LOW-COMPLEXITY, CAPACITY-APPROACHING CODING SCHEMES: DESIGN, ANALYSIS AND APPLICATIONS

A Dissertation

by

JING LI

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 2002

Major Subject: Electrical Engineering

LOW-COMPLEXITY, CAPACITY-APPROACHING CODING SCHEMES:

DESIGN, ANALYSIS AND APPLICATIONS

A ack,biblio

by

JING LI

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Approved as to style and content by:

Krishna R. Narayanan (Co-Chair of Committee)	Costas N. Georghiades (Co-Chair of Committee)
Andrew K. Chan (Member)	Wei Zhao (Member)
(Member)	Chanan Singh (Head of Department)

December 2002

Major Subject: Electrical Engineering

ABSTRACT

Low-Complexity, Capacity-Approaching Coding Schemes:

Design, Analysis and Applications. (December 2002)

Jing Li, B.Sc., Peking University, Beijing, China;

M.Eng., Texas A&M University

Co–Chairs of Advisory Committee: Dr. Krishna R. Narayanan
Dr. Costas N. Georghiades

The discovery (and rediscovery) of turbo codes and low density parity check (LDPC) codes has revolutionized the coding research with novel ideas and techniques on code concatenation, iterative decoding and graph-based structure. Although these remarkable codes have demonstrated performance very close to the Shannon limit on memoryless channels, it is fair to say that practice still lags behind theory by some margin due to constructional difficulty, decoding complexity, structural irregularity and/or other implementation issues. This work endeavors to fill some of these gaps concerning the design, analysis, application and evaluation of simple but good codes. The primary interest is to construct a class of bandwidth-efficient and power-efficient "good" codes that are simple to construct, implement and analyze, and to investigate related issues concerning their applications.

We direct our research focus on graph-based, soft iteratively decodable codes, particularly those that are of high performance and low complexity. The work is composed of various pieces, but the main components are developed within the following two main themes.

1. The first is to design and analyze the proposed codes, namely *product accu-*

mulate codes or PA codes, from the code theoretic perspective, which includes structural properties, decoding algorithms and efficiency, performance bounds and asymptotic thresholds.

2. The second is to investigate and evaluate PA codes as well as other state-of-the-art codes like turbo product codes and LDPC codes in a number of application scenarios, including wireless communications, optical fiber communications, digital data storage systems and packet data networks. The features we will address include performance bounds and thresholds, rate compatibility, joint differential detection and decoding, joint decoding and equalization, binary precoding, and thresholds of coded intersymbol interference channels.

The former is more coding theory centric and the latter is more application oriented. Through the study, we hope to shed insight into the understanding and practice of coding theory and techniques in a unified framework, and to reveal and bench-mark the potentials of some of the best-known codes in a variety of applications.

To my parents and my advisors.

ACKNOWLEDGMENTS

Looking back at the years I spent at Texas A&M University, I cannot help feeling blessed for being able to work with intelligent and inspiring people. My deepest gratitude and respect first go to my advisors Prof. Krishna R. Narayanan and Prof. Costas N. Georghiades. It was their wonderful lectures on communications and coding theory (ELEN 455 Digital Communications by Prof. Georghiades and ELEN 689 Advanced topics on Channel Coding by Prof. Narayanan) that first attracted me into this general field of telecommunications and later into this specific area of error correction coding. During this long tournament, I have had several ups and downs where I struggled to grow personally and professionally. Had it not been for their valuable advice, direction, patience, encouragement, and other unconditional support, this work could not have been completed.

Various mentors and managers have guided and influenced me during the course of my graduate studies, including Dr. Erozan Kurtas at Seagate Technology, with whom I have had the pleasure of working on several projects on error correction coding for data storage systems; Dr. Alexei Pilipetskii, Dr. Alan Lucero, Dr. Gerald Lenner and Dr. Yi Cai at Tyco Telecommunications, where I spent a summer working on DSP and FEC for long haul optical fiber communications. This work includes and concludes some of the research work I have collaborated with them. I would like to recognize the wisdom and support of Prof. Xiaodong Wang of Columbia University, Prof. Andrew Chan, Prof. Scott Miller, Prof. Chanan Singh, Prof. Wei Zhao, Prof. Bill Stout and Prof. Zixiang Xiong at Texas A&M University, Dr. Gregory Silvus, Dr. Robert Lynch and Dr. Ganping Ju at Seagate Technology, Prof. Rick Blum, Prof. Bruce Fritchman and Prof. Kenneth Tzeng at Lehigh University, and Prof. Marc Fossorier at University of Hawaii.

My heartfelt thanks also go to my parents, family and friends who have offered generous love, care and help in times of need. A special thanks goes to Frank Zhang, Nancy Yang, Ben Lu, Dong Guo, Vivek Gulati, Yongzhe Xie and Tom Cassaro for the spiritual support as well as the many helpful discussions in my research work. A special thanks goes to Lingjuan Wang, Hugh Wang, Yingsong Wang, Gangping Ju, Christy Butch, Shan Jin, John Cao, Huaidong Meng, Honggeng Zhou, Daming Zhang, Jing Zhang, Qinghua Li and Ruiyu Wang, Zhongmin Liu and Hunter Weng, for giving me the strength to accomplish the things I can and the grace to accept the things I cannot. Sharing with them the joy and frustration has made a graduate student's life fruitful and complete.

I have also benefited greatly from the gracious and enriching educational opportunities and support from Texas A&M University, Peking University (Beida) and Hangzhou Foreign Language School (HFLS). I am very grateful to all the inspiring teachers and mentors along my learning process including Prof. Liangzhi Wu (BeiDa), Huizhi Huang (HFLS), Lina Zhang (HFLS) and many others. I am fortunate to have encountered so many.

This dissertation is dedicated to my advisors, Prof. Krishna R. Narayanan and Prof. Costas N. Georghiades, and my parents, Shuxin and Jun. Although my parents do not read English and do not really understand the research work I am conducting, their importance to me I shall not try to put in words.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

FIGURE Page

FIGURE Page

FIGURE Page

CHAPTER I

INTRODUCTION

A. Introduction

Fifty years after Claude Shannon determined the ultimate capacity limit of memoryless channels, we have finally constructed practical error correction coding schemes that yield capacity approaching performance. Although random coding theory implies that arbitrary codes tend to perform close to the capacity as the code length goes to infinity, it is not until the discovery of turbo codes and the rediscovery of low density parity check (LDPC) codes in recent years that performance that is very close to the capacity limit has been achieved. However, for commercial deployment, it is fair to say that encoding/decoding complexity, structural complexity, and other issues remain to be solved for efficient and cheap implementation in hardware.

This work is primarily concerned with the construction, analysis and evaluation of high-performance error correction codes, and particularly those low-complexity codes based on graphs. The goal is firstly to design a class of low-complexity, high-performance, high-rate, and easily-implementable codes for use in applications where powerful error protection is needed but where high cost can not be afforded. Then we wish to investigate and evaluate some of the state-of-the-art codes and coding techniques for several applications including wireless communications, optical fiber communications (OFC), digital data recording systems and packet data networks.

The journal model is *IEEE Transactions on Automatic Control.*

B. State of the Art

The error correction coding techniques can be roughly divided into three categories.

The first is block codes using algebraic decoding, like Hamming codes [1], BCH (Bose-Chaudhuri-Hocquenghem) codes [2] [3], and RS (Reed-Solomon) codes [4]. These block codes can be implemented efficiently in hardware, but most of them lack a convenient soft decoding algorithm. Further it is not always possible to change code lengths.

The second category is convolutional codes using trellis decoding or list decoding. Unlike block codes, convolutional codes are convenient to change code lengths, and they have efficient soft decoding algorithms, like soft output Viterbi algorithm (SOVA) [5] and *a posteriori* probability (APP) algorithm [6] (whose hard output algorithm is known as maximum *a posteriori* or MAP), which is of great advantage on fading channels and/or with high-order modulations. The technique of puncturing further allows convolutional codes to change code rates flexibly without inducing extra complexity.

The third category, and possibly the most exciting of all, is the recently developed concatenated and compound codes which are built upon block and/or convolutional codes and which use (soft) iterative decoding. It was first found by Forney in 1966 [7] that by concatenating an inner code and an outer code, it is possible to construct codes whose probability of error decreases exponentially at rates less than capacity, while decoding complexity increases only algebraically. The discovery of turbo codes by Berrou *et al.* [8] and the rediscovery of low density parity check codes [9] by MacKay, Luby and Richardson *et al.* [10] [11] [12] are the milestones in the advancement of this field. Prompted by their impressively near-capacity performance [13] [14], parallel concatenated convolutional codes (PCCC) [15], serial concatenated convolutional

codes (SCCC) [16], hybrid concatenated convolutional codes (HCCC) [17] and other concatenated codes are proposed and shown to provide similar coding gains. Examples include turbo product codes (TPC) [18] [19] [20] and regular/irregular repeat accumulate (RA/IRA) codes [21] [22]. Specifically, *product accumulate* (PA) codes [23] [24], which is the focus and the primary contribution of this work also belongs to this category. These codes have two major features in common: (1) A (random) interleaver is usually incorporated (explicitly or implicitly) in the code structure, (2) soft-in soft-out (SISO) iterative decoding techniques are exploited to approximate the performance of an optimal (yet prohibitively complex) decoder. Analysis reveals that they possess good distance spectrum and that iterative decoding provides a good approximation to the otherwise unmanageable maximum likelihood (ML) decoding.

As the title indicates, the primary interest of this dissertation is to study the high-performance coding schemes in this third category, particularly those having (relatively) low complexity and analyzable using code graphs. Specifically, the proposition and discussion of product accumulate codes will be the focus. We plan to study their theoretical foundation, the structural properties, iterative decoding algorithms, and applications. Other graph-based codes like LDPC codes and TPC codes will also be investigated in the context of wireless communications, digital data storage systems, and packet data networks.

C. Background

1. Serial and Parallel Turbo Codes

The original turbo codes proposed by Berrou *et al.* [8] are composed of 2 branches of parallel concatenated convolutional codes with a random interleaver between them. In the general sense, turbo codes can be referred to as either parallel or serial concate-

nated convolutional codes with random interleavers between component codes. (see Figure 1). Puncturing can be used to get high rate turbo codes from a rate-1/3 or rate-1/2 mother code. Using an iterative soft decoding (see decoder structure in Figure 2), the above interleaved concatenated codes can yield performance impressively close to the capacity limit.

The key structural elements in parallel and serial turbo codes include the choice of the component codes and the use of a (random) interleaver in-between the component codes. Recursive systematic convolutional (RSC) codes are used in both branches of a parallel turbo code, as well as the inner code of a serial turbo code [25] [21]. Unlike non-recursive convolutional codes, a single weight input sequence does not cause an error event (which is defined as a patch in trellis that diverges and remerges to the reference path) in RSC codes and, hence, the minimum input weight that will cause error events for RSC codes is 2. If we are able to separate the two input weights far apart (before sending the sequence to RSC codes), chances are that the resulting error event will be long enough to be detected and/or corrected. This is made possible by the random interleaver used between the component codes. In a parallel concatenation, the random interleaver makes the input sequence that produces low weight output in the first branch produce (with high probability) high weight output in the second branch, and vice versa. Similarly, in a serial concatenation, the random interleaver makes the low weight sequences at the output of the outer code produce (with high probability) high weight sequences at the output of the inner RSC code. This is known as the *spectrum thinning* phenomenon and accounts for the well-known *interleaving gain*.

The decoding of turbo codes is an iterative approach based on the subdecoders of the component codes, see Figure 2). The subdecoders can implement any SISO algorithm, but most popular ones are the BCJR algorithm or the SOVA. The turbo

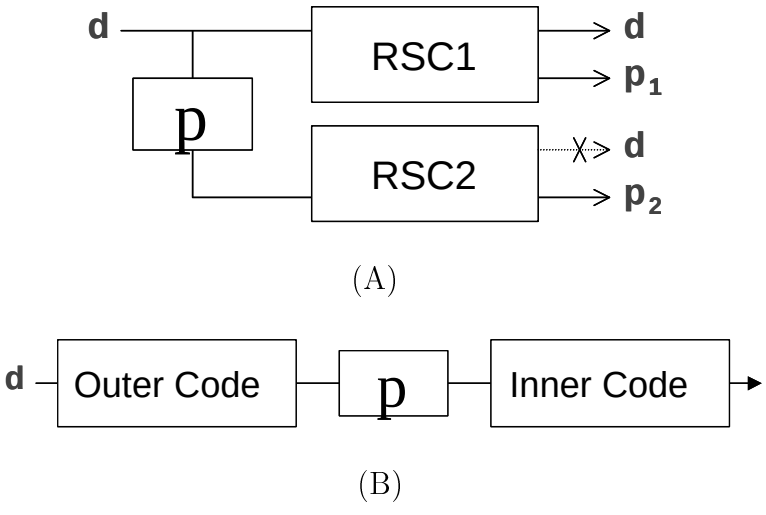


Fig. 1. Structure of parallel and serial concatenated codes. (A) Parallel turbo codes. (B) Serial turbo codes.

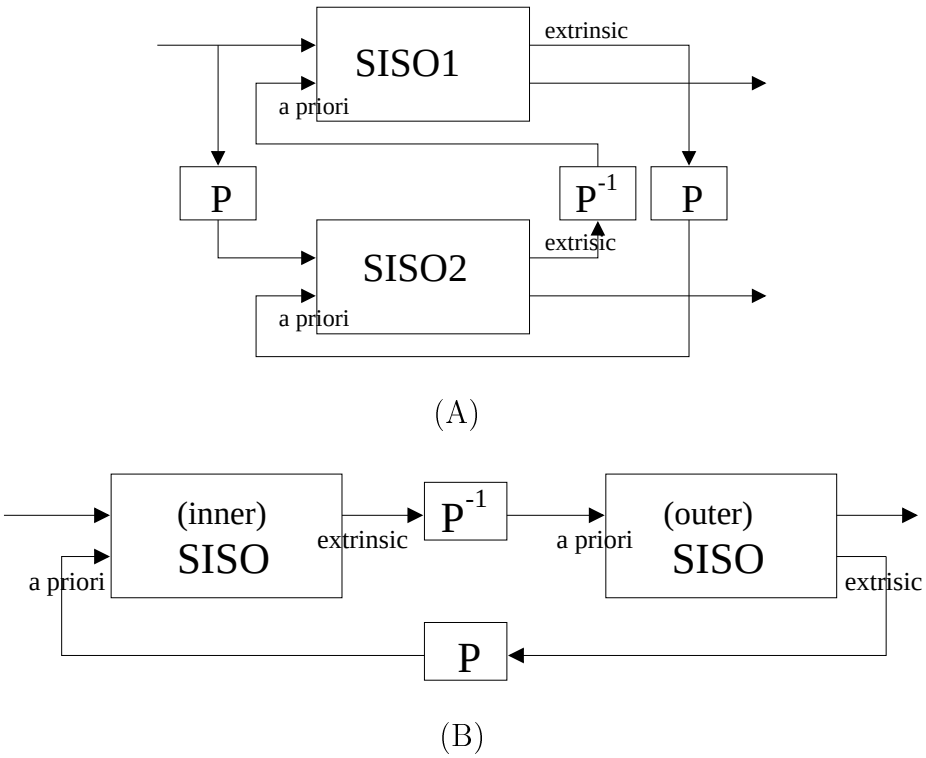


Fig. 2. Turbo decoder structure. (A) Parallel turbo decoder. (B) Serial turbo decoder.

principle is strictly followed to help the iterative process to approximate the optimal solution; that is, during each iteration of the message exchange, out-bound information (known as the extrinsic information) from a local processor (either the inner decoder or the outer decoder) is constrained to have the least correlation with the in-bound information (known as the *a priori* information) to this processor. Although the turbo decoding algorithm works well, the complexity is quite high. The BCJR algorithm is about 4 times as complex as the Viterbi algorithm [26]. Since a turbo decoder takes several iterations to converge where each iteration involves two rounds of BCJR decoding (corresponding to 2 subdecoders), it is usually expensive to implement. To reduce the complexity, SOVA can be used in replace of BCJR at the cost of performance degradation of about 0.3-0.5 dB.

Turbo codes, particularly their concatenated code structure and the iterative decoding principle, serve as the foundation of many of the newly-discovered good codes, including the ones investigated in this research. Further, the structural model of serial turbo codes can be extended for applications in a variety of systems such as coding over inter-symbol interference (ISI) channels [27] [28] [29], which will be investigated in Chapter V.

We mention that in the analysis of interleaved concatenated codes, the concept of *uniform interleaver* is typically used. Since obtaining the performance of a specific interleaver in use is practically intractable, a uniform interleaver serves as a mathematical convenience to investigate the average performance of the code ensemble where all possible interleavers are taken into account with equal weight.

2. Low Density Parity Check Codes

An (N, K) low density parity check code [9] [10] is defined by a sparse parity check matrix H in a non-systematic form (Figure 3). Since each row of the H matrix is

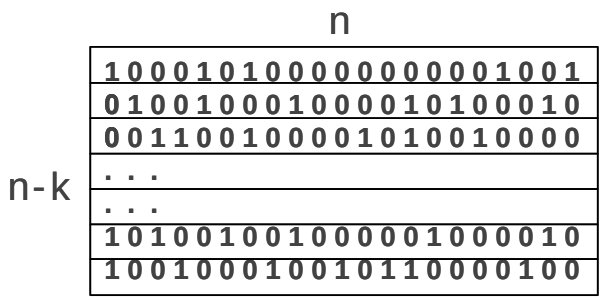


Fig. 3. Parity check matrix H of an LDPC code.

a single parity check, an LDPC code can be loosely viewed as the concatenation of $(N - K)$ single parity check codes in parallel.

Another important way to represent LDPC codes is using a Tanner graph [30] which is a bipartite graph using *bit* nodes and *check* nodes to represent the columns and rows of the H matrix and using inter-connecting *edges* to represent the relations between bits and checks (Figure 4). There are regular and irregular LDPC codes, corresponding to uniform or non-uniform column weights. Major parameters for an LDPC code include the column weight (or the bit node degree) γ, the row weight (or the check node degree) ρ and the *girth* (which is defined as the length of the shortest cycle in the Tanner graph). An LDPC code is said to be (γ, ρ)-*regular* if all columns have weight γ and all rows have weight ρ. For irregular LDPC codes, which are not constrained to uniform row or column weights, degree profiles, $\rho(i)$ and $\gamma(i)$, are usually used to describe the distributions of row weights and column weights, where $\rho(i)$ and $\gamma(i)$ represent the percentage of rows and columns with weight i, respectively. The decoding of an LDPC code uses an iterative message-passing algorithm which is essentially an instance of Pearl's belief propagation algorithm operated on the graph representation of the code. The message-passing algorithm is also known as the sum-product algorithm and has a low-complexity approximation known as the min-sum

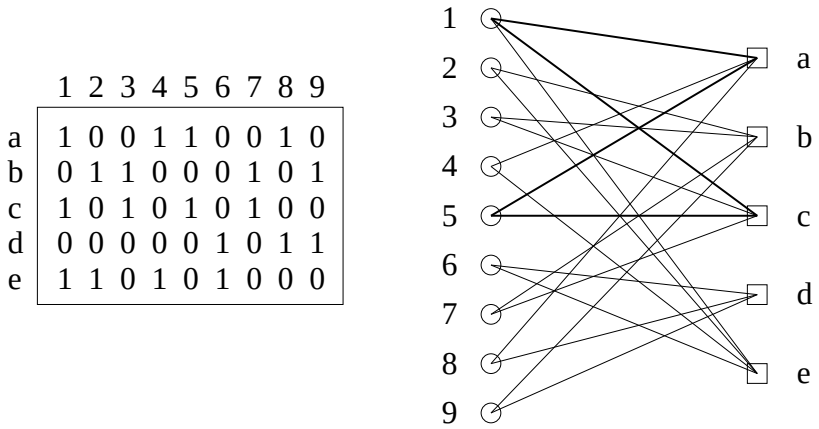


Fig. 4. Parity check matrix presentation and Tanner graph presentation of an irregular (9,4) LDPC code. (LDPC degree profile $\lambda(2) = 8/9$, $\lambda(3) = 1/9$, $\rho(3) = 1/5$, $\rho(4) = 4/5$ and girth of 4.

algorithm [10] [31], where (soft) messages are passed from bits to checks and checks to bits. The girth is important to LDPC codes because the efficiency of the message-passing decoder is adversely affected by the existence of short cycles. In the ideal case where the graph is tree-like (i.e., cycle-free), the message-passing decoder converges to the optimal solution.

It has been shown that regular LDPC codes have minimum distance (averaged over the ensemble of the code) which increases linearly with the block size, provided the column weight is at least 3 [9]. This implies that they have excellent asymptotic performance in the code length. Irregular LDPC codes provide unequal error protection to the code bits and have demonstrated a "wave" effect in the decoding process, such that highly protected bits tend to be decoded first and then help with the less protected bits. With carefully designed row and column degree profiles, irregular LDPC codes can outperform regular LDPC codes [11] [14] [32].

Although in practice short cycles are unavoidable in an LDPC code, the existing (suboptimal) decoder nevertheless performs quite well. Simulations have shown

LDPC codes perform as well as turbo codes (and may even be better for very large block sizes), yet with a lower decoding complexity. It is worth mentioning though that LDPC codes from random constructions (which is typically the case) could have encoding complexity quadratic with the code length ($O(N^2)$), although careful pre-processing can be performed to make the encoding complexity (near-)linear [33].

Possibly the biggest concern in implementing an LDPC in hardware is its irregular structure. Except for a few types that are constructed from finite geometries [34] and combinatorial designs [35] [36] [37], the majority of LDPC codes have very random structure which makes designing/wiring of data flow paths very difficult. Another drawback of LDPC codes is that it is not flexible to change the code rate and/or code length. Although puncturing and extending are possible for LDPC codes (which we will address in detail in Chapter VI), change of rate and/or length would usually require a reconstruction of the parity check matrix H and the corresponding generator matrix G.

3. Turbo Product Codes

A turbo product code [1] [18] [19], also known as a block turbo code (BTC) [38], is composed of a multi-dimensional array of codewords from linear block codes, such as parity check codes, Hamming codes and BCH codes. The structure of a 2-dimensional TPC code is shown in Figure 5. Here, "turbo" refers to the iterative decoding process, and "product" refers to the fact the parameters of the TPC code are the product of the parameters of its component codes. Let $\mathcal{C}_1 \sim (n_1, k_1, d_1)$, $\mathcal{C}_2 \sim (n_2, k_2, d_2), \cdots, \mathcal{C}_s \sim (n_s, k_s, d_s)$ denote s linear binary block codes, where $n_i, k_i, d_i, i = 1, 2, \cdots, s$, are

[1] The code structure was introduced back in 1954, although the term "turbo" and the current soft iterative decoding strategy was not proposed until after the discovery of turbo codes in 1993.

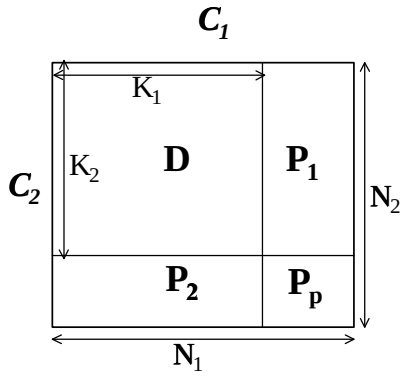


Fig. 5. Structure of 2-dimensional turbo product codes. (D: user data; P_1: parity from component code $\mathcal{C}_1$; P_2: parity from component code $\mathcal{C}_2$; P_p: parity on parity.)

the codeword length, user data sequence length and minimum distance respectively. An s-dimensional turbo product code, $\mathcal{C} = \mathcal{C}_1 \otimes \mathcal{C}_2 \otimes \cdots \otimes \mathcal{C}_s$, thus has parameters $(n_1 n_2 \cdots n_s, k_1 k_2 \cdots k_s, d_1 d_2 \cdots d_s)$, and its generator matrix is the Kronecker product of the generator matrices of its component codes: $G = G_1 \otimes G_2 \otimes \cdots \otimes G_s$. It is known that very simple (almost useless) component codes could result in an overall powerful TPC code.

Particularly of interest to our work is the simplest type of TPC codes whose component codes are single parity check codes, namely, single-parity check turbo product codes (TPC/SPC). We consider a 2-D TPC code as an example whose encoding is straightforward: The data bits are placed in a $K_1 \times K_2$ block and each row is first encoded using $\mathcal{C}_1$, (parity P_1 is added, see Figure 5). Then, each column is encoded using $\mathcal{C}_2$, (parity P_2 and P_p are added). Due to the linearity of the code, the order of encoding is unimportant and all rows will be valid codewords of $\mathcal{C}_1$ (call it row code) and all columns will be valid codewords of $\mathcal{C}_2$ (call it column code). The encoding algorithm can be easily extended to the multidimensional case. In the general case, a TPC code may or may not have "parity-on-parity" bits (P_p in Figure 5). A TPC

code without parity-on-parity is essentially a parallel concatenation with a block interleaver in between, and a TPC code with parity-on-parity is a serial concatenation with a block interleaver in between.

The decoding of TPC codes uses an iterative approach based on soft-in soft-out (SISO) decoders for each of its component codes. Decoding of TPC codes is generally via the Chase algorithm [39], a controlled-search procedure. However, with single-parity check component codes, the decoding can be handled in a simpler and more efficient manner. Observe that a TPC/SPC code can be interpreted from different perspectives. One particular view-point is to model it as a serial concatenation of its component codes with a linear block interleaver in between. From the graph-based point of view, it can also be viewed as a special type of structured regular LDPC code where each row in each dimension satisfies a check (see Figure 6). This observation leads to a convenient adoption of the message-passing algorithm (or the sum-product algorithm) from LDPC codes [27] [40]. Since each bit is expressed as the modulo-2 sum of the rest of the bits in the check, this message-passing decoding algorithm is in fact an extension of replication decoding [20]. The exact decoding algorithm can be found in Appendix A. Since high rates are desirable, we will focus primarily on 2-dimensional TPC/SPC codes in this work, but the above encoding/decoding algorithm (as well as properties), are readily extendible to the multi-dimensional case.

D. Outlines of the Research Work

1. Objective and Scope of the Research

This research work is based on the recent advances in the design, analysis and evaluation of iterative decoding systems as exemplified by turbo codes [8], low density

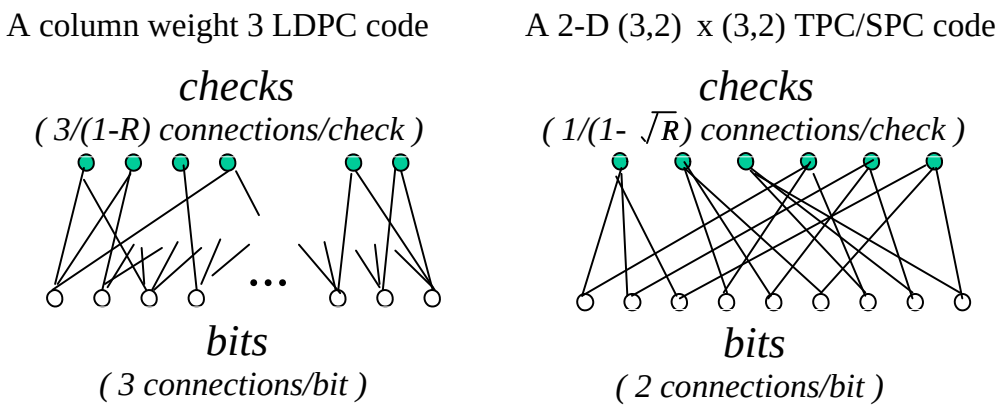


Fig. 6. Bipartite graph representation of LDPC and 2-dimensional TPC/SPC codes.

parity check codes [9] [10], turbo product codes [18] [19], etc. The primary interest of this work is to find a class of high-rate and well-performing codes that are simple to construct, simple to implement in hardware and (hopefully) simple enough to perform theoretical analysis. We are also interested in the analysis and performance evaluation of the aforementioned state-of-the-art codes in various application scenarios.

In the design for simple and good codes, we propose the interleaved serial concatenation of an outer single-parity check turbo product code and an inner rate-1 recursive convolutional code, and name it *product accumulate* (PA) codes [23] [24]. We will present and motivate the fundamental ideas and indicate some technical aspects with proofs and implementation details. The work is composed of various pieces, but the main components will be developed within the following two main themes.

1. The first is to investigate, design and analyze the proposed codes, namely, product accumulate codes and its extension, generalized product accumulate codes (GPA), from the code theoretic perspective, which will include code structural properties, iterative decoding algorithms and their efficiency, interleaving gain,

as well as asymptotic performance bounds/thresholds from the ML analysis and the iterative analysis.

2. The second theme is to investigate, extend and evaluate the state-of-the-art codes in the context of wireless communication, optical fiber communication, data recording, and packet data transmission networks. Specifically, for land mobile wireless communication we investigate and analyze the performance of short and long product accumulate codes on independent/correlated flat Rayleigh fading channels with LDPC codes as a comparison study. For long-haul optical fiber communication with amplitude spontaneous emission (ASE) noise, we investigate and analyze the performance of high-rate, long product accumulate codes on Chi-square channels with turbo codes and BCH codes as a comparison study. For high-capacity data recording applications, we investigate single parity check turbo product codes and LDPC codes on several types of partial response (PR) channels and Lorentzian channels. For packet data networks where retransmission is allowed, we construct and investigate efficient rate-compatible LDPC (RC-LDPC) codes for use in hybrid automatic repeat request (ARQ) systems with code combining and packet combining. The features we will address include joint (iterative) differential detection and decoding, joint (iterative) equalization and decoding, rate compatibility, binary precoding for ISI channels, and thresholds of coded ISI channels.

The first theme is more coding theory centric and the latter is more application-oriented, but some of the coding theories, like the turbo principle and the interleaving gain principle, and some of the analytical tools, like the union bound, Divsalar's simple bound, density evolution (DE) and extrinsic information transfer (EXIT) chart will be used throughout the work. Through the study, we hope to shed insight into the

understanding and practice of coding theory and techniques, to offer a comparing and unifying view of some of the best known codes, and to reveal their potential in a variety of applications.

2. Organization of the Dissertation

The dissertation is organized into six chapters following this introduction.

