Analysis and Design of Turbo-like Codes

Thesis by

Hui Jin

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

2001

(Submitted May 23, 2001)

To
my Parents
and
my wife Yingying

Acknowledgements

Looking back at the years I spent at Caltech, I would like to express my deepest gratitude to my advisor Prof. Robert J. McEliece. Without his support and guidance, this thesis would have been simply a dream. It was his wonderful lectures on information and coding theory that first attracted me into this field, then his appreciation led me into his group as an undergraduate and later as a graduate student. Not only Prof. McEliece's insights and enthusiasm on research problems, but also his devotion to education and academia have inspired me as to what a true world-renowned scientist should be. He is a role model that I can always look up to.

My heartfelt thanks go to Dr. Dariush Divsalar at Jet Propulsion Laboratory. His knowledge in each and every corner of communication theory has been a tremendous resource for me. Discussions and collaborations with him are always informative and fruitful.

I also would like to thank Prof. Robert J. McEliece, Prof. Jehoshua (Shuki) Bruck, Prof. Michelle Effros at Caltech, and Dr. Dariush Divsalar, Dr. Samuel Dolinar at JPL for serving in my Ph.D. examination committee, perusing my thesis manuscripts, and giving their insightful advice.

Dr. Masayuki Hottori was very supportive of my work for providing me a summer scholar position at Sony information-network technologies lab, Tokyo, in 2000. That experience was truly valuable.

I am deeply grateful to our group secretary Lilian Porter for her professional assistance on administrative matters and her care and kindness. My special thanks also go to our former and current system administrators Robert Freeman, Naveed Near-Ansari, and Dimitris Sakellariou, for their effort to keep the computers working smoothly.

This thesis is made possible also by the gracious and enriching environment of California Institute of Technology. The glorious legacy of Caltech and the stimulating atmosphere abundant on campus have been always an encouragement for me to keep pursuing. In particular, I am happy to convey my deep appreciation to my good friends Yi Li (Caltech), Tai Lam (UW), Changchun Shi (UC Berkeley), Alan Yue (Caltech); to my current and former group mates Aamod Khandekar, Ravi Palanki, Jeremy Thorpe, Srinivas M. Aji, Gavin Horn, Meina Xu, Mohamed-Slim Alouini, Lifang Li, Neelesh B. Mehta, Diego G. Dugatkin, Hanying Feng, Michael Felming, Sidharth Jaggi, and Qian Zhao.

Lastly, I would like to thank my family. My parents, Shunyun Jin and Meizheng Li, taught me those most valuable “theorems of life” which I could not learn from any text book or paper. Their love, support, and encouragement have been with me every single moment. I am deeply indebted to them, as well as to my sister Ping Jin and my grandmother. My wife Yingying has offered me the moral support and loving encouragement only she can offer. This thesis is dedicated to her and my parents as an inadequate but sincere expression of appreciation and love.

Abstract

Fifty years after Shannon determined the capacity of memoryless channels, we finally know of practical encoding and decoding algorithms that closely approach this limit. This remarkable feat was first achieved by the invention of turbo codes in 1993 [5]. Since then, turbo codes have essentially revolutionized the coding field and became one of the central research problems in recent years. While there has been a great deal of excellent theoretical work on turbo codes, it is fair to say practice still leads theory by a considerable margin.

This thesis endeavors to fill some of that gap. The main body of the thesis concerns coding theorems for general turbo codes. By "coding theorems" we mean in a classical Shannon sense: for a code ensemble, there exists a code threshold, if the channel noise is below that, the error probability of optimal decoder goes to zero as the code length approaches infinity. We first prove coding theorems for some simple serial turbo code ensembles on the AWGN channel. Then we generalize the results for a broader class of turbo-like codes on any memoryless channel. To closely estimate the noise threshold in the coding theorems, we develop a method based on "typical pairs decoding." This method is powerful enough to reproduce Shannon's original coding theorems on any memoryless binary-input symmetric channels. Finally we introduce a class of codes of linear encoding and decoding complexity with performance provably close to Shannon's limit.

One main contribution of both theoretical and practical interests in this thesis is the introduction of (regular and irregular) "repeat-accumulate" codes. RA codes are shown to be a serious competitor against turbo codes and LDPC codes.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Coding for Digital Data Transmission

The need for efficient and reliable data communication over noisy channels has been rising rapidly for decades. This includes applications to telephone modems, wireless communication, deep-space communication, internet data transfer, data storage devices, etc.

The fundamental approach to the problems of efficiency and reliability in communication systems is contained in the Noisy Channel Coding Theorem developed by C. E. Shannon [46] in 1948. Shannon's theorem states that over a noisy channel, if R is less than the channel capacity C, there exists a coding scheme of code rate R with arbitrarily small error probability. The proof to the theorem is essentially non-constructive. It shows that for long block length, almost all codes of rate R ($< C$) would be reliable. However, it does not give an explicit construction of capacity-approaching codes, nor does it lay out practical decoding algorithms.

In the 50 years since Shannon determined the capacity of noisy channels, the construction of capacity-approaching coding schemes has been the supreme goal of coding research. But it was not until the early 90s that we saw the first class of codes whose performance practically approaches Shannon's theoretical limit. In 1993, Berrou, Glavieux, and Thitimajshima [5] introduced turbo codes to the world. The performance of turbo codes is typically within 1 dB of the Shannon limit on the AWGN channel.

In brief, the novelty of turbo codes lies in the pseudorandom interleaver and iterative decoding. The pseudorandom interleaver introduces enough randomness to achieve reliable communication at data rates near capacity, yet it has enough structure to allow practical encoding and decoding algorithms. The invention of turbo codes has revolutionized the field of error-correcting codes. It led to the rediscovery of low-density parity-check codes [24, 52, 38], and the discovery of the connection between iterative decoding and belief propagation and inference problems [52, 40, 3, 34]. Also, it has ignited an explosion of interest in the graphical model representations of codes [48, 52, 34]. Based on these insights, many different powerful schemes have been proposed. Finally today, we know of several practical codes and decoding algorithms that closely approach the capacity of some classical memoryless channels. Among those, turbo codes and low-density parity-check codes are still the two central techniques. While theoretical research on low-density parity-check codes was laid out by Gallager [24] in 1963, most research on turbo codes to date has been experimental, and many theoretical questions still remain open.

This thesis deals with the analysis and design of turbo and turbo-like codes. Two main problems motivate this thesis: proving coding theorems for general turbo codes and designing better turbo codes. The first problem concerns a theoretical justification of turbo codes: we move beyond the experimental demonstrations of the sterling performance to prove that turbo codes are indeed a class of "good" codes. (By "good" codes we mean code families that achieve arbitrarily small error probability at some fixed non-zero code rate, which may be less than channel capacity.) The second problem is an attempt to come up with a design methodology that can lead to turbo codes achieving rates closer to the channel capacity.

1.2 Thesis Outline

The thesis is organized in such a way that different chapters can be read independently, except Chapter 2, which contains some common material for all the other chapters.

In Chapter 2, general code ensembles are first defined. We focus on turbo-like code ensembles, which include classical turbo codes and serial turbo codes as special instances. Then different channel models and union bounds over those channels are briefly reviewed. Finally, a conjecture (Interleaving Gain Exponent conjecture) about maximum-likelihood decoding performance of general turbo-like codes is presented.

Proving this conjecture is the main subject of Chapters 3, 4, and 5.

In Chapter 3, a simple class of serial turbo codes called repeat-accumulate (RA) codes is introduced. The coding theorem for RA codes is proved for additive white Gaussian noise channels by using the classical union bound. We believe this theorem is the first rigorous proof of a coding theorem for any class of turbo-like codes. We present the Tanner graph representation and the corresponding iterative decoding method for RA codes. Surprisingly, despite their extremely simple structure and sub-optimum iterative decoding, RA codes have performance comparable to the full-fledged turbo codes.

In Chapter 4, several variations of RA codes are investigated. We show that those codes generally have better maximum likelihood decoding thresholds, but not necessarily better performance with iterative decoding.

It is shown in Chapter 5 that the techniques developed in Chapter 3 are sufficient to prove the IGE conjecture for general turbo codes. This is proven for all parallel and serial turbo codes on any memoryless binary-input channel, subject to only mild restrictions.

In Chapter 6, a general upper bound on the probability of decoding error for symmetric binary-input channels with typical pairs decoding is derived for both individual codes and arbitrary ensembles of codes. This bound is most useful in the infinite block length scenario, as it provides an easy way to closely estimate the code thresholds with maximum likelihood decoding. This method completely decouples the channel from the code ensembles. In this, it resembles the classical union bound, but unlike the union bound, it is powerful enough to reproduce Shannon's channel coding theorem on many standard channels.

In Chapter 7, an ensemble of codes called *irregular repeat accumulate* (IRA) codes is introduced. IRA codes are a generalization of the repeat-accumulate codes. With careful design, IRA codes can achieve channel capacity on the binary erasure channel, and perform close to channel capacity on the additive white Gaussian channel. Remarkable performance is also obtained on the Rayleigh fading channels, although no design techniques have been proposed.

There are five appendices which contain supplementary material. In particular, Appendix E shows a hardware implementation of iterative decoding algorithm of RA codes introduced in Chapter 3.

Chapter 2 Turbo-like Code Ensembles

Throughout this thesis, we will mainly consider ensembles of coding systems which are built from fixed convolutional codes interconnected with random interleavers. We call these systems "turbo-like" codes and they include as special cases both the classical turbo codes [5, 6, 7] and the serial concatenation of interleaved convolutional codes [8]. In Chapters 3, 4, and 5 we will be concerned with a general conjecture about the behavior of those turbo-like code ensemble (maximum-likelihood decoder) word/bit error probability as the word length approaches infinity. This chapter provides some background material for the following chapters.

2.1 Code Ensembles

In this section we will give a careful definition of a code ensemble. By an *ensemble* of linear codes we mean a sequence $\mathcal{C}_{n_1}, \mathcal{C}_{n_2}, \ldots$ of sets of linear codes, where $\mathcal{C}_{n_i}$ is a set of (n_i, k_i) codes with common rate $R_i = k_i/n_i$. We assume that the sequence $n_1, n_2, \ldots$ approaches infinity, and that $\lim_{i\to\infty} R_i = R$, where R is called the rate of the ensemble.

We shall be concerned with the weight structure of the ensemble, and with this in mind we introduce some notation. If C is an (n, k) linear code, we denote its weight enumerator by the list $A_0(C), A_1(C), \ldots, A_n(C)$. In other words, $A_h(C)$ is the number of words of weight h in C, for $h = 0, 1, \ldots, n$. When no ambiguity is likely to occur, we denote the weight enumerator simply by $A_0, A_1, \ldots, A_n$. We also introduce the *cumulative weight enumerator*

$$A_{\leq h} = \sum_{d=1}^{h} A_d \qquad \text{for } h = 1, \ldots, n. \tag{2.1}$$

In words, $A_{\leq h}$ is the number of *nonzero* codewords of weight $\leq h$.

When the code C is viewed as the set of possible outputs of a particular encoder E, then we denote by $A^{(E)}_{w,h}$ the number of (x, y) pairs where the encoder input x has weight w and the corresponding encoder output y (codeword) has weight h. Usually the encoder will be understood, and the simpler notation $A_{w,h}$ will do. The set of numbers $A_{w,h}$ where w ranges from 0 to k and h ranges from 0 to n is called the input-output weight enumerator (IOWE) for the code. In analogy with (2.1) we define the cumulative input-output weight enumerator (CIOWE):

$$A_{w,\leq h} = \sum_{d=1}^{h} A_{w,d}. \tag{2.2}$$

Returning now to the ensemble, we define the *average weight enumerator* for the set $\mathcal{C}_n$ as the list

$$\overline{A}^{(n)}_0, \overline{A}^{(n)}_1, \ldots, \overline{A}^{(n)}_n,$$

where

$$\overline{A}^{(n)}_h \triangleq \frac{1}{|\mathcal{C}_n|} \sum_{C \in \mathcal{C}_n} A_h(C) \qquad \text{for } h = 0, 1, \ldots, n. \tag{2.3}$$

Similarly, we define the average cumulative weight enumerator $\overline{A}^{(n)}_{\leq h}$, the average IOWE $\overline{A}^{(n)}_{w,h}$, and the average CIOWE $\overline{A}^{(n)}_{w,\leq h}$.

We define, for each n in the sequence $n_1, n_2, \ldots$, the nth spectral shape function

$$r_n(\delta) \triangleq \frac{1}{n} \log \overline{A}^{(n)}_{\lfloor \delta n \rfloor} \qquad \text{for } 0 < \delta < 1. \tag{2.4}$$

Thus $\overline{A}^{(n)}_h = e^{n r_n(\delta)}$, where $\delta = h/n$.

Finally, we define the *asymptotic spectral shape* :

$$r(\delta) \triangleq \lim_{n \to \infty} r_n(\delta) \qquad \text{for } 0 < \delta < 1, \tag{2.5}$$

provided the limit exists. In this case, we can say, roughly, that for large n, if the ratio $\delta = h/n$ is fixed, then

$$\overline{A}_h^{(n)} \sim e^{nr(\delta)}.$$

2.2 Memoryless Binary-Input Channels and the Union Bound

Since turbo codes, as we have defined them, are binary codes, we consider using them on memoryless binary input channels. Such a channel has binary input alphabet $\{0, 1\}$ and arbitrary output alphabet Ω. If the channel input is a binary random variable X, then the channel output is a random variable Y. If Ω is finite, then Y is characterized by transition probabilities $p(y|0)$, $p(y|1)$, i.e., for $y \in \Omega$, $i \in \{0, 1\}$,

$$\Pr\{Y = y|X = i\} = p(y|i).$$

If Ω is a subset of R^r, where R is the real line, then Y is characterized by transition probability densities $p(y|0)$, $p(y|1)$, i.e., if S is a measurable subset of Ω, $i \in \{0, 1\}$.

$$\Pr\{Y \in S|X = i\} = \int_S p(y|i)dy.$$

The "noisiness" of the channel can be summarized by the *Bhattacharya parameter* γ, which is defined by

$$\gamma = \sum_{y \in \Omega} \sqrt{p(y|0)p(y|1)}, \tag{2.6}$$

if Ω is finite and

$$\gamma = \int_\Omega \sqrt{p(y|0)p(y|1)}dy, \tag{2.7}$$

if $\Omega = R^r$. It is easy to see (by the Cauchy-Schwarz inequality) that $\gamma \leq 1$, with equality if and only if $p(y|0) = p(y|1)$ for all y, in which case the channel has capacity zero.[1]

For example, for a binary erasure channel with erasure probability p, we have

$$\gamma_{\mathrm{BEC}} = p.$$

Similarly, for a binary symmetric channel with crossover probability p we have

$$\gamma_{\mathrm{BSC}} = 2\sqrt{p(1-p)}.$$

Also, for the asymmetric "Z" channel, we have

$$\gamma_{\mathrm{Z}} = \sqrt{p}.$$

For an additive Gaussian channel with $\Omega = R$ and

$$\begin{aligned} p(y|0) &= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y-1)^2/2\sigma^2} \\ p(y|1) &= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y+1)^2/2\sigma^2}, \end{aligned}$$

a short calculation using (2.7) gives

$$\gamma_{AGC} = e^{-1/2\sigma^2}.$$

As a final example, for the binary input coherent Rayleigh fading channel with perfect channel state information available to the receiver, we have $\Omega = R \times R^+$, and for $(y,a) \in \Omega$

$$\begin{aligned} p(y,a|0) &= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y-a)^2/2\sigma^2} 2ae^{-a^2} \\ p(y,a|1) &= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y+a)^2/2\sigma^2} 2ae^{-a^2}. \end{aligned}$$

[1]The so-called cutoff rate for this kind of channel is $R_0 = 1 - \log_2(1+\gamma)$, which is positive if and only if the capacity is positive, i.e., $\gamma < 1$.

In this case (2.7) yields

$$\gamma_{RF,CSI} = 1 + \frac{1}{2\sigma^2}.$$

The significance of γ is that γ^h is an upper bound on the maximum-likelihood decoder error probability for a binary code with two codewords separated by a Hamming distance of h (see [39, Theorem 7.5]). It follows that for an (n, k) binary linear code with A_h codewords of weight h, we have the following upper bound, usually called the union bound, on the ML decoder word error probability:

$$\begin{aligned} P_W &\leq \sum_{h=1}^{n} A_h \gamma^h && (2.8) \\ &= \sum_{h=1}^{n} A_h e^{-\alpha h}, && (2.9) \end{aligned}$$

where $\alpha = -\log \gamma \geq 0$ is what we shall call the *noise exponent* for the channel. Similarly, we can use the union bound to estimate the ML decoder *bit* error probability:

$$P_b \leq \sum_{h=1}^{n} \sum_{w=1}^{k} \frac{w}{k} A_{w,h} \gamma^h. \tag{2.10}$$

Since the union bound is linear on weight enumerators, it also applies to ensembles of codes, with A_h replaced by $\overline{A}_h^{(n)}$, the average number of codewords of weight h in $\mathcal{C}_n$:

$$\begin{aligned} \overline{P}_W^{(n)} &\leq \sum_{h=1}^{n} \overline{A}_h^{(n)} e^{-\alpha h} && (2.11) \\ &= \sum_{h=1}^{n} e^{-n(\alpha\delta - r_n(\delta))}, && (2.12) \end{aligned}$$

where in (2.12) $\delta = h/n$. For the ensemble bit error probability we have

$$\overline{P}_b^{(n)} \leq \sum_{h=1}^{n} \sum_{w=1}^{k} \frac{w}{k} \overline{A}_{w,h}^{(n)} e^{-\alpha h}. \tag{2.13}$$

2.3 "Turbo-Like" Code Ensembles

In this section, we consider a general class of concatenated coding systems of the type depicted in Figure 2.1, with q encoders (circles) and $q-1$ interleavers (boxes). The ith code C_i is an (n_i, N_i) linear block code, and the ith encoder is preceded by an interleaver (permuter) P_i of size N_i, except C_1 which is not preceded by an interleaver, but rather is connected directly to the input. The overall structure must have no loops, i.e., it must be a graph-theoretic tree. We call a code of this type a "turbo-like" code.

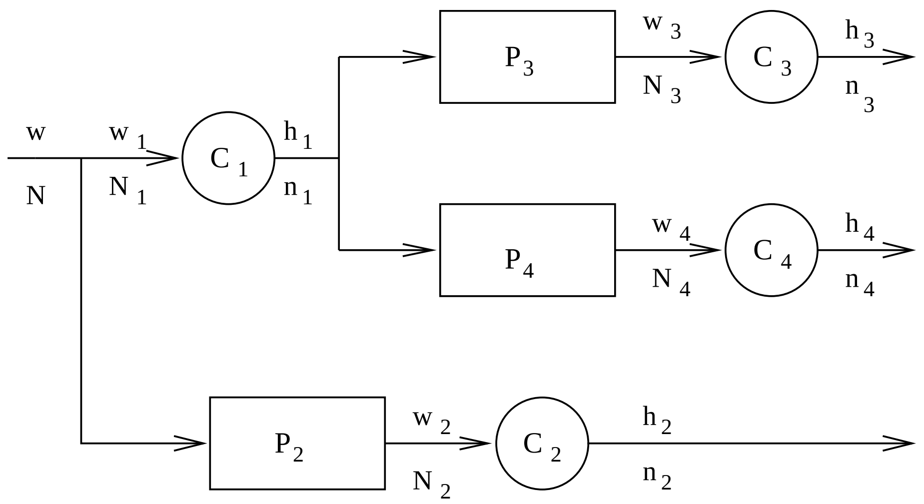

Figure 2.1: A "turbo-like" code with $s_I = \{1, 2\}$, $s_O = \{2, 3, 4\}$, $\overline{s}_O = \{1\}$.

Define $s_q = \{1, 2, \ldots, q\}$ and subsets of s_q by $s_I = \{i \in s_q : C_i \text{ connected to input}\}$, $s_O = \{i \in s_q : C_i \text{ connected to output }\}$, and its complement $\overline{s}_O$. The overall system depicted in Figure 2.1 is then an encoder for an (n, N) block code with $n = \sum_{i \in s_O} n_i$. Because of the difficulty of analyzing specific interleavers P_i, we instead look at the ensemble codes generated by all possible interleavers. This is also known as the *uniform interleaver* technique [6]. (A uniform interleaver is defined as a probabilistic device that maps a given input word of weight w into all distinct $\binom{N_i}{w}$ permutations of it with equal probability $p = 1/\binom{N_i}{w}$.) This generates a code ensemble.

If we know the IOWE $A^{(i)}_{w_i,h_i}$'s for the constituent codes C_i, we can calculate the *ensemble* IOWE $A_{w,h}$ for the overall system (averaged over the set of all possible interleavers) [6]. The result is

$$\overline{A}_{w,h} = \sum_{\substack{h_i : i \in s_O \\ \Sigma h_i = h}} \sum_{h_i : i \in \overline{s}_O} A^{(1)}_{w_1,h_1} \prod_{i=2}^{q} \frac{A^{(i)}_{w_i,h_i}}{\binom{N_i}{w_i}} \tag{2.14}$$

In (2.14) we have $w_i = w$ if $i \in s_I$, and $w_i = h_j$ if C_i is preceeded by C_j (see Figure 2.2). We do not give a proof of formula (2.14), but it is intuitively plausible if we note that the term $A^{(i)}_{w_i,h_i}/\binom{N_i}{w_i}$ is the probability that a random input word to C_i of weight w_i will produce an output word of weight h_i.

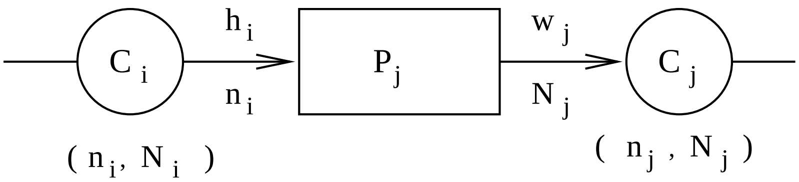

Figure 2.2: C_i (an (n_i, N_i) encoder) is connected to C_j (an (n_j, N_j) encoder) by an interleaver of size N_j. We have the "boundary conditions" $N_j = n_i$ and $w_j = h_i$.

For example, for the $(n_2 + n_3 + n_4, N)$ encoder of Figure 2.1 the formula (2.14) becomes

$$\begin{aligned} \overline{A}_{w,h} &= \sum_{\substack{h_1,h_2,h_3,h_4 \\ (h_2+h_3+h_4=h)}} A^{(1)}_{w_1,h_1} \frac{A^{(2)}_{w_2,h_2}}{\binom{N_2}{w_2}} \frac{A^{(3)}_{w_3,h_3}}{\binom{N_3}{w_3}} \frac{A^{(4)}_{w_4,h_4}}{\binom{N_4}{w_4}} \\ &= \sum_{\substack{h_1,h_2,h_3,h_4 \\ (h_2+h_3+h_4=h)}} A^{(1)}_{w,h_1} \frac{A^{(2)}_{w,h_2}}{\binom{N}{w}} \frac{A^{(3)}_{h_1,h_3}}{\binom{n_1}{h_1}} \frac{A^{(4)}_{h_1,h_4}}{\binom{n_1}{h_1}} \end{aligned}$$

2.4 The Interleaving Gain Exponent Conjecture

In this section we consider systems of the form depicted in Figure 2.1, in which the individual encoders are truncated convolutional encoders, and study the behavior of the average ML decoder error probability as the input block length N approaches infinity. If $\overline{A}^N_{w,h}$ denotes the ensemble IOWE when the input block has length N, we introduce the following notation for the union bound (2.8) for systems of this type:

$$P_W^{\mathrm{UB}} \triangleq \sum_{h=1}^{n} \left(\sum_{w=1}^{N} \overline{A}_{w,h}^N \right) \gamma^h. \tag{2.15}$$

Next we define, for each fixed $w \geq 1$ and $h \geq 1$,

$$\alpha(w,h) = \limsup_{N\to\infty} \log_N \overline{A}_{w,h}^N. \tag{2.16}$$

It follows from this definition that if w and h are fixed,

$$\overline{A}_{w,h}^N \gamma^h = O(N^{\alpha(w,h)+\epsilon}) \qquad \text{as } N \to \infty, \tag{2.17}$$

for any $\epsilon > 0$. Thus if we define

$$\beta_M = \max_{h\geq 1} \max_{w\geq 1} \alpha(w,h) \tag{2.18}$$

it follows that for all w and h,

$$\overline{A}_{w,h}^N \gamma^h = O(N^{\beta_M+\epsilon}) \qquad \text{as } N \to \infty, \tag{2.19}$$

for any $\epsilon > 0$. The parameter β_M, which we shall call the *interleaving gain exponent* (IGE), was first introduced in [6] and [7] for parallel concatenation and later in [8] for serial concatenation. Extensive numerical simulations, and theoretical considerations that are not fully rigorous, lead to the following conjecture about the behavior of the union bound for systems of the type shown in Figure 2.1.

The IGE Conjecture. *There exists a positive number γ_0, which depends on the q component convolutional codes and the tree structure of the overall system, but not on N, such that for any fixed $\gamma < \gamma_0$, as the block length N becomes large,*

$$P_W^{\mathrm{UB}} = O(N^{\beta_M}). \tag{2.20}$$

Eq. (2.20) implies that if $\beta_M < 0$, then for a given $\gamma < \gamma_0$ the *word* error probability of the concatenated code decreases to zero as the input block size is increased. This is summarized by saying that there is *word error probability interleaving gain.* [2]

In [18], the calculation of $\alpha(w,h)$ and β_M for a concatenated system of the type depicted in Figure 2.1 was discussed, using analytical tools introduced in [7] and [8]. For example, for the parallel concatenation of q codes, with $q-1$ interleavers, we have

$$\beta_M \leq -q+2,$$

with equality if and only if each of the component codes is recursive. For a "classical" turbo code with $q=2$, we have $\beta_M = 0$, so there is no word error probability interleaving gain. This suggests that the word error probability for classic turbo codes will not improve with input block size, which is in agreement with simulations.

As another example, consider the serial concatenation of two convolutional codes. If the inner code is recursive, then

$$\beta_M \leq -\left\lfloor \frac{d^{\mathrm{o}}_{\mathrm{free}}+1}{2} \right\rfloor + 1, \tag{2.21}$$

where $d^{\mathrm{o}}_{\mathrm{free}}$ is the minimum distance of the outer code. Therefore, for serial concatenated codes, if $d^{o}_{f} \geq 3$ there is interleaving gain for word error probability. (If the inner code is nonrecursive $\beta_M \geq 0$ and there is no interleaving gain.)

[2]There is a similar conjecture for the bit error probability which we do not discuss here. Suffice it to say that the interleaving gain exponent for bit error probability is $\beta_M - 1$.

Chapter 3 Repeat Accumulate Codes

In this chapter we prove the *Interleaving Gain Exponent* conjecture introduced in Section 2.4 for a simple class of rate $1/q$ serially concatenated codes where the outer code is a q-fold repetition code and the inner code is a rate 1 convolutional code with transfer function $1/(1+D)$. We believe this represents the first rigorous proof of a coding theorem for any class of turbo-like codes. The iterative decoding performance of RA codes is seen to be remarkably good, despite the simplicity of the codes and the suboptimality of the decoding algorithm. Finally, we prove RA codes achieve the ultimate Shannon limit -1.592 dB as the code rate goes to zero on the AWGN channel. For practical interests, a hardware implementation of the decoding algorithm of RA codes is shown in Appendix E.

3.1 Introduction

This chapter is an attempt to illuminate the "near Shannon limit" capabilities of turbo-like codes on the AWGN channel.

Our specific goal is to prove AWGN coding theorems for "turbo-like" codes (Section 2.3). Our proof technique is to derive an explicit expression for the ensemble input-output weight enumerator (IOWE) and then to use this expression, in combination with either the classical union bound, or the "improved" union bound of Viterbi and Viterbi [50], of Divsalar [15], or the recent "typical pairs" bound [2], to show that the maximum likelihood word error probability approaches zero as the block length is increased, provided signal-to-noise ratio exceeds some threshold. In the chapter we demonstrate this technique for some very simple coding systems, which we call *repeat and accumulate* codes. It is satisfying to have rigorously proved coding theorems for even a restricted class of turbo-like codes. In Chapter 5 we prove coding theorems for a much broader class of interleaved concatenated codes.

3.2 RA Code Structure

In this section we introduce a class of turbo-like codes which are simple enough so that we can prove the IGE conjecture. We call these codes *repeat and accumulate* (RA) codes. The general idea is shown in Figure 3.1. An information block of length

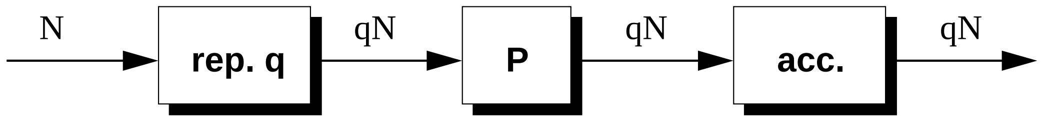

Figure 3.1: Encoder for a (qN, N) RA code. The "rep. q" component repeats its N-bit input block q times; the "P" block represents an arbitrary permutation of its qN-bit input block; and the "acc." is an accumulator, whose outputs are the mod-2 partial sums of its inputs.

N is repeated q times, scrambled by a $qN \times qN$ interleaver, and then encoded by a rate 1 *accumulator*. The accumulator can be viewed as a truncated rate-1 recursive convolutional encoder with transfer function $1/(1+D)$, but we prefer to think of it as a block code whose input block $[x_1, \ldots, x_n]$ and output block $[y_1, \ldots, y_n]$ are related by the formula

$$\begin{aligned}
y_1 &= x_1 \\
y_2 &= x_1 + x_2 \\
y_3 &= x_1 + x_2 + x_3 \\
&\vdots \\
y_n &= x_1 + x_2 + x_3 + \cdots + x_n.
\end{aligned} \tag{3.1}$$

To apply the union bound from Section 2.2 to the class of RA codes, we need the input-output weight enumerators for both the (qn, n) repetition code, and the (n, n) accumulator code. The outer repetition code is easy: if the input block has length n, we have

$$A^{(o)}_{w,h} = \begin{cases} 0 & \text{if } h \neq qw \\ \binom{n}{w} & \text{if } h = qw. \end{cases} \tag{3.2}$$

The inner accumulator code is less trivial, but it is possible to show that (again assuming the input block has length n):

$$A^{(i)}_{w,h} = \binom{n-h}{\lfloor w/2 \rfloor}\binom{h-1}{\lceil w/2 \rceil - 1}. \tag{3.3}$$

(Sketch of the proof: Assume the output sequence $y_1, \ldots, y_n$ of weight h is divided into runs of zeros and ones. Clearly x_i is a 1 if and only if i is at the conjunction of runs, hence $w = 2k$ or $w = 2k+1$ if k is the number of runs of ones. So $A_{w,h}$ is the number of sequences which have hamming weight h and have k runs of ones, where $k = \lfloor w/2 \rfloor$.)

It follows then from the general formula (2.14), that for the (qN, N) RA code represented by Figure 3.1, the ensemble IOWE is

$$\begin{aligned} \overline{A}^{(N)}_{w,h} &= \sum_{h_1=0}^{qN} \frac{A^{(o)}_{w,h_1} A^{(i)}_{h_1,h}}{\binom{qN}{qw}} \\ &= \frac{\binom{N}{w}\binom{qN-h}{\lfloor qw/2 \rfloor}\binom{h-1}{\lceil qw/2 \rceil - 1}}{\binom{qN}{qw}}. \end{aligned} \tag{3.4}$$

From (3.4) it is easy to compute the parameters $\alpha(w,h)$ and β_M in (2.16) and (2.18). The result is

$$\alpha(w,h) = -\left\lceil \frac{(q-2)w}{2} \right\rceil \tag{3.5}$$

$$\beta_M = -\left\lceil \frac{(q-2)}{2} \right\rceil. \tag{3.6}$$

It follows from (3.6) and the IGE conjecture (Section 2.4) that an RA code can have word error probability interleaving gain only if $q \geq 3$.

We are now prepared to use the union bound to prove the IGE conjecture for RA codes. In order to simplify the exposition as much as possible, we will assume for the rest of this section that $q = 4$, the extension to arbitrary $q \geq 3$ being straightforward but rather lengthy. For $q = 4$, (3.6) becomes $\beta_M = -1$, so the IGE conjecture is $P_W^{\mathrm{UB}} = O(N^{-1})$ for $E_b/N_0 > \gamma_0$ in this case.

The union bound (2.8) for the ensemble of $q = 4$ RA codes is, because of (3.4),

$$P_W^{\mathrm{UB}} = \sum_{h=2}^{4N} \sum_{w=1}^{h/2} \frac{\binom{N}{w}\binom{4N-h}{2w}\binom{h-1}{2w-1}}{\binom{4N}{4w}} \gamma^h. \tag{3.7}$$

Denote the (w, h)th term in the sum (3.7) by $T_N(w, h)$:

$$T_N(w, h) \triangleq A_{w,h} z^h = \frac{\binom{N}{w}\binom{4N-h}{2w}\binom{h-1}{2w-1}}{\binom{4N}{4w}} \gamma^h.$$

Using standard techniques (e.g., [41, Appendix A]), it is possible to show that for all (w, h),

$$T_N(w, h) \leq D 2^{h[F(x,y) + \log_2 \gamma]}, \tag{3.8}$$

where $D = 4/\sqrt{\pi}$ is a constant, $x = w/4N$, $y = h/4N$,

$$F(x, y) = \frac{-\frac{3}{4} H_2(4x) + (1-y) H_2(\frac{2x}{1-y}) + y H_2(\frac{2x}{y})}{y},$$

and $H_2(x) = -x \log_2(x) - (1-x)\log_2(1-x)$ is the binary entropy function. The maximum of the function $F(x, y)$ in the range $0 \leq 2x \leq y \leq 1 - 2x$ occurs at $(x, y) = (0.100, 0.371)$ and is 0.562281, so that if $\log_2 \gamma < -0.562281$, the exponent in (3.8) will be negative.

Let us therefore assume that $\log_2 \gamma < -0.562281$, which is equivalent to $E_b/N_0 = -(1/r)\ln\gamma = -4\ln\gamma \geq 4 \cdot \ln 2 \cdot 0.562281 = 1.559 = 1.928$ dB. If E is defined to be $E = -\log_2 \gamma + 0.562281$, it follows from (3.8) for all w and h,

$$T_N(w, h) \leq D 2^{-hE}. \tag{3.9}$$

What (3.9) tells is is that if $E_b/N_0 > 1.928$ dB, most of the terms in the union bound (3.7) will tend to zero rapidly, as $N \to \infty$. The next step in the proof is to break the sum in (3.7) into two parts, corresponding to those terms for which (3.9) is helpful, and those for which it is not. To this end, define

$$h_N \triangleq (\log_2 N)^2,$$

and write

$$\begin{aligned} P_W^{\mathrm{UB}} &= \sum_{h=2}^{4N} \sum_{w=1}^{h/2} T_N(w,h) \\ &= \sum_{h=2}^{h_N} \sum_{w=1}^{h/2} T_N(w,h) + \sum_{h=h_N+1}^{4N} \sum_{w=1}^{h/2} T_N(w,h) \\ &= S_1 + S_2. \end{aligned}$$

It's easy to verify that when N is large enough, $\overline{A}_{w+1,h}/\overline{A}_{w,h} < 1$ for $h \le h_N$ and $w \le h/2 \le h_N/2$, which shows $\overline{A}_{w,h}$ is a decreasing function of w for large N. Thus the sum S_1 can be overbounded as follows (we omit some details):

$$\begin{aligned} S_1 &= \sum_{h=2}^{h_N} \sum_{w=1}^{h/2} T_N(w,h) \\ &= \sum_{h=2}^{h_N} T_N(1,h) + \sum_{h=2}^{h_N} \sum_{w=2}^{h/2} T_N(w,h) \\ &= O(N^{-1}) + \sum_{h=2}^{h_N} \sum_{w=2}^{h/2} T_N(w,h) \\ &\le O(N^{-1}) + \sum_{h=2}^{h_N} \sum_{w=2}^{h/2} A_{2,h}\gamma^h \\ &= O(N^{-1}) + \sum_{h=2}^{h_N} \sum_{w=2}^{h/2} O(h^3/N^2)\gamma^h \\ &= O(N^{-1}) + O(h_N^5/N^2) \\ &= O(N^{-1}). \end{aligned}$$

For the sum S_2, we bound each term $T_N(w,h)$ by (3.9):

$$
\begin{aligned}
S_2 &= \sum_{h=h_N+1}^{4N} \sum_{w=1}^{h/2} T_N(w,h) \\
&\leq \sum_{h_N+1}^{4N} \sum_{w=1}^{h/2} D2^{-hE} \\
&= D/2 \sum_{h_N+1}^{4N} h2^{-hE} \\
&\leq D\frac{2^{-Eh_N}(h_N+1)}{(1-2^{-E})^2} \\
&= O(2^{-E(\log_2 N)^2}(\log_2 N)^2) \\
&= o(N^{-2}).
\end{aligned}
$$

We have therefore shown that for the ensemble of $q = 4$ RA codes, if $E_b/N_0 >$ 1.928 dB,

$$P_W^{\mathrm{UB}} = S_1 + S_2 = O(N^{-1}) + o(N^{-1}) = O(N^{-1}), \tag{3.10}$$

which as we saw above, is the IGE conjecture in this case.

Although the union bound is adequate to prove the IGE conjecture for RA codes, the threshold value γ_0 derived is by no means the best possible. A tighter bound generally yields smaller threshold values. For example, Viterbi-Viterbi bound [50] and Divsalar bound [15] can be applied to improve the ensemble thresholds. That is discussed in Appendix A. In particular, for RA codes, we have the following theorem.

Theorem 3.1 *For an RA code ensemble with spectral shape $r(\delta)$, we define*

$$c_o \triangleq \sup_{0<\delta<1} \frac{1-\delta}{\delta}\frac{1-e^{-2r(\delta)}}{2},$$

if $E_b/N_o > 1/Rc_o$, the average maximum-likelihood word error probability for the ensemble code approaches 0 *as the block length is increased.*

Using this theorem, we can lower the value of γ_0 considerably, e.g., for $q = 4$ from

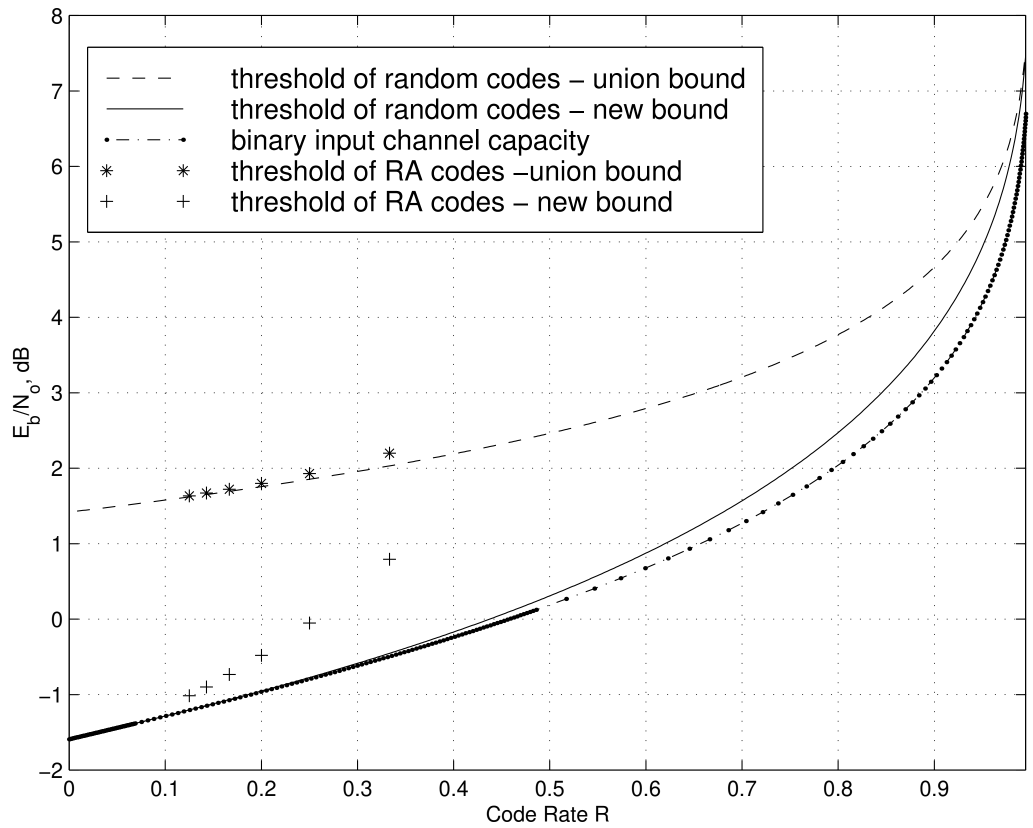

Figure 3.2: Comparing the RA code "cutoff threshold" to the cutoff rate of random codes using both the classical union bound and the Divsalar bound, along with binary-input channel capacity.

1.928 dB to −0.052 dB. In Figure 3.2 and Table 3.1 we display our numerical results on RA codes. There we compare the "cutoff threshold" γ_0 for RA codes with q in the range $3 \leq q \leq 8$ using both the classical union bound and the Divsalar improved union bound to the cutoff threshold for the ensemble of all codes (i.e., "random codes") of a fixed rate. These values of γ_0 can be further reduced, by "typical pairs" decoding method, and that will be discussed in Chapter 6.

3.3 Iterative Decoding of RA Codes

The results of the previous section show that the performance of RA codes *with maximum-likelihood decoding* is very good. However, the complexity of ML decoding

Rate	1/3	1/4	1/5	1/6	1/7	1/8
RA Codes (UB)	2.200	1.928	1.798	1.721	1.670	1.631
Random Codes(UB)	2.031	1.853	1.775	1.694	1.651	1.620
RA codes (DB)	0.792	-0.052	-0.480	-0.734	-0.900	-1.015
Random Codes(DB)	-0.453	-0.774	-0.953	-1.066	-1.146	-1.203
Binary Shannon Limit	-0.495	-0.794	-0.963	-1.073	-1.150	-1.210

Table 3.1: Numerical data gleaned from Figure 3.2.

of RA codes, like that of all turbo-like codes, is prohibitively large. But an important feature of turbo-like codes is the availability of a simple iterative, message passing decoding algorithm that closely approximates ML decoding. In this section, we discuss the iterative decoding of RA codes.

