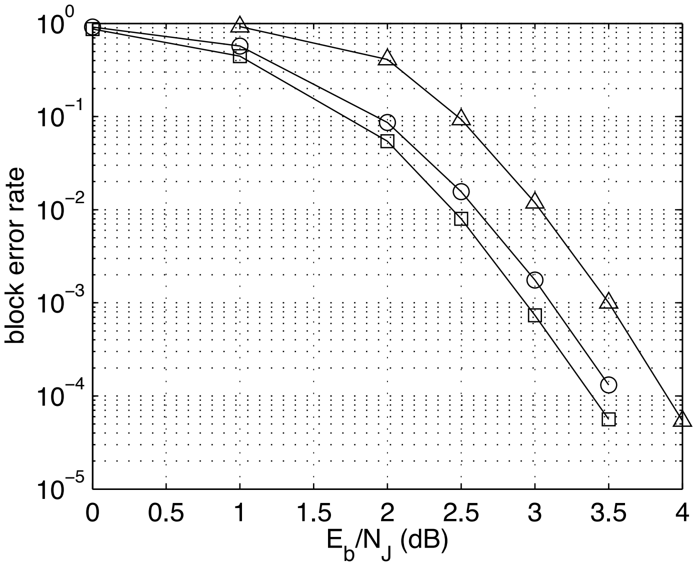


Figure 4.3: Receiver performance comparison for the two-state jamming channel with block memory. '□' side information; '○' channel estimation; '△' no channel estimation. This is a block length 1024, rate 1/2 LDPC code on a channel with jamming fraction $\rho_J = 0.75$, hop length $m = 8$, and unjammed SNR 10 dB.

4.3 shows the block error rate of the receivers as a function of the signal-to-jamming noise ratio (SJNR) E_b/N_J for a fixed SNR (E_b/N_0) of 10 dB. The channel estimation receiver looses about 0.1 dB relative to the perfect side information case. Because soft outputs are available and the channel information only provides improved reliability estimates and thus is not essential to the decoding process, the prior-only decoder only looses an additional 0.4 dB relative to the proposed receiver.

As expected, there is a clear trade-off between channel estimation and effective diversity. The perfect side information receiver performs best for hop length $m = 1$ and its performance decreases with the effective diversity as the hop length increases.

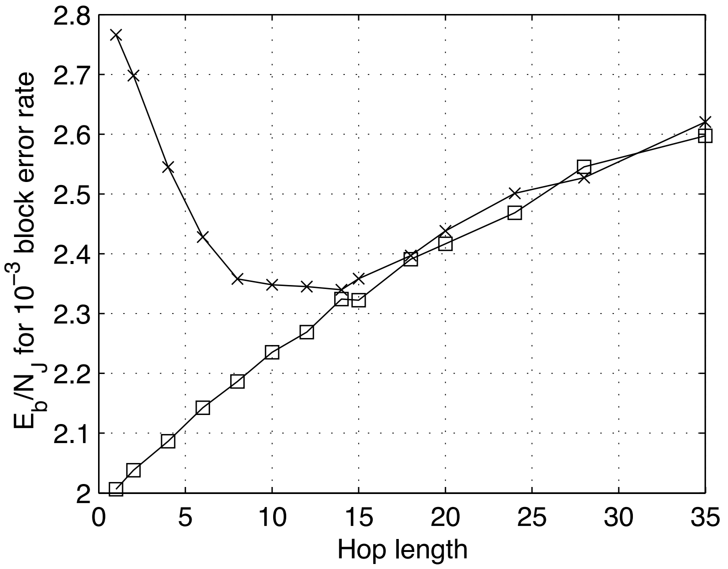


Figure 4.4: Required SJNR E_b/N_J as a function of hop length m for the two-state jamming channel with block memory. '$\square$' side information; '$\times$' channel estimation. This is a block length 2520, rate 1/2 LDPC code on a channel with jamming fraction $\rho_J = 0.75$ and unjammed SNR 10 dB.

The channel estimation receiver suffers from poor channel estimation at small hop lengths and at larger hop lengths tracks the performance of the perfect CSI receiver. Figure 4.4 shows the required SJNR for 10^{-3} block error rate as a function of the hop length m. The gap between the perfect side information receiver and proposed channel estimation receiver is largest for $m = 1$. The occasional crossing of the curves is due to statistical uncertainty in the simulation results. For hop lengths larger than $m = 15$, the performance is actually essentially the same for the channel estimation and side information receivers.

It would be interesting to know directly how well the channel estimation works. An important question is whether the channel estimation improves as iterations update the channel estimate values. If the estimate is essentially perfect after only one round of channel estimation, then further computation is unnecessary. Figure 4.5 shows two sample profiles of the number of channel estimation errors at each stage for different individual runs. Runs where the correct codeword was found at different total numbers of iterations are shown for comparison. Note that most trials find the correct codeword in less than 15 iterations. We plot both the number of incorrect

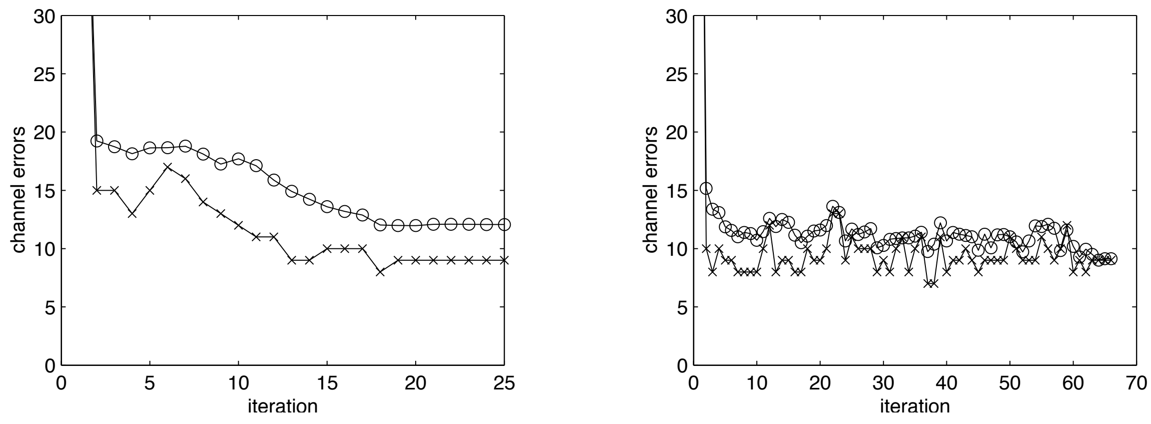


Figure 4.5: Channel estimation error as a function of iteration for two typical trials. '∘' soft errors, '×' hard errors. The runs take different numbers of iterations to converge to a valid codeword. This is a block length 2520, rate 1/2 LDPC code on a channel with jamming fraction $\rho_J = 0.75$, SJNR 2 dB, hop length $m = 8$ and unjammed SNR 10 dB.

channel estimates and, because the channel estimates are actually soft, the total normalized likelihood of all the incorrect channel values. This gives some idea of whether the estimates are decisive or ambiguous. If all the estimates were decisive (i.e. either 0 or 1 likelihoods), then we would get the same result for the soft values as for counting errors. The difference shows that even when further iterations seem to increase the number of errors, usually the soft estimates are actually improving.

In general, the initial number of estimation errors, before the code is used to compute bit likelihoods, is quite large–about one third of the hops. After one iteration, the bit likelihoods are incorporated and the number of errors drops dramatically. Even when a valid codeword has been found, some channel estimation errors often remain. This behavior indicates that including the soft channel estimation in the iteration is valuable, but that correct channel estimates are not essential for good performance.

Although it is not our primary focus, it is interesting to look at how the number of bit errors in a codeword varies as the algorithm proceeds. Figure 4.6 shows the number of hard bit errors and the total soft likelihood for the incorrect values after each iteration. The number of errors tends to decrease rapidly in the first few iterations, then after some slow progress, the last few errors are eliminated quickly. From the soft information, we see that in the end, all the bits have quite high certainty, and

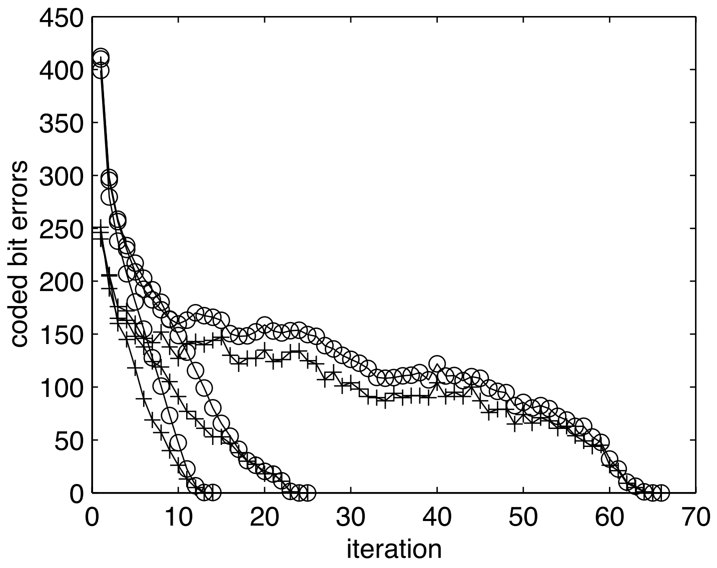


Figure 4.6: Bit errors as a function of the number of iterations for three typical trials. '$\circ$' soft errors, '+' hard errors. The runs take different numbers of iterations to converge to a valid codeword. This is a block length 2520, rate 1/2 LDPC code on a channel with jamming fraction $\rho_J = 0.75$, SJNR 2 dB, hop length $m = 8$ and unjammed SNR 10 dB.

the algorithm is basically flipping bits that cause problems. Note that neither the hard error count, nor the soft information is monotonically decreasing.

Although this example is quite simple, it behaves entirely as expected. In particular, the trade-off between effective diversity and channel estimation is immediately evident and it is clear that as the hop length gets larger, the performance of the joint channel estimation and decoding receiver approaches that of the perfect CSI receiver. We will see this same basic behavior repeatedly as we consider more complex channel models.

4.2 Two-State Jamming Channel with Markov Memory

The two-state channel with Markov memory was first proposed to model bursty errors [23] in telephone networks. The concept of memory was defined and more detailed analysis of the channel capacity presented in [52]. We consider a pulsed jamming model for comparison with the partial band jamming model considered above. Here the jamming fraction ρ_J represents the average fraction of time that a pulsed jammer is active and assumes that the jammer follows some Markov model. Although it is unlikely that a hostile jammer would adopt such a predictable strategy, unintentional interferers might have this behavior. Also, a close cousin of this channel, the fading channel with Markov memory, occurs frequently in mobile communication systems.

Just as for the two-state jamming channel with block memory, there are two possible channel states corresponding to either jammed or unjammed symbols. Precisely,

$$y_i = \sqrt{E_s} \cdot (-1)^{x_i} + \sigma(u_i) n_i \tag{4.10}$$

where y_i is the channel output at time i, x_i is the unknown channel input bit at time i, $\sigma^2(u_i)$ is the noise variance for channel state u_i, and n_i is an additive Gaussian noise sample with mean 0 and variance 1. As for the two-state jamming channel, the noise variance $\sigma^2(u_i)$ in the good (unjammed) state is $N_0/2$ and in the bad (jammed) state is $\frac{1}{2}(N_J/\rho_J + N_0)$. Instead of the channel state being fixed for blocks of m symbols, the channel state can change between any two symbols and the state process is modeled as a Markov chain. We parameterize the channel according to the jamming fraction ρ_J and the memory $\mu = 1 - g - b$, originally defined in [52]. The desired steady state distribution is $\mathrm{p}(B) = \rho_J$ and $\mathrm{p}(G) = 1 - \rho_J$. In order to make the Markov chain stationary, we choose $\mathrm{p}(u_0)$ equal to the steady state distribution. The transition probabilities are

$$\mathrm{p}(u_i = B | u_{i-1} = G) = b = \rho_J(1 - \mu) \tag{4.11}$$

$$\mathrm{p}(u_i = G | u_{i-1} = B) = g = (1 - \rho_J)(1 - \mu) \tag{4.12}$$

$$\mathrm{p}(u_i = G | u_{i-1} = G) = 1 - b \tag{4.13}$$

$$\mathrm{p}(u_i = B | u_{i-1} = B) = 1 - g. \tag{4.14}$$

Our primary motivation in considering this channel is to compare it to the two-state jamming channel with block memory in order to relate the hop length and the memory parameter μ.

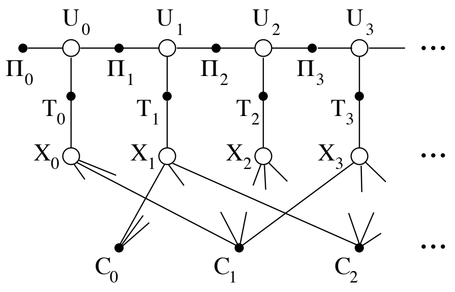


Figure 4.7: Factor graph for the two-state jamming channel with Markov memory. $\Pi_k \triangleq \mathrm{p}(u_k|u_{k-1})$, $\Pi_0 \triangleq \mathrm{p}(u_0)$, $T_i \triangleq \mathrm{p}(y_i|x_i, u_i)$, $C_j \triangleq \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\}$

4.2.1 Receiver Design

Suppose we communicate over this channel with a LDPC code and an iterative receiver. The objective function $G(\underline{x}, \underline{u})$ can be factored as

$$G(\underline{x}, \underline{u}) = \prod_{j=0}^{r-1} \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\} \cdot \prod_{i=1}^{N-1} \frac{1}{\sqrt{2\pi\sigma^2(u_i)}} \exp\left(-\frac{(y_i - \sqrt{E_s}(-1)^{x_i})^2}{2\sigma^2(u_i)}\right) \cdot \\ \mathrm{p}(\underline{u}_0) \cdot \prod_{k=1}^{N-1} \mathrm{p}(\underline{u}_k|\underline{u}_{k-1}) \quad (4.15)$$

which has the associated factor graph shown in Fig. 4.7. The updates for all the nodes are straightforward using the standard formulas. The schedule is similar to that described for the two-state jamming channel with block memory, except that the state nodes U_i are updated as a chain. That is, we begin with U_0, propagate its new state information to U_1 by updating Π_1, then update U_1, and so on. The Π and U nodes are updated alternately down the chain until we get to the last U node. We then reverse the process, starting from the last U node and updating the nodes along the chain until we reach U_0. This update schedule is equivalent to using the BCJR algorithm to estimate the channel state Markov chain.

Various researchers have considered similar channels with turbo codes and designed receivers using the traditional SISO algorithm approach. In [32], turbo decoders modified to operate on a two-state jamming model with Markov memory are presented. The approach effectively adds an additional trellis-type channel estimator and thus is similar to the receiver presented here. However, we believe that the factor graph development makes the design more transparent and suggests scheduling varia-

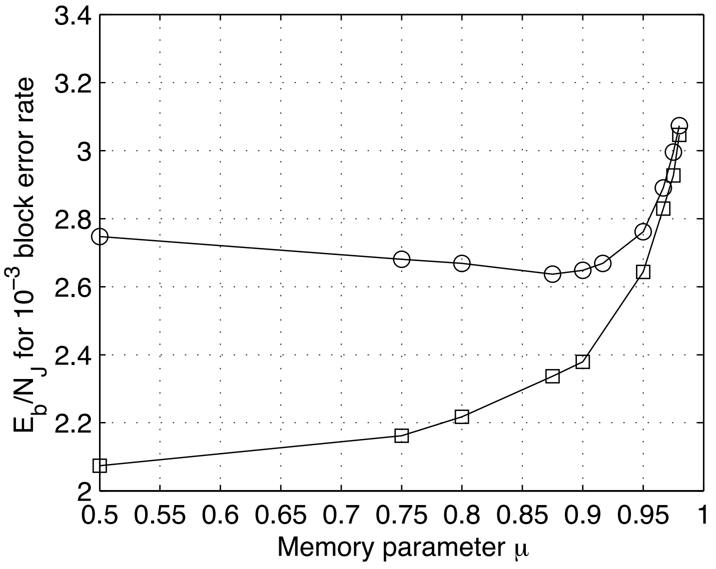


Figure 4.8: Required SJNR E_b/N_J vs. memory μ for the two-state jamming channel with Markov memory. '○' proposed receiver, '□' perfect CSI. This was a block length 2520, rate 1/2 LDPC code on a channel with unjammed SNR $E_b/N_0 = 10$ dB, and jamming fraction $\rho_J = 0.75$.

tions that might improve performance. A receiver for turbo codes on a fading channel with Markov memory based on the super-trellis approach mentioned in section 3.3 is presented in [20]. Garcia-Frias and Villasenor [22] have also proposed an alternate receiver that uses a separate soft channel estimator more like our proposed receiver. Previous work has not considered the effects of memory length on performance.

4.2.2 Simulation Results

When we plot the required SJNR vs. the memory parameter μ in Fig. 4.8, we get quite a different picture than we did for the channel block memory. However, after some consideration, we see that it has the same basic properties. As expected, the required SJNR for the perfect CSI receiver increases with the memory. The required SJNR for the proposed receiver decreases slightly as the channel estimation improves and then increases as the decrease in effective diversity due to longer memory takes its toll. Both the perfect CSI and proposed receiver curves increase sharply as the memory gets close to 1 because this represents the maximum possible memory.

In order to compare the Markov memory and block memory cases, we need a

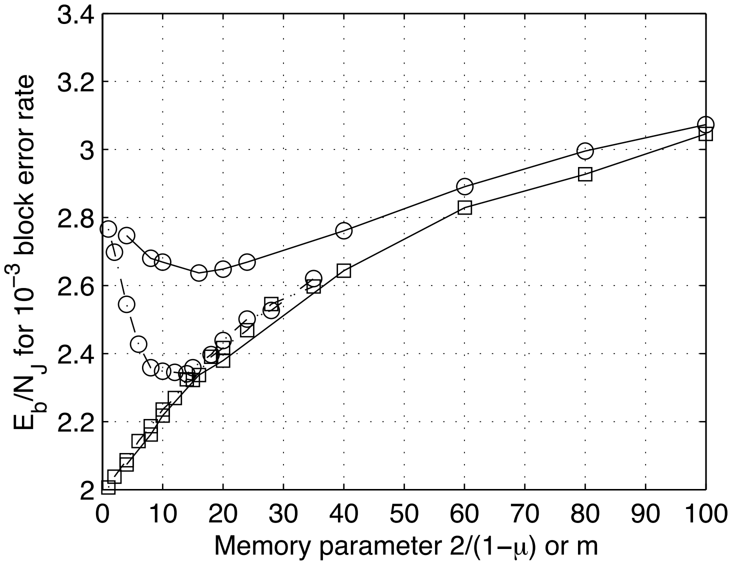


Figure 4.9: Required SJNR E_b/N_J vs. memory length $2/(1-\mu)$ or m for two-state jamming channels. '○' proposed receivers, '□' perfect CSI. Solid lines denote Markov memory; dash-dotted lines are for block memory. The block length 2520, rate 1/2 LDPC code, unjammed SNR $E_b/N_0 = 10$ dB, and jamming fraction $\rho_J = 0.75$ were the same for both channels.

compatible way to express the memory length. We propose to use the hop length for the block memory case. Since the memory parameter μ varies between 0 and 1 (for persistent memory), and the maximum memory occurs when $\mu = 1$, it seems reasonable that the memory for the Markov case should be related to $1/(1-\mu)$. Empirically, we find that $2/(1-\mu)$ works well. More theoretical justification for this result is a topic for further study. When we plot the required SJNR vs. the memory in terms of $2/(1-\mu)$, the perfect CSI curves for block and Markov line up very nicely as shown in Fig. 4.9. The optimal hop length (again in terms of $2/(1-\mu)$) for the iterative receiver on the Markov memory channel, about 16 symbols, is a little larger than the optimum hop length for the block memory channel. At the optimal hop lengths, the iterative receiver does more than 0.2 dB worse than it did on the block memory channel. This makes sense because, with block memory, the receiver knew that the state could only change at the end of each block. With Markov memory, the channel state can change at any time which makes state estimation more difficult.

4.3 Coherent Rayleigh Fading Channel with Block Memory

Frequency-hopping spread spectrum is often applied to combat frequency-selective fading that changes very slowly with time. By changing frequently to different carrier frequencies, the technique avoids getting stuck with a severely faded channel. If the coherence bandwidth is larger than the signal bandwidth for any particular hop, then each hop experiences a flat Rayleigh fading level. We assume that the coherence time of the channel is much longer than the hop duration, so the fade level is approximately fixed throughout each hop. Thus, each hop experiences an independent Rayleigh fade level for its entire duration.

The same channel model applies to time-division multiple access (TDMA) in a slow time-selective Rayleigh fading channel. If the time-variation of the channel is much slower than the TDMA burst duration, but much faster than the time between a particular user's bursts, then the fade level is fixed during the burst and essentially independent from burst to burst. This duality allows us to use the same strategies for both channels by dealing with the underlying, more abstract, block fading channel model.

Suppose that the channel hop length is m symbols and that we use BPSK signaling. The channel output is

$$y_i = u_{\lfloor \frac{i}{m} \rfloor}(-1)^{x_i}\sqrt{E_s} + n_i \tag{4.16}$$

where $\sqrt{E_s}$ is the average signal energy, and n_i is a real white Gaussian random sequence with mean 0 and variance $N_0/2$. The fading levels u_k are a sequence of independent Rayleigh random variables with pdf's

$$\mathrm{p}(u_k) = 2u_k e^{-u_k^2}. \tag{4.17}$$

4.3.1 Receiver Design

The receiver structure for LDPC codes is essentially the same as for the two-state jamming channel with block memory above. We factor $G(\underline{x}, \underline{u})$ as

$$G(\underline{x},\underline{u}) = \prod_{j=0}^{r-1} \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\} \cdot \prod_{i=0}^{N-1} \mathrm{p}(y_i|x_i, u_{\lfloor \frac{i}{m} \rfloor}) \cdot \prod_{k=0}^{\frac{N}{m}-1} \mathrm{p}(\underline{u}_k)$$
$$\propto \prod_{j=0}^{r-1} \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\} \cdot \prod_{i=0}^{N-1} \exp\left(-\frac{(y_i - u_{\lfloor \frac{i}{m} \rfloor}(-1)^{x_i}\sqrt{E_s})^2}{N_0}\right) \prod_{k=0}^{\frac{N}{m}-1} 2u_k e^{-u_k^2} \tag{4.18}$$

where we have taken the normalizing terms for $\mathrm{p}(y_i|x_i, u_{\lfloor \frac{i}{m} \rfloor})$ as the constant of proportionality which does not affect the optimization. The factor graph for this is the same as Fig. 4.1 except that now nodes T_i represent the functions

$$\exp\left(-\frac{(y_i - u_{\lfloor \frac{i}{m} \rfloor}(-1)^{x_i}\sqrt{E_s})^2}{N_0}\right)$$

and nodes Π_k represent the functions

$$2u_k e^{-u_k^2}.$$

Because the channel state space is continuous, the node updates involving the channel state nodes become problematic. To find realizable node updates for the T_i and U_k nodes, we start with the nominal node updates. Beginning with $\mu_{T_i \to U_k}$, we have

$$\mu_{T_i \to U_k}(u_k) = \exp\left(-\frac{(y_i - u_k\sqrt{E_s})^2}{N_0}\right)\mu_{X_i \to T_i}(0) + \exp\left(-\frac{(y_i + u_k\sqrt{E_s})^2}{N_0}\right)\mu_{X_i \to T_i}(1). \tag{4.19}$$

This could easily be represented in terms of the parameters y_i, $\mu_{X_i \to T_i}(0)$, and $\mu_{X_i \to T_i}(1)$. The other two variables, E_s and N_0, are the same for all i and are known. Unfortunately, if we take this approach, the message $\mu_{U_k \to T_j}$ becomes

$$\mu_{U_k \to T_j}(u_k) = \mu_{\Pi_k \to U_k}(u_k) \cdot \prod_{\substack{i=km \\ i \neq j}}^{km+m-1} \mu_{T_i \to U_k}(u_k)$$
$$= 2u_k e^{-u_k^2} \prod_{\substack{i=km \\ i \neq j}}^{km+m-1} \left[\exp\left(-\frac{(y_i - u_k\sqrt{E_s})^2}{N_0}\right)\mu_{X_i \to T_i}(0) + \right.$$
$$\left. \exp\left(-\frac{(y_i + u_k\sqrt{E_s})^2}{N_0}\right)\mu_{X_i \to T_i}(1)\right] \tag{4.20}$$

which is not particularly difficult to compute for any fixed value of u_k, but is very unwieldy to use in later update equations. To underscore this point, note that

$$\mu_{T_i \to X_i}(x_i) = \int\limits_{\mathcal{U}} \exp\left(-\frac{(y_i - u_k(-1)^{x_i}\sqrt{E_s})^2}{N_0}\right) \mu_{U_k \to T_i}(u_k)\, du_k \tag{4.21}$$

which is very complicated if $\mu_{U_k \to T_i}$ is not a relatively simple function.