Chapter II proposes and discusses a novel class of high-rate, high-performance, low-complexity, well-structured and provably "good" codes which are a serial concatenation of a single-parity check turbo product code, an interleaver and a rate-1 recursive convolutional code. (A "good" code is defined as a code for which there exists a threshold above which an arbitrarily low error rate can be achieved as block size goes to infinity [10].) The proposed codes, termed *product accumulate* codes or PA codes, and their extension, *generalized product accumulate* codes or GPA codes, are investigated and analyzed in full. Two slightly different structures, PA-I and PA-II, are presented where the latter can be viewed as a special case of the former with slightly lower complexity[2]. The motivation, the code structure, the iterative decoding algorithm, and encoding and decoding complexity are discussed. In particular, two message-passing decoding algorithms are proposed: sum-product decoding and min-sum decoding. It is shown that a particular update schedule of the sum-product decoding algorithm is equivalent to conventional turbo decoding of the serial concatenated code, but with significantly lower complexity.

In the analysis of the proposed (generalized) product accumulate codes, the focus is on two principal issues: (1) to investigate the properties of the ensemble of these codes and (2) to investigate the performance of the associated iterative decoding

[2]Throughout the work, unless otherwise stated, product accumulate codes or PA codes refer to PA-I codes.

algorithms. For the former, the investigation is concentrated on the ensemble average of distance spectrum and the quantification of the *interleaving gain*, which will give us an idea of how the performance of the code improves as the block size increases. Several performance bounds based on distance spectrum are computed, including the popular union bound and a tighter *simple bound* due to Divsalar [41] which can overcome the cut-off rate limitation. For the latter, we employ density evolution [14] [12] [32] [11], a useful tool in the analysis and design of iteratively decodable codes, to compute the thresholds or the asymptotic performance limit of the subject codes using the existing iterative decoding algorithm. Comparing the bounds calculated in the ML sense (like the union bound, the simple bound, etc) and the thresholds calculated in the iterative sense, performance loss due to the suboptimality in the iterative decoding as well as their implication on the code design can be obtained.

Through analysis, we show that product accumulate codes are linear time encodable and linear time decodable, and are "good" both in the maximum likelihood sense and under iterative decoding. Asymptotic thresholds show that these codes are capable of performance within a few tenths of a dB away from the Shannon limit on additive White Gaussian noise (AWGN) channels. Simulation results confirm these claims and show that these codes provide similar performance to turbo codes but with significantly less decoding complexity and without an obvious error floor. For example, for a rate 0.8, data block size 16K PA code with properly designed S-random interleavers, simulations show that the error floor does not appear until as low as bit error rate (BER) of 10^{-8} to 10^{-9} (Chapter IV).

The chapter then extends PA codes to *generalized product accumulate* codes or GPA codes, where the outer code is relaxed from 2 parallel SPC branches to M (≥ 2) parallel branches. Through similar approaches, we show that all the nice properties of PA codes also hold for GPA codes. The nice thing of GPA codes is that they

are rich in code rate and code construction, and thus allow rate adaptivity. GPA codes exist for virtually any code rate from 0 to 1, and may have more than one construction for a given code rate (and length). Through threshold computation and computer simulations we provide guidelines for choosing the best construction. We expect GPA codes to be useful for systems where high rate codes are usually required (for bandwidth efficiency) but low rate codes are occasionally used due to severe channel noise/distortion (for stronger error protection capability).

Following the proposition and construction of new codes in Chapter II, the next four chapters direct the focus on the practical issue of applying the state-of-the-art error correction codes and coding techniques to four applications, wireless communication, optical fiber communication, magnetic and magneto-optical recording system and packet data networks.

Chapter III investigates the performance of PA codes on wireless communication channels which, without line of sight, are modeled as flat Rayleigh fading channels. We analyze and evaluate the performance of PA codes with both coherent and noncoherent detection, and extend the discussion to a more general case where the outer code can be any LDPC code (as will be discussed in Chapter II, the outer code of PA codes can be viewed as a special type of LDPC code). The motivation is two-fold: first, previous work on PA codes (Chapter II) has established them as a class of low-complexity, capacity-approaching good codes on AWGN channels. Second, PA codes are inherently differentially coded which permits noncoherent detection on wireless fading channels.

The performances of PA codes using coherent binary phase shift keying (BPSK) signaling on independent Rayleigh fading channels with and without channel state information (CSI) as well as on correlated land mobile Rayleigh channels with CSI is first studied. Divsalar's simple bounds and iterative thresholds using density evolution

are computed to quantify the performance of PA codes with finite lengths and infinite lengths, respectively. Both analysis and simulations show that PA codes perform as well as LDPC codes on fading channels (with coherent detection), and that their performance is close to the channel capacity on Rayleigh fading channels too.

To exploit the intrinsic differential encoder in PA codes, noncoherent differential detection/decoding is then investigated for PA codes on fast-fading channels. A simple, noncoherent iterative differential detection and decoding (IDDD) receiver is presented and discussed. Simulations show that the IDDD received is robust for different Doppler rates, and can perform within 1 dB from the coherent case with very little extra complexity and bandwidth expansion. The impact of pilot spacing on the overall performance is also investigated.

Extrinsic information transfer charts are used to facilitate the analysis. We first show that the popular practice of inserting pilot symbols to terminate the trellis of the differential code will cause an inherent loss in capacity and that a better way is to separate them from the trellis. Next, we show that although PA codes perform remarkably, a general differentially coded LDPC code does not, since (conventional) LDPC codes do not match in convergence behavior with a differential code. However, without a differential code, more pilot symbols are usually required to track the channel. Hence, on fast fading channels where noncoherent detection is needed and where bandwidth expansion is limited, conventional LDPC codes are unable to perform as efficiently as PA codes.

Following the analysis, a convergence-constraint density evolution is proposed for designing good LDPC ensembles matched with differential coding. We observe that the LDPC ensemble optimal for differential coding always contains degree-1 and 2 variable nodes, and that for high code rates, these nodes are dominant. The optimized LDPC code (with differential coding) shows a 1.04 dB gain over PA codes. We expect

the method to be useful for designing good LDPC ensembles for a variety of inner code, modulation and receiver.

Chapter IV investigates forward error correction (FEC) techniques for use in long-haul optical fiber communication. FEC techniques have become increasingly important in improving the throughput/capacity of optical fiber communications. The traditional use of BCH codes, although simple and efficient, seems to be inadequate for future needs of high data rate, high throughput optical fiber communication. In the search for good FEC codes for future optical fiber communication, we investigate product accumulate codes for the apparent reasons that they are high rate, low complexity and well performing. For comparison purposes, turbo codes are also studied.

We consider an optical fiber communication channel where the amplified spontaneous emission noise dominates all other noise sources. Three channel models are investigated, including the asymmetric Chi-square channel (which is by far the most accurate channel model for long-haul optical fiber communication with ASE), its asymmetric Gaussian approximation, and the symmetric Gaussian approximated channel models (where the channel is actually reduced to the conventional AWGN channels). Binary on-off keying (OOK) modulation and an iterative soft-decision message-passing decoding are used for PA codes. At low signal-to-noise ratios (SNR), code performance is evaluated with simulations of typical PA coding schemes. For high SNRs beyond the simulation capabilities, we derive the pairwise error probability based on the three channel models and explore an average upper bound on the performance of the ensemble of PA codes. Extensive simulations show that PA codes can yield impressive performance on fiber optic channels also (with 9 dB gain over uncoded systems at bit error rates of 10^{-8}) and that the error floor does not appear as low as BER of 10^{-8} to 10^{-9}. Another interesting finding of the work is that AWGN

channels, although fundamentally different from Chi-square channels, can serve as a convenient reference to approximate the code performance at high rates on Chi-square channels.

Chapter V focuses on coding techniques for digital recording systems. Future high-density magnetic recording systems require very high rate codes ($R > 0.88$) and very low bit error rate (around 10^{-14}). Such low error rate is only achieved by the concatenation of two levels of coding schemes, where a Reed Solomon code is usually used in the second level to clear up the residual errors. The first-level error correction codes are the key to assure the overall error control quality. Not only is low bit error rate required, but burstiness in errors should be avoided. Further, due to the concern for delay and speed, the complexity of the codes should be kept as low as possible. Concerning such requirements, we investigate the potential of several high-rate coding schemes, including serial turbo (punctured convolutional codes), LDPC codes and TPC/SPC codes [27] [40]. In the first phase, we assume a perfectly equalized partial response channel model with additive white Gaussian noise. We model the channel as a rate-1, binary-input and real-valued-output convolutional code and exploit iterative decoding and equalization (IDE). The effect of binary precoding is studied in detail. Several channel models in the partial response class IV family, including PR4, EPR4, E^2PR4, and ME^2PR4 are evaluated. A comprehensive evaluation is performed which includes complexity, bit/frame error rate and bit/symbol error statistics. Further, to facilitate the understanding, density evolution is used to calculate the thresholds of each coding system, which serves as the practical capacity limit [42]. Some interesting issues related to the optimization of the decoding strategies will also be discussed [40].

The next step is to use more realistic channel models, namely, the Lorentzian channels with colored noise. A front-end filter is first used to equalize the channel to some suitable targets so that the same decoding strategy as in the perfect PR channel

model can be applied. However, in this case, the final performance is not only related to the performance of the codes, but also to the effectiveness of the equalization targets as a function of the normalized recording density and the code rates. Hence, in addition to the investigation of error rate and error statistics, the effect of different equalization targets and the impact of the code rate are also evaluated.

In addition to the forward error correction techniques addressed in the previous chapters, another useful way of error control is automatic repeat request which is widely used for systems where a feedback channel is available and where transmission delay is acceptable. In particular, ARQ systems combined with a (good) FEC coding scheme, known as ARQ/FEC or hybrid ARQ systems, can achieve capacity-approaching throughput with guaranteed reliable transmission and, hence, are very popular in practical packet data networks as well as wireless communications. Hybrid ARQ systems make use of rate-compatible (RC) error correction codes to achieve high throughput efficiency through incremental retransmission. The key is to employ a wise ARQ strategy and, more importantly, to use an efficient rate compatible code.

The focus of Chapter VI is on the construction of good rate-compatible low density parity check codes. The conventional approach of puncturing is first studied. Investigation of the code ensemble and the asymptotic performance using density evolution reveals that puncturing produces efficient RC-LDPC codes only at high rates and only when the amount of puncturing is small. To extend the dynamic rate range, a special approach of extending is proposed and investigated whose performance is shown to be very encouraging at low rates. Combining both approaches, efficient RC-LDPC codes are constructed to offer strong error correction capability over a wide code rate range. Using the constructed RC-LDPC, an LDPC-coded hybrid ARQ system using both code-combining and packet-combining is evaluated. The encouraging result is that the proposed LDPC-ARQ system can achieve throughput about 1 dB

away from the theoretical limit, which is comparable to turbo ARQ systems [43, 44], yet with lower decoding complexity.

Finally, Chapter VII summarizes the whole work and presents concluding remarks.

For readers' convenience, an overview of the content and the organization of the dissertation is illustrated in Figure 7. Further, all the abbreviations used in the dissertation are listed in Appendix C.

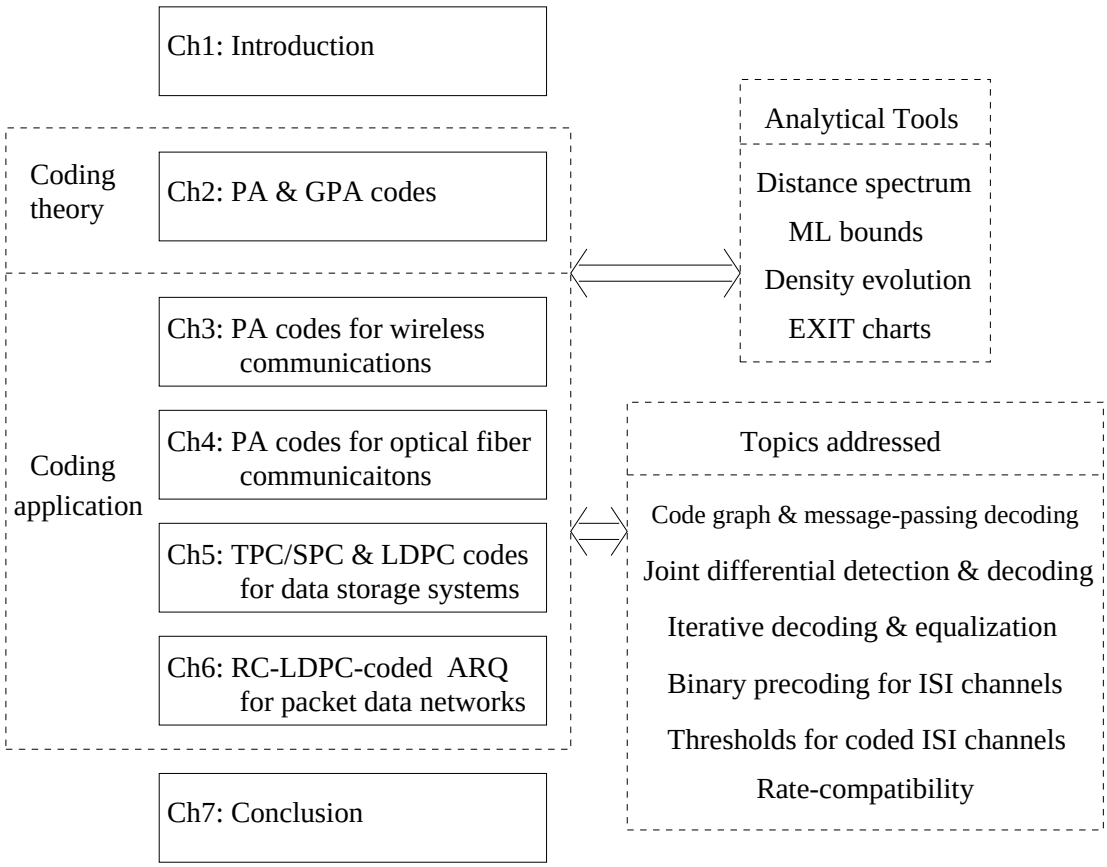


Fig. 7. Content and organization of the dissertation.

CHAPTER II

PRODUCT ACCUMULATE CODES AND GENERALIZED PRODUCT ACCUMULATE CODES

A. Motivation and Outline of the Chapter

In this chapter, we propose and discuss a novel class of provably "good" codes which are termed *product accumulate* (PA) codes [23] [24]. (A "good" code is defined as a code for which there exists an SNR threshold such that, when the channel is better than this threshold, arbitrarily low error rates can be achieved as block size goes to infinity [10].) The proposed codes are shown to possess many inviting properties including close-to-capacity performance, low decoding complexity, regular structure and easy rate adaptivity uniformly for all rates from 1/2 and higher.

The work was initiated by the search for good, high-rate codes which permit soft decision and soft-output decoding. Several applications require the use of (soft-decodable) high rate codes. Some widely used high-rate codes are Reed-Solomon codes, (punctured) convolutional codes, (punctured) turbo codes and low density parity check codes. Until very recently, soft decision decoding of RS codes has been a major computational problem, and the recent developments are yet to be benchmarked to know the exact performance of soft decision decoding of RS codes. In order to get good performance from high-rate punctured convolutional codes and turbo codes, the (component) convolutional codes usually have to be of long constraint length, making the decoding complexity rather high. Low density parity check codes, on the other hand, provide good performance at possibly lower complexity; however, except for a few subsets of structured designs (like finite geometry and combinatorial design), the encoding usually requires explicit storage of a generator matrix and,

moreover, the random and irregular structure of LDPC codes makes it difficult to design and wire the layout in hardware. Further, good high-rate LDPC codes are difficult to construct for short block lengths.

In an effort to construct good, simple, soft-decodable, high-rate codes, we investigated single-parity check turbo product codes which can be soft decoded by a message-passing algorithm (also referred to as the sum-product algorithm). TPC/SPC codes have recently been investigated for potential application on high-density magnetic recording channels and have demonstrated encouraging performance via the turbo approach [27]. Since the TPC/SPC itself is not a "good" code, we consider the concatenation of a rate-1 inner code (differential encoder or accumulator) with the TPC/SPC code through an interleaver. Through analysis and simulations, we find this class of codes to be remarkably good in bit error rate at high code rates ($R \geq 0.7$) when used with an iterative message-passing decoding algorithm. We show that the performance of these codes can be further improved by replacing the block interleaver in the conventional TPC/SPC outer code with a random interleaver. We refer to such codes as product accumulate (PA-I) codes. Clearly, when the outer code is a conventional TPC/SPC code, it is a special case of the proposed codes and we refer to those as PA-II codes.

To facilitate understanding the structure and potential of the proposed codes, we compute tight upper bounds on the performance of these codes using the bounding technique developed by Divsalar in [41]. We also study the graph structure of these codes. Thresholds are computed using density evolution [12] and shown to be within a few tenths of a dB from the Shannon limit for all rates $R \geq 1/2$. By studying the graph structure, a message-passing (sum-product) decoding algorithm and its low-complexity approximation, a min-sum algorithm, can be developed to iteratively decode the outer code and the inner code. We show that a particular update schedule

for this algorithm when applied to the graph of the inner code results in optimal decoding of the inner code $1/(1 \oplus D)$. That is, the sum-product algorithm applied to the decoding of $1/(1 \oplus D)$ is equivalent to the BCJR (or *a posteriori* probability) algorithm [6] and the min-sum algorithm is equivalent to the Max-log-MAP algorithm [26]. However, the message-passing algorithm can be implemented with significantly lower complexity than the BCJR equivalents. Simulation results with long block lengths confirm the thresholds, and simulations with short block lengths show that performance close to turbo codes can be achieved with significantly lower complexity.

We propose the class of product accumulate codes as a prospective class which not only enjoys good performance, low complexity and soft decodability, but also maintains a simple and regular structure uniformly for all block sizes and for all rates above 1/2. The regular structure, as well as the ease in construction, are particularly appealing properties in practical implementation and in applications that require rate adaptivity.

The rest of the chapter is organized as follows. In Section B, the drawback of TPC/SPC codes is briefly discussed followed by the description of the system model for PA codes, where two (slightly) different types, namely, PA-I and PA-II codes, are compared. Section C discusses the decoding algorithm for PA codes and in particular a graph-based sum-product algorithm is described and shown to be optimal for the inner rate-1 convolutional codes, yet with low complexity. Section D analyzes in detail some properties of PA codes, including upper bounds on the performance under ML (maximum likelihood) decoding and thresholds of the codes under iterative decoding. Section E discusses an algebraic construction which is useful in practical implementation. Section F extends product accumulate codes (PA-I) to generalized product accumulate codes, where similar properties as PA codes are discussed for GPA codes. Section G presents simulation results for PA and GPA codes on AWGN

channels to bench-mark their performance. Section H compares PA codes with other good codes proposed recently, in an attempt to provide a unified viewpoint of the recent advancements in codes and code graphs. Finally Section I summarizes the chapter.

B. Structure of the Proposed Product Accumulate Codes

1. A TPC/SPC Code by Itself Is Not a "Good" Code

As introduced in Chapter I, single parity check turbo product codes, or TPC/SPC codes, have several inviting properties including intrinsically high rate, simplicity and soft decodability, and, hence, seem a viable candidate for low-cost, high-rate applications. The simple and regular structure of a TPC/SPC code makes it possible to analyze the code properties. In particular, the weight spectrum of a 2-dimensional TPC/SPC code with parameter $\mathcal{C} \sim (n_1 n_2, (n_1-1)(n_2-1))$ (Figure 5) can be calculated by the following equation [20]

$$A(h) = 2^{-n_1} \sum_{a=0}^{n_1} \binom{n_1}{a} \left[\sum_{m=0, m\ even}^{n_1} P_m(a; n_1) h^m \right]^{n_2}, \tag{2.1}$$

where

$$P_m(a; n) = \sum_{k=0}^{m} (-1)^k \binom{a}{k} \binom{n-a}{m-k}. \tag{2.2}$$

As expected, $A(h)$ is symmetric in n_1 and n_2. It has been shown that the weight distribution of TPC/SPC codes asymptotically approaches that of a random code if the dimension of the code and the lengths of all component codes go to infinity [20]. However, increasing the dimension decreases the code rate and, hence, makes the codes less bandwidth efficient.

Although TPC/SPC codes have been considered for some high-rate applications,

such as data storage systems, a TPC/SPC code by itself is not a "good" code. To see why TPC/SPC codes are not "good", note that an s-dimensional TPC/SPC code always has minimum distance of 2^s irrespective of the block size. Assuming maximum likelihood decoding, the lower bound on the word error rate (WER) (also known as frame error rate or FER) is

$$P_w(e) \geq Q\left(\sqrt{\frac{2^{s+1}RE_b}{N_o}}\right), \tag{2.3}$$

where R is the code rate. Obviously, the lower bound is not a function of block size. In other words, unless the dimensionality of a TPC/SPC code, s, goes to infinity, its WER performance is always bounded away from zero independent of the block size. Hence, a TPC/SPC code alone is not good in the ML sense.

In an effort to improve the performance of TPC/SPC codes, some attempts have been made to increase its minimum distance by carefully adding more parity checks by increasing the dimensionality [45] [19]. However, adding dimensionality obviously reduces code rate. Further, for any TPC/SPC code of a given dimensionality, the minimum distance is fixed and does not improve with block size. In other words, except for the asymptotic case where $s \to \infty$, multi-dimensional TPC/SPC codes are still not "good" codes. Moreover, when $s \to \infty$, $R \to 0$, and, hence, these codes are not of much practical interest.

In this work we take a different approach which is to group several blocks of TPC/SPC codewords together, interleave them and further encode them with a rate-1 recursive convolutional code. The resulting serial concatenation brings a significant improvement to TPC/SPC codes in their fundamental structural properties, for, as will be explained in later sections, the resulting serial concatenated code now becomes a "good" code with linear encoding and decoding complexity. Furthermore, we will

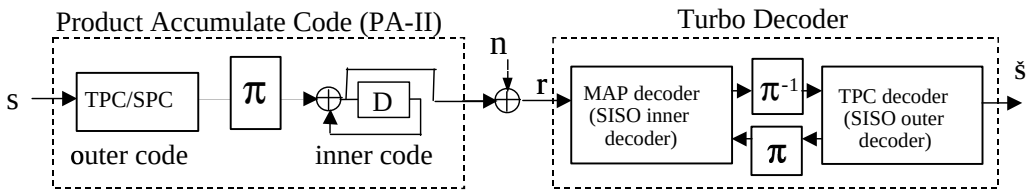


Fig. 8. System model for product accumulate (PA-II) codes and iterative decoder. (The outer TPC/SPC code is presented in such a way to illustrate that although there is only one TPC/SPC code, several blocks of TPC/SPC codewords are combined before interleaving.)

discuss a modification to the interleaving scheme within the TPC/SPC code which results in a better code structure.

2. Proposed Code Structure

Figure 8 shows the overall structure of the proposed high-rate code, which we call a *product accumulate* code. It comprises a 2-D TPC/SPC outer code of rate $(t/(t+1))^2$ ($t/(t+1)$ in each dimension), a random interleaver, a rate-1 recursive convolutional inner code of the form $1/(1 \oplus D)$ (also known as the accumulator). For a TPC/SPC code of length $(t+1)^2$, P such codewords are taken and interleaved using a random interleaver of size $N = P(t+1)^2$, and further encoded using a rate-1 recursive convolutional encoder to form a codeword of length N. Thus, the resulting code is a $(P(t+1)^2, Pt^2)$ code with rate $(t/(t+1))^2$. The random interleaver works to break up the correlation between the messages (extrinsic information) and, in conjunction with the recursive inner code, to map low-weight error events to high-weight error events, which results in a good distance spectrum.

Since P codewords of the TPC/SPC code are interleaved together before passing through the accumulator, the input block length (and, hence, latency) is Pt^2 information bits. A conventional TPC/SPC code itself is a 2-D code where the data is

interleaved using a block interleaver. Therefore, the structure of the outer code is equivalent to Pt^2 information bits being interleaved using P separate block interleavers. Instead of this structure, it is possible to replace the interleaver within the TPC/SPC with one random interleaver of size Pt^2 bits. When a random interleaver is used, it is no longer trivial to add parity-on-parity bits and, hence, we do not use parity-on-parity bits. The structure of the resulting code is shown in Figure 9. To be precise, the resulting code (i.e., our proposed product accumulate code) is a serial concatenation of an outer code, an interleaver and a rate-1 inner code. The outer code itself is a parallel concatenation of two $(t+1, t)$ single parity check codes with an interleaver between the two parallel branches. (For notational convenience, we still refer to the outer code as a TPC/SPC code, and those that use block interleavers as *conventional* TPC/SPC codes.) For an input of $K = Pt^2$ bits, the outer codewords (without parity-on-parity) are of length $P(t^2+2t)$ and, therefore, the rate of the code is $t/(t+2)$. We refer to this structure as PA-I codes. The code using the conventional TPC/SPC code as the outer code (introduced in the previous paragraph) can be thought of as a special case of this PA-I structure where the interleaver in the parallel concatenation is constrained to be P separate block interleavers and which has parity on the parity bits (which leads the resulting rate to be $(t/(t+1))^2$ rather than $t/(t+2)$). We refer to this special case as PA-II codes.

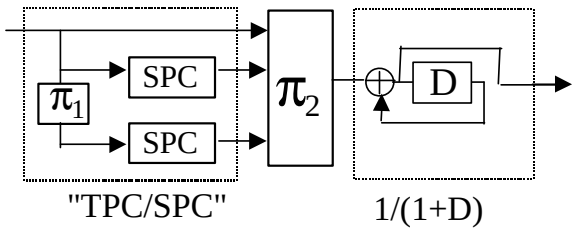


Fig. 9. System model of product accumulate (PA-I) codes.

The idea of concatenating an outer code and an interleaver with a rate-1 recursive inner code, particularly of the form of $1/(1 \oplus D)$, to achieve coding gains (interleaving gain) without reducing the overall code rate is widely recognized [17] [29] [46]. For low rate codes (rate-1/2 or less), convolutional codes, and even very simple repetition codes [22], are good outer code candidates to provide satisfactory performance. However, the construction of very high rate codes based on this concept poses a problem. The key problem here is that, from Benedetto *et al.*'s result [25] [21], the outer code must have a minimum distance of at least 3 in order to obtain an interleaving gain. To obtain good high-rate convolutional codes through puncturing, and in particular to maintain a d_{min} of 3 after puncturing, the original convolutional codes must have fairly long constraint length, which makes it computationally inefficient. On the other hand, 2-dimensional TPC/SPC possess several inviting properties for a concatenated high-rate coding structure, such as high rate, simplicity and the availability of an efficient soft decoding algorithm. Recall that for an interleaved serial concatenated code (with a recursive inner code) to achieve interleaving gain in word error rate, the outer code needs to have minimum distance $d_{min} \geq 3$ [21] [25]. For PA-II codes, the outer code has $d_{min} = 4$ for any rate and, hence, an interleaving gain results. In Section 2, we will show that although the outer code of PA-I codes has $d_{min} = 2$ in the worst case, such an interleaving gain still exists for the ensemble of PA-I codes.

In the sections below, we will perform a comprehensive analysis and evaluation of the proposed product accumulate codes. The focus is on PA-I codes, since they are the more general case and since they generally achieve better performance than PA-II codes (noticeably at medium rates around $R = 0.5$). However, with respect to some interesting aspects, the special case of PA-II codes is worth a separate discussion. The reason we would also like to specifically address PA-II codes is that PA-II codes are the initial model of our construction, who are simpler to analyze and implement than PA-

II codes (block interleaver is usually cheaper and easier to implement in hardware). Further, as we will show later, at rate $R > 0.7$, PA-II codes are sufficient to achieve near-capacity performance, alleviating the need for another random interleaving in the outer TPC/SPC.

C. Iterative Decoding of PA Codes

The turbo principle is used to iteratively decode a serially concatenated system, where soft extrinsic information in log-likelihood ratio (LLR) form is exchanged between the inner and the outer code. The extrinsic information from one subdecoder is used as *a priori* information by the other subdecoder. The decoding of the outer TPC/SPC code is using a message-passing algorithm similar to that of LDPC codes as described previously. The inner rate-1 convolutional code is typically decoded using a 2-state BCJR algorithm, which generates the extrinsic information for bit x_i in the k_{th} turbo iteration, denoted $L^{(k)}_{i,ext}(x_i)$. The outer decoder uses $L^{(k)}_{i,ext}(x_i)$ as *a priori* information and produces extrinsic information $L^{(k)}_{o,ext}(x_i)$. However, a more computationally efficient approach is to use message-passing decoding directly on the graph of the product accumulate code including the inner code, whose sub-graph has no cycles.

It has been recognized that the message-passing algorithm is an instance of Pearl's belief propagation which converges to the optimal solution if the operating graph is cycle-free. The basic idea of probability inference decoding is implied in Tanner's pioneering work in 1981 [30], and later studied by Wiberg [31], Frey [47], Forney [48] *et al.*, as it gained enormous success in the decoding of LDPC codes. However, little has been reported for convenient application on convolutional codes. This is because the code graph of a convolutional code is in general complex and involves

many cycles which either make the message flow hard to track or make the algorithm ineffective (due to the significant amount of correlation in the messages caused by the cycles). Nevertheless, for the specific case of $1/(1 \oplus D)$ code, a cycle-free Tanner graph presenting the relation of $y_i = x_i \oplus y_{i-1}$ ($\oplus$ denotes modulo-2 addition) can be constructed, using which the message flow can be conveniently traced. Below, we describe in detail how the message-passing algorithm can be efficiently applied to $1/(1 \oplus D)$ code. Recently, message-passing on the the graph structure of $1/(1 \oplus D)$ inner code has been used with irregular repeat accumulate (IRA) codes by Jin, Khandekar and McEliece [22] and by Divsalar *et al.* [41] to analyze 2-state codes. Here, we derive a message-passing algorithm from the BCJR algorithm and show the relationship between the two. Specifically, we show that a serial update in the graph (rather than the parallel update as used in [22] and [41]) is equivalent to the the BCJR algorithm, but has an order of magnitude lower complexity. Similarly, we show that the low-complexity approximation, the min-sum update on the graph, is equivalent to the Max-log-MAP algorithm.