3.3.1 Tanner graph representation of RA codes

We start with a consideration of a Tanner graph realization of RA codes. A Tanner graph [48, 52] $G = (V, E)$ is a bipartite graph whose vertices can be partitioned into variable nodes V_m and check nodes V_c, with edges $E \subseteq V_m \times V_c$. In a Tanner graph, check nodes represent certain “local constraints” (or generalized “equation system”) on subsets of variable nodes; an edge indicates that a particular variable is present in a particular constraint.

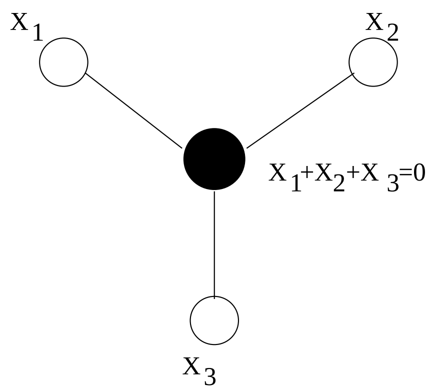

Figure 3.3: Tanner graph of a $(3, 2)$ parity check code.

Figure 3.3 shows a Tanner graph of a $(3, 2)$ code C. The definition of this code C is a vector (x_1, x_2, x_3) belongs to C if and only if $x_1+x_2+x_3 = 0$. In the Tanner graph, the empty circles represent three variables x_1, x_2, x_3; and the filled circle represents the constraint over those variables, namely, their mod-2 sum should be zero.

The Tanner graph representation of RA codes is quite simple. For a repetition q RA code with block length n, let's denote the n information bits by u_i $_{(i\in[n])}$, the qn code bits by y_i $_{(i\in[qn])}$, and the intermediate bits (which are the outputs of outer code and inputs to the inner code) by x_i $_{(i\in[qn])}$. We know y_i and x_i are related by the formula

$$y_i = \begin{cases} x_1 & \text{if } i = 1, \\ x_i + y_{i-1} & \text{otherwise.} \end{cases} \tag{3.11}$$

Notice every x_i is a replica of some u_j (where the mapping $\phi : i \to j$ is completely determined by the permutation $\pi \in s_{qn}$), so if we represent all qn equations in (3.11) by check nodes c_i $_{(i\in[qn])}$ and represent both information bits u_i and code bits y_i by variable nodes, the edges can be naturally generated by connecting each check node to the u_is and y_is present in its equation. Using the notation $C = \{c_i, i \in [qn]\}$, $U = \{u_i, i \in [n]\}$, $Y = \{y_i, i \in [qn]\}$, we have a Tanner Graph representation of RA codes, with $V_m = U \bigcup Y$ and $V_c = C$.

Figure 3.4 shows a Tanner Graph for a repetition 3 block length 2 RA code, with permutation $\pi = (1, 2, 5, 3, 4, 6)$. In the graph, we have also included the received version of code bits y, which are denoted by y_r. The received bits y_r provide evidence in the decoding procedure, but they are not part of the Tanner Graph.

Generally, in a Tanner Graph for a repetition q RA code, regardless of the block length n, every u_i is present in q check nodes. Hence every vertex $u \in U$ has degree q. Similarly, every vertex $c \in C$ has degree 3 (except the first vertex c_1, $d(c_1) = 2$), and every vertex $y \in Y$ has degree 2 (except the last vertex y_{qn}, $d(y_{qn}) = 1$). Those facts are illustrated in Figure 3.4.

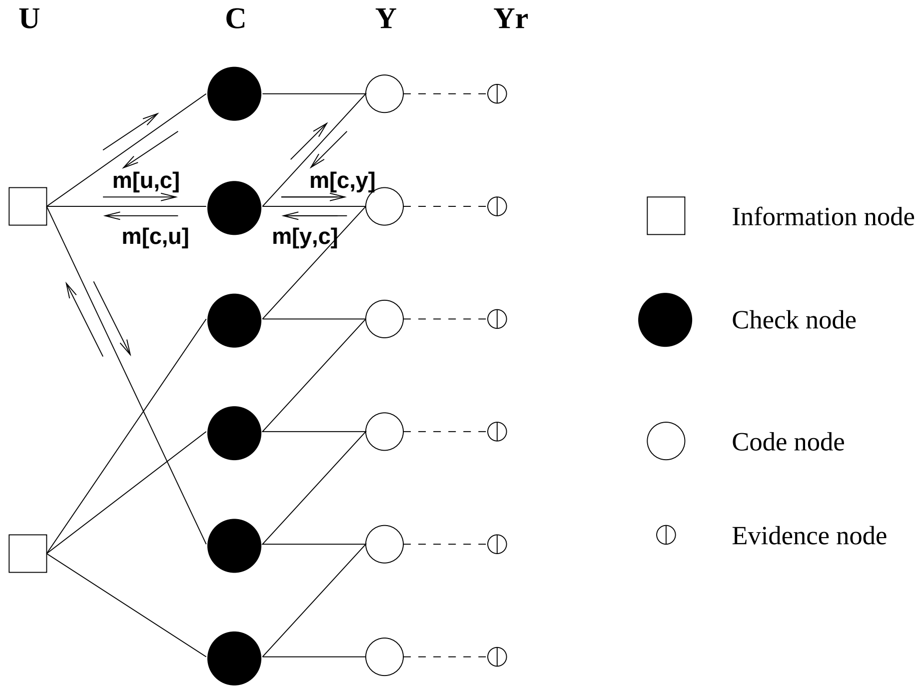

Figure 3.4: Tanner Graph of a repetition 3, length 2 RA code, with permutation $\pi = (1, 2, 5, 3, 4, 6)$. Each information node is present in three check nodes; each check node checks the parity sum of two adjacent code nodes and one information node.

3.3.2 Message passing on Tanner graph realization

Using the Tanner graph realization, message passing algorithms (belief propagation) can be applied to decode RA codes. This algorithm is an instance of the GDL algorithm [3] with a specific scheduling. In a belief propagation decoding algorithm, the messages passed on an edge e represent posterior densities on the bit associated with the variable node. A probability density on a bit is pair of non-negative reals p_0, p_1 satisfying $p_0 + p_1 = 1$. Such a pair can be represented by its log likelihood ratio $\log \frac{p_1}{p_0}$, and we shall assume that the messages here use this representation.

There are four distinct classes of messages in our belief propagation decoding algorithm of RA codes, namely messages sent (received) by some vertex $u \in U$ to (from) some vertex $c \in C$, which are denoted by $m[u, c]$ ($m[c, u]$), and messages sent (received) by some vertex $y \in Y$ to (from) some vertex $c \in C$, which are denoted by $m[y,c]$ ($m[c,y]$). Messages are passed along the edges in the Tanner graph, as shown in Figure 3.4. Both $m[u,c]$ and $m[c,u]$ have the the conditional value of $\log \frac{p(u=1)}{p(u=0)}$, and both $m[y,c]$ and $m[c,y]$ have the conditional value of $\log \frac{p(y=1)}{p(y=0)}$. Each code node y also has the belief provided by received bit y_r, which is denoted by $B(y) = \log \frac{p(y=1|y_r)}{p(y=0|y_r)}$. With this notation, the belief propagation decoding algorithm of RA codes can be described as follows: (A hardware implementation of this algorithm is shown in Appendix E.)

Initialize all messages $m[u,c], m[c,u], m[y,c], m[c,y]$ to be zero. The messages are continually updated in K rounds (the number K is pre-fixed or is determined dynamically by some halting rule). Each round consists of a sequential execution of the following script:

- ***Update* $\mathbf{m[y,c]}$:**

$$m[y,c] = \begin{cases} B(y) & \textit{if } y = y_{qn}, \\ B(y) + m[c',y] & \textit{otherwise, where } (c',y) \in E \textit{ and } c' \neq c. \end{cases}$$

- ***Update* $\mathbf{m[u,c]}$:**

$$m[u,c] = \sum_{c'} m[u,c'], \ \textit{ where } (u,c') \in E \textit{ and } c' \neq c.$$

- ***Update at check nodes,* $\mathbf{m[c,y]}$ *: and* $\mathbf{m[c,u]}$:**

$$m[c,y] = \begin{cases} m[u,c] & \textit{if } c = c_1, \textit{ where } (u,c) \in E \textit{ and } u \in U, \\ \frac{e^{m[u,c]} + e^{m[y',c]}}{1 + e^{m[u,c] + m[y',c]}} & \textit{otherwise, where } (u,c), (y',c) \in E \textit{ and } y \neq y' \in Y. \end{cases}$$

$$m[c,u] = \begin{cases} m[y,c] & \textit{if } c = c_1, \textit{ where } (y,c) \in E \textit{ and } y \in Y, \\ \frac{e^{m[y,c]} + e^{m[y',c]}}{1 + e^{m[y,c] + m[y',c]}} & \textit{otherwise, where } (y,c), (y',c) \in E \textit{ and } y \neq y' \in Y. \end{cases}$$

After K iterations, we calculate $s(u) = \sum_c m[u, c]$ for every $u \in U$, where the summation is over all the c such that $(u, c) \in E$. If $s(u) \geq 0$, bit u is decoded to be 1*; otherwise, it is decoded to be* 0.

It turns out that the iterative decoding capacity of RA codes can be obtained by the use of density evolution [44]. By "capacity" we mean if the noise is below that, iterative decoding of RA codes can achieve arbitrarily small error probability for long enough block length; while if the noise is above that, iterative decoding always has non-zero error probability. Those numbers were first reported by Richardson and Urbanke (refer to Table 6.4). For example, for $q = 3$ RA code, the iterative decoding capacity is 0.479 dB; for $q = 4$ RA code, it is 0.106 dB.

3.3.3 Performance of RA codes with iterative decoding

We wrote a computer program to implement this "turbo-like" decoding for RA codes with $q = 3$ (rate 1/3) and $q = 4$ (rate 1/4), and the results are shown in Figure 3.5. We simulated codes of information block length 4096 and 16384. Those performance curves are for word error rate (WER) instead of bit error rate (BER). We see in Figure 3.5, for example, that the empirical cutoff threshold for RA codes for $q = 3$ appears to be less than 1 dB, which confirms its iterative decoding capacity 0.479 dB.

3.4 RA Codes Achieve Channel Capacity

For the theorist, the most important thing about RA codes is that their combinatorial properties are reasonably well understood. For the practitioner, the most important thing is the experimentally verified fact that if an iterative decoding algorithm derived from belief propagation on the appropriate Tanner graph is applied to them, their AWGN channel performance is scarcely inferior to that of full-fledged turbo codes. In this section, we will give a partial explanation of the latter property, by using the former property to show that on the AWGN channel, RA codes have the *potential* for achieving channel capacity. That is, as the rate of the RA code approaches zero,

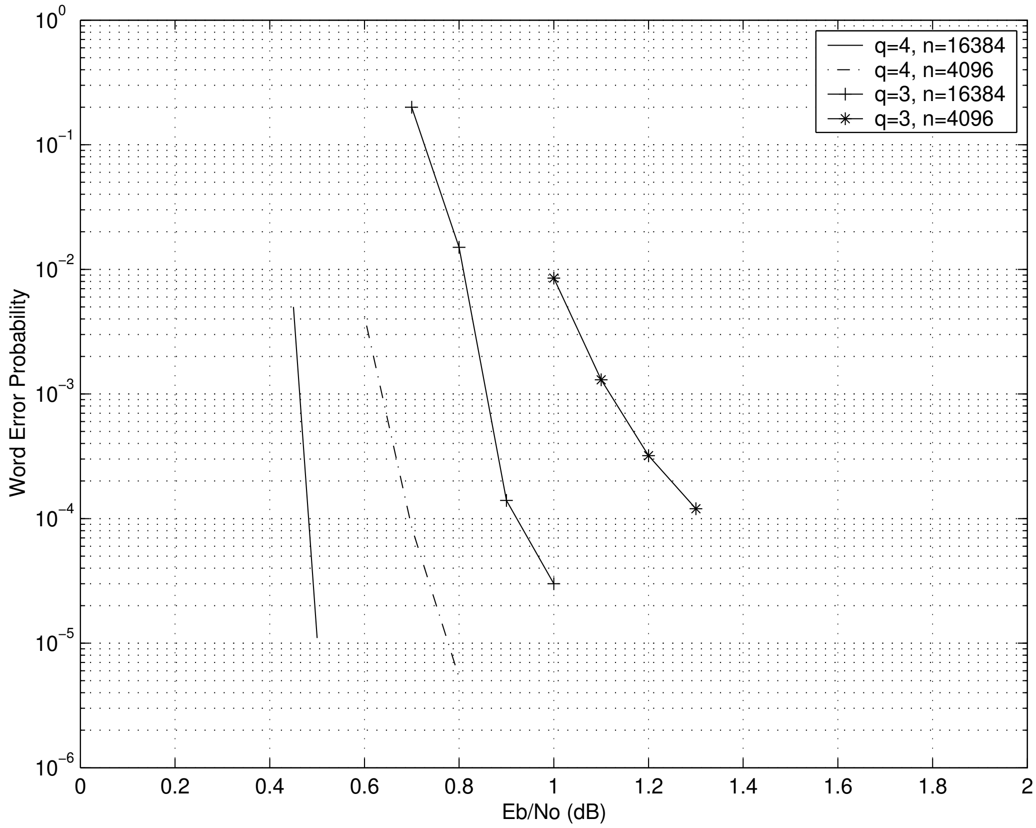

Figure 3.5: Simulated performance of iterative decoding of RA codes on an AWGN channel.

the average required bit E_b/N_0 for arbitrarily small error probability with maximum-likelihood decoding approaches $\log 2$, which is the Shannon limit.

Consider the ensemble of rate $R = 1/q$ RA codes. It can be shown that the spectral shape for this ensemble of codes is given by the formula

$$r_q(\delta) = \max_{0 \le u \le \min(2\delta, 2-2\delta)} \left\{ f(u,\delta) + \frac{H(u)}{q} \right\}, \tag{3.12}$$

where

$$f(u,\delta) \stackrel{\Delta}{=} -H(u) + (1-\delta)H(\frac{u}{2(1-\delta)}) + \delta H(\frac{u}{2\delta}), \tag{3.13}$$

and $H(u) = -(u \log u + (1-u)\log(1-u))$ is the (natural) entropy function. Figure 3.6

shows the function $r_q(\delta)$ for $q = 4$.

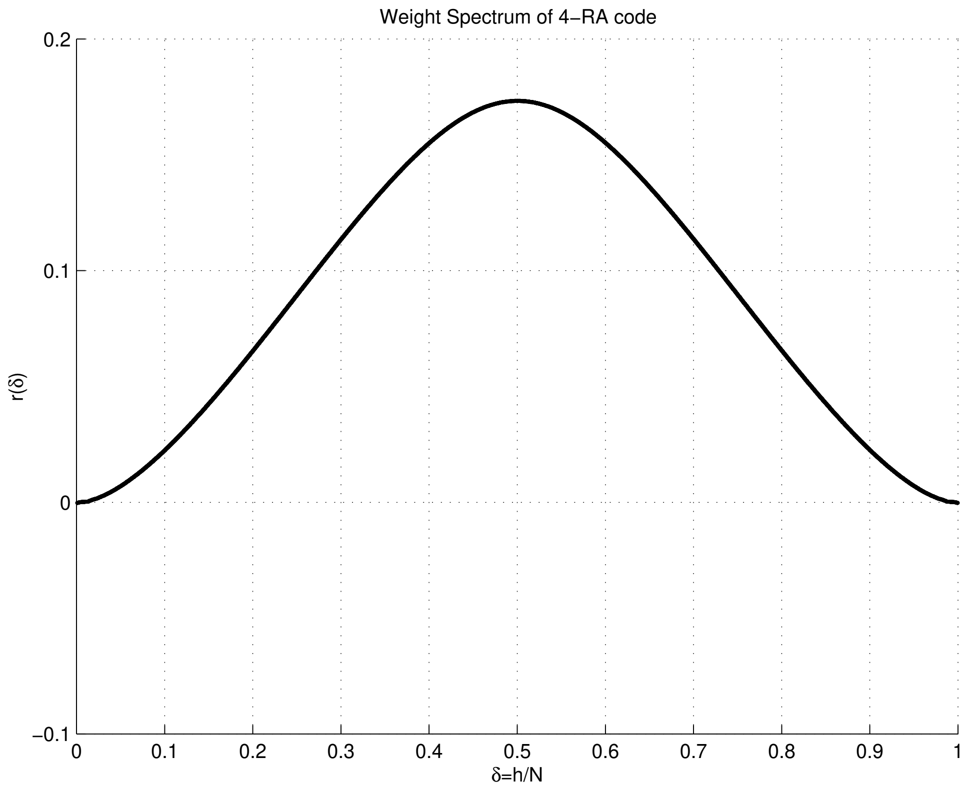

Figure 3.6: The function of $r_4(\delta)$.

Theorem 3.1 says that if we define, for each integer $q \geq 2$,

$$c_o(q) \triangleq \sup_{0<\delta<1} \frac{1-\delta}{\delta} \frac{1-e^{-2r_q(\delta)}}{2} \tag{3.14}$$

and

$$\gamma_q \triangleq q c_0(q), \tag{3.15}$$

then if $q \geq 3$, and $E_b/N_0 > \gamma_q$, as $n \to \infty$, the average maximum-likelihood word error probability for the ensemble of rate $1/q$ RA codes approaches 0. A short table of these thresholds, together with the corresponding AWGN Shannon limit, is given below.

q	R	γ_q (dB)	Shannon (dB)
2	1/2	3.384	0.184
3	1/3	0.792	-0.495
4	1/4	-0.052	-0.794
5	1/5	-0.480	-0.963
6	1/6	-0.734	-1.071
7	1/7	-0.900	-1.15
8	1/8	-1.015	-1.210
$\vdots$	$\vdots$	$\vdots$	$\vdots$
∞	0	(-1.592)	-1.592

For example, the $q = 3$ line of this table tells us that if $E_b/N_0 > 0.792$ dB, then as $n \to \infty$, the word error probability for the ensemble of rate 1/3 RA codes approaches zero.[1] On the other hand, for $E_b/N_0 < -0.495$ dB, (the Shannon limit for binary codes of rate 1/3) as $n \to \infty$, the word error probability for any sequence of codes of rate 1/3 must approach 1.

In the table, and more clearly in Figure 3.7, we see that as the rate R approaches zero, the Shannon limit approaches $\log 2 = -1.592$ dB. This is of course well known, and in fact the value -1.592 dB is usually referred to as the Shannon limit for the AWGN channel. The interesting thing for us, however, is that the RA code thresholds also seem to approach -1.592 dB. This empirical observation is in fact true, and it is the object of this section to prove the following theorem.

Theorem 3.2 *We have* $\lim_{q\to\infty} \gamma_q = \log 2$*, i.e., RA codes achieve the Shannon limit for the AWGN channel.*

We will give a proof of Theorem 3.2 in Section 3.4.1. But here we note that it is easy to prove that the limit cannot be smaller than $\log 2$:[2]

[1] We have included the threshold for $q = 2$ in the table because it can be shown that for rate 1/2 RA codes, if $E_b/N_0 > \gamma_2$, the ensemble *bit* error probability approaches zero, although the word error probability does not.

[2] Of course, Theorem 3.3 follows from the fact that log 2 is the Shannon limit for the channel, but it is interesting to have an "elementary" proof.

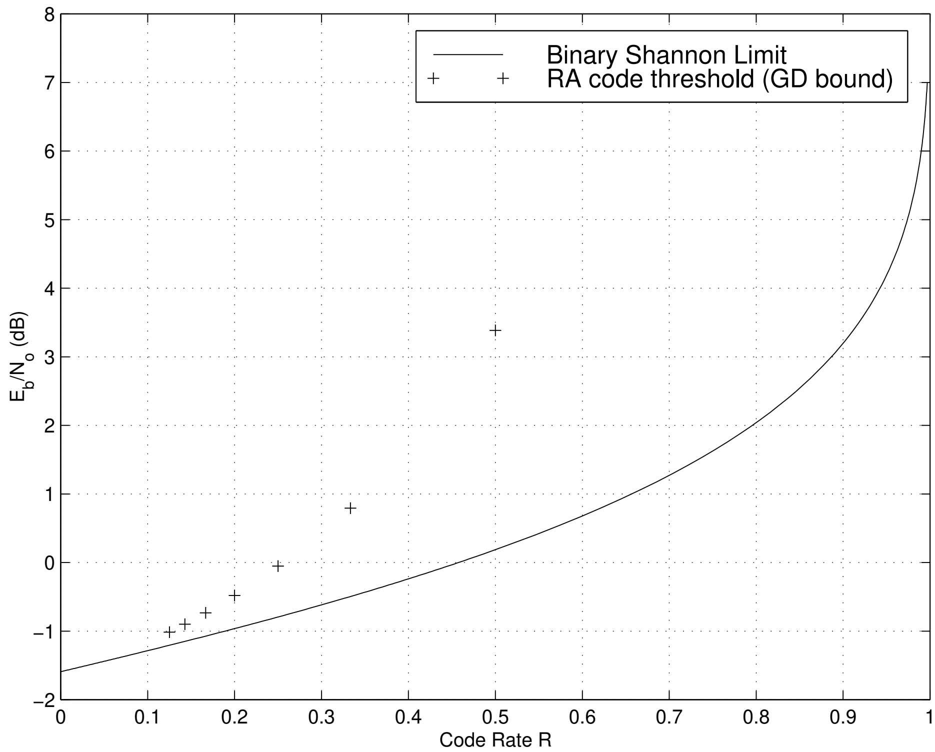

Figure 3.7: The RA thresholds.

Theorem 3.3 *With the threshold* γ_q *defined as above,* $\liminf_{q\to\infty} \gamma_q \geq \log 2$.

Proof: First note that in (3.13), we have $f(u, 1/2) = 0$, so that

$$r_q(1/2) = \max_{0 \leq u \leq 1} H(u)/q = H(1/2)/q = \frac{\log 2}{q}.$$

Thus by taking $\delta = 1/2$ in the definition (3.14) of $c_0(q)$, we have the lower bound

$$\begin{aligned} c_0(q) &\geq \frac{1-(1/2)}{(1/2)} \frac{1-e^{-2r_q(1/2)}}{2} \\ &= \frac{1-e^{-2\log 2/q}}{2}. \end{aligned}$$

It therefore follows that

$$\gamma_q = qc_0(q) \geq \frac{1-e^{-(2/q)\log 2}}{(2/q)}. \tag{3.16}$$

The desired result now follows by observing that the limit, as $q \to \infty$, of the right side of (3.16) is $\log 2$. ■

3.4.1 Proof of main theorem

In this section, we present the proof of Theorem 3.2. We begin with some preliminary technical results which concern the spectral shape functions $r_q(\delta)$.

Lemma 3.1 *The function $f(u,\delta)$ defined in (3.13) is nonpositive, i.e., for any $(u,\delta) \in [0,1] \times [0,1]$, $f(u,\delta) \le 0$. Furthermore, $f(u,\delta) = 0$ if and only if $\delta = 1/2$ or $u = 0$.*

Proof: Jensen's inequality, and the fact that $H(u)$ is strictly convex $\cap$, implies that for any $\delta \in [0,1]$, and any two numbers $u_1, u_2 \in [0,1]$, we have

$$H(\delta u_1 + (1-\delta)u_2) \ge \delta H(u_1) + (1-\delta)H(u_2),$$

with equality if and only if $u_1 = u_2$ and/or $\delta = 0$ or 1. Letting $u_1 = u/(2\delta)$ and $u_2 = u/(2(1-\delta))$, we obtain the desired result. ■

Corollary 3.1 *We have, for each $q \ge 2$ and $\delta \in [0,1]$,*

$$r_q(\delta) \le \frac{\log 2}{q},$$

with equality if and only if $\delta = 1/2$.

Proof: Let $u(q,\delta)$ denote the optimizing value of u in the computation of $r_q(\delta)$, i.e.,

$$u(q,\delta) = \arg \max_{0 \le u \le \min(2\delta, 2-2\delta)} \left\{ f(u,q) + \frac{H(u)}{q} \right\}. \tag{3.17}$$

Then by the definition of $r_q(\delta)$, we have

$$r_q(\delta) = f(u(q,\delta),\delta) + H(u(q,\delta))/q.$$

But since by Lemma 3.1, $f(u,\delta) \le 0$, we have

$$r_q(\delta) \le H(u(q,\delta))/q \le \frac{\log 2}{q}. \tag{3.18}$$

For equality to hold in (3.18), we must have, for $u = u(q,\delta)$, both $f(u,\delta) = 0$ and $u = 1/2$. But by Lemma 3.1, if $f(u,\delta) = 0$, then either $u = 0$ or $\delta = 1/2$. Thus equality can hold only when $\delta = 1/2$, as asserted. ■

Lemma 3.2 *If $u(q,\delta)$ is defined as in (3.17), then*

$$u(q,\delta) \cdot (1 - u(q,\delta))^{q-1} \le \left(2\sqrt{\delta(1-\delta)}\right)^q. \tag{3.19}$$

Proof: It is easy to check that the "max" in (3.12) does not occur at either endpoint (at $u = 0$ the function is zero and its derivative is infinite, and at $u = \min(2\delta, 2(1-\delta))$, the function is negative), so at the point $u = u(q,\delta)$, the partial derivative of the function in braces on the right side of (3.12), with respect to u, must be zero. This condition works out to be

$$-(1-1/q)\log(1-u) - (1/q)\log u + (1/2)\log(2-2\delta-u) + (1/2)\log(2\delta-u) = 0,$$

or in exponential form,

$$\frac{\sqrt{(2-2\delta-u)(2\delta-u)}}{(1-u)^{1-1/q}u^{1/q}} = 1. \tag{3.20}$$

But since $0 \le u \le \min(2\delta, 2(1-\delta))$, the numerator on the left side of (3.20) $\sqrt{(2-2\delta-u)(2\delta-u)}$ is $\le 2\sqrt{\delta(1-\delta)}$. Therefore, at the maximizing point $u = u(q,\delta)$, we must have

$$\frac{2\sqrt{\delta(1-\delta)}}{(1-u)^{1-1/q}u^{1/q}} \ge 1. \tag{3.21}$$

Rearranging this, we get (3.19). ■

Corollary 3.2 *If* (δ_q) *is a sequence of real numbers in the range* $[0,1]$, *and* $\lim_{q\to\infty}\delta_q = \delta^* \neq 1/2$, *then* $\lim_{q\to\infty} u(q,\delta_q) = 0$.

Proof: If $\delta^* \neq 1/2$, then $2\sqrt{\delta^*(1-\delta^*)} < 1$, and so the right-hand side of (3.19) approaches zero as $q \to \infty$. Thus by (3.19), $u(q,\delta)$ must approach either zero or one. But since $u \leq \min(2\delta^*, 2-2\delta^*) < 1$, the only possibility is $u(q,\delta_q) \to 0$. ■

Corollary 3.3 *There exists a* $q_0 \geq 2$ *and* $\delta_0 > 0$ *such that if* $q \geq q_0$ *and* $\delta \leq \delta_0$, *then* $u(q,\delta) < \delta^2$.

Proof: Let $\delta < 1/2$. Then $u < \min(2\delta, 2-2\delta) = 2\delta$, so that the left side of (3.19) is lower bounded by the quantity $u(1-2\delta)^q$. Thus from (3.19), we obtain

$$u(q,\delta) \leq \left(\frac{2\sqrt{\delta(1-\delta)}}{1-2\delta}\right)^q. \tag{3.22}$$

If the quantity in parentheses in (3.22) is less than one, which is true for $\delta < 0.14$, then the right-hand side is decreasing in q. For example, if $q \geq 6$, we have

$$u(q,\delta) \leq \left(\frac{2\sqrt{\delta(1-\delta)}}{1-2\delta}\right)^6, \tag{3.23}$$

which, being of order δ^3, will plainly be $\leq \delta^2$ for small enough δ. ■

In the definition of $c_0(q)$ in (3.14), let $\delta(q)$ be the optimizing value of δ, i.e.,

$$\delta(q) \triangleq \arg\sup_{0<\delta<1} \frac{1-\delta}{\delta}\frac{1-e^{-2r_q(\delta)}}{2}. \tag{3.24}$$

The following proposition is the key to the proof of Theorem 3.2.

Property 3.1 *The sequence* $\delta(q)$ *approaches* $1/2$ *as* q *goes to infinity.*

Proof: We will show that the set $\{\delta(q)\}$ can have no accumulation points in $[0,1]$ except $1/2$. We begin by proving an inequality, eq. (3.26), below. By definition,

$$\begin{aligned}\gamma_q &= qc_0(q) \\ &= q\frac{1-\delta(q)}{\delta(q)}\frac{1-e^{-2r_q(\delta(q))}}{2}.\end{aligned}$$

But since we have the elementary inequality

$$(1 - e^{-2r})/2 \leq r, \tag{3.25}$$

it follows that

$$\gamma_q \leq \frac{1-\delta(q)}{\delta(q)} q r_q(\delta(q)).$$

But as noted in (3.18), $q r_q(\delta) \leq H(u(q,\delta))$, so that

$$\gamma_q \leq \frac{1-\delta(q)}{\delta(q)} \cdot H(u(q,\delta(q))). \tag{3.26}$$

Now we assume that there is a subsequence of q's for which the $\delta(q)$'s approach a limit $\delta^* \neq 1/2$. There are two cases to consider, $\delta^* \neq 0$, and $\delta^* = 0$. In both cases, we have from Corollary 3.2 that $u(q,\delta(q)) \to 0$. Thus if $\delta^* \neq 0$, it follows from (3.26) that with q restricted to the given subsequence,

$$\lim_{q\to\infty} \gamma_q \leq \frac{1-\delta^*}{\delta^*} H(0) = 0,$$

which contradicts Theorem 3.3.

On the other hand, if $\delta^* = 0$, we have from Corollary 3.3 that for q large enough $u(q,\delta(q)) < \delta(q)^2$. Thus from (3.26) again, for large enough q, we have

$$\gamma_q \leq (1-\delta(q)) \frac{H(\delta(q)^2)}{\delta(q)}.$$

But since $H(x^2)/x \to 0$ as $x \to 0$, with q restricted to the given subsequence,

$$\lim_{q\to\infty} \gamma_q \leq \lim_{q\to\infty} \frac{H(\delta(q)^2)}{\delta(q)} = 0,$$

again contradicting Theorem 3.3.

We have therefore shown that the only possible accumulation point of the set $\{\delta(q)\}$ is 1/2, which proves that the limit of the $\delta(q)$'s exists and equals 1/2. $\blacksquare$

We can now prove the main theorem:

Proof of Theorem 3.2. We have, for every q,

$$\gamma_q = qc_0(q) = q\frac{1-\delta(q)}{\delta(q)}\frac{1-e^{-2r_q(\delta(q))}}{2}.$$

Applying the elementary inequality (3.25), we obtain

$$\gamma_q \leq q\frac{1-\delta(q)}{\delta(q)}r_q(\delta(q)).$$

Now from Corollary 3.1, we know that $r_q(\delta) \leq (\log 2)/q$, so that we have

$$\gamma_q \leq \frac{1-\delta(q)}{\delta(q)}\log 2.$$

But by Proposition 3.1, $\lim_{q\to\infty}\delta(q) = 1/2$, so

$$\limsup_{q\to\infty}\gamma_q \leq \log 2.$$

But we saw in Theorem 3.3 that $\liminf_q \gamma_q \geq \log 2$. Therefore

$$\lim_{q\to\infty}\gamma_q = \log 2.$$

$\blacksquare$

Chapter 4 Beyond RA Codes

4.1 Introduction

In Chapter 3 we analyzed the class of repeat-accumulate codes. The insights gleaned from that study are twofold. First, the magic of pseudorandom interleaver. Although neither the repeat nor the accumulate code has coding gain, the repeat-accumulate code with interleaver has coding capability (maximum likelihood decoding) close to Shannon's theoretical limit. Second, the remarkable iterative decoding performance despite the simplicity of the code structure. One reason for this surprise is that the underlying Tanner graph representation is very sparse so that the belief propagation algorithm is a good approximation to maximum likelihood decoding.

In this chapter we introduce several simple variations of RA codes. The hope is that a small change in the structure, with some sacrifice in complexity, would produce even stronger codes (at least in the case of ML decoding).[1] We concentrate on codes whose weight enumerators can be derived, so that we can compute their maximum likelihood decoding threshold. Also, we avoid complicated combinations of codes, because such codes would have dense graph representation and hence possible poor practical iterative decoding performance.

Three natural modifications are investigated in this chapter: i) changing the inner code; ii) changing the outer code; and iii) using more than one interleaver.

Our first idea is to replace the inner code in RA codes by a different rate 1 convolutional code, one with transfer function $1/(1 + D + D^2)$. We call this class of codes Repeat-Delay-Delay (RDD) codes. A coding theorem for this class of codes will be proved in detail. It will be seen that for RDD codes, not only the ML code

[1] Of course this is not generally true for other decoding methods, including iteration decoding. A strong code doesn't always perform well with iterative decoding. A good example of that is the ensemble of low-density parity-check codes discussed in [24].

threshold, but also the iterative decoding performance, will improve considerably over the RA codes.

The second idea is to replace the outer code (the repeat code) in RA codes by a rate 1/2 convolutional code. This code structure was first analyzed in [51]. We call it the Convolution-Accumulate (CA) code. We are able to compute the ML code threshold for this rate 1/2 code, and simulation shows this code also performs well with iterative decoding.

The third class of codes is generated by extending the serial interconnection. Instead of one serial concatenation, we add one more accumulate code, interconnected through another random interleaver. We call this class of codes Repeat-Accumulate-Accumulate (RAA) codes. This class of codes will be shown to have terrific ML code threshold, but since its Tanner graph realization is dense, it does not perform well with iterative decoding method.

4.2 RDD Codes

In this section we introduce and analyze a class of codes named Repeat-Delay-Delay (RDD) codes. The theoretical analysis of RDD codes is similar to that of RA codes. However, in iterative decoding, this class of codes should be realized using a Tanner-Wiberg graph instead of a Tanner graph. Simulation shows that on a Tanner graph representation of RDD codes, iterative decoding performance is very poor, mainly due to the presence of many short cycles.

4.2.1 Code structure

The general structure of RDD codes is shown in Figure 4.1. An information block of length N is repeated q times, scrambled by a $qN \times qN$ interleaver, and then encoded by a rate 1 convolutional encoder with transfer function $1/(1 + D + D^2)$. In practice, this convolutional encoder is also truncated to length qN. We consider the ensemble of this class of codes generated by a uniform interleaver.

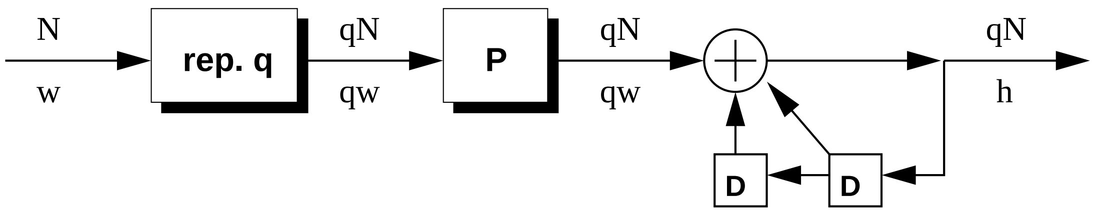

Figure 4.1: Encoder for a (qN, N) RDD code. The numbers *above* the input-output lines indicate the *length* of the corresponding block, and those *below* the lines indicate the *weight* of the block.

Our intention is to prove the IGE conjecture for this class of codes. To apply the union bound from Section 2.2, we need the input-output weight enumerators (IOWE) for both the (qn, n) repetition code, and the truncated (n, n) convolutional code. We already know the IOWE of the repetition code,

$$A^{(o)}_{w,h} = \begin{cases} 0 & \text{if } h \neq qw \\ \binom{n}{w} & \text{if } h = qw. \end{cases} \tag{4.1}$$

The IOWE of the convolutional code, which is derived in Appendix B.1, has the form,

$$A^{(i)}_{w,h} = D \sum_{k=1}^{h-1} \sum_{t=0}^{k-1} \binom{k}{t} \binom{k}{\lfloor s \rfloor} \binom{h-k-1}{k-t-1} \binom{n-h-k-1}{k-\lceil s \rceil - 1}, \tag{4.2}$$

where $s = (h-w)/2 + t$, and D is a constant between 1 and 7.

It follows from the general formula (2.14), that for the (qN, N) RDD code represented by Figure 4.1, the ensemble IOWE is:

$$\begin{aligned} A^{(qN)}_{w,h} &= \sum_{h_1=1}^{qN} \frac{A^{(o)}_{w,h_1} A^{(i)}_{h_1,h}}{\binom{qN}{h_1}} \\ &= D \frac{\binom{N}{w}}{\binom{qN}{qw}} \sum_{k=1}^{h-1} \sum_{t=0}^{k-1} \binom{k}{t} \binom{k}{\lfloor s \rfloor} \binom{h-k-1}{k-t-1} \binom{qN-h-k-1}{k-\lceil s \rceil - 1}, \end{aligned} \tag{4.3}$$

where $s = (h-qw)/2 + t$. Because $k - t \leq h - k$, we have $k - t \leq h/2$. Therefore, $k - \lceil s \rceil - 1 = k - t - \lceil \frac{h-qw}{2} \rceil - 1 \leq \frac{h}{2} - \lceil \frac{h-qw}{2} \rceil - 1 \leq \lfloor qw/2 \rfloor - 1$, with equality if and only if $t = 0$. Now from (4.3) is is easy to compute the parameter $\alpha(w,h)$ in (2.16),

$$\alpha(w,h) = \max_s w - qw + k - \lceil s \rceil - 1 = w - qw + \lfloor qw/2 \rfloor - 1. \tag{4.4}$$

Hence, the *interleaving exponent* β_M in (2.18) is

$$\begin{aligned} \beta_M &= \max_w \max_h \alpha(w,h) \\ &= \lceil -q/2 \rceil. \end{aligned} \tag{4.5}$$

It follows that the IGE conjecture in this case is that a q-repetition RDD code has word error probability interleaving gain only if $q \geq 2$, instead of $q \geq 3$ for the RA codes.

4.2.2 Coding theorem

In this subsection, we prove the IGE Conjecture for the q-repetition RDD code ensemble. Our main analytical tool is Theorem A.1 from Appendix A. The following theorem is an instantiation of Theorem A.1 for Divsalar bound.

Theorem 4.1 *If D_n is a sequence of positive integers such that*

$$\lim_{n \to \infty} \frac{n\theta_n}{D_n} = 0,$$

and

$$c_o \triangleq \sup_{0<\delta<1} \frac{1-\delta}{\delta} \frac{1-e^{-2r(\delta)}}{2},$$

then if $c > c_0$, there exists an integer n_0 and positive constants K and ϵ such that for $n \geq n_0$,

$$P_W^{(n)} \leq Z^{(n)}(D_n) + Ke^{-\epsilon D_n}. \tag{4.6}$$

The following property, which is also a minimum distance property, gives an upper bound of the first term in (4.6). This is proved in Appendix B.2.

Property 4.1 *For RDD codes, asymptotically*

$$Z^{(n)}(D_n) = O(n^{\lceil -q/2 \rceil + \epsilon}), \tag{4.7}$$

for any constant $\epsilon > 0$.

From formula (4.3), the spectral shape $r(\delta)$ in (2.5) can be shown (Appendix B.3) to be

$$\begin{aligned} r(\delta) &= \max_{0<x,u,v<1} \Big(-\frac{q-1}{q} H(qx) + (\delta - u)H(\frac{u-s}{\delta - u}) + (1-\delta-u)H(\frac{u-s-\delta/2+qv/2}{1-\delta-u}) \\ &+ uH(v/u) + uH(\frac{\delta/2 - qx/2 + v}{u})\Big), \end{aligned} \tag{4.8}$$

where $H(x) = -x\log x - (1-x)\log(1-x)$ is the binary entropy function. Also, as n becomes large, $r_n(\delta) \le r(\delta) + 3\log n/n$ (see Appendix B.3).

Taking $D_n = \log^2 n$ and combining with Property 4.1, Theorem 4.1 gives the following conclusion:

Theorem 4.2 *For a q-repetition RDD Code, if* $E_b/N_o > qc_o$*, then asymptotically*

$$P_W^{(n)} = O(n^{\beta_M + \epsilon}) \tag{4.9}$$

where $\epsilon > 0$*, and* $\beta_M = \lceil -q/2 \rceil$ *is the interleaving exponent, and*

$$c_o = \sup_{0<\delta<1} \frac{1 - e^{-2r(\delta)}}{2} \frac{1-\delta}{\delta}.$$

Table 4.1 lists the SNR threshold values for RDD codes using Theorem 4.2, where q ranges from 3 to 8. Compared with RA codes, RDD codes have considerable improvements with respect to ML decoding performance, with just a little addition of complexity. For example, a rate 1/4 RDD code has ML code threshold -0.687 dB, while a rate 1/4 RA code has threshold -0.052 dB.

q code rate	3 1/3	4 1/4	5 1/5	6 1/6	7 1/7	8 1/8
RA Code	0.792	-0.052	-0.480	-0.734	-0.900	-1.015
RDD Code	-0.189	-0.687	-0.920	-1.050	-1.142	-1.207
Random Code	-0.453	-0.774	-0.952	-1.066	-1.145	-1.203
Binary Shannon Limit	-0.495	-0.794	-0.963	-1.071	-1.150	-1.210

Table 4.1: Numerical data for E_b/N_o thresholds of RDD codes.

4.2.3 Performance of RDD codes with iterative decoding

The results of the previous subsection show that the performance of RDD codes with ML decoding is better than RA codes. In this subsection, we consider the performance of RDD codes with iterative decoding. Iterative decoding is applied to a Tanner-Wiberg graph realization [48, 52] of this class of codes. In brief, a Tanner-Wiberg graph is an extention of the Tanner graph. It introduces a new set of codes, "state" nodes, in addition to the variable and check nodes in a Tanner graph.

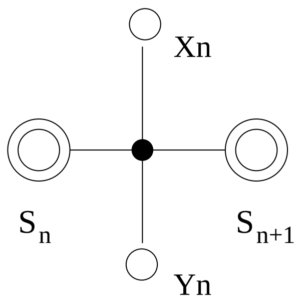

Figure 4.2: Tanner-Wiberg graph representation of a convolutional code.

For example, since a convolutional code can be regarded as a finite state machine, whose current output y_n and next state s_{n+1} depend uniquely on the current state s_n and the current input x_n, it has a Tanner-Wiberg graph realization as shown in Figure 4.2.

Now it is well known that if we decode convolutional codes on this graph realization, the min-sum algorithm becomes the traditional Viterbi algorithm, which is optimal with respect to word error probability; the sum-product algorithm becomes BCJR algorithm, optimal with respect to bit error probability [52, 3].