To solve this problem, we introduce two simple canonical distributions for $\mu_{U_k \to T_j}$. One possibility is to discretize the fading level to one of L preselected levels $\hat{u}_l$

$$\mu'_{U_k \to T_j}(u_k) = \sum_{l=0}^{L-1} a_l \delta(u_k - \hat{u}_l). \tag{4.22}$$

A natural way to choose the weights a_l is to sample the nominal message $\mu_{U_k \to T_j}$ as given in (4.20) at the discretization levels $\hat{u}_l$. The parameters to be passed are just the weights a_l since the levels $\hat{u}_l$ are fixed and known in advance. This update could be realized by passing samples of the messages $\mu_{T_i \to U_k}$ at the points $\hat{u}_l$ and then multiplying them just as if U_k were a discrete variable with L states. With the canonical distribution (4.22), the update for $\mu_{T_i \to X_i}$ becomes

$$\mu_{T_i \to X_i}(x_i) = \sum_{l=0}^{L-1} a_l \exp\left(-\frac{(y_i - \hat{u}_l(-1)^{x_i}\sqrt{E_s})^2}{N_0}\right) \tag{4.23}$$

which is quite practical to compute.

A second possibility is to simply estimate the fade level. Because small changes in the fade level do not have a drastic effect on the reliabilities and the fade level is known to be positive, this works well. In the canonical distribution framework, using an estimate corresponds to choosing the canonical distribution

$$\mu'_{U_k \to T_j}(u_k) = \delta(u_k - \hat{u}_{k,j}) \tag{4.24}$$

where $\hat{u}_{k,j}$ is the estimate for u_k to be used by T_j. A simple approach is to begin with the nominal message $\mu_{U_k \to T_j}$, assume that the hard decisions on the bits are correct, and then use a maximum likelihood estimator. The resulting estimate, obtained by applying the assumptions to the nominal message and setting its derivative to 0, is

$$\hat{u}_{k,j} = \frac{1}{2}\left[\frac{1}{\sqrt{E_s}(m-1+\frac{N_0}{E_s})}\sum_{\substack{i=km\\ i\neq j}}^{km+m-1}|y_i| + \sqrt{\left(\frac{1}{\sqrt{E_s}(m-1+\frac{N_0}{E_s})}\sum_{\substack{i=km\\ i\neq j}}^{km+m-1}|y_i|\right)^2 + \frac{2}{(1+(m-1)\frac{E_s}{N_0})}}\right] \tag{4.25}$$

which is approximately

$$\hat{u}_{k,j} = \frac{1}{(m-1)\sqrt{E_s}}\sum_{\substack{i=km\\ i\neq j}}^{km+m-1}|y_i| \tag{4.26}$$

when the average SNR E_s/N_0 is large. Without the effect of the Rayleigh prior distribution, this result holds independent of the average SNR.

We use the same update schedule as for the two-state jamming channel with block memory above.

4.3.2 Simulation Results

For comparison, we again consider a perfect channel side information receiver. The structure is the same as in (4.6) except that now the conditional probability $\mathrm{p}(y_i|x_i,u_i)$ is

$$\mathrm{p}(y_i|x_i,u_i) = \frac{1}{\sqrt{\pi N_0}}\exp\left(-\frac{(y_i - u_i(-1)^{x_i}\sqrt{E_s})^2}{N_0}\right). \tag{4.27}$$

Note that the perfect CSI receiver has the exact continuous-valued fading levels u_i for each channel output provided clairvoyantly by the channel.

The Rayleigh fading channel with block memory repeats the now familiar pattern of trading channel estimation against effective diversity. In Fig. 4.10, we see that for this case the optimal hop length is somewhat shorter than for the two-state jamming channel. For the quantized canonical distribution, the optimal hop length is about 4 symbols. For the estimate canonical distribution it is a little longer–about 8 symbols. Because fading affects the mean output magnitude and not just the variance as for jamming, estimating the fading state is easier. Also, severe fading is in some sense

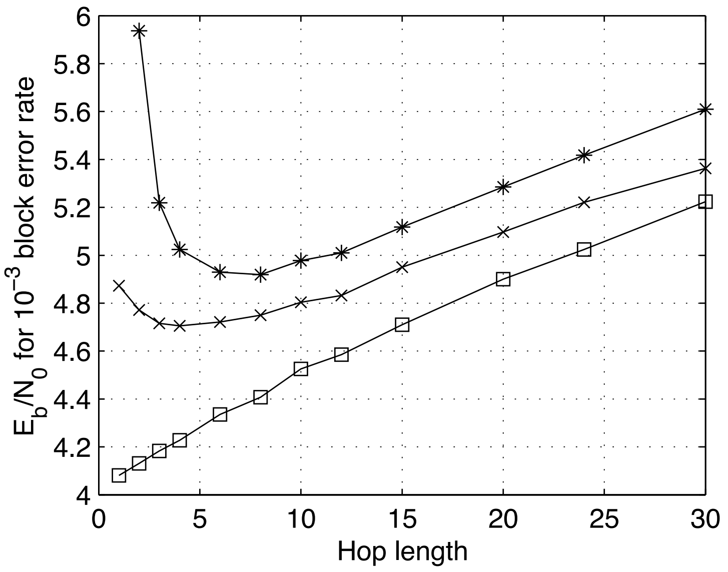


Figure 4.10: Performance vs. hop length for coherent Rayleigh fading with different canonical distributions

accounted for symbol-by-symbol in the conditional distribution. For example, a very small output value must be due to either a small fade level or a very severe noise sample. In either case, the resulting bit likelihood will reflect a low reliability. The quantized canonical distribution (CD) does uniformly better than the estimate CD, presumably because maintaining more soft information improves performance. The extremely poor performance of the estimate CD receiver for small hop lengths is due to the fact that the estimator is very sensitive to noise when it has few observations to work with. This suggests that if complexity demands that a simple estimation scheme be used, some bias towards smaller, or more pessimistic, estimates of the fading level should be used. For example, a minimum mean square error estimator, which is biased toward zero depending on the observation noise variance and number of observations, might be more suitable. Note, however, that these estimators do not use the information provided by the code, and therefore remove the possibility of iterative improvement between the channel estimator and the decoder.

Comparing the channel estimation errors for the receivers as a function of block length adds some further insights. We consider the mean squared estimation error between the actual fade level and the estimated fade level for the estimate CD. For

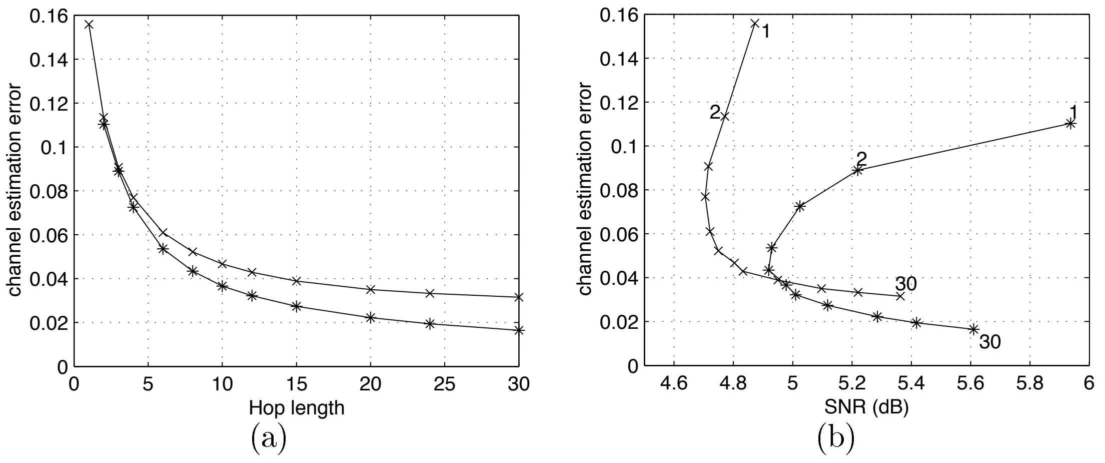


Figure 4.11: Channel estimation error for coherent Rayleigh fading with different canonical distributions. '$\times$' quantized CD, '$*$' estimate CD. (a) shows the error as a function of the hop length for the SNR that gives a block error rate of 10^{-3}, (b) shows the error vs. the SNR where the curves are parametric in the hop length. The numbers near the curves are the hop lengths for the adjacent points. This is a block length 2520, rate 1/2 LDPC code on a Rayleigh fading channel.

the quantized CD, we take as the squared error the weighted average of the squared error between the actual fade level and the quantization points where the weights are the distribution parameters a_l

$$\bar{e} = \sum_l a_l (u - \hat{u}_l)^2. \tag{4.28}$$

The final mean squared error for both receivers is plotted in two ways in Fig. 4.11. In Fig. 4.11a, it is shown as a function of the hop length and in Fig. 4.11b it is shown as a function of the SNR where the curves are parametric in the hop length. The data points all correspond to block error rates of 10^{-3}. As the hop length increases, the estimate CD data has a significantly smaller mean-squared error. The fact that the performance differs substantially for the same SNR with different hop lengths indicates that hop length is more significant in determining the channel estimation error. For the lower halves of the curves, where the hop length is large, it is especially clear that even when the SNR is the same, the estimate CD has a smaller mean square error and accomplishes this with a smaller hop length. Since the quantized CD receiver still performs better, we conclude that the channel estimation error, at least as we have defined it, is not particularly important to performance. Maintaining good soft information contributes much more to good performance.

4.4 Non-Coherent Flat Rayleigh Fading Channel with Block Memory

In our discussion of the flat Rayleigh fading channel with block memory above, we assumed that the carrier phase was known to the receiver. In real systems, this is often not true because the short hop duration makes phase acquisition difficult. Furthermore, phase acquisition overhead becomes significant when the hop length is of the same order as the acquisition time. To account for this issue, we consider a non-coherent flat Rayleigh fading channel with block memory, where the receiver must estimate the fixed, unknown phase offset for each hop in addition to the fade magnitude. This gives a block memory channel with complex output

$$y_i = u_{\lfloor \frac{i}{m} \rfloor}(-1)^{x_i}\sqrt{E_s} + n_i \tag{4.29}$$

where u_k is complex Gaussian with mean 0 and variance $1/2$ and n_i is complex Gaussian with mean 0 and variance $N_0/2$ in each dimension. It is well-known that a complex Gaussian random variable with iid real and imaginary parts has a Rayleigh-distributed magnitude and a uniformly-distributed phase. Thus, the channel is essentially the same as in the coherent flat Rayleigh fading case above except that the phase is unknown and uniformly distributed.

4.4.1 Receiver Design

Receiving messages on the non-coherent Rayleigh fading channel is fundamentally more difficult than the previous cases that we have considered because the bit decisions depend completely on the channel estimate. Without an estimate of the channel, the bits are all equally likely. Furthermore, with a very bad phase estimate, all the bit decisions will be wrong and yet their supposed reliabilities may still be high.

Although a true maximum likelihood decoder can use the correlation of the codeword bits directly for decoding on the non-coherent channel, iterative receivers need some way to get the first round of channel estimates to start the decoding. Therefore, for this example we use an LDPC code with pilot symbols. For some original LDPC code $\mathcal{C}'$ and channel hop length m, the code with pilot symbols $\mathcal{C}$ is constructed by inserting a "0" pilot symbol before every $m-1$ symbols in each codeword from $\mathcal{C}'$. These *a priori* known bits help form an initial phase estimate for each hop. The value is chosen to be 0 so that the code $\mathcal{C}$ is still linear, although this is not necessary for the receiver derivation.

The objective function factors much as in (4.18) except that we split the fade level into amplitude and phase components $u_{\alpha,k}$ and $u_{\theta,k}$ to get

$$G(\underline{x},\underline{u}) = \prod_{l=0}^{\frac{N}{m}-1} \mathrm{I}\left\{x_{lm}=0\right\} \prod_{j=0}^{r-1} \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\} \prod_{i=0}^{N-1} \mathrm{p}(y_i | x_i, u_{\alpha,\lfloor\frac{i}{m}\rfloor}, u_{\theta,\lfloor\frac{i}{m}\rfloor}) \cdot \prod_{k=0}^{\frac{N}{m}-1} \mathrm{p}(u_{\alpha,k}) \prod_{k=0}^{\frac{N}{m}-1} \mathrm{p}(u_{\theta,k}) \quad (4.30)$$

where $\underline{h}_j$ is the j^{th} row of the parity check matrix for the LDPC code $\mathcal{C}'$ and the $\mathrm{I}\left\{x_{lm}=0\right\}$ factors account for the pilot symbols. Substituting the exact distributions and dropping constant factors

$$G(\underline{x},\underline{u}) = \prod_{l=0}^{\frac{N}{m}-1} \mathrm{I}\left\{x_{lm}=0\right\} \prod_{j=0}^{r-1} \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\} \cdot \prod_{i=0}^{N-1} \exp\left(-\frac{1}{N_0}\left|y_i - u_{\alpha,\lfloor\frac{i}{m}\rfloor} \exp(ju_{\theta,\lfloor\frac{i}{m}\rfloor})(-1)^{x_i}\sqrt{E_s}\right|^2\right) \cdot \prod_{k=0}^{\frac{N}{m}-1} 2u_{\alpha,k}e^{-u_{\alpha,k}^2} \prod_{k=0}^{\frac{N}{m}-1} \mathrm{I}\left\{u_{\theta,k} \in [0, 2\pi)\right\}. \quad (4.31)$$

The factor graph corresponding to this factorization is shown in Fig. 4.12.

We could also have separated the complex fade level into its real and imaginary components. In this case, we would have ambiguity in the sign of both the real and imaginary parts, so both would be multi-modal. The product distribution that would be assumed by the T nodes would often have four peaks instead of the two that we will find for the magnitude and phase approach. In some sense, there are too many degrees of freedom in the real and imaginary model. When we estimate the real and imaginary parts, both can be either positive or negative. For the magnitude

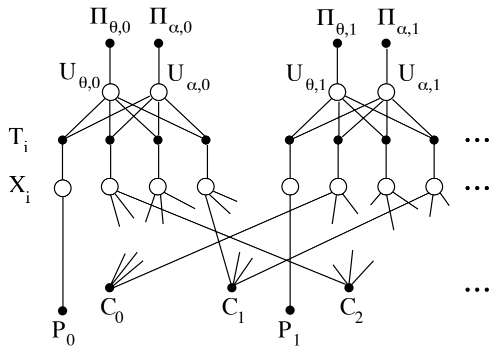


Figure 4.12: Factor graph for the non-coherent Rayleigh fading channel with block memory. $\Pi_{\theta,k} \triangleq \mathrm{p}(u_{\theta,k})$, $\Pi_{\alpha,k} \triangleq \mathrm{p}(u_{\alpha,k})$, $T_i \triangleq \mathrm{p}(y_i|x_i, u_{\theta,\lfloor \frac{i}{m} \rfloor}, u_{\alpha,\lfloor \frac{i}{m} \rfloor})$, $P_l \triangleq \mathrm{I}\{x_{lm} = 0\}$, $C_j \triangleq \mathrm{I}\{\underline{h}_j \cdot \underline{x} = 0\}$

and phase, the magnitude should always be positive. Another issue is that, for the real and imaginary parts, the coupling between the two is essential to get the phase. In the magnitude and phase approach, the two are less strongly connected. We use magnitude and phase in the sequel.

The updates for the U_α, U_θ, and T nodes, which correspond to the magnitude and phase variables and the channel conditional probability factors, are difficult because both $U_{\alpha,k}$ and $U_{\theta,k}$ are continuous-valued and they both affect the T node update in a relatively complex way. Our approach is to select a very simple canonical distribution and parameter selection scheme for the outgoing messages from the U_α nodes. This will allow us to use the nominal T node messages and a variety of canonical distributions for the phase node messages. Some choices were made specifically so that the receivers would have channel state node update complexities that are roughly linear in the hop length m.

Therefore, we choose an estimate canonical distribution as in (4.24) for $\mu_{U_{\alpha,k} \to T_j}$. The parameter estimation scheme (4.26) for the fade level in the coherent flat Rayleigh fading case still makes sense because taking the magnitude of y_i eliminates the effect of both the unknown phase and the data.

To design canonical distributions for the phase node messages, we begin with the nominal message $\mu_{U_{\theta,k} \to T_j}(u_{\theta,k})$. Presuming the estimate canonical distribution for

$\mu_{U_{\alpha,k}\to T_j}$, we get

$$\begin{aligned}\mu_{U_{\theta,k}\to T_j}(u_{\theta,k}) &= \prod_{\substack{i=km\\ i\neq j}}^{km+m-1} \mu_{T_i\to U_{\theta,k}}(u_{\theta,k}) \\ &= \prod_{\substack{i=km\\ i\neq j}}^{km+m-1} \exp\left(-\frac{\left|y_i - \hat{u}_{\alpha,k,i}\exp(ju_{\theta,k})\sqrt{E_s}\right|^2}{N_0}\right)\mu_{X_i\to T_i}(0) + \\ &\qquad \exp\left(-\frac{\left|y_i + \hat{u}_{\alpha,k,i}\exp(ju_{\theta,k})\sqrt{E_s}\right|^2}{N_0}\right)\mu_{X_i\to T_i}(1)\end{aligned} \tag{4.32}$$

so each $\mu_{T_i\to U_{\theta,k}}(u_{\theta,k})$ contributes two terms with peaks at phases 180° apart. The relative heights of the peaks depend on the bit likelihoods $\mu_{X_i\to T_i}$. Because the underlying phase is actually fixed for the whole block, $\mu_{U_{\theta,k}\to T_j}$ should also have two peaks roughly 180° apart. A key strength of canonical distributions is that they can capture this qualitative behavior. The bimodal canonical distribution

$$\mu'_{U_{\theta,k}\to T_j}(u_{\theta,k}) = a_1\delta(u_{\theta,k} - \hat{u}_{\theta,k}) + a_2\delta(u_{\theta,k} - \hat{u}_{\theta,k} + \pi) \tag{4.33}$$

captures the ambiguity in the phase estimate in a simple way. Although the phase estimate parameter $\hat{u}_{\theta,k}$ could depend on the destination node T_j, computational cost motivates us to use only a single estimate for all the messages leaving $U_{\theta,k}$. Given the phase estimate $\hat{u}_{\theta,k}$, the weights a_1 and a_2 can be computed by sampling the nominal messages at $u_{\theta,k} = \hat{u}_{\theta,k}$ and $u_{\theta,k} = \hat{u}_{\theta,k} + \pi$. This can be accomplished for all m outgoing messages in a total of $O(m)$ operations by breaking (4.32) into forward and backward partial product recursions as described previously for general variable nodes.

To find the estimate $\hat{u}_{\theta,k}$, we begin by splitting y_i into its real and imaginary parts $y_{R,i}$ and $y_{I,i}$ and rewriting the nominal message (4.32) as a sum of products

$$\mu_{U_{\theta,k}\to T_j}(u_{\theta,k}) = \sum_{\underline{v}\in\{0,1\}^{m-1}} C_j(\underline{v})\cdot\exp[A_j(\underline{v})\cos(u_{\theta,k}) + B_j(\underline{v})\sin(u_{\theta,k})] \tag{4.34}$$

with

$$A_j(\underline{v}) = \frac{2}{N_0}\sqrt{E_s}\sum_{\substack{i=km\\ i\neq j}}^{km+m-1} (-1)^{v(i)}\hat{u}_{\alpha,k,i}\,y_{R,i} \tag{4.35}$$

$$B_j(\underline{v}) = \frac{2}{N_0}\sqrt{E_s}\sum_{\substack{i=km\\ i\neq j}}^{km+m-1}(-1)^{v(i)}\hat{u}_{\alpha,k,i}\,y_{I,i} \tag{4.36}$$

$$C_j(\underline{v}) = \prod_{\substack{i=km\\ i\neq j}}^{km+m-1}\mu_{X_i\to T_i}(v(i)). \tag{4.37}$$

We would like to use the $u_{\theta,k}$ that maximizes this message for the phase estimate but this is too computationally difficult due to the large number of terms. Fortunately, a small number of the terms usually dominate the message value. Thus, the phase can be estimated by choosing the term, or equivalently the vector $\underline{v}$, whose maximum over $u_{\theta,k}$ is largest and using that maximizing $u_{\theta,k}$ as the estimate.

We would like to compute

$$\hat{\underline{v}}_j = \arg\max_{\underline{v}} C_j(\underline{v})\exp(D_j(\underline{v})), \tag{4.38}$$

where

$$D_j(\underline{v}) = \sqrt{A_j^2(\underline{v}) + B_j^2(\underline{v})},$$

but for computational reasons will settle for a common estimate for all outgoing messages. This corresponds to finding

$$\hat{\underline{v}} = \arg\max_{\underline{v}} C_{-1}(\underline{v})\exp(D_{-1}(\underline{v})) \tag{4.39}$$

where the -1 subscript means that no values are left out of the summations over i for the A, B, C, and D expressions. To approximate $\hat{\underline{v}}$, we propose a simple bit flipping algorithm. Begin with a $\underline{v}$ that maximizes $C_{-1}(\underline{v})$. For each bit in order, flip the bit if this increases $C_{-1}(\underline{v})\exp(D_{-1}(\underline{v}))$. Repeat this process until no bit in $\underline{v}$ is flipped. At the end of each iteration, we also try the closest vector $\underline{v}'$ to $-\underline{v}$ for which $C_{-1}(\underline{v}') > 0$, accepting it only if it increases the objective function. This algorithm may not find the global maximum, but it seems to work well enough and its complexity is still roughly linear in m. To find the shared phase estimate $\hat{u}_{\theta,k}$ for the bimodal CD, we compute the parameters $A_{-1}(\hat{\underline{v}})$ and $B_{-1}(\hat{\underline{v}})$ and take

$$\hat{u}_{\theta,k} = \angle(A_{-1}(\hat{\underline{v}}) + B_{-1}(\hat{\underline{v}})\sqrt{-1}). \tag{4.40}$$

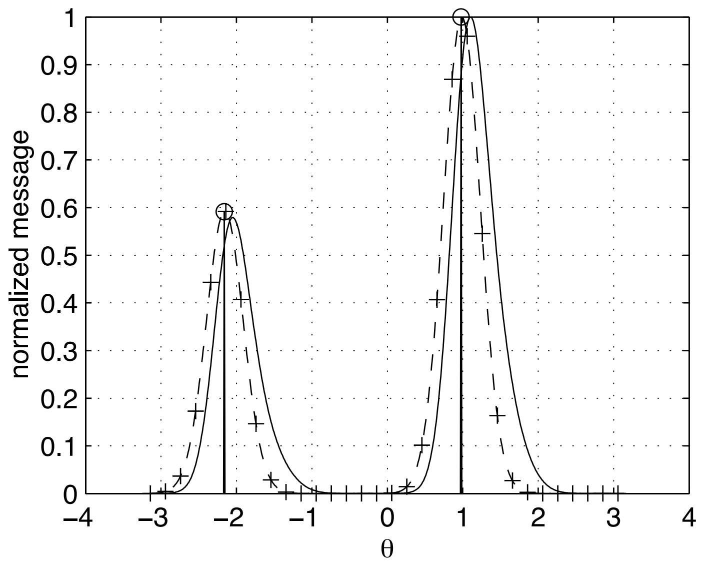


Figure 4.13: Comparison of canonical distributions for the phase message. '○'–bimodal CD, '+'and dashed–two-term exponential CD, solid–nominal message. These are for a typical realization of 16 bits, assuming low reliability bit estimates with error rate 0.1, and $E_s/N_0 = 1$ dB. The actual phase is $\pi/2 \cong 1.57$ so none of the estimates are particularly good.