1. Message-Passing Algorithm

As shown in Figure 10(A), the combination of the outer code, the interleaver and the inner code can be represented using one graph which contains bit nodes (representing the actual bits) and check-nodes (representing a constraint such that connecting bit nodes should sum up (modulo-2) to zero). Figure 10(B) illustrates how messages evolve within the $1/(1 \oplus D)$ code. The outgoing message along an edge should contain information from all other sources except the incoming message from this edge. For example, the LLR of bit y_i consists of $L_{ch}(y_i)$ from the channel, $L_{e_f}(y_i)$ from check i and $L_{e_b}(y_i)$ from check i+1 (Figure 10(B)). When y_i participates in check i, to calculate the extrinsic $L_e(x_i)$ for x_i, the information content $L_{e_f}(y_i)$, which was

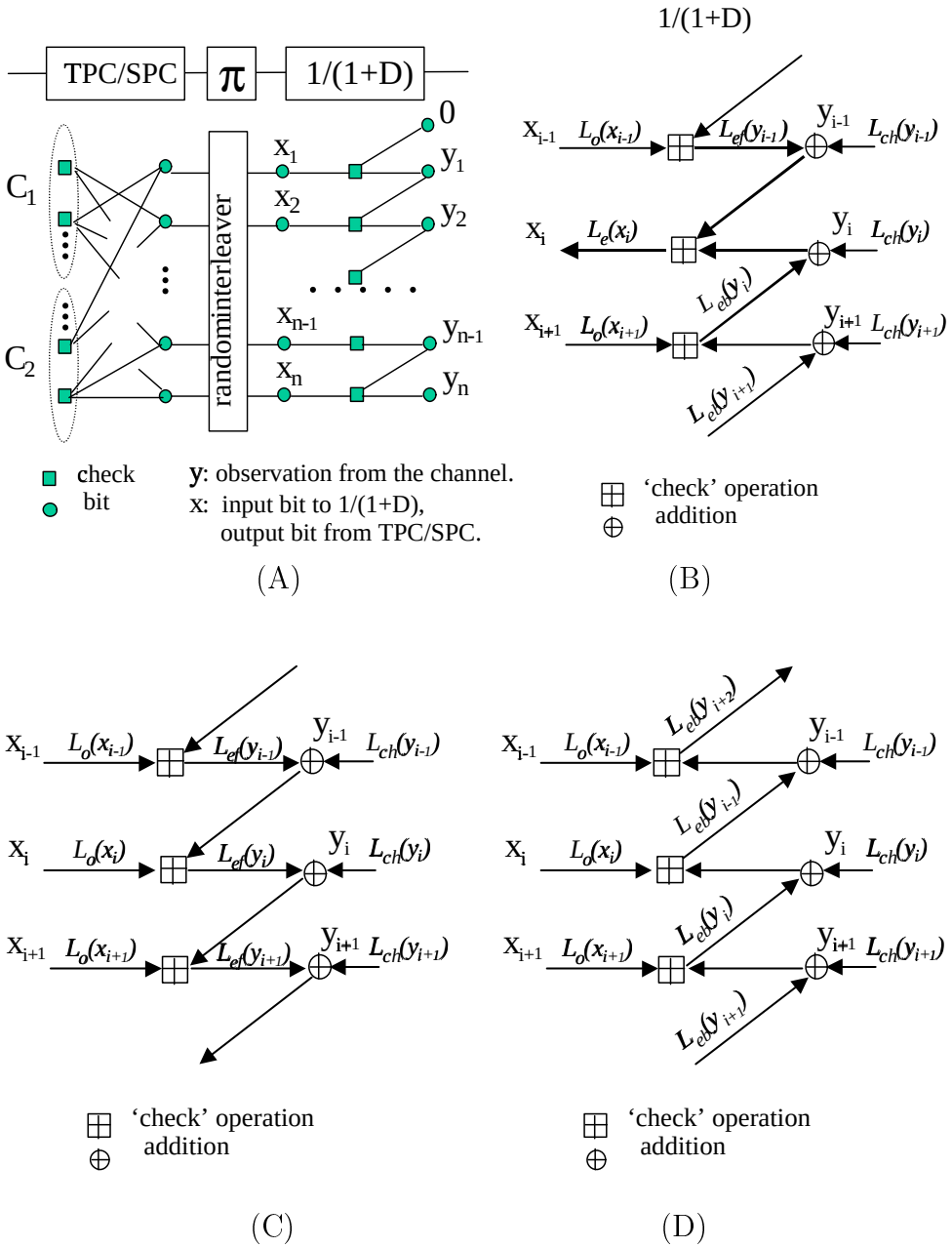


Fig. 10. Code graph and message-passing decoding algorithm. (A) Graph presentation of PA codes. (B) Message-passing decoding for inner code $1/(1 \oplus D)$. (C) Forward pass of message flow. (D) Backward pass of message flow.

obtained from check i previously, should not be passed back to check i. Similar rules hold for all other bits. Hence the extrinsic messages sent out at bit x_i at the k_{th} turbo iteration, $L_e^{(k)}(x_i)$, should be computed as

$$L_e^{(k)}(x_i) = \Big(L_{ch}(y_{i-1}) + L_{e_f}^{(k)}(y_{i-1})\Big) \boxplus \Big(L_{ch}(y_i) + L_{e_b}^{(k)}(y_i)\Big), \tag{2.4}$$

where $L_{ch}(y_i) = \log \frac{\Pr(r_i|y_i=0)}{\Pr(r_i|y_i=1)}$ denotes the message (LLR) obtained from the channel (r_i is the received signal corresponding to the coded bit y_i), $L_{e_f}(y_i)$ and $L_{e_b}(y_i)$ denote the (extrinsic) messages passed "forward" and "backward" to bit y_i from the sequence of bits/checks before and after the i_{th} position, respectively. Superscript (k) denotes the k_{th} turbo iteration between the inner and the outer decoders (as opposed to the local iterations in the decoding of the outer TPC/SPC code) and subscript i denotes the i_{th} bit/check.

Operation $\boxplus$ refers to a "check" operation or the *tanh* operation. It can be shown that if α and β are the LLRs passed along an incoming edge into a $\boxplus$ operation, then the out-going extrinsic information is given by

$$\gamma = \alpha \boxplus \beta \iff \gamma = \log \frac{1 + e^{\alpha+\beta}}{e^{\alpha} + e^{\beta}}, \tag{2.5}$$

$$\iff \gamma = 2 \tanh^{-1} \left(\tanh \frac{\alpha}{2} \cdot \tanh \frac{\beta}{2} \right). \tag{2.6}$$

Messages $L_{e_f}(y_i)$ and $L_{e_b}(y_i)$ correspond to a forward and backward pass, respectively, along the code graph. As illustrated in Figure 10(C)(D), at the k_{th} turbo iteration (between the inner and outer decoder), they can be calculated as

$$L_{e_f}^{(k)}(y_i) = L_o^{(k-1)}(x_i) \boxplus \Big(L_{ch}(y_{i-1}) + L_{e_f}^{(k)}(y_{i-1})\Big), \quad \forall 2 \le i \le N, \tag{2.7}$$

$$L_{e_b}^{(k)}(y_i) = L_o^{(k-1)}(x_{i+1}) \boxplus \Big(L_{ch}(y_{i+1}) + L_{e_b}^{(k)}(y_{i+1})\Big), \quad \forall 1 \le i \le N-1, \tag{2.8}$$

where $L_o^{(k-1)}(x_i)$ is the message received from the outer TPC/SPC code in the $(k–1)_{th}$

turbo iteration (between inner and outer codes). Clearly, $L_o^{(0)}(x_i) = 0, \forall i$, since in the first turbo iteration, the inner code gets no information from the outer code. The boundary conditions are

$$L_{e_f}^{(k)}(y_1) = L_o^{(k-1)}(x_1) \boxplus \infty = L_o^{(k-1)}(x_1), \quad \forall k \geq 1, \tag{2.9}$$

$$L_{e_b}^{(k)}(y_N) = 0, \quad \forall k \geq 1. \tag{2.10}$$

From the above computation, it can be seen that the outbound message at the present time instance i, $L_e^{(k)}(x_i)$, has utilized all dependence among the past and the future (through $L_{e_f}^{(k)}(x_{i-1})$ and $L_{e_b}^{(k)}(x_i)$) without any looping back of the same information.

Lemma 1: *The aforementioned message-passing (sum-product) decoder is identical to the BCJR algorithm for a* $1/(1 \oplus D)$ *inner code.*

Proof: For completeness, we first briefly describe the BCJR algorithm [6]. Due to space limitation, we skip basic introduction to the BCJR algorithm. Interested readers are referred to [6] [26]. We use x_t, y_t, s_t, r_t to represent data bit, coded bit, (binary) modulated bit (signals to be transmitted over the channel) and received bit (noise corrupted), respectively. Their relations are illustrated as follows:

$$x_t \in (0,1) \xrightarrow{y_t = y_{t-1} \oplus x_t} y_t \in (0,1) \xrightarrow{BPSK} s_t \in (\pm 1) \xrightarrow{+\text{noise}} r_t$$

The following definitions and notations are needed in the discussion:

- $\Pr(S_t = m)$ — the probability decoder is in state m at time instance t, ($m \in \{0, 1\}$ in a 2-state case).
- $\mathbf{r}_i^j \triangleq (r_i, r_{i+1}, \cdots, r_j)$ — received sequence.
- $\alpha_t(m) \triangleq \Pr(S_t = m, \mathbf{r}_1^t)$ — forward path metric.

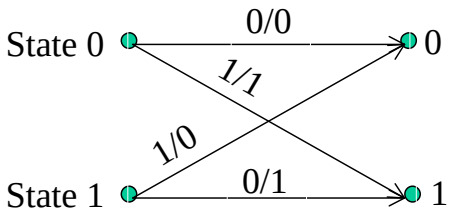


Fig. 11. Trellis of $1/(1 \oplus D)$ code.

- $\beta_t(m) \triangleq \Pr(\mathbf{r}_{t+1}^N | S_t = m)$ — backward path metric.
- $\gamma_t(m',m) \triangleq \Pr(S_t = m, r_t | S_{t-1} = m')$ — branch metric.
- $\Lambda_t \triangleq \log \frac{\Pr(x_t=0|\mathbf{r}_1^N)}{\Pr(x_t=1|\mathbf{r}_1^N)}$ — output LLR of bit x_t.

The branch metric of $1/(1 \oplus D)$ code is given by (see trellis in Figure 11)

$$\gamma_t(0,0) = \Pr(x_t = 0) \Pr(r_t | y_t = 0), \tag{2.11}$$

$$\gamma_t(0,1) = \Pr(x_t = 1) \Pr(r_t | y_t = 1), \tag{2.12}$$

$$\gamma_t(1,0) = \Pr(x_t = 1) \Pr(r_t | y_t = 0), \tag{2.13}$$

$$\gamma_t(1,1) = \Pr(x_t = 0) \Pr(r_t | y_t = 1). \tag{2.14}$$

Note that in the symbol-by-symbol MAP decoding, we have the following forward and backward recursive equations

$$\alpha_t(m) = \frac{\sum_{m'} \alpha_{t-1}(m') \gamma_t(m', m)}{\Pr(y_t | \mathbf{Y}_1^{t-1})}, \tag{2.15}$$

$$\beta_t(m) = \frac{\sum_{m'} \beta_{t+1}(m') \gamma_{t+1}(m, m')}{\Pr(y_{t+1} | \mathbf{Y}_1^{t})}. \tag{2.16}$$

Consider the ratio $\alpha_t(0)/\alpha_t(1)$ in the forward recursion

$$\frac{\alpha_t(0)}{\alpha_t(1)} = \frac{\alpha_{t-1}(0)\gamma_t(0,0) + \alpha_{t-1}(1)\gamma_t(1,0)}{\alpha_{t-1}(0)\gamma_t(0,1) + \alpha_{t-1}(1)\gamma_t(1,1)}. \tag{2.17}$$

Substituting (2.11)-(2.14) to (2.17), and dividing both the numerator and the denominator by $\alpha_{t-1}(1)\Pr(x_t=1)\Pr(r_t|y_t=1)$, we get

$$\frac{\alpha_t(0)}{\alpha_t(1)} = \frac{\left(\frac{\alpha_{t-1}(0)}{\alpha_{t-1}(1)}\frac{\Pr(x_t=0)}{\Pr(x_t=1)}+1\right)\cdot\frac{\Pr(r_t|y_t=0)}{\Pr(r_t|y_t=1)}}{\frac{\alpha_{t-1}(0)}{\alpha_{t-1}(1)}+\frac{\Pr(x_t=0)}{\Pr(x_t=1)}}. \tag{2.18}$$

Define

$$\bar{\alpha}_t \triangleq \log\frac{\alpha_t(0)}{\alpha_t(1)}, \tag{2.19}$$

$$L_{ch}(y_t) \triangleq \log\frac{\Pr(r_t|y_t=0)}{\Pr(r_t|y_t=1)}, \tag{2.20}$$

$$L_o(x_t) \triangleq \log\frac{\Pr(x_t=0)}{\Pr(x_t=1)}. \tag{2.21}$$

Taking logarithm on both sides of (2.18), we get

$$\bar{\alpha}_t = \log\frac{e^{\bar{\alpha}_{t-1}}\cdot e^{L_o(x_t)}+1}{e^{\bar{\alpha}_{t-1}}+e^{L_o(x_t)}} + L_{ch}(y_t), \tag{2.22}$$

$$= \left(\bar{\alpha}_{t-1}\boxplus L_o(x_t)\right) + L_{ch}(y_t). \tag{2.23}$$

Likewise, in the backward recursion, we have

$$\bar{\beta}_t \triangleq \log\frac{\beta_t(0)}{\beta_t(1)}, \tag{2.24}$$

$$= \log\frac{\gamma_{t+1}(0,0)\beta_{t+1}(0)+\gamma_{t+1}(0,1)\beta_{t+1}(1)}{\gamma_{t+1}(1,0)\beta_{t+1}(0)+\gamma_{t+1}(1,1)\beta_{t+1}(1)}, \tag{2.25}$$

$$= \log\frac{\frac{\Pr(x_{t+1}=0)}{\Pr(x_{t+1}=1)}\cdot\frac{\Pr(r_{t+1}|y_{t+1}=0)}{\Pr(r_{t+1}|y_{t+1}=1)}\cdot\frac{\beta_{t+1}(0)}{\beta_{t+1}(1)}+1}{\frac{\Pr(x_{t+1}=0)}{\Pr(x_{t+1}=1)}+\frac{\Pr(r_{t+1}|y_{t+1}=0)}{\Pr(r_{t+1}|y_{t+1}=1)}\cdot\frac{\beta_{t+1}(0)}{\beta_{t+1}(1)}}, \tag{2.26}$$

$$= \log\frac{e^{L_o(x_{t+1})}\cdot e^{L_{ch}(y_{t+1})+\bar{\beta}_{t+1}}+1}{e^{L_o(x_{t+1})}+e^{L_{ch}(y_{t+1})+\bar{\beta}_{t+1}}}, \tag{2.27}$$

$$= L_o(x_{t+1})\boxplus\left(L_{ch}(y_{t+1})+\bar{\beta}_{t+1}\right). \tag{2.28}$$

Finally we compute the output (extrinsic) information using

$$\Lambda_t = \log \frac{\Pr(x_t = 0|\mathbf{Y}_1^N)}{\Pr(x_t = 1|\mathbf{Y}_1^N)}, \tag{2.29}$$

$$= \log \frac{\sum_m \sum_{m'} \alpha_{t-1}(m')\gamma_t(x_t = 0, m'm)\beta_t(m)}{\sum_m \sum_{m'} \alpha_{t-1}(m')\gamma_t(x_t = 1, m'm)\beta_t(m)}, \tag{2.30}$$

$$= \log \frac{\alpha_{t-1}(0)\Pr(r_t|y_t = 0)\beta_t(0) + \alpha_{t-1}(1)\Pr(r_t|y_t = 1)\beta_t(1)}{\alpha_{t-1}(0)\Pr(r_t|y_t = 1)\beta_t(1) + \alpha_{t-1}(1)\Pr(r_t|y_t = 0)\beta_t(0)}. \tag{2.31}$$

Dividing the numerator and denominator by $\alpha_{t-1}(1)\Pr(r_t|y_t = 1)\beta_t(1)$, we get

$$\Lambda_t = \log \frac{e^{\bar{\alpha}_{t-1}} \cdot e^{L_{ch}(y_t)+\bar{\beta}_t} + 1}{e^{\bar{\alpha}_{t-1}} + e^{L_{ch}(y_t)+\bar{\beta}_t}}, \tag{2.32}$$

$$= \bar{\alpha}_{t-1} \boxplus (L_{ch}(y_t) + \bar{\beta}_t). \tag{2.33}$$

It can then be seen that the message-passing algorithm described in the previous section can be derived from the BCJR algorithm where $\bar{\alpha}_t = L_{ch}(y_t) + L_{e_f}(y_t)$, $\bar{\beta}_t = L_{e_b}(y_t)$ and $\Lambda_t = L_e(x_t)$. For clarity, Table I summarizes the above results from the BCJR algorithm and compares them with the message-passing algorithm described in the previous section. The key advantage however is that the message-passing decoding obviates the need to compute $\log(e^\alpha + e^\beta)$ and the need to explicitly normalize at each step. Instead, a single operation $\log(\tanh \frac{\alpha}{2})$ is used which can be implemented using table lookup. As such, a significant amount of complexity is saved. Hence, the message-passing decoding of $1/(1 \oplus D)$ presents an efficient alternative of the conventional BCJR algorithm. □

The message-passing algorithm used by Jin *et al.* [22] and Divsalar *et al.* [41] is a parallel version of the sequential update of $L_{e_f}^{(k)}$ and $L_{e_b}^{(k)}$ in (2.7) and (2.8). This is desired in hardware implementation with one processing unit for each bit/check.

Table I. Summary of sum-product and MAP decoding of $1/(1 \oplus D)$.

	BCJR	Sum-product
forward pass	$\bar{\alpha}_t = (\bar{\alpha}_{t-1} \boxplus L_o(x_t)) + L_{ch}(y_t)$	$L_{e_f}(y_t) = (L_{e_f}(y_{t-1}) + L_{ch}(y_{t-1})) \boxplus L_o(x_t)$
backward pass	$\bar{\beta}_t = (\bar{\beta}_{t+1} + L_{ch}(y_{t+1})) \boxplus L_o(x_{t+1})$	$L_{e_b}(y_t) = (L_{e_b}(y_{t+1}) + L_{ch}(y_{t+1})) \boxplus L_o(x_{t+1})$
extrinsic LLR	$\Lambda_t = \bar{\alpha}_{t-1} \boxplus (\bar{\beta}_t + L_{ch}(y_t))$	$L_e(x_t) = (L_{e_f}(y_{t-1}) + L_{ch}(y_{t-1})) \boxplus (L_{e_b}(y_t) + L_{ch}(y_t))$

Equations (2.7) and (2.8) can be conveniently modified as

$$L_{e_f}^{(k)}(y_i) = L_o^{(k-1)}(x_i) \boxplus (L_{ch}(y_{i-1}) + L_{e_f}^{(k-1)}(y_{i-1})), \tag{2.34}$$

$$L_{e_b}^{(k)}(y_i) = L_o^{(k-1)}(x_{i+1}) \boxplus (L_{ch}(y_{i+1}) + L_{e_b}^{(k-1)}(y_{i+1})). \tag{2.35}$$

Obviously, since the parallel version uses the information from the last iteration rather than the most recent, the convergence may be a little slower. But for practical block sizes and for moderate decoding time, simulations have shown that the compromise in performance is very little, about 0.1 dB or so after 15 to 30 iterations.

2. Min-Sum Algorithm

The main complexity in the decoder comes from the $\boxplus$ operation in both the outer TPC/SPC and inner $1/(1 \oplus D)$ decoding. Each turbo iteration (composed of one round of $1/(1 \oplus D)$ decoding followed by one round of TPC/SPC decoding[1] re-

[1]Simulation results show that the best performance/complexity gain is achieved with only one local iteration of TPC/SPC decoding in each turbo iteration between the inner and outer decoders.

quires at least 5 $\boxplus$ operations per coded bit. A straight-forward implementation of $\boxplus$ may require as many as 1 addition and 3 table lookups (assuming $\log(\tanh(\cdot))$ and $\log(\tanh^{-1}(\cdot))$ are implemented via table lookups). Although this is already lower complexity than turbo codes, it is possible and highly practical to further reduce the complexity with a slight compromise in performance. Just like the Max-log-MAP algorithm in turbo codes, the $\boxplus$ operation has a similar approximation

$$\begin{aligned}
\gamma &= \alpha \boxplus \beta, && (2.36)\\
&= 2\tanh^{-1}\big(\tanh\frac{\alpha}{2}\cdot\tanh\frac{\beta}{2}\big), && (2.37)\\
&= \log\frac{1+e^{\alpha+\beta}}{e^{\alpha}+e^{\beta}}, && (2.38)\\
&= \mathrm{sign}(\alpha)\cdot\mathrm{sign}(\beta)\cdot\min(|\alpha|,|\beta|)+\log\frac{1+e^{-|\alpha+\beta|}}{1+e^{-|\alpha-\beta|}}, && (2.39)\\
&\approx \mathrm{sign}(\alpha)\cdot\mathrm{sign}(\beta)\cdot\min(|\alpha|,|\beta|). && (2.40)
\end{aligned}$$

If the approximation in (2.40) is used, i.e., a mere "min" operation is used instead of $\boxplus$, then a considerable reduction in complexity is achieved, and the message-passing algorithm, or the sum-product algorithm, is then reduced to the min-sum algorithm.

Lemma 2: *Min-sum decoding of* $1/(1\oplus D)$ *is equivalent to Max-log-MAP decoding.*

Proof: Now that we have shown the equivalence of the sum-product decoding and the MAP decoding for the code $1/(1\oplus D)$, it is straightforward to show the equivalence of the min-sum decoding and the max-log-MAP decoding. Max-log-MAP decoding is a suboptimal algorithm of the MAP or log-MAP, where the calculation of $\log(e^{\alpha}+e^{\beta})$ is approximated as

$$\log(e^{\alpha}+e^{\beta})\approx\max(\alpha,\beta). \qquad (2.41)$$

Likewise, the only difference between the min-sum algorithm and the message-

passing algorithm is that a simple "min" operation is used instead of $\boxplus$ operation. This approximation is in fact a direct derivation of (2.41)

$$\begin{aligned} \gamma &= \alpha \boxplus \beta, && (2.42)\\ &= \log(e^0 + e^{\alpha+\beta}) - \log(e^\alpha + e^\beta), && (2.43)\\ &\approx \max(0,\ \alpha+\beta) - \max(\alpha,\ \beta), && (2.44)\\ &= \text{sign}(\alpha)\cdot\text{sign}(\beta)\cdot\min(\alpha,\ \beta). && (2.45) \end{aligned}$$

It thus follows that the min-sum algorithm is a computationally efficient realization of the Max-log-MAP algorithm for the decoding of $1/(1 \oplus D)$. $\square$

D. Properties of Product Accumulate Codes

Before going through numerical results, we first show some properties of product accumulate codes to facilitate the understanding of their performance. The proposed product accumulate codes possess the following properties [23] [24]:

1. **Property I**: Product accumulate codes are linear time encodable and linear time decodable.

2. **Property II**: They are "good" under ML decoding approach, which assures their good performance asymptotically.

3. **Property III**: They are "good" under the practical iterative decoding approach.

To show Property I, we conduct complexity analysis for the encoding and decoding procedures. To show Property II, we quantify the interleaving gain whose existence shows that the bit error rate vanishes as the block/interleaver size goes

Table II. Decoding complexity (operations per data-bit per iteration).

Code	outer TPC/SPC		inner $1/(1 \oplus D)$			
decoder	sum-pro.	min-sum	log-map	max-log-map	sum-pro.	min-sum
addition	$5 + \frac{1}{R}$	2	$39/R$	$31/R$	$5/R$	$2/R$
min	-	$5 - \frac{1}{R}$	$8/R$	$8/R$	-	$3/R$
lookup	$2 + \frac{2}{R}$	-	$8/R$	-	$5/R$	-

R: code rate

to infinity. We will also compute the ensemble distance spectrum and derive tight upper bounds to facilitate the understanding from the ML perspective. To show Property III, we will use density evolution to compute the thresholds which mark the performance limit of PA codes using the existing iterative (suboptimal) decoder.

1. Encoding and Decoding Complexity

The encoding and decoding complexity for PA codes are linear in the length of the codewords. The encoding process involves only parity check in each dimension of the outer code (see Appendix A), interleaving and encoding of a rate-1 inner convolutional code (see Figure 8), the complexity is linear in the block length. The decoding complexity is proportional to the number of iterations of the outer TPC/SPC code and the inner convolutional code, both of which have linear decoding complexity.

Table II summarizes the complexity of different decoding strategies for the inner and outer code. We assume that in sum-product decoding, $\log(\tanh \frac{\alpha}{2})$ is implemented using table lookup. The complexity of the log-MAP and the Max-log-MAP algorithms is evaluated using [26] (based on the conventional implementation of the BCJR algorithm). As can be seen, the sum-product and the min-sum decoding of

$1/(1 \oplus D)$ requires only about 1/6 and 1/8 the complexity of their BCJR equivalents. For a rate 1/2 PA code, message-passing decoding requires about 33 operations per data bit per iteration, while min-sum decoding requires only about 15 operations, which is significantly less than the number of operations involved in a turbo code.

2. Performance under ML Decoding

In the ML-based analysis of PA codes, we first quantify the interleaving gain and then derive a tight upper bound on the word error performance. We show that under maximum likelihood decoding, the probability of word error is proportional to P^{-1} for large E_b/N_o, where P is the number of TPC/SPC codewords concatenated before interleaving. Further, we show that these codes can perform close to capacity by computing thresholds for these codes based on the tight upper bound on the word error rate due to Divsalar [41].

a. Ensemble Distance Spectrum and Interleaving Gain

Although Property III that the proposed codes are "good" under iterative decoding implies that they are also "good" under ML decoding, it is instructive to study the distance spectrum of PA codes. From the results of Benedetto *et al.* [25] and Divsalar, Jin and McEliece [21], we know that for a general serial concatenated system with recursive inner code there exists a threshold γ such that for any $E_b/N_o \geq \gamma$, the asymptotic word error rate is upper bounded by

$$P_w^{UB} = O\left(N^{-\lfloor \frac{d_m^o - 1}{2} \rfloor}\right), \tag{2.46}$$

where d_m^o is the minimum distance of the outer code and N is the interleaver size. Although the above results offer a useful guideline for designing concatenated schemes, it is well worth computing the exact interleaving for GPA codes since it reveals some-

what important and interesting results that are not obvious from (2.46).

The result in (2.46) indicates that if the minimum distance of the outer code is at least 3, then an interleaving gain can be obtained. However, the outer codewords of PA codes (with random interleavers) have minimum distance of only 2. On the other hand, if S-random interleavers are used such that bits within S distance are mapped to at least S distance apart, then the outer codewords are guaranteed to have a minimum distance of at least 3 as long as $S \geq t$. Since a block interleaver can be viewed as a structured S-random interleaver, it follows that interleaving gain exists for PA-II codes. Below we show that although the minimum distance of the outer codewords is only 2 over the ensemble of interleavers, an interleaving gain still exists for PA codes with random interleavers (PA-I codes). Since from (2.46), outer codewords of weight 3 or more will lead to an interleaver gain, we focus the investigation on weight-2 outer codewords only, and show that the number vanishes as P increases. The all-zero sequence is used as the reference since the code is linear.

It is convenient to employ the uniform interleaver which represents the average behavior of the ensemble of the codes. Let $A_{w,h}^{(j)}, j = 1, 2$, denote the *input output weight enumerator* (IOWE) of the j_{th} SPC branch code (concatenated in parallel with the outer code). The IOWE of the outer codewords, $A_{w,h}^{(o)}$, averaged over the code ensemble is given as

$$A_{w,h}^{(o)} = \sum_{h_1} \frac{A_{w,h_1}^{(1)} A_{w,h-h_1}^{(2)}}{\binom{K}{w}}, \tag{2.47}$$

where $K = Pt$ is the input sequence length.

Define the *input-output weight transfer probability* (IOWTP) of the j_{th} branch code, $P_{w,h}^{(j)}$, as the probability that a particular input sequence of weight w is mapped

to an output sequence of weight h

$$P_{w,h}^{(j)} = \frac{A_{w,h}^{(j)}}{\binom{K}{w}}. \tag{2.48}$$

Substituting (2.48) to (2.47), we get

$$A_{w,h}^{(o)} = \sum_{h_1} A_{w,h_1}^{(1)} P_{w,h-h_1}^{(2)}. \tag{2.49}$$

For each branch where P $(t+1,t)$ SPC codewords are combined, the IOWE function is given as (assuming even parity check)

$$\begin{aligned} A^{SPC}(w,h) &= \left(1+\binom{t}{1}wh^2+\binom{t}{2}w^2h^2+\binom{t}{3}w^3h^4+\cdots+\binom{t}{t}w^th^{2\lceil t/2\rceil}\right)^P, && (2.50)\\ &= \left(\sum_{j=0}^{t}\binom{t}{j}w^jh^{2\lceil j/2\rceil}\right)^P, && (2.51) \end{aligned}$$

where the coefficient of the term w^uh^v denotes the number of codewords with input weight u and output weight v. Using (2.51), we can compute the IOWEs of the first SPC branch code, denoted as $A_{u,v}^{(1)}$ $(= A_{u,v}^{SPC})$. For the second branch of SPC code, since only parity bits are transmitted, $A_{u,v}^{(2)} = A_{u,v+u}^{(1)}$.