It is not difficult to find the Tanner-Wiberg graph representation of RDD codes. Figure 4.3 shows the Tanner-Wiberg realization of a $q = 3$ (6,2) RDD code.

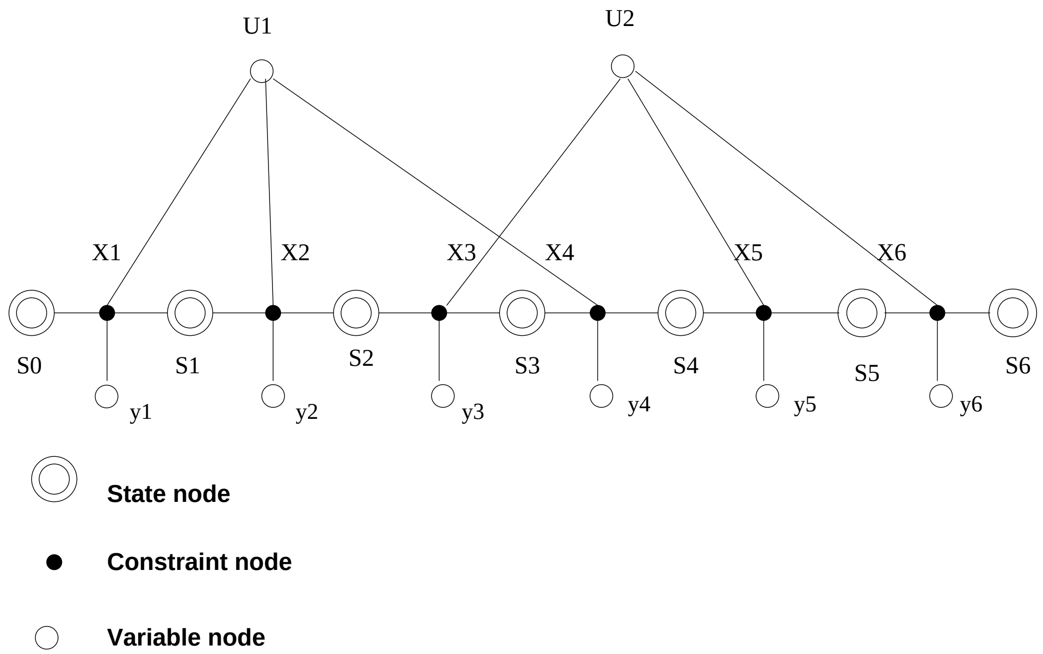

Figure 4.3: Tanner-Wiberg graph representation of a RDD code.

Iterative decoding performance was simulated on an AWGN channel. Figure 4.4 shows the performance of a rate 1/3 RDD code. The interleaver deployed is an algebraic LCM interleaver. Block lengths 1024 and 4096 are presented. In Figure 4.4, solid lines show the BER performance and dashed lines show the WER performance. This should be compared to the performance of RA codes with iterative decoding. With the same block length 4096, to achieve BER of 10^{-4} it requires 1.3 dB for the

RA codes but only requires 0.6 dB for the RDD code, which is a 0.7 dB improvement.

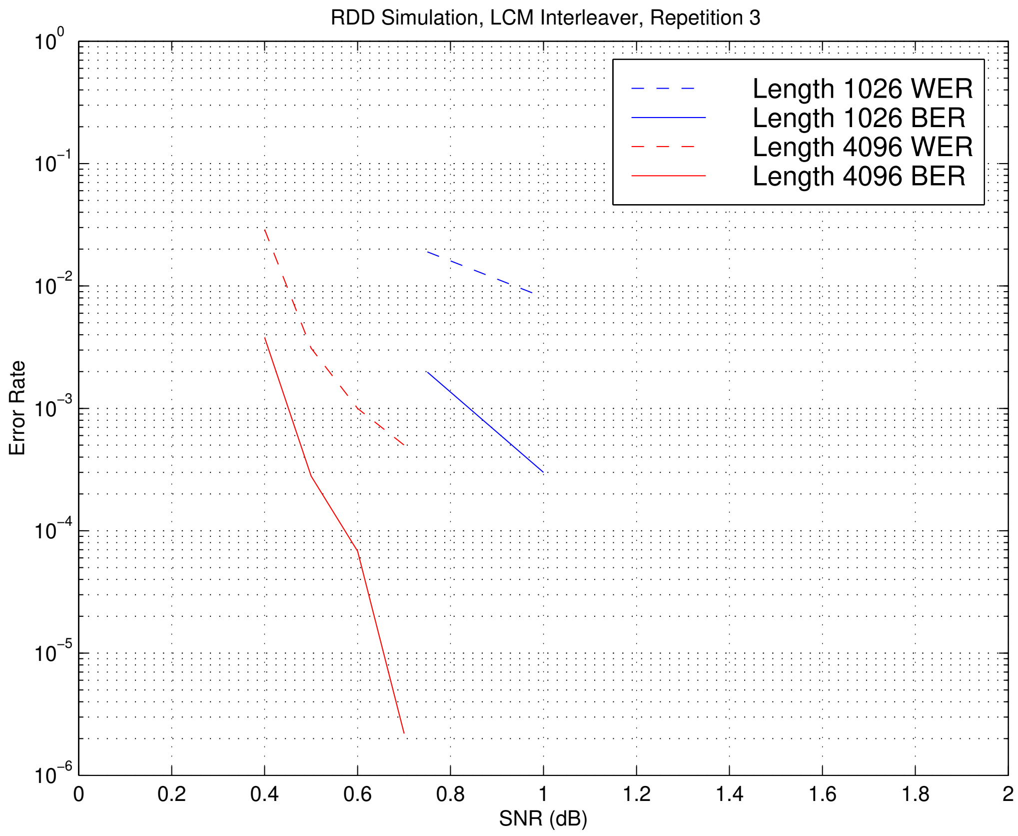

Figure 4.4: Simulated performance of iterative decoding of RDD codes on an AWGN channel.

4.3 Convolution-Accumulate Code

In this section we study a serial turbo code introduced in [51]. This serial concatenation code consists of a rate 1/2 convolutional outer code and an accumulate inner code, interconnected by a uniform interleaver. Some theoretical analyses for this code ensemble were given in [51], where the authors recursively calculated the weight enumerators for block length up to 4096 and applied that to an improved union bound to predict its maximum likelihood decoding performance. Due to the computation load, this technique is limited to relatively short block lengths. In this section, we will derive the performance limit of this code ensemble for large block length. We first derive a general expression of the weight enumerators for this ensemble, from which we are able to prove an AWGN coding theorem by applying Theorem 4.1. The threshold γ_o of this code is shown to be 0.384 dB, only about 0.2 dB from the binary-input Shannon limit.

Figure 4.5 shows the structure of the CA code. The outer code is a truncated convolutional code with transfer function $(1+D^2, 1+D+D^2)$, the inner code is an accumulate code.

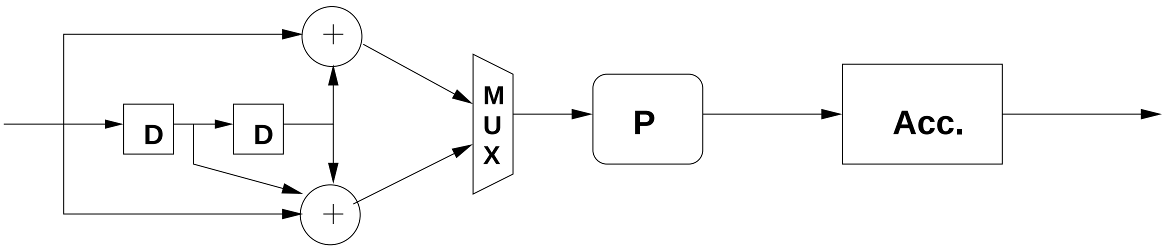

Figure 4.5: Convolution-Accumulate code.

For the outer code, the input block $[x_1, \ldots, x_n]$ and output block $[y_1^{(1)}, y_1^{(2)}, \ldots, y_n^{(1)}, y_n^{(2)}]$ are related by the formula

$$\begin{aligned} y_1^{(1)} &= x_1, & y_1^{(2)} &= x_1 \\ y_2^{(1)} &= x_2, & y_2^{(2)} &= x_2 \\ y_3^{(1)} &= x_3 + x_1, & y_3^{(2)} &= x_3 + x_2 + x_1 \\ &\vdots \\ y_n^{(1)} &= x_{n-2} + x_n, & y_n^{(2)} &= x_{n-2} + x_{n-1} + x_n. \end{aligned} \tag{4.10}$$

4.3.1 IOWE of the outer code

In this subsection, we derive the weight enumerator of the outer code. The outer code can be viewed as a composition of two convolutional codes, $C^{(1)}$ and $C^{(2)}$, where $C^{(1)}$ has transfer function $1 + D^2$ and $C^{(2)}$ has transfer function $1 + D + D^2$. If we are able to obtain weight enumerators for both codes, the overall weight enumerator will simply be:

$$A_{w,h} = \sum_{h_1} A^{(1)}_{w,h_1} A^{(2)}_{w,h-h_1}.$$

$C^{(2)}$ is a RDD code if we exchange the roles of input and output, i.e., we view the input as output and the output as input. Hence the IOWE $A^{(2)}_{w,h}$ of $C^{(2)}$ is simply formula (4.3) after swapping w and h everywhere.

The IOWE of $C^{(1)}$ also can be obtained using previous results. A bit closer look reveals that the input and output sequence of $C^{(1)}$ can be completely decoupled into one evenly indexed and one oddly indexed subsequence, both being a pair of accumulate code sequences. In other words, $(x_1, x_3, \dots)$ and $(y_1, y_3, y_5, \dots)$ satisfy the accumulate condition, and so do $(x_2, x_4, \dots)$ and $(y_2, y_4, \dots)$. This facilitates the computation of weight enumerator of $C^{(1)}$,

$$A^{(1)}_{w,h} = \sum_{h_1+h_2=h, w_1+w_2=w} A_{w_1,h_1} A_{w_2,h_2}, \tag{4.11}$$

where A_{w_1,h_1} is given in eq. (3.3).

4.3.2 Weight spectral shape

Given the weight enumerators of its outer and inner codes, the spectral shape of CA code can be easily obtained (omitting the details):

$$\begin{aligned} r(\delta) \;=\; & \max_{x,u,v,z} uH\left(\frac{z-x}{4u}\right) + uH\left(\frac{v}{u}\right) + (x-u)H\left(\frac{u-v}{x-u}\right) \\ & +(1/2-x-u)H\left(\frac{(z-x)/4}{1/2-x-u}\right) + (1-\delta)H\left(\frac{z/2}{1-\delta}\right) \\ & +\delta H\left(\frac{z/2}{\delta}\right) - H(z). \end{aligned}$$

This is plotted in Figure 4.6 together with rate 1/2 random codes and RA codes. Compared to RA codes, its spectral shape fits more closely to the random codes, which suggests a better ML decoding threshold. Indeed, its SNR threshold as computed by Theorem A.1 is 0.384 dB, very close to the channel capacity 0.184 dB.

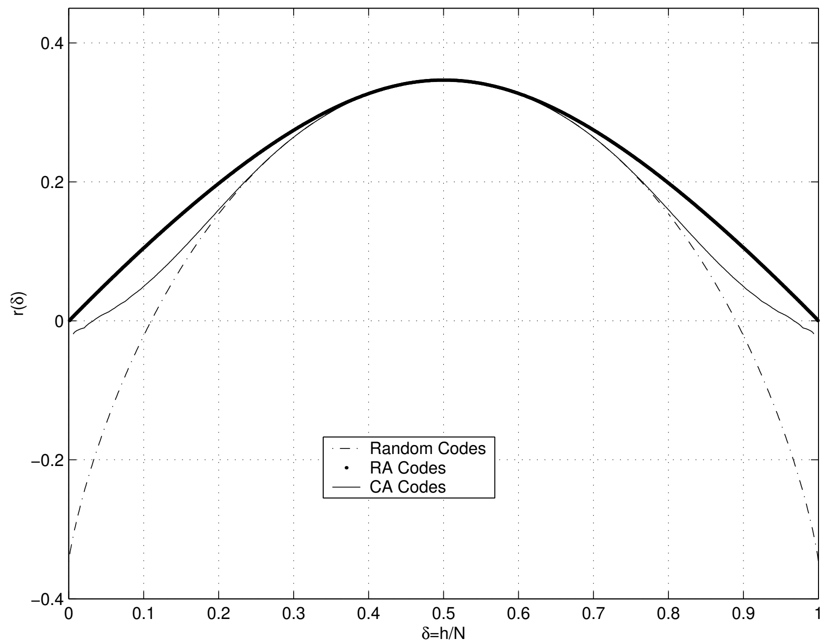

Figure 4.6: Weight spectral shape of Convolution-Accumulate code.

4.3.3 Performance of CA code with iterative decoding

We decode CA code using the traditional BCJR (forward-backward) algorithm, instead of the LDPC-like sum-product algorithm used for RA codes and RDD codes.

Figure 4.7 shows the performance of CA code with information block lengths 1024 and 16384. To achieve BER at 10^{-4}, CA code of length 16384 needs E_b/N_0 about 1.25 dB, 1.0 dB away from Shannon's limit.

4.4 Repeat-Accumulate-Accumulate Codes

RA codes, RDD codes, and CA code demonstrate the importance of the interleaver in a turbo-like code structure. In this section, we explore the possible advantage of

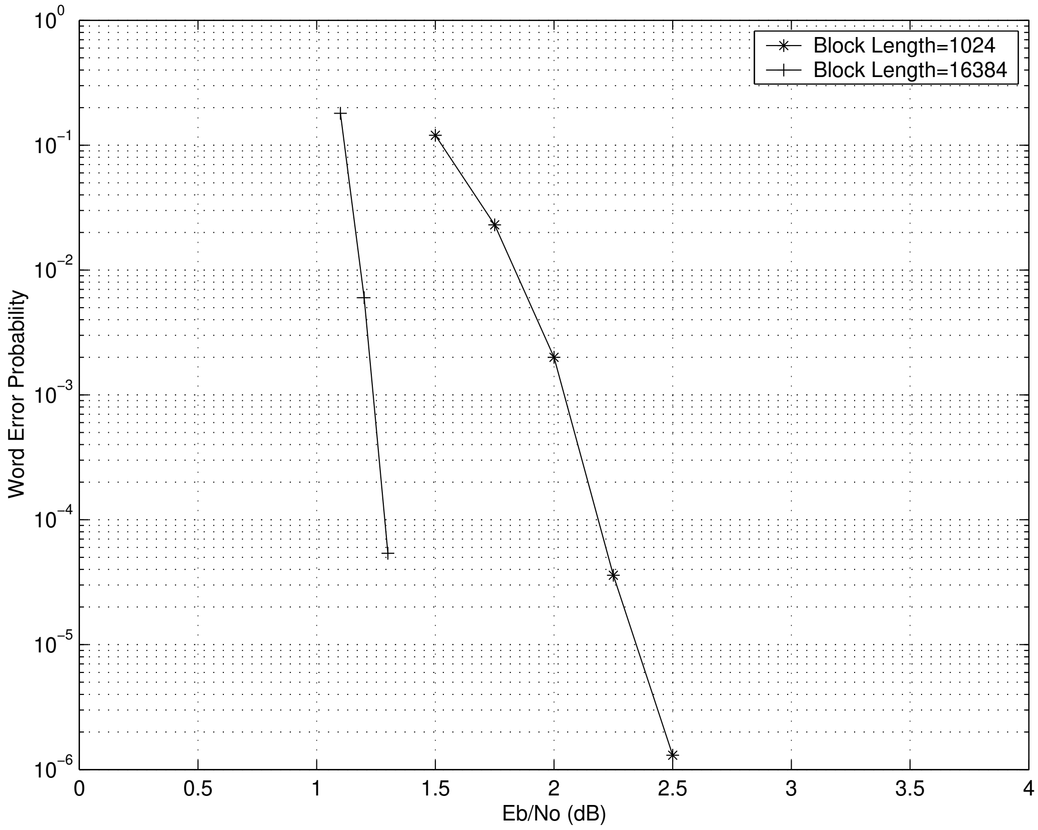

Figure 4.7: Performance of iterative decoding of Convolution-Accumulate code on an AWGN channel.

using more than one interleaver in a serial concatenation scheme. In this direction, the only code we have investigated consists of one repetition code and two accumulate codes. Shown in Figure 4.8, those three codes are serially interconnected by two pseudorandom interleavers, making an overall repeat-accumulate-accumulate (RAA) code of rate $1/q$. This code was independently investigated in [9] and [42].

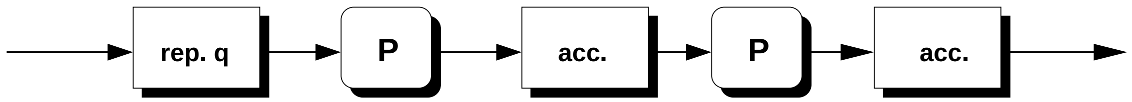

Figure 4.8: Structure of RAA codes.

We consider the ensemble generated by two uniform interleavers. The ensemble weight enumerator is

$$A_{w,h} = \sum_{d_1,d_2} A^{(r)}_{w,d_1} \frac{A^{(a)}_{d_1,d_2}}{\binom{n}{d_1}} \frac{A^{(a)}_{d_2,h}}{\binom{n}{d_2}}, \tag{4.12}$$

where $A^{(r)}_{w,h}$ and $A^{(a)}_{w,h}$ are the IOWE for the repetition code and the accumulate code respectively. Those are given in formula (3.2) and (3.3). Then it is straightforward to obtain the spectral shape of this code ensemble:

$$r_q(\delta) = \max_{0 \le x \le \min(2\delta, 2(1-\delta))} \max_{0 \le y \le \min(2x, 2(1-x))} H(y)/q - F(y,x) - F(x,\delta), \tag{4.13}$$

where $F(y,x)$ is defined as

$$F(y,x) = H(y) - xH(\frac{y}{2x}) - (1-x)H(\frac{y}{2-2x}).$$

We can thus prove coding theorems for RAA codes by applying Theorem 4.1. Omitting details, we display the code thresholds for RAA codes in Table 4.2. Those thresholds are extremely close to Shannon's theoretical limit. For example, at rate 1/4, RAA codes have SNR threshold -0.437 dB, while the Shannon limit is -0.495 dB.

q code rate	2 1/2	3 1/3	4 1/4	5 1/5	6 1/6	7 1/7	8 1/8
RAA Code	0.4455	-0.437	-0.771	-0.951	-1.066	-1.145	-1.203
Random Code	0.308	-0.453	-0.774	-0.952	-1.066	-1.145	-1.203
Binary Shannon Limit	0.184	-0.495	-0.794	-0.963	-1.071	-1.150	-1.210

Table 4.2: Numerical data for E_b/N_o thresholds of RAA codes.

Although their estimated performance with ML decoding are stunningly close to the channel capacity, RAA codes don't perform well with iterative decoding. Figure 4.9 shows performance curves for a rate 1/3 RAA code of $k = 1024$, along with an RDD code and an RA code of the same rate and block length. As shown, RAA codes are worse than RDD codes, despite their consistently superior ML thresholds, although they are still better than RA codes. It was shown in [19] that the iterative decoding capacity of a rate 1/3 RAA code is about 0.4 dB, only 0.09 dB better than the RA code of same rate.

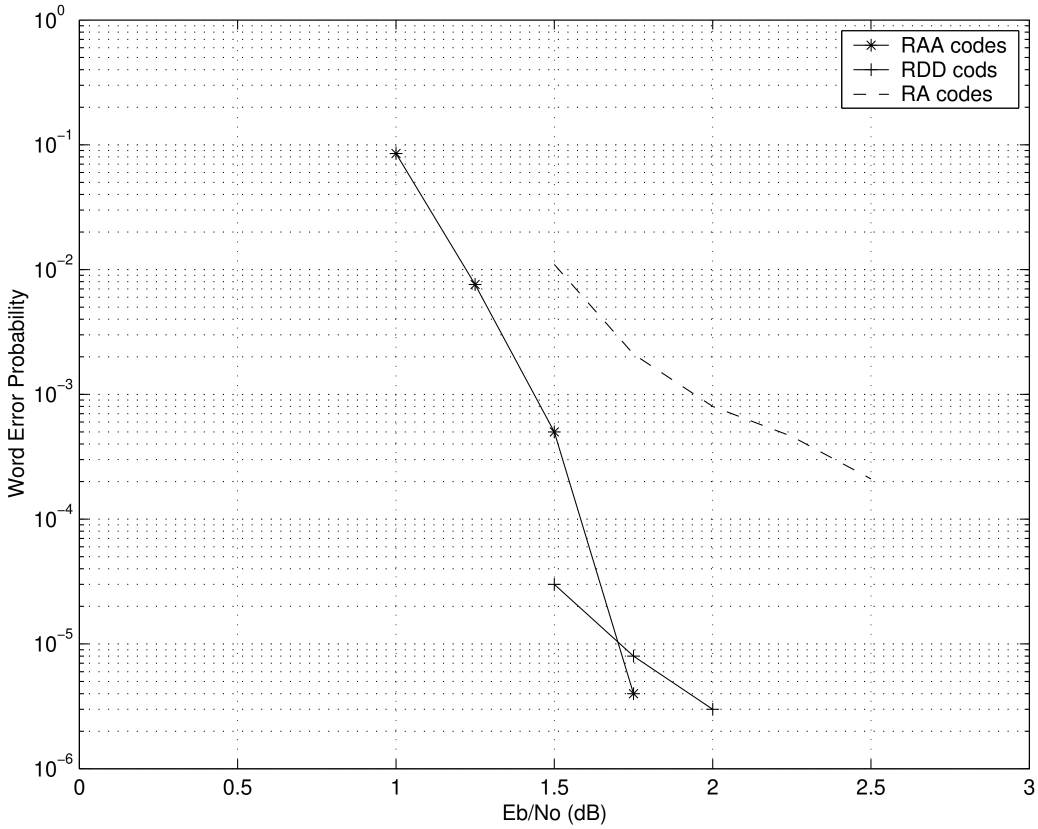

Figure 4.9: Performance comparison of rate 1/3 RAA, RDD, and RA codes with iterative decoding, block length 1024.

Chapter 5 Coding Theorems for Turbo Code Ensembles

5.1 Introduction

This chapter is devoted to a Shannon-theoretic study of turbo codes. By "Shannon-theoretic" we mean a study of the average performance of the codes in the turbo code ensemble under maximum likelihood decoding. While there appears to be no possibility of implementing exact maximum-likelihood decoding of turbo codes, the celebrated iterative message-passing turbo decoding algorithm seems to be, in most cases, a close approximation to MLD. Thus it is of interest to know the MLD potential for this class of codes.

In a way, this chapter is simply an elaboration of the following remark, which was made in [40]:

"The presence [in turbo-codes] of the pseudorandom interleavers between the component codes ensures that the resulting overall code behaves very much like a long random code, and by Shannon's theorems, a long random code is likely to be "good" . . . "

In this chapter, building on ideas pioneered by Benedetto et al. [4], we will prove that turbo codes are indeed good, in the following sense. For a turbo code ensemble, parallel or serial, defined by a set of fixed component codes (subject only to to mild necessary restrictions), if the channel's Bhattacharya noise parameter is sufficiently small, we will show that the average maximum-likelihood decoder error probability approaches zero, at least as fast as $n^{-\beta}$, where β is the (ensemble-dependent) "interleaver gain" exponent first defined by Benedetto et al. in 1996 [4].

Here is an outline of this chapter. Section 5.2 defines the parallel and serial turbo code ensemble. Section 5.3 proves a coding theorem for general code ensembles, combining the ensemble weight enumerator with the union bound. Section 5.4 and 5.5 give estimates of the weight enumerators of the parallel and serial turbo codes respectively. Section 5.6 states the main results and gives a proof. Section 5.7 illustrates our main results with several examples. And finally, Section 5.8 contains our discussion and conclusions. Some complementary material is given in Appendix C.

5.2 The Turbo Code Ensembles

The general structure of a parallel turbo code is shown in Figure 5.1. There are J interleavers (pseudorandom scramblers) $P_1, P_2, \ldots, P_J$[1] and J recursive convolutional encoders $E_1, E_2, \ldots, E_J$. An information block of length k is scrambled by interleaver P_i and then encoded (and truncated) by E_i, producing a codeword of length n_i, for $i = 1, 2, \ldots, J$. These J codewords are then sent to the channel. The overall code is therefore a (n, k) linear block code, with $n = \sum_{i=1}^{J} n_i$. If $R_i = k/n_i$ is the rate of the ith component code E_i, then overall code rate is easily seen to be $R = (\sum_{i=1}^{J} R_i^{-1})^{-1}$. Because there are $k!$ choices for each interleaver, there are a large number of codes with the structure shown in Figure 5.1. We call this set of codes the $[E_1 \| E_2 \| \cdots \| E_J]$ ensemble. (A more precise definition of code ensemble is given in Section 2.2.)

Our first main result (Theorem 5.3) is that if $J \geq 2$, the $[E_1 \| E_2 \| \cdots \| E_J]$ ensemble is "good" on any memoryless binary input channel with nonzero capacity.

A serial turbo code has the general structure shown in Figure 5.2. An information block of length k is encoded by an outer encoder E_1 into a codeword of length N, which is scrambled by an interleaver P, and then encoded by an inner encoder E_2 into a codeword of length n. The outer code E_1 is a truncated convolutional code,[2] and the inner code E_2 is a truncated recursive convolutional code. The overall code is therefore an (n, k) linear block code, with rate $R = R_1 R_2$, where R_1 is the rate of the outer code and R_2 is the rate of the inner code. Because of the choices for the

[1] Without loss of generality, we may assume that P_1 is the identity permutation, so that there are only $J - 1$ interleavers.

[2] We note that a block code can be viewed as a convolutional code without memory, so that E_1 may be a block encoder.

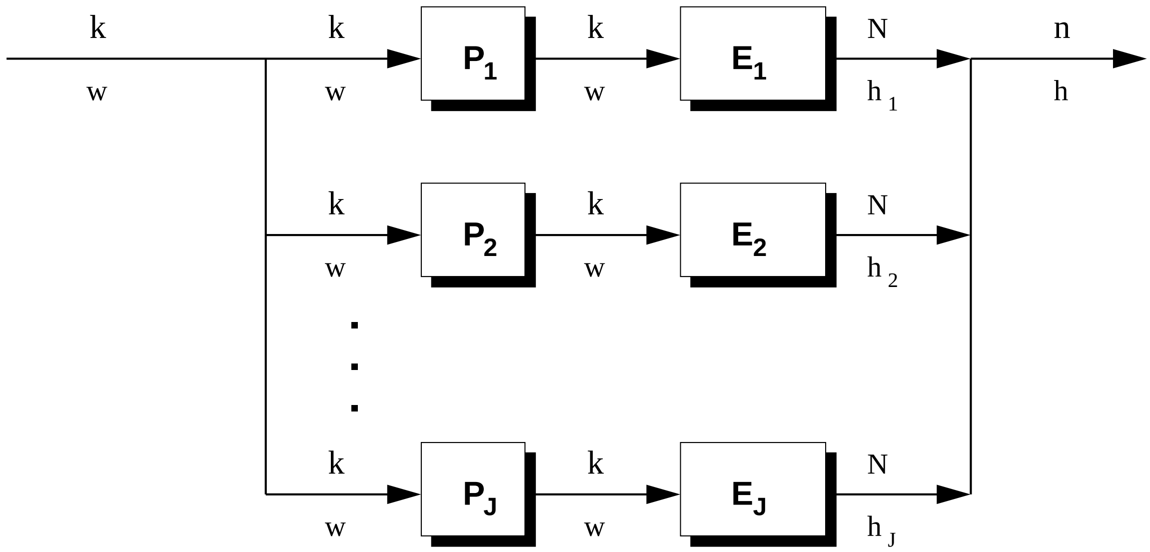

Figure 5.1: Encoder for a parallel turbo code with J branches. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the Hamming weight of the block.

interleaver, there are $N!$ codes with the structure shown in Figure 5.2. We call this set of codes the $[E_1 \Rightarrow E_2]$ ensemble.

Our second main result (Theorem 5.4) is that if the minimum distance of the outer code E_1 is at least three, the $[E_1 \Rightarrow E_2]$ ensemble is also "good" on any memoryless binary input channel with nonzero capacity.

It is worth noting here that the main difficulty in proving our main results (Theorems 5.3 and 5.4) is that we are unable to compute $r(\delta)$ for the $[E_1 \| E_2 \| \cdots \| E_J]$ and $[E_1 \Rightarrow E_2]$ ensembles. Instead, we have had to resort to upper bounds on $r(\delta)$ (see (5.18) and (5.26)), based on the work of Kahale and Urbanke [35], which render our results mere existence theorems.

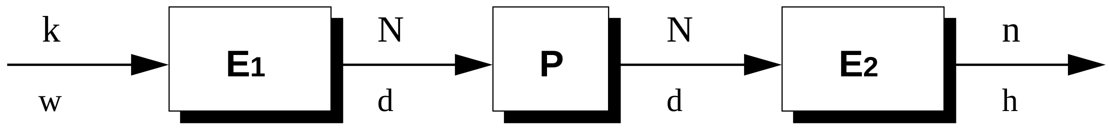

Figure 5.2: Encoder for a serial turbo code with J branches. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the Hamming weight of the block.

5.3 A Coding Theorem

In this section, we present an upper bound on the ML decoder word error probability for an ensemble of binary linear codes.[3] It shows that under certain conditions, there exists a threshold c_0 such that if the channel noise exponent α exceeds c_0, the ensemble word error probability approaches 0. We shall see that the low-weight codewords determine whether or not the threshold c_0 is finite.

To begin, we introduce some notation. First, let D_n be a fixed sequence of integers satisfying

$$\frac{D_n}{n^\epsilon} \to 0, \qquad \text{for all } \epsilon > 0 \tag{5.1}$$

$$\frac{\log n}{D_n} \to 0. \tag{5.2}$$

For example, $D_n = \log^2 n$ will do. Second, we define the *noise thresholds* for the ensemble:

$$c_0^{(n)} \triangleq \sup_{D_n/n<\delta\leq 1} r_n(\delta)/\delta \tag{5.3}$$

[3] Parallel and serial turbo codes are important examples of code ensembles, but this result applies to other ensembles, as well. Refer to Section 2.1 for a general definition of a code ensemble.

(weight spectral shape $r_n(\delta)$ is defined in Section 2.1) and

$$c_0 \triangleq \limsup_{n\to\infty} c_0^{(n)}. \tag{5.4}$$

Finally, the nth *innominate sum* is defined as follows:

$$Z^{(n)}(D) \triangleq \sum_{h=1}^{D} \overline{A}_h^{(n)},$$

where D is an integer with $1 \le D \le n$. In words, $Z^{(n)}(D)$ is the average number of words of weight $\le D$ for a code in the set C_n. (It is also an upper bound on the probability that the minimum distance of a code in C_n is $\le D$.)

Theorem 5.1 *Suppose the ensemble threshold c_0 defined in (5.4) is finite, and the channel error exponent α satisfies $\alpha > c_0$. Then if $P_W^{(n)}$ denotes the ensemble maximum-likelihood decoder error probability, there exists an integer n_0 and positive constants K and ϵ such that for $n > n_0$,*

$$\overline{P}_W^{(n)} \le Z^{(n)}(D_n) + Ke^{-\epsilon D_n}. \tag{5.5}$$

Proof: Since the channel error exponent α is nonnegative, we have

$$A_h e^{-\alpha h} \le A_h.$$

Therefore by formulas (2.11) and (2.12) in Section 2.2,

$$\begin{aligned} \overline{P}_W^{(n)} &\le \sum_{h=1}^{D_n} \overline{A}_h^{(n)} + \sum_{h>D_n} \overline{A}_h^{(n)} e^{-\alpha h} \\ &= Z^{(n)}(D_n) + \sum_{h>D_n} e^{-h(\alpha - r_n(\delta)/\delta)}. \end{aligned} \tag{5.6}$$

If $\alpha > c_0$, then there exists an integer n_0, and an $\epsilon > 0$ such that for $n > n_0$,

$\alpha - c_0^{(n)} > \epsilon$. Hence for $n > n_0$ and $h > D_n$, we have

$$\alpha - \frac{r_n(\delta)}{\delta} \geq \alpha - c_0^{(n)} > \epsilon,$$

so that

$$e^{-h(\alpha - r_n(\delta)/\delta)} \leq e^{-h\epsilon}. \tag{5.7}$$

Thus

$$\sum_{h>D_n} e^{-h(\alpha - r_n(\delta)/\delta)} \leq \sum_{h>D_n} e^{-h\epsilon} = Ke^{-D_n\epsilon}, \tag{5.8}$$

where $K = e^{-\epsilon}/(1-e^{-\epsilon})$. Substituting (5.8) into (5.6), we have (5.5). ■

Corollary 5.1 *If, in addition, $Z^{(n)}(D_n) = O(n^{-\beta})$ where $\beta > 0$, then for $\alpha > c_0$,*

$$P_W^{(n)} = O(n^{-\beta}). \tag{5.9}$$

Proof: Note that $n^{-\beta} = e^{-\beta \log n}$. The result now follows from (5.5) and (5.2).

The question as to whether c_0 is finite is partially answered by the following two technical results.

Theorem 5.2 *For a code ensemble $\mathcal{C}$, the code threshold c_0 is finite if and only if for all sequences ϵ_n such that $\epsilon_n > D_n/n$ and $\epsilon_n \to 0$,*

$$c_0' = \lim_{n\to\infty} \sup_{D_n/n<\delta<\epsilon_n} r_n(\delta)/\delta \tag{5.10}$$

is finite.

Proof: Clearly

$$\sup_{D_n/n<\delta<\epsilon_n} \frac{r_n(\delta)}{\delta} \leq \sup_{D_n/n<\delta\leq 1} \frac{r_n(\delta)}{\delta},$$

so that if c_0 as defined in (5.3) is finite, so is c_0', for any choice of ϵ_n.

To complete the proof, we now show that if c_0' is finite, so is c_0, via the contrapositive. If c_0 is infinite, then there is a convergent sequence $\delta_n \to \delta_0$ such that $D_n/n < \delta_n \leq 1$ with

$$\lim_{n\to\infty} \frac{r_n(\delta_n)}{\delta_n} = \infty. \tag{5.11}$$

If $\delta_0 > 0$, note that $\overline{A}_h^{(n)} \leq \binom{n}{h} \leq e^{nH(\delta)}$,[4] hence $r_n(\delta) = \log \overline{A}_h^{(n)}/n \leq H(\delta)$. Thus

$$\lim_{n\to\infty} \frac{r_n(\delta_n)}{\delta_n} \leq \frac{H(\delta_0)}{\delta_0},$$

which contradicts (5.11). Thus $\delta_0 = 0$. Hence if we define $\epsilon_n = \min(2\delta_n, 1)$, we have

$$\sup_{D_n/n<\delta<\epsilon_n} \frac{r_n(\delta)}{\delta} \geq \frac{r_n(\delta_n)}{\delta_n} \to \infty.$$

Thus (5.11) diverges, which shows that c_0' is infinite. ■

Corollary 5.2 *If there exists a function $s(\delta)$ and constants $\gamma_n = O(D_n/n)$ such that $r_n(\delta) \leq \gamma_n + s(\delta)$ for all sufficiently small δ and all sufficiently large n, then the ensemble noise threshold c_0 is finite provided*

$$\limsup_{\delta\to 0} \frac{s(\delta)}{\delta} < \infty. \tag{5.12}$$

Proof: We use Theorem 5.2. Thus let ϵ_n be a sequence such that $\epsilon_n > D_n/n$ and $\epsilon_n \to 0$. Then

$$\begin{aligned}
\lim_{n\to\infty} \sup_{D_n/n<\delta<\epsilon_n} r_n(\delta)/\delta &\leq \lim_{n\to\infty} \sup_{D_n/n<\delta<\epsilon_n} (\gamma_n + s(\delta))/\delta \\
&\leq \limsup_{n\to\infty} (n\gamma_n/D_n) + \lim_{n\to\infty} \left(\sup_{0<\delta<\epsilon_n} s(\delta)/\delta \right) \\
&\leq K + \limsup_{\delta\to 0} s(\delta)/\delta \\
&< \infty.
\end{aligned}$$

[4]We have collected several useful inequalities on binomial coefficients in Appendix C.2.

Thus by Theorem 5.2, the code threshold c_0 is finite. ■

5.4 Weight Enumerator Estimates for Parallel Turbo Code Ensembles

For the $[E_1\|E_2\|\cdots\|E_J]$ ensemble, the average IOWE can be obtained from the IOWEs of the component codes using the "uniform interleaver" technique [4] (see Section 2.3 for general formula of turbo-like codes):

$$\overline{A}^{(n)}_{w,h} = \frac{1}{\binom{k}{w}^{J-1}} \sum_{\sum_{i=1}^{J} h_i = h} \prod_{i=1}^{J} A^{[i]}_{w,h_i}, \tag{5.13}$$

where $A^{[i]}_{w,h_i}$ is the IOWE for the ith component code C_i (see Figure 5.1). Therefore, the *accumulative weight enumerator* (defined in Section 2.1)

$$\begin{aligned}
\overline{A}^{(n)}_{w,\leq h} &= \frac{1}{\binom{k}{w}^{J-1}} \sum_{\sum_{i=1}^{J} h_i \leq h} \prod_{i=1}^{J} A^{[i]}_{w,h_i} \\
&\leq \frac{1}{\binom{k}{w}^{J-1}} \sum_{h_1=1}^{h} \cdots \sum_{h_J=1}^{h} \prod_{i=1}^{j} A^{[i]}_{w,h_i} \\
&= \frac{1}{\binom{k}{w}^{J-1}} \prod_{i=1}^{J} (\sum_{h_i=1}^{h} A^{[i]}_{w,h_i}) \\
&= \frac{\prod_{i=1}^{J} A^{[i]}_{w,\leq h}}{\binom{k}{w}^{J-1}}.
\end{aligned}$$

Next, we apply the bound of Theorem C.3 from Appendix C to each $A^{[i]}_{w,\leq h}$ in (5.14). The truncation length for each C_i is less than its code length n_i, which in turn is strictly less than $n = \sum_i n_i$. Defining $\eta = \max_i \eta_i$, and noting that the binomial coefficient $\binom{n}{j}$ is an increasing function of n, we have

$$A^{[i]}_{w,\leq h} \leq \theta_i^w \sum_{j=0}^{\lfloor w/2 \rfloor} \binom{n}{j} \binom{\eta h}{w-j}. \tag{5.14}$$

If $\eta h < n$, then by Proposition C.1, $\binom{n}{j}\binom{\eta h}{w-j}$ attains its maximum for $1 \le j \le \lfloor w/2 \rfloor$ at $j = \lfloor w/2 \rfloor$. Thus (provided $h < n/\eta$), each $A^{[i]}_{w,\le h}$ can be bounded as follows:

$$\begin{aligned} A^{[i]}_{w,\le h} &\le \theta_i^w \left(\lfloor w/2 \rfloor + 1\right) \binom{n}{\lfloor w/2 \rfloor} \binom{\eta h}{\lceil w/2 \rceil} \\ &\le (2\theta_i)^w \binom{n}{\lfloor w/2 \rfloor} \binom{\eta h}{\lceil w/2 \rceil} \end{aligned} \tag{5.15}$$

The second inequality follows since $\lfloor w/2 \rfloor + 1 \le 2^w$.

Combining (5.14) and (5.15), we obtain

$$\overline{A}^{(n)}_{w,\le h} \le \theta^w \frac{\binom{n}{\lfloor w/2 \rfloor}^J \binom{\eta h}{\lceil w/2 \rceil}^J}{\binom{k}{w}^{J-1}},$$

for some constant $\theta > 1$. Consequently, for small δ, $\overline{A}^{(n)}_{\le h}$ can be upper bounded as follows:

$$\begin{aligned} \overline{A}^{(n)}_{\le h} &\le \sum_{w=1}^{\mu h} A^{(n)}_{w,\le h} \\ &\le \sum_{w=1}^{\mu h} \theta^w \frac{\binom{n}{\lfloor w/2 \rfloor}^J \binom{\eta h}{\lceil w/2 \rceil}^J}{\binom{k}{w}^{J-1}}. \end{aligned}$$

(The sum in (5.16) stops at μh rather than k because of Theorem C.1). Equation (5.16) will be used to bound the innominate sum $Z^{(n)}(D_n)$ that appears in Theorem 5.1.

To bound $r_n(\delta)$ for small δ, we simplify (5.16) by replacing the summation with the maximum term times the number of terms. Since $\binom{\eta h}{l} < 2^{\eta h}$ for any integer l, and $\mu h \le n$, we have

$$\overline{A}^{(n)}_{\le h} \le n 2^{J\eta h} \theta^{\mu h} \max_{1 \le w \le \mu h} \frac{\binom{n}{\lfloor w/2 \rfloor}^J}{\binom{k}{w}^{J-1}}. \tag{5.16}$$

Using the inequalities in (C.5), we have

$$\begin{aligned}
\frac{\binom{n}{\lfloor w/2 \rfloor}^J}{\binom{k}{w}^{J-1}} &\le \frac{e^{nJH(x/2)}}{e^{nR(J-1)H(x/R)}}(k+1)^{J-1} \\
&\le \frac{e^{nJH(x/2)}}{e^{nR(J-1)H(x/R)}}n^{J-1}, \qquad (5.17)
\end{aligned}$$

where $R = k/n$ (the rate of the overall code), and $x = w/n$.

Combining the definition of $r_n(\delta)$ (2.4) with (5.16) and (5.17), we have

$$\begin{aligned}
r_n(\delta) &= \frac{1}{n}\log \overline{A}_h^{(n)} \\
&\le \frac{1}{n}\log \overline{A}_{\le h}^{(n)} \\
&\le J\frac{\log n}{n} + T\delta + \sup_{0<x\le\mu\delta}\{JH(\frac{x}{2}) - R(J-1)H(\frac{x}{R})\}, \qquad (5.18)
\end{aligned}$$

where T is a constant. Equation (5.18) will used with Theorem 5.2 to prove that c_0 is finite for the $[E_1\|\cdots\|E_J]$ ensemble.