It turns out that there are usually two large terms in (4.34), one corresponding to $\hat{\underline{v}}$ and the other to $\hat{\underline{v}}'$ close to $-\hat{\underline{v}}$ as in the $\underline{v}'$ step above. The alternate bits $\hat{\underline{v}}'$ are not exactly equal to $-\hat{\underline{v}}$ because the first element needs to match the pilot symbol so $C_{-1}(\underline{v})$ is not too small. Using just the two terms corresponding to $\hat{\underline{v}}$ and $\hat{\underline{v}}'$ for the canonical distribution gives the two-term exponential CD

$$\mu'_{U_{\theta,k}\to T_j}(u_{\theta,k}) = C_1 \exp(A_1 \cos(u_{\theta,k}) + B_1 \sin(u_{\theta,k})) + C_2 \exp(A_2 \cos(u_{\theta,k}) + B_2 \sin(u_{\theta,k})) \quad (4.41)$$

where we compute the A, B, and C parameters from $\hat{\underline{v}}$ and $\hat{\underline{v}}'$ using the equations (4.35)-(4.37) above. There is a common estimate for the vector $\underline{v}$, but the parameters are all computed individually for the outgoing messages. Figure 4.13 shows the nominal message, the bimodal CD, and the two-term exponential CD for a representative trial.

Both of these canonical distributions result in realizable updates for the message

$\mu_{T_i \to X_i}$. With the bimodal CD,

$$\mu_{T_i \to X_i}(x_i) = a_1 \exp\left(-\frac{\left|y_i - \hat{u}_{\alpha,k,i} \exp(j\hat{u}_{\theta,k})(-1)^{x_i}\sqrt{E_s}\right|^2}{N_0}\right) + \\ a_2 \exp\left(-\frac{\left|y_i - \hat{u}_{\alpha,k,i} \exp(j(\hat{u}_{\theta,k} + \pi))(-1)^{x_i}\sqrt{E_s}\right|^2}{N_0}\right) \quad (4.42)$$

The two-term exponential CD gives a little more complicated result

$$\mu_{T_i \to X_i}(x_i) = C_1 I_0(K_1(x_i)) + C_2 I_0(K_2(x_i)) \quad (4.43)$$

where $I_0(\cdot)$ is the modified Bessel function of order 0 and

$$K_l(x_i) = \sqrt{[\tfrac{2}{N_0}\hat{u}_{\alpha,k,i}(-1)^{x_i} y_{R,i} + A_l]^2 + [\tfrac{2}{N_0}\hat{u}_{\alpha,k,i}(-1)^{x_i} y_{I,i} + B_l]^2}$$

for $l = 1, 2$.

We could also consider more conventional canonical distributions. The estimate canonical distribution

$$\mu'_{U_{\theta,k} \to T_j}(u_{\theta,k}) = \delta(u_{\theta,k} - \hat{u}_{\theta,k,j}) \quad (4.44)$$

depends only on the phase estimate $\hat{u}_{\theta,k,j}$. We compute $\hat{u}_{\theta,k,j}$ by finding a common estimate $\underline{\hat{v}}$ for all the outgoing messages as described above and computing phase estimates individually using

$$\hat{u}_{\theta,k,j} = \angle(A_j(\underline{\hat{v}}) + B_j(\underline{\hat{v}})\sqrt{-1}). \quad (4.45)$$

There are at least two possible approaches to discretization. The quantized CD uses

$$\mu'_{U_{\theta,k} \to T_j}(u_{\theta,k}) = \sum_{l=0}^{L-1} a_l \delta(u_{\theta,k} - \hat{u}_l) \quad (4.46)$$

where the $\hat{u}_l$ values are selected in advance. The a_l values are easily computed by sampling the original message at the values $u_{\theta,k} = \hat{u}_l$. The piece-wise constant CD

$$\mu'_{U_{\theta,k} \to T_j}(u_{\theta,k}) = \sum_{l=0}^{L-1} a_l r_l(u_{\theta,k}) \quad (4.47)$$

with $r_l(x) = 1$ if $2\pi l/L \leq x < 2\pi(l+1)/L$, and $r_l(x) = 0$ otherwise, preserves more of the ambiguity in the phase estimate. The a_l parameters could be computed by integrating the incoming messages $\mu_{T_i \to U_{\theta,k}}$ and then multiplying them in the usual way

$$a_l = \prod_{\substack{i=km \\ i \neq j}}^{km+m-1} \left[\int_{2\pi l/L}^{2\pi(l+1)/L} \mu_{T_i \to U_{\theta,k}}(u_{\theta,k}) \, du_{\theta,k} \right]. \tag{4.48}$$

Although this appears computationally difficult, the integrals of $\mu_{T_i \to U_{\theta,k}}$ can be computed relatively easily using tabulated functions. The resulting receiver is equivalent to a design proposed by Komninakis and Wesel [36]. This scheme could also be used to compute the a_l parameters for the quantized CD. The message $\mu_{T_i \to X_i}$ is straightforward to compute with the estimate CD or quantized CD for $\mu_{U_{\theta,k} \to T_j}$. With the piece-wise constant CD, some approximation to the integrals is needed, but this is not too difficult.

This channel and its relatives with Markov memory or other time-correlated fading processes have been widely studied. Differentially encoded phase shift keying (DPSK) reduces the impact of phase variation by encoding the data in the phase difference between successive transmitted symbols. If the phase is slowly varying then the difference between successive symbols will be maintained even though the absolute phase is unknown. This technique requires that a reference symbol be transmitted to provide the phase difference for the first symbol, and thus requires an additional symbol in each hop just as with pilot symbols. Numerous iterative receivers for DPSK on non-coherent fading channels have been proposed [27, 43, 44, 54]. Most of them use some variant of multiple-symbol differential detection for the DPSK modulation combined with a typical SISO decoder for the error-control code. The DPSK portion is decoded, the results are interleaved and passed to the error-control code decoder, and this process is iterated. A few researchers have also considered using pilot symbols for non-coherent channels with time-correlated fading [64, 67, 72]. These proposed designs rarely include soft estimation of the channel phase process, as ours do. Incorporating a DPSK inner code in receiver designs based on factor graphs is straightforward. The same canonical distributions for the channel phase state suggested above can still be used.

4.4.2 Simulation Results

Perfect knowledge of the channel phase offset allows us to compensate for it, effectively reducing the channel to the coherent Rayleigh channel discussed in the previous section. Therefore, we compare our proposed receivers with the simulation results for the coherent Rayleigh channel with perfect knowledge of the fade level. However, an important difference between this case and the coherent Rayleigh fading system is that this system uses pilot symbols to help in acquiring the phase. To show how the pilot symbols affect the performance, we also include the perfect side information results when pilot symbols are used. The pilot symbols simply add an energy penalty; the error probabilities stay exactly the same since there is no advantage to using pilot symbols if the channel is known. Throughout this section E_b/N_0 refers to the average signal to noise ratio in the Rayleigh fading, which is equal to the SNR when the fade level is one because of the way we have defined the fade level distribution.

We begin with a comparison of receivers based on the various proposed canonical distributions at a hop length of 21 symbols, which is approximately the optimum hop length. Figure 4.14 compares the bit error rate curves for receivers using the quantized, two-term exponential, bimodal, and estimate canonical distributions. The perfect CSI performance both with and without pilot symbols is also shown. The bimodal CD receiver is within about 0.2 dB of the quantized CD, with the two-term exponential CD receiver falling in between. It makes sense that the two-term exponential CD receiver would do a little better because it includes more of the uncertainty in the phase distribution. Significantly, the estimate CD receiver does very poorly, losing more than 1.5 dB relative to the bimodal CD receiver. This indicates that maintaining soft information, especially for the phase, is very important.

The required SNR as a function of the hop length, shown in Fig. 4.15, is mostly as expected. The optimum hop length for both the bimodal and quantized canonical distributions is about 21 symbols. Unlike our previous examples, the gap between the perfect side information case and the iterative receivers is still about 1 dB for relatively large hop lengths, even when the loss due to the pilot symbols is accounted for. This suggests that significant problems estimating the phase still exist even for a hop length of 57 symbols. The only the reason that the optimum hop length is

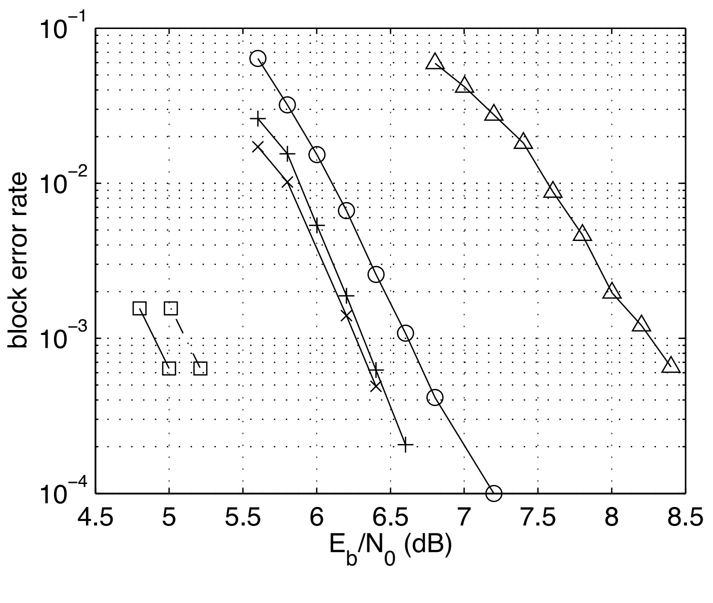


Figure 4.14: Comparison of receivers for the non-coherent Rayleigh fading channel. '$\times$' quantized CD, '$+$' two-term exponential CD, '$\circ$' bimodal CD, '$\triangle$' estimate CD, '$\square$' perfect CSI (dash-dotted includes pilot symbols). This was a block length 2520, rate 1/2 LDPC code (before pilot symbols) with one pilot symbol per hop on a non-coherent Rayleigh fading channel with hop length $m = 21$.

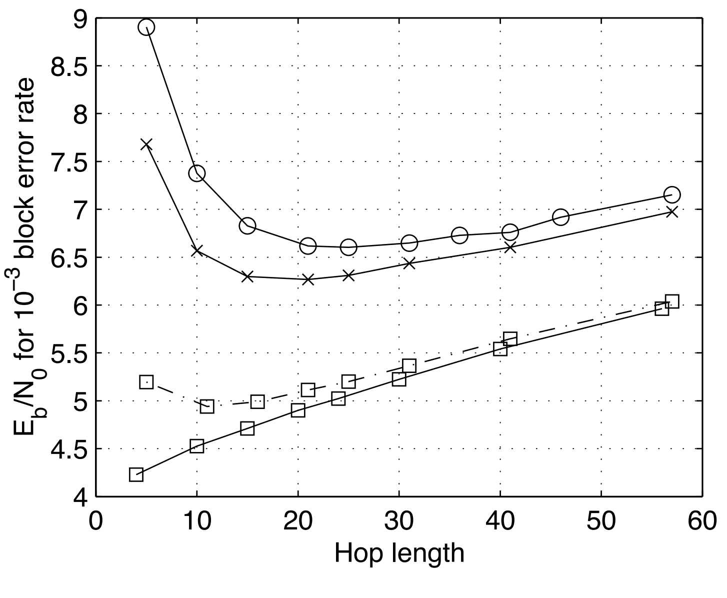


Figure 4.15: Performance vs. hop length for the non-coherent Rayleigh fading channel. '$\times$' quantized CD, '$\circ$' bimodal CD, '$\square$' perfect CSI (dash-dotted includes pilot symbols). This was a block length 2520, rate 1/2 (before pilot symbols) LDPC code with one pilot symbol per hop on a non-coherent Rayleigh fading channel.

so much less than that is that we need quite a lot of effective diversity in order to combat the Rayleigh fading of the magnitude. If the fading statistics were less severe than the Rayleigh distribution, the optimum hop length could be significantly larger. The quantized CD receiver gains more than 1 dB over the bimodal CD receiver for small hop lengths which suggests that soft information is especially important in that case.

4.5 General Discussion

Although these examples have all used LDPC codes, the factor graph technique readily allows those to be replaced by turbo codes, or nearly any other codes for which iterative decoding based on their graphical representations works well. Note however that most codes should use interleaving between the encoder and the channel in order to prevent related bits from experiencing correlated channel effects, such as fading. In general, this interleaving does not need to be random, but care must be taken to spread related bits from different component codes. While our technique can handle this interleaving without significant modifications, other approaches based on combining the channel and code trellises, often referred to as super-trellis techniques, are ineffective when the code is interleaved before the channel.

Note that interleaving was not necessary for LDPC codes because all randomly interleaved versions of an LDPC code are members of the same class. Typically, interleaving does not make any systematic difference in the structure of an LDPC code. Further research might address how to construct LDPC codes that are especially robust to certain correlated error patterns.

Implementation of these receivers should not be too difficult. Their complexity is on the same order as that for turbo decoding, which has been successfully implemented even at high data rates. All our algorithms are of roughly linear complexity in the codeword length and the hop length. Furthermore, unlike turbo decoding, which typically requires sequential processing for each of the component codes, these algorithms are highly parallelizable. Within each group, the updates may often be done simultaneously. For example, all the C nodes can be updated simultaneously or in any order.

The capability of these structures to handle a wide variety of channel and code scenarios makes this attractive when radios must function in different environments with different signaling schemes. This is exactly the task faced by software radios. Factor graph techniques are especially attractive for software radios because they can represent a variety of channels and codes in a consistent format. The similarity of the node update operations means that hardware accelerators that could easily be used for almost all cases should be possible.

We believe that these examples have demonstrated that the technique is extremely flexible, relatively straightforward, and results in clear, precise descriptions of the algorithms. As the approach is applied to more complicated scenarios, we expect that the limitations and capabilities of iterative receivers will become more clear.

CHAPTER 5

Coding Theory for Channels with Block Memory

In the previous chapters, we have seen that iterative receivers provide the possibility of approximating optimal joint channel estimation and decoding with manageable complexity. Given this capability, we would like to find the fundamental performance limits of joint channel estimation and decoding and determine when it works well. Because analysis of iterative receivers is extremely difficult, we consider information theoretic bounds on the performance of optimal maximum likelihood receivers. Experiments show that iterative receivers behave very much like the bounds for reasonably long block lengths. We will be especially interested in how the trade-off between channel estimation and effective diversity is reflected in the bounds.

Our principal tool in this chapter will be Gallager's random coding error exponent bound [19, Ch. 5], which bounds the probability of a codeword error P_e as

$$P_e \leq e^{-NE_r(R)} \tag{5.1}$$

where $E_r(R)$ is the channel reliability function which depends on the code rate R and N is the codeword length. This bound is attractive because it includes the effect of the codeword length, which is essential to see how effective diversity affects performance. The channel capacity, a purely asymptotic measure, is not sufficient to capture this effect because it essentially assumes infinite effective diversity regardless of the hop length.

Previous research has often considered capacity for channels with block memory. The capacity for fading channels with and without side information has been extensively studied [11]. Marzetta and Hochwald [45, 46] give an especially nice treatment of the capacity for the general case of non-coherent Rayleigh fading channels with block memory and multiple transmit and receive antennas. The mismatch capacity [62, 63], which gives the capacity when the channel model used for reception does not match the actual channel, has been widely used for analyzing systems with incomplete side information [39, 51]. In these analyses, it is assumed that the channel state is estimated using some *ad hoc* technique and that the receiver uses this estimate as if it is exact. Another approach allows the receiver to use the statistics of the channel estimate as well as the channel outputs and their statistics [12, 50]. These techniques effectively assume that the channel estimator can use the channel memory, but the decoder cannot. Furthermore, they consider only the capacity and therefore provide no information about the trade-off between effective diversity and channel estimation. Our approach also allows us to bypass the issue of channel estimator design by assuming optimal processing using the joint channel transition probability.

Some work has considered the reliability function for the Rayleigh fading channel [1–3, 33]. The emphasis tends to be on techniques for calculating the error exponent in the continuous input, continuous output case rather than the effect of channel memory. The concept of diversity in channels with memory has been considered in [66]. Error exponent bounds on channel state estimation performance with joint channel estimation and decoding have also been studied [28, 29]. Our simulation results show that good decoding performance is possible even when the channel estimation is relatively poor, so these results are not particularly helpful for our work.

We consider channels with block memory, which, as described above, are especially well suited to modeling frequency-hopping spread spectrum communication systems. These channels can be thought of as memoryless super-channels with vector inputs and outputs of length m. Channel behavior is completely specified by the transition probability density function $\mathrm{p}(\underline{y}|\underline{x})$ for the super-channel, which depends on the specific details of the channel model. For the channels of interest, the correlation within each block is captured by the channel state $u \in \mathcal{U}$ so that

$$\mathrm{p}(\underline{y}|\underline{x}) = \int_{\mathcal{U}} \mathrm{p}(u) \prod_{l=0}^{m-1} \mathrm{p}(y_l|x_l, u)\, du, \tag{5.2}$$

and thus the channel outputs are conditionally independent and identically distributed given the inputs and the channel state. The channel state is unknown to the receiver except through its observation of the channel outputs $\underline{y}$. Because the error exponent bound is difficult to manipulate or even compute in general for hop lengths larger than two or three, we will often consider simplified channel models that capture the important effects in more realistic scenarios.

The chapter begins with derivations of the bounds for discrete input and output alphabets. We then prove some properties predicted during our previous discussions about the trade-off between effective diversity and channel estimation. The second part of the chapter presents numerical results and specific characteristics for some simple, but relevant, examples.

5.1 General Bounds and Properties

The random coding error exponent is described in general in [19, Ch. 5]. Briefly, consider a discrete-time channel with arbitrary finite input and output alphabets $\mathcal{X}$ and $\mathcal{Y}$ respectively. Select a code $\mathcal{C}$ of rate R bits per channel use by drawing $e^{RN \ln 2}$ vectors of length N from $\mathcal{X}^N$ according to the probability density

$$\mathrm{p}(\underline{x}) = \prod_{i=0}^{N-1} Q(x_i) \tag{5.3}$$

where $Q(\cdot)$ is the channel input probability mass function. Let $P_e(\mathcal{C})$ be the codeword error probability, assuming equally likely messages, if code $\mathcal{C}$ is used. An exponential bound on the average codeword error probability $P_e = E\{P_e(\mathcal{C})\}$ for this case is given by

$$\begin{aligned} P_e &\leq e^{-NE_r(R)} \\ E_r(R) &= \max_{0 \leq \rho \leq 1} \max_{\mathbf{Q}} E_0(\rho, \mathbf{Q}) - \rho R \ln 2 \\ E_0(\rho, \mathbf{Q}) &= -\ln \sum_{y \in \mathcal{Y}} \left(\sum_{x \in \mathcal{X}} Q(x)\, \mathrm{p}(y|x)^{\frac{1}{1+\rho}}\, dx \right)^{1+\rho} \end{aligned} \tag{5.4}$$

where $\mathbf{Q}$ stands for the particular channel input probability mass function. Some code must perform as well as the average over all codes, so there exists a code that achieves the bound. The bound extends directly to continuous input and output alphabets, except that in this case the input distribution must meet some further constraint to eliminate the unrealistic possibility of transmitting at infinite power to overwhelm the channel noise.

We can derive the error exponent for channels with block memory by considering a super-channel where each block of m symbols from the original channel forms a single input. This super-channel is a memoryless channel with input alphabet $\mathcal{X}^m$, output alphabet $\mathcal{Y}^m$, and transition probabilities $\mathrm{p}(\underline{y}|\underline{x})$. Applying the bound to the super-channel, but letting the parameters continue to refer to the original channel, gives

$$\begin{aligned} P_e &\le e^{-\frac{N}{m}E_r'(R)} \\ E_r'(R) &= \max_{0\le\rho\le 1}\max_{\mathbf{Q}} E_0'(\rho,\mathbf{Q}) - \rho R m \ln 2 \\ E_0'(\rho,\mathbf{Q}) &= -\ln \sum_{\underline{y}\in\mathcal{Y}^m}\left(\sum_{\underline{x}\in\mathcal{X}^m} Q(\underline{x})\,\mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}}\right)^{1+\rho} \end{aligned} \tag{5.5}$$

which is equivalent to the usual formulation (5.4) if we slightly modify the Gallager function

$$E_0(\rho,\mathbf{Q},m) = -\frac{1}{m}\ln \sum_{\underline{y}\in\mathcal{Y}^m}\left(\sum_{\underline{x}\in\mathcal{X}^m} Q(\underline{x})\,\mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}}\right)^{1+\rho} \tag{5.6}$$

to get

$$E_r(R,m) = \max_{0\le\rho\le 1}\max_{\mathbf{Q}} E_0(\rho,\mathbf{Q},m) - \rho R \ln 2. \tag{5.7}$$

Including the channel state gives

$$E_0(\rho,\mathbf{Q},m) = -\frac{1}{m}\ln \sum_{\underline{y}\in\mathcal{Y}^m}\left(\sum_{\underline{x}\in\mathcal{X}^m} Q(\underline{x})\left[\int_{\mathcal{U}} \mathrm{p}(u)\prod_{i=0}^{m-1}\mathrm{p}(y_i|x_i,u)\,du\right]^{\frac{1}{1+\rho}}\right)^{1+\rho}. \tag{5.8}$$

The reliability function $E_r(R)$ has some well-known general properties [19, Ch. 5]. These all apply to $E_r(R,m)$ with fixed m because they hold for the memoryless super-channel that we used to derive $E_r(R,m)$. The reliability function $E_r(R,m)$ is

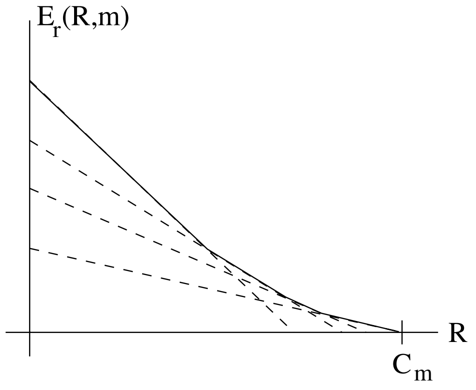


Figure 5.1: Sketch of $E_r(R,m)$ vs. R for fixed m. Solid lines–$E_r(R,m)$, dashed lines–$E_0(\rho,\mathbf{Q},m)$ for several values of ρ. Note that $E_r(R,m)=0$ for $R \geq C_m$.

convex $\cup$ and decreasing with respect to R. Let C_m be the channel capacity in bits per symbol for a channel with memory length m

$$C_m = \max_{\mathbf{Q}} \frac{1}{m} I(\underline{X}_m; \underline{Y}_m) \tag{5.9}$$

where $I(\underline{X}_m; \underline{Y}_m)$ is the mutual information between the inputs $\underline{X}_m$ and outputs $\underline{Y}_m$

$$I(\underline{X}_m; \underline{Y}_m) = \sum_{\underline{y}\in\mathcal{Y}^m} \sum_{\underline{x}\in\mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}) Q(\underline{x}) \log_2 \frac{\mathrm{p}(\underline{y}|\underline{x})}{\mathrm{p}(\underline{y})} \tag{5.10}$$

where $\underline{X}_m$ and $\underline{Y}_m$ are vectors of length m. Then the reliability function $E_r(R,m) > 0$ if and only if $R < C_m$, otherwise $E_r(R,m) = 0$. We can interpret $E_r(R,m)$ for fixed m as the upper envelope of a collection of lines with slopes $-\rho$ and $R=0$ intercepts $\max_{\mathbf{Q}} E_0(\rho,\mathbf{Q},m)$ as shown in Fig. 5.1.