With a little computation, it is easy to see that the number of weight-2 outer codewords is given by

$$\begin{aligned} A_{h=2}^{(o)} &= \sum_w A_{w,h=2}^{(o)}, && (2.52)\\ &= P\binom{t}{2}\frac{P\binom{t}{2}}{\binom{Pt}{2}}, && (2.53)\\ &= O(t^2), && (2.54) \end{aligned}$$

where the last equation assumes a large P (i.e., large block size). Equation (2.54) shows that the number of weight-2 outer codewords is a function of a single parameter, t, which is related only to the rate of SPC codes and not the block length. Now

considering the serial concatenation of the outer codewords with the inner $1/(1 \oplus D)$ code, the overall output weight enumerator (OWE), A_h^{PA}, is computed as

$$A_{h=s}^{PA} = \sum_{h'} A_{h'}^{(o)} \frac{A_{h',h}^{1/(1+D)}}{\binom{N}{h'}}, \tag{2.55}$$

$$= \sum_{h'} \sum_{w} A_{w,h'}^{(o)} \frac{A_{h',h}^{1/(1+D)}}{\binom{N}{h'}}, \tag{2.56}$$

where the IOWE of $1/(1 \oplus D)$ code is given by [21]

$$A_{w,h}^{1/(1\oplus D)} = \binom{N-h}{\lfloor w/2 \rfloor} \binom{h-1}{\lceil w/2 \rceil - 1}. \tag{2.57}$$

In particular, the number of weight-s PA codewords produced by weight-2 outer codewords (for small-s), denoted as $A_{h=s}^{PA2}$, is given as

$$A_{h=s}^{PA2} = \frac{(t-1)^2}{2} \frac{N-s}{\binom{N}{2}}, \tag{2.58}$$

$$= O(tP^{-1}), \tag{2.59}$$

where $N = P(t+2)$ is the PA codeword length. This indicates that the number of small weight s codewords of the overall PA code due to weight-2 outer codewords (caused by weight-2 input sequences) vanishes as P increases. When the input weight is greater than 2, the outer codeword always has weight greater than 2 and, hence, an interleaving gain can be guaranteed. Hence, an interleaving gain exists for PA codes and it is proportional to P.

b. Upper bounds

To further our insight into the asymptotic performance ($N \to \infty$) of PA codes under ML decoding, we compute thresholds for this class of codes based on the bounding technique recently proposed by Divsalar [41]. The threshold here refers to the capacity of the codes under ML decoding, i.e., the minimum E_b/N_o for which the probability

of error decreases exponentially in N and, hence, tends to zero as $N \to \infty$.

Among the various bounding techniques developed, the union bound is the most popular but is fairly loose above the cutoff rate. Tighter and more complicated bounds include the tangential sphere bound by Poltyrev [49], the Viterbi and Viterbi bound [50], Duman-Salehi bound [51], the Hughes bound [52]. These new tight bounds are essentially based on the bounding techniques developed by Gallager [9]

$$\Pr(error) \leq \Pr(error, \bar{y} \in \Re) + \Pr(\bar{y} \notin \Re), \tag{2.60}$$

where $\bar{y}$ is the received codeword (noise-corrupted), and $\Re$ is a region in the observed space around the transmitted codeword. To get a tight bound, the above methods usually require optimization and integration to determine a meaningful $\Re$.

Recently, Divsalar developed a *simple bound* on error probability over AWGN channels [41]. The bound is also based on (2.60), but a simple closed-form expression is derived and shown that the computed minimum SNR threshold can serve as a tight upper bound on the ML capacity of nonrandom codes. The simple bound is the tightest closed-from bound developed so far. It is also shown that, as block size goes to infinity, this simple bound is equivalent to the tangential sphere bound [41]. Below we apply this simple bounding technique to the analysis of PA codes.

We first quote and summarize the main results of [41]. Define the *spectral shape* of a code, $\gamma_N(\delta)$, as the normalized weight distribution averaged over the code ensemble $\mathcal{C}_N$

$$\gamma_N(\delta) \stackrel{\Delta}{=} \frac{1}{N} \ln(A_{h=\lfloor \delta N \rfloor}), \qquad 0 < \delta < 1, \tag{2.61}$$

where N is the code length, and A_h is the (average) output weight enumerator of the

code. Further, define the *ensemble spectral shape* as

$$\gamma(\delta) \stackrel{\Delta}{=} \lim_{N\to\infty} r_N(\delta), \qquad 0 < \delta < 1. \tag{2.62}$$

It can be shown that the probability of word error can be upper bounded by

$$P_w(e) \leq \sum_h e^{-NE(E_b/N_o,h)}. \tag{2.63}$$

The threshold $\mathbf{C}^*_{ML}$ is defined as the minimum E_b/N_o such that $E(E_b/N_o, h)$ is positive for all h and, hence, for all $E_b/N_o \geq \mathbf{C}^*_{ML}$, $P_w(e) \to 0$ as $N \to \infty$. The threshold can be computed as [41]

$$\mathbf{C}^*_{ML} = \frac{1}{R} \max_{0<\delta\leq(1-R)} c_0(\delta), \tag{2.64}$$

where R is the code rate. For the simple bound, $c_0(\delta)$ is given by:

$$\text{Simple:} \quad c_0(\delta) = \frac{1-\delta}{2\delta}\left(1 - e^{-2\gamma(\delta)}\right). \tag{2.65}$$

Similar forms are also derived for Viterbi-Viterbi bounds, Hughes bounds, and Union bounds [41]

$$\text{Viterbi:} \quad c_0(\delta) = \frac{(1-\delta)}{\delta}\gamma(\delta), \tag{2.66}$$

$$\text{Hughes:} \quad c_0(\delta) = \frac{1}{2\delta}\left(1 - e^{-2\gamma(\delta)}\right), \tag{2.67}$$

$$\text{Union:} \quad c_0(\delta) = \frac{\gamma(\delta)}{\delta}. \tag{2.68}$$

Since the above bounds are based on the ensemble spectral shape $r(\delta)$, they serve as the asymptotic performance limit (i.e., $N \to \infty$) of the code ensemble assuming ML decoding.

There is no simple closed form expression for the ensemble spectral shape of PA codes. However, the spectral shape can be computed to a good accuracy numerically

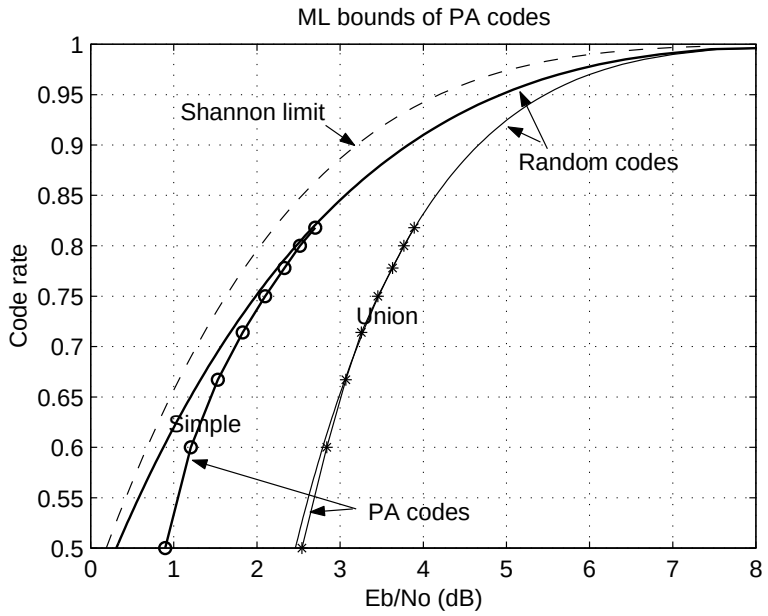


Fig. 12. The union bound and the simple bound of product accumulate codes (PA-I).

since the component codes of the concatenation are single parity check codes. Specifically, using (2.49), (2.56) and (2.61) we can compute the spectral shape of PA codes, which is a function of N, P, t. We approximate the ensemble spectral shape by choosing a large N. Whenever possible, the input output weight transfer probability, $P_{w,h}$, should be used instead of input output weight enumerator, $A_{w,h}$, to eliminate numerical overflow. The bounds for GPA codes are computed and plotted in Figure 12 (for clarity, only the simple bound and the union bound are shown). For comparison, also shown are the bounds for random codes and the Shannon limit. Several things can be observed: 1) the simple bounds of PA codes are very close to those of random codes, indicating that PA codes have good distance spectrum; 2) the higher the rates, the tighter the bounds, indicating that GPA codes are probably more advantageous at high rates than low rates (as opposed to repeat accumulate codes).

The implication of the above analysis is that PA codes are capable of performance a few tenths of a dB away from the capacity limit with ML decoding. However,

since there does not exist a computationally feasible ML decoder, it is desirable to investigate iterative decoding to give a more meaningful evaluation of the code performance with a practical decoder.

3. Asymptotic Performance under Iterative Decoding

In this section we show that product accumulate codes are "good" codes also in the iterative sense. We show this by computing a threshold (minimum E_b/N_o) for this class of codes, such that when the channel signal-to-noise ratio is higher than the threshold the error probability goes to zero (with infinite block size). We compute this threshold by means of density evolution. Density evolution has been shown to be a very powerful tool in the analysis and design of LDPC codes [12] [14] [11] [32]. Following the message-passing decoding on the code graph (Figure 10), we will explain how the DE procedure can be applied to compute the thresholds for PA codes. By examining the distribution of the messages passed within and in-between the subdecoders, we are able to determine the fraction of incorrect messages (extrinsic messages of the wrong sign). The basic idea is that if the fraction of incorrect messages goes to zero with the increase of iterations, then the decoding procedure will eventually converge to the correct codeword.

The analysis of product accumulate codes involves computation of the probability density function (pdf) of the message flow within the outer decoder, the inner decoder and in-between the two. However the unconstrained DE procedure is quite complex since the pdf that evolves with iterations may not have closed-form expressions and one must keep track of an infinite dimensional vector. It is worth mentioning that a simplified approximation can be made by assuming that the messages passed in each step follow Gaussian distributions. This Gaussian assumption trades a little accuracy for a considerate reduction in computational complexity when combined

with the consistency condition which states that the distribution f of messages w passed in each step satisfies $f(w) = f(-w)e^w$ [32]. Here, to preserve the accuracy, we perform the exact density evolution.

Exploiting the linearity of the code, we assume the all-zero codeword is transmitted. It is convenient to use log-likelihood ratios as messages to examine the decoding process. The threshold, which serves as the practical capacity limit for a given code (given rate and decoding strategy), is thus formulated as

$$\mathbf{C}^*_{iterative} = \inf_{SNR} \left\{ SNR : \lim_{k\to\infty} \lim_{N\to\infty} \int_{-\infty}^{0} f_{L^{(k)}_{o,ext}(x)} dx \to \infty \right\}, \tag{2.69}$$

where $f_{L^{(k)}_{o,ext}(x)}$ is the pdf of the messages evaluated at the output of the outer decoder, (note that due to the i.i.d. assumption, we have dropped the dependence i on x_i,) superscript (k) denotes the k_{th} iteration, and N is the block size. Before we describe how DE is performed numerically for the case of PA codes, we first discretize messages. Let $\mathrm{Q}(w)$ denote the quantization operation on message w with a desired quantization interval (accuracy) Δ.

a. Message Flow within the Outer Decoder

The outer code of the general product codes (PA-I) consists of 2 parallel concatenated branches where each branch is formed of P blocks of $(t+1, t)$ SPC codewords. The outer code (alone) can also be considered as a special case of LDPC codes whose parity check matrix has $2P$ rows with uniform row weight of $(t+1)$, and $(t+1)^2$ columns with $\frac{t}{t+2}$ percent of the columns having weight 2 and the rest weight 1. Therefore, the exact decoding algorithm for LDPC codes can be applied to the outer code. However, for a more efficient convergence, we could make use of the fact that the checks in the outer code can be divided into two groups (corresponding to the upper and lower branch, respectively) such that the corresponding sub-graph (Tanner graph) of each

group is cycle-free. It thus leads to a serial message-passing mode where each group of checks takes turns to update (as opposed to the parallel update of all checks in LDPC codes).

The fundamental element in the decoding of the outer code is the decoding of SPC codes. Consider the upper branch. Suppose data bits $d_{i,1}, d_{i,2}, \cdots, d_{i,t}$ and parity bit p_i participate in the i_{th} SPC codeword ($1 \le i \le P$). Then the messages (extrinsic information) for each bit obtained from this check (during the k_{th} turbo iteration and l_{th} local iteration) are

$$\begin{aligned}
\text{data bit:} \quad L_{e1}^{(k,l)}(d_{i,j}) &= \left(\sum_{1\le k\le t, k\ne j}^{\boxplus} \left(L_o^{(k)}(d_{i,k}) + L_{e2}^{(k,l-1)}(d_{i,k}) \right) \right) \boxplus L_o^{(k)}(p_i), \\
\iff \tanh \frac{L_{e1}^{(k,l)}(d_{i,j})}{2} &= \tanh \frac{L_o^{(k)}(p_i)}{2} \prod_{1\le k\le t, k\ne j} \tanh \frac{L_o^{(k)}(d_{i,k}) + L_{e2}^{(k,l-1)}(d_{i,k})}{2}, \qquad (2.70) \\
\text{parity bit:} \quad L_{e1}^{(k,l)}(p_i) &= \left(\sum_{1\le k\le t}^{\boxplus} \left(L_o^{(k)}(d_{i,k}) + L_{e2}^{(k,l-1)}(d_{i,k}) \right) \right), \\
\iff \tanh \frac{L_{e1}^{(k,l-1)}(p_i)}{2} &= \prod_{1\le k\le t} \tanh \frac{L_o^{(k)}(d_{i,k}) + L_{e2}^{(k,l-1)}(d_{i,k})}{2}, \qquad (2.71)
\end{aligned}$$

where $L_o(\cdot)$ denotes the messages (*a priori* information) received from the inner code, $L_{e1}(\cdot)$ denotes the messages (extrinsic information) obtained from the upper SPC branch to be passed to the lower branch and $L_{e2}(\cdot)$ denotes the messages to be passed from the lower branch to the upper branch. After interleaving, similar operations of (2.70) and (2.71) are performed within the lower branch. We assume $L_{e1}(\cdot)$ and $L_{e2}(\cdot)$ to be independent and identical distributed (i.i.d) and drop the dependence on $d_{i,j}$ and p_i.

We use superscript (k,l) to denote the k_{th} turbo iteration between the outer decoder and the inner decoder and the l_{th} iteration within the outer decoder (local iterations). For independent messages to add together, the resulting pdf of the

sum is the discrete convolution of the component pdf's. This calculation can be efficiently implemented using an fast Fourier transform (FFT). For the tanh operation on messages, define

$$\gamma = \mathcal{R}(\alpha, \beta) \stackrel{\Delta}{=} \mathrm{Q}\left(2\tanh^{-1}\left(\tanh\frac{\alpha}{2}\tanh\frac{\beta}{2}\right)\right), \tag{2.72}$$

where α, β and γ are quantified messages. The pdf f_γ of γ can be computed using

$$f_\gamma[k] = \sum_{(i,j):\, k\Delta = \mathcal{R}(i\Delta, j\Delta)} f_\alpha[i] \cdot f_\beta[j]. \tag{2.73}$$

To simplify the notation, we denote this operation (2.73) as

$$f_\gamma = \tilde{\mathcal{R}}(f_\alpha, f_\beta). \tag{2.74}$$

In particular, using induction on the above equation, we can denote

$$\tilde{\mathcal{R}}^k(f_\alpha) \stackrel{\Delta}{=} \underbrace{\tilde{\mathcal{R}}(f_\alpha, (\tilde{\mathcal{R}}(f_\alpha, \cdots, \tilde{\mathcal{R}}(f_\alpha, f_\alpha)\cdots))}_{k \text{ levels of } \tilde{\mathcal{R}}}. \tag{2.75}$$

It then follows from (2.70), (2.71) and (2.75) that the pdf of the extrinsic messages obtained from the upper branch $f_{Le1}(\cdot)$ and the lower branch, $f_{Le2}(\cdot)$, are given by

$$\text{Upper branch: data:} \quad f_{L_{e1}}^{(k,l)}(d) = \tilde{\mathcal{R}}\Big(f_{L_o}^{(k)}(p),\, \tilde{\mathcal{R}}^{l-1}\big(f_{L_o}^{(k)}(d) * f_{L_{e2}}^{(k,l-1)}(d)\big)\Big), \tag{2.76}$$

$$\text{parity:} \quad f_{L_{e1}}^{(k,l)}(p) = \tilde{\mathcal{R}}^{l}\Big(f_{L_o}^{(k)}(d) * f_{L_{e2}}^{(k,l-1)}(d)\Big), \tag{2.77}$$

$$\text{Lower branch: data:} \quad f_{L_{e2}}^{(k,l)}(d) = \tilde{\mathcal{R}}\Big(f_{L_o}^{(k)}(p),\, \tilde{\mathcal{R}}^{l-1}\big(f_{L_o^{(k)}}(d) * f_{L_{e1}}^{(k,1)}(d)\big)\Big), \tag{2.78}$$

$$\text{parity:} \quad f_{L_{e2}}^{(k,l)}(p) = \tilde{\mathcal{R}}^{l}\Big(f_{L_o}^{(k)}(d) * f_{L_{e1}}^{(k,l)}(d)\Big), \tag{2.79}$$

where $f_{L_o}^{(k)}(\cdot)$ denotes the pdf of the messages $L_o^{(k)}(\cdot)$ from the inner $1/(1 \oplus D)$ code in the k_{th} turbo iteration, $f_{L_{e1}}^{(k,l)}(\cdot)$ and $f_{L_{e2}}^{(k,l)}(\cdot)$ denote the pdf's of the extrinsic information from the upper and lower branch of the outer code, $L_{e1}^{(k,l)}(\cdot)$ and $L_{e2}^{(k,l)}(\cdot)$, respectively, and $*$ denotes discrete convolution.

Since the systematic bits (data) and the parity bits of the outer code are treated the same in the inner $1/(1 \oplus D)$ code, we have $f_{L_o}^{(k)}(d) = f_{L_o}^{(k)}(p) = f_{L_{e,x}}^{(k)}$, where $f_{L_{e,x}}^{(k)}$ is pdf of the extrinsic information $L_{e,x}^{(k)}$ obtained from $1/(1 \oplus D)$ (refer to the next section for a detailed explanation). For PA-I codes, the local iterations within the outer code only involve the exchange of messages associated with data bits (as can be seen from the above equations). After L local iterations, the messages the outer code passes along to the inner code include those of data bits ($L_{e1}(d)$ and $L_{e2}(d)$) and parity bits ($L_{e1}(p)$ and $L_{e2}(p)$), which leads to a mixed-Gaussian message density with a fraction $t/(t+2)$ having pdf $(f_{L_{e1}}(d) * f_{L_{e2}}(d))$ and a fraction $1/(t+2)$ having mean $f_{L_{e1}}(p)$ and $f_{L_{e2}}(p)$, respectively. This will in turn serve as the pdf of the *a priori* information, $f_{L_{o,x}}^{(k+1)}$, to the inner decoder.

A similar serial update procedure can also be used with PA-II codes, and the message-passing analysis is much the same. For a PA-II code with the conventional $(K_1 + 1, K_1) \times (K_2 + 1, K_2)$ TPC/SPC codes (with block interleavers and parity-on-parity bits) as the outer code, the means of the extrinsic messages associated with row code and column code, $L_{e1}(\cdot)$ and $L_{e2}(\cdot)$, can be computed using (also refer to Appendix I for the decoding algorithm of TPC/SPC codes)

$$f_{L_{e1}}^{(k,l)} = \tilde{\mathcal{R}}^{K_1}\left(f_{L_o}^{(k)} * f_{L_{e2}}^{(k,l-1)}\right), \tag{2.80}$$

$$f_{L_{e2}}^{(k,l)} = \tilde{\mathcal{R}}^{K_2}\left(f_{L_o}^{(k)} * f_{L_{e1}}^{(k,l)}\right). \tag{2.81}$$

Unlike PA-I codes, the data bits and the parity bits are treated exactly the same in the outer code of PA-II codes. Hence, the pdf of the messages passing along to the inner $1/(1 \oplus D)$ decoder is given by $(f_{L_{e1}}^{(k,L)} * f_{L_{e2}}^{(k,L)})$ after L rounds of local iterations.

b. Message Flow within the Inner $1/(1 \oplus D)$ Decoder

Similar to the treatment of TPC/SPC code, we assume that messages (LLRs) are i.i.d. in $1/(1 \oplus D)$ code also. From (2.34) and (2.35), it is obvious that for sufficiently long sequence, messages $L_{e_f}(y)$ and $L_{e_b}(y)$ follow the same distribution and, therefore, has the same mean value (denote it as $M_{e,y}$). Note we are somewhat abusing the notation here by dropping the dependencies on i, which denotes the transmission at the i_{th} epoch. This is because on a memoryless channel, the pdf's of $L_{e_f}(y_i)$ and $L_{e_b}(y_i)$ are independent of i. Further as can be seen from the message-passing algorithm, the forward and backward passes are symmetric and, hence, for large block sizes, $L_{e_f}(y)$ and $L_{e_b}(y)$ follow the same pdf's. Thus we drop the subscript and use $L_e(y)$ to represent both $L_{e_f}(y)$ and $L_{e_b}(y)$. It was verified by simulations that the serial (see (2.7) and (2.8)) and parallel (see (2.34) and (2.35)) modes do not differ in performance significantly (only about 0.1 dB as will be shown later), especially with sufficient number of turbo iterations. It is convenient to use the parallel mode for analysis here. Denote $M_{e,x}$, $M_{o,x}$ and $M_{ch,y}$ as the mean value of $L_e(x)$, $L_o(x)$ and $L_{ch}(y)$, which represent the messages passed from inner to outer code, from outer to inner code, and from channel to inner code, respectively (also see Figure 10). Hence messages (LLRs) as formulated in (2.4) and (2.34), (2.35) have their means evolve as

$$f_{Le,x}^{(k)} = \tilde{\mathcal{R}}^2\big(f_{L_{ch,y}} * f_{L_{e,y}}^{(k)}\big), \tag{2.82}$$

where

$$f_{L_{e,y}}^{(k)} = \tilde{\mathcal{R}}\big(\, f_{L_{o,x}}^{(k-1)},\, f_{L_{ch,y}} * f_{L_{e,y}}^{(k-1)}\,\big). \tag{2.83}$$

The initial conditions are $f_{L_{ch,y}} = \mathcal{N}(2/\sigma^2, 4/\sigma^2)$ (Gaussian distribution of mean $2/\sigma^2$ and variance $4/\sigma^2$) and $f_{L_{o,x}}^{(0)} = f_{L_{e,y}}^{(0)} = \delta(0)$ (Kronecker delta function).

c. Message Flow between the Inner and Outer Code

The message flow between the inner and the outer code is straight-forward. The pdf of the outbound message, $f_{L_{e,x}}^{(k)}$ in (2.82), becomes the pdf of the *a priori* information, $f_{L_o}^{(k)}(d)$ and $f_{L_o}^{(k)}(p)$ in (2.76) - (2.79) (PA-I code) and $f_{L_o}^{(k)}$ in (2.80) and (2.81) (PA-II code). Likewise, the pdf of the extrinsic information from the outer TPC/SPC code, $\left(\frac{t}{t+2}(f_{L_{e1}}^{(k,L)}(d) * f_{L_{e2}}^{(k,L)}(d)) + \frac{1}{t+2} f_{L_{e1}}^{(k,L)}(p) + \frac{1}{t+2} f_{L_{e2}}^{(k,L)}(p)\right)$ for PA-I codes and $(f_{L_{e1}}^{(k,L)} * f_{L_{e2}}^{(k,L)})$ for PA-II codes, becomes the pdf of *a priori* information, $f_{L_{o,x}}^{(k+1)}$ in (2.83), for the inner $1/(1 \oplus D)$ code. For clarity, a complete procedure of the discretized density evolution for product accumulate codes (PA-I) as well as the relevant notations are summarized in Appendix B.

It should be noted that the assumption that all messages passed are independent and identically distributed is required for the derivation of (2.70) and (2.71) for PA-I codes. However, the same cannot be directly used to analyze PA-II codes. Due to the use of the random interleaver in the PA-I code structure, it is reasonable to assume that the neighborhood of each node is tree-like. However, in the case of PA-II codes, when a block interleaver is used in the TPC/SPC code, length-8 cycles are unavoidable (even when $N \to \infty$). Hence, partial independence holds only when the message flow in the decoding has not closed a length-8 cycle. In other words, the number of times (2.80) and (2.81) can be applied consecutively is strictly limited to be no more than $\log_2 \frac{8}{2} = 2$, before messages need to be passed to the $1/(1 \oplus D)$ decoder [40]. In fact, as shown in [40], even 2 local iterations will incur the looping of the same message and, hence, we take $L = 1$ for analysis. Further, during every global iteration (k), the extrinsic messages within the TPC/SPC code generated in the previous iterations, $M_{e1}^{(k-1,L)}$ and $M_{e2}^{(k-1,L)}$, should not be used again since this represents correlated information. Due to the above reasons, the resulting thresholds

are an upper bound (pessimistic case).

Figure 13 illustrates how messages evolve with the number of iterations in a product accumulate code. From the trajectories, we can determine the threshold to be around 4.315 dB for a rate-0.94 PA-II code.

Figure 14 shows the thresholds for PA-I codes for several rates $R \geq 0.5$. It can be seen that the thresholds are within 0.7 dB from the Shannon limit for BPSK on an AWGN channel. The thresholds are closer as the rate increases, suggesting that these codes are better at higher rates. The thresholds for PA-II codes are shown in Figure 15. The plotted thresholds in Figure 15 are a lower bound on the capacity (upper bound on the thresholds) since only one iteration is performed in the outer TPC/SPC decoding in each turbo iteration (i.e., $L = 1$ in (2.81)) [40]. Note that at high rates ($R > 0.7$), the capacity of product accumulate codes (both PA-I an PA-II) is within 0.5 dB from the Shannon limit. However, at lower rates the gap becomes larger especially for PA-II codes. Simulation results for fairly long block sizes are also shown in both Figure 14 and Figure 15. A block size of $K = 64K$ data bits was used for $R = 1/2$ and for the higher rates $K = 16K$ was used and a BER of 10^{-5} is taken as reference. It can be seen that the simulation results are quite close to the thresholds. This shows that PA-I codes and PA-II codes are both capable of good performance at high rates; however, at lower rates PA-I codes are significantly better.

E. Algebraic Interleaver

Observe that a rate-K/N PA-I code involves two random interleavers of size K and N, where K and N are the user data block size and codeword block size, respectively. Interleaving and deinterleaving using look-up tables can be quite inefficient in hardware and, hence, we study the performance of PA codes under algebraic interleaving.

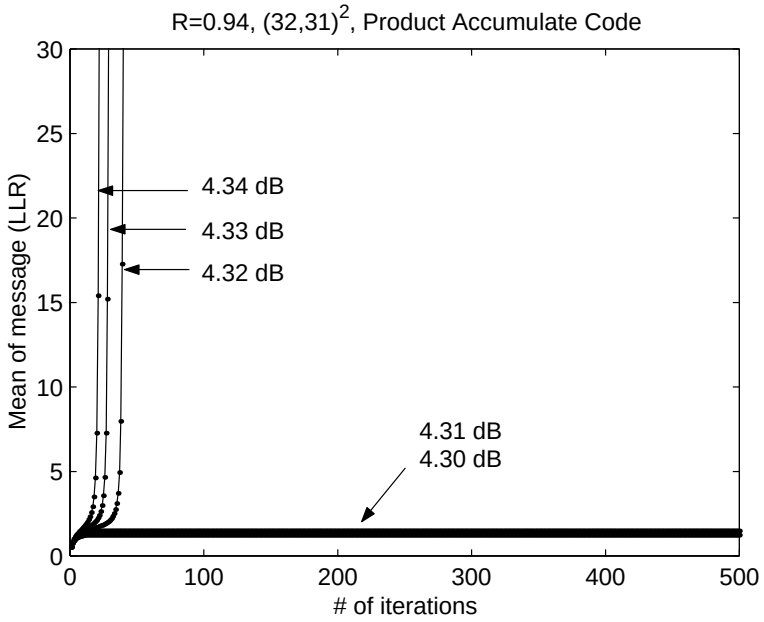


Fig. 13. Trajectories of the mean of the messages for a rate $R = (31/32)^2$ PA-II code.

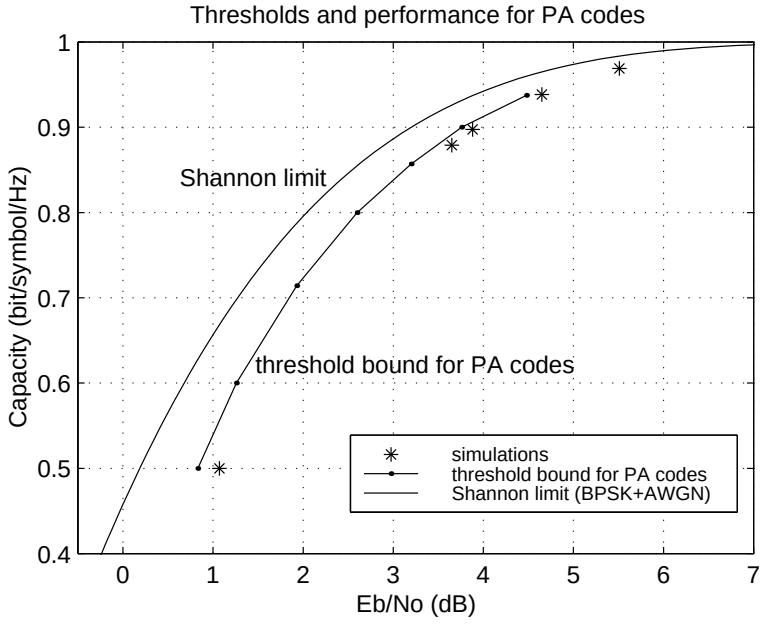


Fig. 14. Thresholds for PA-I codes. (Simulated PA-I codes have data block sizes K=32K for $R > 0.8$ and K=64K for $R = 0.5$. Performance is evaluated at BER of 1^{-5}.)