5.5 Weight Enumerator Estimates for Serial Turbo Code Ensembles

For the $[E_1 \Rightarrow E_2]$ ensemble, the average IOWE can be obtained from the weight enumerator of the outer code C_1 and the IOWE of the inner code C_2 [8] (see Figure 5.2):

$$\overline{A}_h^{(n)} = \sum_{d=1}^{N}\frac{A_d^{[1]}A_{d,h}^{[2]}}{\binom{N}{d}}. \qquad (5.19)$$

Hence

$$\overline{A}_{\le h}^{(n)} = \sum_{d=1}^{N}\frac{A_d^{[1]}A_{d,\le h}^{[2]}}{\binom{N}{d}}. \qquad (5.20)$$

Since if $A_{d,h}^{[2]} \ne 0$, d is less than μh by Theorem C.1 (where $\mu = \mu(E_2)$), applying Theorem C.2 to the outer code C_1 and Theorem C.3 to the inner code C_2 with L_1 as the trellis length for C_1 and L_2 as the trellis length for C_2, we obtain

$$\begin{aligned}
\overline{A}_{\leq h}^{(n)} &= \sum_{d=1}^{\mu h} \frac{A_d^{[1]} A_{d,\leq h}^{[2]}}{\binom{N}{d}} \\
&\leq \sum_{d=1}^{\mu h} \theta^d \frac{\binom{L_1}{\lfloor d/d_1 \rfloor}}{\binom{N}{d}} \sum_{j=0}^{\lfloor d/2 \rfloor} \binom{L_2}{j} \binom{\eta h}{d-j},
\end{aligned} \tag{5.21}$$

where d_1 is the free distance of C_1. If C_1 is an (n_1, k_1, m_1) code of rate $R_1 = k_1/n_1$, and C_2 is a (n_2, k_2, m_2) code with rate $R_2 = k_2/n_2$, then we have $L_1 = N/n_1$, $L_2 = n/n_2$, and $N = R_2 n$, so that

$$\begin{aligned}
L_1 &= \frac{k_2}{n_1 n_2} n = \alpha n \\
N &= \frac{k_2}{n_2} n = \beta n \\
L_2 &= \frac{1}{n_2} n = \gamma n.
\end{aligned}$$

Thus (5.21) becomes

$$\overline{A}_{\leq h}^{(n)} \leq \sum_{d=1}^{\mu h} \theta^d \frac{\binom{\alpha n}{\lfloor d/d_1 \rfloor}}{\binom{\beta n}{d}} \sum_{j=0}^{\lfloor d/2 \rfloor} \binom{\gamma n}{j} \binom{\eta h}{d-j}. \tag{5.22}$$

For $\delta = h/n$ small enough, we have $\eta h = \eta \delta n < n$, hence

$$\binom{\gamma n}{j} \binom{\eta h}{d-j} \leq \binom{\gamma n}{\lfloor d/2 \rfloor} \binom{\eta h}{\lceil d/2 \rceil}, \tag{5.23}$$

for any $1 \leq j \leq \lfloor d/2 \rfloor$ by Proposition C.1. Therefore, replacing the inner sum in (5.22) with $\lfloor d/2 \rfloor + 1$ times the right side of (5.23), we have

$$\begin{aligned}
\overline{A}_{\leq h}^{(n)} &\leq \sum_{d=1}^{\mu h} \theta^d \frac{\binom{\alpha n}{\lfloor d/d_1 \rfloor}}{\binom{n}{\beta n}} (\lfloor d/2 \rfloor + 1) \binom{\gamma n}{\lfloor d/2 \rfloor} \binom{\eta h}{\lceil d/2 \rceil} \\
&\leq \sum_{d=1}^{\mu h} (2\theta_1)^d \frac{\binom{\alpha n}{\lfloor h/d_1 \rfloor}}{\binom{\beta n}{d}} \binom{\gamma n}{\lfloor d/2 \rfloor} \binom{\eta h}{\lceil d/2 \rceil}.
\end{aligned} \tag{5.24}$$

(The last inequality because $\lfloor d/2 \rfloor + 1 \leq 2^d$.) The inequality (5.24) will be used to bound the innominate sum $Z^{(n)}(D_n)$.

To bound $r_n(\delta)$, we further simplify (5.24). Using the inequality $\binom{\eta h}{l} < 2^{\eta h}$, and bounding the summation in (5.24) by the number of terms times the maximum term, we have

$$\overline{A}^{(n)}_{\leq h} \leq n\theta^{\mu h} 2^{\eta h} \max_{1 \leq d \leq \mu h} \binom{\alpha n}{\lfloor h/d_1 \rfloor} \binom{\gamma n}{\lfloor d/2 \rfloor} / \binom{\beta n}{d}. \tag{5.25}$$

Using techniques like those that led from (5.16) to (5.18), the spectral shape can thus be upper bounded by the following expression, where $x = d/n$:

$$r_n(\delta) \leq 2\frac{\log n}{n} + T\delta + \sup_{0 < x \leq \mu\delta} \{\alpha H(\frac{x}{\alpha d_1}) + \beta H(\frac{x}{2\beta}) - \gamma H(\frac{x}{\gamma})\}, \tag{5.26}$$

where T is a constant. Equation (5.26) will used with Corollary 5.2 to prove that c_0 is finite for the $[E_1 \Rightarrow E_2]$ ensemble.

5.6 Proof of Main Results

In this section, we give the proofs of our main theorems. These theorems first appeared as conjectures, implicitly in [4] and [8] and explicitly in [18]. Theorem 5.3 can be summarized by saying that the $[E_1 \| \cdots \| E_J]$ ensemble has word error probability interleaving gain exponent $-J+2$, and bit error probability interleaving gain exponent $-J+1$. Theorem 5.4 can be summarized by saying that the $[E_1 \Rightarrow E_2]$ ensemble has word error probability interleaving gain exponent $-\lfloor \frac{d_1-1}{2} \rfloor$, and bit error probability interleaving gain exponent $-\lfloor \frac{d_1+1}{2} \rfloor$.

Theorem 5.3 *For the $[E_1 \| \cdots \| E_J]$ ensemble, if $J \geq 2$, then there exists a positive number c_0, such that for any fixed $\alpha > c_0$ and $\epsilon > 0$,*

$$\begin{aligned} P_W &= O(n^{-J+2+\epsilon}) \\ P_b &= O(n^{-J+1+\epsilon}). \end{aligned}$$

Proof: Given Theorem 5.1 and Corollary 5.1, it will be sufficient to prove the following two lemmas.

Lemma 5.1 *For the $[E_1\|\cdots\|E_J]$ ensemble, if $J \geq 2$, then c_0 is finite.*

Proof: We use Corollary 5.2, with the upper bound (5.18) on the code spectral shape:

$$\begin{aligned}
\gamma_n &= \frac{J\log n}{n} \\
s(\delta) &= T\delta + \sup_{0<x\leq\mu\delta}(JH(x/2) - R(J-1)H(x/R)).
\end{aligned}$$

To show that $\limsup s(\delta)/\delta < \infty$, we need to show that the following limit is finite:

$$\lim_{\delta\to 0}\frac{1}{\delta}\sup_{0<x<\mu\delta}(JH(\frac{x}{2}) - R(J-1)H(\frac{x}{R})).$$

But by Proposition C.3, this is true, since $J/2 - R(J-1)/R = -J/2+1 \leq 0$, for $J \geq 2$. ■

Lemma 5.2 *For the $[E_1\|\cdots\|E_J]$ ensemble, if $J \geq 2$,*

$$Z^{(n)}(D_n) = O(n^{-J+2+\epsilon}),$$

for any positive ϵ.

Proof: Using the upper bound (5.16) on $\overline{A}^{(n)}_{\leq h}$, we have

$$\begin{aligned}
Z^{(n)}(D_n) &= \sum_{h=1}^{D_n}\overline{A}^{(n)}_n = \overline{A}^{(n)}_{\leq D_n} \\
&\leq \sum_{w=1}^{\mu D_n}\theta^w \frac{\binom{n}{\lfloor w/2\rfloor}^J\binom{\eta D_n}{\lceil w/2\rceil}^J}{\binom{Rn}{w}^{J-1}} \\
&\overset{(a)}{\leq} \sum_{w=1}^{\mu D_n}\Theta^w n^{J\lfloor w/2\rfloor-(J-1)w}D_n^{(2J-1)w} && (5.27)\\
&\overset{(b)}{=} O(n^{-J+2+\epsilon}). && (5.28)
\end{aligned}$$

In (a), we have used the following inequalities (see (C.4)): $\binom{n}{\lfloor w/2\rfloor} \le n^{\lfloor w/2\rfloor}$; $\binom{\eta D_n}{\lceil w/2\rceil} \le (\eta D_n)^{\lceil w/2\rceil} < (\eta D_n)^w$; $\binom{Rn}{w} \ge (Rn)^w/w^w \ge (Rn)^w/(\mu D_n)^w$. Here Θ represents a new constant. For (b), the sum in (5.27) can be upper bounded by μD_n times the largest term, which by Proposition C.2 is the $w = 2$ term for large enough n. Notice $D_n = o(n^\epsilon)$ for any positive ϵ by (5.1). ■

Next, we prove the corresponding theorem for serial turbo codes.

Theorem 5.4 *For the $[E_1 \Rightarrow E_2]$ ensemble, with E_2 recursive, if the free distance of the outer code satisfies $d_1 \ge 3$, then there exists a positive number c_0, such that for any fixed $\alpha > c_0$,*

$$\begin{aligned} P_W &= O(n^{-\lfloor \frac{d_1-1}{2} \rfloor + \epsilon}) \\ P_b &= O(n^{-\lfloor \frac{d_1+1}{2} \rfloor + \epsilon}) \end{aligned}$$

for arbitrary $\epsilon > 0$.

Again, because of Theorem 5.1 and Corollary 5.1, it's sufficient to prove the following two lemmas.

Lemma 5.3 *For the $[E_1 \Rightarrow E_2]$ ensemble, if the free distance of the outer code satisfies $d_1 \ge 2$, then c_0 is finite.*

Proof: Corollary 5.2 makes it sufficient to show:

$$\lim_{\delta \to 0} \frac{1}{\delta} \sup_{0<x<\mu\delta} (H(\frac{x}{d_1}) + H(\frac{x}{2}) - H(x)) < \infty.$$

But by Proposition C.3, this is true, since $1/d_1 + 1/2 - 1 \le 0$, for $d_1 \ge 2$. ■

Lemma 5.4 *For the $[E_1 \Rightarrow E_2]$ ensemble, if $d_1 \ge 3$,*

$$Z^{(n)}(D_n) = O(n^{-\lfloor \frac{d_1-1}{2} \rfloor + \epsilon})$$

for arbitrary $\epsilon > 0$.

Proof: With the bound (5.24), we have

$$
\begin{aligned}
Z^{(n)}(D_n) &= \overline{A}^{(n)}_{\le D_n} \\
&\le \sum_{d=d_1}^{\mu D_n} \theta^d \frac{\binom{\alpha n}{\lfloor d/d_1 \rfloor}}{\binom{\beta n}{d}} \binom{n}{\lfloor d/2 \rfloor} \binom{\eta D_n}{\lceil d/2 \rceil} \\
&\overset{(a)}{\le} \sum_{d=d_1}^{\mu D_n} \Theta^d n^{\lfloor d/d_1 \rfloor - \lceil d/2 \rceil} D_n^{d+\lceil d/2 \rceil} \\
&\overset{(b)}{=} O(n^{-\lfloor \frac{d_1-1}{2} \rfloor + \epsilon}).
\end{aligned}
$$

In step (a), we have used the following inequalities (see (C.4)): $\binom{\alpha n}{\lfloor d/d_1 \rfloor} \le (\alpha n)^{\lfloor d/d_1 \rfloor}$; $\binom{\beta n}{d} \ge (\beta n)^d/d^d \ge (\beta n)^d/(\mu D_n)^d$; $\binom{n}{\lfloor d/2 \rfloor} \le n^{\lfloor d/2 \rfloor}$; and $\binom{\eta D_n}{\lceil d/2 \rceil} \le (\eta D_n)^{\lceil d/2 \rceil}$. For step (b), the sum is upper bounded by μD_n times the biggest term, which by Prop. C.2 is the $d = d_1$ term, as n becomes large. The conclusion follows, since $D_n = o(n^\epsilon)$ for any positive ϵ. ■

5.7 Examples

The original, classic, turbo code introduced by Berrou et al. [5] is a parallel concatenation with $J = 2$ recursive convolutional component codes, with $R_1 = 1/2$, $R_2 = 1$, and overall rate $R = 1/3$. Extensive experimental evidence with this ensemble on the AWGN channel suggests that for any value of E_b/N_0 greater than around 0.5 dB, the bit error probability can be made arbitrarily small, in approximately inverse proportion to the block size, but the word error probability does not go to zero. If we apply Theorem 5.3 to this same ensemble, we get no quantitative information about the noise threshold, but we find that above the threshold, we have $\overline{P}_b^{(n)} = O(1/n)$, and $\overline{P}_W^{(n)} = O(1)$, in gratifying agreement with experiment. It is important to bear in mind, however, that: (1) the experiments are with suboptimum iterative decoding, whereas Theorem 5.3 deals with maximum-likelihood decoding; (2) Theorem 5.3 only provides an upper bound on code performance, and does not preclude the possibility that a more rapid decrease in decoder error probability is possible; and (3) experiments always deal with particular interleavers, whereas Theorem 5.3 treats the average over all interleavers.

The repeat-accumulative (RA) codes we introduced are serial turbo code ensembles with a $R_1 = 1/q$ q-fold repetition code as the outer code, and a $R_2 = 1$ recursive convolutional code, with transfer function $1/(1+D)$, as the inner code. The outer code has minimum distance $d_1 = q$. Hence, by Theorem 5.4, on all memoryless binary input channels, RA codes have word error probability approaching zero for $q \geq 3$ and bit error probability approaching zero for $q \geq 2$. For this ensemble, we can say something quantitative about the noise thresholds, since we can compute the exact spectral shape (refer to Section 3.4)

$$r(\delta) = \max_{0 \leq x \leq 1/q} \{-\frac{q-1}{q} H(qx) + (1-\delta) H(\frac{qx}{2(1-\delta)}) + \delta H(\frac{qx}{2\delta})\}.$$

Two short tables of these thresholds, on the binary symmetric channel and the Gaussian channel respectively, are given below.

q	R	γ_q	Shannon Limit
3	1/3	0.091	0.133
4	1/4	0.132	0.191
5	1/5	0.163	0.228
6	1/6	0.187	0.254
7	1/7	0.207	0.274

Table 5.1: RA ensemble thresholds on the BSC, obtained using the union bound.

q	R	γ_q (dB)	Shannon Limit (dB)
3	1/3	2.20	-0.5
4	1/4	1.93	-0.8
5	1/5	1.80	-1.0
6	1/6	1.72	-1.10
7	1/7	1.67	-1.15

Table 5.2: RA ensemble thresholds on the AWGN, obtained using the union bound.

In Table 5.1, the noise threshold γ_q is given as the largest value of the channel crossover probability for which the union bound guarantees good code performance for the corresponding RA ensemble. In Table 5.2, the threshold is given as the largest value of E_b/N_0 for which the union bound guarantees good performance. If the union bound is replaced with a more powerful tool, these thresholds can be considerably improved. For example, using the "typical pairs" method (refer to Chapter 6), we can obtain the thresholds in Table 5.3 and Table 5.4 for RA codes on the BSC and AWGN channel.

q	R	UB:γ_q	TP:γ_q	Shannon Limit
3	1/3	0.091	0.132	0.174
4	1/4	0.132	0.191	0.215
5	1/5	0.163	0.228	0.243
6	1/6	0.187	0.254	0.265
7	1/7	0.207	0.274	0.281

Table 5.3: Comparison of RA ensemble thresholds using the union bound to those obtainable using the "typical pairs" technique on the BSC.

q	R	UB:γ_q(dB)	TP:γ_q(dB)	Shannon Limit(dB)
3	1/3	2.20	0.739	-0.495
4	1/4	1.93	-0.078	-0.794
5	1/5	1.803	-0.494	-0.963
6	1/6	0.187	-0.742	-1.071
7	1/7	0.207	-0.905	-1.150

Table 5.4: Comparison of RA ensemble thresholds using the union bound to those obtainable using the "typical pairs" technique on the AWGN channel.

5.8 Discussion and Conclusions

The results in this chapter are in a sense of the culmination of previous Chapters 3 and 4. In those chapters we were interested in computing channel noise thresholds for specific code ensembles on specific channels; in this chapter we have considered general ensembles on general channels. However, we have paid a price for this generality: whereas in the earlier chapters our estimates for the noise thresholds were computed numerically, in this chapter we only prove the existence of the thresholds. To get good numerical thresholds using our methodology would require at least two improvements. First, we would have to replace the union bound with a more powerful technique; and second, we would need much more accurate estimates for the asymptotic weight spectrum $r(\delta)$ of the ensembles in question.

The next chapter addresses the first of these two problems. We will develop a method, the "typical pairs" decoding bound, which is capable of reproducing Shannon's theorem for the ensemble of random linear codes.

Chapter 6 Typical Pairs Decoding

6.1 Introduction

In this chapter, we consider the performance of ensembles of codes on general memoryless binary-input symmetric channels, including the binary symmetric channel and the additive white Gaussian channel as the most important special cases. Our particular focus is on the question as to whether or not a given ensemble is "good," in the sense of MacKay [38]. In short, an ensemble of codes is said to be good, if there is a $\gamma > 0$ such that the ensemble word error probability (with maximum-likelihood decoding) on the channel with Bhattacharya noise parameter γ approaches zero as the block length approaches infinity. The largest such γ for a given ensemble is called the (channel noise) *threshold* for the ensemble. Our main result (Theorem 6.2/6.8) is a technique for finding a lower bound on the ensemble threshold, which is based on the ensemble's weight enumerator.

Of course the classical union bound provides one way of using weight enumerators to estimate ensemble thresholds, but the estimates are poor. Gallager [24, Chapter 3] gave a variational method for upper bounding the probability of maximum-likelihood decoding error for an arbitrary binary code, or ensemble of codes (given an expression for the average weight-enumerator function) on a general class of binary-input channels. Gallager's technique, however, is quite complex, and even in the special case of the BSC it is difficult to apply to the problem of finding ensemble thresholds.[1] Other alternatives, such as Viterbi-Viterbi bound [50] and Divsalar bound [15] on the additive white Gaussian channel, do improve the estimate considerably (see Section 3.2), but they are not powerful enough to reproduce Shannon's theorem.

[1] We have been able to show that the thresholds obtained by our method are the same as the best obtainable by the Gallager methodology.

In this chapter we abandon the full maximum-likelihood decoder, and instead focus on a slightly weaker decoding algorithm, which is much easier to analyze, the *typical pairs* decoder. This technique was pioneered by Shannon [46, Theorem 11], but as far as we can tell was not used to analyze ensembles other than the ensemble of all codes (which we call the Shannon ensemble in Section 6.4.2, below) until the 1999 paper of MacKay [38], in which it was used to analyze certain ensembles of low-density-parity check codes. In brief, when presented with a received word $\mathbf{y}$, the typical pairs decoder seeks a codeword $\mathbf{x}$ such that the pair $(\mathbf{x}, \mathbf{y})$ belongs to the set T of "typical pairs." (We give a precise definition of T in Section 6.3, which follows.) In Section 6.3, we develop an upper bound on the typical-pairs decoder's error probability (Theorem 6.1) which, like the classical union bound, decouples the code's weight enumerator from the channel, but unlike the union bound, when combined with the law of large numbers, gives good estimates for code thresholds. Theorem 6.2 and Theorem 6.8 specialize the general results to the BSC and the AWGN channel.

We then apply our method to three families of binary code ensembles: (1) The Shannon ensemble, consisting of all linear codes of rate R; (2) the Gallager ensemble, consisting of (j, k) low-density parity-check codes; and (3) the ensemble of Repeat-Accumulate codes introduced by Divsalar, Jin and McEliece [17]. In the case of the Shannon ensembles, we show that our method yields thresholds identical to those implied by Shannon's theorem. Thus the typical sequence method, despite its sub-optimality, loses nothing (in terms of coding thresholds) for the Shannon ensemble (Theorem 6.7 and 6.9).

Finally, we compare our thresholds to the *iterative* thresholds for the Gallager and RA ensembles recently obtained by Richardson and Urbanke [44], in order to estimate the price paid in coding threshold for the benefits of iterative decoding. In most cases, this loss is quite small, and in the case of $j = 2$ LDPC codes, there appears to be no penalty at all.

The method described in this chapter can be readily extended to many other channel models, including channels with memory (cf. [38, Section II]).

6.2 Memoryless Binary-Input Symmetric Channels

Suppose C is an (n, k) binary linear code, with weight enumerator $(A_0, A_1, \ldots, A_n)$, i.e., C contains exactly A_h words of Hamming weight h, for $h = 1, \ldots, n$. We suppose that at the transmitter, a codeword $\mathbf{x}$ is selected at random, transmitted over a memoryless binary input channel, corrupted by a noise vector $\mathbf{z}$, and then received as $\mathbf{y} = \mathbf{x} \oplus \mathbf{z} = (x_1 \oplus z_1, \ldots, x_n \oplus z_n)$. The operation $\oplus$ is determined by the specific channel, but we always assume $\mathbf{0} \oplus \mathbf{z} = \mathbf{z}$. The input alphabet will be $X = \{0, 1\}$, or $\{+1, -1\}$ assuming BPSK modulation maps 0 to $+1$ and 1 to -1.

We further assume that the channel is symmetric, i.e., for the set of all possible channel outputs Y, there exists a *negation* operation : $Y \to Y$, denoted by $\overline{y}$, such that for any $y \in Y$, $\overline{\overline{y}} = y$, and $p(y|0) = p(\overline{y}|1)$. The following gives several examples of memoryless binary-input symmetric channels.

Example 6.1 Binary symmetric channel.
Input set $X = \{0, 1\}$. *Output set* $Y = \{0, 1\}$ *with negation operation:* $\overline{0} = 1$ *and* $\overline{1} = 0$. *Noise vector* $\mathbf{z}$ *is a sequence of i.i.d. random variables over* $GF(2)$. *Here the operation* $\oplus$ *is the conventional addition "+" over* $GF(2)$.

Example 6.2 Binary erasure channel.
Input set $X = \{-1, +1\}$, *output set* $Y = \{-1, 0, 1\}$, *with negation operation:* $\overline{y} = -y$. *Noise vector* $\mathbf{z}$ *is a sequence of i.i.d. random variables over* $\{0, 1\}$. *The operation* $\oplus$ *is ordinary multiplication. For example, suppose* $\mathbf{x} = (+1, +1, +1, -1, -1)$, $\mathbf{z} = (1, 1, 0, 0, 1)$, *then* $\mathbf{y} = (+1, +1, 0, 0, -1)$.

Example 6.3 Binary-input symmetric channel with finite output alphabet.
Input set $X = \{-1, +1\}$, *output set* $Y = \{v_1, v_2, \ldots, v_k, -v_1, \ldots, -v_k\}$, *with negation operation* $\overline{v} = -v$. *The noise vector* $\mathbf{z}$ *is a sequence of i.i.d. random variables over* Y, *and the operation* $\oplus$ *is again ordinary multiplication. For example, suppose* $\mathbf{x} = (+1, +1, +1, -1, -1)$, $\mathbf{z} = (v_1, -v_2, v_1, -v_1, v_2)$, *then* $\mathbf{y} = (v_1, -v_2, v_1, v_1, -v_2)$.

The above three examples all have discrete finite output alphabet, with Example 6.3 being the generalization of Example 6.1 and 6.2. We call them binary-input

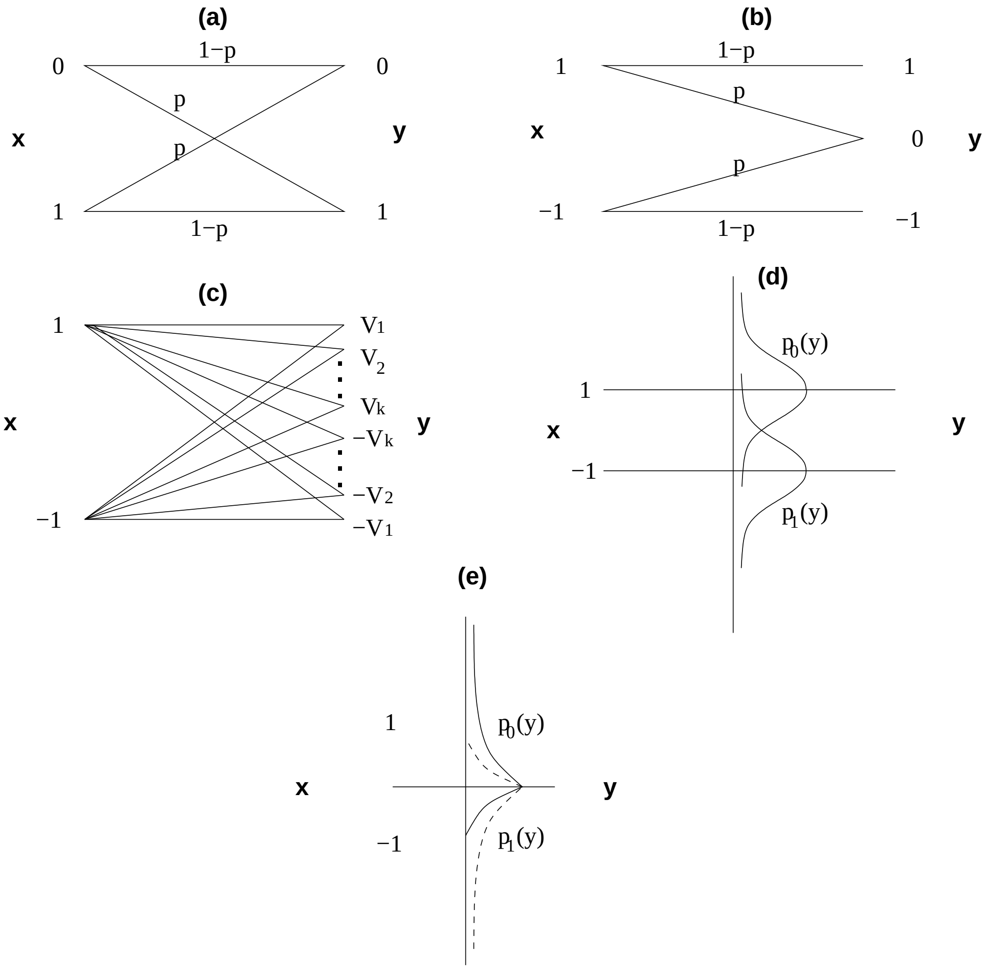

Figure 6.1: Examples of memoryless binary-input symmetric channels. (a) Binary symmetric channel. (b) Binary erasure channel. (c) Binary input finite output channel. (d) AWGN channel. (e) Rayleigh fading channel.

discrete output channels. The following are examples of binary-input *continuous output* channels.

Example 6.4 Additive white Gaussian noise channel.

$X = \{+1, -1\}$. $Y = R$ (real line) with negation operation $\overline{y} = -y$ for all $y \in R$. Noise vector $\mathbf{z}$ is a sequence of i.i.d. random Gaussian variables with mean zero and variance σ^2. The operation $\oplus$ is ordinary addition: $x \oplus z \stackrel{\Delta}{=} x + z$.

Example 6.5 Rayleigh fading channel with incoherent receiver.[24, p.65]

$X = \{+1, -1\}$. $Y = R$ (real line) with negation operation $\overline{y} = -y$ for all $y \in R$.

Noise vector $\mathbf{z}$ *is a sequence of i.i.d. random variables distributed as*

$$p_z(z) = \begin{cases} \frac{1+A}{A(2+A)} e^{-\frac{z}{A}} & z \geq 0, \\ \frac{1+A}{A(2+A)} e^{\frac{z(1+A)}{A}} & z \leq 0, \end{cases} \tag{6.1}$$

where $A = E_s/N_o$. *Operation* $\oplus$ *is ordinary multiplication.*

A general decoder tries to infer $\mathbf{x}$ based on knowledge of the code C, the garbled codeword $\mathbf{y}$, and the channel noise statistics. A specific kind of decoder, the typical set decoder, is defined in the next section.

6.3 Typical Set Decoder

Let T be a set of vectors of length n which is closed under coordinate permutations, and let $\mathbf{z} = (z_1, z_2, \ldots, z_n)$ be the noise vector, i.e., the z_i's are i.i.d. random variables over alphabet V with common density function. If we define the set T to be a set of "typical" noise vectors, then T represents the set of typical channel outputs if the zero-word is transmitted, and the set $\mathbf{x} \oplus T$ represents the set of typical channel outputs if the codeword $\mathbf{x}$ is transmitted. The *typical set* decoder, T-decoder, works as follows.

For every codeword $\mathbf{x}_i$, the ith "pseudonoise" $\mathbf{z}_i = \mathbf{y} \ominus \mathbf{x}_i$ is computed, where $\ominus$ is the inverse of $\oplus$, i.e., $\mathbf{z}_i = \mathbf{y} \ominus \mathbf{x}_i$ iff $\mathbf{y} = \mathbf{x}_i \oplus \mathbf{z}_i$. If there are no indices i for which $\mathbf{z}_i \in T$, the decoder fails. Otherwise, among those indices such that $\mathbf{z}_i \in T$, the decoder chooses the $\mathbf{x}_i$ for which $\mathbf{z}_i$ is most probable. In a sense, this decoder is "typical pairs $\bigwedge$ maximum likelihood" decoding. We will see later the "maximum likelihood" part is not essential, but it simplifies the technical proof.

In the typical set decoder, decoder errors can result if the channel output is in the typical set of more than one codeword. Since the channel is symmetric, the probability of this event is independent of the transmitted codeword $\mathbf{x}$, which we will subsequently take to be the zero-word. We are therefore interested in the quantity $Pr\{\mathbf{z} \in T \cap (\mathbf{x} \oplus T)\}$. Since T is invariant under coordinate permutations, this probability depends only on the weight of $\mathbf{x}$. Thus we define, for $h = 0, 1, \ldots, n$,

$$P_h(T) = Pr\{\mathbf{z} \in T \cap (\mathbf{x} \oplus T)\}, \tag{6.2}$$

where $\mathbf{x}$ is any vector of weight h. The quantity $P_h(T)$ is then the probability of error in a typical-set decoder in the case of a code having only two codewords separated by a Hamming distance h.

For example,

$$P_0(T) = \Pr\{\mathbf{Z} \in T\}. \tag{6.3}$$

We do not distinguish between decoder error and failure, and denote the probability of decoder error (or failure) by P_E.

Theorem 6.1 *If P_E denotes the probability that the T-decoder does not correctly identify the transmitted codeword, then*

$$P_E \leq (1 - P_0(T)) + \sum_{h=1}^{n} A_h \min(\gamma^h, P_h(T)), \tag{6.4}$$

where γ is the channel Bhattacharyya parameter (refer to Chapter 2).[2]

Proof: Let $(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{M-1})$ be an ordering of the code with $\mathbf{x}_0$ being the all-zeros word, and suppose $\mathbf{x}_0$ is transmitted. For $i = 0, 1, \ldots, M-1$, define the following events:

$$\begin{aligned}
T_i &= \{\mathbf{z}_i \in T\} \qquad (\mathbf{z}_i \text{ is typical}) \\
V_i &= \{Pr(\mathbf{z}_i) \geq Pr(\mathbf{z}_0)\} \qquad (\mathbf{z}_i \text{ is more likely than } \mathbf{z}_0) \\
S_i &= T_i \cap V_i \qquad (\mathbf{z}_i \text{ is typical and is more likely than } \mathbf{z}_0)
\end{aligned}$$

Then the T-decoder will fail only if at least one of the events $T_0', S_1, \ldots, S_{M-1}$ occurs. Thus if $\mathcal{E}$ denotes the event "T-decoder fails, given that $\mathbf{x}_0$ was transmitted,"

[2]The term γ^h in (6.4) is present for technical reasons, e.g., the proof of Theorem 6.2. Normally, it will be smaller than the term $P_h(T)$ only for very small values of h.

we have (here T_0' denotes the complement of T_0)

$$\begin{aligned}
\mathcal{E} &= T_0' \cup \left(\bigcup_{i=1}^{M-1} S_i \right) \\
&= T_0' \cup \left(T_0 \cap \bigcup_{i=1}^{M-1} S_i \right) \\
&= T_0' \cup \left(\bigcup_{i=1}^{M-1} (T_0 \cap S_i) \right). \qquad (6.5)
\end{aligned}$$

Therefore, the probability of T-decoder error, given that $\mathbf{x}_0$ was transmitted, can be upper bounded as follows:

$$\Pr\{\mathcal{E}|\mathbf{x}_0\} \leq \Pr\{T_0'|\mathbf{x}_0\} + \sum_{i=1}^{M-1} \Pr\{T_0 \cap S_i|\mathbf{x}_0\}. \qquad (6.6)$$

But $\Pr\{T_0'|\mathbf{x}_0\} = 1 - \Pr\{T_0|\mathbf{x}_0\}$, and $\Pr\{T_0|\mathbf{x}_0\} = \Pr\{\mathbf{z} \in T\} = P_0(T)$, from (6.3). Thus

$$\Pr\{T_0'|\mathbf{x}_0\} = 1 - P_0(T). \qquad (6.7)$$

Also, since $S_i = T_i \cap V_i$, it follows that

$$\Pr\{T_0 \cap S_i|\mathbf{x}_0\} \leq \min(\Pr\{V_i|\mathbf{x}_0\}, \Pr\{T_0 \cap T_i|\mathbf{x}_0\}).$$

By the familiar union bound argument [39, Theorem 7.5], we have

$$\Pr\{V_i|\mathbf{x}_0\} \leq \gamma^{h_i},$$

where h_i is the Hamming weight of $\mathbf{x}_i$.

Also note that by definition $\mathbf{z}_i = \mathbf{y} \ominus \mathbf{x}_i$, and so we have, for $i = 1, \ldots, M-1$, $\mathbf{z}_i = \mathbf{x}_0 \oplus \mathbf{z}_0 \ominus \mathbf{x}_i = \mathbf{z}_0 \ominus \mathbf{x}_i$, since $\mathbf{x}_0$ is the all-zeros word. Thus $\{z_i \in T_i\} \Leftrightarrow \{\mathbf{z}_0 \in$

$T + \mathbf{x}_i\}$, and so

$$\begin{aligned}\Pr\{T_0 \cap T_i|\mathbf{x}_0\} &= \Pr\{\mathbf{z} \in T \cap (T + \mathbf{x}_i)\}\\ &= P_{h_i}(T)\end{aligned}$$

where h_i is the Hamming weight of $\mathbf{x}_i$. Hence

$$\begin{aligned}\sum_{i=1}^{M-1} \Pr\{T_0 \cap S_i|\mathbf{x}_0\} &\leq \sum_{i=1}^{M-1} \min(\gamma^{h_i}, P_{h_i}(T)),\\ &= \sum_{h=1}^{n} A_h \min(\gamma^h, P_h(T)), \end{aligned} \tag{6.8}$$

since there are exactly A_h words of Hamming weight h in $\mathcal{C}$. Combining (6.6) with (6.7) and (6.8) gives (6.4). ■

From now on, we restrict ourselves to a slightly smaller set of memoryless binary-input symmetric channels. Although typical set decoder does work generally on any memoryless binary-input symmetric channel, in order to meaningfully define a threshold effect, we need to assume the channel transition probability is characterized by one parameter, say τ, such that the *Bhattacharya* parameter is an increasing function of τ.[3] In other words, on a *discrete output channel*, the conditional probability distribution $p(v_i|0) = f_\tau(i)$, and $p(-v_i|0) = f_\tau(-i)$, such that

$$\gamma = \sum_{i=1}^{k} 2\sqrt{f_\tau(i) f_\tau(-i)} \tag{6.9}$$

is an increasing function of τ. On a *continuous output channel*, the conditional probability density function $p(y|0) = f_\tau(y)$, such that

$$\gamma = \int_{y \geq 0} 2\sqrt{f_\tau(y) f_\tau(-y)} dy \tag{6.10}$$

in an increasing function of τ. This additional condition is inherent to many channels,

[3] A generalization is possible. If channel transition probability is characterized by two or more parameters, the threshold will be actually a high-dimension curve.

including the binary symmetric channel ($\tau = p$), the binary erasure channel ($\tau = p$), and the additive Gaussian channel ($\tau = \sigma$).

In the following text, we will treat *discrete output* channels and *continuous output* channels separately. For clarity, we use the binary symmetric channel and the additive Gaussian channel as their representatives, which also are the most important special cases, in deriving the general result.

6.4 Binary Symmetric Channel

The "right" definition of the typical set T is self-evident for the binary symmetric channel, the binary erasure channel, and the binary-input symmetric channel with a finite output alphabet. Throughout the section, we use binary symmetric channel to exemplify the derivation of the code threshold for the typical set decoder. Essentially identical steps can be applied to a more general binary-input symmetric channel with a finite output alphabet.

6.4.1 Typical pairs

Let $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_n)$ be the BSC noise vector, i.e., the Z_i's are i.i.d. random variables with common density

$$\Pr\{Z = 0\} = 1 - p, \qquad \Pr\{Z = 1\} = p. \tag{6.11}$$

Since any set T which is invariant under coordinate permutations must consist of all vectors of weight $k \in K$, where K is a subset of $\{0, 1, \ldots, n\}$, the probabilities $P_h(T)$ depend only on the set K. A short combinatorial calculation gives

$$P_h(T) = \sum_{k_1 \in K} p^{k_1}(1-p)^{n-k_1} \sum_{k_2 \in K} \binom{h}{(h+k_1-k_2)/2} \binom{n-h}{(k_1-h+k_2)/2}. \tag{6.12}$$

This is because a vector of weight k_1 has probability $p^{k_1}(1-p)^{n-k_1}$, and there are exactly $\binom{h}{(h+k_1-k_2)/2}\binom{n-h}{(k_1-h+k_2)/2}$ vectors of weight k_1, which have the property that when the first h components are complemented, i.e., the vector $\mathbf{x} = (\overbrace{11\cdots1}^{h}\overbrace{00\cdots0}^{n-h})$ is added, the resulting vector has weight k_2. Applying (6.12) to the case $h = 0$, we obtain

$$P_0(T) = \sum_{k\in K} p^k(1-p)^{n-k}\binom{n}{k}, \tag{6.13}$$

in agreement with (6.3).

In our main application (Theorem 6.2) the set T will be the "typical sequences" of length n and so will be denoted by T_n. The definition of T_n is

$$T_n = \left\{\mathbf{z} : \left|\frac{\mathrm{wt}(\mathbf{z})}{n} - p\right| \leq \epsilon_n\right\}, \tag{6.14}$$

where ϵ_n is a sequence of real numbers approaching zero more slowly than $n^{-1/2}$, i.e., $\epsilon_n\sqrt{n} \to \infty$. Then by a straightforward extension of the weak law of large numbers,

$$\lim_{n\to\infty} \Pr\{\mathbf{Z} \in T_n\} = 1. \tag{6.15}$$

Proof: Following the definition,

$$\begin{aligned}
\lim_{n\to\infty} \Pr\{\mathbf{Z}\in T_n\} &= 1 - \lim_{n\to\infty} \Pr\{\mathbf{Z}\notin T_n\} \\
&= 1 - \lim_{n\to\infty} Pr\left\{\mathbf{z} : \left|\frac{\mathrm{wt}(\mathbf{z})}{n} - p\right| \geq \epsilon_n\right\} \\
&\geq 1 - \epsilon_n^{-2} n^{-1} p(1-p) \qquad \text{by Chebyshev's inequality} \\
&= 1. \qquad \blacksquare
\end{aligned}$$

Furthermore, by defining $K_n = \{k : n(p-\epsilon_n) \leq k \leq n(p+\epsilon_n)\}$, and using the formula (6.12), it is relatively easy to prove that for any δ in the range $0 \leq \delta \leq 2p$, we have

$$\lim_{n\to\infty} -\frac{1}{n}\log P_{\delta n}(T_n) = K(\delta, p), \tag{6.16}$$

where $K(\delta, p)$ is given by the two equivalent formulas

$$K(\delta, p) = H(p) - \delta \log 2 - (1-\delta) H\left(\frac{p - \delta/2}{1-\delta}\right) \tag{6.17}$$

$$= H(\delta) - pH\left(\frac{\delta}{2p}\right) - (1-p)H\left(\frac{\delta}{2(1-p)}\right), \tag{6.18}$$

where $H(x)$ is the entropy function, i.e., $H(x) = -x \log x - (1-x)\log(1-x)$. (In fact, these formulas are true only for $\delta < 2p$; for $\delta \geq 2p$, $K(\delta, p)$ is infinite, since $P_h(T_n) = 0$ for $h > 2n(p + \epsilon_n)$.) In Figure 6.2, we have plotted the function $K(\delta, p)$ for several values of p.[4]

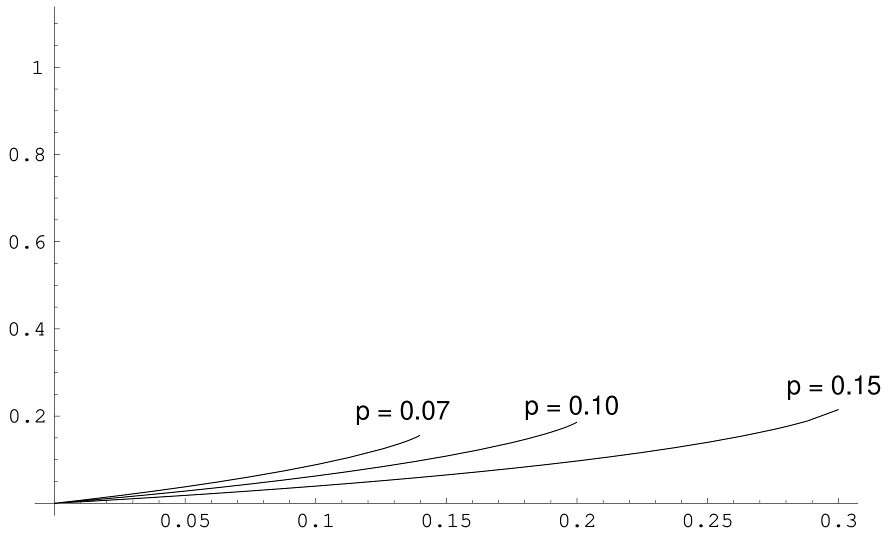

Figure 6.2: The function $K(\delta, p)$ for $p = 0.07, 0.10., 0.15$.

In fact, a closer examination of the limit in (6.16) shows that for a fixed value of p, the limit is *uniform*. That is, for a fixed p, there exists a sequence of positive numbers $\beta_n \to 0$, such that

$$\left| -\frac{1}{n} \log P_{\delta n}(T_n) - K(\delta, p) \right| < \beta_n \quad \text{for all } 0 < \delta < 2p. \tag{6.19}$$

[4]In Figure 6.2, and all the other figures in the paper, computations using logarithms use natural logarithms.