5.1.1 Channel Classes

In order to design communication systems, we would like to choose the best channel memory for joint channel estimation and decoding. This is particularly feasible in frequency-hopping spread spectrum systems where we can select the hop length. We begin with the idea that a hop length m^* is optimal for a particular set of fixed channel parameters, such as signal-to-noise ratio (SNR), and communication rate R, if $E_r(R,m^*) \geq E_r(R,m)$ for all m. This implies that under these conditions, hop length m^* gives the smallest codeword error probability bound. We propose to classify channels according to how their optimal hop lengths depend on R. If hop length

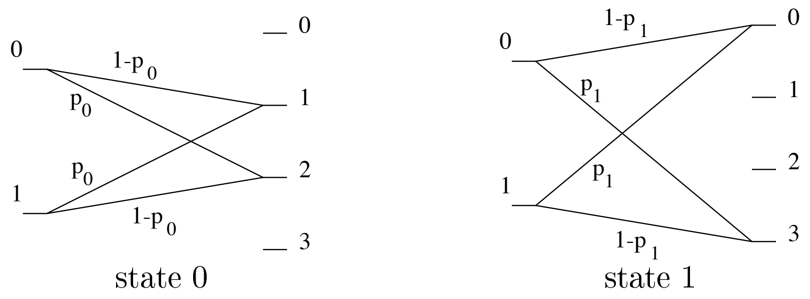


Figure 5.2: Example diversity dominated channel. In state 0, outputs 1 and 2 are possible and the cross-over probability is p_0. In state 1, outputs 0 and 3 are possible and the cross-over probability is p_1.

$m^* = 1$ is optimal regardless of R, then we say that the channel is *diversity dominated*. These channels are not of much interest here because the bound suggests that at most minor performance improvements are possible by using channel memory with joint estimation and decoding. If hop length $m^* = N$, the largest possible hop length, is optimal, then we say that the channel is *estimation dominated*. Sophisticated joint channel estimation and decoding techniques may provide significant performance gains for these channels. We will show that most channels are neither diversity dominated nor estimation dominated, and in these cases, the optimum hop length depends on the code rate.

First, we provide an example of each class of channel. We first consider a two-state channel with two inputs, and four outputs, which is shown in Fig. 5.2. The channel states have probabilities P_0 and P_1. Two of the outputs are possible in state 0 and two are possible in state 1. Because the channel state can be perfectly estimated from a single channel output, there is no estimation advantage to using long hops. On the contrary, longer hops increase the probability that the channel with the higher error probability will occur more than would be expected. Thus, the channel is diversity dominated.

Consider a channel with two equally likely states. In state 0, it is a binary symmetric channel with cross-over probability p; in state 1, it is a binary symmetric channel with cross-over probability $1 - p$. Clearly, if we have a good estimate of the channel state, the block length does not matter since we can simply flip our observations when necessary and recover a channel with uniform error probability $\min(p, 1 - p)$, independent of the hop length. For hop length 1, this channel (without side information) is useless since both outputs have equal probability irrespective of the inputs. Longer hop lengths allow better estimation of the channel state so this channel is estimation dominated.

Finally, suppose that given the state, the channel is a binary symmetric channel with cross-over probability p_u which depends on the channel state and that $p_u < \frac{1}{2}$ for all u. Because the cross-over probability depends on u and $p_u \neq 1 - p_{u'}$, smaller hop lengths improve performance by increasing the effective diversity. On the other hand, knowledge of the state gives better reliability information for the bits which also improves performance. Thus, the described trade-off of channel estimation vs. effective diversity applies and the channel is neither diversity dominated nor estimation dominated.

5.1.2 Properties of the Reliability Function

We begin by showing that for most channels of interest, the optimal input distribution $\mathbf{Q}$ is uniform. Recall that the transition matrix for a discrete, memoryless channel contains as its elements the transition probabilities $\mathrm{p}(y|x)$ from the input associated with each row, to the output associated with each column. Our channel models have transition probabilities that depend on the channel state u. These channels are completely specified by the transition probabilities $\mathrm{p}(y|x,u)$ conditioned on the state. Thus, we think of the transition matrix as containing functions of u. This leads to the following generalization of the usual definition for a symmetric channel.

Definition 5.1 *A discrete memoryless channel with state u and transition probabilities $\mathrm{p}(y|x,u)$ is* symmetric *if the columns of its transition probability matrix can be partitioned so that the rows of each partition are permutations of each other and the columns of each partition are permutations of each other.*

Note that because the elements of the transition matrix are functions of u, the same permutation must hold for all u. We begin with the following lemma.

Lemma 5.1 *If the channel* $\mathrm{p}(y|x,u)$ *is symmetric, then the memoryless super-channel*

$$\mathrm{p}(\underline{y}|\underline{x},u) = \prod_{i=0}^{m-1} \mathrm{p}(y_i|x_i,u) \tag{5.11}$$

is symmetric.

Proof: Since $\mathrm{p}(y|x,u)$ is symmetric, there exists a partition $\mathcal{Y}_1,\ldots,\mathcal{Y}_n$ of $\mathcal{Y}$, which by definition has $\bigcup_i \mathcal{Y}_i = \mathcal{Y}$ and $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset$ for all $i \neq j$, such that for any $y, y' \in \mathcal{Y}_i$

$$\{\mathrm{p}(y|x,u) : x \in \mathcal{X}\} = \{\mathrm{p}(y'|x,u) : x \in \mathcal{X}\} \tag{5.12}$$

and for any $x, x' \in \mathcal{X}$

$$\{\mathrm{p}(y|x,u) : y \in \mathcal{Y}_i\} = \{\mathrm{p}(y|x',u) : y \in \mathcal{Y}_i\}. \tag{5.13}$$

Now consider the super-channel with output set $\mathcal{Y}^m$ and take the partition $\tilde{\mathcal{Y}}_1,\ldots,\tilde{\mathcal{Y}}_{n^m}$ formed by taking products of the partitions of the elements so that $\tilde{\mathcal{Y}}_i = \mathcal{Y}_{i_0} \times \mathcal{Y}_{i_1} \times \cdots \times \mathcal{Y}_{i_{m-1}}$ for some indices $i_0, i_1, \ldots, i_{m-1}$ between 1 and n. From this construction, it is clear that for any $\underline{y}, \underline{y}' \in \tilde{\mathcal{Y}}_i$

$$\left\{\prod_{i=0}^{m-1} \mathrm{p}(y_i|x_i,u) : \underline{x} \in \mathcal{X}^m\right\} = \left\{\prod_{i=0}^{m-1} \mathrm{p}(y_i'|x_i,u) : \underline{x} \in \mathcal{X}^m\right\} \tag{5.14}$$

and for any $\underline{x}, \underline{x}' \in \mathcal{X}^m$

$$\left\{\prod_{i=0}^{m-1} \mathrm{p}(y_i|x_i,u) : \underline{y} \in \tilde{\mathcal{Y}}_i\right\} = \left\{\prod_{i=0}^{m-1} \mathrm{p}(y_i|x_i',u) : \underline{y} \in \tilde{\mathcal{Y}}_i\right\}. \tag{5.15}$$

where the elements of the sets are still functions of u. Thus, $\mathrm{p}(\underline{y}|\underline{x},u)$ is symmetric.

■

Theorem 5.1 *If the channel with state* $\mathrm{p}(y|x,u)$ *is symmetric, then the maximizing distribution* $\mathbf{Q}$ *for* $E_0(\rho,\mathbf{Q},m)$ *is uniform over* $\mathcal{X}^m$.

Proof: Suppose

$$\mathrm{p}(\underline{y}|\underline{x},u) = \prod_{i=0}^{m-1} \mathrm{p}(y_i|x_i,u), \tag{5.16}$$

and note that by Lemma 5.1 above, the channel $\mathrm{p}(\underline{y}|\underline{x},u)$ is symmetric. Then, the channel

$$\mathrm{p}(\underline{y}|\underline{x}) = \int_{\mathcal{U}} \mathrm{p}(u)\,\mathrm{p}(\underline{y}|\underline{x},u)\,du, \tag{5.17}$$

has the property that for the partition $\tilde{\mathcal{Y}}_i$,

$$\{\mathrm{p}(\underline{y}|\underline{x}) : \underline{x} \in \mathcal{X}^m\} = \{\mathrm{p}(\underline{y}'|\underline{x}) : \underline{x} \in \mathcal{X}^m\} \qquad ;\ \underline{y}, \underline{y}' \in \tilde{\mathcal{Y}}_i \tag{5.18}$$

and

$$\{\mathrm{p}(\underline{y}|\underline{x}) : \underline{y} \in \tilde{\mathcal{Y}}_i\} = \{\mathrm{p}(\underline{y}|\underline{x}') : \underline{y} \in \tilde{\mathcal{Y}}_i\} \qquad ;\ \underline{x}, \underline{x}' \in \mathcal{X}^m. \tag{5.19}$$

It is well-known that the optimal input distribution with respect to both the error exponent and the capacity is uniform for channels with the above symmetry property. We include the details here for completeness. By Gallager's Theorem 5.6.5 [19], $\mathbf{Q}$ is a maximizing distribution for $E_0(\rho, \mathbf{Q}, m)$ if, and only if,

$$\sum_{\underline{y}\in\mathcal{Y}^m} \mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}} \alpha_{\underline{y}}(\mathbf{Q})^{\rho} \geq \sum_{\underline{y}\in\mathcal{Y}^m} \alpha_{\underline{y}}(\mathbf{Q})^{1+\rho} \qquad ;\ \forall \underline{x} \in \mathcal{X}^m \tag{5.20}$$

with equality for all $\underline{x}$ such that $Q(\underline{x}) > 0$ where

$$\alpha_{\underline{y}}(\mathbf{Q}) = \sum_{\underline{x}\in\mathcal{X}^m} Q(\underline{x})\,\mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}}.$$

Substituting for $\mathbf{Q}$ and $\mathrm{p}(\underline{y}|\underline{x})$, we need only show that

$$\begin{aligned}
\sum_{\underline{y}\in\mathcal{Y}^m} \mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}} \alpha_{\underline{y}}(\mathbf{Q})^{\rho} &= \sum_{\underline{y}\in\mathcal{Y}^m} \alpha_{\underline{y}}(\mathbf{Q})^{1+\rho} \\
\left(\frac{1}{|\mathcal{X}^m|}\right)^{\rho} \sum_{\underline{y}\in\mathcal{Y}^m} \mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}} \left[\sum_{\underline{x}'\in\mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}')^{\frac{1}{1+\rho}}\right]^{\rho} &= \\
\left(\frac{1}{|\mathcal{X}^m|}\right)^{1+\rho} &\sum_{\underline{y}\in\mathcal{Y}^m} \left[\sum_{\underline{x}'\in\mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}')^{\frac{1}{1+\rho}}\right]^{1+\rho} \\
\sum_{\underline{y}\in\mathcal{Y}^m} \mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}} |\mathcal{X}^m| \left[\sum_{\underline{x}'\in\mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}')^{\frac{1}{1+\rho}}\right]^{\rho} &= \sum_{\underline{y}\in\mathcal{Y}^m} \left[\sum_{\underline{x}'\in\mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}')^{\frac{1}{1+\rho}}\right]^{1+\rho} \\
\sum_{i=0}^{m-1} \sum_{\underline{y}\in\tilde{\mathcal{Y}}_i} \mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}} |\mathcal{X}^m| \left[\sum_{\underline{x}'\in\mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}')^{\frac{1}{1+\rho}}\right]^{\rho} &= \sum_{i=0}^{m-1} \sum_{\underline{y}\in\tilde{\mathcal{Y}}_i} \left[\sum_{\underline{x}'\in\mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}')^{\frac{1}{1+\rho}}\right]^{1+\rho}
\end{aligned} \tag{5.21}$$

for all $\underline{x}$. Now, let $\underline{y}_i$ be any element of $\tilde{\mathcal{Y}}_i$. By the symmetry of the channel, for any $\underline{y} \in \tilde{\mathcal{Y}}_i$

$$\left[\sum_{\underline{x}' \in \mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}')^{\frac{1}{1+\rho}}\right]^{1+\rho} = \left[\sum_{\underline{x}' \in \mathcal{X}^m} \mathrm{p}(\underline{y}_i|\underline{x}')^{\frac{1}{1+\rho}}\right]^{1+\rho} \tag{5.22}$$

which means that we only need to show that

$$\sum_{i=0}^{m-1}\left[\sum_{\underline{x}' \in \mathcal{X}^m} \mathrm{p}(\underline{y}_i|\underline{x}')^{\frac{1}{1+\rho}}\right]^{\rho} |\mathcal{X}^m| \sum_{\underline{y} \in \tilde{\mathcal{Y}}_i} \mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}} = \sum_{i=0}^{m-1}\sum_{\underline{y} \in \tilde{\mathcal{Y}}_i}\left[\sum_{\underline{x}' \in \mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}')^{\frac{1}{1+\rho}}\right]^{1+\rho} \tag{5.23}$$

for any $\underline{x}$. Now, for any $\underline{x}, \underline{x}' \in \mathcal{X}^m$,

$$\sum_{\underline{y} \in \tilde{\mathcal{Y}}_i} \mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}} = \sum_{\underline{y} \in \tilde{\mathcal{Y}}_i} \mathrm{p}(\underline{y}|\underline{x}')^{\frac{1}{1+\rho}}. \tag{5.24}$$

Therefore

$$|\mathcal{X}^m| \sum_{\underline{y} \in \tilde{\mathcal{Y}}_i} \mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}} = \sum_{\underline{x} \in \mathcal{X}^m} \sum_{\underline{y} \in \tilde{\mathcal{Y}}_i} \mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}} \tag{5.25}$$

Interchanging the order of the summations and substituting, we find

$$\sum_{i=0}^{m-1}\left[\sum_{\underline{x}' \in \mathcal{X}^m} \mathrm{p}(\underline{y}_i|\underline{x}')^{\frac{1}{1+\rho}}\right]^{\rho} |\mathcal{X}^m| \sum_{\underline{y} \in \tilde{\mathcal{Y}}_i} \mathrm{p}(\underline{y}|\underline{x})^{\frac{1}{1+\rho}} =$$

$$\sum_{i=0}^{m-1}\sum_{\underline{y} \in \tilde{\mathcal{Y}}_i}\left[\sum_{\underline{x}' \in \mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}')^{\frac{1}{1+\rho}}\right]^{\rho}\left[\sum_{\underline{x}' \in \mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}')^{\frac{1}{1+\rho}}\right] \tag{5.26}$$

which satisfies (5.23) to complete the proof. ■

Note that the theorem holds independent of ρ so this distribution is the capacity achieving distribution as well.

Channels with block memory have an important property which is summarized in the following lemma.

Lemma 5.2 *Consider a block memory channel with any hop length m. Let*

$$\underline{X}_m^{(i)} = \begin{bmatrix} X_0 & X_1 & \ldots & X_{i-1} & X_{i+1} & \ldots & X_{m-1} \end{bmatrix}$$

and $\underline{Y}_m = \begin{bmatrix} Y_0 & Y_1 & \dots & Y_{m-1} \end{bmatrix}$, *then*

$$H(X_i|\underline{X}_m^{(i)}, \underline{Y}_m) = H(X_j|\underline{X}_m^{(j)}, \underline{Y}_m) \tag{5.27}$$

for any $i, j \in \{0, 1, \dots, m-1\}$.

Proof: The proof is based on symmetry in the channel model.

$$H(X_i|\underline{X}_m^{(i)}, \underline{Y}_m) = -\sum_{x_i\in\mathcal{X}}\sum_{x_j\in\mathcal{X}}\sum_{\underline{x}_m^{(i,j)}\in\mathcal{X}^{m-2}}\sum_{\underline{y}\in\mathcal{Y}^m} \mathrm{p}(x_i, x_j, \underline{x}_m^{(i,j)}, \underline{y}) \log_2 \mathrm{p}(x_i|x_j, \underline{x}^{(i,j)}, \underline{y}) \tag{5.28}$$

$$= -\sum_{x_i\in\mathcal{X}}\sum_{x_j\in\mathcal{X}}\sum_{\underline{x}_m^{(i,j)}\in\mathcal{X}^{m-2}}\sum_{\underline{y}\in\mathcal{Y}^m} \int_{\mathcal{U}} \frac{1}{|\mathcal{X}|^m}\,\mathrm{p}(u)\prod_{i=0}^{m-1}\mathrm{p}(y_i|x_i,u)\,du\,\cdot \tag{5.29}$$

$$\log_2 \frac{\int_{\mathcal{U}} \frac{1}{|\mathcal{X}|^m}\,\mathrm{p}(u)\prod_{i=0}^{m-1}\mathrm{p}(y_i|x_i,u)\,du}{\sum_{\underline{x}_m'^{(i,j)}\in\mathcal{X}^{m-2}}\sum_{\underline{x}_j'\in\mathcal{X}}\sum_{\underline{y}'\in\mathcal{Y}^m}\int_{\mathcal{U}} \frac{1}{|\mathcal{X}|^m}\,\mathrm{p}(u)\prod_{i=0}^{m-1}\mathrm{p}(y_i'|x_i',u)\,du}$$

$$= -\sum_{x_i\in\mathcal{X}}\sum_{x_j\in\mathcal{X}}\sum_{\underline{x}_m^{(i,j)}\in\mathcal{X}^{m-2}}\sum_{\underline{y}\in\mathcal{Y}^m} \int_{\mathcal{U}} \frac{1}{|\mathcal{X}|^m}\,\mathrm{p}(u)\prod_{i=0}^{m-1}\mathrm{p}(y_i|x_i,u)\,du\,\cdot \tag{5.30}$$

$$\log_2 \frac{\int_{\mathcal{U}} \frac{1}{|\mathcal{X}|^m}\,\mathrm{p}(u)\prod_{i=0}^{m-1}\mathrm{p}(y_i|x_i,u)\,du}{\sum_{\underline{x}_m'^{(i,j)}\in\mathcal{X}^{m-2}}\sum_{\underline{x}_i'\in\mathcal{X}}\sum_{\underline{y}'\in\mathcal{Y}^m}\int_{\mathcal{U}} \frac{1}{|\mathcal{X}|^m}\,\mathrm{p}(u)\prod_{i=0}^{m-1}\mathrm{p}(y_i'|x_i',u)\,du}$$

$$= H(X_j|\underline{X}_m^{(j)}, \underline{Y}_m) \tag{5.31}$$

■

Conventional wisdom is that memory increases the channel capacity. We make this concrete with the following theorem

Theorem 5.2 *For a symmetric channel with block memory, the capacity of the associated block memory channel is either increasing in the hop length* m *or independent of* m.

Proof: We need to show that either $C_{m+1} > C_m$ for all m, or $C_{m+1} = C_m$ for all m. Consider the following sequence of equivalent statements where if one holds with equality, all hold with equality,

$$C_{m+1} \geq C_m \tag{5.32}$$

$$\frac{1}{m+1} I(\underline{Y}_{m+1}; \underline{X}_{m+1}) \geq \frac{1}{m} I(\underline{Y}_m; \underline{X}_m) \tag{5.33}$$

$$I(\underline{Y}_{m+1}; \underline{X}_{m+1}) \geq I(\underline{Y}_m; \underline{X}_m) + \frac{1}{m} I(\underline{Y}_m; \underline{X}_m) \tag{5.34}$$

$$H(\underline{X}_{m+1}) - H(\underline{X}_{m+1}|\underline{Y}_{m+1}) \geq (1+\frac{1}{m})H(\underline{X}_m) - (1+\frac{1}{m})H(\underline{X}_m|\underline{Y}_m) \tag{5.35}$$

Since the optimal channel inputs for a symmetric channel are equally likely by the lemma above,

$$\begin{aligned} H(\underline{X}_{m+1}) &= (m+1)\log|\mathcal{X}| \\ &= H(\underline{X}_m) + H(X) \\ &= (1+\frac{1}{m})H(\underline{X}_m). \end{aligned} \tag{5.36}$$

Using the chain rule for entropy, we find

$$H(\underline{X}_{m+1}|\underline{Y}_{m+1}) = H(X_{m+1}|\underline{X}_m, Y_{m+1}, \underline{Y}_m) + H(\underline{X}_m|Y_{m+1}, \underline{Y}_m) \tag{5.37}$$

Using the fact that conditioning decreases entropy, we get

$$H(\underline{X}_{m+1}|\underline{Y}_{m+1}) \leq H(X_{m+1}|\underline{X}_m, Y_{m+1}, \underline{Y}_m) + H(\underline{X}_m|\underline{Y}_m) \tag{5.38}$$

By another application of the chain rule for entropy,

$$\frac{1}{m} H(\underline{X}_m|\underline{Y}_m) = \frac{1}{m}\sum_{i=0}^{m-1} H(X_i|X_{i-1}, X_{i-2}, \ldots, X_0, \underline{Y}_m). \tag{5.39}$$

and conditioning on all of the X values except X_i instead of just the first i of them

$$\frac{1}{m} H(\underline{X}_m|\underline{Y}_m) \geq \frac{1}{m}\sum_{i=0}^{m-1} H(X_i|\underline{X}_m^{(i)}, \underline{Y}_m). \tag{5.40}$$

Combining (5.35) and (5.36), it is sufficient to show that

$$H(\underline{X}_{m+1}|\underline{Y}_{m+1}) \leq (1+\frac{1}{m})H(\underline{X}_m|\underline{Y}_m), \tag{5.41}$$

which, using (5.38), and (5.40), reduces to

$$H(X_{m+1}|\underline{X}_m, Y_{m+1}, \underline{Y}_m) \leq \frac{1}{m}\sum_{i=0}^{m-1} H(X_i|\underline{X}_m^{(i)}, \underline{Y}_m). \tag{5.42}$$

Using Lemma 5.2,

$$\begin{aligned}\frac{1}{m}\sum_{i=0}^{m-1} H(X_i|\underline{X}_m^{(i)},\underline{Y}_m) &= \frac{1}{m}\sum_{i=0}^{m-1} H(X_0|\underline{X}_m^{(0)},\underline{Y}_m) \\ &= H(X_0|\underline{X}_m^{(0)},\underline{Y}_m).\end{aligned} \tag{5.43}$$

Applying Lemma 5.2 again,

$$H(X_{m+1}|\underline{X}_m, Y_{m+1}, \underline{Y}_m) = H(X_0|X_{m+1}, \underline{X}_m^{(0)}, Y_{m+1}, \underline{Y}_m), \tag{5.44}$$

and

$$H(X_0|X_{m+1}, \underline{X}_m^{(0)}, Y_{m+1}, \underline{Y}_m) \leq H(X_0|\underline{X}_m^{(0)}, \underline{Y}_m) \tag{5.45}$$

which, combined with (5.43), proves (5.42). Since all of the inequalities either hold with equality or do not, independent of m, this completes the proof. ■

For most interesting channels, the capacity is increasing in m from the above theorem. One of our core results is that this implies that, for high enough rates, a large block length is optimal.