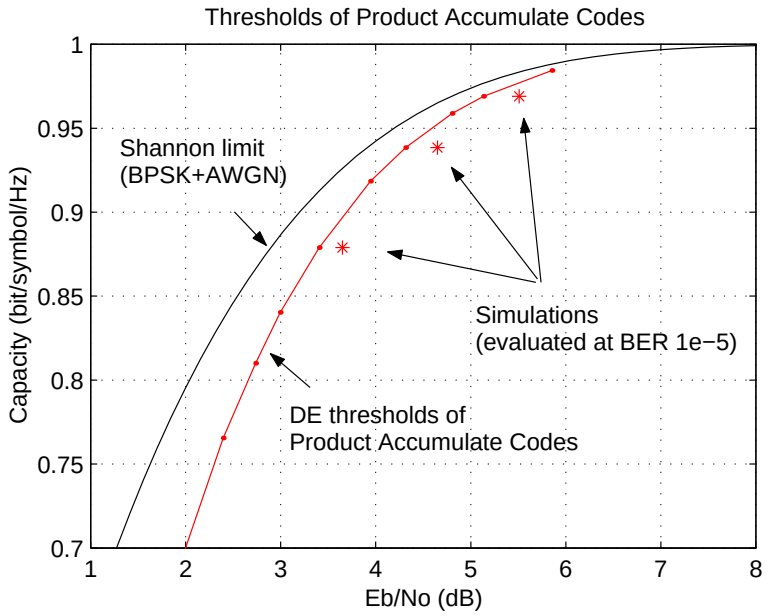


Fig. 15. Thresholds for PA-II codes.

That is, we use interleavers where the interleaving pattern can be generated on the fly without having to store the interleaving pattern. We consider congruential sequence generated according to [53]

$$A_{n+1} = (a \cdot A_n + b) \bmod N. \tag{2.84}$$

To assure that this generates a maximal length sequence from 0 to $N-1$, parameters a and b need to satisfy

1. $a < N$, $b < N$, b be relatively prime to N;
2. (a-1) be a multiple of p, for every prime p dividing N; and
3. particularly, (a-1) be a multiple of 4 if N is a multiple of 4.

It is also desirable, though not essential, that a be relatively prime to N; we consider such an interleaver for both the interleavers in the proposed code. This can also be

considered as an algebraic design of the code graph since the graph structure can be directly specified by the interleaving sequence. Hence, given an N and t, the choice of a and b completely specifies the code graph and, hence, the encoding and decoding operations.

Another direct benefit of using algebraic interleavers is that it allows great flexibility for PA codes to change code rate as well as code length. With LDPC codes, however, it is not easy to change code lengths nor code rates using one encoder/decoder structure. Although LDPC codes can be defined with a bit/check degree profile and a random interleaver, its encoding requires the availability of the generator matrix. In other words, with LDPC codes, for each code rate and code length, not only does the code structure (connections between bits and checks) need to be devised specifically, but the generator matrix needs to be stored individually. Although possible, it requires special treatment to accommodate several rates/block sizes in one LDPC encoder/decoder pair.

F. Generalized Product Accumulate Codes

1. Motivation

Product accumulate codes as discussed so far are a class of promising *high-rate* codes whose code rates are limited to $R \geq 1/2$. Although high rates are desirable for bandwidth efficiency, some applications require relatively low rates (for stronger error protection) and/or flexible rate adaptivity. In this section, we extend product accumulate codes to rates below $1/2$, which lead to *generalized product accumulate* codes or GPA codes.

Through similar analysis techniques from both the ML perspective and the iterative perspective as we did for PA codes, we show that the same nice properties of PA

codes also exist for GPA codes. The new aspect of GPA codes is that their richness in code rate and code structure make it possible for several different constructions to achieve one given code rate (and code length). By computing the thresholds using density evolution for different constructions we not only illustrate the asymptotic performance difference between ML decoding and iterative decoding, but also indicate guidelines for choosing the best construction.

2. Structure of GPA Codes

The outer code of product accumulate codes (PA-I) has two parallel branches of single-parity check codes. In general, the number of parallel branches can be more than 2. As shown in Figure 16, $M \geq 2$ branches of $(t+1, t)$ SPC codes can be interleaved and parallelly concatenated as the outer code. We call it *generalized product accumulate* code or GPA code. From the analysis of PA-I codes in [24] [54], in order to obtain interleaving gain, q blocks of SPC codewords need to be combined and jointly interleaved in each branch, since the block error rate is inversely proportional to P rather than the interleaver size as is typically expected. This will be shown to also hold for GPA codes. The resulting GPA code thus has rate $R = t/(t+M)$ and length $N = P(t+M)$. It is interesting to note that in an extreme case when SPC codes are reduced to (2,1) repetition codes, then the corresponding GPA codes are reduced to regular repeat accumulate (RA) codes [21], which, despite their simplicity, have demonstrated surprisingly good performance and are shown to have the potential for achieving AWGN channel capacity [55]. This also holds for GPA codes. RA codes achieve good performance primarily at very low rates; however, GPA codes are capable of good performance for a wide rate range, like $R = 1/3, 1/2, 2/3$. The capability to provide good performance for a wide range of rates using one simple structure is very useful in a variety of practical applications.

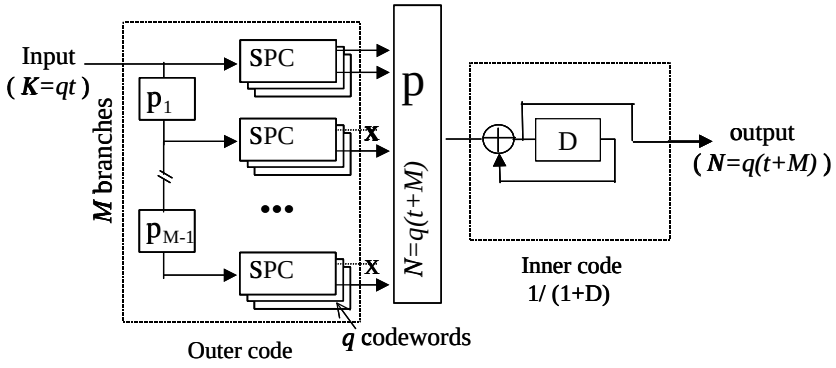


Fig. 16. System model of GPA codes.

The decoding of GPA codes is essentially the same as in PA codes, that is, an iterative decoding employing the turbo principle (message passing decoding) is used where soft information in log-likelihood ratio (LLR) form iterates among different component codes. Specifically, the sum-product and min-sum algorithm [24], which is described in a previous section to decode product accumulate codes, can be readily applied to GPA codes. Hence, GPA codes are also linear-time encodable and decodable.

3. ML-based Analysis for GPA Codes

a. Weight Distribution and Interleaving Gain

The weight distribution of GPA codes can be computed much in the same way as that of PA codes (see (2.56)), except that the outer code now has $M \geq 2$ multiple branches. Instead of (2.49), we have the average IOWE for the M-branch parallel concatenation as

$$A_{w,h}^{(o)} = \sum_{h_1+h_2+\cdots+h_M=h} A_{w,h_1}^{(1)} P_{w,h_2}^{(2)} P_{w,h_3}^{(3)} \cdots P_{w,h_M}^{(M)}, \qquad \forall M \geq 2, \tag{2.85}$$

where $A_{u,v}^{(1)} = A_{u,v}^{SPC}$ as defined in (2.51), and $A_{u,v}^{(2)} = A_{u,v}^{(3)} = \cdots = A_{u,v}^{(M)} = A_{u,v+u}^{(1)}$.

The case of $M = 2$ was discussed in Section D. For $M \geq 3$, we obtain the number of weight-2 outer codewords as

$$A_{h=2}^{(o)} = \sum_{w} A_{w,h=2}^{(o)}, \tag{2.86}$$

$$= P\binom{t}{2}\left(\frac{P\binom{t}{2}}{\binom{Pt}{2}}\right)^{M-1}, \tag{2.87}$$

$$= O\left(\frac{(t-1)^2}{2}P^{-(M-2)}\right), \quad \forall M \geq 2, \tag{2.88}$$

which suggests that the number of weight-2 outer codewords decreases at some power of P when $M \geq 3$. In other words, when there are at least 3 SPC branches, weight-2 outer codewords vanishes with the increase of block size and, hence, an interleaving gain exists. Substituting (2.85) and (2.57) in (2.56), the weight distribution of GPA codes for $M \geq 3$ can be computed straightforwardly.

Equation (2.88) indicates that, the number of outer weight-2 codewords vanishes (asymptotically) when GPA codes have $M \geq 3$. Similar behavior is observed for other low weight outer codewords. For example, the number of weight-3 outer-codewords, $A_{h=3}^{(o),M} = O(Pt)$, for $M = 2$, and $A_{h=3}^{(o),M} = 0, \forall M \geq 3$. The number of weight-4 outer codewords, $A_{h=4}^{(o),M} = O(\frac{3(t-1)^4}{2}(\frac{3}{2P})^{M-2}), \forall M \geq 2$. It is interesting to point out that the observation that the number of low weight codewords of a parallel concatenation with M branches of SPC codes does not decrease like any power of the block size for $M = 2$, seems to have some bearing with Kahale and Urbanke's result that the minimum distance of a parallel concatenated code with M parallel branches grows with the increase of block size only when $M \geq 3$ [56]. However, the case they studied was for recursive convolutional component codes and they were able to give a much stronger result that the growth rate of the minimum distance therein is $N^{\frac{M-2}{M}}$ [56]. For the $1/(1 \oplus D)$ code, the output weight is at least half of the input weight (the worst

case happens when input weights are paired in such a way that the two weights in each pair are consecutive). Since high-weight outer codewords (pM-codewords) tend to produce high-weight GPA codewords, we can thus draw the following conjecture from (2.54): GPA codes with a large number of SPC branches in the outer code have better weight spectrum than codes with a smaller number of branches. In other words, if ML decoding were used, GPA codes with larger M are expected to outperform those with smaller M at the same rate. This is confirmed by the ML-based bounds computed immediately below.

b. ML Decoding Based bounds

Similar to the treatment of PA codes, we quantify the asymptotic performance of GPA codes with ML decoding by examining several upper bounds, and in particular the union bounds and the simple bounds.

Given the weight distribution of GPA codes as discussed above, the spectral shape can be conveniently obtained for GPA codes, which is a function of all the parameters of the code, including P, t, M. Figure 17 compares the spectral shape of rate 1/2 GPA codes with $M = 2, 4, 8$ and $N = 400$, respectively. As expected, for GPA codes of the same rate, larger M leads to better spectral shape (and therefore better code in the ML sense).

The union bound and the simple bound for GPA codes are computed and plotted in Figure 18, together with the Shannon limit and the bounds for random codes and RA codes. For clarity, we have only plotted the curves for GPA codes with $M = 2, 4$. Similarly to what we have observed with PA codes, we see that the simple bounds are very close to those of the random codes (i.e., GPA codes are good) and that the higher the code rate the closer the bounds (i.e., GPA codes are best at high rates). Further, from the plot, we see that the larger the value of M, the closer the bound, which

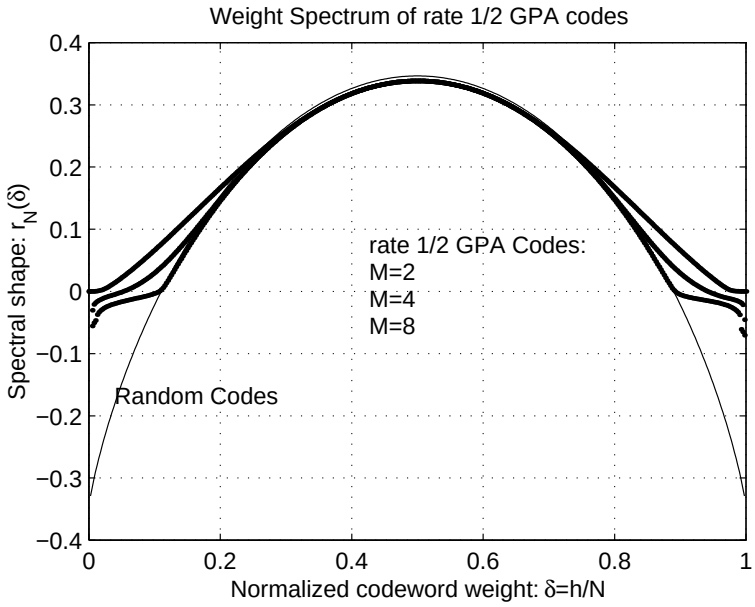


Fig. 17. Spectral shape of GPA codes.

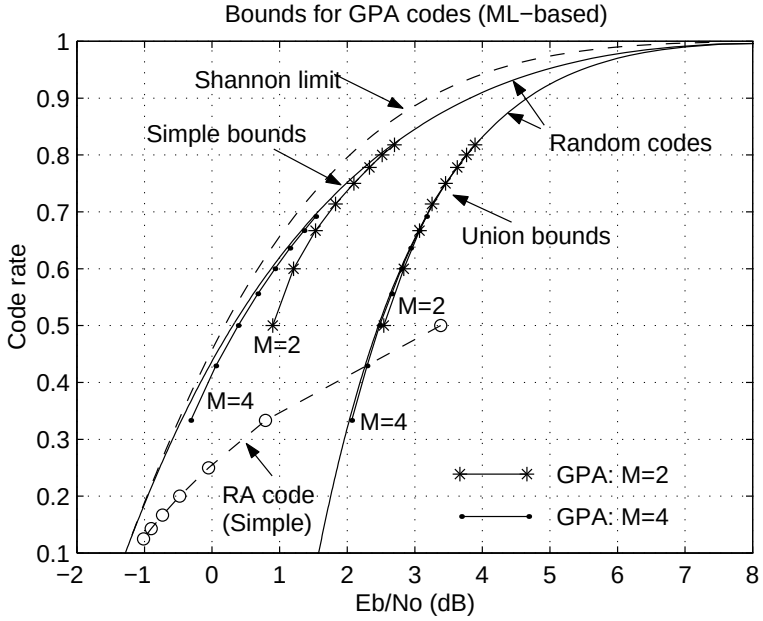


Fig. 18. The simple bound and the union bound.

matches with our analysis in the previous subsection. It is expected that as $M \to \infty$, the simple bound as well as the distance spectrum of GPA codes will converge to those of the random codes. In other words, like repeat accumulate codes, GPA codes also have the potential for achieving AWGN channel capacity such that, as the rate approaches 0, the average required E_b/N_0 for arbitrarily small error probability with ML decoding approaches $\log 2$. This is obvious, for, as mentioned above, RA codes are the special case of GPA codes where all the (t+1,t) SPC codes have parameter $t = 1$. Further, it can be seen from the plot that, for an RA code of a given rate, we can always find a better GPA code (in the ML sense) of the same rate by increasing both t and M (recall that GPA codes have rate $R = t/(t + M)$).

The implication of the above analysis is that GPA codes are by nature good codes, and that larger M (i.e., more SPC branches) leads to better distance spectrum. However, due to the lack of a practical ML decoder, this behavior may not be observable in practice. Since the performance of the suboptimal iterative decoder is also a function of M, the analysis from the iterative perspective as shown below actually reveals just the opposite.

4. Analysis of GPA Codes Using Density Evolution

Similar to PA codes, GPA code can be represented using a bipartite graph of *bits* and *checks*, such that density evolution can be performed "analytically" (as opposed to the Monte Carlo treatment of convolutional codes and turbo codes). The detailed steps for PA codes are described in Section C and in [24] and can be extended straightforwardly to GPA codes. For simplicity, instead of using the exactly discretized density evolution (without any approximation), we employ a Gaussian approximation in the computation of the thresholds for GPA codes.

Recall that when messages are represented in LLR form, the outgoing message

along an edge is simply the "sum" of the received messages which includes messages from all edges (and the channel if applicable) except the message coming along this very edge. For bit nodes, this "sum" is a regular sum in the real domain, and for check nodes, it is a check sum denoted as $\boxplus$ operation or $tanh$ operation (see (2.5) and (2.6)). Since no simple closed form is available on the relations of pdf's under the nonlinear check operation, discretized density evolution takes a numerical method which can determine the threshold to a desired degree of accuracy. To further simplify the computation, densities can be approximated as a mixture of Gaussian densities, which leads to only a slight decrease in accuracy [32]. It has been shown that for binary-input, output-symmetric memoryless channels and binary linear codes, the distribution f of messages passed in each step satisfies $f(x) = f(-x)e^x$ [14]. This consistency condition, when applied to Gaussian distributions, leads to the constraint that the variance of the message equals twice the mean. That is, the pdf of the messages are approximated as $\mathcal{N}(\mu, 2*\mu)$, where the mean of the message, μ, becomes the one single quantity governing the process. The complexity is then greatly reduced.

Figure 19 plots the thresholds of GPA codes computed using density evolution with Gaussian approximation for $M = 2, 3, 4$, respectively. As can be seen, at rates $R \geq 1/2$, even with iterative approach, GPA codes can perform close to capacity. At low rates, the thresholds are about 1 dB from the capacity. The plot shows that the more the branches, the worse the thresholds (and the more the complexity). That GPA codes with fewer SPC branches can perform better under iterative decoding is just opposite to what is inferred from the weight spectrum analysis. This disagreement indicates that the performance loss due to the suboptimality of the iterative decoder may be quite severe in the presence of several component codes. Hence, in practice, it is desirable to use a small number of SPC branches with more powerful SPC codes in each branch, but at least $M = 2$ branches are needed in order to ob-

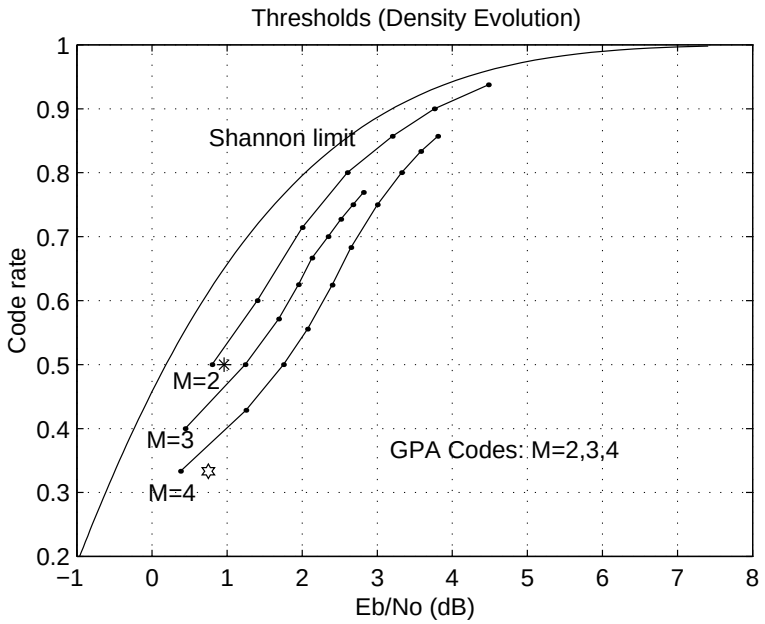


Fig. 19. Iterative thresholds for GPA codes.

tain the interleaving gain [24]. This way, the best performance is achieved with the least complexity. However, with smaller M, the achievable rate range is also smaller ($\frac{2}{2+M} \leq R < 1$).

The codes are hence best suited for practical applications which involve changing the rate of the code constantly, where high rates ($R > 1/2$) are used most of the time, whereas occasionally due to poor channel conditions, lower rates are required. The regular structure of these codes makes it easy to change the rate at both the transmitter and the receiver.

G. Simulation Results of PA and GPA Codes

1. Performance of PA Codes

We first investigate the performance of PA-I codes at medium rates. Figure 20 shows the performance of a rate-1/2 PA-I code of data block size $64K$, $4K$ and $1K$, respec-

tively. As can be seen, the larger the block size, the steeper the performance curve, which clearly depicts the interleaving gain phenomenon. For comparison purposes, the performance of a (2K,1K) turbo code from [22] and the most recently reported irregular repeat accumulate (IRA) codes [22] of the same parameter are also shown. As can be seen, (2K, 1K) PA-I codes perform as well as turbo codes at BER of 10^{-5}, yet without error floors. From Table II, we can see that the decoding complexity of rate-1/2 PA codes with 30 iterations is approximately 1/16 that of a 16-state turbo code with 8 iterations. It is also important to note that the complexity savings are higher as the rate increases, since the decoding complexity of punctured turbo codes does not reduce with increasing rate; whereas the decoding complexity of PA codes is inversely proportional to the rate. It should also be noted that the curve of PA-I codes is somewhat steeper than that of turbo codes or irregular repeat accumulate codes, and therefore may outperform them at lower BERs.

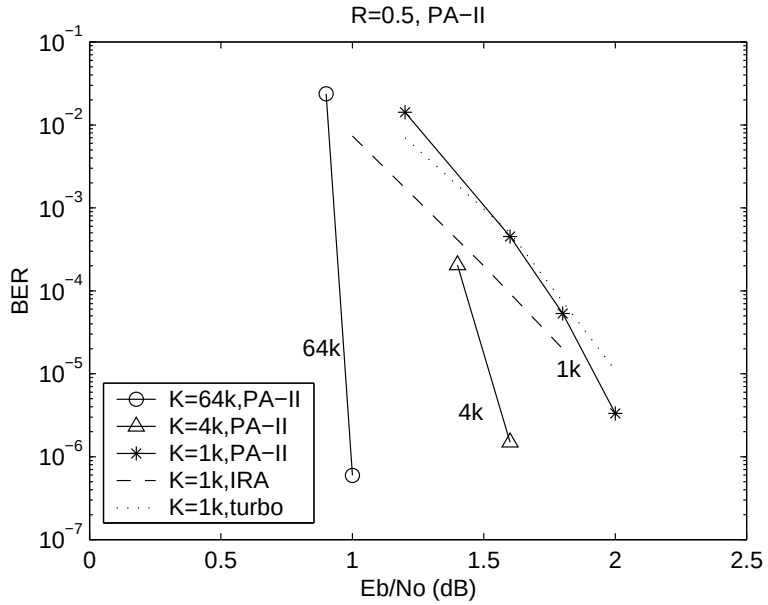


Fig. 20. Performance of PA-I codes at code rate $R = 1/2$. (Data block size is 1K, 4K and 64K, respectively.)

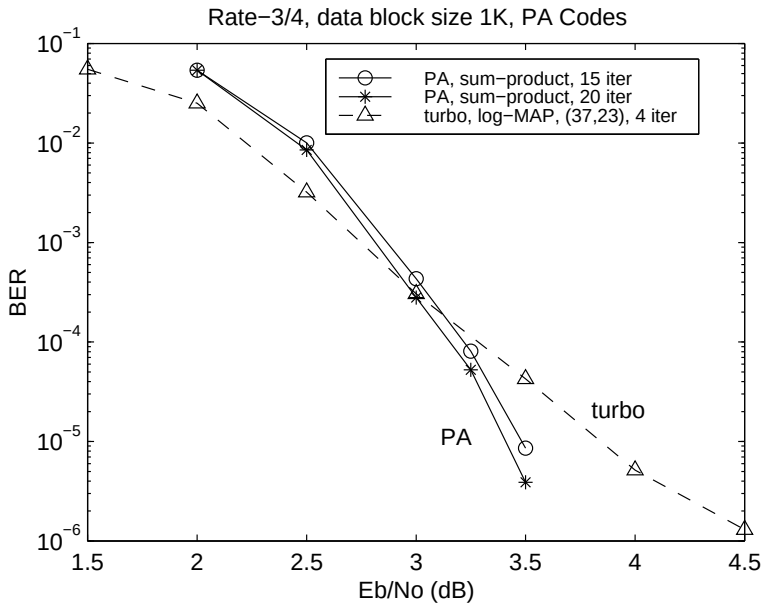


Fig. 21. Performance of PA-I codes at code rate $R = 3/4$.

As indicated by both ML-based and iterative-based analysis, product accumulate codes are most advantageous at high rates. Figure 21 compares the performance of a rate-3/4 PA-I code at 15_{th} and 20_{th} iteration with a 16-state turbo code of polynomials $(37, 23)_o$ at the 4_{th} iteration. The data block size is $K = 1002$ for both codes. Clearly, while the PA-I code is comparable to the turbo code (Figure 20) at rate-1/2, it significantly outperforms turbo codes at rate-3/4 (much steeper curves and no error floor). Further, the PA-I code at the 15_{th} and 20_{th} iteration requires only about 23% and 30% the complexity of the turbo code at the 4_{th} iteration, respectively. Hence, PA codes are expected to find promising applications at high-rates with the advantages of low-complexity, high-performance and no error floor.

The BER performance of PA-II codes at high rates is shown in Figure 22. The codes simulated have rates 0.88, 0.94 and 0.97, which are formed from $(16, 15)^2$, $(32, 31)^2$ and $(64, 63)^2$ outer 2-D TPC/SPC codes, respectively. Since interleaving gain is directly proportional to the number of TPC/SPC blocks in a codeword, sev-

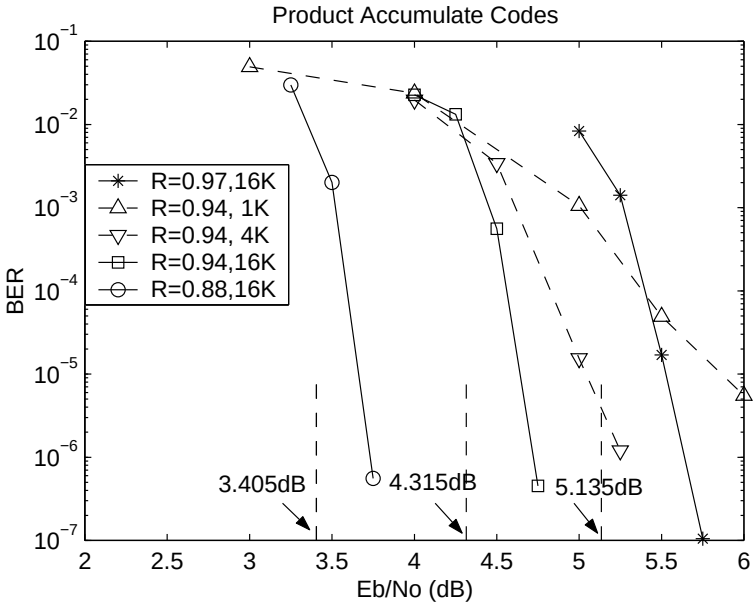


Fig. 22. Performance of PA-II codes at high rates.

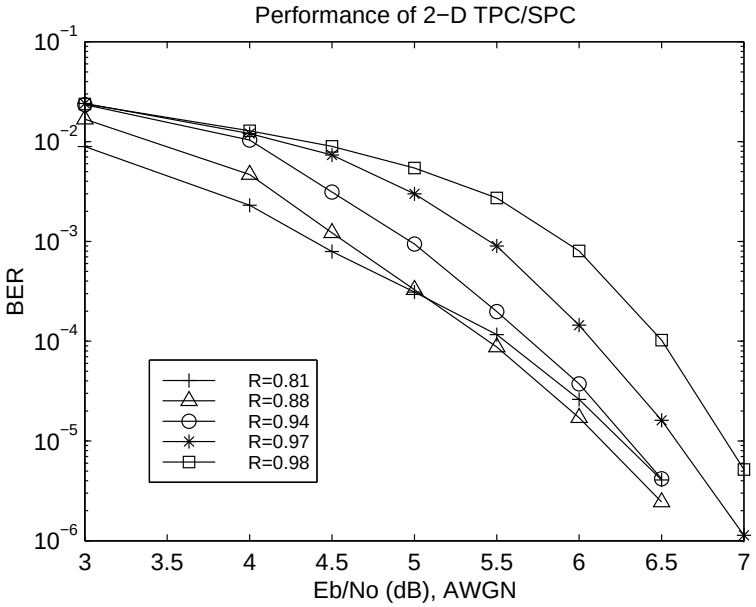


Fig. 23. Performance of plain TPC/SPC codes over AWGN channels.

eral TPC/SPC blocks may be combined to achieve a large effective block size when needed. Corresponding threshold bounds calculated by density evolution are also shown. Two things can be immediately seen from the plot. (1) Product accumulate codes demonstrate a significant performance improvement compared to plain TPC/SPC codes (Figure 23). A 1 dB gain is achieved for rate-0.97 codes, while as much as a 3 dB gain is achieved for rate-0.88 codes. (2) With a data block size of $K = 16K$, the performance of PA-II codes is as close as within 0.3 dB from the capacity bound at BER of 10^{-5}, showing a very good fit. All curves shown are after 15 turbo iterations. Although not plotted here, a reduction from 15 to 8 iterations incurs only about 0.1 dB loss.

Performance comparison between PA-I codes with PA-II codes is shown in Figure 24. As expected, PA-I codes tend to perform better than PA-II codes at higher rates also, but the difference is relatively small, since PA-II codes are already close to the Shannon limit at high rates. The codes simulated are of rate 0.94. As can be seen, the difference seems more noticeable at a moderate size ($16K$) than either very short block size ($1K$) where the random interleaver is too short to play a significant role or very long block size ($64K$) where the performance is already close to the threshold.

We also compare the performances of PA codes using different encoding/decoding strategies, including min-sum decoding vs (serial) sum-product decoding, S-random interleaver vs algebraic interleaver, and parallel vs serial sum-product decoding. Figure 25 compares the performance of a rate 0.5 PA-I code with the sum-product decoding to the low-complexity min-sum decoding. Performance at $5, 10, 15, 20$ iterations is evaluated. At all those iterations, the min-sum decoding incurs only about 0.2 dB loss, while saving more than half the complexity. Hence, the min-sum algorithm provides a good tradeoff between performance and complexity, and is thus very appealing for simple, low-cost systems. Figure 26 compares the performance of PA-I

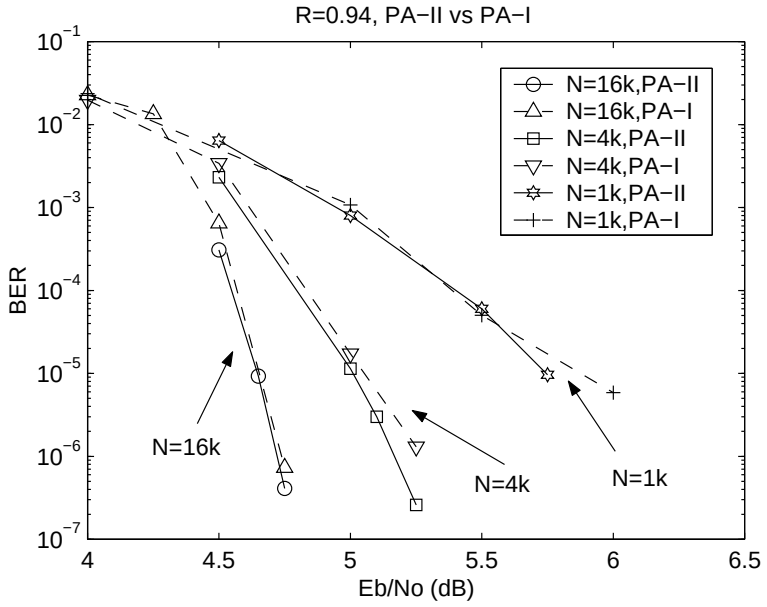


Fig. 24. Performance comparison of PA-I and PA-II codes at high rates.