Alternatively, we can write (6.19) as

$$P_h(T_n) = e^{-n(K(\delta,p)+o(1))}, \tag{6.20}$$

where $\delta = h/n$.

On a binary symmetric channel (BSC), the typical set decoder defined in Section 6.3 can be simplified to the following: For every codeword $\mathbf{x}_i$, the ith "pseudonoise" $\mathbf{z}_i = \mathbf{y} - \mathbf{x}_i$ is computed. If there are no indices i for which $\mathbf{z}_i \in T$, the decoder fails. Otherwise, among those indices such that $\mathbf{z}_i \in T$, the decoder chooses one for which the Hamming weight $w(\mathbf{z}_i)$ is smallest.

Suppose we have a code ensemble $\mathcal{C}_n$ (as defined in Chapter 2), with average weight enumerator

$$\overline{A}_h^{(n)} = e^{n(r(\delta)+o(1))}, \tag{6.21}$$

where $\delta = h/n$.

Now we apply Theorem 6.1, using the set T_n, defined in (6.14), to a code C in ensemble $\mathcal{C}_n$:

$$P_E \leq \eta_n + \sum_{h=1}^{n} A_h(C) P_h(T_n), \tag{6.22}$$

where $\eta_n = \Pr\{T_n'\} \to 0$ by (6.15). If we average (6.22) over all codes in the ensemble $\mathcal{C}_n$, we obtain the following upper bound on $\overline{P}_E^{(n)}$, the ensemble decoder error probability:

$$\overline{P}_E^{(n)} \leq \eta_n + \sum_{h=1}^{n} \overline{A}_h^{(n)} P_h(T_n). \tag{6.23}$$

Replacing $\overline{A}_h^{(n)}$ with the right side of (6.21), and $P_h(T_n)$ with the right side of (6.20), (6.23) becomes

$$\overline{P}_E^{(n)} \leq \eta_n + \sum_{h=1}^{n} e^{-n(K(\delta,p)-r(\delta)+o(1))}. \tag{6.24}$$

It now appears that if p is chosen so that the function $K(\delta, p) - r(\delta)$ is positive for all $0 < \delta < 1$, so that the exponent in the sum in (6.24) is always negative, the ensemble word error probability $\overline{P}_E^{(n)}$ will approach zero, as $n \to \infty$. This is in fact true, provided we make the following two technical assumptions about the behavior of $\overline{A}_h^{(n)}$, for $h = o(n)$.

Assumption 1: There exists a sequence of integers d_n such that $d_n \to \infty$ and

$$\lim_{n\to\infty} \sum_{h=1}^{d_n} \overline{A}_h^{(n)} = 0. \tag{6.25}$$

(This assumption says, roughly, that the minimum distance of the ensemble is at least d_n.)

Assumption 2: There exists a sequence of real numbers $\theta_n \geq 0$ such that

$$r_n(\delta) \leq r(\delta) + \theta_n, \quad \textit{where} \quad \lim_{n\to\infty} \frac{n\theta_n}{d_n} = 0. \tag{6.26}$$

We now state our main result:

Theorem 6.2 *Suppose the code ensemble has spectral shape $r(\delta)$, and also that it satisfies Assumptions 1 and 2. Then if the crossover probability $p < 1/2$ of the channel satisfies*

$$K(\delta, p) > r(\delta) \qquad \textit{for } 0 < \delta < 2p, \tag{6.27}$$

then $\overline{P}_E^{(n)} \to 0$ as $n \to \infty$.

There is a slightly weaker version of Assumption 1 that guarantees that the ensemble *bit* error probability approaches zero:

Assumption $1'$: *There exists a sequence of integers* d_n *such that* $d_n \to \infty$ *and*

$$\lim_{n\to\infty} \sum_{h=1}^{d_n} \frac{h}{n} \overline{A}_h^{(n)} = 0.$$

The corresponding modification of Theorem 6.2 follows.

Theorem 6.3 *Suppose the code ensemble has spectral shape* $r(\delta)$, *and also that it satisfies Assumptions* $1'$ *and 2. Then if the crossover probability* $p < 1/2$ *of the channel satisfies*

$$K(\delta, p) > r(\delta) \qquad \textit{for } 0 < \delta < 2p, \tag{6.28}$$

then $\overline{P}_b^{(n)} \to 0$ *as* $n \to \infty$, *where* P_b *denotes the* T*-decoder's bit error probability.*

(A proof of Theorem 6.2 can be found in the Appendix D. The proof of Theorem 6.3 is similar and is omitted.)

In the following three subsections, we will apply Theorem 6.2 to three different ensembles of binary linear codes: (1) The Shannon ensemble, consisting of all linear codes of rate R; (2) the Gallager ensemble, consisting of (j, k) low-density parity-check codes; and (3) the ensemble of Repeat-Accumulate codes introduced by Divsalar, Jin and McEliece [17].

6.4.2 The Shannon ensemble

For the set of random linear codes of rate R, we have

$$\overline{A}_h^{(n)} = \binom{n}{h} 2^{-n(1-R)}, \tag{6.29}$$

from which it follows via a routine calculation that

$$r(\delta) = H(\delta) - (1 - R)\log 2. \tag{6.30}$$

This function is shown for $R = 1/3$ in Figure 6.3.

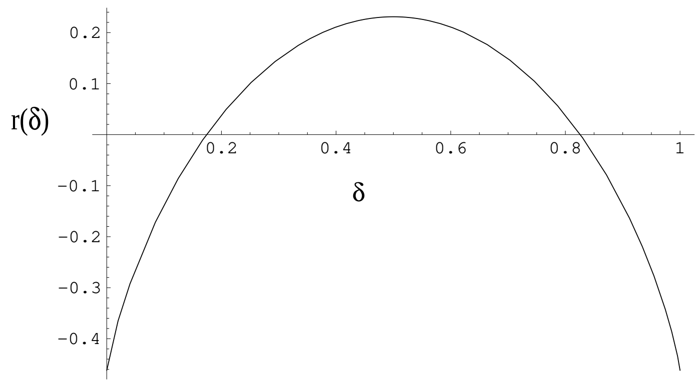

Figure 6.3: The function $r(\delta)$ for the ensemble of $R = 1/3$ random codes.

To apply Theorem 6.2 to the Shannon ensemble,[5] for a given rate R we must find the largest p such that $K(\delta, p) > H(\delta) - (1-R)\log 2$ for all $0 < \delta < 2p$.

Using (6.30) and (6.18), this inequality becomes

$$pH\left(\frac{\delta}{2p}\right) + (1-p)H\left(\frac{\delta}{2(1-p)}\right) < (1-R)\log 2. \tag{6.31}$$

The maximum of the left side of (6.31) in the range $0 < \delta < 2p$ occurs at $\delta = 2p(1-p)$, and is $H(p)$. Thus the inequality $K(\delta, p) > H(\delta) - (1-R)\log 2$ required by Theorem 6.2 becomes simply $H(p) < (1-R)\log 2$, or $H_2(p) < 1 - R$, where $H_2(p)$ is the binary entropy function. Thus we have proved

Theorem 6.4 *The ensemble of random linear codes of rate R is good on a BSC with crossover probability p if $H_2(p) < 1 - R$.*

The idea of the proof is illustrated in Figure 6.4, where we see the function $K(\delta, 0.174)$ just touching the $r(\delta)$ curve of Figure 2. This shows that the thresh-

[5]Assumptions 1 and 2 are satisified with $d_n = Kn$ for a suitable positive constant $K = K(R)$, and $\theta_n = 0$.

old for the ensemble of $R = 1/3$ linear codes is $p = 0.174$, which reflects the fact that $H_2(0.174) = 1 - 2/3$.

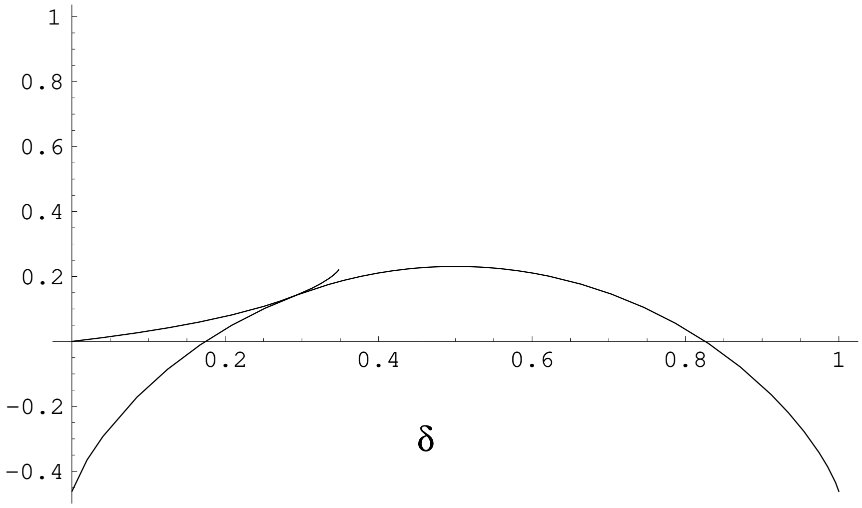

Figure 6.4: The function $r(\delta)$ for the ensemble of $R = 1/3$ linear codes, together with the function $K(\delta, p)$ for $p = 0.174$.

Of course, Theorem 6.4 is just Shannon's theorem for linear codes on the BSC. We have included it only to demonstrate that Theorem 6.2 is powerful enough to reproduce Shannon's theorem. In the next two sections we will apply it to more interesting ensembles.

6.4.3 The Gallager ensemble

In this subsection, we discuss the application of Theorem 6.2 to the ensemble of (j, k) low-density parity-check codes defined by Gallager [24].[6] In brief, every code in Gallager's (j, k) ensemble is defined by a parity-check matrix which has j ones in each column and k ones in each row. The rate of each code in the ensemble is at least $R_{j,k} = 1 - (j/k)$.

The spectral shape $r_{j,k}(\delta)$ for the (j, k) ensemble was determined by Gallager[24].

[6]There are numerous ways to define this ensemble. The definition we follow was given by Gallager [24, Section 2.2], and differs, e.g., from the ensemble analyzed by MacKay in [38, Section II].

It can be expressed in parametric form, as follows:

$$\begin{aligned}\delta_{j,k}(s) &= \frac{1}{k}\frac{\partial\mu}{\partial s}(s,k)\\ r_{j,k}(s) &= \frac{j}{k}\left(\mu(s,k) - s\frac{\partial\mu}{\partial s}(s,k) + (k-1)\log 2\right) - (j-1)H\left(\frac{1}{k}\frac{\partial\mu}{\partial s}(s,k)\right)\end{aligned}$$

where the parameter s ranges from $-\infty$ to $+\infty$, $H(x)$ is the entropy function $-(x\log x + (1-x)\log(1-x))$, and the function $\mu(s,k)$ is defined by

$$\mu(s,k) \triangleq \log\frac{(1+e^s)^k + (1-e^s)^k}{2^k}.$$

Figure 6.5 shows the function $r_{j,k}$ for $(j,k) = (3,6)$.

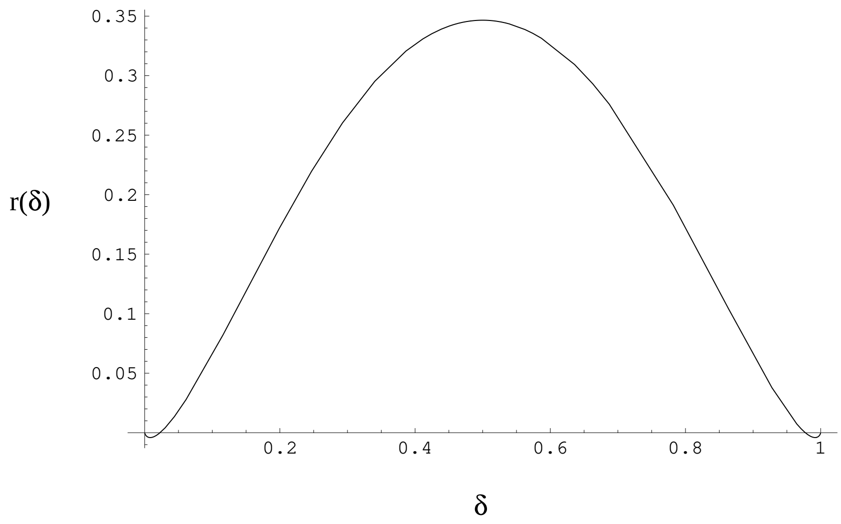

Figure 6.5: The function $r(\delta)$ for the ensemble of $(3,6)$ LDPC codes.

Given the spectral shape, it is an easy task to apply Theorem 6.2 to find the corresponding BSC ensemble thresholds.[7]

A short table of these thresholds, together with the corresponding Shannon limit, is given below.

[7]To satisfy Assumptions 1 and 2 for $j \geq 3$, we can take $d_n = Kn$ for a suitable constant $K = K(j)$, and $\theta_n = 0$. For $j = 2$, we can prove the existence of a sequence of d_n's which satisfy Assumptions $1'$ and 2 with $\theta_n = 0$, though we do not have an explicit expression for them.

(j,k)	$R_{j,k}$	$p_{j,k}$	RU limit	Shannon limit
(3,6)	1/2	0.0915	0.084	0.109
(3,5)	2/5	0.129	0.113	0.145
(4,6)	1/3	0.170	0.116	0.174
(3,4)	1/4	0.205	0.167	0.214
(2,3)	1/3	0.0670	0.0670	0.174
(2,4)	1/2	0.0286	0.0286	0.109

Table 6.1: BSC thresholds for LDPC codes.

For example, consider the "$(3,5)$" line in the table. The corresponding Gallager ensemble consists of codes which have parity-check matrices with 3 ones per column and 5 ones per row. The rate of all codes in this ensemble at least $R_{3,5} = 1-(3/5) = 2/5$. Using Theorem 6.2, it is calculated that for any BSC with crossover probability $p < 0.129$, the $(3,5)$ ensemble is good, i.e., the average word error probability of the T-decoder approaches 0, as $n \to \infty$. This should be compared to the Shannon limit for the ensemble of all linear codes of rate 2/5 (cf. Theorem 6.4), which is $p = 0.145$, which indicates the price which is paid for having the $(3,5)$ structure. Finally, we note that the Richardson-Urbanke limit [44] for the $(3,5)$ ensemble is $p = 0.113$, i.e., with belief propagation–style iterative decoding, the ensemble decoder error probability approaches 0 if and only if $p < 0.113$.

(The values $p_{j,k}$ for $(j,k) = (3,6)$, $(3,5)$, $(4,6)$, and $(3,4)$ given in the above table appear to agree with the values given by Gallager [24] in his Figure 3.5, although he gave no numerical values. However, as we mentioned above, we have been able to show that the thresholds obtained from our Theorem 6.2 are the same as the best obtainable using Gallager's methodology, so our threshold values are at least as good as Gallager's.)

We conclude this section with some remarks on the ensemble of $(2,k)$ LDPC codes. Originally dismissed by Gallager because their minimum distance is $O(\log n)$ [24, Theorem 2.5], they are nevertheless quite interesting, and are variously called "graph-theoretic," "circuit," or "cycle" codes [41, Section 5.8], [33] because of their close connection to finite undirected graphs. Using Theorem 6.3, we can show that for $p < p^*(k)$, the *bit error probability* for T-decoding of the $(2,k)$ ensemble approaches zero, where $p^*(k)$ is given by the exact formula

$$p^*(k) = \frac{1}{2}\left(1 - \sqrt{1 - \frac{1}{(k-1)^2}}\right). \tag{6.32}$$

(The ensemble word error probability does not approach zero for any $p > 0$.)

Furthermore, Wiberg [52, Example 5.1] showed that with *iterative* decoding, the ensemble of $(2,k)$ cycle codes has ensemble bit error probability aproaching zero for $p < p^*(k)$. Numerically, the Richardson-Urbanke method appears to give the same value, so it seems safe to say that (6.32) gives the exact iterative threshold for the Gallager $(2,k)$ ensemble.[8]

Finally, it was shown by Decreusefond and Zémor [14] that for an "expurgated" ensemble of $(2,k)$ cycle codes, the *exact* maximum-likelihood BSC coding threshold is equal to $p^*(k)$. Since as we have seen, the threshold for the unexpurgated ensemble is at least this good, it seems very likely that $p^*(k)$ is the exact ML threshold for the unexpurgated ensemble as well. These results strongly suggest that that for $(2,k)$ cycle codes, the iterative and maximum-likelihood thresholds are the same, and are given by the formula (6.32).

6.4.4 The ensemble of Repeat-Accumulate codes

In brief, for an integer $q \geq 2$, the ensemble of q-repeat accumulate codes consists of those codes which can be encoded by the serial concatenation of a q-ary repetition encoder, followed by a pseudorandom permutation, followed by a rate 1 code with

[8] For a survey of iterative decoding of cycle codes, see [28].

(square) generator matrix of generic shape

$$G = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

The basic combinatorial fact about the ensemble of (qk, k) RA codes is the following formula for the average number of input words of weight w which are encoded into output words of weight h [17, eq. (5.4)]:

$$\overline{A}_{w,h}^{(qk)} = \frac{\binom{k}{w}}{\binom{qk}{qw}} \binom{qk-h}{\lfloor qw/2 \rfloor} \binom{h-1}{\lceil qw/2 \rceil - 1}. \tag{6.33}$$

It follows then that if $\overline{A}_h^{(qk)}$ denotes the average number of words of weight h,

$$\overline{A}_h^{(qk)} = \sum_{w=1}^{N} \overline{A}_{w,h}^{(qk)}. \tag{6.34}$$

From (6.33) and (6.34), it can be shown that the the spectral shape $r(\delta)$ for the ensemble of q-RA codes is as follows:

$$r(\delta) = \max_{0 \le x \le 1/q} \left\{ -\frac{q-1}{q} H(qx) + (1-\delta) H(\frac{qx}{2(1-\delta)}) + \delta H(\frac{qx}{2\delta}) \right\}. \tag{6.35}$$

Figure 6.6 shows the $r(\delta)$ curve for the ensemble of $q = 3$ RA codes.[9]

Combining (6.35) with Theorem 6.2, it is a straightforward computation to obtain the thresholds in the Table 6.2.

For example, consider the $q = 3$ line of the table. It indicates that the common rate for all $q = 3$ RA codes is $R = 1/3$, and that this ensemble is good on any BSC with crossover probability $p < 0.132$. By way of comparison, the Shannon threshold

[9]To satisfy Assumptions 1 and 2 for $q \ge 3$, we can take $d_n = \log^2 n$ and $\theta_n = (K \log n)/n$ for suitable constants $K = K(q)$. For $q = 2$, we can only show the existence of a sequence d_n satisfying Assumptions $1'$ and 2 by taking $d_n = 2$ and $\theta_n = (K \log n)/n$.

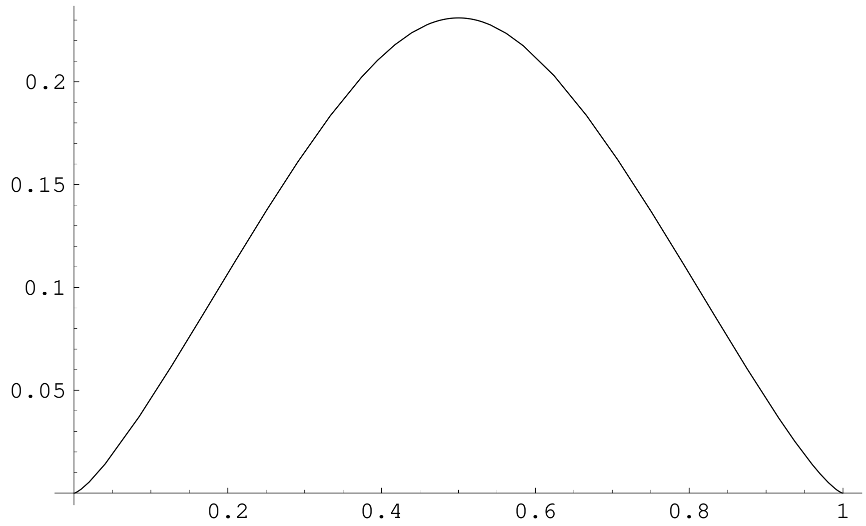

Figure 6.6: The function $r(\delta)$ for the ensemble of $R = 1/3$ RA codes.

q	R_q	p_q	RU limit	Shannon limit
2	1/2	0.029*	–	0.109
3	1/3	0.132	0.142	0.174
4	1/4	0.191	0.188	0.215
5	1/5	0.228	0.216	0.243
6	1/6	0.254	0.235	0.264
7	1/7	0.274	0.250	0.281

Table 6.2: BSC thresholds for RA codes.

for the ensemble of all rate 1/3 linear codes is seen to be $p < 0.174$. Finally, the Richardson-Urbanke iterative decoding threshold [Richardson and Urbanke, private commmunication] is $p < 0.142$. Since we can show that the T-decoding algorithm always gives the same ensemble threshold as does maximum-likelihood decoding, which must be at least as good as the iterative threshold, this apparently shows that the thresholds given in Theorem 6.2 are not always the best possible for T-decoding. A resolution of this paradox would be very welcome.

Finally we note that for the ensemble of $q = 2$ RA codes, the word error probability for T-decoding does not approach zero for any $p > 0$, but, again by using Theorem 6.3, we can show that the ensemble *bit* error probability approaches zero for $p < 0.029$.

6.5 Generalization to Discrete Output Channels

In this subsection, we generalize the results in binary symmetric channel to the general memoryless binary-input symmetric channel with finite output alphabet satisfying the one parameter condition (6.9). Suppose the output alphabet is $V_k = \{v_k, \ldots, v_2, v_1, -v_1, \ldots, -v_k\}$. A noise vector $\mathbf{z}$ is a sequence of i.i.d. random variables over V_k with probability distribution function $p(v_i) \triangleq p(v_i|+1) = f_\tau(i)$, and $p(-v_i) \triangleq p(-v_i|+1) = f_\tau(-i)$, such that $\sum_i f_\tau(i) = 1$. The *Bhattacharya* parameter of this channel is $\gamma_k = \sum_{i=1}^{k} 2\sqrt{p(v_i)p(-v_i)}$, which is an increasing function of τ. We define the typical set T_n as follows. Let $I_n(v, \mathbf{z})$ be the number of appearances of v in the sequence $\mathbf{z}$, then the typical set is

$$T_n^{(k)} = \{\mathbf{z} : |\frac{I_n(v,\mathbf{z})}{n} - p(v)| \le \epsilon_n \ \ \forall v \in V_k.\}, \tag{6.36}$$

where ϵ_n is a sequence of real numbers approaching zero more slowly than $n^{-1/2}$, i.e., $\epsilon_n\sqrt{n} \to \infty$. Then by a straightforward extension of the weak law of large numbers,

$$\lim_{n\to\infty} Pr\{\mathbf{z} \in T_n^{(k)}\} = 1. \tag{6.37}$$

Now we compute $P_h(T_n^{(k)})$ as defined in (6.2). For simplicity of notation, we define $p_i = p(v_i)$ and $p_{-i} = p(-v_i)$ for $i = 1, \ldots k$, $K_n = \{(j_1, \ldots, j_k, j_{-1}, \ldots, j_{-k}) : n(p_i - \epsilon_n) \le j_i \le n(p_i + \epsilon_n) \ \forall i$, and $\sum_{i=1}^{k} j_i + j_{-i} = n\}$, and $H_n(h) = \{(h_1, \ldots, h_k) : \sum_{i=1}^{k} h_i = h\}$. A simple combinatorial calculation gives us the following:

$$\begin{aligned} P_h(T_n^{(k)}) &= \sum_{\mathbf{j}\in K_n} \prod_{i=1}^{k} p_i^{j_i} p_{-i}^{j_{-i}} \sum_{\mathbf{j}'\in K_n} \sum_{\mathbf{h}\in H_n(h)} \frac{(n-h)!h!}{\prod_i h_i!(j_i + j_{-i} - h_i)!} \times \\ & \prod_{i=1}^{k} \binom{h_i}{(h_i - j_i + j_i')/2} \binom{j_i + j_{-i} - h_i}{(j_i + j_i' - h_i)/2}. \end{aligned} \tag{6.38}$$

It's relatively easy to prove that for any δ, we have a uniform limit:

$$\lim_{n\to\infty} -\frac{1}{n}\log P_{\delta n}(T_n^{(k)}) = K^{(k)}(\delta,\tau), \tag{6.39}$$

where $K^{(k)}(\delta,\tau)$ is given by the following formula:

$$K^{(k)}(\delta,\tau) = H(\delta) - \max_{\sum_i \delta_i=\delta} \sum_{i=1}^{k} p_i H(\frac{\delta_i}{2p_i}) + p_{-i}H(\frac{\delta_i}{2p_i}).$$

$K^{(k)}(\delta,\tau)$ can be simplified (as shown in Appendix D)

$$K^{(k)}(\delta,\tau) = H(\delta) - \sum_{i=1}^{k}(p_i H(\frac{\delta_i^*}{2p_i}) + p_{-i}H(\frac{\delta_i^*}{2p_{-i}})) \tag{6.40}$$

where $\delta = h/n$ and

$$\delta_i^* = \frac{4p_ip_{-i}}{p_i + p_{-i} + \sqrt{(p_i+p_{-i})^2 + 4(1-c)p_ip_{-i}}}.$$

The parameter $c > 0$ is determined by the condition $\sum_{i=1}^{k}\delta_i^* = \delta$. We therefore have,

$$P_h(T_n^{(k)}) = e^{-n(K^{(k)}(\delta,\tau)+o(1))}. \tag{6.41}$$

Now applying Theorem 6.1, using the above typical set $T_n^{(k)}$, to the code ensemble with weight enumerator (6.21), we obtain

$$\overline{P}_E^{(n)} \le \eta_n + \sum_{h=1}^{n} e^{-n(K^{(k)}(\delta,\tau)-r(\delta)+o(1))}. \tag{6.42}$$

Theorem 6.5 *Suppose the code ensemble has spectral shape $r(\delta)$, and also that it satisfies Assumptions 1 and 2. Then if the noise parameter τ of the channel satisfies*

$$K^{(k)}(\delta,\tau) > r(\delta), \tag{6.43}$$

then $\overline{P}_E^{(n)} \to 0$ as $n\to\infty$.

Theorem 6.6 *Suppose the code ensemble has spectral shape $r(\delta)$, and also that it satisfies Assumptions $1'$ and 2. Then if the noise parameter τ of the channel satisfies*

$$K^{(k)}(\delta, \tau) > r(\delta), \tag{6.44}$$

then $\overline{P}_b^{(n)} \to 0$ as $n \to \infty$, where P_b denotes the T-decoder's bit error probability.

(A proof of Theorem 6.5 can be found in Appendix D. The proof of Theorem 6.6 is similar and is omitted.)

We apply Theorem 6.5 only to the Shannon ensemble. The result is a generalization of Theorem 6.4.

Theorem 6.7 *The ensemble of random linear codes of rate R is good on a binary input discrete output channel if*

$$\sum_{i=1}^{k} (p_i + p_{-i}) H_2 \left(\frac{p_i}{p_i + p_{-i}} \right) < 1 - R.$$

(The proof is similar to the BSC case and hence is omitted.)

6.6 AWGN Channel

In this section, we will develop typical pairs decoding on the additive white Gaussian channel.[10] The difficulty lies in the *right* choice of typical set T. In the additive white Gaussian noise (AWGN) channel, the noise vector $\mathbf{z} = (z_1, z_2, \ldots, z_n)$ is a sequence of i.i.d. Gaussian random variables with common probability density function:

$$p_z(z) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-z^2/2\sigma^2}. \tag{6.45}$$

By the weak law of large numbers, the noise sequence $\mathbf{z}$ will exhibit certain typicality properties as n gets large. For instance, for large n, it is almost certain that the mean

[10] The exact same procedure is applicable to any binary-input symmetric channel with continuous output alphabet where the transition probability density function can be characterized by one parameter τ.

of the sequence will be close to zero and the variance will be close to σ^2. If we define the typical set to be the set of all sequences satisfying both these conditions, then the code threshold produced by Theorem 6.1 will be the same as the one derived by using Divsalar's bound [15, 1].

But it is possible to devise a stronger notion of typicality, and thus obtain better thresholds. It turns out the optimal typicality is obtained by comparing the histogram of received z's with its distribution. Roughly speaking, for any given interval $[w, w+\delta w]$, the probability that a Gaussian random variable z lies in this interval is $\int_w^{w+\delta w} p(z)dz$. Hence as n approaches infinity, the probability of the following event

$$\left|\frac{I_n[w, w+\delta w]}{n} - \int_w^{w+\delta w} p(z)dz\right| < \epsilon$$

approaches 1 for any small positive number ϵ, where $I_n[w, w+\delta w]$ is the number of z_i's in $[w, w+\delta w]$. In the following, we will make this idea rigorous.

6.6.1 Typical pairs

Given an increasing sequence $0 = s_0 < s_1 < \cdots < s_{k-1} < s_k = +\infty$, we can quantize the received real vector $\mathbf{y}$ into a vector $\mathbf{y}'$ over the finite alphabet $V_k = \{-v_k, \dots, -v_1, v_1, \dots, v_k\}$ by the following rule: *If* $s_i \leq y < s_{i+1}$, *then* $y' = v_{i+1}$; *else if* $-s_{i+1} < y \leq -s_i$, *then* $y' = -v_{i+1}$. Because of symmetry, $Pr(y|+1) = Pr(-y|-1) = p_z(y-1)$ as defined in (6.45). Therefore, $\mathbf{y}'$ has output alphabet V_k with probability distribution $Pr(v_i|+1) = Pr(-v_i|-1) = \int_{s_{i-1}}^{s_i} p_z(s-1)ds$, $Pr(-v_i|+1) = Pr(v_i|-1) = \int_{s_{i-1}}^{s_i} p_z(s+1)ds$, for any $1 \leq i \leq k$. Clearly, symmetry is preserved, in the following, we define $p(v_i) \triangleq Pr(v_i|+1)$ and $p(-v_i) \triangleq Pr(-v_i|+1)$.

Let us look at an example for the above procedure. Suppose $k = 2$, $s_1 = 1$, we thus quantized a real output vector into a vector over 4-elements $\{v_1, v_2, -v_1, -v_2\}$ as shown in Figure 6.7. For instance, assume one received vector is $\mathbf{y} = (-2, 1.4, \pi, 0.3)$, the vector after quantization is $\mathbf{y}' = (-v_2, v_2, v_2, v_1)$.

We now apply the general typical set decoding technique to the sequence $\mathbf{y}'$, instead of the original received sequence $\mathbf{y}$. If we treat $\mathbf{y}'$ as the true channel output,

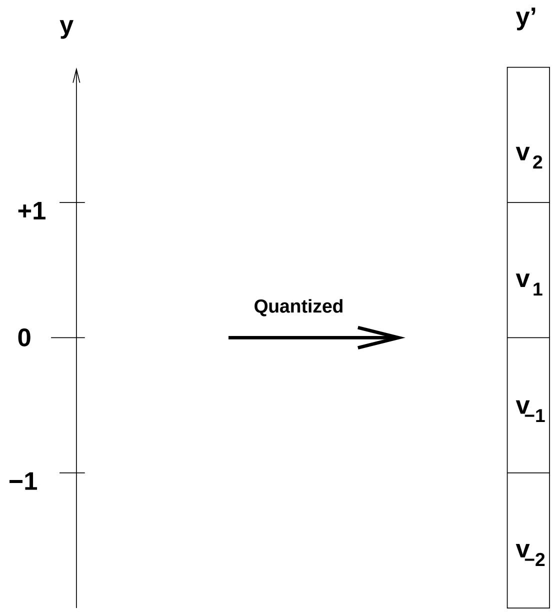

Figure 6.7: Quantizing the AWGN channel.

the channel becomes a memoryless binary-input symmetric channel with finite outputs alphabet V_k as in Example 6.3. The Bhattacharya parameter for this quantized channel is $\gamma_k = \sum_{i=1}^{k} 2\sqrt{p(v_i)p(-v_i)}$. Clearly channel transition probability is characterized by standard deviation σ, for any given sequence $\mathbf{s} = (s_0, s_1, \ldots s_k)$, except it is not clear that γ_k is an increasing function of σ.[11] The typical pairs decoding for this channel is derived in Section 6.5. We call this transformed channel as the *k-quantized channel* and the corresponding typical pairs decoding as the *k-quantized typical pairs decoding.* By increasing k, we make the typicality of an AWGN channel more stringent, thus obtaining a more accurate approximation of the original channel. Indeed, the Gaussian channel can be considered as the limit of a sequence of such channels, if the following assumption about the dividing sequences of those channels is true.

Assumption. *Suppose a sequence of channels $C_1, C_2, \ldots, C_j, \ldots$ with $\mathbf{s}_j = (s_0^{(j)}, \ldots, s_{k_j}^{(j)})$ being the interval-dividing sequence for channel C_j, we require*

[11] Generally this is not true, but we can get around this problem in the proof of Theorem 6.8.

$$\lim_{j\to\infty} s^{(j)}_{k_j-1} = \infty. \tag{6.46}$$

$$\lim_{j\to\infty} \max_{1\le i\le k_j-1} |s^{(j)}_i - s^{(j)}_{i-1}| = 0. \tag{6.47}$$

Condition (6.46) ensures the typicality test covers the entire real line, and condition (6.47) guarantees that the typicality is true for any infinitesimal interval on the real line.

Example 6.6 A sequence of channels satisfying (6.46) and (6.47).
Take $k_j = 2^{2(j-1)} + 1$, $d_j = 2^{-j+1} \to 0$, and $s^{(j)}_i = id_j$ for $i = 1, \ldots k_j - 1$. It's easy to verify that (6.46) and (6.47) are both satisfied. Figure 6.8 shows this sequence of channels. Clearly, this procedure is nothing but partitioning the real line into small intervals.

One quick observation is that given any quantized channel C with dividing sequence $\mathbf{s}$, we can build a sequence of channels starting with $\mathbf{s}$ and satisfying (6.46) and (6.47), by simply partitioning the real line in the same way as in Example 6.6.

For a sequence of channels satisfying (6.46) and (6.47), as $k \to \infty$, every interval $[s_i, s_{i+1}]$ shrinks to zero length. Therefore, roughly speaking, we have $p_i \to p_s ds = p_z(s-1)ds$, so that (6.40) becomes

$$K^{(\infty)}(\delta,\sigma) = H(\delta) - \int_0^\infty (p_s H(\frac{\delta^*_s}{2p_s}) + p_{-s} H(\frac{\delta^*_s}{2p_{-s}}))ds, \tag{6.48}$$

where

$$\delta^*_s = \frac{4p_s p_{-s}}{p_s + p_{-s} + \sqrt{(p_s+p_{-s})^2 + 4(1-c)p_s p_{-s}}}, \tag{6.49}$$

and $c > 0$ satisfies:

$$\int_0^\infty \delta^*_s ds = \delta. \tag{6.50}$$

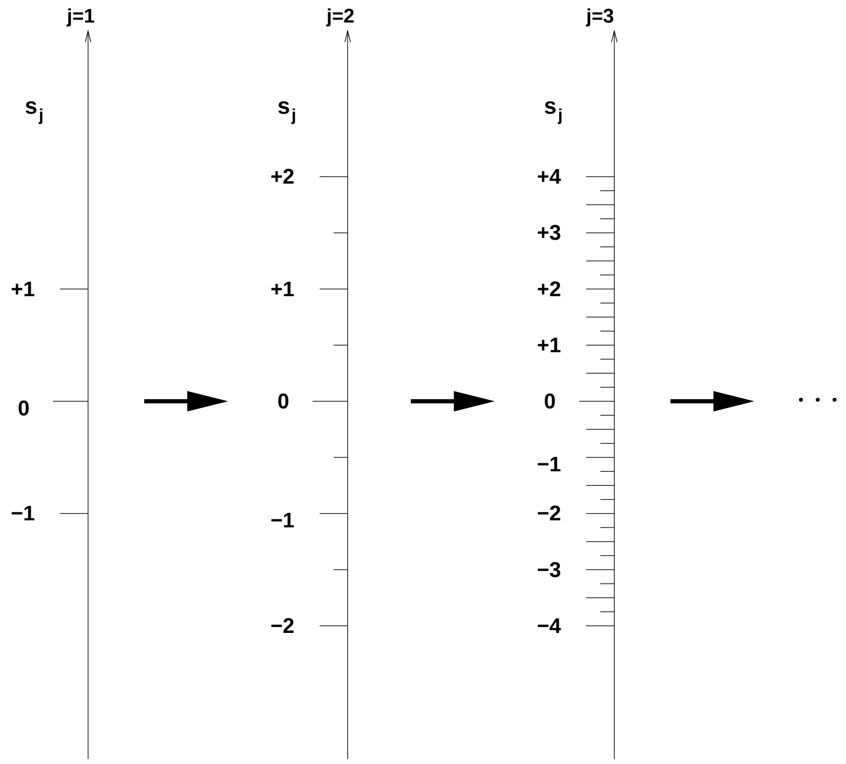

Figure 6.8: A sequence of channels satisfying (6.46) and (6.47).

We therefore have

$$\lim_{k\to\infty} P_h(T_n^{(k)}) = e^{-n(K^{(\infty)}(\delta,\sigma)+o(1))}.$$

Theorem 6.5 says for any k-state quantization, if σ is chosen so that the function $K^{(k)}(\delta,\sigma) - r(\delta)$ is positive for all $0 < \delta < 1$, the ensemble word error probability $P_E^{(n)}$ will approach zero as $n \to \infty$. In the limit case as k approaches infinity, we have the AWGN channel typical pairs decoding threshold theorem,

Theorem 6.8 *Suppose the code ensemble has spectral shape $r(\delta)$, and also that it satisfies the assumptions 1 and 2. If the standard deviation σ of the channel noise satisfies $K^{(\infty)}(\delta,\sigma) > r(\delta)$, for all $0 < \delta < 1$, then there exists a k-quantized channel C_k, the k-quantized typical pairs decoding has error probability $\overline{P}_E^{(n)} \to 0$ as $n \to \infty$.*

(The proof is included in Appendix D.)

Theorem 6.8 indicates to determine σ_0 we need to compare $K^{(\infty)}(\delta,\sigma)$ and $r(\delta)$ for all $0<\delta<1$, thus all $c>0$ in (6.49). That can be simplified a bit. Suppose $c>2$ in (6.49), then $\delta^* > 2\min(p_s, p_{-s})$. However, (6.48) is meaningful only if

$$\delta^* \leq 2\min(p_s, p_{-s}), \tag{6.51}$$

otherwise $K^{(\infty)}(\delta,\sigma)$ is defined to be ∞. Hence we only need to consider $0<c<2$ and

$$\delta \leq \int_0^\infty 2\min(p_s, p_{-s}) ds.$$

In the following sections, we are going to apply Theorem 6.8 to the ensemble of random codes, the Gallager ensemble and the repeat-accumulate ensemble codes.

6.6.2 The Shannon ensemble

To apply Theorem 6.8 to the Shannon ensemble, of which the spectral shape is given in (6.30), we must find the largest σ such that

$$K^{(\infty)}(\delta,\sigma) > H(\delta) - (1-R)\log 2 \qquad \text{for all } 0<\delta\leq 1, \tag{6.52}$$

for a given rate R. Theorem 6.7 implies (as k approaches infinity) (6.52) is equivalent to

$$\int_0^\infty (p_s+p_{-s}) H_2\left(\frac{p_s}{p_s+p_{-s}}\right) < 1-R,$$

i.e.,

$$R < 1 - \int_0^\infty (p_s+p_{-s}) H_2\left(\frac{p_s}{p_s+p_{-s}}\right). \tag{6.53}$$

Working out the integral, the right-hand side of (6.53) becomes

$$C(\sigma) \triangleq \frac{\log_2 e}{\sigma^2} - \int_{-\infty}^{\infty} \frac{e^{-s^2/2}}{\sqrt{2\pi}} \log_2 \cosh(1/\sigma^2 + s/\sigma) ds.$$

Thus we have proved,

Theorem 6.9 *The ensemble of random linear codes of rate R is good on an AWGN channel with noise standard deviation σ if $R < C(\sigma)$.*

Since $C(\sigma)$ is the capacity for the binary input AWGN channel [39, p.103], Theorem 6.9 is just Shannon's theorem for linear codes on the AWGN channel. This demonstrates that Theorem 6.8 is powerful enough to reproduce Shannon's theorem.

6.6.3 The Gallager ensemble

In this subsection we apply Theorem 6.8 to the Gallager ensemble. Unlike on BSC, thresholds of this ensemble on AWGN were not studied in [24]. A short table of these thresholds, together with the correponding Shannon limit, the thresholds derived by the Divsalar bound [15], is given below.

(j,k)	$R_{j,k}$	snr_D	$snr_{j,k}$	Shannon limit	RU limit
(3,6)	1/2	0.793	0.673	0.187	1.110
(4,6)	1/3	-0.381	-0.423	-0.495	1.674
(3,4)	1/4	-0.488	-0.510	-0.794	1.003

Table 6.3: AWGN thresholds for LDPC codes.

6.6.4 The ensemble of Repeat-Accumulate codes

In this subsection we apply Theorem 6.8 to the ensemble of repeat-accumulative codes. Given the spectral shape (6.35), it is a straightforward calculation to obtain the thresholds in the following table. Note that we use the parameter SNR instead of σ ($SNR = 1/(2R\sigma^2)$).

q	R_q	SNR_D	SNR_q	Shannon limit	RU limit
3	1/3	0.792	0.739	-0.495	0.479
4	1/4	-0.052	-0.078	-0.794	0.106
5	1/5	-0.480	-0.494	-0.963	0.044
6	1/6	-0.734	-0.742	-1.071	0.085
7	1/7	-0.900	-0.905	-1.150	0.168

Table 6.4: AWGN thresholds for RA codes.

6.7 Generalization to Continuous Output Channels

A general continuous output channel doesn't differ from the AWGN channel in our derivations, given its own one-parameter-characterized probability density function $p(\tau)$. One important result is that the equivalent statement of Theorem 6.9 holds, i.e., typical pairs decoding is powerful enough to prove Shannon's theorem on any of those channels.