Theorem 5.3 *For a channel with C_m increasing in m, there is a sequence of rates R_l with $R_l \leq C_l$ such that the optimal hop length $m^* > l$ when $R > R_l$.*

Proof: Using the properties of the reliability function, $E_r(R, l) = 0$ for $R > C_l$ and $E_r(R, m) > 0$ for $R < C_m$. This implies that for any $m > l$ and $C_l \leq R < C_m$, $E_r(R, m) > E_r(R, l)$. Therefore, there must be some rate $R_l < C_l$ such that for $R > R_l$ the optimal hop length $m^* > l$. ■

A graphical representation of this theorem is shown in Fig. 5.3. This means that if C_m is increasing, the channel is not diversity dominated. Note also that the theorem only provides a lower bound on the optimal hop length. There is no guarantee that any particular hop length is ever optimal although typically there is a series of rates R_l^* such that $R_{l-1}^* < R < R_l^*$ implies that the optimal hop length m^* equals l. At this point, all we can say is that, if this is true, $R_l^* < C_l$. Clearly, estimation dominated channels do not have this property.

The effects of channel estimation can be eliminated by providing the receiver with a perfect observation of the channel state. As in our receiver performance results,

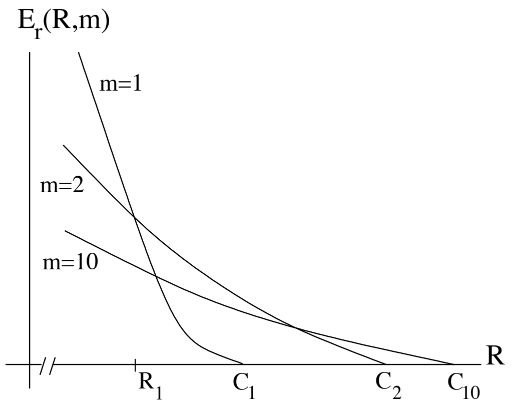


Figure 5.3: Sketch illustrating Theorem 5.3.

many interesting observations will come from comparing the bounds for the channel with perfect channel state information (CSI) to the channel with no side information. We will mark functions for the channel with perfect side information with a bar. To find the Gallager function when the receiver has a perfect observation of the channel state, we augment the channel output with a copy of the channel state

$$\bar{E}_0(\rho, \mathbf{Q}, m) = -\frac{1}{m} \ln \int_{\mathcal{U}} \sum_{\underline{y} \in \mathcal{Y}^m} \left(\sum_{\underline{x} \in \mathcal{X}^m} Q(\underline{x})\, \mathrm{p}(\underline{y}, u | \underline{x})^{\frac{1}{1+\rho}} \right)^{1+\rho} du. \tag{5.46}$$

Using the fact that the channel state u and the channel input $\underline{x}$ are independent,

$$\mathrm{p}(\underline{y}, u | \underline{x}) = \mathrm{p}(\underline{y} | \underline{x}, u)\, \mathrm{p}(u). \tag{5.47}$$

Substituting, we get

$$\bar{E}_0(\rho, \mathbf{Q}, m) = -\frac{1}{m} \ln \int_{\mathcal{U}} \mathrm{p}(u) \sum_{\underline{y} \in \mathcal{Y}^m} \left(\sum_{\underline{x} \in \mathcal{X}^m} Q(\underline{x}) \left[\prod_{i=0}^{m-1} \mathrm{p}(y_i | x_i, u) \right]^{\frac{1}{1+\rho}} \right)^{1+\rho} du. \tag{5.48}$$

$\bar{E}_r(R, m)$ is easily obtained by inserting $\bar{E}_0(\rho, \mathbf{Q}, m)$ in (5.4).

The properties of the reliability function also apply to the reliability function with channel state information. For $R < \bar{C}_m$, $\bar{E}_r(R, m) > 0$, and for $R \geq \bar{C}_m$, $\bar{E}_r(R, m) = 0$ where $\bar{C}_m$ is the capacity with side-information

$$\bar{C}_m = \max_{\mathbf{Q}} \frac{1}{m} I(\underline{X}_m; \underline{Y}_m, U) \tag{5.49}$$

and

$$I(\underline{X}_m; \underline{Y}_m, U) = \int_{\mathcal{U}} \sum_{\underline{y} \in \mathcal{Y}^m} \sum_{\underline{x} \in \mathcal{X}^m} \mathrm{p}(\underline{y}|\underline{x}, u)\, \mathrm{p}(u) Q(\underline{x}) \log_2 \frac{\mathrm{p}(\underline{y}|\underline{x}, u)\, \mathrm{p}(u)}{\mathrm{p}(\underline{y}, u)}\, du.$$

Even with side-information, the optimal input distribution is uniform. In fact, although we do not prove it here, a less strict definition of symmetry for channels with state is sufficient.

Theorem 5.4 *For a symmetric channel* $\mathrm{p}(y|x, u)$*, the input distribution* $\mathbf{Q}$ *that maximizes* $\bar{E}_0(\rho, \mathbf{Q}, m)$ *is uniform over* $\mathcal{X}^m$*.*

Proof: Consider the channel $\mathrm{p}(\underline{y}|\underline{x}, u)$. Since for any fixed u, the channel is symmetric, the integrand in (5.48) is minimized when $\mathbf{Q}$ is uniform, regardless of u. Therefore, the uniform input distribution minimizes the integral, which maximizes the function $\bar{E}_0(\rho, \mathbf{Q}, m)$. ■

Now, we would like to prove a few theorems about the capacity and error exponent with side information and their relationship to the error exponent for joint channel estimation and decoding. These are generally intuitively clear, but it is useful to show that the bounds behave according to our expectations. If we are given the channel state, then there is no need to have longer block lengths to improve channel estimation. Also, we know that the capacity does not depend on the effective diversity. Thus, we expect that the capacity with side information is independent of the hop length.

Theorem 5.5 *For symmetric channels, the capacity with side information* $\bar{C}_m$ *is independent of the hop length.*

Proof: We show that $\bar{C}_m = \bar{C}_1$. The capacity achieving input distribution is $Q(\underline{x}) = \frac{1}{|\mathcal{X}|^m}$.

$$
\begin{aligned}
H(\underline{X}|\underline{Y},U) &= -\int_{\mathcal{U}} \sum_{\underline{y}\in\mathcal{Y}^m} \sum_{\underline{x}\in\mathcal{X}^m} \mathrm{p}(u) \frac{1}{|\mathcal{X}|^m} \prod_{i=0}^{m-1} \mathrm{p}(y_i|x_i,u) \cdot \\
&\qquad \log_2 \frac{\frac{1}{|\mathcal{X}|^m} \prod_{i=0}^{m-1} \mathrm{p}(y_i|x_i,u)}{\sum_{\underline{x}'\in\mathcal{X}^m} \frac{1}{|\mathcal{X}|^m} \prod_{i=0}^{m-1} \mathrm{p}(y_i|x'_i,u)}\, du \\
&= -\int_{\mathcal{U}} \mathrm{p}(u) \sum_{\underline{y}\in\mathcal{Y}^m} \sum_{\underline{x}\in\mathcal{X}^m} \prod_{i=0}^{m-1} \frac{1}{|\mathcal{X}|} \mathrm{p}(y_i|x_i,u) \log_2 \prod_{i=0}^{m-1} \frac{\frac{1}{|\mathcal{X}|} \mathrm{p}(y_i|x_i,u)}{\sum_{x'_i\in\mathcal{X}} \frac{1}{|\mathcal{X}|} \mathrm{p}(y_i|x'_i,u)}\, du \\
&= -\int_{\mathcal{U}} \mathrm{p}(u) \sum_{\underline{y}\in\mathcal{Y}^m} \sum_{\underline{x}\in\mathcal{X}^m} \prod_{i=0}^{m-1} \frac{1}{|\mathcal{X}|} \mathrm{p}(y_i|x_i,u) \sum_{i=0}^{m-1} \log_2 \frac{\frac{1}{|\mathcal{X}|} \mathrm{p}(y_i|x_i,u)}{\sum_{x'_i\in\mathcal{X}} \frac{1}{|\mathcal{X}|} \mathrm{p}(y_i|x'_i,u)}\, du \\
&= -\sum_{i=0}^{m-1} \int_{\mathcal{U}} \mathrm{p}(u) \sum_{y\in\mathcal{Y}} \sum_{x\in\mathcal{X}} \frac{1}{|\mathcal{X}|} \mathrm{p}(y_i|x_i,u) \log_2 \frac{\frac{1}{|\mathcal{X}|} \mathrm{p}(y_i|x_i,u)}{\sum_{x'_i\in\mathcal{X}} \frac{1}{|\mathcal{X}|} \mathrm{p}(y_i|x'_i,u)}\, du \\
&= -mH(X|Y,U).
\end{aligned}
\tag{5.50}
$$

Then,

$$
\begin{aligned}
\bar{C}_m &= \max_{\mathbf{Q}} \frac{1}{m}(H(\underline{X}) - H(\underline{X}|\underline{Y},U)) \\
&= \frac{1}{m}\log_2 |\mathcal{X}|^m - \frac{1}{m} H(\underline{X}|\underline{Y},U) \\
&= \log_2 |\mathcal{X}| - H(X|Y,U) \\
&= \bar{C}_1
\end{aligned}
\tag{5.51}
$$

■

This theorem was also proved in [48] and in a different context in [70]. It is also known that $C_m \leq \bar{C}$ [48]. If we know the channel, then longer hop lengths can only decrease the effective diversity. Thus, the error exponent with side information is either decreasing in the hop length or independent of the memory. This is also supported by analysis.

Theorem 5.6 *For symmetric channels, the reliability function with side information $\bar{E}_r(R,m)$ is either decreasing in m or independent of m.*

Proof: We begin by showing that

$$
\bar{E}_0(\rho,\mathbf{Q},m) \geq \bar{E}_0(\rho,\mathbf{Q},m+1) \tag{5.52}
$$

for all ρ where equality for any m implies equality for all m. Since the channel is symmetric, the optimizing input distribution is uniform and

$$\bar{E}_0(\rho, \mathbf{Q}, m) = -\frac{1}{m} \ln \int\limits_{\mathcal{U}} \mathrm{p}(u) \sum_{\underline{y} \in \mathcal{Y}^m} \left(\sum_{\underline{x} \in \mathcal{X}^m} \frac{1}{|\mathcal{X}|^m} (\mathrm{p}(\underline{y}|\underline{x}, u))^{\frac{1}{1+\rho}} \right)^{1+\rho} du. \tag{5.53}$$

Now consider the sums

$$\begin{aligned}
\sum_{\underline{y} \in \mathcal{Y}^m} \left(\sum_{\underline{x} \in \mathcal{X}^m} \frac{1}{|\mathcal{X}|^m} (\mathrm{p}(\underline{y}|\underline{x}, u))^{\frac{1}{1+\rho}} \right)^{1+\rho} &= \sum_{\underline{y} \in \mathcal{Y}^m} \left(\sum_{\underline{x} \in \mathcal{X}^m} \prod_{i=0}^{m-1} \frac{1}{|\mathcal{X}|} (\mathrm{p}(y_i|x_i, u))^{\frac{1}{1+\rho}} \right)^{1+\rho} \\
&= \sum_{\underline{y} \in \mathcal{Y}^m} \prod_{i=0}^{m-1} \left(\sum_{x_i \in \mathcal{X}} \frac{1}{|\mathcal{X}|} (\mathrm{p}(y_i|x_i, u))^{\frac{1}{1+\rho}} \right)^{1+\rho} \\
&= \prod_{i=0}^{m-1} \sum_{y_i \in \mathcal{Y}} \left(\sum_{x_i \in \mathcal{X}} \frac{1}{|\mathcal{X}|} (\mathrm{p}(y_i|x_i, u))^{\frac{1}{1+\rho}} \right)^{1+\rho} \\
&= \left(\sum_{y \in \mathcal{Y}} \left(\sum_{x \in \mathcal{X}} \frac{1}{|\mathcal{X}|} (\mathrm{p}(y|x, u))^{\frac{1}{1+\rho}} \right)^{1+\rho} \right)^m .
\end{aligned} \tag{5.54}$$

Let

$$g(u) = \sum_{y \in \mathcal{Y}} \left(\sum_{x \in \mathcal{X}} \frac{1}{|\mathcal{X}|} (\mathrm{p}(y|x, u))^{\frac{1}{1+\rho}} \right)^{1+\rho}, \tag{5.55}$$

then

$$\bar{E}_0(\rho, \mathbf{Q}, m) = -\frac{1}{m} \ln \int\limits_{\mathcal{U}} \mathrm{p}(u)(g(u))^m \, du. \tag{5.56}$$

So the claim (5.52) is equivalent to

$$-\frac{1}{m} \ln \int\limits_{\mathcal{U}} \mathrm{p}(u)(g(u))^m \, du \geq -\frac{1}{m+1} \ln \int\limits_{\mathcal{U}} \mathrm{p}(u)(g(u))^{m+1} \, du \tag{5.57}$$

$$\left(\int\limits_{\mathcal{U}} \mathrm{p}(u)(g(u))^m \, du \right)^{\frac{1}{m}} \leq \left(\int\limits_{\mathcal{U}} \mathrm{p}(u)(g(u))^{m+1} \, du \right)^{\frac{1}{m+1}} \tag{5.58}$$

This is a well-known fact about norms in function spaces, which we prove as follows. The function $f(x) = x^{\frac{m+1}{m}}$ is convex $\cup$ so

$$f\left(\int\limits_{\mathcal{U}} \mathrm{p}(u)(g(u))^m \, du \right) \leq \int\limits_{\mathcal{U}} \mathrm{p}(u) f((g(u))^m) \, du \tag{5.59}$$

by Jensen's inequality. Simplifying,

$$\left(\int_{\mathcal{U}} \mathrm{p}(u)(g(u))^m \, du\right)^{\frac{m+1}{m}} \leq \int_{\mathcal{U}} \mathrm{p}(u)(g(u))^{\frac{m(m+1)}{m}} \, du \tag{5.60}$$

$$\left(\int_{\mathcal{U}} \mathrm{p}(u)(g(u))^m \, du\right)^{\frac{1}{m}} \leq \left(\int_{\mathcal{U}} \mathrm{p}(u)(g(u))^{m+1} \, du\right)^{\frac{1}{m+1}} \tag{5.61}$$

with equality for all m if and only if $g(u)$ is independent of u. This proves the claim (5.52) for all ρ, so

$$\bar{E}_0(\rho, \mathbf{Q}, m) - \rho R \ln 2 \geq \bar{E}_0(\rho, \mathbf{Q}, m+1) - \rho R \ln 2. \tag{5.62}$$

Now suppose that ρ^* maximizes $\bar{E}_0(\rho, \mathbf{Q}, m+1) - \rho R \ln 2$, which gives

$$\bar{E}_r(R, m+1) = \bar{E}_0(\rho^*, \mathbf{Q}, m+1) - \rho^* R \ln 2 \leq$$
$$\bar{E}_0(\rho^*, \mathbf{Q}, m) - \rho^* R \ln 2 \leq \max_{0 \leq \rho \leq 1} \bar{E}_0(\rho, \mathbf{Q}, m) - \rho R \ln 2 = \bar{E}_r(R, m) \tag{5.63}$$

where again, if equality holds for any m, then $g(u)$ is independent of u and equality holds for all m. ■

Before we prove the next theorem, we need the following lemma [25, p. 31], which is a variant of Minkowski's inequality,

Lemma 5.3 *For $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_n$ non-negative numbers and $0 < p \leq 1$*

$$\left(\sum_{k=1}^{n} (a_k + b_k)^p\right)^{\frac{1}{p}} \geq \left(\sum_{k=1}^{n} a_k^p\right)^{\frac{1}{p}} + \left(\sum_{k=1}^{n} b_k^p\right)^{\frac{1}{p}} \tag{5.64}$$

with equality if $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_n$ are proportional or $p = 1$.

We expect that the performance with perfect knowledge of the channel state should always be at least as good as the performance without channel state information. Intuitively, we could just throw away the channel state information and return to the no channel state information case. This is supported by the following theorem.

Theorem 5.7 *For any symmetric channel $\mathrm{p}(y|x, u)$ and all m,*

$$E_r(R, m) \leq \bar{E}_r(R, m). \tag{5.65}$$

Proof: We prove that

$$E_0(\rho, \mathbf{Q}, m) \leq \bar{E}_0(\rho, \mathbf{Q}, m) \tag{5.66}$$

for any ρ from which we can conclude, as in the previous theorem, that $E_r(R, m) \leq \bar{E}_r(R, m)$.

Let

$$f(\underline{p}) = \left(\sum_{\underline{x} \in \mathcal{X}^m} p_{\underline{x}}^{\frac{1}{1+\rho}} \right)^{1+\rho} \tag{5.67}$$

where $p_{\underline{x}}$ is a vector indexed by $\underline{x}$ and may depend on $\underline{y}$ and u. Then the claim (5.66) is equivalent to

$$-\frac{1}{m} \ln \sum_{\underline{y} \in \mathcal{Y}^m} \frac{1}{|\mathcal{X}|^{m(1+\rho)}} f(E_u\{\underline{p}\}) \leq -\frac{1}{m} \ln \sum_{\underline{y} \in \mathcal{Y}^m} \frac{1}{|\mathcal{X}|^{m(1+\rho)}} E_u\{f(\underline{p})\} \tag{5.68}$$

and thus, we need only show that

$$f(E_u\{\underline{p}\}) \geq E_u\{f(\underline{p})\} \tag{5.69}$$

for all $\underline{y}$. The result follows immediately from Jensen's inequality provided that $f(\underline{p})$ is convex $\cap$ which we now show. Let λ_1 and λ_2 be two non-negative vectors and let $0 \leq \theta \leq 1$. Then

$$\begin{aligned} f(\theta\lambda_1 + (1-\theta)\lambda_2) &= \left(\sum_{\underline{x} \in \mathcal{X}^m} (\theta\lambda_{1,\underline{x}} + (1-\theta)\lambda_{2,\underline{x}})^{\frac{1}{1+\rho}} \right)^{1+\rho} && (5.70) \\ &\geq \left(\sum_{\underline{x} \in \mathcal{X}^m} (\theta\lambda_{1,\underline{x}})^{\frac{1}{1+\rho}} \right)^{1+\rho} + \left(\sum_{\underline{x} \in \mathcal{X}^m} ((1-\theta)\lambda_{2,\underline{x}})^{\frac{1}{1+\rho}} \right)^{1+\rho} && (5.71) \\ &\geq \theta \left(\sum_{\underline{x} \in \mathcal{X}^m} \lambda_{1,\underline{x}}^{\frac{1}{1+\rho}} \right)^{1+\rho} + (1-\theta) \left(\sum_{\underline{x} \in \mathcal{X}^m} \lambda_{2,\underline{x}}^{\frac{1}{1+\rho}} \right)^{1+\rho} && (5.72) \end{aligned}$$

where we use the inequality from Lemma 5.3. Thus, $f(\underline{p})$ is convex $\cap$. ■

5.1.3 An Alternate Approach to the Optimal Hop Length

For typical parameters, the rate R_1 above which some hop length greater than 1 is optimal is quite close to $\bar{C}$, the capacity with side information, which is an upper bound on the capacity without side information for any m. Therefore, the range of rates where hop lengths larger than 1 are interesting appears to be quite small. This motivates us to consider an alternate approach to optimality. Often, we would like to minimize the average signal-to-noise ratio or similar parameter, given a codeword length specified by the tolerable delay, a desired signaling rate, and a target codeword error rate. Together the codeword length N and error rate P_e fix the required error exponent $\hat{E}_r$. Precisely,

$$\hat{E}_r = -\frac{\ln P_e}{N}. \tag{5.73}$$

In this case, we need the hop length m that minimizes the SNR required to achieve that error exponent for the given rate R. If we plot the reliability function for fixed R and various values of m versus the channel signal to noise ratio, the desired optimal hop length can be determined from the upper envelope of the curves and depends on the combination of the target error probability bound and the codeword length. For reasonably large codeword lengths, this often leads to hop lengths significantly greater than 1. Intuitively, this makes sense because, for larger codeword lengths, we expect to operate quite close to the capacity and larger hop lengths should improve performance in that region.

5.2 Approaches for Wireless Channels

We would like to apply reliability function techniques to analyzing more realistic wireless channel models. As for iterative receiver design, fading and jamming channels encountered in frequency-hopping spread spectrum are of particular interest because the channel memory is well-defined and is at least somewhat under our control. Typically, these channels allow continuous inputs, usually with an average or peak power constraint, and produce continuous, sometimes complex-valued, outputs. Unfortunately, for hop lengths more than 1 or 2 symbols, computing the error exponent for these channels leads to high dimensional integrals which are computationally infeasible. To obtain results, we consider simple models which exhibit characteristics similar to the channels of interest. By considering binary-input, binary-output models with various state dependencies, we can obtain much simpler expressions for the error exponent which have computational complexity linear in the hop length.

We begin by specializing our general expression for the Gallager function (5.8), to the binary-input, binary-output case

$$E_0(\rho, \mathbf{Q}, m) = -\frac{1}{m} \ln \sum_{\underline{y} \in \{0,1\}^m} \left(\sum_{\underline{x} \in \{0,1\}^m} Q(\underline{x}) \left[\int_{\mathcal{U}} \mathrm{p}(u) \prod_{i=0}^{m-1} \mathrm{p}(y_i | x_i, u)\, du \right]^{\frac{1}{1+\rho}} \right)^{1+\rho}. \tag{5.74}$$

Now let d or $d(\underline{x}, \underline{y})$ be the number of places where $\underline{x}$ and $\underline{y}$ differ (the Hamming distance) and p_u be the channel transition probability associated with state u. Substituting, we find

$$E_0(\rho, \mathbf{Q}, m) = -\frac{1}{m} \ln \sum_{\underline{y} \in \{0,1\}^m} \left(\sum_{\underline{x} \in \{0,1\}^m} \frac{1}{2^m} \left[\int_{\mathcal{U}} \mathrm{p}(u)(1-p_u)^{m-d(\underline{x},\underline{y})} p_u^{d(\underline{x},\underline{y})}\, du \right]^{\frac{1}{1+\rho}} \right)^{1+\rho} \tag{5.75}$$

where we have used the fact that the optimum input distribution $\mathbf{Q}$ is uniform. Since the inner set of parentheses only depends on $d(\underline{x}, \underline{y})$, we can fix some particular $\underline{y}$ and then take the sum over $\underline{x}$. Since all possible binary vectors of length m are included in the sum over $\underline{x}$, it does not matter what vector $\underline{y}$ is chosen. This gives

$$E_0(\rho, \mathbf{Q}, m) = -\frac{1}{m} \ln 2^{m-m(1+\rho)} \left(\sum_{\underline{x} \in \{0,1\}^m} \left[\int_{\mathcal{U}} \mathrm{p}(u)(1-p_u)^{m-d(\underline{x},\underline{y})} p_u^{d(\underline{x},\underline{y})}\, du \right]^{\frac{1}{1+\rho}} \right)^{1+\rho} \tag{5.76}$$

for any fixed $\underline{y}$. There are $\binom{m}{d}$ binary vectors at distance d from any fixed vector, so we can change the sum over $\underline{x}$ into a sum over the distance

$$E_0(\rho, \mathbf{Q}, m) \;=\; -\frac{1}{m} \ln \left(\sum_{d=0}^{m} \binom{m}{d} \left[\int_{\mathcal{U}} \mathrm{p}(u)(1-p_u)^{m-d} p_u^{d}\, du \right]^{\frac{1}{1+\rho}} \right)^{1+\rho} \;+\; \rho \ln 2. \tag{5.77}$$

Similarly, for the channel with perfect side information, we have

$$\bar{E}_0(\rho, \mathbf{Q}, m) \;=\; -\frac{1}{m} \ln \int_{\mathcal{U}} \mathrm{p}(u) \left(\sum_{d=0}^{m} \binom{m}{d} \left[(1-p_u)^{m-d} p_u^d\right]^{\frac{1}{1+\rho}} \right)^{1+\rho} du \;+\; \rho \ln 2. \tag{5.78}$$

5.2.1 Coherent Jamming and Fading Channels

We start with the two-state jamming channel discussed in section 4.1 above. Because it is too difficult to compute the reliability function for the continuous output case, we consider the related binary-output or hard decision channel. The channel still has two states corresponding to jammed and unjammed hops and occurring with probabilities ρ_J and $(1-\rho_J)$ respectively. In the jammed state, the cross-over probability is

$$p_{eb} = Q\left(\sqrt{\frac{2E_s}{N_0 + N_j/\rho_j}}\right) \tag{5.79}$$

where $Q(\cdot)$ is the complementary cumulative distribution function of a standard normal random variable. In the unjammed state, the cross-over probability is

$$p_{eg} = Q\left(\sqrt{\frac{2E_s}{N_0}}\right). \tag{5.80}$$

These cross-over probabilities correspond to coherent reception of BPSK on additive noise channels with the noise variances for their respective channel states.