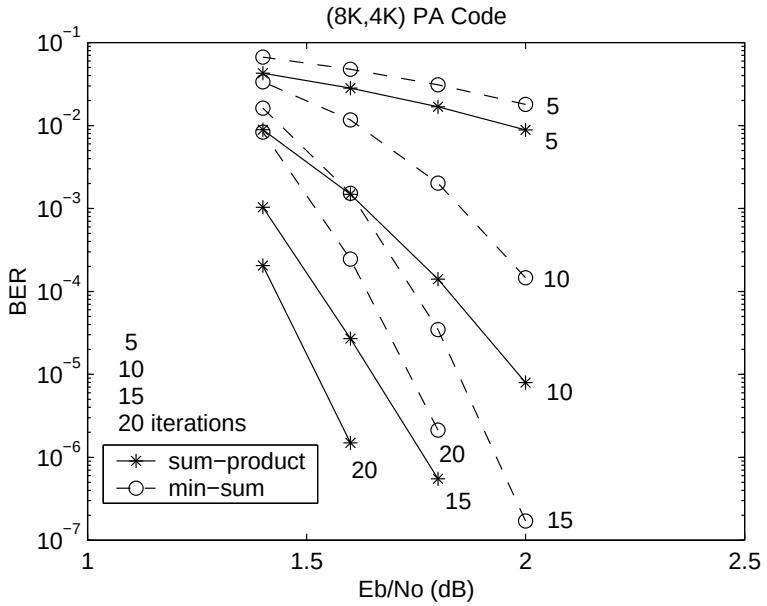


Fig. 25. Sum-product decoding vs min-sum decoding.

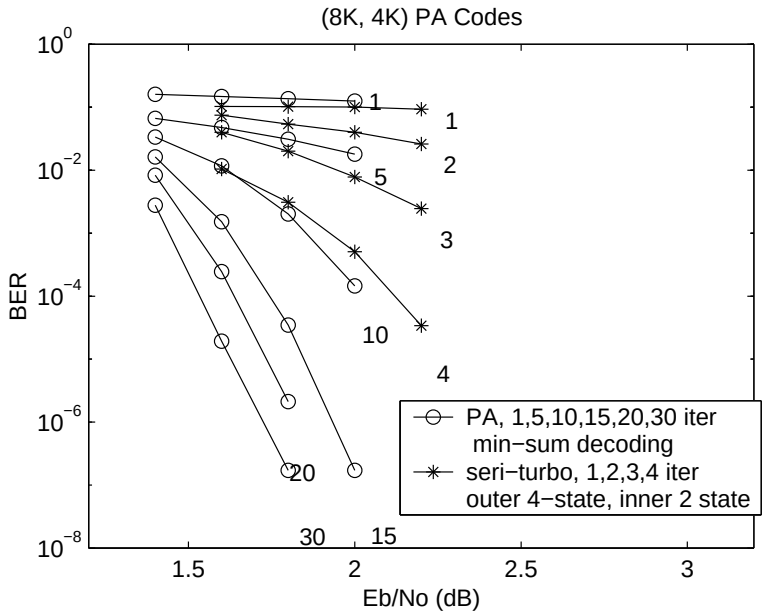


Fig. 26. Min-sum decoding of PA-I codes.

codes using the min-sum decoding to the performance of a serial serial turbo of the same code rate and block size. The serial turbo code is composed of an outer 4-state and an inner 2-state convolutional code. It is interesting to see that even with the low-complexity min-sum decoding, the PA-I code still outperforms the serial turbo code. Comparing the performance of the PA-I code (using min-sum decoding) at the 15_{th} iteration with that of the serial turbo at 4_{th} iteration, we see that a 0.4 dB performance gain is achieved at BER of 10^{-5} with only about 60% of the complexity, which is impressive (see Table II for complexity analysis).

Figure 27 compares the performance of a rate-0.5 PA-I code with S-random interleavers and algebraic interleavers. Interestingly, replacing S-random interleavers with algebraic interleavers results in hardly any performance degradation. Since the length of algebraic interleavers can be conveniently changed, using algebraic interleavers can lend another degree of flexibility to PA codes.

Figure 28 compares the performance of a rate-0.5 PA-I code with serial sum-

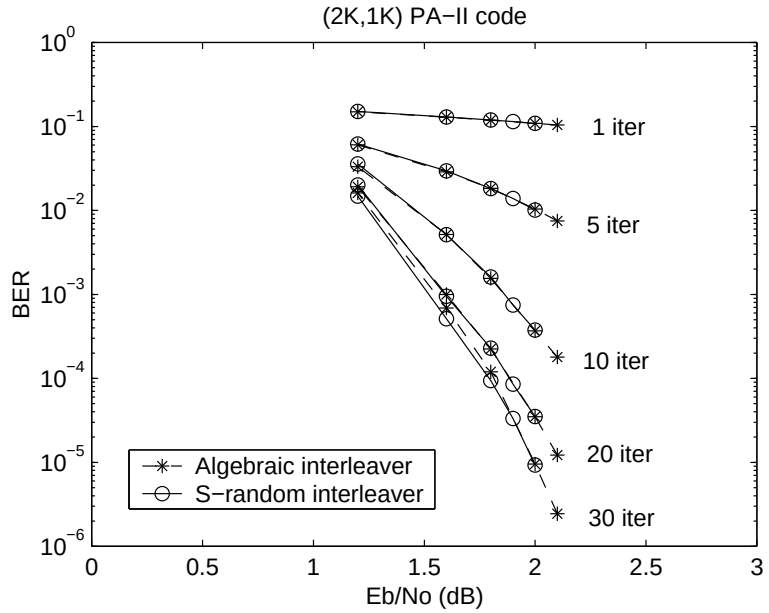


Fig. 27. Algebraic interleaver vs S-random interleaver.

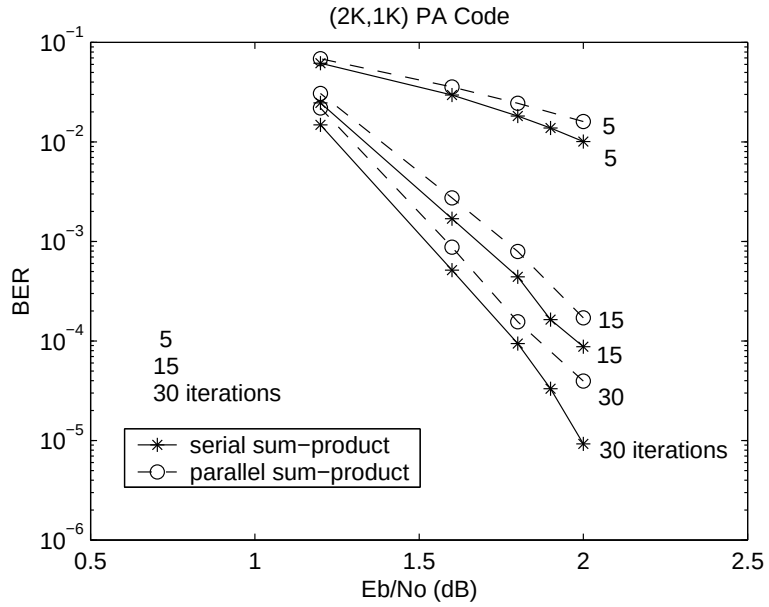


Fig. 28. Serial sum-product decoding vs parallel sum-product decoding.

product decoding and parallel sum-product decoding. We see that the difference between the two approaches is about 0.1 dB at a bit error rate of 10^{-5}. Hence, parallel sum-product decoding serves as a good candidate for hardware implementation.

2. Performance of GPA Codes

Figure 29 shows the BER curves of a rate 1/2 GPA code with 2 branches of (3,2) SPC codes and a rate 1/3 GPA code with 4 branches of (3,2) SPC codes, both employing the sum-product decoding. Interleaving gain is obvious from the plot, and the performance is only 0.8 and 1.1 dB away from the Shannon limit, respectively.

The effect of the number of SPC branches on the performance of GPA codes is investigated in Figure 30. The (2000,1000) GPA code simulated has two different settings: 2 branches of (3,2) SPC codes with 500 SPC codewords combined in each branch, and 4 branches of (5,4) SPC codes with 250 codewords combined in each branch. We see that 2-branch GPA codes outperform 4-branch GPA codes in addition to the saving of about 1/3 of the complexity, which confirms the results from the iterative analysis. For comparison purpose, also shown is the performance of the turbo code of the same parameter [22]. Clearly, the GPA code with $M = 2$ is as good as turbo codes (yet with lower complexity). Further, there are no observable error floor due to the serial concatenation with a $1/(1 \oplus D)$ inner code.

H. Comparison to Other Related Codes

Graphical representation of codes has shed great insight into the understanding of many codes [30] [31] [47] [48], including turbo codes, LDPC codes and (irregular) repeat accumulate codes [22] [21]. This section revisits PA codes from the graph perspective for a comparison and unification of PA codes and other capacity-approaching

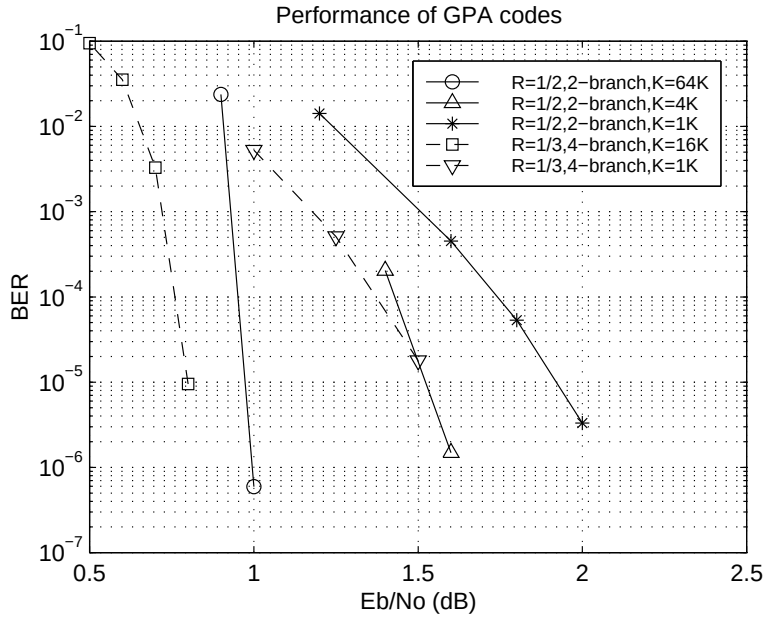


Fig. 29. Performance of GPA codes with code rate R=1/3 and 1/2.

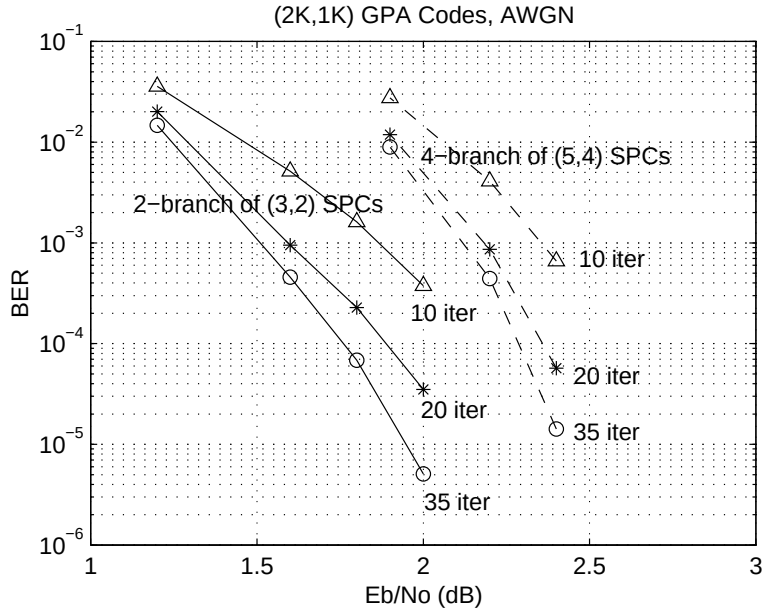


Fig. 30. Comparison of 2-branch and 4-branch GPA Codes

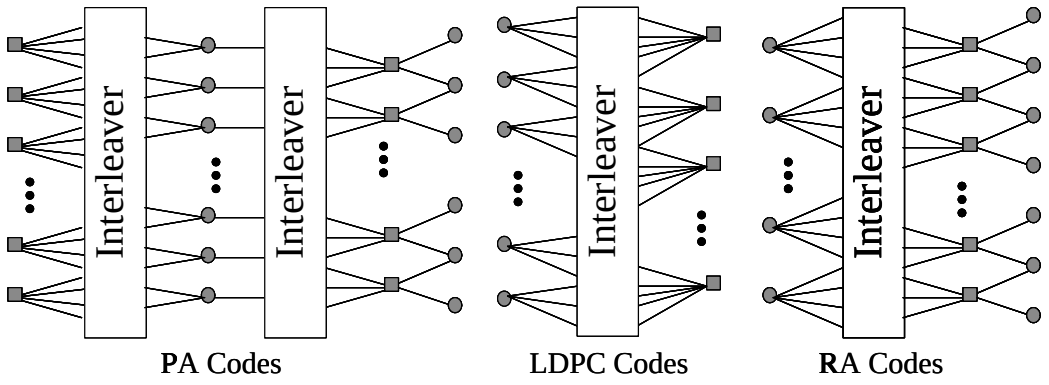


Fig. 31. Comparison of several codes using Tanner graph. (A) PA codes. (B) LDPC codes. (C) RA codes.

codes.

The Tanner graph structure shown in Figure 31(A) reveals that PA codes are essentially LDPC codes with two levels of checks instead of one as in conventional LDPC codes (see Figure 31(B) where small circles denote bits and small boxes denote checks). However, the encoding complexity of PA codes is linear and the encoder is easy to implement since it does not require explicit storage of a generator matrix.

Repeat accumulate codes [21], and their improved version, irregular repeat accumulate codes [22], are a class of very nice codes which are linear time encodable and provide near capacity performance. A careful study of the code graph shows an intrinsic connection between the structure of the proposed PA codes and RA/IRA codes, although our initial motivation for PA codes is from the encouraging performance of TPC/SPC codes over partial response channels in magnetic recording systems [27]. Figures 31(A) and (C) presents the Tanner graphs of the proposed PA and RA/IRA codes. One difference is that a PA code is non-systematic whereas the systematic bits are transmitted explicitly in RA/IRA codes. One major advantage of PA codes is its regular code structure which facilitates implementation and further, the rate of

the code can be easily changed at the transmitter and receiver, since the structure of the codes is identical for all rates. For IRA codes, a different irregularity pattern and associated graphs have to be designed and stored. Similarly, the length of PA codes can be easily changed if algebraic interleavers are used since the structure remains unchanged with length.

Recently, concatenated tree (CT) codes were shown to be a good lower complexity alternative to turbo codes in [57]. Simulation results for rates $1/2, 0.7$ and 0.88 show that the BER performance and decoding complexity of PA codes are similar to those of concatenated tree codes. In fact, the decoding complexity seems to be slightly lower for PA codes. Again, the advantage is in the ease of rate change at the transmitter and receiver. It should also be noted that the performance of PA codes is on par with finite-geometry LDPC based codes proposed by Shou, Lin and Fossorier [34].

I. Summary

A class of interleaved serially concatenated codes called *product accumulate* codes has been constructed and shown to possess good BER performance, linear encoding/decoding complexity as well as an efficient soft-decoding algorithm. The proposed codes consist of an outer 2-D TPC/SPC code, a random interleaver, and a rate-1 recursive convolutional inner code of the form $1/(1 \oplus D)$. The main advantages of the proposed codes are very low decoding complexity compared to turbo codes especially for high rates, good BER performance and ease of implementation. Through analysis and simulations we show this class of proposed codes perform well for almost all rates $R \geq 1/2$ and for long and short block sizes alike.

Generalized product accumulate codes are investigated and shown to be provably "good" both in the ML sense and in the iterative sense. They have low complexity and

the performance using sum-product decoding is close to the capacity over the entire code range. Bounds and thresholds are examined from both the ML perspective and the iterative perspective. Although the ML analysis indicates that more SPC branches lead to better distance spectrum, iterative analysis as well as simulation results reveal that, whenever possible, the number of SPC branches should be kept small (but should be at least 2). This is the best choice in terms of both performance and complexity. Many good features about PA codes as proposed in [23] [24] can be conveniently adopted for GPA codes, like algebraic interleaving, which will make it flexible for GPA codes to change rate and length adaptively. The regular structure of GPA codes makes it appealing for hardware implementation particularly in adaptive rate coding. GPA codes can conveniently adopt to the rate change by reducing the rate of the SPC code and/or increasing the number of parallel branches.

As indicated by the research on irregular LDPC codes and RA/IRA codes, irregularity seems to be the key for a further improvement in performance. Irregularity offers unequal error protection where highly protected bits tend to be decoded first and then help with the less protected bits. With irregular GPA codes, input bits will not uniformly participate in every SPC branch. Rather, the number of SPC branches (checks) each bit is involved in, will follow a carefully-designed profile. It is interesting to point out that irregular GPA codes thereby become irregular repeat accumulate codes [22].

CHAPTER III

PRODUCT ACCUMULATE CODES FOR WIRELESS COMMUNICATION

A. Introduction

In this chapter, we investigate the performance of product accumulate codes on flat Rayleigh fading channels with both coherent and noncoherent detection. The motivation is two-fold. First, the work of Chapter II has established them as a class of low-complexity, capacity-approaching good codes on additive white Gaussian noise channels [24]. Second, PA codes are inherently differentially coded which permits simple noncoherent differential detection (that is desirable for wireless applications especially in fast fading environment).

The performance of PA codes using coherent binary phase shift keying signaling on independent and correlated Rayleigh channels either with or without channel state information (CSI) is first investigated[1]. Divsalar's simple bounds are computed to mark the performance of PA codes with finite code length [58], and iterative thresholds using density evolution are computed to mark the performance of PA codes with infinite code length. Comparing the asymptotic threshold of PA codes with that of LDPC codes, we see that PA codes about 0.6 to 0.7 dB better than the regular LDPC codes but are about 0.5 dB worse than optimal irregular LDPC codes which have maximum left degree of 50 (with The resulting optimal LDPC code shows a 1.04 dB gain over the existing PA code. coherent detection). Simulations of fairly long block lengths show a good agreement with the analytical results.

One big motivation and advantage of considering PA codes for fading channels

[1]By "coherent", we mean the phases of the signal samples are perfectly estimated. "Perfect CSI" means that fading amplitudes are also known to the receiver, and "no CSI" means that fading amplitudes are unknown.

is that they are inherently *differentially coded*. Coherent phase shift keying, although a popular modulation scheme, requires coherent detection which may not always be convenient or feasible, due to several practical issues like complexity, acquisition time, sensitivity to tracking errors and phase ambiguity. The nice thing about differential encoding is that it admits simple differential detection which solves phase ambiguity and which requires only frequency synchronization (often more readily provided than phase synchronization). The nice thing about PA codes is that a differential encoder (the accumulator) is already part of the code structure. Hence, PA codes (and for this purpose all "accumulated" codes such as nonsystematic regular/irregular repeat accumulate codes [22] and convolutional accumulated codes) are intrinsically suitable for use with binary differential encoding and differential detection (binary DPSK or BDPSK).

A conventional differential detector operates on two symbol intervals and recovers the information by subtracting the phase of the previous signal sample from the current signal sample. Although cheap to implement, they could suffer as much as 4 to 5 dB in BER performance [59]. To fill the gap between the conventional differential decoder and the differentially encoded coherent detection, researchers have generally focused on the extension of the observation interval beyond two symbol intervals, which results in block or multiple symbol (differential) detection, like maximum-likelihood multiple symbol detection, trellis-based multiple symbol detection with per-survivor processing and their variations (see, for example, [60] [61] [62] [63] [64] [65] and the references therein). Maximum-likelihood multiple detection [62] assumes that the phase are near constant over the observation interval and, hence, is most suitable for static (or slow time-varying) channels where the performance has shown to asymptotically achieve that of a coherent detection. Trellis-based multiple symbol detection [63] [60], on the other hand, works for both stationary and time-varying

channels. It expands the differential encoder to a super trellis with 2^{N_w} states (N_w being the observation window size). *A posteriori* probability decoding (like the BCJR algorithm) is exploited together with linear prediction and per-survivor process techniques to help estimate the channel. However, for both types of multiple symbol detection, the complexity increases exponentially with the observation window size and, hence, limits its use in practical applications. To preserve the simplicity of PA codes, we propose and discuss a simple iterative differential detection and decoding receiver, which can perform within 1 dB from the coherent detection on fast fading channels with little additional complexity and bandwidth expansion. Different implementation strategies of the IDDD receiver is discussed and the impact of pilot spacing and filter length is investigated. We show that the proposed receiver is robust for different Doppler spreads, and that the amount of pilot insertion plays an important role in determining the overall performance especially on very fast fading channels.

We use extrinsic information transfer charts to facilitate the discussion of a number of interesting issues concerning differential encoding. First, we show that the popular practice of inserting pilot symbols to periodically terminate the differential trellis incurs an intrinsic loss in code capacity[2] due to a "trellis segmentation" effect. Hence, a better way of inserting pilot symbols should be to separate them from the trellis structure. Second, in studying the convergence property of the iterative process, we show that while a (high-rate) PA code yields desirable performance with noncoherent (differential) detection, a general differentially coded LDPC code does not (recall that the outer code of PA codes can be viewed as a special type of LDPC codes with weight-1 and 2 variable nodes only). EXIT analysis reveals that the performance/convergence behavior of a conventional LDPC code does not match with

[2]We use capacity to loosely denote information rate.

an inner differential decoder in an iterative process. Consequently, for noncoherent detection, conventional LDPC codes usually require more pilot symbols to estimate the channel (since they should not be used with differential encoding) than PA codes.

To further our insight into what type of (outer) codes are good for differential encoding and how to optimize them, a "convergence-constraint" design method is proposed which uses Gaussian approximated density evolution on EXIT charts to design and optimize good LDPC ensembles for differential encoding and iterative differential detection and decoding. Unlike the conventional "threshold-constraint" method that targets at the best asymptotic threshold, the convergence-constraint method focuses on the convergence property of the iterative interaction between the inner differential code and the outer LDPC code.

The proposed method provides an efficient means of optimizing the degree profiles matched (in convergence) to a given receiver (provided that the EXIT curve the receiver can be properly evaluated). Empirical results show that the (conventional) LDPC codes that have minimum variable node degree of 2 and that are optimal for nonrecursive modulations are not good for differential encoding (or more generally a recursive inner code/modulation), and that the optimal degree profiles need to contain weight-1 variable nodes also. Furthermore, We observe that at high rates, the degree profile of the outer code of PA codes is near-optimal for differential encoding, yet at medium rates, it can and should be optimized in order to achieve better performance. Finally, examples are provided which show that an optimized differentially coded LDPC code can outperform PA codes by more than 1 dB using noncoherent detection.

The rest of the chapter is organized as follows. Section B presents the system model and a brief overview of PA codes. Section C analyzes the coherent performance of PA codes on fading channels using the simple bound and iterative thresholds. Section D discusses noncoherent iterative differential detection and decoding of PA

codes on correlated fading channels and conducts EXIT analysis. Section E proposes and discusses the convergence-constraint density evolution method for optimizing the degree profiles of outer LDPC codes with inner differential encoders. Simulation results are provided along with the discussion. Finally Section F summarizes the paper.

B. System Model

We consider BPSK signaling ($0 \rightarrow +1, 1 \rightarrow -1$) over flat Rayleigh fading channels. Assuming proper sampling of the outputs from the matched filter, the received discrete-time baseband signal can be written as

$$r_k = \alpha_k e^{j\theta_k} s_k + n_k, \tag{3.1}$$

where s_k is the BPSK modulated signal, n_k is the i.i.d. complex AWGN with zero mean and power spectrum density $\sigma^2 = N_0/2$ in each dimension. The fading amplitude α_k is modeled as a normalized Rayleigh random variable with $E[\alpha_k^2] = 1$ and pdf $p_A(\alpha_k) = 2\alpha_k \exp(-\alpha_k^2)$ for $\alpha_k > 0$, and the fading phase θ_k is assumed to be uniformly distributed over $[0, 2\pi]$. For coherent detection, θ_k is perfectly known to the receiver and for noncoherent detection, θ_k is unknown.

For fully interleaved channels, α_k's and θ_k's are independent. For insufficiently interleaved channels, they are correlated. We use the Jakes' isotropic scattering land mobile Rayleigh channel model to describe the correlated Rayleigh process which has auto-correlation $\mathcal{R}_k = \frac{1}{2}\mathcal{J}_0(2k\pi f_d T_s)$, where $f_d T_s$ is the normalized Doppler spread, and $\mathcal{J}_0(\cdot)$ is the 0_{th} order Bessel function of the first kind.

As discussed in Chapter II, the decoding of PA codes is an iterative process where soft extrinsic information is passed between component codes confirming to

the turbo principle. The inner $1/(1+D)$ code can be decoded using a conventional BCJR algorithm, where the computation of the branch matrix γ should be modified to incorporate the appropriate channel statistics. Alternatively, an efficient message-passing algorithm can be performed on the code graph, which, in addition to the magnitude less of complexity, decouples the code structure (code graph) from the channel [24]. In other words, the same message-passing decoding algorithm works for all channels as long as the initial log likelihood ratio values from the channel are properly computed. For Rayleigh fading channels with perfect CSI, i.e., α_k is known $\forall k$, the initial LLRs from the channel can be computed using

$$L_{ch}^{CSI}(s_k) = \frac{4\alpha_k}{N_0} r_k, \tag{3.2}$$

and for Rayleigh fading channels without CSI

$$L_{ch}^{NCSI}(s_k) = \frac{4E[\alpha_k]}{N_0} r_k, \tag{3.3}$$

where $E[\alpha] = \sqrt{\pi}/2$ is the mean. The rest of the decoding algorithm can be found in Chapter II [24].

C. Coherent Detection

1. Ensemble-Average of Divsalar's Simple Bounds

As mentioned in Chapter II, simple bounds proposed by Divsalar is a tight bound that can overcome the cut-off limitation [41] [58]. By using numerical integration instead of a Chernoff bound and by reducing the number of codewords to be included in the bound, Divsalar showed that the bound was tight when applied to repeat accumulate codes and turbo codes on AWGN channels and independent Rayleigh fading channels with channel state information [41] [58].

In this Section, we apply the simple bounding technique to the analysis of PA codes on independent Rayleigh fading channels. We first quote and summarize the key point of [41] [58] where channel state information is known. A simple extension is then made to include the case where channel state information is unavailable.

a. Independent Rayleigh Channels with CSI

Notice that for Rayleigh fading channels, the decision metric is based on the minimization of the norm

$$|| \mathbf{r} - \gamma \boldsymbol{\alpha} \mathbf{s} ||, \tag{3.4}$$

where $\mathbf{s}$, $\mathbf{r}$ and $\boldsymbol{\alpha}$ are the transmitted signal, received signal and the fading amplitude in vector form, respectively, and γ is the amplitude of the transmitted signal such that $\frac{\gamma^2}{2} = E_s/N_o$.

For a good approximation of the error using the Gallager bounds (2.60), and for computational simplicity, the decision region $\Re$ was chosen as an N-dimensional hyper-sphere centered at $\eta\gamma\boldsymbol{\alpha}\mathbf{s}$ and with radius $\sqrt{N}\mathcal{R}$, where η and $\mathcal{R}$ are the parameters to be optimized.

For independent fading channels with perfect CSI, to compensate for the effect of fading, a linear transformation can be performed on $\gamma\boldsymbol{\alpha}\mathbf{s}$. In particular, it has been shown that a rotation $e^{j\varphi}$ and a rescaling ζ yield a good and analytically feasible solution [58]

$$\Re = \{ \mathbf{r} \mid ||\mathbf{r} - \zeta e^{j\varphi}\gamma\boldsymbol{\alpha}\mathbf{s}||^2 \leq N\mathcal{R}^2 \}. \tag{3.5}$$

Based on this choice of the decision region, and after some algebra, Divsalar has

shown that the error probability of an (N, K, R) code can be upper bounded by [58]

$$P(e) \leq \sum_{h=2}^{2\sqrt{N-K+1}} \min \left\{ e^{-N\mathbb{E}(c,\delta,\rho,\beta,\kappa,\phi)},\ e^{N\boldsymbol{\gamma}_N(\delta)} \frac{1}{\pi} \int_0^{\pi/2} \Big[\frac{\sin^2\theta}{\sin^2\theta + c}\Big]^h d\theta \right\}, \tag{3.6}$$

where

$$\begin{aligned} \mathbb{E}(c,\delta,\rho,\beta,\kappa,\phi) &= \frac{\rho}{2}\log\frac{\beta}{\rho} + \frac{1-\rho}{2}\log\frac{1-\beta}{1-\rho} + \rho\delta\log\Big(1 + c(1-2\kappa\phi)\Big) \\ &\quad -\rho\boldsymbol{\gamma}_N(\delta) + \rho(1-\delta)\log\Big[1 + c(1 - 2\kappa\phi - \frac{(1-\kappa^2)\rho}{\beta})\Big] \\ &\quad +(1-\rho)\log\left[1+c\Big(\frac{1-\rho(1-2\kappa\phi)}{1-\rho} - \frac{(1-\rho(1-\kappa))^2}{(1-\rho)(1-\beta)}\Big)\right], \end{aligned} \tag{3.7}$$

$$c = \frac{\gamma^2}{2} = \frac{E_s}{N_0} = R\frac{E_b}{N_0}, \tag{3.8}$$

$$\delta = \frac{h}{N}, \tag{3.9}$$

$$\boldsymbol{\gamma}(\delta) = \boldsymbol{\gamma}_N(\frac{h}{N}) = \begin{cases} \frac{1}{N}\log(A_h), & \text{for word error rate,} \\ \frac{1}{N}\log(\sum_w \frac{w}{K} A_{w,h}), & \text{for bit error rate.} \end{cases} \tag{3.10}$$

b. Independent Rayleigh Channels without CSI

Another simple and reasonable choice of the decision region (3.4) is an ellipsoid centered at $\eta\gamma\mathbf{s}$, which can be obtained by rescaling each coordinate of $\mathbf{r}$ so as to compensate for the effect of fading:

$$\Re = \{\, \mathbf{r} \mid ||\, \boldsymbol{\alpha}^{-1}\mathbf{r} - \eta\gamma\mathbf{s}\,||^2 \leq N\mathcal{R}^2 \,\}, \tag{3.11}$$

where η and $\mathcal{R}$ need to be optimized. For independent Rayleigh channels without CSI, since accurate information on $\boldsymbol{\alpha}$ is unavailable, we resort to the expectation of the fading coefficient (a similar philosophy as used in the decoding procedure), i.e., a loose approximation $\boldsymbol{\alpha}^{-1} \approx E[\boldsymbol{\alpha}^{-1}] = \frac{1}{0.8862}\mathbf{I}$ is used in (3.11), where $\mathbf{I}$ is an identity matrix.