Theorem 6.10 *The ensemble of random linear codes of rate R is good on a continous channel with capacity $C(\tau)$ if $R < C(\tau)$, where τ is the channel noise parameter.*

Proof: Applying Theorem 6.7, as k approaches infinity, we have reliable communication with a typical set decoder as long as

$$\int_0^\infty (p_s + p_{-s}) H_2\left(\frac{p_s}{p_s + p_{-s}}\right) < 1 - R, \tag{6.54}$$

where $p_s = p_z(s-1)ds$, $p_{-s} = p_z(-s-1)ds$ and $H_2(\,)$ is the binary entropy function. On the other hand, the capacity of the channel is given by (in bits)

$$\begin{aligned} C = \max_{p(x)} I(X;Y) &= \max_{p(x)} E_{p(x,y)} \log_2 \frac{p(y|x)}{p(y)} \\ &= \max_{p(x)} E_{p(x,y)} \log_2 \frac{p(y|x)}{p(x=+1)p(y|+1) + p(x=-1)p(y|-1)}. \end{aligned}$$

Because the channel is symmetric, the capacity is achieved when the input signals are equal probable, i.e., $p(x=+1)=p(x=-1)=1/2$. Hence the capacity is

$$\begin{aligned}
C &= \int_{-\infty}^{+\infty} p(+1,y)\log_2\frac{2p(y|+1)}{p(y|+1)+p(y|-1)} + p(-1,y)\log\frac{2p(y|-1)}{p(y|+1)+p(y|-1)}dy \\
&= \int_{0}^{+\infty} p(y|+1)\log_2\frac{2p(y|+1)}{p(y|+1)+p(y|-1)} + p(y|-1)\log\frac{2p(y|-1)}{p(y|+1)+p(y|-1)}dy \\
&= 1-\int_0^{\infty}(p_y+p_{-y})H_2\left(\frac{p_y}{p_y+p_{-y}}\right). \qquad (6.55)
\end{aligned}$$

Comparing (6.55) with (6.54), we conclude that as long as

$$R < C(\tau),$$

we have reliable communication using random codes with a typical set decoder. ■

Chapter 7 Irregular Repeat-Accumulate Codes

7.1 Introduction

With the hindsight provided by the past seven years of research in turbo-codes and low-density parity-check codes, one is tempted to propose the following problem as the final problem for channel coding researchers: *For a given channel, find an ensemble of codes with (1) a linear-time encoding algorithm, and (2) which can be decoded reliably in linear time at rates arbitrarily close to channel capacity.* For turbo-codes, both parallel and serial, (1) holds, but according to the recent work by Divsalar, Dolinar, and Pollara [19], on the AWGN channel there appears to be a gap, albeit usually not a large one, between channel capacity and the iterative decoding thresholds for any turbo ensemble. For LDPC codes, the natural encoding algorithm is quadratic in the block length, and from the work of Richardson and Urbanke [44] we know that for regular LDPC codes, on the binary symmetric and AWGN channels there is a gap between capacity and the iterative decoding thresholds. On the positive side, however, Luby, Shokrollahi et al. [36, 37, 47], have established the remarkable fact that on the binary erasure channel *irregular* LDPC codes satisfy (2). Recent work by Richardson, Shokrollahi and Urbanke [43] shows that on the AWGN channel, irregular LDPC codes are markedly better than regular ones, but whether or not they can reach capacity is not yet known. In summary, as yet there is no known noisy channel for which the final problem has been solved, although researchers are very close on the AWGN channel and extremely close on the binary erasure channel.

In this chapter, we will introduce a class of codes called *irregular repeat-accumulate* codes, which generalizes the repeat-accumulate codes of [17]. After defining the codes in Section 7.2, and observing that they have a simple linear-time encoding algorithm, in Section 7.3 we will prove rigorously that IRA codes solve the final problem for the binary erasure channel. In Section 7.4, we will discuss, less rigorously, the performance of IRA codes on the AWGN channel, and show that their performance is remarkably good. In Section 7.5 we will show their performance on some non-standard channels.

7.2 Definition of IRA Codes

Figure 7.1 shows a Tanner graph of an irregular RA code with parameters $(f_1, \dots, f_J; a)$, where $f_i \geq 0$, $\sum_i f_i = 1$ and a is a positive integer. The Tanner graph is a bipartite graph with two kinds of nodes: variable nodes (open circles) and check nodes (filled circles). There are k variable nodes on the left, called information nodes; there are $r = (k \sum_i i f_i)/a$ check nodes; and there are r variable nodes on the right, called parity nodes. Each information node is connected to a number of check nodes: the fraction of information nodes connected to exactly i check nodes is f_i. Each check node is connected to exactly a information nodes. These connections can be made in many ways as indicated in Figure 7.1 by the "random permutation" of the ra edges joining information nodes and check nodes. The check nodes are connected to the parity nodes in the simple zigzag pattern shown in the figure.

If the "random permutation" in Figure 7.1 is fixed, the Tanner graph represents a binary linear code with k information bits $(u_1, \dots, u_k)$ and r parity bits $(x_1, \dots, x_r)$, as follows. Each of the information bits is associated with one of the information nodes; and each of the parity bits is associated with one of the parity nodes. The value of a parity bit is determined uniquely by the condition that the mod-2 sum of the values of the variable nodes connected to each of the check bits is zero. To see this, let us conventionally set $x_0 = 0$. Then if the values of the bits on the ra edges coming out of the information nodes are $(v_1, \dots, v_{ra})$, we have the recursive formula

$$x_j = x_{j-1} + \sum_{i=1}^{a} v_{(j-1)a+i} \tag{7.1}$$

for $j = 1, 2, \dots, r$. This is in effect the encoding algorithm, and so if a is fixed and

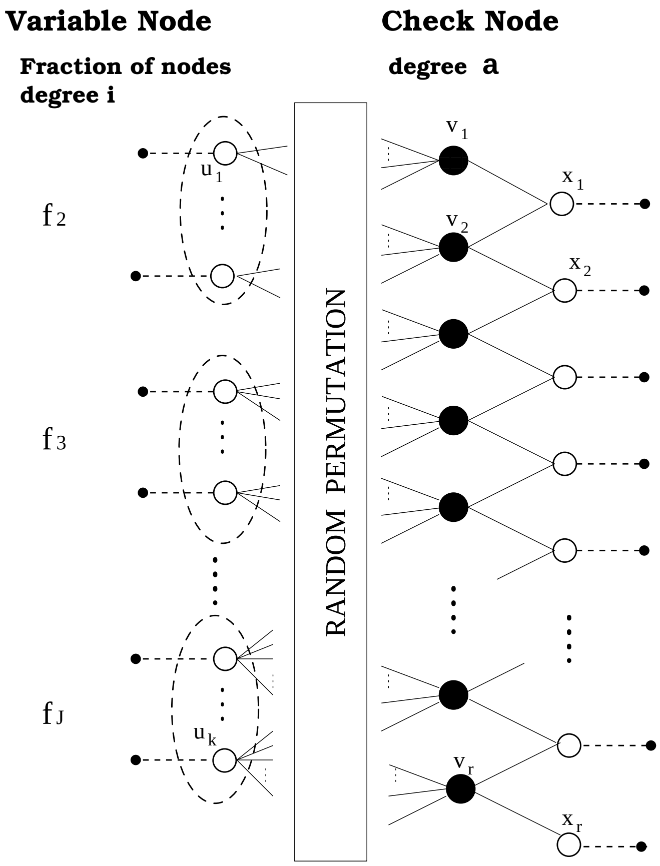

Figure 7.1: Tanner graph for $(f_1, \ldots, f_J; a)$ IRA Code.

$n \to \infty$, the encoding complexity is $O(n)$.

There are two versions of the IRA code in Figure 7.1: the *nonsystematic* and the *systematic* versions. The nonsystematic version is an (r, k) code, in which the codeword corresponding to the information bits $(u_1, \ldots, u_k)$ is $(x_1, \ldots, x_r)$. The systematic version is a $(k + r, k)$ code, in which the codeword is

$$(u_1, \ldots, u_k; x_1, \ldots, x_r).$$

The rate of the *nonsystematic* $(f_1, \ldots, f_j; a)$ ensemble is easily seen to be

$$R_{\text{nsys}} = \frac{a}{\sum_i i f_i}, \tag{7.2}$$

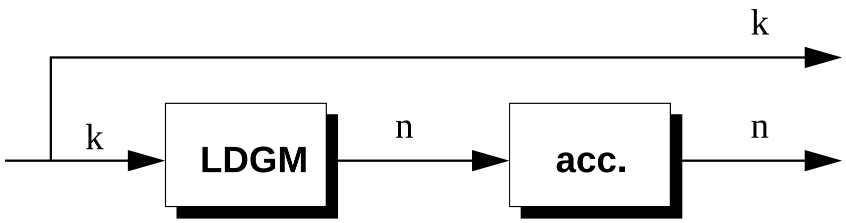

Figure 7.2: IRA code as a serial turbo code.

whereas for the systematic ensemble the rate is

$$R_{\text{sys}} = \frac{a}{a + \sum_i i f_i}. \tag{7.3}$$

For example, the original RA codes are nonsystematic IRA codes with $a = 1$ and exactly one f_i equal to 1, say $f_q = 1$, and the rest zero, in which case (7.2) simplifies to $R = 1/q$. (However, in this chapter we will be concerned almost exclusively with systematic IRA codes.)

A closer look of the structure of IRA codes reveals that it is a serial concatenation of two familiar codes, low-density generator-matrix code and accumulate code. This is shown in Figure 7.2.

As the name implies, low-density generator-matrix (LDGM) codes are codes whose generator matrices are sparse. Like low-density parity-check codes, LDGM codes can also be realized on a Tanner graph. In [11] it has been shown LDGM codes are performing comparable to LDPC codes under the iterative decoding method.

In the serial turbo structure of IRA codes, the usual interleaver is excluded since randomness is already present in the structure of LDGM code.

In an iterative sum-product message-passing decoding algorithm, all messages are assumed to be log-likelihood ratios, i.e., of the form $m = \log(p(0)/p(1))$. The outgoing message from a variable node u to a check node v represents information about u, and a message from a check node u to a variable node v represents information about u. Initially, messages are sent from variable nodes which represent transmitted symbols.

The outgoing message from a node u to a node v depends on the incoming messages from all neighbors w of u except v. If v is a variable node, the outgoing message is

$$m(u \to v) = \sum_{w \neq v} m(w \to u) + m_0(u), \tag{7.4}$$

where $m_0(u)$ is log-likelihood message associated with the channel observation of the codeword bit u. (If u is not a codeword node, this term is absent.) If v is a check node the corresponding formula is [25]

$$\tanh \frac{m(u \to v)}{2} = \prod_{w \neq v} \tanh \frac{m(w \to u)}{2}. \tag{7.5}$$

7.3 IRA Codes on the Binary Erasure Channel

The sum-product algorithm defined equations (7.4) and (7.5) simplifies considerably on the Binary Erasure Channel (BEC). The BEC is a binary input channel with three output symbols, 0, 1 and "erasure." The input symbol is received as an erasure with probability p and is received correctly with probability $1 - p$. It is important to note that no errors are ever made on this channel.

It is not difficult to see that the messages defined in (7.4) and (7.5) can assume only three values on the BEC, viz. $+\infty$, $-\infty$ or 0, corresponding to a variable value 0, 1 or "unknown." No errors can occur during the running of the algorithm; if a message is $\pm\infty$, the corresponding variable is guaranteed to be 0 or 1, respectively. The operations at the nodes in the graph given by equations (7.4) and (7.5) can be stated much more simply and intuitively in this case. At a variable node, the outgoing message is equal to any non-erasure incoming message, or an erasure if any incoming messages is an erasure. At a check node, the outgoing message is an erasure if any incoming message is an erasure, and otherwise is the binary sum of all incoming messages.

7.3.1 Notation

In this section and the next, it will be convenient to use a slightly different representation for an IRA code than the one used in Section 7.2. Firstly, we will begin with the assumption that the degrees of both the information nodes and the check nodes are non-constant, though we will soon restrict attention to the "right-regular" case, in which the check nodes have constant degree.

Secondly, let λ_i be the fraction of *edges* between the information and the check nodes that are adjacent to an information node of degree i, and let ρ_i be the fraction of such *edges* that are adjacent to a check node of degree $i+2$ (i.e., one which is adjacent to i information nodes). We will use these edge fractions λ_i and ρ_i to represent the IRA code rather than the corresponding node fractions. We define $\lambda(x) = \sum_i \lambda_i x^{i-1}$ and $\rho(x) = \sum_i \rho_i x^{i-1}$ to be the generating functions of these sequences. The pair (λ, ρ) is called a *degree distribution*. It is quite easy to convert between the two representations. We demonstrate the conversion with the information node degrees. Let the f_i's be as defined in Section 7.2 and let $L(x) = \sum_i f_i x^i$. Then we have

$$f_i = \frac{\lambda_i / i}{\sum_j \lambda_j / j}, \qquad \text{and} \tag{7.6}$$

$$L(x) = \int_0^x \lambda(t)dt \Big/ \int_0^1 \lambda(t)dt. \tag{7.7}$$

The rate of the systematic IRA code (we shall be dealing only with these) given by this degree distribution is given by

$$\text{Rate} = \left(1 + \frac{\sum_j \rho_j / j}{\sum_j \lambda_j / j}\right)^{-1}. \tag{7.8}$$

(This is an easy exercise. For a proof, see [47].)

7.3.2 Fixed point analysis of iterative decoding

In [44], it was shown that if for a code ensemble, the probability of the *depth-l neighborhood* of an edge (in the Tanner graph) being cycle-free goes to 1 as the length of the code goes to infinity (we will call this condition the *cycle-free condition*), the *density evolution* gives an accurate estimate of the bit error rate after l iterations, again as the length of the code goes to infinity. In density evolution, we evolve the probability density of the messages being passed according to the operations being performed on them, assuming that all incoming messages are independent. The cycle-free condition does indeed hold for IRA codes. The proof of this fact is almost exactly the same as in the irregular LDPC case, which was done in [44].

Now, in the case of the erasure channel, we have seen that the messages are only of three types, so in effect we have a discrete density function, and the probability of error is merely the probability of erasure. With this in mind, we will now study the evolution of the erasure probability, and derive conditions which guarantee that it goes to zero as the number of iterations goes to infinity. Under these conditions iterative decoding will be successful in the sense of [44], i.e., it will achieve arbitrarily small BERs, given enough iterations and long enough codes.

Let p be the channel probability of erasure. We will iterate the probability of erasure along the edges of the graph during the course of the algorithm. Let x_0 be the probability of erasure on an edge from an information node to a check node, x_1 the probability of erasure on an edge from a check node to a non-information variable node (the rightmost column in Figure 7.1), x_2 the probability of erasure on an edge from a non-information variable node to a check node, and x_3 the probability of erasure on an edge from a check node to an information node. The prior probability of erasure on the message bits is p.

We now assume that we are in a fixed point of the decoding algorithm and solve for x_0. We get the following equations:

$$
\begin{aligned}
x_1 &= 1-(1-x_2)R(1-x_0), \\
x_2 &= px_1, \\
x_3 &= 1-(1-x_2)^2\rho(1-x_0), \\
x_0 &= p\lambda(x_3),
\end{aligned}
$$

where the $R(x)$ is the polynomial in which the coefficient of x^i denotes the fraction of parity check *nodes* of degree i. $R(x)$ is given by (cf. eq. (7.7))

$$R(x) = \frac{\int_0^x \rho(t)dt}{\int_0^1 \rho(t)dt}. \tag{7.9}$$

We eliminate x_1 from the first two of these equations to get x_2 in terms of x_0 and then keep substituting forwards to get an equation purely in x_0, henceforth denoted by x. We thus get the following equation for a fixed point of iterative decoding:

$$p\lambda\left(1 - \left[\frac{1-p}{1-pR(1-x)}\right]^2 \rho(1-x)\right) = x. \tag{7.10}$$

If this equation has no solution in the interval $(0, 1]$, then iterative decoding converges to probability of erasure zero. Therefore, if we have

$$p\lambda\left(1 - \left[\frac{1-p}{1-pR(1-x)}\right]^2 \rho(1-x)\right) < x, \quad \forall x \neq 0, \tag{7.11}$$

iterative decoding is successful.

7.3.3 Capacity achieving sequences of degree distributions

We will now derive sequences of degree distributions that can be shown to achieve channel capacity. First, we restrict attention to the case $\rho(x) = x^{a-1}$ for some $a \geq 1$, since it turns out that we can achieve capacity even with this restriction. In this case, $R(x) = x^a$, and the condition for convergence to zero BER now becomes

$$p\lambda\left(1 - \left[\frac{1-p}{1-p(1-x)^a}\right]^2 (1-x)^{a-1}\right) < x, \quad \forall x \neq 0. \tag{7.12}$$

We now make the following new definitions

$$f_p(x) \triangleq 1 - \left[\frac{1-p}{1-p(1-x)^a}\right]^2 (1-x)^{a-1}. \tag{7.13}$$

$$h_p(x) \triangleq 1 - \left[\frac{1-p}{1-p(1-x)^a}\right]^2 (1-x)^a \tag{7.14}$$

$$g_p(x) \triangleq h_p^{-1}(x). \tag{7.15}$$

Notice that $f_p(x)$, $h_p(x)$ and $g_p(x)$ are all monotonic functions in $[0,1]$ and attain the values 0 at 0 and 1 at 1. In addition, $h_p(x)$ can be inverted by hand (by making the substitution $(1-x)^a = y$ and it can be shown that $g_p(x)$ has a power series expansion around 0 with non-negative coefficients. Let this expansion be $g_p(x) = \sum_i g_{p,i}x^i$.

Now, the condition (7.12) can be rewritten as

$$p\lambda(f_p(x)) < x, \quad \forall x \neq 0 \tag{7.16}$$

which can be rewritten as

$$\lambda(x) < \frac{f_p^{-1}(x)}{p}. \tag{7.17}$$

We make the following choice of $\lambda(x)$:

$$\lambda(x) = \frac{1}{p}\left(\sum_{i=1}^{N-1} g_{p,i}x^i + \epsilon x^N\right), \tag{7.18}$$

where $0 < \epsilon < g_{p,N}$ and $\sum_{i=1}^{N-1} g_{p,i} + \epsilon = p$. Such a choice of N and ϵ exists and is unique since the $g_{p,i}$'s are non-negative and $\sum_{i=1}^{\infty} g_{p,i} = g_p(1) = 1$. For this choice of $\lambda(x)$, we have

$$p\lambda(x) < g_p(x) = h_p^{-1}(x) < f_p^{-1}(x) \quad \forall x \neq 0 \tag{7.19}$$

where the last inequality follows because $f_p(x) < h_p(x) \;\; \forall x \neq 0$.

Thus, the condition (7.17) for BER going to zero is satisfied and the degree distributions we have thus defined yield codes with thresholds that are greater than or equal to p. We now wish to compute the rate of these codes in the asymptotic of $a \to \infty$ to show that they achieve channel capacity. Now, the rate of the code is given by eq. (7.8) which simplifies to $(1 + (a\sum_i \lambda_i/i)^{-1})^{-1}$ in the right-regular case. Now,

$$\lim_{a\to\infty} a\sum_i \frac{\lambda_i}{i} = \lim_{a\to\infty} a\left(\sum_{i=1}^{N-1} \frac{g_{p,i}}{i} + \frac{\epsilon}{N}\right). \tag{7.20}$$

We also have

$$\lim_{a\to\infty} a\sum_{i=N}^{\infty} \frac{g_{p,i}}{i} \leq \lim_{a\to\infty} \frac{a}{N}\sum_{i=N}^{\infty} g_{p,i} \leq \lim_{a\to\infty} \frac{a}{N} = 0, \tag{7.21}$$

where the last equality is a property of the function $g_p(x)$ and is also proved by manual inversion of $h_p(x)$. We now have

$$\begin{aligned}
\lim_{a\to\infty} a\sum_i \frac{\lambda_i}{i} &= \lim_{a\to\infty} a\sum_{i=1}^{\infty} \frac{g_{p,i}}{i} \\
&= \lim_{a\to\infty} a\int_0^1 g_p(x)dx \\
&= a\left(1 - \int_0^1 h_p(x)dx\right) \\
&= a\int_0^1 \left(\frac{1-p}{1-px^a}\right)^2 x^a dx.
\end{aligned}$$

The integrand on the right can be expanded in a power series with non-negative coefficients, with the first non-zero coefficient being that of x^a. Keeping in mind that we are integrating this power series, it is easy to see that

$$\begin{aligned}
\frac{a}{a+1}\int_0^1 \left(\frac{1-p}{1-px^a}\right)^2 x^{a-1}dx &< 1 - \int_0^1 h_p(x)dx \\
&< \int_0^1 \left(\frac{1-p}{1-px^a}\right)^2 x^{a-1}dx.
\end{aligned} \tag{7.22}$$

Both bounds in the above equation can be computed easily and both tend to $(1-p)/p$ in the limit of large a. Plugging this result into the formula for the rate, we finally get that the rate tends to $1-p$ in the limit of large a, which is indeed the capacity of the BEC.

Thus the sequence of degree distributions given in eq. (7.18) does indeed achieve channel capacity.

7.3.4 Some numerical results

We have seen that the condition for BER going to zero at a channel erasure probability of p is $p\lambda(x) < f_p^{-1}(x)\ \forall x \neq 0$. We later enforced a stronger condition, namely $p\lambda(x) < h_p^{-1}(x) = g_p(x)\ \forall x \neq 0$ and derived capacity-achieving degree sequences satisfying this condition. The reason we needed to enforce the stronger condition was that $h_p^{-1}(x) = g_p(x)$ has non-negative power-series coefficients, while the same cannot be said for $f_p^{-1}(x)$. However, from (7.22) we see that enforcing this stronger condition costs us to the extent of a factor of $1 - a/(a+1) = 1/(a+1)$ in the rate which is very large for values of a that are of interest, and therefore the resulting codes are not very good.

If, however, $f_p^{-1}(x)$ were to have non-negative power series coefficients, then we could use it to define a degree distribution and we would no longer lose this factor of $1/(a+1)$. We have found through direct numerical computation in all cases that we tried, that enough terms in the beginning of this power series are non-negative to enable us to define $\lambda(x)$ by an equation analogous to eq. (7.18), replacing $g_p(x)$ by $f_p^{-1}(x)$. Of course, the resulting code is not theoretically bound to have a threshold $\geq p$, but again numerical computation showed that the threshold is either equal to or very marginally less than p.

This design turns out to yield very powerful codes, in particular codes whose performance is in every way comparable to the irregular LDPC codes listed in [47] as far as decoding performance is concerned. The performance of some of these distributions is listed in Table 7.1. The threshold values p are the same as those in [47] for corresponding values of a (IRA codes with right degree $a+2$ should be compared to irregular LDPC codes with right degree a, so that the decoding complexity is about the same), so as to make comparison easy. The codes listed in [47] were shown to have certain optimality properties with respect to the tradeoff between $1 - \delta/(1-R)$ (distance from capacity) and a (decoding complexity), so it is very heartening to note that the codes we have designed are comparable to these.

a	δ	N	$1-R$	$\delta/(1-R)$
4	0.20000	1	0.333333	0.6000
5	0.23611	3	0.317101	0.7448
6	0.28994	6	0.329412	0.8802
7	0.31551	11	0.336876	0.9366
8	0.32024	16	0.333850	0.9592
9	0.32558	26	0.334074	0.9744
4	0.48090	13	0.502141	0.9577
5	0.49287	28	0.502225	0.9814

Table 7.1: Performance of some codes designed using the procedure described in Section 7.3.4. at rates closes to 2/3 and 1/2. δ is the code threshold, N the number of terms in $\lambda(x)$, and R is the rate of the code.

We end this section with a brief discussion of the case $a = 1$. In this case, it turns out that $f_p^{-1}(x)$ does indeed have non-negative power-series coefficients. The resulting degree sequences yield codes that are better than conventional RA codes at small rates. An entirely similar exercise can be carried out for the case of non-systematic RA codes with $a = 1$ and the codes resulting in this case are significantly better than conventional RA codes for most rates. However, non-systematic RA codes turn out to be useless for higher values of a, as can be seen by manually following the decoding algorithm for one iteration, which immediately shows that decoding does not proceed at all. This is the reason that all the preceding analysis was performed for systematic RA codes.

7.4 IRA Codes on the AWGN Channel

In this section, we will consider the behavior of IRA codes on the AWGN channel. Here there are only two possible inputs, 0 and 1, but the output alphabet is the set of real numbers: if the x is the input, then the output is $y = (-1)^x + z$, where z is a mean zero, variance σ^2 Gaussian random variable. For a given noise variance σ^2, our objective will be to find a left degree sequence $\lambda(x)$ such that the ensemble iterative decoding error probability approaches zero, while the rate is as large as possible. Unlike the BEC, where we deal only with probabilities, in the case of the AWGN channel we must deal with probability densities. This complicates the analysis, and forces us to resort to approximate design methods.

7.4.1 Gaussian approximation

Wiberg has shown [52] that the messages passed in iterative decoding on the AWGN channel can be well approximated by Gaussian random variables, provided the messages are in log-likelihood ratio forms. In [12], this approximation was used to design good LDPC codes for the AWGN channel.

In this subsection, we use this Gaussian approximation to design good IRA codes for the AWGN channel. Specifically, we approximate the messages from the check nodes to the variable nodes (both information and parity) as Gaussian at every iteration. For a variable node, if all the incoming messages are Gaussian distributed, then all the outgoing messages are also Gaussian because of (7.4). A Gaussian distribution $f(x)$ is called *consistent* if $f(x) = f(-x)e^x$ for $\forall x \leq 0$. The consistency condition implies that the mean and variance satisfies $\sigma^2 = 2\mu$. For the sum-product algorithm, it has been shown [43] that consistency is preserved at message updates of both the variable and check nodes. Thus if we assume Gaussian messages, and require consistency, we only need to keep track of the means. To the end, we define a *consistent Gaussian density* with mean μ to be

$$G_\mu(z) = \frac{1}{\sqrt{4\pi\mu}} e^{-(z-\mu)^2/4\mu}. \tag{7.23}$$

The expected value of $\tanh\frac{z}{2}$ for a consistent Gaussian distributed random variable z with mean μ is then

$$E[\tanh\frac{z}{2}] = \int_{-\infty}^{+\infty} G_\mu(z)\tanh\frac{z}{2}dz \triangleq \phi(\mu). \tag{7.24}$$

It is easy to see that $\phi(u)$ is a monotonic increasing function of u; we denote its inverse function by $\phi^{(-1)}(y)$. Let $\mu_L^{(l)}$ and $\mu_R^{(l)}$ be the means of the message from check nodes to information nodes and parity nodes respectively at lth iteration. We want to obtain expressions for $\mu_L^{(l+1)}$ and $\mu_R^{(l+1)}$ in terms of $\mu_L^{(l)}$ and $\mu_R^{(l)}$. A message from a degree-i information node to a check node at lth iteration is Gaussian with mean $(i-1)\mu_L^{(l)}+\mu_o$, where μ_o is the mean of message m_o in (7.4). Hence if v_L denotes the message on a randomly selected edge from an information node to a check node, the density of v_L is

$$\sum_{i=1}^{J}\lambda_i G_{(i-1)\mu_L^{(l)}+\mu_o}(v_L). \tag{7.25}$$

From (7.25) and (7.24) we obtain:

$$E[\tanh\frac{v_L}{2}]=\sum_{i=1}^{J}\lambda_i\phi((i-1)\mu_L^{(l)}+\mu_o). \tag{7.26}$$

Similarly if v_R denotes the message on a randomly selected edge from a parity node to a check node,

$$E[\tanh\frac{v_R}{2}]=\phi(\mu_R^{(l)}+\mu_o). \tag{7.27}$$

Because of (7.5) we have

$$E[\tanh\frac{m(u\to v)}{2}]=\prod_{w\neq v}E[\tanh\frac{m(w\to u)}{2}]. \tag{7.28}$$

Denote a message from a check node to an information node, resp. parity node by u_L, resp. u_R. Replacing $E[\tanh\frac{v_{e'}}{2}]$ with the right side of (7.26) or (7.27) depending upon the message comes from the left or right, (7.28) becomes

$$\begin{aligned}E[\tanh\frac{u_L^{(l+1)}}{2}] &= E[\tanh\frac{v_L}{2}]^{a-1}E[\tanh\frac{v_R}{2}]^2\\ &= (\sum_{i=1}^{J}\lambda_i\phi((i-1)\mu_L^{(l)}+\mu_o))^{a-1}(\phi(\mu_R^{(l)}+\mu_o))^2,\end{aligned}$$

and

$$
\begin{aligned}
E[\tanh\frac{u_R^{(l+1)}}{2}] &= E[\tanh\frac{v_L}{2}]^a E[\tanh\frac{v_R}{2}] \\
&= (\sum_{i=1}^{J}\lambda_i\phi((i-1)\mu_L^{(l)}+\mu_o))^a\phi(\mu_R^{(l)}+\mu_o).
\end{aligned}
$$

Using the definition of $\phi(\mu)$ in (7.24), we have thus the recursion for $\mu_L^{(l)}$ and $\mu_R^{(l)}$:

$$
\begin{aligned}
\phi(\mu_L^{(l+1)}) &= (\sum_{i=1}^{J}\lambda_i\phi((i-1)\mu_L^{(l)}+\mu_o))^{a-1}\times(\phi(\mu_R^{(l)}+\mu_o))^2, &\quad (7.29)\\
\phi(\mu_R^{(l+1)}) &= (\sum_{i=1}^{J}\lambda_i\phi((i-1)\mu_L^{(l)}+\mu_o))^{a}\times\phi(\mu_R^{(l)}+\mu_o). &\quad (7.30)
\end{aligned}
$$

In order to have arbitrary small bit error probability, the means $\mu_L^{(l)}$ and $\mu_R^{(l)}$ should approach infinity as l approaches infinity. In the next subsection, we derive a sufficient condition for this.

7.4.2 Fixed point analysis

We now assume that iterative decoding has reached a fixed point of (7.29) and (7.30), i.e., $\mu_L^{(l+1)} = \mu_L^{(l)} = \mu_L$ and $\mu_R^{(l+1)} = \mu_R^{(l)} = \mu_R$. Denote $\sum_{i=1}^{J}\lambda_i\phi((i-1)\mu_L+\mu_o)$ by x. From (7.26) we can see that $0 < x < 1$ and $x \to 1$ if and only if $\mu_L \to \infty$. From (7.30) it's easy to show that μ_R is a function of x, denoted by f, i.e., $\mu_R = f(x)$. Then, dividing (7.29) by the square of (7.30) gives us

$$
\phi(\mu_L) = \phi^2(\mu_R)/x^{a+1} = \phi^2(f(x))/x^{a+1}. \qquad (7.31)
$$

Now replacing μ_L with $\phi^{(-1)}(\phi^2(f(x))/x^{a+1})$ into the definition of x, we obtain the following equation for the fixed point x:

$$
x = \sum_{i=1}^{J}\lambda_i\phi(\mu_o+(i-1)\phi^{(-1)}(\frac{\phi^2(f(x))}{x^{a+1}})). \qquad (7.32)
$$

If this equation doesn't have a solution in the interval $[0, 1]$, then the decoding bit error probability converges to zero. Therefore, if we have

$$F(x) \triangleq \sum_{i=1}^{J} \lambda_i \phi(\mu_o + (i-1)\phi^{(-1)}(\frac{\phi^2(f(x))}{x^{a+1}})) > x, \tag{7.33}$$

for any $x \in [x_0, 1)$, where x_0 is the value of x at first iteration, then (the Gaussian approximation to) iterative decoding is successful.

Since the rate of the code is given by (cf. (7.8)):

$$\frac{\sum_i \lambda_i/i}{a + \sum_i \lambda_i/i}, \tag{7.34}$$

to maximize the rate we should maximize $\sum_i \lambda_i/i$. Thus, under the Gaussian approximation, the problem of finding a good degree sequence for IRA codes is converted to the following linear programming problem:

Linear Programming Problem. *To maximize*

$$\sum_{i=1}^{J} \lambda_i/i, \tag{7.35}$$

under the condition

$$F(x) > x, \quad \forall x \in [x_0, 1]. \tag{7.36}$$

Using this linear programming methodology, we have designed some degree sequences for IRA codes. The results and presented in Tables 7.2 (code rate $\sim 1/3$) 7.3 (code rate $\sim 1/2$). After using the heuristic Gaussian approximatiob method to design the degree sequence, we used exact density evolution program to determine the actual noise threshold. (In every case, the true iterative decoding capacity was better than the one predicted by Gaussian approximation.)

For example, consider the "$a = 3$" column in Table 7.2. We adjust Gaussian approximation noise threshold σ_{GA} to be 1.2415 to have the returned optimal sequence

a	2	3	4
λ_2	0.139025	0.078194	0.054485
λ_3	0.222155	0.128085	0.104315
λ_5		0.160813	
λ_6	0.638820	0.036178	0.126755
λ_{10}			0.229816
λ_{11}			0.016485
λ_{12}		0.108828	
λ_{13}		0.487902	
λ_{14}			
λ_{16}			
λ_{27}			0.450302
λ_{28}			0.017842
rate	0.333364	0.333223	0.333218
σ_{GA}	1.1840	1.2415	1.2615
σ^*	1.1981	1.2607	1.2780
$(\frac{E_b}{N_o})^*(dB)$	0.190	-0.250	-0.371
S.L. (dB)	-0.4953	-0.4958	-0.4958

Table 7.2: Good degree sequences of rate one-third for the AWGN channel and with $a = 2, 3, 4$. For each sequence the Gaussian approximation noise threshold, the actual sum-product decoding threshold, and the corresponding $(\frac{E_b}{N_o})^*$ in dB are given. Also listed is the Shannon limit (S.L.).

having rate 0.333223. Then applying the exact density evolution program on this code, we obtain the actual sum-product decoding capacity $\sigma^* = 1.2607$, which corresponds $(\frac{E_b}{N_o})^* = -0.250$ dB. This should be compared to the Shannon limit for the ensemble of all linear codes of the same rate, which is -0.4958 dB. As we increase parameter a, the ensemble improves. For $a = 4$, the best code we have found has iterative decoding capacity $(\frac{E_b}{N_o})^* = -0.371$ dB, which is only 0.12 dB away from Shannon limit.

The above analysis is for bit error probability. In order to have zero *word* error probability, it is necessary to have $\lambda_2 = 0$. (This can be easily proved by the following argument: if $\lambda_2 > 0$, then in the ensemble the number of weight 2 codewords $\overline{A}_2 > 0$, hence even the optimal maximum-likelihood decoder would have non-zero decoding error probability.) In Table 7.3, we compare the noise thresholds of codes with and without λ_2.

a	8	8
λ_2		0.057713
λ_3	0.252744	0.117057
λ_7		0.218992
λ_8		0.033384
λ_{11}	0.081476	
λ_{12}	0.327162	
λ_{18}		0.214722
λ_{20}		0.075226
λ_{46}	0.184589	
λ_{48}	0.154029	
λ_{55}		0.080868
λ_{58}		0.202038
rate	0.50227	0.49794
σ^*	0.9589	0.9720
$(\frac{E_b}{N_o})^*(dB)$	0.344	0.266
Shannon limit	0.197	0.178

Table 7.3: Good degree sequences of rate roughly one-half for the AWGN channel and with $a = 8$. These two sequences are found by including or excluding λ_2 in the linear programming. For each sequence, the rate of the code, the actual sum-product decoding threshold, and the corresponding $(\frac{E_b}{N_o})^*$ in dB are given. Also listed is the Shannon limit.

We choose rate one-half because we wanted to compare our results with the best irregular LDPC codes obtained in [43]. Our best IRA code has signal-to-noise threshold 0.266 dB, while the best rate one-half irregular LDPC code found in [43] has threshold 0.25 dB. These two codes roughly have the same decoding complexity, but unlike LDPC codes, IRA codes have straightforward linear encoding.

7.4.3 Simulation

We simulated the rate one-half code with $\lambda_2 = 0$ in Table 7.3. Figure 7.3 shows the performance of that particular code, with information block lengths 10^3, 10^4, and 10^5.

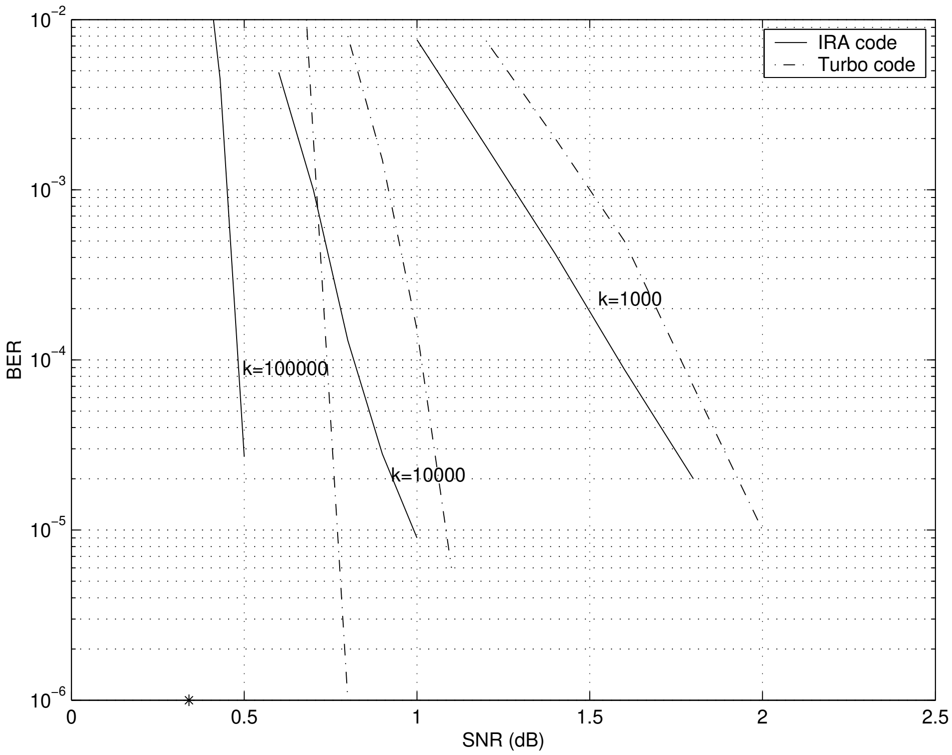

Figure 7.3: Comparison between turbo codes (dashed curves) and IRA codes (solid curves) of lengths $k = 10^3, 10^4, 10^5$. All codes are of rate one-half.

7.5 IRA Codes on the Fading Channels

7.5.1 Introduction

We know of practical codes and decoding algorithms that can closely approach channel capacity of some classical channels, e.g., the binary erasure channel, the binary symmetric channel, and the additive white Gaussian noise (AWGN) channel. While it appears on those channels there is not much left to be done (at least practically), other channels remain to be investigated. Those include channels of non-binary input, non-symmetric, and/or with memory.

In this section, we will focus on one of those non-standard channels, namely Rayleigh fading channels. The fading channels stand out as wireless communication becomes more important nowadays. Since the Rayleigh fading channels are time-varying Gaussian channels, it is natural to wonder whether powerful codes designed for the Gaussian channel still work in a fading environment. In [26], turbo codes were analyzed and simulated on the Rayleigh fading channel with coherent BPSK signaling. In this section we investigate the performance of IRA codes on the Rayleigh Fading channels using BPSK signaling. Three different Rayleigh fading models are considered: coherent detection and known (only to the receiver) fading coefficients; coherent detection and unknown fading coefficients; and noncoherent detection. For those three different channel models, we show that IRA codes can generally perform close to corresponding channel capacities.

7.5.2 Review: decoding of IRA codes

The decoding rules for IRA codes are given in eqs. (7.4) and (7.5). Before decoding, messages $m(w \to u)$ and $m(u \to v)$ are all initialized to be zero, and $m_0(u)$ is initialized to be the log-likelihood ratio based on the channel received information. If the channel is memoryless, and y is the output of the channel code bit u, then $m_0(u) = \log(p(u=0|y)/p(u=1|y))$. After this initialization, the decoding process runs in a fully parallel and local manner. In each iteration, every variable/check node receives messages from its neighbors, and sends back updated messages. Decoding is terminated after a fixed number of iterations or detecting that all the constraints are satisfied. Upon termination, the decoder outputs a decoded sequence based on the messages $m(u) = \sum_w m(w \to u)$.

Thus, on various channels iterative decoding only differs in the initial messages $m_0(u)$. For instance, on the binary erasure channel, $y \in \{0, E, 1\}$ with E being the erasure, then

$$m_0(u) = \begin{cases} +\infty & \text{if } y = 0 \\ 0 & \text{if } y = E \\ -\infty & \text{if } y = 1. \end{cases} \tag{7.37}$$

On the binary symmetric channel, $y \in \{0, 1\}$, then

$$m_0(u) = \begin{cases} \log \frac{1-p}{p} & \text{if } y = 0 \\ -\log \frac{1-p}{p} & \text{if } y = 1. \end{cases} \tag{7.38}$$

And finally, on the additive white Gaussian channel with BPSK signaling which maps 0 to the symbol with amplitude $\sqrt{E_s}$ and 1 to the symbol with amplitude $-\sqrt{E_s}$, output $y \in R$, then

$$m_0(u) = 4y\sqrt{E_s}/N_o,$$

where $N_o/2$ is the noise power spectral density.

7.5.3 Rayleigh fading channels

We consider BPSK signaling over a Rayleigh fading channel. The BPSK signaling maps bit 0 to the amplitude $\sqrt{E_s}$ and maps bit 1 to the amplitude $-\sqrt{E_s}$. The fading coefficient is assumed constant over 1 symbol time, but changes to a different constant at the next symbol time, and so on. With appropriate sampling, the discrete representation of this channel is

$$r = ae^{j\beta}x + n, \tag{7.39}$$

where r is the complex channel output, x is a BPSK symbol amplitude ($\pm\sqrt{E_s}$), and n is an i.i.d. AWGN component with zero mean and power spectral density $N_0/2$. The fading amplitude a is Rayleigh distributed with pdf,

$$p_A(a) = 2ae^{-a^2} \qquad \text{for } a > 0. \tag{7.40}$$

The random phase β is uniformly distributed over $[0, 2\pi]$. Assuming the channel is fully interleaved, the a's are mutually independent.