Using these probabilities in the Gallager function expression gives

$$E_0(\rho, \mathbf{Q}, m) =$$
$$-\frac{1}{m} \ln \left(\sum_{d=0}^{m} \binom{m}{d} \left[(1-\rho_J)\cdot(1-p_{eg})^{m-d} p_{eg}^d + \rho_J \cdot (1-p_{eb})^{m-d} p_{eb}^d\right]^{\frac{1}{1+\rho}} \right)^{1+\rho} + \rho \ln 2. \tag{5.81}$$

This is computationally tractable, allowing us to calculate the reliability function by numerically optimizing the free parameter ρ. Figure 5.4a shows the error exponent as a function of the rate for several values of the hop length m. In this figure, we have fixed E_s/N_J, and not E_b/N_J, in order to show the classic shape of the curves.

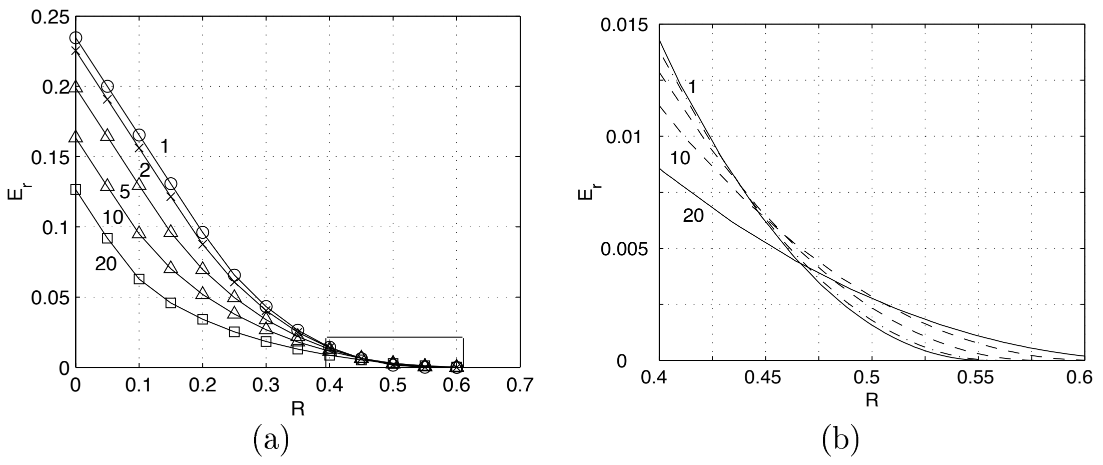


Figure 5.4: Reliability function $E_r(R, m)$ vs. rate R for the two-state jamming channel. (b) expands the boxed portion of (a). The hop lengths for the curves, in order down the ordinates, are 1, 2, 5, 10, and 20 as labeled. The channel parameters are $E_s/N_0 = 7$ dB, $E_s/N_J = -1.5$ dB, $\rho_J = 0.4$.

For $\rho_J = 0.75$ and $E_s/N_0 = 10$ dB, the capacity with CSI $\bar{C}$ is 0.682 bits/channel use and R_1, the rate at which hop lengths longer than 1 become optimal, is about 0.43 bits/channel use. The interesting region is expanded in Fig. 5.4b. Note that there is quite a large region (about 0.51 to 0.68 bits/channel use) where hop lengths of at least 20 are optimal. However, in this region, the reliability function is small, so very long codeword lengths would be needed to achieve reasonable block error rates.

If we consider the alternate definition of optimality based on the error exponent as a function of the channel parameters, we can get an idea of what kind of gains might be achieved by choosing the hop length carefully. Here we fix the channel signal to noise ratio E_b/N_0 and consider how the reliability function depends on the signal-to-jamming power ratio E_b/N_J. This is a measure of how robust the system would be to interference. For a target block error rate bound of 10^{-3} and a block length of 2520, we find that the error exponent target $\hat{E}_r = 0.0027$. Figure 5.5 shows the curves for several hop lengths. The horizontal line corresponds to $\hat{E}_r$. Thus, for the target error rate and block length, the optimal hop length is the one whose curve crosses the $\hat{E}_r$ line furthest to the left. From the picture it is clear that if $\hat{E}_r$ were smaller, perhaps because of a larger codeword length, the optimal hop length would be larger and the gain over hop length $m = 1$ would also increase. Note that because

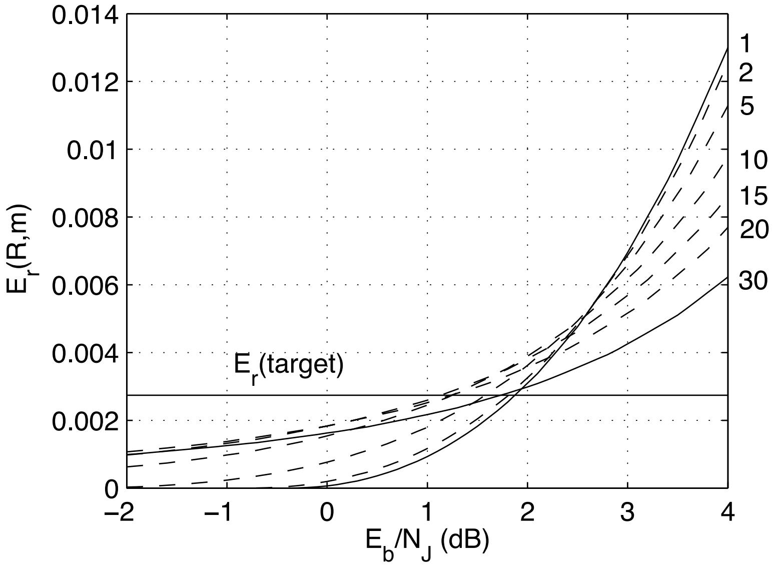


Figure 5.5: Reliability function $E_r(R, m)$ vs. SJNR E_b/N_J for two-state jamming channels. The curves are labeled on the right with their hop lengths. This data has fixed SNR $E_b/N_0 = 12$ dB, and $\rho_J = 0.4$. The horizontal line is the target error exponent for block length 2520 and error probability 10^{-3}.

the bound may be off by some constant factor, the exact numbers are less important than the general behavior.

We can obtain a plot of required E_b/N_J vs. hop length similar to the ones in Chapter 4 by extracting the intersections between the $E_r(R, m)$ curves and the target error exponent. This produces the curve shown in Fig. 5.6. Although it shows the same general trends, this curve is not directly comparable to the ones in Chapter 4 because it is for a hard decision decoder and the decoders described previously use continuous (soft) channel outputs.

The bounding technique seems to provide useful results for LDPC codes operating on these hard decision channels. Although the predicted SJNR values are not very close, the relative performance of receivers with different hop lengths is predicted quite well. Figure 5.7 shows performance bounds and LDPC code simulation results for a block length 2048, rate 1/2 code on the two-state jamming channel. As the bounds predict, hop length $m = 8$ gives the least required SJNR for block error probability 10^{-3}. For $m = 64$, the iterative receiver performs almost as well as the iterative decoder with perfect CSI, just as the bounds suggest. The gaps between the

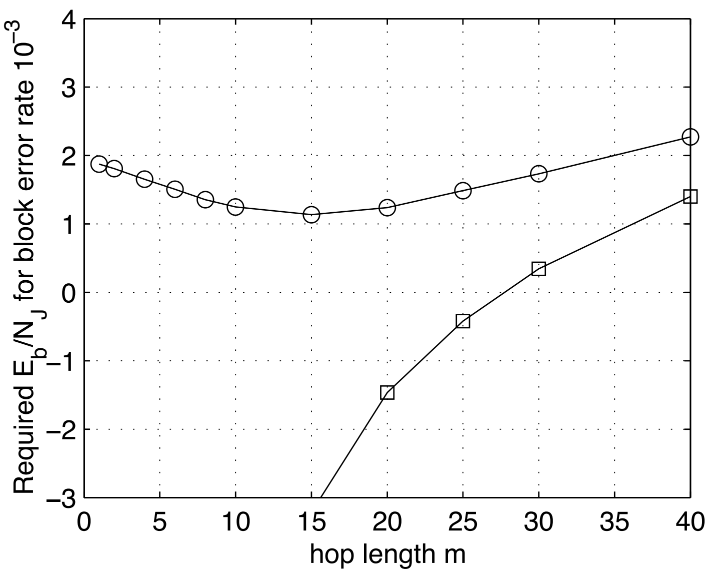


Figure 5.6: Required E_b/N_J for block error rate 10^{-3} vs. hop length for the two-state jamming channel. '$\circ$' joint estimation and decoding, '$\square$' perfect CSI. This is for fixed SNR $E_b/N_0 = 12$ dB, and $\rho_J = 0.4$.

perfect CSI and joint estimation and decoding curves are larger for the bounds, and the LDPC codes without CSI loose about 2.2 dB relative to the bounds for short hop lengths and less for longer hop lengths. We expect that better code constructions than the simple regular LDPC codes that we have used would come substantially closer to the error-exponent bounds.

A key application of these bounds is exploring the parameter space for a class of channels. A classic problem of this type is determining optimum strategies for jammers, transmitters, and receivers in frequency-hopping spread spectrum communication systems. As described previously, we can model these systems as two-state channels with block memory. Here we will be concerned with the jammer's selection of the partial-band jamming fraction ρ_J and the communicators' selection of the hop length m. The objective function or payoff, which the communicator would like to minimize and the jammer maximize, is the communicator's transmitted energy per bit to achieve its desired probability of block error. We assume that the background noise power is fixed, the channel attenuation is fixed, and the jammer has chosen what equivalent full band power spectral density $N_J/2$ it will use. The objective function is plotted as a function of the two game parameters in Fig. 5.8. Figure 5.9 shows

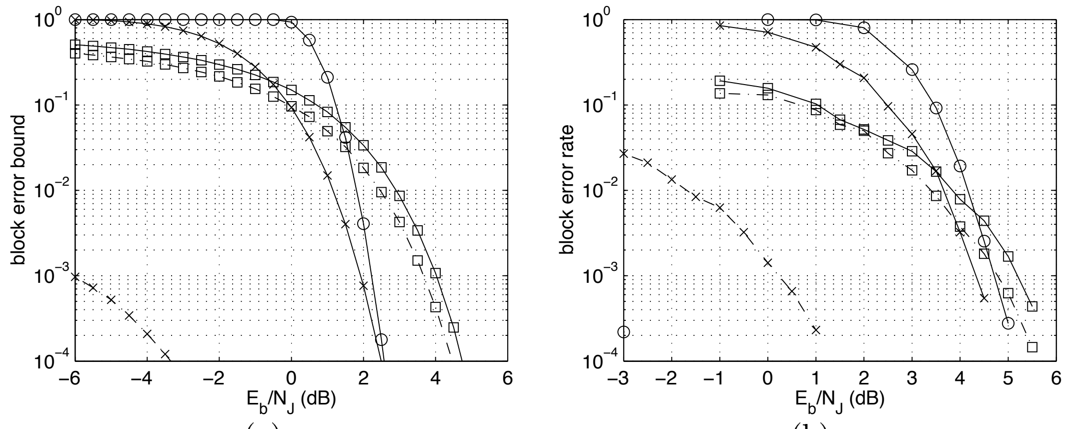


Figure 5.7: Performance from bounds and simulation for the two-state jamming channel. '○' $m = 1$, '×' $m = 8$, '□' $m = 64$. solid–joint channel estimation and decoding, dash-dotted–perfect CSI. (a) bounds from the error exponent, (b) simulation results for a LDPC code. This was for block length 2048 codes with SNR $E_b/N_0 = 10$ dB and $\rho_J = 0.4$.

the same data as several required E_b vs. hop length m curves for different jamming fractions.

The communicator must choose m so that no matter what the jammer's choice of ρ_J, E_b will be as small as possible. Looking at the surface plot (Fig. 5.8), we see that the communicator should choose $m = 1$ since this is the optimum hop length if the jammer chooses $\rho_J \approx 1$ to maximize the communicator's potential required E_b. Presumably, the jammer will choose $\rho_J \approx 1$ because this gives the worst possible performance for the communicator. However, it might be difficult for the jammer to generate a signal that covers the entire hopping bandwidth with sufficient power and therefore the jammer might settle for a smaller jamming fraction. Since using a hop length of about 12 gives very little loss for the large jamming fractions, and has the potential to gain 14 dB or more if the jamming fraction is smaller, we might choose to use a hop length of 12 symbols.

We can also apply the bounding technique to Rayleigh fading channels, such as the coherent Rayleigh fading channel described in section 4.3 except with binary outputs. The channel state is the fade level which can be any non-negative real number. The

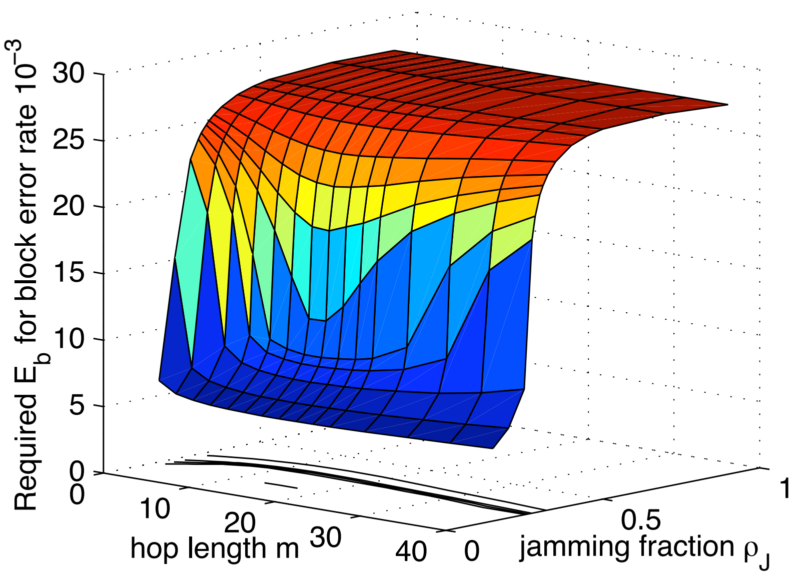


Figure 5.8: Objective function for partial-band jamming game, required transmitter energy E_b vs. jamming fraction ρ_J and hop length m. The required E_b is given in dB relative to the background noise power spectral density. The jammer's equivalent full-band power spectral density is $N_J/2 = 25$ dB relative to the background noise. This data is from the random coding bound with block length 2520 and rate 1/2. Contours of the surface are drawn on the ρ_J-m plane.

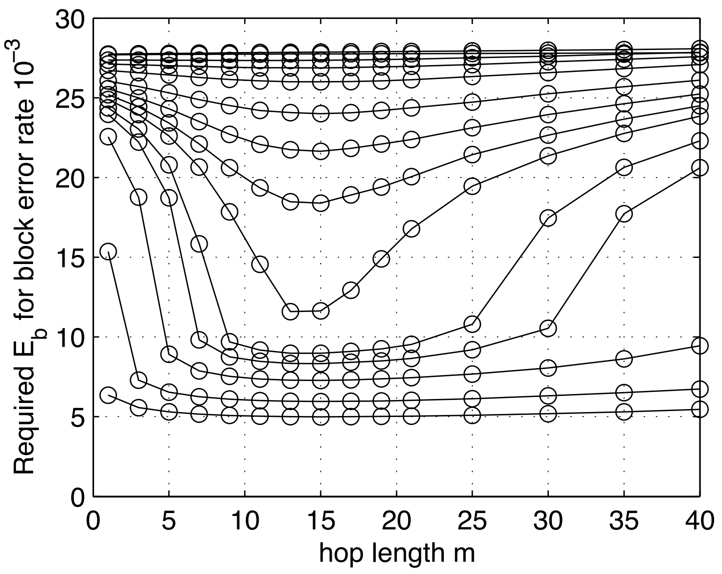


Figure 5.9: Required bit energy E_b vs. hop length for the two-state jamming channel. The required E_b is given in dB relative to the background noise power spectral density. The curves are for increasing ρ_J starting at the bottom. The jammer's equivalent full-band power spectral density is $N_J/2 = 25$ dB relative to the background noise.

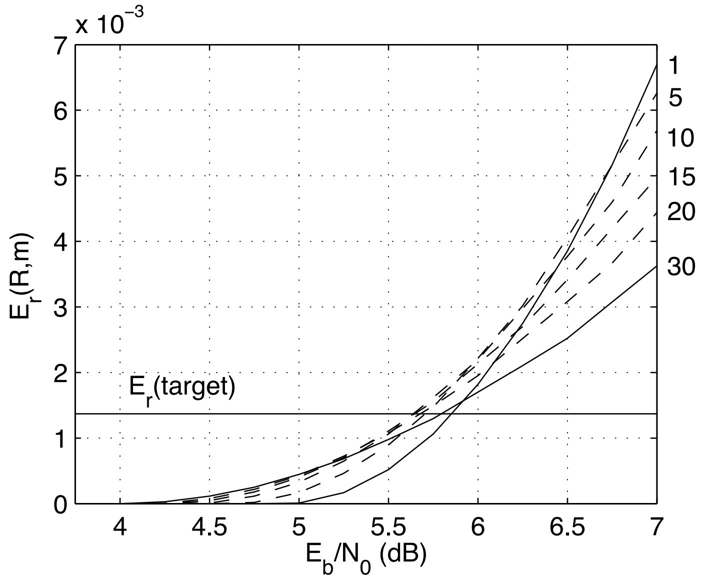


Figure 5.10: Reliability function $E_r(R, m)$ vs. average SNR E_b/N_0 for the coherent Rayleigh fading channel. The hop lengths are labeled on the right side. The horizontal line is the target error exponent for block length 5040 and error probability 10^{-3}.

level is Rayleigh distributed

$$\mathrm{p}(u) = 2ue^{-u^2} \tag{5.82}$$

and the channel cross-over probability p_u is a function of the fade level

$$p_u = Q\left(\sqrt{\frac{2u^2 E_s}{N_0}}\right). \tag{5.83}$$

Substituting this in (5.77), and computing the integrals numerically, it is possible to approximate $E_r(R, m)$. Fig. 5.10 shows the reliability function plotted against the average signal to noise ratio E_b/N_0 for several hop lengths. As in the previous cases, this can be converted into a plot of required average SNR vs. hop length (see Fig. 5.11). Compared to the coherent Rayleigh fading simulations in Chapter 4, the gap between perfect side information and estimation is larger for the binary output bounds. The optimal hop length, about 10 symbols, is also slightly larger for the binary output bounds, although the performance does not depend very much on hop length for these SNR's. Both of these effects are well-explained by the fact that the hard channel outputs provide less information about the fading than the soft outputs considered in Chapter 4.

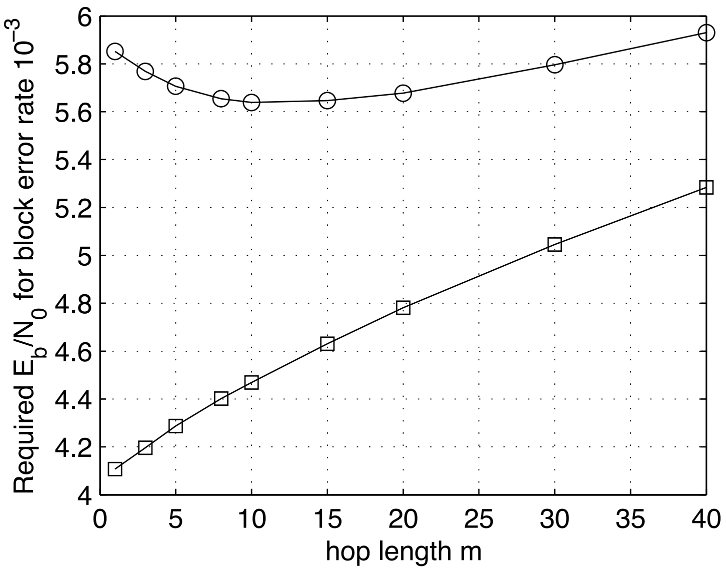


Figure 5.11: Required average SNR E_b/N_0 vs. hop length for the coherent Rayleigh fading channel. '○' joint estimation and decoding, '□' perfect CSI. This assumes a block length 5040, rate 1/2 code.

5.2.2 Non-Coherent Fading Channels

Non-coherent channels, where the receiver does not know the phase of the complex fading factor, pose a difficult problem for joint estimation and decoding because no bit estimates can be formed without either an estimate of the phase offset or some knowledge about the structure of the codewords. The flip channel, used above as an example of an estimation dominated channel, was designed as a very rough model for a binary-input, binary-output non-coherent channel with block memory. It has two equally likely states. In one state, it flips the values of the bits before passing them through a binary symmetric channel, and in the other it does not. The cross-over probability of the binary symmetric channel is taken to be

$$p = Q\left(\sqrt{\frac{2E_s}{N_0}}\right). \tag{5.84}$$

If a typical Costas loop-type phase estimation algorithm were used for BPSK in a frequency-hopping spread spectrum system with AWGN, the phase would be roughly correct up to a factor of 180° that is lost when the data is squared. This gives the behavior of the flip channel if hard decisions are made after the phase recovery loop.

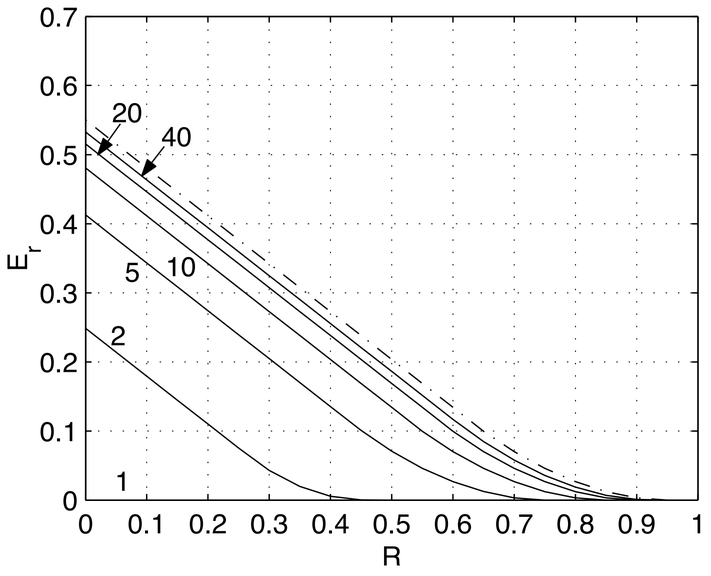


Figure 5.12: Reliability function $E_r(R, m)$ vs. rate R for the flip channel. The curves are labeled with their corresponding hop lengths. This assumes a channel SNR $E_s/N_0 = 5$ dB. Note that we have fixed E_s/N_0, not E_b/N_0 as usual.