By replicating the computations described in [41] [58], we obtain the simple

bound for the bit error rate on independent Rayleigh channels without CSI

$$P(e) \leq \sum_{h=2}^{2\sqrt{N-K+1}} \min\left\{ e^{-N\mathbb{E}(c,\delta,\rho)},\, \exp\Big(\frac{h\upsilon N\boldsymbol{\gamma}_N(\delta)}{c}\Big)\Big(1-\frac{2}{\sqrt{1+2/\upsilon}+1}\Big)^h \right\}, \quad (3.12)$$

where

$$\begin{aligned}
\mathbb{E}(c,\delta,\rho) &= c\left(1+\frac{1-\delta}{\delta}\Big(1+\frac{1-\rho}{\rho}e^{-2\boldsymbol{\gamma}_N(\delta)}\Big)\right)^{-1} - \frac{1}{2}\log\left(1-\rho+\rho e^{2\boldsymbol{\gamma}_N(\delta)}\right), & (3.13)\\
\rho &= \left(1+\frac{1-\beta}{\beta}e^{2\boldsymbol{\gamma}_N(\delta)}\right)^{-1}, & (3.14)\\
\beta &= \left\{2c\frac{1-\delta}{\delta}\big(1-e^{-2\boldsymbol{\gamma}_N(\delta)}\big)^{-1} + \Big(\frac{1-\delta}{\delta}\Big)^2\Big[(1+c)^2-1\Big]\right\}^{1/2} \\
&\quad -\Big(1+c\Big)\frac{1-\delta}{\delta}, & (3.15)\\
c &= E^2[\alpha]\frac{\gamma^2}{2} = 0.8862^2 R\frac{E_b}{N_0}, & (3.16)\\
\upsilon &= \sqrt{(\gamma^2/2)^2-1} = \sqrt{(RE_b/N_0)^2-1}, & (3.17)\\
\delta &= \frac{h}{N}, & (3.18)\\
\boldsymbol{\gamma}_N(\delta) &= \boldsymbol{\gamma}_N(\frac{h}{N}) = \begin{cases} \frac{1}{N}\log(A_h), & \text{for word error rate,} \\ \frac{1}{N}\log(\sum_w \frac{w}{K}A_{w,h}), & \text{for bit error rate.} \end{cases} & (3.19)
\end{aligned}$$

We note that this is a simple extension of the bound to the fading case where no CSI is available. It may not be as tight as in the case of AWGN channels, but it is easily computable. It is possible that more sophisticated transformation will lead to tighter bounds but may not render any feasible analytical expression.

c. Evaluation of Simple Bounds for PA Codes

As an example, we examined the distance spectrum of a $(1024, 512)$ PA code and computed Divsalar's bounds on the BER performance of the code over independent Rayleigh fading channels with and without CSI. As shown in Figure 32, the bounds are fairly tight at SNRs below the cutoff rate. We note that we have used the ensemble

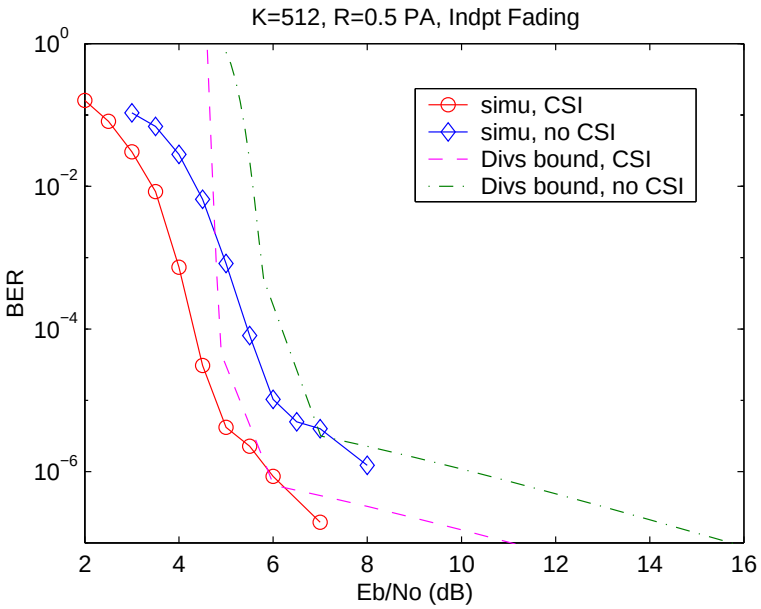


Fig. 32. Divsalar simple bounds for PA codes on independent Rayleigh channels with and without CSI. ($(N, K) = (1024, 512)$, $R = 0.5$.)

average distance spectrum in the computation. Hence, the bounds refer to the average performance and may still not be an accurate indication for the specific PA code that was used in the simulation. However, this is a shortcoming that can hardly be overcome for the analysis of concatenated codes involving random interleavers, since the specific interleaver in use is yet intractable even for moderate lengths.

2. Thresholds Using the Iterative Analysis

a. Computation of the Thresholds for Rayleigh Channels

As mentioned above, performance bounds assuming a maximum likelihood decoder may be optimistic for the existing iterative decoder. To factor in the suboptimality in the decoder, we use density evolution to compute the asymptotic thresholds of PA codes on Rayleigh channels like what we have done with AWGN channels.

As discussed in Chapter II, by examining the probability density function of the

passing messages in each step, the entire decoding process can be literally tracked using density evolution. In general, we are more interested in the thresholds, or the extrinsic information transfer chart, where the former denotes the asymptotic performance limit given infinite code length and infinite number of iterations, and the latter reveals the convergence property.

The computation of the thresholds of PA codes on Rayleigh fading channels is much the same as that on AWGN channels, expect for the initial messages obtained from the channel. We note that a Gaussian approximation can usually be used on AWGN channels to simplify the computation (at the cost of slight loss in accuracy). For fading channels, however, since the initial LLRs from the channel are very different from Gaussian distribution, exact density evolution should be used to preserve accuracy.

Since density evolution evaluates the asymptotic performance of the code with infinite block size and ideal interleaving, the fading parameters are therefore independent. We consider BPSK modulation, the all-zero sequence as the reference sequence and the LLRs as the messages, to compute the initial pdf of the messages from the Rayleigh channel.

Rayleigh Fading Channels with CSI: As discussed in [66], for independent Rayleigh channels, the conditional pdf, $f(L_{ch,y}|\alpha)$, of the initial message $L_{ch,y}$, (which is obtained from the channel and is to be passed from bit to check) follows a Gaussian distribution with mean $4\alpha^2/N_0$ and variance $8\alpha^2/N_0$. That is

$$p(q_0|\alpha) = \frac{1}{4\alpha}\sqrt{\frac{N_0}{\pi}}\exp\left(-\frac{(q_0 - 4\alpha^2/N_0)^2}{16\alpha^2/N_0}\right), \tag{3.20}$$

where α is the normalized Rayleigh fading factor and N_0 is the variance of the complex additive white Gaussian noise as defined before.

Hence, with perfect CSI (α), the initial messages obtained from the fading chan-

nel has the following pdf

$$f_{L_{ch,y}}(q_0) = \int_0^\infty f(L_{ch,y}|\alpha)p(\alpha)d\alpha, \tag{3.21}$$

$$= \sqrt{\frac{N_0}{4\pi}}\exp\left(-\frac{q_0(\sqrt{N_0+1}-1)}{2}\right)\int_0^\infty \exp\left(-\frac{\left(\frac{q_0N_0}{4\alpha}-\alpha\sqrt{N_0+1}\right)^2}{N_0}\right)d\alpha. \tag{3.22}$$

Using integral from [67], (3.22) can be further simplified to

$$f^{CSI}_{L_{ch,y}}(q_0) = \frac{N_0}{4\sqrt{1+N_0}}\cdot\exp\left(\frac{q_0-|q_0|\sqrt{1+N_0}}{2}\right). \tag{3.23}$$

Rayleigh Fading Channels without CSI: Similarly, for Rayleigh fading channels without CSI, with the assumption that the received signals are Gaussian distributed in the most probable region, we can approximate the conditional pdf of the initial message as a Gaussian distribution with mean $4(E[\alpha])^2/N_0$ and variance $8(E[\alpha])^2/N_0$, where $E[\alpha]\approx 0.8862$. The pdf of the initial messages can thus be derived as [66]

$$f_{L_{ch,y}}(q_0) = \frac{\Delta^2\sqrt{N_0}\lambda^{N_0}}{\sqrt{8}E[\alpha]}\left(\sqrt{\frac{2}{\pi}}\lambda + \frac{\Delta q_0}{E[\alpha]}Q\Big(-\frac{\Delta q_0}{2E[\alpha]}\Big)\right), \tag{3.24}$$

where $\Delta = \sqrt{\frac{N_0}{2(N_0+1)}}$, $\lambda = \exp\left(-\frac{\Delta^2 q_0^2}{8(E[\alpha])^2}\right)$, and $Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-z^2/2}dz$.

To track the evolution of the pdf's within the message-passing decoder, either a Monte Carlo simulation can be used, or, more accurately and more efficiently, an analytical form can be derived for the discretized density evolution on PA codes. This has been discussed in detail in Chapter II and summarized in Appendix B. By substituting (3.23) for perfect CSI case or (3.24) for no CSI case in (B.5) and (B.6) in Appendix B, the thresholds of PA codes on Rayleigh channels can then be computed through (B.4) to (B.13) (see Appendix B).

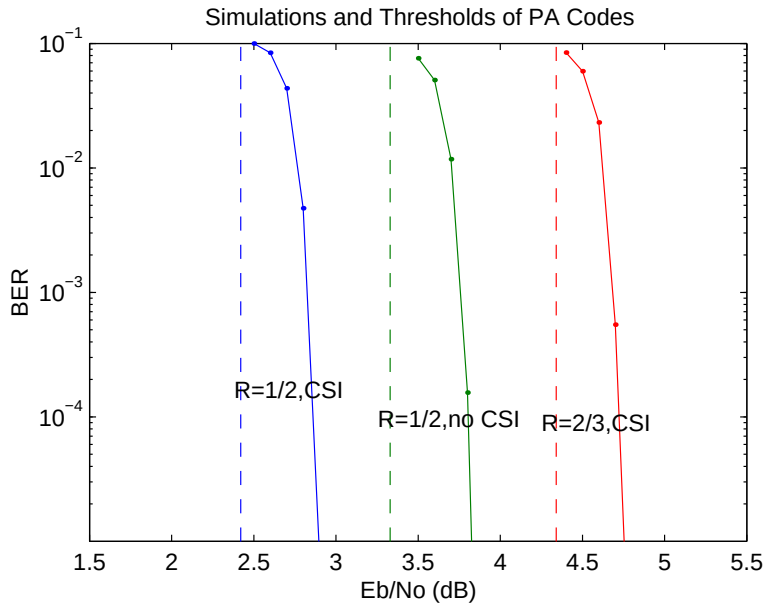


Fig. 33. DE thresholds and simulations of PA codes on independent Rayleigh fading channels. (Data block size $K = 64$K.)

b. Evaluation of the Iterative Thresholds

The thresholds computed using the density evolution technique discussed above indicates the asymptotic performance of iterative decoding of PA codes on Rayleigh fading channels. Figure 33 compares the thresholds and the simulation results for independent Rayleigh channels with and without CSI. As can be seen, the analytical results are consistent with the simulation results of fairly large block sizes (simulations are evaluated at 50_{th} iteration). We conjecture that as the block size and the number of iterations go to infinity, the simulation results will converge to the thresholds.

In Table III we list the thresholds of PA codes of several code rates computed using density evolution. We see that the thresholds are about 0.6 dB from the Shannon limit, and simulations of fairly large block sizes are about 0.3 to 0.4 dB from the thresholds, which is a reasonably good fit. Thresholds of rate 1/2 LDPC codes on Rayleigh channels were reported in [66]. Rate 1/2 PA codes are about 0.6 to

Table III. Thresholds of PA Codes and (3,ρ)-regular LDPC codes on Rayleigh Channels (E_b/N_0 in dB). ((3,ρ) LDPC data by courtesy of Hou, Siegel and Milstein.)

	flat Rayleigh with CSI (dB)			flat Rayleigh without CSI (dB)		
Rate	Capacity	PA	LDPC	Capacity	PA	LDPC
0.5	1.8	2.42	3.06	2.6	3.33	4.06
0.6	3.0	3.56	-	3.8	4.48	-
2/3	3.7	4.24	4.72	4.4	5.15	5.74

0.7 dB better (asymptotically) than regular LDPC codes with column weight 3, but are about 0.5 dB worse (asymptotically) than irregular LDPC codes with maximal column weight of 50.

3. Simulation Results for Coherent Detection

This subsection provides simulation results to benchmark the performance of coherently detected PA codes on Rayleigh fading channels. In each (global) turbo iteration (i.e., iteration between the inner differential encoder and the outer code), two (local) iterations of the outer decoding are performed. This scheduling is found to be a good tradeoff between complexity and performance (with coherent detection).

We first investigate the performance using coherent BPSK on independent Rayleigh fading channels. Figure 34 and 35 show the performances of rate 1/2 PA codes on independent Rayleigh fading channels with and without channel state information, respectively. Bit error rates after 20, 30 and 50 (global) iterations are plotted, and data block sizes of 512, 1K, 4K, and 64K are evaluated to demonstrate the interleaving gain. For comparison purpose, the corresponding channel capacities are also shown

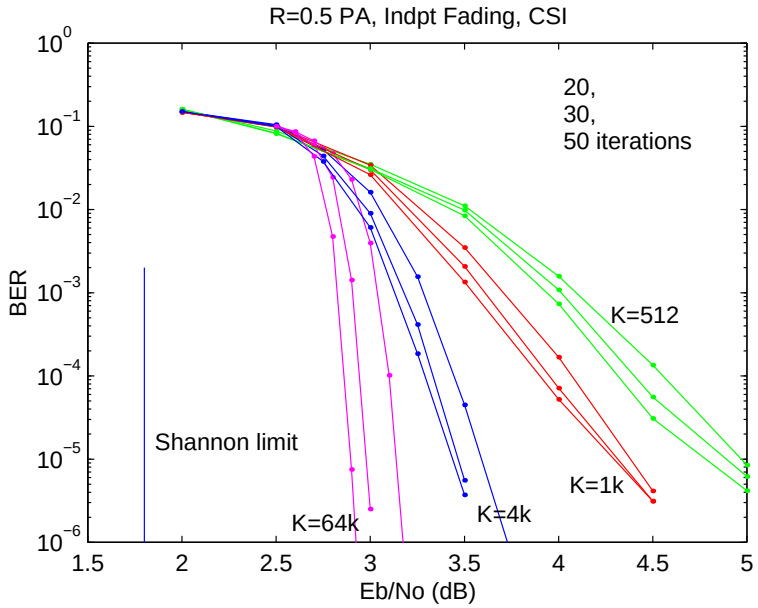


Fig. 34. Performance of PA codes on independent Rayleigh fading channels with CSI. (Rate 0.5, data block size 512, 1K, 4K, 64K.)

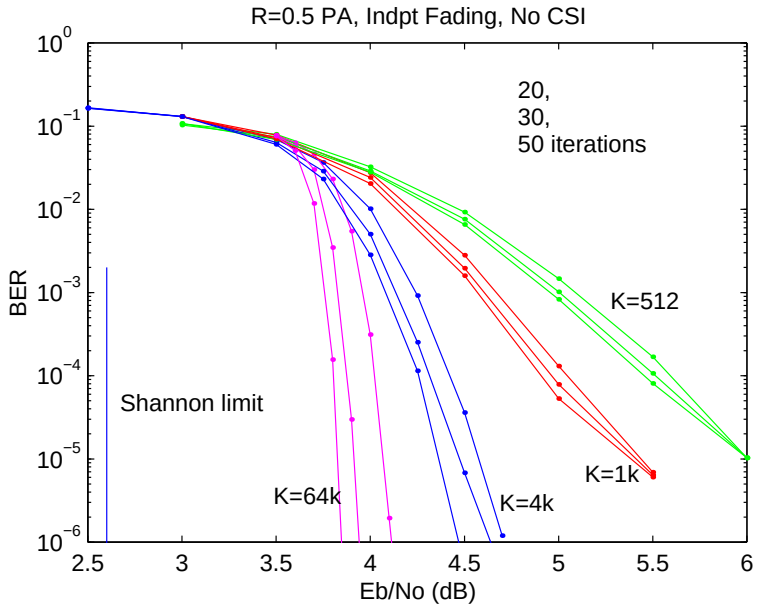


Fig. 35. Performance of PA codes on independent Rayleigh fading channels without CSI. (Rate 0.5, data block size 512, 1K, 4K, 64K.)

for each case. The degradation in simulation performance due to the lack of channel knowledge is about 0.9 dB, which is consistent with the gap between the respective channel capacities.

Compared to the (3,6)-regular LDPC codes reported in [66], the performance of a rate 1/2, codeword length $N = 128 \times 1024 = 1.3 \times 10^5$ PA code is about 0.4 and 0.25 dB better than a regular LDPC code with length $N = 10^5$ and 10^6 on independent Rayleigh channels either with or without CSI. It is possible that optimized irregular LDPC codes will outperform PA codes (as indicated by their thresholds), but for regular codes, PA codes seem one of the best.

The coherent performance of PA codes on correlated Rayleigh fading channels is shown in Figure 36. A rate 1/2 PA code is simulated for data block sizes K = 1K and 4K. Perfect CSI is assumed on the receiver side, and two common fading scenarios with normalized Doppler shifts $f_d T_s = 0.01$ and 0.001 are evaluated. As expected, the performance deteriorates rapidly as $f_d T_s$ decreases, since slower fading process brings less diversity order. Further, due to the interleaver between the PA code and the channel, the impact of slow fading is less severe for larger block sizes than for smaller ones. As can be seen, whereas K = 1K PA code loses about 7 dB at BER=10^{-4} when $f_d T_s$ is changed from 0.01 to 0.001, the loss with K=4K PA code is less than 5 dB.

To benchmark the performance of PA codes, Figure 37 compares the performance of a rate 0.75 PA code and that of a 16-state turbo code with whose component convolutional codes have polynomials $(23, 35)_{oct}$. Data block size is 4K and S-random interleavers are used in both codes to lower the possible error floors. Solid lines are for PA codes after 10 iterations and dashed lines are for turbo codes after 6 iterations. We observe that turbo codes perform about 0.6 and 0.7 dB better than PA codes for $f_d T_s = 0.001$ and 0.01, respectively. However, this performance gain comes at a price of a considerably higher complexity. While the message-passing decoding of this

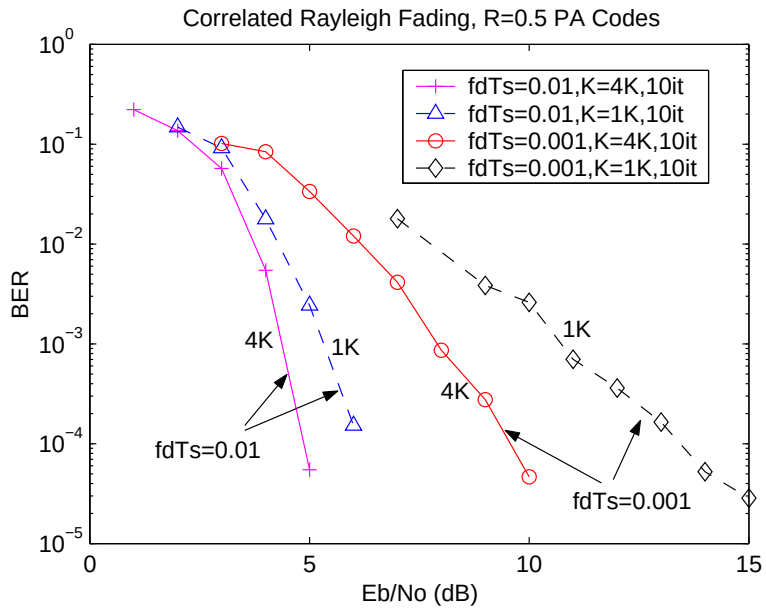


Fig. 36. Performance of PA codes on correlated Rayleigh fading channels with CSI. (Rate 0.5, normalized Doppler shift 0.01, 0.001, data block size 1K, 4K.)

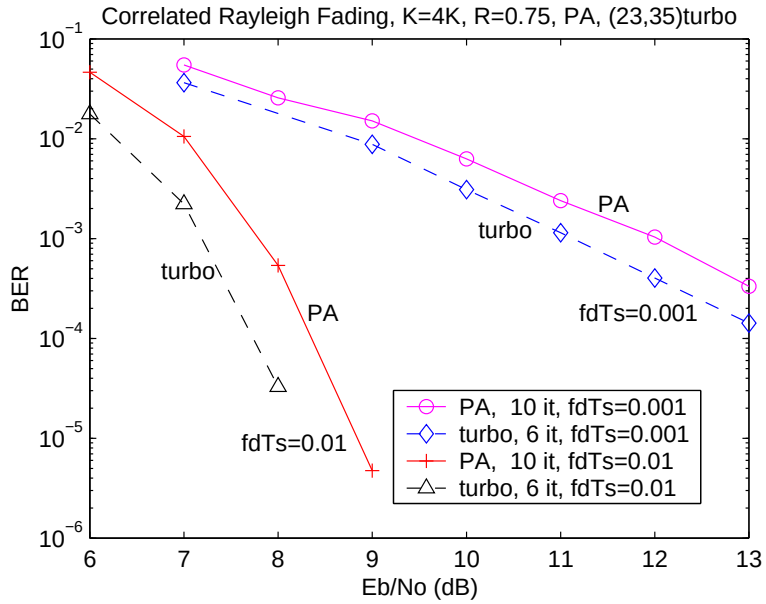


Fig. 37. Comparison of PA codes and (23,35) turbo codes on correlated Rayleigh fading channels with CSI. (Rate 0.75, data block size 4K, normalized Doppler shift 0.01, 0.001.)

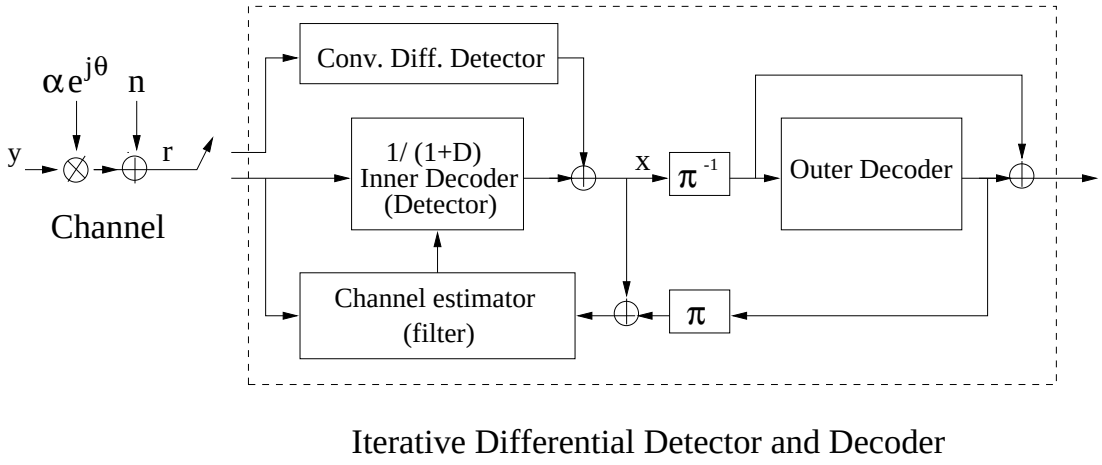


Fig. 38. Structure of iterative differential detection and decoding receiver.

PA code at 10_{th} iteration requires about 267 operations per bit [24], the log-domain BCJR decoding of turbo code at 6_{th} iteration requires as many as 9720 operations per data bit [26], which is more than 35 times as complex. Hence, PA codes are a good choice for providing reasonable performance at low lost.

D. Nocoherent Differential Detection

1. Iterative Differential Detection and Decoding

As mentioned before, PA codes are inherently differentially coded which makes it convenient for noncoherent differential detection. For complexity reasons, we consider a simple iterative differential detection and decoding receiver, whose structure is shown in Figure 38. The IDDD receiver is composed of a conventional differential detector, a phase tracking filter and the original (coherent) PA decoder. Trellis structure is used for the detection and decoding of the inner differential code $1/(1+D)$, but unlike the case of multiple symbol detection, the trellis is not extended (i.e., having 2 states only). Soft information feeds back and forth among different parts of the receiver according to the turbo principle to successively improve the estimation and

decoding. We use x to denote the input to the differential code (or the output from the outer code), and y the output from the differential code (or the symbol to be put on the channel) (see Figure 38). The differential encoder implements $y_k = x_k y_{k-1}$ for $x_k, y_k \in \{\pm 1\}$ (BPSK signal mapping $0 \rightarrow +1, 1 \rightarrow -1$). The received symbols are given by $r_k = \alpha_k e^{j\theta_k} y_k + n_k$, where the channel amplitudes (α_k's) and phases (θ_k's) are correlated, and the complex white Gaussian noise samples (n_k's) are independent.

Although in theory pilot symbols are not needed for differential decoding, in practice, however, pilot symbols are inserted periodically (even with multiple symbol detection) in order to avoid catastrophic performance caused by error propagation in the phase tracking. This is particularly important on fast fading channels where phases (θ_k) are changing rapidly (as will show later). Hence, some of the r_k's (and y_k's) in the received sequence are pilots symbols.

We use L to denote the LLR information, superscript (q) the q_{th} (global) iteration, and subscript i, o, ch and e the quantities associated with the inner code, the outer code, the fading channel and "the extrinsic", respectively. The IDDD receiver operates as follows.

a. IDDD Receiver

In the first iteration, the switch in Figure 38 is flipped up. The samples of the received symbols, r_k, are fed into the conventional differential detector which computes $u_k = \text{Real}(r_k r_{k-1}^*)$ ($*$ denotes the complex conjugate) and subsequently soft LLR $L_{ch}(x_k)$ from u_k. $L_{ch}(x_k)$ is then treated as $L_{e,i}^{(1)}(x_k)$ and fed into the outer decoder, which, in return, generates $L_{e,o}^{(1)}(x_k)$ for use in the next iteration of the inner detection/decoding. Starting from the second iteration, the switch in Figure 38 is flipped down, and a channel estimation of $\hat{\alpha_k}$ and $\hat{\theta}_k$ is performed before the "coherent" detection and decoding of the inner and outer code. After Q iterations, a decision is made by

combining the extrinsic information from both the inner and outer decoders: $x_k = \text{sign}(L_{e,i}^{(Q)}(x_k)+L_{e,o}^{(Q)}(x_k))$. For ease of proposition, we have ignored the existence of the random interleaver in the above discussion, but proper interleaving and deinterleaving should be conducted whenever needed.

b. Conventional Differential Detector

With the assumption that the carrier phases are near constant between two neighboring symbols, the conventional differential detector (in the first iteration) performs $u_k \stackrel{\Delta}{=} \text{Real}(r_k r_{k-1}^*)$ with the assumption that the carrier phases are near constant between two neighboring symbols. Hard decision of x_k can be obtained by simply checking the sign of u_k. In order to obtain soft information $L_{ch}(x_k)$ from u_k, pdf of u_k is needed. The conditional pdf of u_k given α_k and x_k is shown to be [68]

$$f_{U|\alpha,X}(u|\alpha,x) = \begin{cases} \frac{1}{2N_0}\exp\left(\frac{xu-\alpha^2/2}{N_0}\right), & -\infty < xu \le 0; \\ \frac{1}{2N_0}\exp\left(\frac{xu-\alpha^2/2}{N_0}\right)\mathcal{Q}\left(\sqrt{\frac{\alpha^2}{N_0}}, \sqrt{\frac{4xu}{N_0}}\right), & 0 < xu < \infty; \end{cases} \tag{3.25}$$

where $\mathcal{Q}(a,b)$ is the Marcum Q function. It is then possible to get the true pdf of u_k using

$$f_{U|X}(u|x) = \int_0^\infty f_{U|\alpha,X}(u|\alpha,x)\ f_\alpha(\alpha)\ d\alpha, \tag{3.26}$$

$$= 2\int_0^\infty f_{U|\alpha,X}(u|\alpha,x)\ \alpha e^{-\alpha^2}\ d\alpha. \tag{3.27}$$

However, an exact evaluation of (3.27) is difficult, since the computation of Marcum Q function can be slow and may not even converge at large values. This makes it quite difficult to compute LLR information. A simple compromise is to evaluate

(3.25) with α substituted by its mean value $E[\alpha]$, which leads to

$$f_{U|X}(u|x) = \begin{cases} \frac{1}{2N_0}\exp\left(\frac{xu-\pi/8}{N_0}\right), & -\infty < xu \le 0, \\ \frac{1}{2N_0}\exp\left(\frac{xu-\pi/8}{N_0}\right)\mathcal{Q}\left(\sqrt{\frac{\pi}{4N_0}}, \sqrt{\frac{4xu}{N_0}}\right), & 0 < xu < \infty. \end{cases} \tag{3.28}$$

The corresponding LLR from the channel can then be computed by

$$L_{ch}(x_k) = \log \frac{\Pr(u_k|x_k=+1)}{\Pr(u_k|x_k=-1)}, \tag{3.29}$$

$$= \text{sign}(u_k)\left(\frac{2|u_k|}{N_0} + \log\left(\mathcal{Q}\left(\sqrt{\frac{\pi^2}{4N_0}}, \sqrt{\frac{4|u_k|}{N_0}}\right)\right)\right). \tag{3.30}$$

A second treatment, which is more convenient, is to assume that u_k is Gaussian distributed as in [69] and other previous works. With this Gaussian assumption, we have

$$f_{U|X}(u|x) \approx \mathcal{N}\left(x, 2N_0 + N_0^2\right), \tag{3.31}$$

$$L_{ch}(x_k) \approx \frac{2u_k}{2N_0 + N_0^2}. \tag{3.32}$$

Alternatively, instead of using the conventional differential decoding in the first iteration, a channel estimation followed by the decoding of the inner $1/(1+D)$ code can be used, which makes the first iteration exactly the same as subsequent iterations. This then becomes pilot symbol assisted modulation (PSAM), which has slightly higher complexity than using differential detection in the first iteration.