Assume s is the channel information provided to the receiver. Three different

Rayleigh fading models are considered, all being symmetric:

i) $s = (a, \beta)$, i.e., coherent detection and known fading amplitudes. With coherent detection, discrete representation of this channel eq. (7.39) can be simplified to an equivalent channel model

$$y = ax + n \tag{7.41}$$

where y is now a real number, and a, x, n are the same as in the original model. Conditioned on the fading amplitude a and transmitted signal x, the received signal y is a Gaussian random variable, with conditional probability density function

$$p(y|x = \sqrt{E_s}, a) = \mathcal{N}(a\sqrt{E_s}, N_0/2), \tag{7.42}$$

a Gaussian pdf with a mean $a\sqrt{E_s}$ and variance $N_0/2$.

ii) $s = (\beta)$, i.e., coherent detection and unknown fading amplitudes. Because of coherent detection, eq. (7.41) still holds. However, without the knowledge of a, the conditional probability distribution is

$$p(y|x = \sqrt{E_s}) = E_{p_A(a)}[p(y|x = \sqrt{E_s}, a)], \tag{7.43}$$

where $p(y|x = \sqrt{E_s}, a)$ is given in eq. (7.42) and $p_A(a)$ is given in eq. (7.40).

iii) $s = \phi$, i.e., noncoherent detection and unknown fading amplitudes. In [24, pp.63-65], it is shown this channel is equivalent to the following channel model. The output of the channel is $y \in R$, with transition density function

$$p(y|x = \sqrt{E_s}) = \begin{cases} \frac{1+A}{A(2+A)} e^{-\frac{y}{A}} & y \geq 0 \\ \frac{1+A}{A(2+A)} e^{\frac{y(1+A)}{A}} & y \leq 0 \end{cases} \tag{7.44}$$

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

We now consider the channel capacity of each channel model. Channel capacity is defined as the maximum over the input distribution $P_X(x)$ of the mutual information between the channel output and input $I(X;R)$. For the fading channel if the fading phase and/or fading amplitude is known, the mutual information is conditioned on this knowledge. We calculate the maximum of $I(X;Y)$ for these channels models with input X and output Y, conditioned on a if that is known. For this, we write the capacity expression as

$$\begin{aligned} C^{BPSK} &= \max_{P_X(x)} I(X;R|S) \\ &= \max_{P_X(x)} I(X;Y|Z) \\ &= \max_{P_X(x)} E_{p(x,y,z)}[\log_2(\frac{p(y|x,z)}{p(y|z)})] \\ &= \max_{P_X(x)} E_{p(x,y,z)}[\log_2(\frac{p(y|x,z)}{\sum_{x'} p_X(x')p(y|x',z)})]. \end{aligned}$$

$I(X;Y|Z)$ is the mutual information between X and Y conditioned on knowledge of the channel side information Z (it only exists for the first channel model, in which case, $z = a$.) $E_{p(x,y,z)}[\ ]$ is the expectation over the distribution $p(x,y,z)$. Based on the independence of z and x, this expectation can be written as

$$p(x,y,z) = p(y|x,z)p_X(x)p_Z(z). \tag{7.45}$$

Since in all three models the input is symmetric, the maximization in the capacity definition is achieved by an equiprobable input distribution $P_X(x = \sqrt{E_s}) = P_X(x = -\sqrt{E_s}) = 1/2$. Thus, the channel capacity could be evaluated given the relation eq. (7.45) and eqs. (7.42), (7.43), (7.44).

Numerical results of different channel capacities are shown in Figure 7.4. In the figure, from the top to the bottom, the first curve is incoherent detection and unknown fading amplitudes; the second curve is coherent detection without knowledge of fading amplitudes; the third curve is coherent with the knowledge of fading amplitudes. The fourth curve is additive white Gaussian channel, which is included for comparison.

We list the numerical numbers for those different channel capacities at rate $R = 1/2, 1/3, 1/4, 1/5$ in Table 7.4. Table 7.4 tells us, for instance, at rate 1/2, an error-

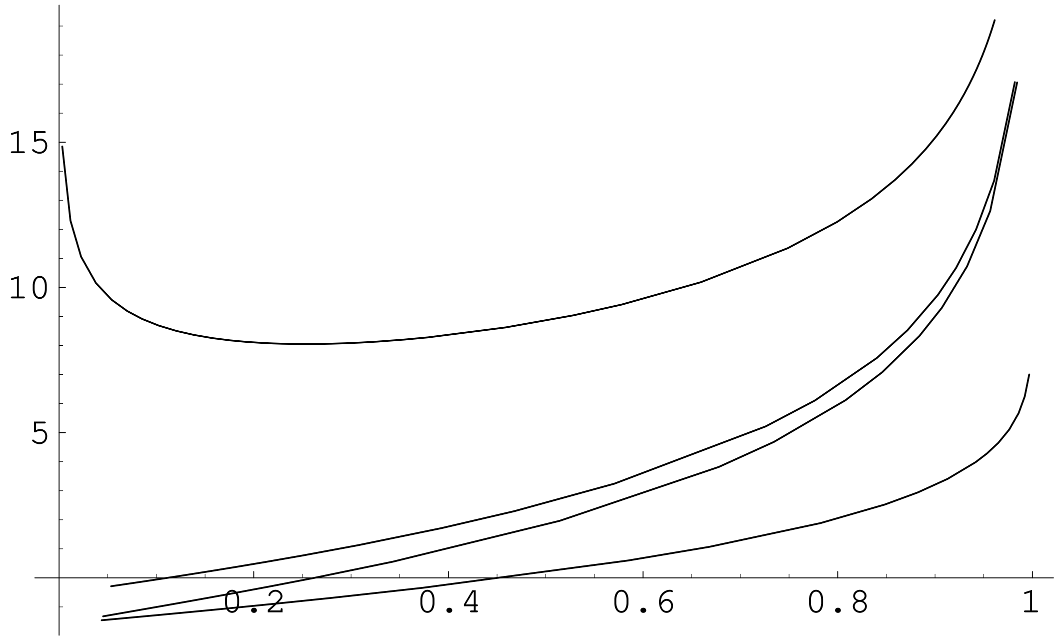

Figure 7.4: Rayleigh fading channel capacities, where the x-axis represents the rate between 0 and 1, and the y-axis represents the signal-to-noise ratio E_b/N_o in dB.

free transmission scheme would require a minimum signal-to-noise ratio of 0.2 dB on the Gaussian channel. If the channel experiences Rayleigh fading but the receiver does coherent detection, then the minimum SNR requirement increases to 1.8 dB if the receiver has side information; and 2.7 dB if the receiver does not have. If the receiver is unable to do coherent detection, then the minimum signal-to-noise ratio increases to 9.0 dB.

Rate	1/2	1/3	1/4	1/5
Rayleigh-SI (dB)	1.8	0.7	-0.1	-0.4
Rayleigh-NSI (dB)	2.7	1.2	0.8	0.5
Rayleigh-Noncoherent (dB)	9.0	8.2	8.0	8.1
AWGN (dB)	0.2	-0.5	-0.8	-1.0

Table 7.4: Channel capacities: numerical data gleaned from Figure 7.4.

7.5.4 IRA codes on Rayleigh fading channels

As described in Section 7.5.2, on various channels iterative sum-product decoding algorithm for IRA codes only differs in initializing messages $m_0(u)$. Suppose u_k is the codebit transmitted during time k; x_k is the transmitted symbol, a_k is the fading amplitude; y_k is the received symbol. With coherent detection and known fading amplitudes, the likelihood ratio is

$$
\begin{aligned}
\frac{p(u_k = 0|y_k, a_k)}{p(u_k = 1|y_k, a_k)} &= \frac{p(u_k = 0, y_k|a_k)}{p(u_k = 1, y_k|a_k)} \\
&= \frac{p(y_k|u_k = 0, a_k)}{p(y_k|u_k = 1, a_k)} \\
&= \frac{p(y_k|x_k = \sqrt{E_s}, a_k)}{p(y_k|x_k = -\sqrt{E_s}, a_k)} \\
&= e^{4a_k y_k \sqrt{E_s}/N_0}.
\end{aligned}
$$

Hence the initial message is

$$m_0(u_k) = \log \frac{p(u_k = 0|y_k, a_k)}{p(u_k = 1|y_k, a_k)} = 4a_k y_k \sqrt{E_s}/N_0. \tag{7.46}$$

With coherent detection and unknown fading amplitudes, the likelihood ratio is

$$
\begin{aligned}
\frac{p(u_k = 0|y_k)}{p(u_k = 1|y_k)} &= \frac{p(u_k = 0, y_k)}{p(u_k = 1, y_k)} \\
&= \frac{E_{p_A(a_k)}[p(y_k|u_k = 0, a_k)]}{E_{p_A(a_k)}[p(y_k|u_k = 1, a_k)]} \\
&\triangleq \Phi(y_k),
\end{aligned}
$$

where $E_{p_A(a)}[p(y|x, a)]$ is given in eq. (7.43). Hence the initial message $m_0(u_k)$ is given by

$$m_0(u_k) = \log \Phi(y_k). \tag{7.47}$$

The function $\Phi(y)$ has no known closed form, so we use an approximation to $\log \Phi(y_k)$. Figure 7.5 shows a typical $\log \Phi(y_k)$ function. It indicates that this function can be well approximated by the linear function $2Cy_k\sqrt{E_s}/N_0$. Conventionally this constant C is chosen to be $C = 0.8862$ [26] (although we believe $C = 0.82$ is a more accurate approximation); that number is used in our simulation as well.

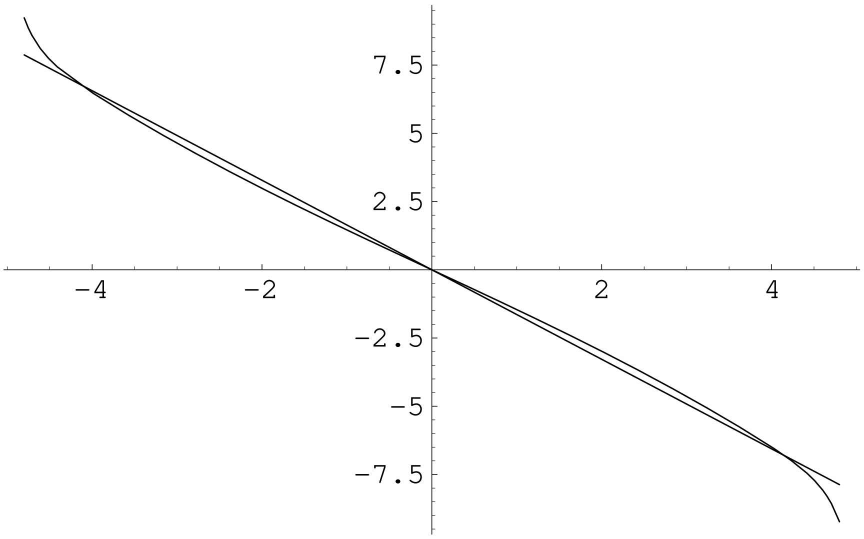

Figure 7.5: Plot of $\log \Phi(y)$, for $E_s = 1, N_0/2 = 1.0$.

For the noncoherent channel without any side information, the initial message $m_0(u_k)$ is [24, eq. (6.7)]

$$m_0(u_k) = y_k. \tag{7.48}$$

All three Rayleigh fading channels were simulated by using a pseudorandom number generator to produce outputs y according to the probability distribution of eq. (7.42), (7.43), or (7.44). Since we have not designed IRA codes specifically for any of these channels, we used some of the degree profiles designed for the AWGN channel from Section 7.4. We expect reasonable performance because Rayleigh fading channel can be regarded as a time-varying Gaussian channel.

In Table 7.5, the first column shows the degree profile of an IRA code of rate approximately 1/2. The second columns gives the degree profile of an IRA code of

rate	0.509095	0.333218
a	*8*	*4*
f_2		0.218052
f_3	0.667892	0.278317
f_6		0.169094
f_{10}		0.183948
f_{11}	0.058720	0.011995
f_{12}	0.216137	
f_{27}		0.133492
f_{28}		0.005100
f_{46}	0.031812	
f_{48}	0.025439	

Table 7.5: Degree sequences of the IRA codes in simulation.

rate 1/3. Both are taken from Section 7.4.[1]

In all simulations, 20 word errors were accumulated to reduce the variance in the bit error rate estimates.

Figure 7.6 shows the performance of the rate 1/2 IRA codes in Table 7.5. The code has information block length 10^4. Reading from right to left, the curves represent the performance on a Rayleigh fading channel with noncoherent detection, Rayleigh fading with coherent detection and unknown fading coefficient, Rayleigh fading with coherent detection and known fading coefficient, and AWGN channel. The marks on the x-axis are the channel capacities for each of those channels.

In Figure 7.6, for the SI channel, BER=10^{-4} is achieved at signal-to-noise ratio of 2.75 dB, while the channel capacity is 1.8 dB. For the NSI channel, the same BER is achieved approximately at 4.8 dB, while the channel capacity is 2.7 dB. And for the non-coherent case, this is achieved at about 9.8 dB, comparing with the Shannon limit 9.0 dB.

Figure 7.7 shows the performance of the IRA codes with rate 1/3 in Table 7.5. The code has information block length 10^4. From the right to left, the performance is on Rayleigh fading channel with incoherent detection, Rayleigh fading with coherent

[1]Except here we use node fraction, instead of edge fraction, to represent the degree profile. For both notations, refer to Section 7.3.

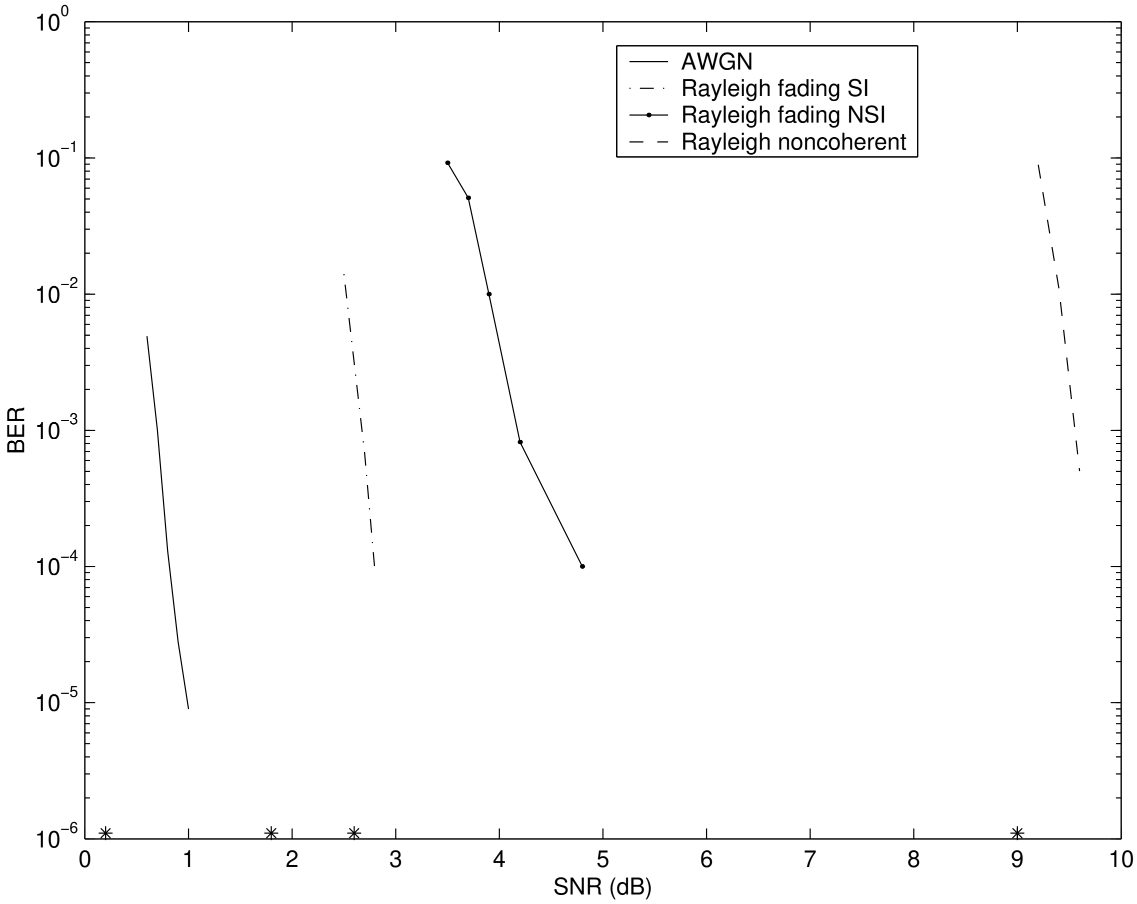

Figure 7.6: Performance of IRA code, rate 1/2, information block length 10^4.

detection and unknown fading coefficient, and Rayleigh fading with coherent detection and known fading coefficient. The marks on the x-axis are the channel capacities for each of those channels.

Again, IRA codes are performing within about 1 dB of the channel capacity. To achieve BER of 10^{-4}, on the SI channel it requires SNR about 1.0 dB; on the NSI channel it requires SNR about 2.0 dB; on the noncoherent channel, it requires 8.9 dB. The corresponding channel capacities are 0.7, 1.2, and 8.2 dB respectively.

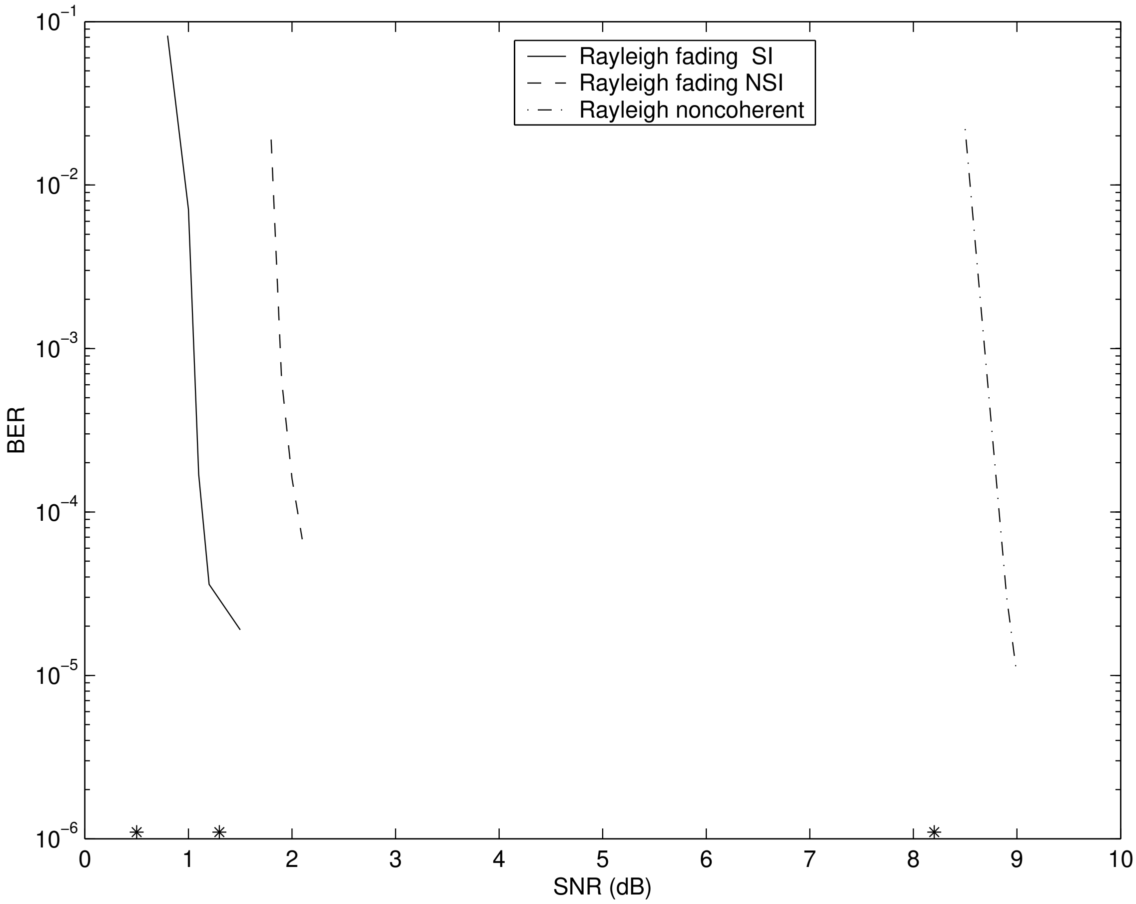

Figure 7.7: Performance of IRA code, rate 1/3, information block length 10^4.

7.6 Conclusions

In this chapter, we have introduced a class of codes, the IRA codes, that combines many of the favorable attributes of turbo codes and LDPC codes. Like turbo codes (and unlike LDPC codes), they can be encoded in linear time. Like LDPC codes (and unlike turbo codes), they are suited to an exact density evolution style analysis. In simulated performance on some typical channels, including the AWGN channel and Rayleigh Fading channels, they appear to be slightly superior to turbo codes of comparable complexity, and just as good as the best known irregular LDPC codes.

Appendix A AWGN Error Exponents

In this appendix we will discuss the general class of AWGN error exponents, which can be used to prove coding theorems from weight enumerators (for detail, refer to [16]). Such an exponent is a function $E(\delta, r, c)$ of three variables:

$$\begin{aligned} \delta : 0 &\leq \delta \leq 1 \\ r : 0 &\leq r \leq H(\delta) \\ c : c &\geq 0. \end{aligned}$$

Here are two examples of error exponents that enjoy the properties we describe below: the UB (union bound) exponent and the DB (Divsalar bound) exponent.

- The UB exponent:

$$E_U(\delta, r, c) = \max(0, \delta c - r). \tag{A.1}$$

- The DB exponent [15]:

$$E_D(\delta, r, c) = \max_{0 \leq \beta \leq 1} (-r + \frac{1}{2} \ln(\beta + (1-\beta)e^{2r}) + \frac{\delta \beta c}{1 - \delta(1-\beta)}. \tag{A.2}$$

(Note that if the maximum in (A.2) is restricted to the two values $\beta = 0$ and $\beta = 1$, the bound (A.1) results.)

Here are the postulated properties of $E(\delta, r, c)$.

Property A.1 *If C is a binary code with A_h words of weight h, and if the all-zeros word is transmitted over the AWGN channel, the probability that ML decoder will* *prefer some word of weight h to the all-zeros word is bounded as follows:*

$$P_{E,h}(c) \leq e^{-nE(\delta,r,c)}, \tag{A.3}$$

where $\delta = h/n$, $r = (1/n)\log A_h$, *and* $c = E_s/N_o = RE_b/N_o$.

Property A.2 $E(\delta, r, c)$ *must satisfy the following conditions:*

$$\begin{aligned} E(\delta, r, c) &\geq 0 && \text{(A.4)} \\ \frac{\partial E}{\partial c}(\delta, r, c) &\geq 0 && \text{(A.5)} \\ 0 \geq \frac{\partial E}{\partial r}(\delta, r, c) &\geq 0 && \text{(A.6)} \\ \frac{\partial^2 E}{\partial^2 r}(\delta, r, c) &\geq 0 && \text{(A.7)} \end{aligned}$$

We now define

$$c_0(\delta, r) = \inf\{c : E(\delta, r, c) > 0\}. \tag{A.8}$$

- For the UB:

$$c_0(\delta, r) = r/\delta.$$

- For the DB:

$$c_0(\delta, r) = \frac{1-\delta}{\delta}(\frac{1-e^{-2r}}{2}).$$

Property A.3 *For any* $\Delta > 0$, *there exists* $\epsilon > 0$ *such that if* $c - c_0(\delta, r) \geq \Delta > 0$, *then*

$$E(\delta, r, c) > \delta\epsilon.$$

Proof: We only prove this porperty for $E_D(\delta, r, c)$ because it is obvious for $E_U(\delta, r, c)$. Define

$$\beta(\delta, r, c) \triangleq \underset{0\le\beta\le 1}{\operatorname{argmax}}(-r + \frac{1}{2}\ln(\beta + (1-\beta)e^{2r}) + \frac{\delta\beta c}{1-\delta(1-\beta)}. \tag{A.9}$$

The expression is given in [15]:

$$\beta(\delta, r, c) = \min(1, \frac{1-\delta}{\delta}(\sqrt{\frac{c - c_0(\delta, r)}{c_0(\delta, r)} + (1+c)^2} - (1+c))). \tag{A.10}$$

Denote $\beta(\delta, r, c_0(\delta, r) + \Delta/2)$ by β^*. For $c \ge c_0(\delta, r) + \Delta$,

$$\begin{aligned}
E(\delta, r, c) &= E(\delta, r, \beta(\delta, r, c), c) \\
&\overset{(a)}{\ge} E(\delta, r, \beta^*, c) \\
&\overset{(b)}{\ge} E(\delta, r, \beta^*, c_0(\delta, r) + \Delta/2) + \frac{\delta\beta^*}{1-\delta(1-\beta^*)}\frac{\Delta}{2} \\
&\overset{(c)}{\ge} \frac{\delta\beta^*}{1-\delta(1-\beta^*)}\frac{\Delta}{2} \\
&= \delta\frac{\Delta/2}{(1-\delta)/\beta^* + \delta}
\end{aligned} \tag{A.11}$$

where (a) is according to definition of $\beta(\delta, r, c)$; (b) follows from the fact $E(\delta, r, c)$ is increasing and linear in c; (c) because $E(\delta, r, c)$ is non-negative. If $\beta^* = 1$, the lemma is proved by letting $\epsilon = \Delta/2$. If $\beta^* < 1$, the exact expression is given by (A.10),

$$\beta^* = \frac{1-\delta}{\delta}(\sqrt{\frac{\Delta}{2c_0(\delta, r)} + K^2} - K),$$

where $K = 1 + c_0(\delta, r) + \Delta/2$.

Taking this formula for β^*, we obtain

$$\begin{aligned}(1-\delta)/\beta^* + \delta &\leq \frac{\delta}{\sqrt{\frac{\Delta}{2c_0(\delta,r)} + K^2} - K} + \delta \\ &= \frac{\sqrt{2c_0(\delta,r)\Delta + (2c_0(\delta,r)K)^2} + K}{\Delta} + 1 \qquad \text{(A.12)} \\ &\leq F(M,\Delta). \qquad \text{(A.13)}\end{aligned}$$

The last inequality (A.13) follows as (A.12) is an increasing function of $c_0(\delta, r)$, which is bounded by M. Hence, by setting $\Delta/(2F(M,\Delta)) = \epsilon$ in (A.11), we obtain $E(\delta, r, c) \geq \delta\epsilon$. ■

Property A.4 *For any (δ, r) and $\theta \geq 0$, we have*

$$c_0(\delta, r+\theta) \leq c_0(\delta, r) + \theta/\delta.$$

Proof: Simple algebra gives us

$$\begin{aligned}(1-e^{-2(r+\theta)})\frac{1-\delta}{2\delta} &= (1-e^{-2\theta}e^{-2r})\frac{1-\delta}{2\delta} \\ &\leq (1-(1-2\theta)e^{-2r})\frac{1-\delta}{2\delta} \\ &\leq (1-e^{-2r})\frac{1-\delta}{2\delta} + \frac{\theta}{\delta}. \qquad ■\end{aligned}$$

Let us assume that we have some information about the rate at which $r_n(\delta)$ approaches $r(\delta)$, of the form

$$r_n(\delta) \leq r(\delta) + \theta_n, \qquad \text{(A.14)}$$

where $\theta_n \geq 0$, is a sequence of positive real numbers tending to zero.

Now we define the *ensemble thresholds* for a code ensemble with spectral shape

$r(\delta)$,

$$c_0 \triangleq \sup_{0<\delta<1} c_0(\delta, r(\delta)). \tag{A.15}$$

The nth *innominate sum* is defined as follows:

$$Z^{(n)}(D) \triangleq \sum_{h=1}^{D} \overline{A}_h{}^{(n)},$$

where D is an integer with $1 \leq D \leq n$.

The following is our main theorem of this section.

Theorem A.1 *If D_n is a sequence of positive integers such that*

$$\lim_{n\to\infty} \frac{n\theta_n}{D_n} = 0, \tag{A.16}$$

and if $c > c_0$, there exists an integer n_0 and positive constants K and ϵ such that for $n \geq n_0$,

$$P_W^{(n)} \leq \sum_{h=1}^{n} P_{E,h}(c) \leq Z^{(n)}(D_n) + Ke^{-\epsilon D_n}. \tag{A.17}$$

Proof: By Property A.1, we have

$$P_{E,h}(c) \leq e^{-nE(\delta, r_n(\delta), c)}.$$

For any $\delta \geq D_n/n$,

$$\begin{aligned}
c_0(\delta, r_n(\delta)) &\leq c_0(\delta, r(\delta) + \theta_n) && \text{assumption A.14} \\
&\leq c_0(\delta, r(\delta)) + \theta_n/\delta && \text{Property A.4} \\
&\leq c_0(\delta, r(\delta)) + n\theta_n/D_n.
\end{aligned}$$

With assumption (A.16), we have

$$\lim_{n\to\infty} c_0(\delta, r_n(\delta)) = c_0(\delta, r(\delta)).$$

Hence, for $c \geq c_0 = \sup_\delta c_0(\delta, r(\delta))$, there exists an integer n_0 and $\Delta > 0$, such that for $n \geq n_0$

$$c - c_0(\delta, r_n(\delta)) \geq \Delta \tag{A.18}$$

for any $\delta \geq D_n/n$. By Property A.3, there exists $\epsilon > 0$ such that

$$E(\delta, r_n(\delta), c) > \delta\epsilon.$$

Hence, when $n \geq n_0$, we have

$$\begin{aligned}\sum_{h=1}^{n} P_{E,h}(c) &\leq \sum_{h=1}^{D_n} A_h + \sum_{h\geq D_n} e^{-nE(\delta, r_n(\delta,c))} \\ &\leq Z^{(n)}(D_n) + \sum_{h\geq D_n} e^{-\epsilon h} \\ &\leq Z^{(n)}(D_n) + Ke^{-\epsilon D_n}. \qquad \blacksquare\end{aligned}$$

Corollary A.1 *If in addition,* $Z^{(n)}(D_n) \to 0$ *and* $D_n \to 0$*, then for* $E_b/N_o > (1/R)c_0$*,*

$$P_W^{(n)} \to 0, \tag{A.19}$$

i.e., the ensemble is "good."

Appendix B Miscellaneous Derivations for Chapter 4

B.1 IOWE for the Inner Code of RDD Codes

As defined, the inner code of RDD codes is a truncated rate-1 convolutional code with transfer function $1/(1+D+D^2)$. Thus an input block $\mathbf{x} = [x_1, x_2, \ldots, x_n]$ and an output block $\mathbf{y} = [y_1, y_2, \ldots, y_n]$ are related by the formula:

$$x_k = y_{k-2} + y_{k-1} + y_k, \tag{B.1}$$

if we define $y_{-1} = y_0 = 0$. To count $A_{w,h}$, the number of input-output pairs such that input block has weight w and output block has weight h, we divide the output block $\mathbf{y}$ into runs of 1s (1-run) and 0s (0-run). There are four possibilities regarding the structure of y: (1) $\mathbf{y}$ starts with an 1-run, ends with a 0-run; (2) $\mathbf{y}$ starts and ends with an 1-run; (3) $\mathbf{y}$ starts with a 0-run, ends with an 1-run; (4) $\mathbf{y}$ starts and ends with a 0-run. We first focus on the first case, i.e.,

$$\mathbf{y} = \underbrace{1\cdots1}_{a_1}\overbrace{0\cdots0}^{b_1}\underbrace{1\cdots1}_{a_2}\cdots\underbrace{1\cdots1}_{a_k}\overbrace{0\cdots0}^{b_k}, \tag{B.2}$$

where positive integers a_i and b_i satisfy

$$\sum_{i=1}^{k} a_i = h; \qquad \sum_{i=1}^{k} b_i = n - h. \tag{B.3}$$

According to (B.1), possible 3-tuples (y_{i-2}, y_{i-1}, y_i) generate $x_i = 1$ are $(100), (010), (001), (111)$. To count the number of those 3-tuples, let us first assume there are s 0-runs with length 1 and there are t 1-runs with length 1. Since each 0-run of length ≥ 2

generates 1 (100), there are totally $k-s$ (100)'s. Similarly, there are t (010)'s, $\sum_{a_i\geq 2}(a_i-2)=h+t-2k$ (111)'s, and $k-s$ or $k-s+1$ (001)'s depending whether $b_k=1$. Because the input weight is w, we have the constraint

$$h+2s-2t=w, \quad \text{or} \quad h+2s-2t+1=w. \tag{B.4}$$

Now, $A_{w,h}$ for case (1) is the total number of pairs $(\mathbf{x},\mathbf{y})$ satisfying eqs. (B.3) and (B.4). This computation is straightforward:

$$A^{(1)}_{w,h}=\sum_{k=1}^{h-1}\sum_{t=0}^{k-1}\binom{k}{t}\binom{k-1}{\lfloor s\rfloor}\binom{h-k-1}{k-t-1}\binom{n-h-k-1}{k-\lceil s\rceil-1} \tag{B.5}$$

where $s=\frac{h-w}{2}+t$.

The other three cases involve similar calculation. It is verified that in each case $A_{w,h}$ is bounded by $2A^{(1)}_{w,h}$ in (B.5). Hence the overall $A_{w,h}$ is

$$A_{w,h}=D\sum_{k=1}^{h-1}\sum_{t=0}^{k-1}\binom{k}{t}\binom{k-1}{\lfloor s\rfloor}\binom{h-k-1}{k-t-1}\binom{n-h-k-1}{k-\lceil s\rceil-1}$$

where $s=\frac{h-w}{2}+t$, and $1\leq D\leq 7$.

B.2 Proof of Property 4.1

$$
\begin{aligned}
A_h^{(n)} &\overset{(a)}{=} \sum_{w=1}^{3h/q} A_{w,h}^{(qN)} \\
&\overset{(b)}{=} \sum_{k=1}^{h-1}\sum_{t=0}^{k-1}\binom{k}{t}\binom{h-k-1}{k-t-1}\sum_{w=1}^{3h/q}\underbrace{D\frac{\binom{N}{w}}{\binom{qN}{qw}}\binom{k}{\lfloor\frac{h-qw+2t}{2}\rfloor}\binom{qN-h-k-1}{k-\lceil\frac{h-qw+2t}{2}\rceil-1}}_{S(w)} \\
&\overset{(c)}{=} \sum_{k=1}^{h-1}\sum_{t=0}^{k-1}\underbrace{\binom{k}{\lfloor\frac{h-q+2t}{2}\rfloor}\binom{k}{t}\binom{h-k-1}{k-t-1}O(N^{-q+k-\lceil\frac{h-q+2t}{2}\rceil})}_{Q(t)} \\
&\overset{(d)}{=} \sum_{k=1}^{h-1}\underbrace{\binom{h-k-1}{k-1}\binom{k}{\lfloor\frac{h-q}{2}\rfloor}O(N^{-q+k-\lceil\frac{h-q}{2}\rceil+\epsilon})}_{U(k)} \\
&\overset{(e)}{=} \binom{\lfloor h/2\rfloor}{\lfloor(h-q)/2\rfloor}O(N^{-q+\lfloor h/2\rfloor-\lceil(h-q)/2\rceil}+\epsilon) \\
&\overset{(f)}{=} O(n^{-\lceil q/2\rceil+\epsilon})
\end{aligned}
\tag{B.6}
$$

Hereby, step (a) follows from the fact $s = \lfloor(h-qw+2t)/2\rfloor \geq 0$, and $t \leq k \leq h$, hence $3h \geq h+2t \geq qw$, i.e., $w \leq 3h/q$; (b) is obtained by plugging in the expressions for $A_{w,h}$ and rearrange the sum. To see (c), we can verify for fixed $w, k, t, h = O(\log N)$, $S(w)$ is a decreasing function of w. So the sum can be upperbounded by the first term times the summation range, which is $O(N^\epsilon)$ for any $\epsilon > 0$. For step (d), it's easy to show that for fixed $t, k, h = O(\log N)$, $Q(t)$ is a decreasing function of t. Step (e) follows because for fixed $k, h = O(\log N)$, $U(k)$ is an increasing function of k. Step (f) follows from the fact that the binomial coefficient can be overbouned by N^ϵ for any $\epsilon > 0$, and $n = qN$.

Taking the summation for $h \leq D_n$, we have

$$
Z^{(n)}(D_n) \leq O(n^{-\lceil q/2\rceil+\epsilon}) \tag{B.7}
$$

since the summation range is n^ϵ for any $\epsilon > 0$. ∎

B.3 Spectral Shape of RDD Code Ensembles

Given $A^{(qN)}_{w,h}$ in (4.3), we have

$$A^{(qN)}_{h} \leq DN^3 \max_{w,k,t} \frac{\binom{N}{w}}{\binom{qN}{qw}} \binom{k}{t} \binom{k}{\lfloor s \rfloor} \binom{h-k-1}{k-t-1} \binom{qN-h-k-1}{k-\lceil s \rceil -1} \tag{B.8}$$

where $s = (h-qw)/2 + t$. By definition, $\delta = \frac{h}{qN}$. Further let us define $x = \frac{w}{qN}$, $u = \frac{k}{qN}$, and $v = \frac{t}{qN}$. Therefore, $\frac{s}{qN} = \delta/2 - qx/2 + v$. Follow from the fact that $\frac{1}{n+1} 2^{nH(k/n)} \leq \binom{n}{k} \leq 2^{nH(k/n)}$ for any $1 \leq k \leq n$, (see, e.g., [13]), we have

$$\lim_{n\to\infty} \log \binom{xn+a}{yn+b} / n = xH(\frac{y}{x}) \tag{B.9}$$

where $0 \leq x, y \leq 1$ and a, b are arbitrary. Hence $r(\delta)$, as defined, can be simplified to

$$\begin{aligned} r(\delta) &= \lim_{N\to\infty} \log A_h/(qN) \\ &\leq \max_{x,u,v}((1/q-1)H(qx) + uH(u/v) + uH(\frac{\delta/2 - qx/2 + v}{u}) \\ &+ (\delta - u)H(\frac{u-v}{\delta - u}) + (1-\delta-u)H(\frac{u - v - \delta/2 + qx/2}{1-\delta-u}). \end{aligned} \tag{B.10}$$

Here we can exchange the order of max and lim because the convergence in (B.9) is uniform. On the other side,

$$A^{(qN)}_{h} \geq D \max_{w,k,t} \frac{\binom{N}{w}}{\binom{qN}{qw}} \binom{k}{t} \binom{k}{\lfloor s \rfloor} \binom{h-k-1}{k-t-1} \binom{qN-h-k-1}{k-\lceil s \rceil -1}$$

which again can be simplified to

$$\begin{aligned} r(\delta) &\geq \max_{x,u,v}((1/q-1)H(qx) + uH(u/v) + uH(\frac{\delta/2 - qx/2 + v}{u}) \\ &+ (\delta - u)H(\frac{u-v}{\delta - u}) + (1-\delta-u)H(\frac{u - v - \delta/2 + qx/2}{1-\delta-u}). \end{aligned} \tag{B.11}$$

Combining with (B.10) and (B.11), $r(\delta)$ equals to the RHS of (B.11). Moreover, from (B.8), $r_n(\delta) \leq 3 \log n/n + r(\delta)$. ■

Appendix C Miscellaneous Derivations for Chapter 5

C.1 Weight Enumerator Estimates for Truncated Convolutional Codes

In this Appendix we shall state for reference three useful combinatorial facts about the weight structure of convolutional codes, due essentially to Kahale and Urbanke [35].

Theorem C.1 *(The η-μ theorem.) For a non-catastrophic convolutional encoder E, there exists a constant μ, $\mu = \mu(E)$, such that if the output weight is h, then the input weight is at most μh. Also, there is a constant $\eta = \eta(E)$ such that if a codeword in the truncated code consists of several detours, of total length L_0, then the codeword weight d satisfies $d \geq L\eta$.*

In what follows, $A_h^{(L)}$ denotes the number of codewords of weight h in the Lth truncation of the code and $A_{w,h}^{(L)}$ denotes the corresponding number of codewords with input weight w and output weight h. Thus $A_h^{(L)} = \sum_w A_{w,h}^{(L)}$. Similarly, $A_{w,\leq h}^{(L)}$ denotes the the number of codewords with input weight w and output weight less than or equal to h, i.e., $A_{w,\leq h}^{(L)} = \sum_{d=1}^{h} A_{w,d}^{(L)}$.

Theorem C.2 *([35, Lemma3]) Let C be an (n, k, m) convolutional code, as represented by a noncatastrophic encoder E. Then for the $(nL, kL-m)$ block code obtained by truncating C at depth L,*

$$A_h^{(L)} \leq \theta^h \binom{L}{\lfloor h/d_1 \rfloor}, \tag{C.1}$$

where d_1 is the free distance of the code, and θ is a constant independent of h and n.

We define a *recursive* convolutional code to be one for which any input of weight 1 produces an output of infinite weight.

Theorem C.3 *([35, Lemma1]) Let C be an (n, k, m) recursive convolutional code, with corresponding noncatastrophic encoder E. Then for the $(nL, kL - m)$ block code obtained by truncating the E-trellis representation of C at depth L,*

$$A^{(L)}_{w,\leq h} \leq \theta^w \sum_{j=0}^{\lfloor w/2 \rfloor} \binom{L}{j} \binom{\eta h}{w-j}, \tag{C.2}$$

where θ and η are constants independent of w, h, and n. (For the significance of η, see Theorem C.1.)

C.2 Some Useful Inequalities

Suppose n, k are positive integers, $1 \leq k \leq n$. Then

$$\binom{n}{k} \leq 2^n \tag{C.3}$$

$$\frac{n^k}{k^k} \leq \binom{n}{k} \leq n^k \tag{C.4}$$

$$e^{nH(k/n)}/(n+1) \leq \binom{n}{k} \leq e^{nH(k/n)}. \tag{C.5}$$

(For (C.5), see [13, Example 12.1.3, p. 284])

Proposition C.1 *If $n \geq m$, $1 \leq j \leq \lfloor w/2 \rfloor$, then*

$$\binom{n}{j}\binom{m}{w-j} \leq \binom{n}{\lfloor w/2 \rfloor}\binom{m}{\lceil w/2 \rceil} = \binom{n}{\lfloor w/2 \rfloor}\binom{m}{\lceil w/2 \rceil}. \tag{C.6}$$

Proof: It suffices to show that given the condition, $f(j) = \binom{n}{j}\binom{m}{w-j}$ is an increasing function of j. Consider the ratio

$$\frac{f(j)}{f(j-1)} = \frac{n-j+1}{m-w+j}\frac{w-j+1}{j}.$$

Since $w-j+1 \geq j$ and $n-j+1 \geq m-w+j$, we have $f(j)/f(j-1) \leq 1$. Hence the conclusion follows. ■

Proposition C.2

(1) Given $F_n(w) = \Theta^w n^{J\lfloor w/2 \rfloor - (J-1)w} D_n^{(2J-1)w}$, $1 \leq w \leq \mu D_n$, $F_n(2)$ *will be the largest term as* n *becomes large.*

(2) Given $G_n(d) = \Theta^d n^{\lfloor d/d_1 \rfloor + \lfloor d/2 \rfloor - d} D_n^{2d}$, $d_1 \leq d \leq \mu D_n$, $G_n(d_1)$ *will be the largest term as* n *becomes large.*

Proof: (1): It is easy to show that $F_n(w)$ satisfies $F_n(1) \geq F_n(3) \geq F_n(5) \geq \ldots$ and $F_n(2) \geq F_n(4) \geq F_n(6) \geq \ldots$ as n gets large by taking the ratio of two consecutive terms. Verifying that $F_n(2) \geq F_n(1)$ for large n, we have the claim.