As expected for an estimation dominated channel, the error exponent is larger for larger hop lengths m regardless of the code rate. Figure 5.12 shows that the reliability function is 0 for all rates when the hop length is 1, and increases quite quickly toward the reliability function with side information, which is independent of the hop length. This confirms numerically that the channel is estimation dominated, as we had claimed by heuristic arguments above. Figure 5.13 shows that the required SNR decreases monotonically toward the fixed value needed by the perfect CSI receiver. The difference in required SNR between 5 and 20 symbols per hop is over 1.75 dB. This behavior is dramatically different from what we observed for coherent fading channels, and suggests that there are realistic cases where proper choice of the hop length and effective joint channel estimation and decoding can produce large gains over more naive techniques.

We would like to apply this approach to learn more about the non-coherent Rayleigh fading channel with block memory described in section 4.4. As mentioned above, the error exponent with soft outputs is too computationally difficult. We can, however, prove the following interesting result which supports using "pilot symbols" on non-coherent channels. Consider channel models with block memory where the

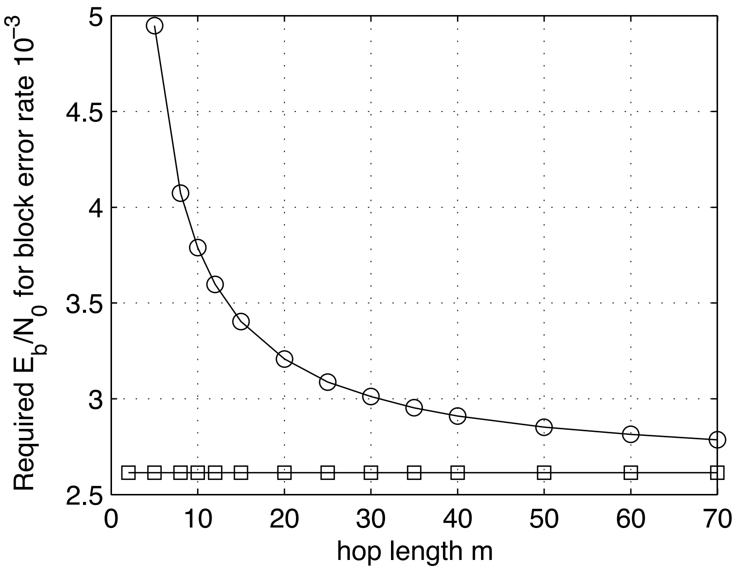


Figure 5.13: Required SNR E_b/N_0 for block error rate 10^{-3} vs. hop length for the flip channel. '$\circ$' joint channel estimation and decoding, '$\square$' perfect CSI. This assumes a block length 2520, rate 1/2 code.

complex output y_i depends on the complex input x_i and the complex channel state u_i according to

$$y_i = u_{\lfloor \frac{i}{m} \rfloor} x_i + n_i \tag{5.85}$$

where n_i is complex additive Gaussian noise with variance $N_0/2$ per dimension. Suppose that the fading factor can be separated into a statistically independent amplitude and phase, and that the phase is uniformly distributed. We will call this a separable non-coherent channel. Note that if we elect to use M-ary phase shift keying (MPSK) symbols as inputs, $x \in \mathcal{X} \subset \mathbb{C}$ is of the form

$$x = Ae^{j\frac{2\pi}{|\mathcal{X}|}l+\theta} \tag{5.86}$$

where A and θ are real constants and l is an integer.

Theorem 5.8 *For a separable non-coherent channel and MPSK signaling, further constraining the signals to have a fixed symbol at the beginning of each hop, i.e. a pilot symbol, does not affect the error exponent.*

Proof: Let $\underline{x}_i \in \{c\} \times \mathcal{X}^{m-1}$ be a vector whose first element is $c \in \mathcal{X}$. Then $\mathcal{X}_i$ is the set of vectors $\underline{x}'$ s.t. $\underline{x}' = e^{j\theta}\underline{x}_i$ for some real $\theta \in [0, 2\pi)$. The sets $\mathcal{X}_i$ form a partition of $\mathcal{X}^m$ because, for any vector $\underline{x}$, there exists $\theta = \angle x_0 - \angle c$ such that $e^{-j\theta}\underline{x} \in \{c\} \times \mathcal{X}^{m-1}$, and if $\underline{x} \in \mathcal{X}_i$ and $\underline{x}' \in \mathcal{X}_j$, then $\underline{x}' \neq e^{j\theta}\underline{x}$ for any $\theta \in [0, 2\pi)$. To verify this last point, suppose that $\underline{x}' = e^{j\theta}\underline{x}$ for $\theta \in [0, 2\pi)$, and note that $\underline{x}' = e^{j\theta'}\underline{x}_i$ and $\underline{x} = e^{j\theta}\underline{x}_j$ implies that $\underline{x}_i = e^{j\theta}\underline{x}_j$ for some $\theta \in [0, 2\pi)$. This is a contradiction because $x_{i,0} = c$ and $x_{j,0} = c$, but $\underline{x}_i \neq \underline{x}_j$.

Consider the Gallager function for a separable non-coherent channel

$$E_0(\rho, \mathbf{Q}, m) = \\ -\frac{1}{m} \ln \int\limits_{\mathcal{Y}^m} \left(\sum_{\underline{x} \in \mathcal{X}^m} Q(\underline{x}) \left[\int\limits_{\mathcal{U}_\alpha} \mathrm{p}(u_\alpha) \int\limits_0^{2\pi} \frac{1}{2\pi} \mathrm{p}(\underline{y}|\underline{x}, u_\alpha, u_\theta)\, du_\theta\, du_\alpha \right]^{\frac{1}{1+\rho}} \right)^{1+\rho} d\underline{y}. \tag{5.87}$$

Partitioning $\mathcal{X}^m$, we find

$$\begin{aligned} &\sum_{\underline{x} \in \mathcal{X}^m} Q(\underline{x}) \left[\int\limits_{\mathcal{U}_\alpha} \mathrm{p}(u_\alpha) \int\limits_0^{2\pi} \frac{1}{2\pi} \mathrm{p}(\underline{y}|\underline{x}, u_\alpha, u_\theta)\, du_\theta\, du_\alpha \right]^{\frac{1}{1+\rho}} \\ &\quad = \sum_i \sum_{\underline{x} \in \mathcal{X}_i} Q(\underline{x}) \left[\int\limits_{\mathcal{U}_\alpha} \mathrm{p}(u_\alpha) \int\limits_0^{2\pi} \frac{1}{2\pi} \mathrm{p}(\underline{y}|\underline{x}, u_\alpha, u_\theta)\, du_\theta\, du_\alpha \right]^{\frac{1}{1+\rho}} \\ &\quad = \sum_i \left(\sum_{\underline{x} \in \mathcal{X}_i} Q(\underline{x}) \right) \left[\int\limits_{\mathcal{U}_\alpha} \mathrm{p}(u_\alpha) \int\limits_0^{2\pi} \frac{1}{2\pi} \mathrm{p}(\underline{y}|\underline{x}_i, u_\alpha, u_\theta)\, du_\theta\, du_\alpha \right]^{\frac{1}{1+\rho}} \end{aligned} \tag{5.88}$$

because for $\underline{x}, \underline{x}' \in \mathcal{X}_i$

$$\mathrm{p}(\underline{y}|\underline{x}, u_\alpha, u_\theta) = \frac{1}{(\sqrt{\pi\sigma^2})^m} \exp\left(-\frac{1}{\sigma^2} \sum_{l=0}^{m-1} |y_l - u_\alpha e^{j u_\theta} x_l|^2 \right) \tag{5.89}$$

$$= \frac{1}{(\sqrt{\pi\sigma^2})^m} \exp\left(-\frac{1}{\sigma^2} \sum_{l=0}^{m-1} |y_l - u_\alpha e^{j(u_\theta - \theta)} x_l'|^2 \right) \tag{5.90}$$

$$= \mathrm{p}(\underline{y}|\underline{x}', u_\alpha, u_\theta - \theta) \tag{5.91}$$

and $\mathrm{p}(\underline{y}|\underline{x}, u_\alpha, u_\theta)$ is periodic in u_θ, so

$$\int\limits_0^{2\pi} \frac{1}{2\pi} \mathrm{p}(\underline{y}|\underline{x}, u_\alpha, u_\theta)\, du_\theta = \int\limits_0^{2\pi} \frac{1}{2\pi} \mathrm{p}(\underline{y}|\underline{x}', u_\alpha, u_\theta)\, du_\theta. \tag{5.92}$$

Choose $\mathbf{Q}'$ so that

$$Q'(\underline{x}_i) = \sum_{\underline{x} \in \mathcal{X}_i} Q(\underline{x}) \tag{5.93}$$

and $Q'(\underline{x}) = 0$ for $\underline{x}$ not in $\{c\} \times \mathcal{X}^{m-1}$. Using (5.88),

$$E_0(\rho, \mathbf{Q}, m) = E_0(\rho, \mathbf{Q}', m) \tag{5.94}$$

which implies that the reliability function is the same for the two sets of inputs, proving the theorem. ■

Note that this theorem actually holds for a larger class of signals that have the right rotational invariance. Because, from a random coding perspective, using DPSK with a reference symbol at the beginning of each hop is equivalent to putting a pilot symbol at the beginning of each hop, this theorem indicates that DPSK does not incur any inherent performance loss. Although it was known that DPSK with an outer code achieves capacity asymptotically in the block length for fixed unknown phase [34], we believe that this result on the error exponent for DPSK with finite memory is new.

Although we cannot compute the reliability function for non-coherent Rayleigh fading with soft outputs, we propose a simple modification of the flip channel that is both tractable and retains the key features. Suppose that the channel either passes the input bits through a binary symmetric channel with cross-over probability p_u which depends on the fading state u, or flips the bits and then passes them through a binary symmetric channel with cross-over probability p_u. The channel state consists of a Bernoulli random variable with probability 1/2 of taking each of its two possible values to determine whether or not to flip the bits, and an independent Rayleigh random variable u for the fading level. As for the coherent Rayleigh fading channel with hard outputs above,

$$p_u = Q\left(\sqrt{\frac{2u^2 E_s}{N_0}}\right), \tag{5.95}$$

and

$$\mathrm{p}(u) = 2ue^{-u^2}. \tag{5.96}$$

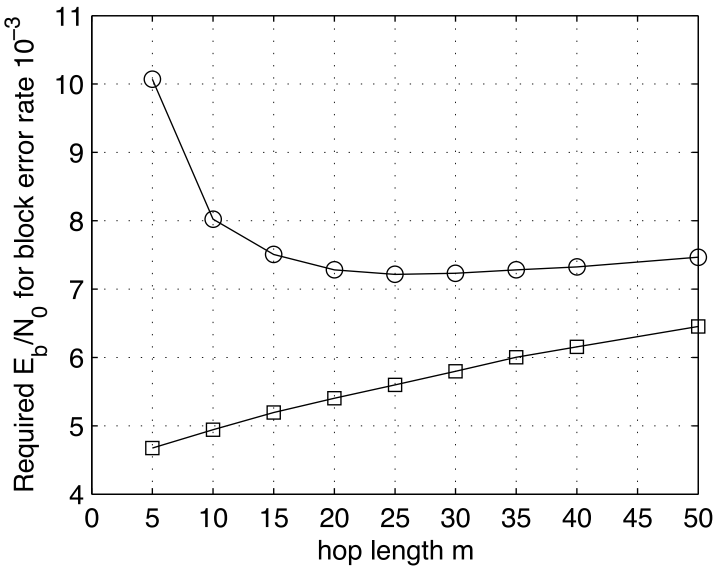


Figure 5.14: Required average SNR E_b/N_0 vs. hop length for the Rayleigh fading flip channel. '$\circ$' joint channel estimation and decoding, '$\square$' perfect CSI. This assumes a block length 2520, rate 1/2 code.

The Gallager function for this case with no channel state information is

$$\begin{aligned} E_0(\rho, \mathbf{Q}, m) &= -\frac{1}{m} \ln\Bigg(\sum_{d=0}^{m} \binom{m}{d} \Bigg[\frac{1}{2} \int_{\mathcal{U}} \mathrm{p}(u)(1-p_u)^{m-d} p_u^d \, du + \\ &\qquad \frac{1}{2} \int_{\mathcal{U}} \mathrm{p}(u)(1-p_u)^d p_u^{m-d} \, du \Bigg]^{\frac{1}{1+\rho}} \Bigg)^{1+\rho} + \rho \ln 2 \\ &= -\frac{1}{m} \ln \left(\sum_{d=0}^{m} \binom{m}{d} \left[\frac{1}{2} \int_{\mathcal{U}} \mathrm{p}(u)((1-p_u)^{m-d} p_u^d + (1-p_u)^d p_u^{m-d}) \, du \right]^{\frac{1}{1+\rho}} \right)^{1+\rho} \\ &\qquad + \rho \ln 2. \end{aligned} \tag{5.97}$$

Figure 5.14 shows the required average SNR E_b/N_0 for this channel model. The optimal hop length, $m = 25$ symbols, gains about 3 dB over hop length $m = 5$. This behavior is similar to that observed for iterative reception on the non-coherent Rayleigh fading channel. The remaining discrepancy can be attributed to two differences: the hard-outputs provided by this channel make channel estimation more difficult, and the code rate for the previous example decreases for small hop lengths because of the pilot symbols.

We have demonstrated that the bounds behave, theoretically and numerically, as predicted by our intuition about effective diversity and channel estimation. The qualitative similarity of the bound results to the simulated performance of the iterative receivers presented in Chapter 4 suggests that the bounds can be used to gain insight for system design.

CHAPTER 6

Iterative Reception: Solutions and Challenges

This thesis has addressed two fundamental questions: "How do we design implementable approximations to joint channel estimation and decoding for arbitrary systems combining multiple coding and modulation techniques?" and "If we could build optimal joint estimation and decoding receivers, how well would they perform?" The effect of channel memory on performance has been key in our discussion.

For the first question, we have proposed a unified framework for designing iterative receivers using factor graphs. Our approach easily provides implementable receivers for a range of common channel models and provides additional insight into previously proposed receivers. It is especially helpful for elucidating the relationship between various designs. The resulting joint channel estimation and decoding algorithms have good performance relative to decoders provided with clairvoyant knowledge of the channel states. We hope that future researchers will use the compactness and precision of factor graphs to document their iterative receiver designs.

Because analysis of iterative receivers remains largely an open problem, we instead considered bounds on the performance of optimal joint estimation and decoding. We suggested an approach based on the reliability function and show analytically and numerically that the bounds follow our intuition about effective diversity and channel estimation accuracy. Most channels exhibit a trade-off between the two that leads to an optimal memory length that increases as we operate at longer codeword lengths. Very simple models can capture the key features of the more complicated channels for which we designed iterative receivers.

A very interesting topic for future work on iterative algorithms is the design of codes especially suited to iterative reception on channels with memory. Nearly all practical channels exhibit time-variation and memory while almost all existing codes are designed for memoryless channels with fixed transition probabilities. Iterative reception poses extra problems for code design because special code properties are needed to start the process, as we saw for non-coherent channels. A theoretical characterization of these code properties and their relationship to the receiver structure would be very interesting.

It would be interesting to extend the techniques of Chapter 5 to channels with Markov memory. This would allow us to treat a number of practically important models for fading channels. It would also provide for a theoretical comparison between the effects of block and Markov memory, much like the experimental one in section 4.2. Extensions to channels that better capture the behavior of continuous channel outputs and techniques for quantitatively predicting the performance of LDPC or turbo codes from the bounds would also be practically useful.

Although the problem of communicating at capacity on the AWGN channel has essentially been solved [56], next generation wireless networks still present a menagerie of challenges. Iterative receivers will be key to taming these problems.

APPENDICES

APPENDIX A

Update Formulas for Example Receivers

This appendix contains the detailed update formulas for the receivers described in Chapter 4. The update formulas for the LDPC code constraint nodes are given in section 2.3.

A.1 Two-State Jamming Channel with Block Memory

Recall the factorization,

$$G(\underline{x}, \underline{u}) = \prod_{j=0}^{r-1} \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\} \cdot \prod_{i=0}^{N-1} \mathrm{p}(y_i | x_i, u_{\lfloor \frac{i}{m} \rfloor}) \cdot \prod_{k=0}^{\frac{N}{m}-1} \mathrm{p}(\underline{u}_k) \tag{A.1}$$

where $\underline{h}_j$ is the j^{th} row of the parity check matrix. From the channel model,

$$\mathrm{p}(y_i | x_i, u_{\lfloor \frac{i}{m} \rfloor}) = \frac{1}{\sqrt{2\pi\sigma^2(u_{\lfloor \frac{i}{m} \rfloor})}} \exp\left(-\frac{(y_i - \sqrt{E_s}(-1)^{x_i})^2}{2\sigma^2(u_{\lfloor \frac{i}{m} \rfloor})} \right) \tag{A.2}$$

and

$$\mathrm{p}(u_k) = \begin{cases} 1 - \rho_J & \text{if } u_k = G \\ \rho_J & \text{if } u_k = B \end{cases} \tag{A.3}$$

The factor graph is shown in Fig. A.1. The updates for the various factor nodes are

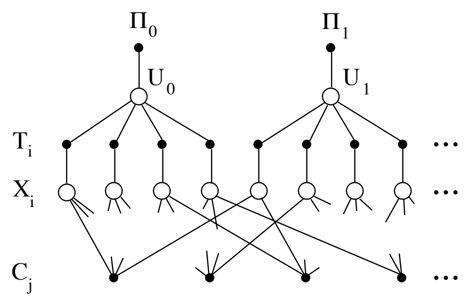


Figure A.1: Factor graph for two-state jamming channel with block memory and LDPC code $m = 4$. $T_i \triangleq \mathrm{p}(y_i|x_i, u_{\lfloor \frac{i}{m} \rfloor})$, $\Pi_k \triangleq \mathrm{p}(u_k)$, $C_j \triangleq \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\}$.

$$\mu_{T_i \to U_k}(u_k) = \frac{1}{\sigma(u_k)} \exp\left(-\frac{(y_i - \sqrt{E_s})^2}{2\sigma^2(u_k)}\right) \mu_{X_i \to T_i}(0) + \frac{1}{\sigma(u_k)} \exp\left(-\frac{(y_i + \sqrt{E_s})^2}{2\sigma^2(u_k)}\right) \mu_{X_i \to T_i}(1) \quad (A.4)$$

$$\mu_{T_i \to X_i}(x_i) = \sum_{u_k \in \mathcal{U}} \frac{1}{\sigma(u_k)} \exp\left(-\frac{(y_i - \sqrt{E_s}(-1)^{x_i})^2}{2\sigma^2(u_k)}\right) \mu_{U_{\lfloor \frac{i}{m} \rfloor} \to T_i}(u_k) \quad (A.5)$$

$$\mu_{\Pi_i \to U_i}(u_i) = \begin{cases} 1 - \rho_J & \text{if } u_i = G \\ \rho_J & \text{if } u_i = B \end{cases} \quad (A.6)$$

A.2 Two-State Jamming Channel with Markov Memory

Recall the factorization

$$G(\underline{x}, \underline{u}) = \prod_{j=0}^{r-1} \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\} \cdot \prod_{i=1}^{N-1} \frac{1}{\sqrt{2\pi\sigma^2(u_i)}} \exp\left(-\frac{(y_i - \sqrt{E_s}(-1)^{x_i})^2}{2\sigma^2(u_i)}\right) \cdot \mathrm{p}(\underline{u}_0) \cdot \prod_{k=1}^{N-1} \mathrm{p}(\underline{u}_k|\underline{u}_{k-1}) \quad (A.7)$$

which has the associated factor graph shown in Fig. A.2.

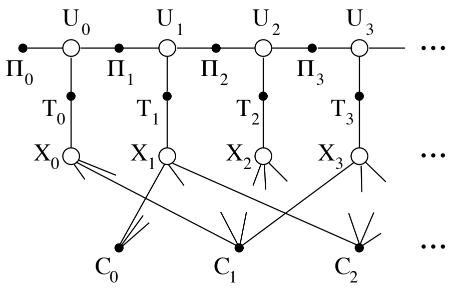


Figure A.2: Factor graph for two-state jamming channel with Markov memory. $\Pi_k \triangleq \mathrm{p}(u_k|u_{k-1})$, $\Pi_0 \triangleq \mathrm{p}(u_0)$, $T_i \triangleq \mathrm{p}(y_i|x_i,u_i)$, $C_j \triangleq \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\}$

The update equations for the factor nodes with appropriate simplifications are

$$\mu_{T_i\to U_i}(u_i) = \frac{1}{\sigma(u_i)}\exp\left(-\frac{(y_i-\sqrt{E_s})^2}{2\sigma^2(u_i)}\right)\mu_{X_i\to T_i}(0) + \frac{1}{\sigma(u_i)}\exp\left(-\frac{(y_i+\sqrt{E_s})^2}{2\sigma^2(u_i)}\right)\mu_{X_i\to T_i}(1) \tag{A.8}$$

$$\mu_{T_i\to X_i}(x_i) = \sum_{u_i\in\mathcal{U}}\frac{1}{\sigma(u_i)}\exp\left(-\frac{(y_i-\sqrt{E_s}(-1)^{x_i})^2}{2\sigma^2(u_i)}\right)\mu_{U_i\to T_i}(u_i) \tag{A.9}$$

$$\mu_{\Pi_0\to U_0}(u_0) = \begin{cases} 1-\rho_J & \text{if } u_0 = G \\ \rho_J & \text{if } u_0 = B \end{cases} \tag{A.10}$$

$$\mu_{\Pi_i\to U_i}(u_i) = \sum_{u_{i-1}\in\mathcal{U}}\mathrm{p}(u_i|u_{i-1})\mu_{U_{i-1}\to\Pi_i}(u_{i-1}) \tag{A.11}$$

$$\mu_{\Pi_i\to U_{i-1}}(u_{i-1}) = \sum_{u_i\in\mathcal{U}}\mathrm{p}(u_i|u_{i-1})\mu_{U_i\to\Pi_i}(u_i) \tag{A.12}$$

A.3 Coherent Rayleigh Fading Channel with Block Memory

Recall the factorization

$$G(\underline{x},\underline{u}) = \prod_{j=0}^{r-1}\mathrm{I}\left\{\underline{h}_j\cdot\underline{x}=0\right\}\cdot\prod_{i=0}^{N-1}\exp\left(-\frac{(y_i-u_{\lfloor\frac{i}{m}\rfloor}(-1)^{x_i}\sqrt{E_s})^2}{N_0}\right)\prod_{k=0}^{\frac{N}{m}-1}2u_k e^{-u_k^2} \tag{A.13}$$

which has the factor graph shown in Fig. A.3.