(2): Similarly, we can show $G_n(d_1) \geq G_n(d_1+1) \geq \ldots \geq G_n(\mu D_n)$ by taking the ratio of two consecutive terms. ■

Proposition C.3 *Given real numbers* α_i, $\mathbf{b}_i$ *for* $i = 1, \ldots, n$, *with* $\mathbf{b}_i \geq 0$, *define*

$$\Delta = \sum_{i=1}^{n} \alpha_i \mathbf{b}_i,$$

and let

$$L = \lim_{\delta \to 0} \frac{1}{\delta} \sup_{0 < x < \mu\delta} \sum_{i=1}^{n} \alpha_i H(\mathbf{b}_i x). \tag{C.7}$$

Then

$$L = \begin{cases} +\infty & \text{if } \Delta > 0 \\ \mu \sum_i \alpha_i \mathbf{b}_i \log(\frac{1}{\mathbf{b}_i}) & \text{if } \Delta = 0 \\ 0 & \text{if } \Delta < 0. \end{cases}$$

Proof: (Sketch). It is easy to see that for small x,

$$H(x) = x \log \frac{1}{x} + x + O(x^2),$$

and so

$$\sum_i \alpha_i H(\mathbf{b}_i x) = \Delta x \log \frac{1}{x} + \left(\sum_i \alpha_i \mathbf{b}_i (1 + \log \frac{1}{\mathbf{b}_i})\right) x + O(x^2).$$

If $\Delta \neq 0$, the first term in the above expansion dominates, and the result follows immediately. If $\Delta = 0$ we have

$$\sum_i \alpha_i H(\mathbf{b}_i x) = \left(\sum_i \alpha_i \mathbf{b}_i \log(\frac{1}{\mathbf{b}_i})\right) x + O(x^2), \tag{C.8}$$

in which case the "sup" in (C.7) is attained at $x = \mu\delta$ as $\delta \to 0$. ■

C.3 Bit Error Probability vs. Word Error Probability

The union bound on the bit error probability for maximum likelihood decoding of an (n, k) binary linear code C with IOWE $(A_{w,k})$ over a memoryless binary input channel has the form

$$P_b^{(n)} \leq \sum_{h=1}^{n} \sum_{w=1}^{k} \frac{w}{k} A_{w,h}^{(n)} e^{-\alpha h}. \tag{C.9}$$

In this appendix, we will state, and sketch a proof of, a theorem on the ensemble bit error probability $\overline{P}_b^{(n)}$, analogous to Theorem 5.2, whch deals with word error probability. To that end, we define another innominate sum:

$$Y^{(n)}(D) \triangleq \sum_{h=1}^{D} \sum_{w=1}^{k} \frac{w}{k} \overline{A}_{w,h}^{(n)}. \tag{C.10}$$

Theorem C.4 *If the threshold c_0 defined in (5.4) is finite, then if $\alpha > c_0$, there exists an integer n_0 and positive constants K and ϵ such that for $n \geq n_0$,*

$$P_b^{(n)} \leq Y^{(n)}(D_n) + K e^{-\epsilon D_n}. \tag{C.11}$$

Proof: (Sketch.) Beginning with eq. (C.9), we have

$$\begin{aligned}
P_b^{(n)} &\leq \sum_{h=1}^{n}\sum_{w=1}^{k}\frac{w}{k}\overline{A}_{w,h}^{(n)}e^{-\alpha h} \\
&\leq \sum_{h=1}^{D}\sum_{w=1}^{k}\frac{w}{k}\overline{A}_{w,h}^{(n)} + \sum_{h>D}\sum_{w=1}^{k}\frac{w}{k}\overline{A}_{w,h}^{(n)}e^{-\alpha h} \\
&= Y^{(n)}(D) + \sum_{h>D}\sum_{w=1}^{k}\frac{w}{k}\overline{A}_{w,h}^{(n)}e^{-\alpha h} \\
&\leq Y^{(n)}(D) + \sum_{h>D}\sum_{w=1}^{k}\overline{A}_{w,h}^{(n)}e^{-\alpha h} \\
&= Y^{(n)}(D) + \sum_{h>D}\overline{A}_{h}^{(n)}e^{-\alpha h}. \qquad \text{(C.12)}
\end{aligned}$$

Theorem C.4 now follows immediately from (C.12) and the proof of Theorem 5.2. ■

Corollary C.1 *If in addition, $Y^{(n)}(D_n) = O(n^{-\beta})$, then for $\alpha > c_0$,*

$$P_b^{(n)} = O(n^{-\beta}) \qquad \text{(C.13)}$$

The following lemma shows why the results on word error probability can be easily extended to bit error probability. In essence, Lemma C.1 shows that $Z^{(n)}(D_n) = O(n^{-\beta})$ if and only if $Y^{(n)}(D_n) = O(n^{-\beta+1})$.

Lemma C.1 *There exists a positive constant μ, such that*

$$Z^{(n)}(D_n)/k \leq Y^{(n)}(D_n) \leq \mu D_n Z^{(n)}(D_n)/k. \qquad \text{(C.14)}$$

Proof: Applying $w/k \geq 1/k$ to (C.10), we obtain the left inequality. From Prop. C.1 we know that if $\overline{A}_{w,h}^{(n)} \neq 0$, then $w \leq \mu h$. Thus if $h \leq D_n$, and $\overline{A}_{w,h}^{(n)} \neq 0$, then $w \leq \mu h \leq \mu D_n$. The right-hand side inequality now follows if we upper bound w/k by $\mu D_n/k$ in (C.10). Finally, since $k = Rn$, where R is the rate of the ensemble, it follows as an immediate corollary that $Z^{(n)}(D_n) = O(n^{\beta})$ iff $Y^{(n)}(D_n) = O(n^{\beta-1})$. ■

Appendix D Miscellaneous Derivations for Chapter 6

D.1 Proof of Theorem 6.2

We first define the *ensemble threshold* as follows:

$$p_0 = \sup\{p : K(\delta, p) > r(\delta), 0 < \delta < 2p\}. \tag{D.1}$$

Lemma D.1 *If $p < p_o$, then there exist real numbers $\alpha_0 > 0$ and $\theta_0 > 0$, and a positive integer N_0, such that for $n \geq N_0$,*

$$\sum_{h=d_n}^{\alpha_0 n} \overline{A}_h^{(n)} \gamma^h = O(e^{-d_n \theta_0}),$$

where $\gamma = 2\sqrt{p(1-p)}$.

Proof: Using the definition 6.17, it is straightforward to show that

$$\lim_{\delta \to 0} \frac{K(\delta, p_0)}{\delta} = \frac{\partial K(0, p_0)}{\partial \delta} = -\log \gamma_0,$$

where $\gamma_0 = 2\sqrt{p_0(1-p_0)}$. Hence for $p < p_0$, we have

$$\begin{aligned} \limsup_{\delta \to 0} \frac{r(\delta)}{\delta} &\leq \lim_{\delta \to 0} \frac{K(\delta, p_0)}{\delta} \\ &= -\log \gamma_0 = -\log(2\sqrt{p_0(1-p_0)}) \\ &< -\log \gamma = -\log(2\sqrt{p(1-p)}). \end{aligned}$$

This, together with Assumption 2, implies that there exists $\alpha_0 > 0$, $\theta_0 > 0$, and a positive integer N_0 such that for $n \geq N_0$, we have

$$\sup_{d_n/n \leq \delta < \alpha_0} \frac{r_n(\delta)}{\delta} < \frac{n\theta_n}{d_n} + \sup_{0 \leq \delta < \alpha_0} \frac{r(\delta)}{\delta} < -\log\gamma - \theta_0.$$

Hence we have, for $n \geq N_0$,

$$\begin{aligned}
\sum_{h=d_n}^{\alpha_0 n} \overline{A}_n^{(n)} \gamma^h &= \sum_{h=d_n}^{\alpha_0 n} e^{-h(\log\gamma - r_n(\delta)/\delta)} \\
&< \sum_{h=d_n}^{\alpha_0 n} e^{-h\theta_0} \\
&< \sum_{h=d_n}^{\infty} e^{-h\theta_0} \\
&= O(e^{-d_n\theta_0}),
\end{aligned}$$

which completes the proof. ■

Now we can give the proof of Theorem 6.2. With the notation being as established above, we have, by Theorem 6.1, for $p < p_0$,

$$P_E \leq \sum_{h=1}^{d_n} \overline{A}_h^{(n)} + \sum_{h=d_n}^{\alpha_0 n} \overline{A}_h^{(n)} \gamma^h + \sum_{h=\alpha_0 n}^{n} \overline{A}_h^{(n)} P_h(T) + o(n). \tag{D.2}$$

The first sum in (D.2) approaches zero by Assumption 1, the second sum approaches zero by Lemma D.1 together with the fact that $d_n \to \infty$. The third sum is

$$\begin{aligned}
\sum_{h=\alpha_0 n}^{n} \overline{A}_h^{(n)} P_h(T) &= \sum_{h=\alpha_0 n}^{n} e^{-n(K(\delta,p) - r(\delta) + o(1))} \\
&\leq \sum_{h=\alpha_0 n}^{n} e^{-n(K(\delta,p) - K(\delta,p_0) + o(1))},
\end{aligned} \tag{D.3}$$

where the first line follows from (6.20) and Assumption 2, and the second line follows from the definition (D.1) of p_0.

Finally, let ϵ be such that

$$K(\delta, p) - K(\delta, p_0) \geq \epsilon \qquad \text{for } \alpha_o \leq \epsilon \leq 2p. \tag{D.4}$$

Then for n sufficiently large, the exponent in (D.3) will be $\geq \epsilon/2$, and so the sum willl be upper bounded by $n \cdot e^{-n\epsilon/2}$, which goes to zero. ■

D.2 Derivation of Equation (6.40)

Using Lagrange multiplier, we construct the function

$$\mathcal{J} = \sum_{i=1}^{k} p_i H(\frac{\delta_i}{2p_i}) + p_{-i} H(\frac{\delta_i}{2p_i}) + \lambda(\sum_i \delta_i - \delta), \tag{D.5}$$

and differentiating with respect to δ_i and setting to zero, we have

$$\log \frac{\sqrt{(2p_i - \delta_i)(2p_{-i} - \delta_i)}}{\delta_i} + \lambda = 0. \tag{D.6}$$

Hence the optimum point are (taking $c = e^{-2\lambda}$)

$$\delta_i^* = \frac{4p_i p_{-i}}{p_i + p_{-i} + \sqrt{(p_i + p_{-i})^2 + 4(1-c)p_i p_{-i}}}. \tag{D.7}$$

The parameter c is determined by the condition $\sum_{i=1}^{k} \delta_i^* = \delta$.

D.3 Proof of Theorem 6.5

The proof is a generalization of the proof for the BSC. We first define the *ensemble threshold* as follows:

$$\tau_0 = \sup\{\tau : K^{(k)}(\delta, \tau) > r(\delta), 0 < \delta < 1\}. \tag{D.8}$$

Lemma D.2 *If $\tau < \tau_o$, then there exist real numbers $\alpha_0 > 0$ and $\theta_0 > 0$, and a* *positive integer N_0, such that for $n \geq N_0$,*

$$\sum_{h=d_n}^{\alpha_0 n} \overline{A}_h^{(n)} \gamma_k^h = O(e^{-d_n \theta_0}),$$

where $\gamma_k = \sum_{i=1}^{k} 2\sqrt{p_i p_{-i}} \triangleq \gamma_k(\tau)$.

Proof: Similar to the proof of Lemma D.1, the key point is to prove that

$$\limsup_{\delta \to 0} \frac{r(\delta)}{\delta} \leq -\log \gamma_k(\tau_0) < -\log \gamma_k(\tau).$$

For that, we first show

Lemma D.3

$$\lim_{\delta \to 0} \frac{K^{(k)}(\delta, \tau)}{\delta} = -\log \gamma_k(\tau).$$

Proof: Using definition of $K^{(k)}(\delta, \tau)$ (6.40), we have

$$\begin{aligned}
\lim_{\delta \to 0} \frac{K^{(k)}(\delta, \tau)}{\delta} &= \frac{\partial K^{(k)}(\delta, \tau)}{\partial \delta} |_{\delta \to 0} \\
&= \log \frac{1-\delta}{\delta} - \sum_{i=1}^{k} \left(\log \frac{\sqrt{(2p_i - \delta_i^*)(2p_{-i} - \delta_i^*)}}{\delta_i^*} \right) \frac{\partial \delta_i^*}{\partial \delta} |_{\delta \to 0} \\
&= \log \frac{1-\delta(c)}{\delta(c)} - \sum_{i=1}^{k} \log \sqrt{c} \frac{\partial \delta_i^*}{\partial \delta} |_{c \to \infty} \\
&= \log(1-\delta(c)) - \sum_{i=1}^{k} \log(\sqrt{c}\delta(c)) \frac{\partial \delta_i^*}{\partial \delta} |_{c \to \infty}. \qquad \text{(D.9)}
\end{aligned}$$

The third line follows by plugging (D.6) and realizing δ is a function of c which approaches 0 as $c \to \infty$. It is easy to verify (by (D.7))

$$\lim_{c \to \infty} \sqrt{c}\delta(c) = \sum_{i=1}^{k} 2\sqrt{p_i p_{-i}}, \qquad \text{(D.10)}$$

and

$$\frac{\partial \delta_i^*}{\partial \delta}|_{c\to\infty} = \frac{2\sqrt{p_i p_{-i}}}{\sum_i 2\sqrt{p_i p_{-i}}}. \tag{D.11}$$

Plugging (D.10) and (D.11) into (D.9), we have

$$\lim_{\delta\to 0}\frac{K(\delta,\tau)}{\delta} = -\log\sum_i^k 2\sqrt{p_i p_{-i}} = -\log\gamma_k(\tau).$$

■

Now, we give the proof of Lemma D.2. For $\tau < \tau_0$, we have

$$\begin{aligned}\limsup_{\delta\to 0}\frac{r(\delta)}{\delta} &\leq \lim_{\delta\to 0}\frac{K^{(k)}(\delta,\tau_0)}{\delta}\\ &= -\log\gamma_k(\tau_0) = -\log(\sum_{i=1}^k 2\sqrt{p_i(\tau_0)p_{-i}(\tau_0)})\\ &< -\log\gamma_k(\tau) = -\log(\sum_{i=1}^k 2\sqrt{p_\tau(i)p_\tau(-i)}).\end{aligned}$$

From this point on, the proof for Lemma D.2 is identical to the proof in Lemma D.1 except γ should be changed to γ_k.

Given Lemma D.2, the proof for Theorem 6.5 is the same as to the one for Theorem 6.1. ■

D.4 Proof of Theorem 6.8

Lemma D.4 *For an k-quantized channel C, $\gamma_k(\sigma) \geq e^{-RE_b/N_0} = e^{-\frac{1}{2\sigma^2}}$.*

Proof: First, assume channel C has dividing sequence $\mathbf{s}$. From the observation followed after Example 6.6, we can build a squence of channels starting with the current quantized channel and satisfying (6.46) and (6.47). Now we show that for such a sequence of channels, $\gamma_{k_j}(\sigma)$ is decreasing with respect to j. Because $\gamma_{k_j}(\sigma) =$

$\sum_{i=1}^{k_j} 2\sqrt{p(v_i)p(-v_i)}$, and

$$\begin{aligned}\gamma_{k_{j+1}} &= \sum_{i=1}^{k_{j+1}} 2\sqrt{p(v_i')p(-v_i')} \\ &= \sum_{i=1}^{k_j}\sum_{m=1}^{m_i} 2\sqrt{p(v_i')p(-v_i')}, \qquad \text{according to the partition}\end{aligned}$$

it suffices to show

$$\sqrt{p(v_i)p(v_{-i})} \geq \sum_{m=1}^{m_i} \sqrt{p(v_i')p(-v_i')}, \tag{D.12}$$

given

$$p(v_i) = \sum_{m=1}^{m_i} p(v_i'), \qquad p(-v_i) = \sum_{m=1}^{m_i} p(-v_i'). \tag{D.13}$$

This can be easily seen (e.g., by applying Cauchy-Schawz inequality). Now, because

$$\gamma_\infty = 2\int_0^\infty p_z(y-1)p_z(y+1)dy = e^{-\frac{1}{2\sigma^2}} = e^{-RE_b/N_0},$$

we have the conclusion in Lemma D.4. ∎

Now we define the *ensemble threshold* of AWGN channel as follows:

$$\sigma_0 = \sup\{\sigma : K^{(\infty)}(\delta,\sigma) > r(\delta), 0 < \delta < 1\}. \tag{D.14}$$

Lemma D.5 *On an AWGN chanel, if $\sigma < \sigma_o$, there exist real numbers $\alpha_0 > 0$ and $\theta_0 > 0$, and a positive integer N_0, such that for $n \geq N_0$,*

$$\sum_{h=d_n}^{\alpha_0 n} \overline{A}_h^{(n)} \gamma^h = O(e^{-d_n\theta_0}).$$

Proof: Again, the key point is to show that

$$\limsup_{\delta\to 0} \frac{r(\delta)}{\delta} < \frac{1}{2\sigma^2}.$$

Because $\sigma < \sigma_0$, by definition (D.14), for a fixed $\sigma_1 \in (\sigma, \sigma_0)$, there exists a k-quantized channel C_k, such that

$$K^{(k)}(\delta, \sigma_1) \geq r(\delta) \qquad \text{for any } \delta. \tag{D.15}$$

Hence we have

$$\begin{aligned}
\limsup_{\delta\to 0} \frac{r(\delta)}{\delta} &\leq \limsup_{\delta\to 0} \frac{K^{(k)}(\delta, \sigma_1)}{\delta} \\
&= -\log \gamma_k(\sigma_1) \qquad \text{By Lemma D.3} \\
&\leq \frac{1}{2\sigma_1^2} \qquad \text{By Lemma D.4} \\
&< \frac{1}{2\sigma^2}.
\end{aligned}$$

The rest of the proof is identical to the proof of Lemma D.1. ■

Now we can give the proof of Theorem 6.8. Suppose that given $\sigma_1 \in (\sigma, \sigma_0)$, a k-quantized channel C_k has

$$K^{(k)}(\delta, \sigma_1) \geq r(\delta) \qquad \text{for any } \delta. \tag{D.16}$$

We now do the typical pairs decoding on the k-quantized channel C_k. With the notation being as established above, we have, by Theorem 6.1, for $\sigma < \sigma_0$,

$$P_E \leq \sum_{h=1}^{d_n} \overline{A}_h^{(n)} + \sum_{h=d_n}^{\alpha_0 n} \overline{A}_h^{(n)} \gamma^h + \sum_{h=\alpha_0 n}^{n} \overline{A}_h^{(n)} P_h(T^{(k)}) + o(n). \tag{D.17}$$

The first sum in (D.17) approaches zero by Assumption 1, the second sum approaches zero by Lemma D.5 together with the fact that $d_n \to \infty$. The third sum is

$$\begin{aligned}
\sum_{h=\alpha_0 n}^{n} \overline{A}_h^{(n)} P_h(T^{(k)}) &= \sum_{h=\alpha_0 n}^{n} e^{-n(K^{(k)}(\delta,\sigma) - r(\delta) + o(1))} \\
&\leq \sum_{h=\alpha_0 n}^{n} e^{-n(K^{(k)}(\delta,\sigma) - K^{(k)}(\delta,\sigma_1) + o(1))},
\end{aligned} \tag{D.18}$$

where the first line follows from (6.20) and Assumption 2, and the second line follows from the definition (D.14) of σ_0.

Finally, let ϵ be such that

$$K^{(k)}(\delta, \sigma) - K^{(k)}(\delta, \sigma_1) \geq \epsilon \qquad \text{for } \alpha_o \leq \epsilon. \tag{D.19}$$

Then for n sufficiently large, the exponent in (D.18) will be $\geq \epsilon/2$, and so the sum will be upper bounded by $n \cdot e^{-n\epsilon/2}$, which goes to zero. ∎

Appendix E Hardware Implementation of Iterative Decoding Algorithm of RA Codes

This appendix considers a hardware implementation of the RA codes iterative decoding algorithm defined in Section 3.3.

We assume messages are 4-bit integers $(m_1m_2m_3m_4) \in \{0,1\}^4$, which we use the convention that the most significant bit m_1 represents a guess on the value of a variable, and $(m_2m_3m_4)$ indicates how reliable the guess is. For example, a message (0111) carries a strong belief that the variable is 0, while a message (1000) says the variable is likely to be 1 without any confidence. This quantization from real to 4-bit value causes a degradation in performance about 0.5 dB.

In this appendix, we focus on the module that executes one iteration in the iterative decoding. To obtain multiple iterations we can either pipeline this module or feed its own outputs back as its inputs. One iteration, in brief, has two main steps: to update messages at check nodes and to update messages at variable nodes. Between those two steps, some messages are exchanged via a pseudo-random permutation which connects the information bits to the parity nodes. It turns out that in this module the computation do not take many gates, it is the permutation step that invokes lots of memory and hence dominates the number of gates on the chip. Usually, an additional block of $N \times 1$ memory is needed to store the permutation, but in this implementation, an LCM interleaver [27] is used to save that memory.

We assume that messages are coming toward check nodes (see Figure 3.4) at the beginning of each iteration. Hence messages are first updated at the check nodes, and then passed to the information bits (with permutation) and parity bits (without permutation). Then messages are updated at the variable nodes before passed back to the check nodes. For simplicity, let's label different messages. First, we order the check nodes as $1, \ldots, N$ from the top to the bottom. Consider a check node k, we name the inward message from some information bit as $AI[k]$, the message from the up-branch parity bit as $CI[k-1]$, and the message from the low-branch parity bit as $BI[k]$. Channel information for those parity bits are called as $RV[k]$. Figure E.1 shows our conventions.

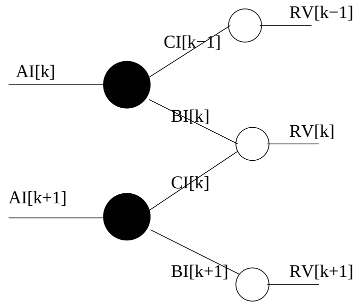

Figure E.1: Notation for the messages.

We assume data feed into this one-iteration module serially. An 'enable' line indicates when a block of N data starts and ends. Using all those notations, a one-iteration decoding process can be decomposed in Figure E.2. Let's consider the data flow of each message. Messages AI are first updated at the check nodes (module CHK-UPDATE), then are permuted (module INT) before updated at the variable nodes (module LV-UPDATE). Those updated messages are then de-permuted (module DEINT) and coming back towards check nodes. Similarly, messages BI, CI are first updated at the check nodes (module CHK-UPDATE), then updated at the variable nodes (module RV-UPDATE). To synchronize them with messages AI, we have to delay them (module DELAY).

In the following, we will briefly go through the design of each module.

Module: CHK-UPDATE.

This module updates messages at the check nodes. The updating rule is a quantized approximation to the exact updating rules in Section 3.3.2. In the updating, the most significant bit of the resulting message is easy to determine. By the definition, a check

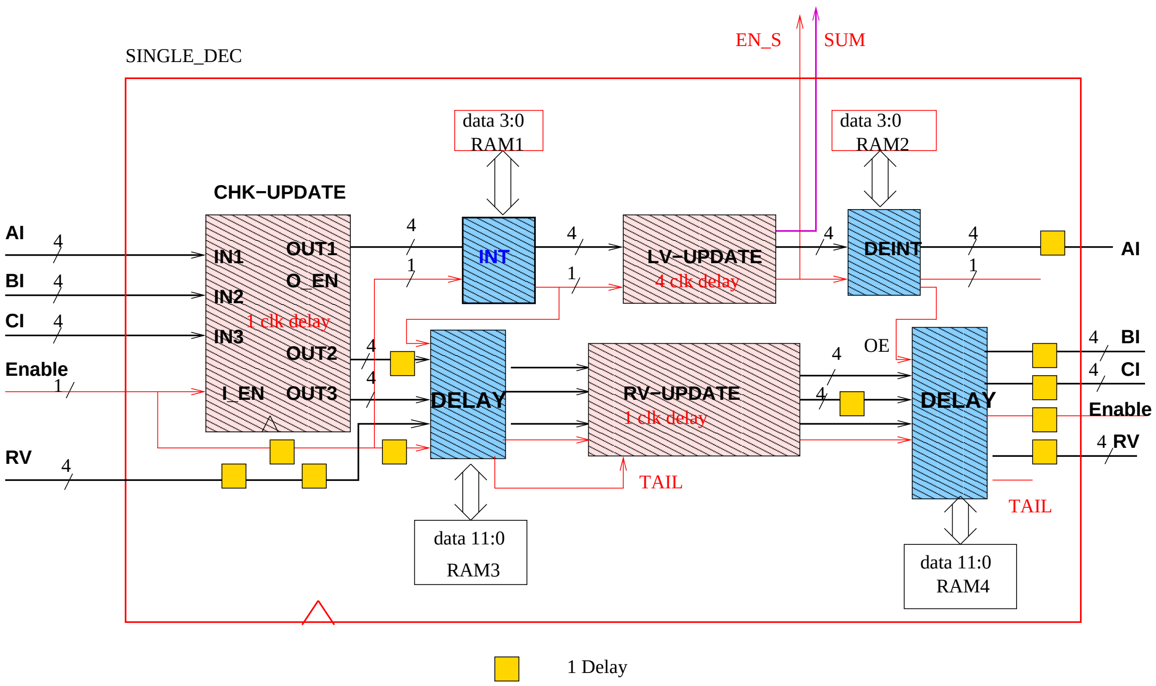

Figure E.2: Data flow of one iteration in decoding RA codes.

node checks the parity of all its incoming bits, hence the resulting message should have the most significant bit satisfy this condition, namely it should be the XOR of the two incoming messages' most significant bits. The other three bits indicate how strong this belief is. Table E.1 shows the updating rules as a function $f(a, b)$, where both its domain and range are 3-bit vectors. Because of symmetry, it is sufficient to show half of the entries.

a	1	1	2	2	3	3	4	4	5	5	6	7
b	1	≥ 2	2,3	≥ 4	3,4	5, 6, 7	4, 5	6,7	5	6, 7	6, 7	7
$f(a, b)$	0	1	1	2	2	3	3	4	4	5	6	7

Table E.1: Updating rule at a check node.

Module: INT.

This module permutes AI. The selected permutation is an LCM interleaver[27]. Its basic idea is Fermat's little theorem, which says that if P is a prime number, then $a^k, k = 1, 2, \ldots, P-1$ modular P is a permutation of $1, 2, \ldots, P-1$ for any $2 \leq a \leq P-1$. Therefore, if we use block length $N = P-1$ for some prime number P, we could have a very simple permutation. Figure E.3 shows an implementation of this module, which also includes an external memory to store data during permutation.

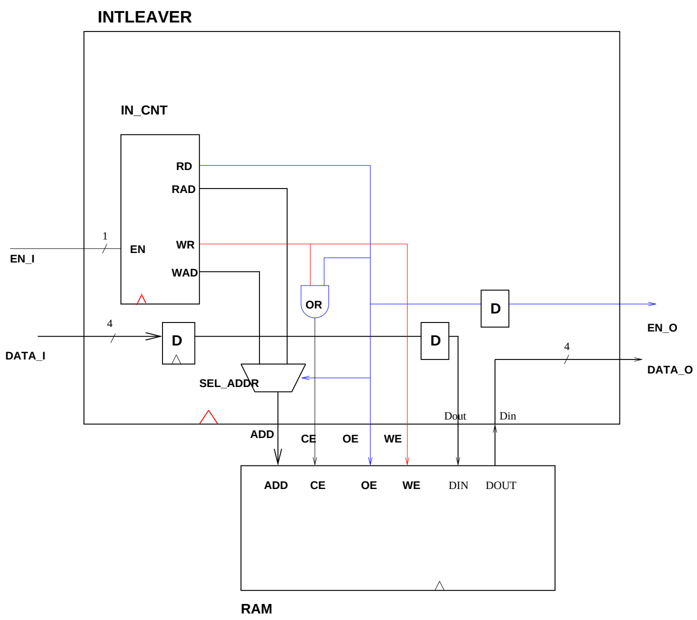

Figure E.3: Data flow of INT module.

In this module, submodule IN-CNT is the core. It generates WE (write enable) and OE (read enable) and ADDR (address) signals. This is done by using two counters, one outputs $1, 2, \ldots, N$ and the other outputs $a, a^2, \ldots, a^N$.

Module DEINT does the de-interleaving. Its implementation is almost identical to INT and hence omitted.

Module: LV-UPDATE

This module updates messages at the information bits. Since each information bit is repeated q times, messages cannot be updated until all the messages of the same information bit arrive. Figure E.4 shows an implementation of this module for $q = 3$.

Inside this module, submodule LV-CAL does the actual messages updating at the information bits. We simply take the addition of the messages and truncate them to

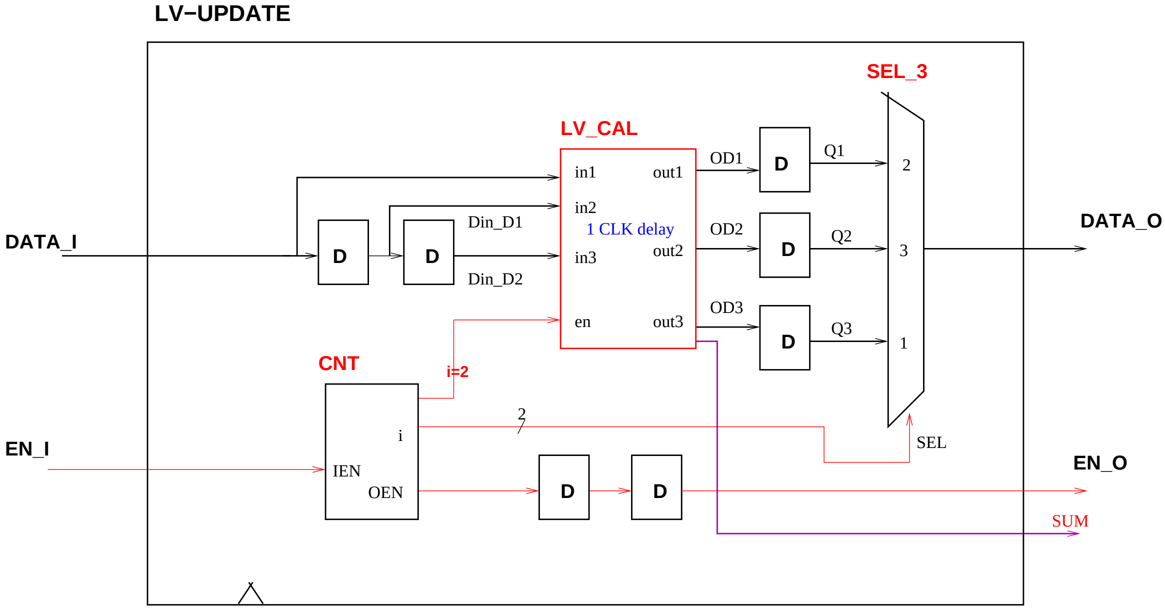

Figure E.4: Data flow of LV-UPDATE module.

magnitude $\in [-7, +7]$. Module RV-UPDATE is almost identical to this module and hence omitted.

Module: DELAY

Module DELAY stores a block of data in memory and outputs that when the data is needed. In structure it resembles the module INT. Figure E.5 shows an implementation.

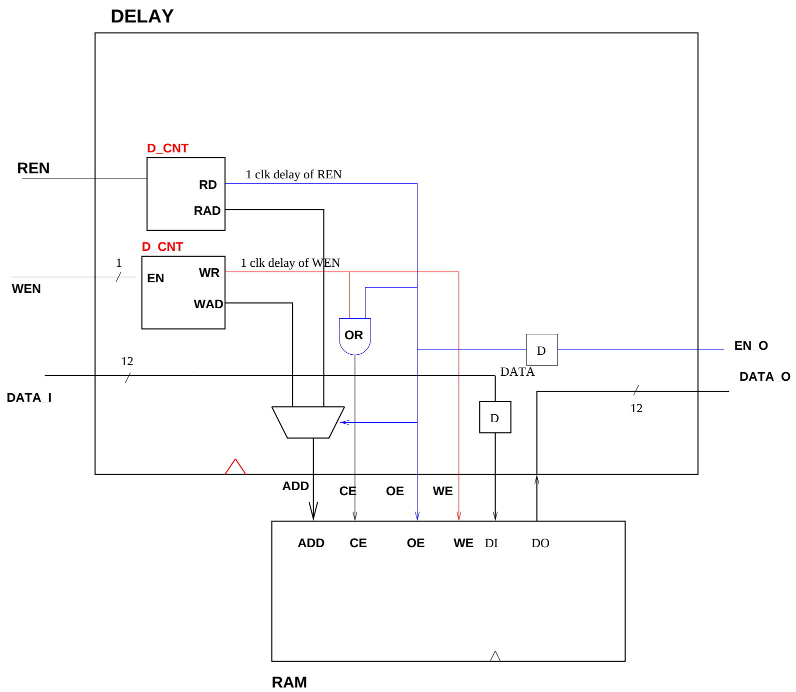

Figure E.5: Data flow of DELAY module.

Bibliography

[1] S. Aji, *Graphical models and iterative decoding*, Ph.D. thesis, California Institute of Technology, Pasadena, 2000.

[2] S. M. Aji, H. Jin, A. Khandekar, D. MacKay, and R. J. McEliece, "BSC Thresholds for Code Ensembles based on 'Typical Pairs' Decoding," *Proc. IMA Workshop on Codes and Graphs*, August 1999, pp. 195–210.

[3] S. Aji and R. J. McEliece, "The Generalized Distributed Law," *IEEE Trans. on Info. Theory*, vol. 32, no. 1, March 2000, pp. 325–343.

[4] S. Benedetto and G. Montorsi, "Unveiling turbo codes: some results on parallel concatenated coding schemes," *IEEE Trans. on Info. Theory*, vol. 42, no. 2, March 1996, pp. 409–428.

[5] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: turbo codes," *Proc. 1993 IEEE International Conference on Communications*, Geneva, Switzerland, May 1993, pp. 1064–1070.

[6] S. Benedetto and G. Montorsi, "Unveiling turbo codes: some results on parallel concatenated coding schemes," *IEEE Trans. on Info. Theory*, vol. 42, no. 2, March 1996, pp. 409–428.

[7] S. Benedetto and G. Montorsi, "Design of parallel concatenated convolutional codes," *IEEE Transactions on Communications*, vol. 44, no. 5, May 1996, pp. 591–600.

[8] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: performance analysis, design, and iterative decoding," *IEEE Trans. on Info. Theory*, vol. 44, no. 3, May 1998, pp. 909–926.

[9] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Analysis, Design, and Iterative Decoding of Double Serially Concatenated Codes with Interleavers," *IEEE J. on Selected Areas in Comm.*, vol. 16, no. 2, February 1998, pp. 231–244.

[10] S. ten Brink, "Rate one-half code for approaching the Shannon limit by 0.1 dB," *Electr. Lett.*, vol. 36, no. 15, July 2000, pp. 1293–1294.

[11] J. F. Cheng, *Iterative Decoding*, Ph.D. thesis, California Institute of Technology, Pasadena, 1998.

[12] S.-Y. Chung, R. Urbanke, and T. J. Richardson, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. on Info. Theory*, vol. 47, no. 2, February 2001, pp. 657–671.

[13] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: John Wiley and Sons, 1991.

[14] L. Decreusefond and G. Zémor, "On the error-correcting capabilities of cycle codes of graphs," *Combinatorics, Probability, and Computing*, vol. 6, 1997, pp. 27–38.

[15] D. Divsalar: "A simple tight bound on error probability of block codes with application to turbo codes," *JPL TMO Progress Report 42-139*, November 1999, pp. 1–35.

[16] D. Divsalar, S. Dolinar, H. Jin, and R. J. McEliece, "AWGN coding theorems from ensemble weight enumerators," *Proc. ISIT 2000*, p. 458.

[17] D. Divsalar, H. Jin and R. J. McEliece, "Coding theorems for 'Turbo–Like' Codes," *Proc. 36th Allerton Conf. on Communication, Control and Computing*, September 1998, pp. 201-210.

[18] D. Divsalar and R. J. McEliece, “On the design of concatenated coding systems with interleavers,” *JPL TMO Progress Report vol. 2-134*, August 1998, pp. 1–22. (http://tmo.jpl.nasa.gov/tmo/progress_report/42-134/134D.pdf.)

[19] D. Divsalar, S. Dolinar, and F. Pollara, “Iterative turbo decoder analysis based on Gaussian density evolution,” submitted to *IEEE J. Selected Areas in Comm.*

[20] D. Divsalar and F. Pollara, “On the design of turbo codes,” *TDA Progress Report* vol. 42-123, November 1995, pp. 99–121.

[21] S. Dolinar, L. Ekroot, and F. Pollara, “Improved Error Probability Bounds for Block Codes for the Gaussian Channel,” *Proc. ISIT 1994*, p. 243.

[22] G. D. Forney, Jr., “Convolutional Codes 1: Algebraic Structure,” *IEEE Trans. on Info. Theory*, vol. 17, 1970, pp. 720-738.

[23] G. D. Forney, Jr., “Codes on graphs: news and views,” *Proc. 2nd International Symposium on Turbo Codes*, Brest France, September 2000, pp. 9–16.

[24] R. Gallager, *Low-Density Parity-Check Codes*. Cambridge, Mass.: The M.I.T. Press, 1963.

[25] J. Hagenauer, E. Offer, and L. Papke, “Iterative Decoding of Binary Block and Convolutional Codes,” *IEEE Trans. on Info. Theory*, vol. 42, no. 2, March 1996, pp. 429–445.

[26] E. K. Hall and S. G. Wilson, “Design and Analysis of Turbo Codes on Rayleigh Fading Channels,” *IEEE J. Selected Areas in Comm.*, vol. 16, No. 2, February 1998, pp. 160–174.

[27] D. Hatori, J. Murayama, and R. J. McEliece, “Pseudorandom and self-terminating interleavers for turbo codes,” *1998 Information Theory Workshop*, San Diego CA, February 1998.

[28] G. B. Horn, “The iterative decoding of cycle codes,” submitted to *IEEE Trans. Inform. Theory.*

[29] H. Jin, A. Kandekkar, and R. J. McEliece, "Irregular Repeat-Accumulative Codes," *Proc. 2nd International Symposium on Turbo Codes*, Brest France, September 2000, pp. 1–8.

[30] H. Jin and R. J. McEliece, "RA codes achieve AWGN Channel Capacity," *13th Symp. on Applied Algebra, Algebraic Algorithms and Error Correcting Codes*, Hawaii, November 1999, pp. 10–18.

[31] H. Jin and R. J. McEliece, "AWGN coding theorems for serial turbo codes," *Proc. 37th Allerton Conf. on Communication, Computation and Control*, September 1999, pp. 893–894.

[32] H. Jin and R. J. McEliece, "Typical pairs decoding on the AWGN channel," *Proc. 2000 International Symp. on Info. Theory and its Applications,* Hawaii, November 2000, pp. 180–183.

[33] D. Jungnickel and S. A. Vanstone, "Graphical codes revisited," *IEEE Trans. on Info. Theory*, vol. 43, January 1997, pp. 136–146.

[34] F. R. Kschischang, B. J. Frey, and H. Loeliger, "Factor Graphs and the Sum-Product Algorithm," *IEEE Trans. on Info. Theory*, vol. 47, no. 2, February 2001, pp. 498–519.

[35] N. Kahale and Rüdiger Urbanke, "On the minimum distance of parallel and serially concatenated codes," submitted to *IEEE Trans. on Info. Theory.*

[36] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann, "Practical loss-resilient codes," *Proc. 29th ACM Symp. on the Theory of Computing*, 1997, pp. 150–159.

[37] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Analysis of low-density codes and improved designs using irregular graphs," *Proc. 30th ACM Symp. on the Theory of Computing*, 1998, pp. 249–258.

[38] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. on Info. Theory*, vol. 45, March 1999, pp. 399–431.

[39] R. J. McEliece, *The Theory of Information and Coding*. Reading, Mass.: Addison-Wesley, 1977.

[40] R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, "Turbo decoding as as instance of Pearl's 'Belief Propagation' algorithm," *IEEE J. Selected Areas in Comm.*, vol. 16, no. 2, February 1998, pp. 140–152.

[41] W. W. Peterson and E. J. Weldon, Jr., *Error-Correcting Codes, 2nd. ed.* Cambridge, Mass.: The MIT Press, 1972.

[42] H. Pfister and P. Siegel, "The serial concatenation of rate-1 codes through uniform random interleavers," *Proc. 37th Allerton Conf. on Communication, control, and computation*, 1999, pp. 260-269.

[43] T. J. Richardson, A. Shokrollahi, and R. Urbanke, "Design of provably good low-density parity-check codes," *IEEE Trans. on Info. Theory*, vol. 47, no. 2, February 2001, pp. 619–637.

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

[45] T. Richardson and R. Urbanke, "Thresholds for turbo codes," *Proc. ISIT 2000*, p. 317.

[46] C. E. Shannon,"A Mathematical Theory of Communication," *Bell system Technical Journal,* 1948.

[47] M. A. Shokrollahi, "New sequences of linear time erasure codes approaching channel capacity," *Proc. 1999 AAECC*, Hawaii, November 1999, pp. 65–76.

[48] M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. on Info. Theory*, vol. 27, September 1981, pp. 533-547.

[49] S. Vialle and J. Boutros, "Performance limits of concatenated codes with iterative decoding," *Proc. ISIT 2000*, p. 150.

[50] A. J. Viterbi and A. M. Viterbi, "Improved union bound on linear codes for the input-binary AWGN channel, with applications to turbo decoding," *Proc. Winter 1998 Information Theory Workshop,* San Diego, California, February 1998, p. 72.

[51] A. J. Viterbi and A. M. Viterbi, "New Results on Serial Turbo Code and Accumulated Convolutional Code Performance," preprint.

[52] N. Wiberg, *Codes and Decoding on General Graphs*. Linköping Studies in Science and Technology. Dissertation no. 440. Linköping University, Linköping, Sweden, 1996.