We roll the Π_k node messages into the updates for the U_k nodes because these use

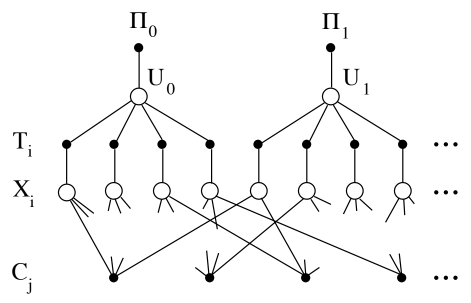


Figure A.3: Factor graph for the Rayleigh fading channel with block memory and LDPC code $m = 4$. $T_i \triangleq \mathrm{p}(y_i|x_i, u_{\lfloor \frac{i}{m} \rfloor})$, $\Pi_k \triangleq \mathrm{p}(u_k)$, $C_j \triangleq \mathrm{I}\left\{\underline{h}_j \cdot \underline{x} = 0\right\}$.

canonical distributions. With the discretized canonical distribution

$$\mu'_{U_k \to T_j}(u_k) = \sum_{l=0}^{L-1} a_l \delta(u_k - \hat{u}_l), \tag{A.14}$$

and the weights a_l are samples of the nominal message at the values $\hat{u}_l$

$$a_l = 2\hat{u}_l e^{-\hat{u}_l^2} \prod_{\substack{i=km \\ i \neq j}}^{km+m-1} \left[\exp\left(-\frac{(y_i - \hat{u}_l\sqrt{E_s})^2}{N_0}\right) \mu_{X_i \to T_i}(0) + \exp\left(-\frac{(y_i + \hat{u}_l\sqrt{E_s})^2}{N_0}\right) \mu_{X_i \to T_i}(1) \right]. \tag{A.15}$$

The parameters y_i, $\mu_{X_i \to T_i}(0)$, and $\mu_{X_i \to T_i}(1)$ are passed to the node U_k in the messages $\mu_{T_i \to U_k}$. The remaining update is

$$\mu_{T_i \to X_i}(x_i) = \sum_{l=0}^{L-1} a_l \exp\left(-\frac{(y_i - \hat{u}_l(-1)^{x_i}\sqrt{E_s})^2}{N_0}\right). \tag{A.16}$$

With the estimate canonical distribution, we again neglect the Π_k messages. We have

$$\mu'_{U_k \to T_j}(u_k) = \delta(u_k - \hat{u}_{k,j}) \tag{A.17}$$

where we select the estimate $\hat{u}_{k,j}$ according to

$$\hat{u}_{k,j} = \frac{1}{(m-1)\sqrt{E_s}} \sum_{\substack{i=km \\ i \neq j}}^{km+m-1} |y_i|. \tag{A.18}$$

The y_i parameters are passed in the messages $\mu_{T_i \to U_k}$. The remaining update is

$$\mu_{T_i \to X_i}(x_i) = \exp\left(-\frac{(y_i - \hat{u}_{k,i}(-1)^{x_i}\sqrt{E_s})^2}{N_0}\right). \tag{A.19}$$

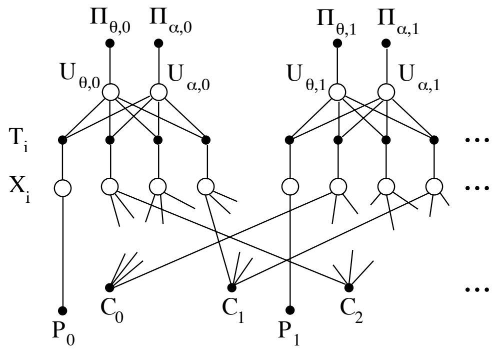


Figure A.4: Factor graph for the non-coherent Rayleigh fading channel with block memory. $\Pi_{\theta,k} \triangleq \mathrm{p}(u_{\theta,k})$, $\Pi_{\alpha,k} \triangleq \mathrm{p}(u_{\alpha,k})$, $T_i \triangleq \mathrm{p}(y_i|x_i, u_{\theta,\lfloor \frac{i}{m} \rfloor}, u_{\alpha,\lfloor \frac{i}{m} \rfloor})$, $P_l \triangleq \mathrm{I}\{x_{lm} = 0\}$, $C_j \triangleq \mathrm{I}\{\underline{h}_j \cdot \underline{x} = 0\}$

A.4 Non-Coherent Rayleigh Fading Channel with Block Memory

The factorization is

$$G(\underline{x}, \underline{u}) = \prod_{l=0}^{\frac{N}{m}-1} \mathrm{I}\{x_{lm} = 0\} \prod_{j=0}^{r-1} \mathrm{I}\{\underline{h}_j \cdot \underline{x} = 0\} \cdot \prod_{i=0}^{N-1} \exp\left(-\frac{1}{N_0}\left|y_i - u_{\alpha,\lfloor \frac{i}{m} \rfloor} \exp(ju_{\theta,\lfloor \frac{i}{m} \rfloor})(-1)^{x_i}\sqrt{E_s}\right|^2\right) \cdot \prod_{k=0}^{\frac{N}{m}-1} 2u_{\alpha,k}e^{-u_{\alpha,k}^2} \prod_{k=0}^{\frac{N}{m}-1} \mathrm{I}\{u_{\theta,k} \in [0, 2\pi)\}. \quad \text{(A.20)}$$

The factor graph corresponding to this factorization is shown in Fig. A.4.

The effects of the Π_α and Π_θ nodes are included in the canonical distribution updates for the $U_{\alpha,k}$ and $U_{\theta,k}$ nodes. For $U_{\alpha,k}$, we use an estimate canonical distribution

$$\mu'_{U_{\alpha,k}\to T_j}(u_k) = \delta(u_{\alpha,k} - \hat{u}_{\alpha,k,j}) \quad \text{(A.21)}$$

where we select estimate $\hat{u}_{\alpha,k,j}$ according to

$$\hat{u}_{\alpha,k,j} = \frac{1}{(m-1)\sqrt{E_s}} \sum_{\substack{i=km \\ i\neq j}}^{km+m-1} |y_i|. \quad \text{(A.22)}$$

The y_i parameters are passed in the messages $\mu_{T_i \to U_{\alpha,k}}$ and $|y_i|$ is the magnitude of the complex observation y_i.

We proposed several canonical distributions for $\mu_{U_{\theta,k}\to T_j}(u_{\theta,k})$. For the bimodal canonical distribution,

$$\mu'_{U_{\theta,k}\to T_j}(u_{\theta,k}) = a_1\delta(u_{\theta,k} - \hat{u}_{\theta,k}) + a_2\delta(u_{\theta,k} - \hat{u}_{\theta,k} + \pi), \tag{A.23}$$

we estimate the phase $u_{\theta,k}$ as

$$\hat{u}_{\theta,k} = \angle(A_{-1}(\underline{\hat{v}}) + B_{-1}(\underline{\hat{v}})\sqrt{-1}) \tag{A.24}$$

where the expressions for $A_{-1}(\underline{v})$, $B_{-1}(\underline{v})$, and an algorithm for finding $\underline{\hat{v}}$ are given in section 4.4.1. The weights a_1 and a_2 are computed by sampling the nominal message at $u_{\theta,k} = u_1$ and $u_{\theta,k} = u_2$ respectively, where $u_1 = \hat{u}_{\theta,k}$ and $u_2 = \hat{u}_{\theta,k} + \pi$, so

$$\begin{aligned} a_l = \prod_{\substack{i=km \\ i\neq j}}^{km+m-1} & \exp\left(-\frac{\left|y_i - \hat{u}_{\alpha,k,i}\exp(ju_l)\sqrt{E_s}\right|^2}{N_0}\right)\mu_{X_i\to T_i}(0) + \\ & \exp\left(-\frac{\left|y_i + \hat{u}_{\alpha,k,i}\exp(ju_l)\sqrt{E_s}\right|^2}{N_0}\right)\mu_{X_i\to T_i}(1) \end{aligned} \tag{A.25}$$

for $l = 1, 2$. The message $\mu_{T_i\to U_{\theta,k}}$ contains the parameters $\hat{u}_{\alpha,k,i}$, y_i, $\mu_{X_i\to T_i}(0)$, and $\mu_{X_i\to T_i}(1)$. The update for $\mu_{T_i\to X_i}$ is

$$\begin{aligned} \mu_{T_i\to X_i}(x_i) = a_1 & \exp\left(-\frac{\left|y_i - \hat{u}_{\alpha,k,i}\exp(j\hat{u}_{\theta,k})(-1)^{x_i}\sqrt{E_s}\right|^2}{N_0}\right) + \\ & a_2\exp\left(-\frac{\left|y_i - \hat{u}_{\alpha,k,i}\exp(j(\hat{u}_{\theta,k}+\pi))(-1)^{x_i}\sqrt{E_s}\right|^2}{N_0}\right). \end{aligned} \tag{A.26}$$

For the two-term exponential CD,

$$\begin{aligned} \mu'_{U_{\theta,k}\to T_j}(u_{\theta,k}) = & C_1\exp(A_1\cos(u_{\theta,k}) + B_1\sin(u_{\theta,k})) + \\ & C_2\exp(A_2\cos(u_{\theta,k}) + B_2\sin(u_{\theta,k})) \end{aligned} \tag{A.27}$$

where we compute the A, B, and C parameters as follows. We select $\underline{\hat{v}}$ and $\underline{\hat{v}}'$ using the algorithm in section 4.4.1. We then compute the parameters for $\mu'_{U_{\theta,k}\to T_j}$ as

$$A_1 = A_j(\underline{\hat{v}}) \qquad A_2 = A_j(\underline{\hat{v}}') \tag{A.28}$$

$$B_1 = B_j(\underline{\hat{v}}) \qquad B_2 = B_j(\underline{\hat{v}}') \tag{A.29}$$

$$C_1 = C_j(\underline{\hat{v}}) \qquad C_2 = C_j(\underline{\hat{v}}'). \tag{A.30}$$

As for the bimodal CD, the message $\mu_{T_i \to U_{\theta,k}}$ contains the parameters $\hat{u}_{\alpha,k,i}$, y_i, $\mu_{X_i \to T_i}(0)$, and $\mu_{X_i \to T_i}(1)$. The update for $\mu_{T_i \to X_i}$ is

$$\mu_{T_i \to X_i}(x_i) = C_1 I_0(K_1(x_i)) + C_2 I_0(K_2(x_i)) \tag{A.31}$$

where $I_0(\cdot)$ is the modified Bessel function of order 0 and

$$K_l(x_i) = \sqrt{[\tfrac{2}{N_0}\hat{u}_{\alpha,k,i}(-1)^{x_i} y_{R,i} + A_l]^2 + [\tfrac{2}{N_0}\hat{u}_{\alpha,k,i}(-1)^{x_i} y_{I,i} + B_l]^2}$$

for $l = 1, 2$.

For the estimate CD, we use

$$\mu'_{U_{\theta,k} \to T_j}(u_{\theta,k}) = \delta(u_{\theta,k} - \hat{u}_{\theta,k,j}) \tag{A.32}$$

and compute the estimate $\hat{u}_{\theta,k,j}$ by finding a common $\underline{\hat{v}}$ as described above and computing phase estimates for each outgoing message using

$$\hat{u}_{\theta,k,j} = \angle(A_j(\underline{\hat{v}}) + B_j(\underline{\hat{v}})\sqrt{-1}). \tag{A.33}$$

The remaining message $\mu_{T_i \to X_i}$ is

$$\mu_{T_i \to X_i}(x_i) = \exp\left(-\frac{1}{N_0} \left| y_i - \hat{u}_{\alpha,\lfloor \frac{i}{m} \rfloor, i} \exp(j\hat{u}_{\theta,\lfloor \frac{i}{m} \rfloor, i})(-1)^{x_i}\sqrt{E_s} \right|^2 \right). \tag{A.34}$$

For the quantized CD, we have

$$\mu'_{U_{\theta,k} \to T_j}(u_{\theta,k}) = \sum_{l=0}^{L-1} a_l \delta(u_{\theta,k} - \hat{u}_l) \tag{A.35}$$

where the quantization points $\hat{u}_l$ are fixed in advance and the weights a_l are computed by sampling the nominal message

$$a_l = \prod_{\substack{i=km \\ i \neq j}}^{km+m-1} \exp\left(-\frac{\left|y_i - \hat{u}_{\alpha,k,i}\exp(j\hat{u}_l)\sqrt{E_s}\right|^2}{N_0} \right) \mu_{X_i \to T_i}(0) + \exp\left(-\frac{\left|y_i + \hat{u}_{\alpha,k,i}\exp(j\hat{u}_l)\sqrt{E_s}\right|^2}{N_0} \right) \mu_{X_i \to T_i}(1). \tag{A.36}$$

This gives the update

$$\mu_{T_i \to X_i}(x_i) = \sum_{l=0}^{L-1} a_l \exp\left(-\frac{1}{N_0} \left| y_i - \hat{u}_{\alpha,\lfloor \frac{i}{m} \rfloor, i} \exp(j\hat{u}_l)(-1)^{x_i}\sqrt{E_s} \right|^2 \right). \tag{A.37}$$

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] W. K. M. Ahmed, *Information Theoretic Reliability Function for Flat Fading Channels*, PhD thesis, Queen's University, Kingston, Ontario, Canada, September 1997.

[2] W. K. M. Ahmed and P. J. McLane, "Random coding error exponents for flat fading channels," in *Proceedings of the IEEE International Symposium on Information Theory ISIT'98*, p. 394, Cambridge, MA, USA, August 1998.

[3] W. K. M. Ahmed and P. J. McLane, "Random coding error exponents for flat fading channels with realistic channel estimation," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 369–379, March 2000.

[4] S. M. Aji and R. J. McEliece, "A general algorithm for distributing information in a graph," in *Proceedings of the International Symposium on Information Theory ISIT'97*, p. 6, June 1997.

[5] S. M. Aji and R. J. McEliece, "The generalized distributive law," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 325–349, March 2000.

[6] A. Anastasopoulos and K. M. Chugg, "Adaptive SISO algorithms for iterative detection with parametric uncertainty," in *Proceedings of the IEEE International Conference on Communication ICC'99*, Vancouver, Canada, June 1999.

[7] A. Anastasopoulos and K. M. Chugg, "Adaptive soft-input soft-output algorithms for iterative detection with parametric uncertainty," *IEEE Transactions on Communications*, vol. 48, no. 10, pp. 1638–1649, October 2000.

[8] C. Antón-Haro, J. A. R. Fonollosa, and J. R. Fonollosa, "Blind channel estimation and data detection using hidden Markov models," *IEEE Transactions on Signal Processing*, vol. 45, no. 1, pp. 241–247, January 1997.

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

[10] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding," in *Proceedings of the International Communications Conference ICC'93*, pp. 1064–1070, May 1993.

[11] E. Biglieri, J. Proakis, and S. Shamai, "Fading channels: Information-theoretic and communications aspects," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2619–2692, October 1998.

[12] G. Caire and S. Shamai, "On the capacity of some channels with channel state information," *IEEE Transactions on Information Theory*, vol. 45, no. 6, pp. 2007–2019, September 1999.

[13] M. J. Chu, D. L. Goeckel, and W. E. Stark, "On the design of Markov models for fading channels," in *Proceedings of the IEEE Vehicular Technology Conference VTC Fall '99*, Amsterdam, The Netherlands, September 1999.

[14] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[15] A. Duel-Hallen and C. Heegard, "Delayed decision-feedback sequence estimation," *IEEE Transactions on Communications*, vol. 37, no. 5, pp. 428–436, May 1989.

[16] B. J. Frey, F. R. Kschischang, H.-A. Loeliger, and N. Wiberg, "Factor graphs and algorithms," in *Proceedings of the 35th Annual Allerton Conference on Communication, Control and Computing*, pp. 666–680, September 1997.

[17] R. G. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. IT-8, no. 1, pp. 21–28, January 1962.

[18] R. G. Gallager, *Low-Density Parity-Check Codes*, M.I.T. Press, Cambridge, MA, 1963.

[19] R. G. Gallager, *Information Theory and Reliable Communication*, John Wiley & Sons, New York, 1968.

[20] J. Garcia-Frias and J. D. Villasenor, "Exploiting binary Markov channels with unknown parameters in turbo decoding," in *Proceedings of the IEEE Global Communications Conference GLOBECOM'98*, pp. 3244–3249, Sydney, Australia, November 1998.

[21] J. Garcia-Frias and J. D. Villasenor, "Blind turbo decoding and equalization," in *Proceedings of the IEEE Vehicular Technology Conference VTC Spring '99*, volume 3, pp. 1881–1885, Houston, TX, May 1999.

[22] J. Garcia-Frias and J. D. Villasenor, "Simplified methods for combining hidden Markov models and turbo codes," in *Proceedings of the 50th IEEE Vehicular Technology Conference VTC Fall '99*, pp. 1580–1584, Amsterdam, The Netherlands, September 1999.

[23] E. N. Gilbert, "Capacity of a burst-noise channel," *Bell System Technical Journal*, vol. XXXIX, no. 5, pp. 1253–1265, Sept. 1960.

[24] J. Hagenauer, "The Turbo principle: Tutorial introduction and state of the art," in *Proceedings of the International Symposium on Turbo Codes*, pp. 1–11, Brest, France, September 1997.

[25] G. H. Hardy, J. E. Littlewood, and G. Pólya, *Inequalities*, Cambridge University Press, London, 1934.

[26] C. Heegard and S. B. Wicker, *Turbo Coding*, Kluwer Academic Publisher, Boston, 1999.

[27] P. Hoeher and J. Lodge, "'Turbo DPSK': Iterative differential PSK demodulation and channel decoding," *IEEE Transactions on Communications*, vol. 47, no. 6, pp. 837–843, June 1999.

[28] S. Jayaraman and H. Viswanathan, "Optimal detection of a jammed channel," in *Proceedings of the 1996 IEEE Global Telecommunications Conference GLOBECOM'96*, pp. 87–91, November 1996.

[29] S. Jayaraman and H. Viswanathan, "Simultaneous communication and detection over compound channels," in *Proceedings of the 1998 International Symposium on Information Theory ISIT'98*, p. 40, August 1998.

[30] A. Jimenez, *Soft Iterative Decoding of Low-Density Convolutional Codes*, Engineering licenciate thesis, Lund University, October 1997.

[31] J. H. Kang and W. E. Stark, "Turbo codes for noncoherent FH-SS with partial band interference," *IEEE Transactions on Communications*, vol. 46, no. 11, pp. 1451–1458, November 1998.

[32] J. H. Kang, W. E. Stark, and A. O. Hero, "Turbo codes for fading and burst channels," in *Proceedings of the IEEE Global Communications Conference GLOBECOM'98: Communication Theory Mini-Conference*, pp. 40–45, November 1998.

[33] G. Kaplan and S. Shamai, "Error exponents and outage probabilities for the block-fading Gaussian channel," in *Proceedings of the IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications PIMRC'91*, pp. 329–334, September 1991.

[34] G. Kaplan and S. Shamai, "On the achievable information rates of DPSK," *IEE Proceedings-I*, vol. 139, no. 3, pp. 311–318, June 1992.

[35] C. Komninakis and R. D. Wesel, "Iterative joint channel estimation and decoding in flat correlated Rayleigh fading," in *Proceedings of the 7^{th} International Conference on Advances in Communications and Control*, Athens, Greece, June 1999.

[36] C. Komninakis and R. D. Wesel, "Pilot-aided joint data and channel estimation in flat correlated Rayleigh fading," in *Proceedings of the IEEE Global Communications Conference GLOBECOM'99*, 1999.

[37] F. R. Kschischang and B. J. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 219–230, February 1998.

[38] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, February 2001.

[39] A. Lapidoth and S. Shamai, "Fading channels: How perfect need "perfect side information" be?," in *Proceedings of the 1999 IEEE Information Theory Workshop*, pp. 36–38, Kruger National Park, South Africa, June 1999.

[40] C. P. Liang, *Resource Allocation and System Optimization for Spread Spectrum and OFDM Networks with Multi-User Detection*, PhD thesis, University of Michigan, Ann Arbor, MI, 2001.

[41] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Efficient erasure correcting codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 569–584, February 2001.

[42] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Transactions on Information Theory*, vol. 45, no. 2, pp. 399–431, March 1999.

[43] I. D. Marsland and P. T. Mathiopoulos, "Multiple differential detection of parallel concatenated convolutional (turbo) codes in correlated fast Rayleigh fading," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 265–274, February 1998.

[44] I. D. Marsland and P. T. Mathiopoulos, "On the performance of iterative non-coherent detection of coded M-PSK signals," *IEEE Transactions on Communications*, vol. 48, no. 4, pp. 588–596, April 2000.

[45] T. L. Marzetta and B. M. Hochwald, "Multiple-antenna communciations when nobody knows the Rayleigh fading coefficients," in *Proceedings of the 35th Allerton Conference on Communication, Control, and Computing*, pp. 1033–1042, September 1997.

[46] T. L. Marzetta and B. M. Hochwald, "Capacity of a mobile multiple-antenna communication link in flat rayleigh fading," *IEEE Transactions on Information Theory*, vol. 45, no. 1, pp. 139–157, January 1999.

[47] R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, "Turbo decoding as an instance of Pearl's 'belief propagation' algorithm," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 140–152, February 1998.

[48] R. J. McEliece and W. E. Stark, "Channels with block interference," *IEEE Transactions on Information Theory*, vol. IT-30, no. 1, pp. 44–53, January 1984.

[49] R. J. McEliece and M. Xu, "Junction tree representations for linear block codes," in *Proceedings of the 1998 International Symposium on Information Theory ISIT'98*, p. 253, August 1998.

[50] M. Medard, "The effect upon channel capacity in wireless communications of perfect and imperfect knowledge of the channel," *IEEE Transactions on Information Theory*, vol. 46, no. 3, pp. 933–946, May 2000.

[51] N. Merhav, G. Kaplan, A. Lapidoth, and S. Shamai, "On information rates for mismatched decoders," *IEEE Transactions on Information Theory*, vol. 40, no. 6, pp. 1953–1967, Nov. 1994.

[52] M. Mushkin and I. Bar-David, "Capacity and coding for the Gilbert-Elliott channels," *IEEE Transactions on Information Theory*, vol. 35, no. 6, pp. 1277–1290, November 1989.

[53] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, 1988.

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

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

[56] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching low-density parity check codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 619–637, February 2001.

[57] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity check codes under message-passing decoding," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 599–618, February 2001.

[58] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 623–656, October 1948.

[59] M. Sipser and D. A. Spielman, "Expander codes," *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1710–1722, Nov. 1996.

[60] W. E. Stark, "Coding for frequency-hopped spread-spectrum communication with partial-band interference–part I: Capacity and cutoff rate," *IEEE Transactions on Communications*, vol. COM-33, no. 10, pp. 1036–1044, October 1985.

[61] W. E. Stark, "Coding for frequency-hopped spread-spectrum communication with partial-band interference–part II: Coded performance," *IEEE Transactions on Communications*, vol. COM-33, no. 10, pp. 1045–1057, October 1985.

[62] I. G. Stiglitz, “Coding for a class of unknown channels,” *IEEE Transactions on Information Theory*, vol. IT-12, no. 2, pp. 189–195, April 1966.

[63] I. G. Stiglitz, “A coding theorem for a class of unknown channels,” *IEEE Transactions on Information Theory*, vol. IT-13, no. 2, pp. 217–220, April 1967.

[64] H.-J. Su and E. Geraniotis, “Improved performance of a PSAM system with iterative filtering and decoding,” in *Proceedings of the 36th Annual Allerton Conference on Communication, Control, and Computing*, pp. 156–166, September 1998.

[65] R. M. Tanner, “A recursive approach to low complexity codes,” *IEEE Transactions on Information Theory*, vol. IT-27, no. 5, pp. 533–547, September 1981.

[66] E. Uysal and R. G. Gallager, “Observations on diversity limitation in slow frequency hopping,” `http://www.stanford.edu/~elif/SFH.pdf`, September 2000.

[67] M. C. Valenti and B. D. Woerner, “A bandwidth efficient pilot symbol technique for coherent detection of turbo codes over fading channels,” in *Proceedings of the 1999 Military Communications Conference MILCOM’99*, October 1999. paper 3.4.

[68] N. Wiberg, *Codes and Decoding on General Graphs*, PhD thesis, Linköping University, Linköping, Sweden, 1996.

[69] N. Wiberg, H.-A. Loeliger, and R. Koetter, “Codes and iterative decoding on general graphs,” *European Transactions on Telecommunications*, vol. 6, no. 5, pp. 513–525, Sept.–Oct. 1995.

[70] J. Wolfowitz, *Coding Theorems of Information Theory*, Springer-Verlag, Berlin, 3rd edition, 1978.

[71] Z.-N. Wu and J. M. Cioffi, “Turbo decision aided equalization for magnetic recording channels,” in *Proceedings of the IEEE Global Telecommunications Conference GLOBECOM’99*, pp. 733–738, 1999.

[72] Y. Xu, H.-J. Su, and E. Geraniotis, “Pilot symbol assisted QAM with iterative filtering and turbo decoding over Raleigh flat fading channels (sic),” in *Proceedings of the 1999 Military Communications Conference MILCOM’99*, Oct. 1999. paper 3.5.