To see how accurate the above treatments are, we plot in Figure 39 several curves approximating the pdf of u_k. From the most sharp and asymmetric to the least sharp and symmetric, these curves denote the exact pdf of $f_{U|X}(u|x=+1)$ from Monte Carlo simulations (which should be the numerical evaluation of (3.27)), the "mean-α approximated" pdf from (3.28) and the Gaussian approximated pdf from (3.31). We see that the Gaussian approximation, although simple, does not reflect the true pdf

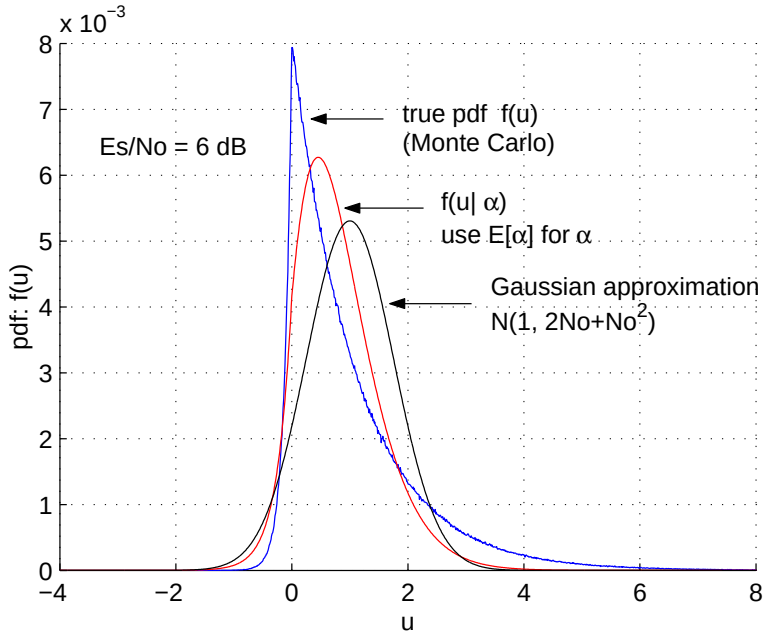


Fig. 39. Distribution of $u_k = Re\{y_k y_{k-1}^*\}$ in a conventional differential detection (assuming "+1" is transmitted).

well. However, it is interesting to note that despite of its inaccuracy, simulations do not show a noticeable degradation in performance compared to the other treatments like mean-α approximation and PSAM (see Figure 45). Hence, Gaussian approximation seems a simple and viable approach for noncoherent differential decoding.

c. Channel Estimator

The channel estimator in the IDDD receiver can be implemented in several ways. We use a linear filter of $(2L+1)$ taps to estimate α_k's and θ_k's in the q_{th} iteration

$$\hat{\alpha}_k^{(q)} e^{j\hat{\theta}_k^{(q)}} = \sum_{l=-L}^{L} p_l \ \hat{y}_{k-l}^{(q-1)} \ r_{k-l} \ , \tag{3.33}$$

where p_l denotes the coefficient of the l_{th} filter tap, and $\hat{y}_k^{(q-1)}$ denotes the estimate on y_k from the feedback of the previous iteration. For soft feedback, this is computed using $\hat{y}_k^{(q-1)} = \tanh(\frac{L_{e,i}^{(q-1)}(y_k)}{2})$, and for hard feedback, $\hat{y}_k^{(q-1)} = \text{sign}(L_{e,i}^{(q-1)}(y_k))$. The

LLR information $L_{e,i}^{(q-1)}(y_k)$ is generated together with $L_{e,i}^{(q-1)}(x_k)$ by the inner decoder in the $(q-1)_{th}$ decoding iteration (please refer to [24] for the exact message-passing decoding algorithm of $1/(1+D)$ code). In the initial iteration, $L_{e,i}^{(0)}(y_k)$'s are set to be zeros for code bits and a large positive number (denoting the point mass of LLR messages) for pilot symbols.

For the choice of the filter, Wiener filter is used since it is optimal for estimating channel gain in the minimum mean square error (MMSE) sense, provided that the correlation of the fading process, $\mathcal{R}_k$'s, are known to the receiver. The filter coefficients are obtained from the Wiener-Hopf equation

$$\begin{pmatrix} \mathcal{R}_0 - N_0 & \mathcal{R}_1 & \cdots & \mathcal{R}_{L-1} \\ \mathcal{R}_1 & \mathcal{R}_0 - N_0 & \cdots & \mathcal{R}_{L-2} \\ \cdots & \cdots & \cdots & \cdots \\ \mathcal{R}_{L-1} & \mathcal{R}_{L-2} & \cdots & \mathcal{R}_0 - N_0 \end{pmatrix} \cdot \begin{pmatrix} p_{-L} \\ p_{-(L-1)} \\ \cdots \\ p_L \end{pmatrix} = \begin{pmatrix} \mathcal{R}_{-L} \\ \mathcal{R}_{-L-1} \\ \cdots \\ \mathcal{R}_L \end{pmatrix}, \quad (3.34)$$

where $\mathcal{R}_k = \frac{1}{2}\mathcal{J}_0(2k\pi f_d T_s)$. Since the computation of p_l's from (3.34) involves an inverse operation on a matrix (one-time job), it may not be computable when the matrix becomes (near) singular. This happens when the channel is very slow fading. In such cases, a low-pass filter, or a simple "moving average" can be used [59].

2. EXIT Analysis

a. EXIT Charts

We conduct EXIT chart analysis [70] to further our insight into iterative differential detection and decoding. In EXIT charts, the exchange of extrinsic information is visualized as a decoding/detection trajectory, which allows the prediction of the convergence and other performance behavior of the iterative process [71]. Several quantities, like the bit error rate, the mean of the extrinsic LLR information and the

equivalent SNR value, can be used to depict the characteristics and relations of the component decoders. In this work, we use mutual information which is shown to be more reliable than the others [71]. The mutual information between the transmitted bit y_k and the corresponding LLR value of the *a priori* or extrinsic information $L_a(y_k)/L_e(y_k)$, denoted as $I(Y, L_{a/e}(Y))$, is defined as

$$I(Y, L(Y)) \triangleq \frac{1}{2}\sum_{y=\pm 1}\int_{-\infty}^{\infty} f_{L(y)}(\eta|y)\,\log_2\frac{2f_{L(y)}(\eta|y)}{f_{L(y)}(\eta\,|+1)+f_{L(y)}(\eta\,|-1)}d\eta, \quad (3.35)$$

$$= \int_{-\infty}^{\infty} f_{L(y)}(\eta\,|+1)\,\log_2\frac{2f_{L(y)}(\eta\,|+1)}{f_{L(y)}(\eta\,|+1)+f_{L(y)}(-\eta\,|+1)}d\eta, \quad (3.36)$$

$$= 1-\int_{\infty}^{\infty} f_{L(Y)}(\eta\,|+1)\;\log_2(1+e^{-\eta})\,d\eta, \quad (3.37)$$

where $f_{L(Y)}(\eta\,|Y=y)$ is the conditional pdf of the LLR messages of y_k. The second equality holds when the channel is output symmetric, i.e., $f_{L(y)}(\eta\,|Y=-y) = f_{L(y)}(-\eta\,|Y=y)$, and the third equality holds when the received messages satisfies the consistency condition (also known as the symmetry condition), i.e., $f_{L(y)}(\eta\,|Y=y) = f_{L(y)}(-\eta\,|Y=y)\,e^{y\eta}$. It should be noted that although the consistency condition is an invariant during the message-passing process on a number of channels including the AWGN channel and the independent Rayleigh fading channel with perfect CSI, it is not preserved on channels with no CSI or with estimated CSI, since the initial density function in these cases is an approximation of the actual pdf of the LLR messages. Hence, (3.36) should be used to compute the mutual information in such cases. We use X-axis to denote the mutual information to the inner code (*a prior*) or from the outer code (extrinsic), denoted as $I_{a,i}/I_{e,o}$, and Y-axis the mutual information from the inner code or to the outer code, denoted as $I_{e,i}/I_{a,o}$.

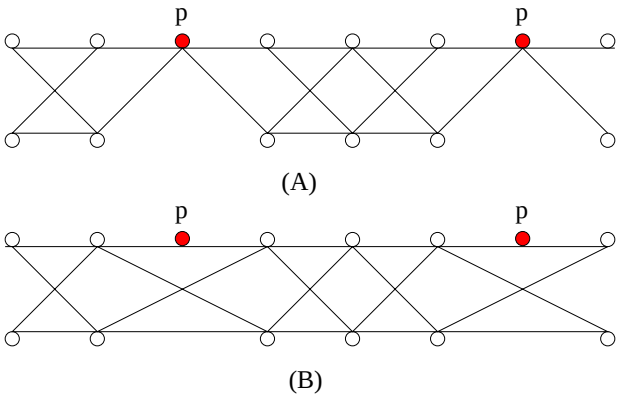


Fig. 40. Trellis diagram of B-DPSK with pilot insertion. (A) Pilot symbols periodically terminate the trellis. (B) Pilot symbols are separated from the trellis structure.

b. Pilot Insertion

We investigate the issue of pilot insertion. It is well-known that while the amount of pilot symbols inserted should be sufficient to track the channel, it should not be in excess so as to cause a waste of bandwidth and energy. Many researchers have reported that excessive pilot symbols could cause a noticeable performance degradation. The explanation was that pilot symbols had averaged down the energy per transmitted symbol too much for a good channel estimation and detection, or equivalently, that the performance gain obtained in channel tracking was not enough to compensate for the energy/rate loss caused by pilot symbols. While this is true, little attention has been paid to the fact that in the case of differential coding, improperly inserted pilot symbols could cause an inherent loss in code capacity. As shown in Figure 40, there are different ways to pilot symbols in a differential code. The most popular approach has been to periodically terminate the trellis (Figure 40(A)) [60], where pilot symbols assume a dual role of channel estimation and $1/(1 \oplus D)$ decoding. Unfortunately, this is in fact not a good strategy since segmenting the trellis into small chunks causes a significant amount of short error events (an "inverse" effect of spectrum thinning),

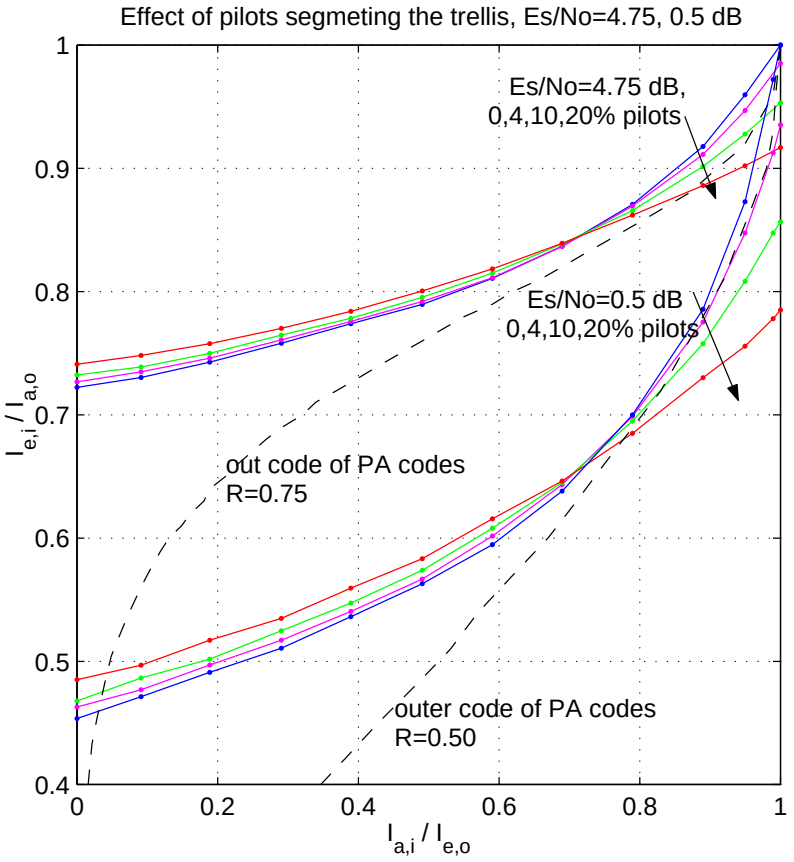


Fig. 41. The effect of pilot symbols segmenting the trellis on the performance of the differential decoder. (Normalized Doppler rate 0.01, E_s/N_0=4.75 dB and 0 dB, perfect CSI.)

and consequently a loss in capacity.

This "segmentation effect" can be best illustrated in Figure 41, where EXIT curves for the differential decoder with 0%, 4%, 10% and 20% pilot insertion are plotted for two different SNR values. We assume that the four curves in each family have the same energy per transmitted symbol and that perfect channel information on the fading phase and amplitude is known to the receiver (irrespective of the amount of pilot symbols). Hence, the difference of the curves in each family is only due to the difference in pilot spacing. At the left end of the curves (i.e., input mutual

information is small), we see that the curves with more pilots are slightly better than those with fewer pilots. This is because when there is little feedback from the outer code, pilot symbols are the major contribution to *a priori* information. However, at the right end, when there is sufficient information provided by the outer code, the curves behave just the opposite to the left, since pilot symbols are no longer an important source of *a priori* information. Rather, their negative impact of segmenting the trellis and shortening the (average) error events becomes dominant, causing a considerable performance loss. The performance degradation is more severe when more pilot symbols are inserted and when the code is operating at a lower SNR level. As can be seen, with 20% of pilot insertion (pilot spacing 5), even when "perfect" input mutual information $I_{a,i}$ is provided, the inner code is unable to produce sufficient output mutual information $I_{e,i}$. The inner EXIT curve tends to intersect the outer EXIT curve at a very early stage of the iterative decoding process, causing the PA decoder to fail at a high BER level. (This is in addition to 20% more of energy consumption than the non-pilot case!)

The immediate implications of the above plot are that, first, when pilots are inserted to terminate the trellis, error free is not possible with PA codes and many other serial concatenated schemes including nonsystematic IRA codes, convolutional accumulated codes and the turbo coding schemes discussed in [60]). Specifically, unless the outer code is capacity achieving at least at some SNR (i.e., the outer code alone is a "good" code, like conventional LDPC codes), there will be error floors. In such cases, the use of pilot symbols should be especially prudent, so that error floors do not occur at a high BER level. Second, it suggests that a better way of inserting pilot symbols is to separate them from the trellis as shown in Figure 40(B), so that pilot symbols do not affect error events.

To verify the analytical results, we simulate the performance of a rate 1/2, data

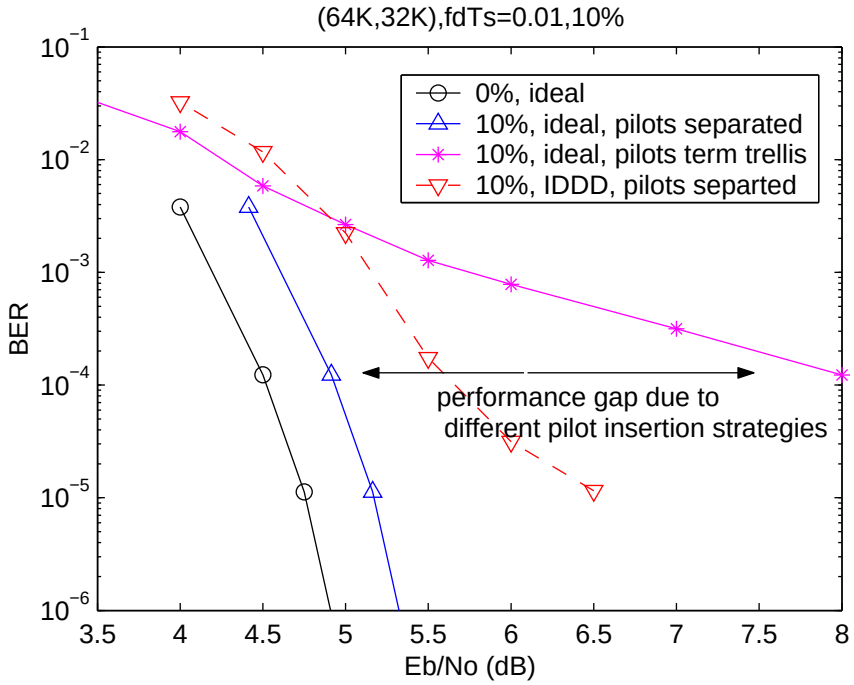


Fig. 42. Simulations of PA codes with different pilot insertion strategies. (Normalized Doppler rate 0.01, code rate 0.5, data block size 32K, 0% or 10% pilot insertion, 10 iterations.)

block size 32K PA code with different strategies of pilot insertion (Figure 42). The normalized Doppler spread is 0.01, and error rate after 10 iterations are evaluated. Solid lines represent the cases where perfect channel knowledge is known to the receiver, and dashed line the case where noncoherent detection is used. We observe a drastic performance gap resulted from different strategies of pilot insertion. In this specific case, by segmenting the trellis every 10 symbols, improper pilot insertion has caused more than 3 dB loss at BER of 10^{-4}. It is interesting to see that if we overlook the impact of the different strategies of pilot insertion, we might get the "surprising" result that noncoherent detection (dashed line) performs (noticeably) better than coherent detection (rightmost solid line)!

c. Code Matched to Differential Coding

As mentioned before, the outer code of PA codes is a special type of LDPC code. Given PA codes perform well (especially at high rates), one tend to ask how a general LDPC code will perform with differential coding. This is an interesting question, since it directs to the solution of other interesting problems like how to perform noncoherent detection with LDPC codes. Before we answer the question, we first note two important facts about EXIT analysis. First, in order for iterative decoding to converge successfully, the outer EXIT curve should be strictly below the inner EXIT curve, so that there is an open passage between the two curves. Second, it has been shown in [70] that the area under the EXIT curve, $\mathcal{A} = \int_0^1 I_e dI_a$, is closely related to the code rate. When the *a priori* information is coming from the erasure channel and when the decoder is an optimal decoder, the area is exactly the code rate. For other channels, this may not be exact, but is nevertheless a good approximation as verified by empirical results. The implication of the above two facts is that, in order to fully achieve the capacity provided by the inner differential code, the outer code needs to have an EXIT curve closely matched in shape and in position to that of the inner code. Unfortunately, this is not the case of a conventional LDPC code (outer code) and a differential code (inner code).

In Figure 43, we plot a set of three outer EXIT curves corresponding to a regular LDPC code, an irregular LDPC code and the outer code of a PA code, and a set of two inner EXIT curves corresponding to a differential code (on correlated Rayleigh channel) and the plain Rayleigh fading channel. The normalized Doppler rate is $f_d T_s = 0.01$, all outer codes have rate 3/4, and perfect channel fading information is assumed to all the receivers. The regular LDPC code in the plot is (3,12)-regular, and the irregular one is optimized with check node degree profile $\rho(x) = x^{20}$ and variable

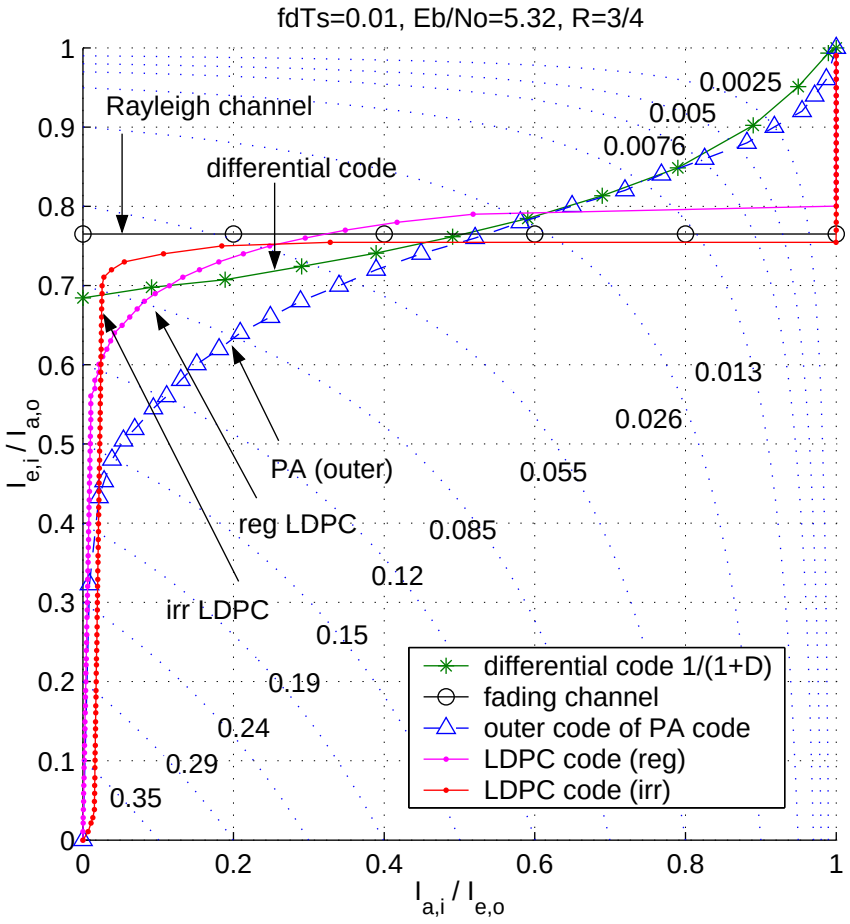


Fig. 43. EXIT curves of LDPC codes, PA codes (outer code only), Rayleigh channels and differential codes. (Normalized Doppler rate 0.01, E_b/N_0=5.32dB, code rate $3/4$, $(3, 12)$-regular LDPC code and optimized irregular LDPC code with $\rho(x) = x^{20}$ and $\gamma(x) = 0.1510x + 0.1978x^2 + 0.2201x^6 + 0.0353^7 + 0.3958x^{29}$.)

node degree profile $\gamma(x) = 0.1510x + 0.1978x^2 + 0.2201x^6 + 0.0353^7 + 0.3958x^{29}$, which has a threshold of 0.6726 (about 0.0576 dB away from the AWGN capacity) [72]. We observe that while the outer code of (high-rate) PA codes shows a good match with an inner differential code, a conventional LDPC code (regular or irregular) will either intersect with the differential code (decoder failure) or leave a huge area between them (a waste in code capacity). The observation that LDPC codes match better with a plain channel than with a differential code indicates that, unless specifically designed, LDPC codes should not be used with a differential code (or more generally with any recursive inner code/modulation [73] [74]). Put another way, an LDPC code that is optimal in the conventional sense (i.e., BPSK modulated on memoryless channels) is not optimal when combined with an inner recursive code/modulation. However, not using differential coding typically requires more pilot symbols in order to track the channel well (especially on fast fading environments). Hence, it is expected that on (fast) fading channels where only limited bandwidth expansion is allowed, (conventional) LDPC codes do not perform well with noncoherent detection (whether or not differential coding is used). On the other hand, (high-rate) PA codes are able to make use of the (intrinsic) differential code for noncoherent detection, and hence are a better choice for bandwidth-limited wireless applications. This is confirmed by simulations shown later.

3. Simulation Results of Noncoherent Detection

The performance of noncoherent detected PA codes on fast Rayleigh fading channels are presented below. Unless otherwise indicated, the BER curves shown are after 10 global iterations, and in each global iteration 4 to 6 local iterations of the outer code are performed. We have chosen these parameters on the basis of a set of simulations which show that they are the best tradeoff.

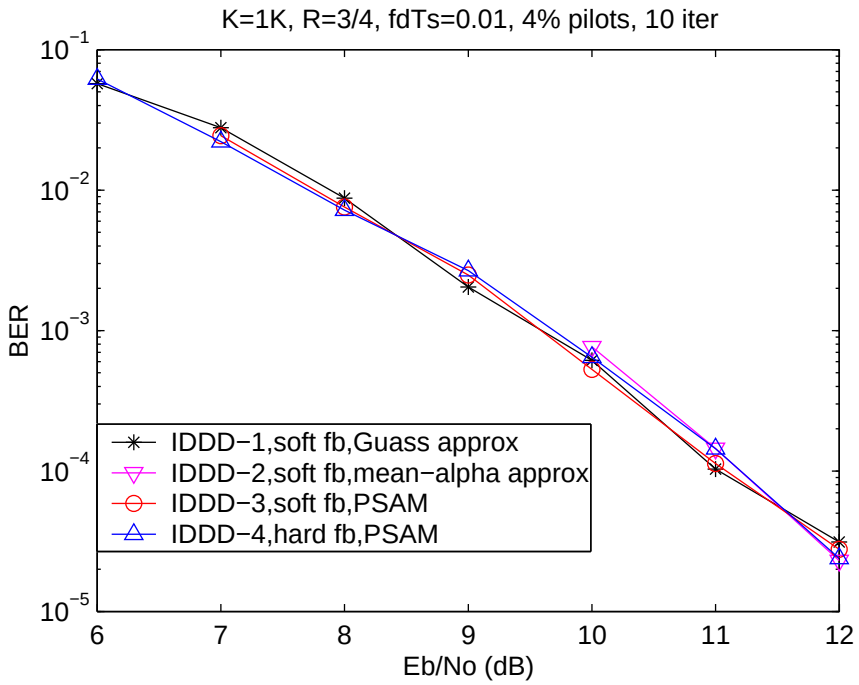


Fig. 44. Comparison of BER performance for several noncoherent receiver strategies on correlated Rayleigh channels with f_dT_s=0.01. (Code rate 0.75, data block size 1K, 4% of bandwidth expansion, filter length 65.)

We first compare the BER performance of 4 types of iterative differential detection and decoding strategies for a $K = 1$K, $R = 3/4$ PA code on a $f_dT_s = 0.01$ Rayleigh fading channel in Figure 44. Curve labeled with IDDD-1 uses conventional differential detection with Gaussian approximation (3.32) to compute $L_{ch}(x_k)$ in the first iteration, and soft feedback of $\hat{y}_k$ in all iterations to assist channel estimation; IDDD-2 uses conventional differential detection with "mean-α" approximation (3.30) in the first iteration and soft feedback in all iterations; IDDD-3 is PSAM with soft feedback; and IDDD-4 is PSAM with hard feedback. In all case, 4% of pilot symbols are inserted and curves shown are after 10 iterations. It is interesting to see that, in this case different decoding strategies in the first iteration does not make much difference, and the performance is not very sensitive to whether the feedback is hard

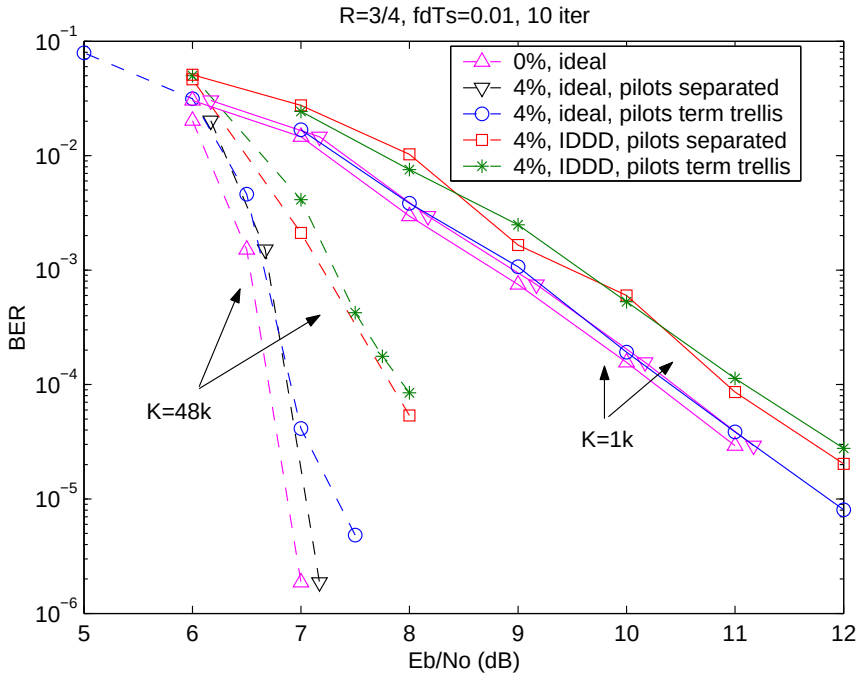


Fig. 45. Comparison of BER performance for several transmission/reception strategies for PA codes of large and small block sizes on correlated Rayleigh channels with f_dT_s=0.01. (Code rate 0.75, data block size 48K and 1K, 4% of bandwidth expansion, filter length 65.)

or soft information. Although not shown, simulations of a long PA code (K=48K) of the same (high) rate (R=3/4) reveal a similar phenomenon. It is possible, however, that the difference in decoding strategies especially the difference in the feedback information, will cause difference in performance in other cases [59].

Figure 45 compares the coherent and noncoherent performance of rate 3/4 PA codes after 10 iterations on fast Rayleigh fading channels with Doppler rate T_sf_d = 0.01. Both short block size of 1K and large block size of 48K are evaluated. In each case, a family of 5 curves showing BER versus E_b/N_o are plotted (rate loss due to pilot symbols are accounted for). The leftmost three curves are the ideal case where fading amplitude and phase is known to the receiver (coherent detection) and the two right curves are the noncoherent case where IDDD is used. First, we observe

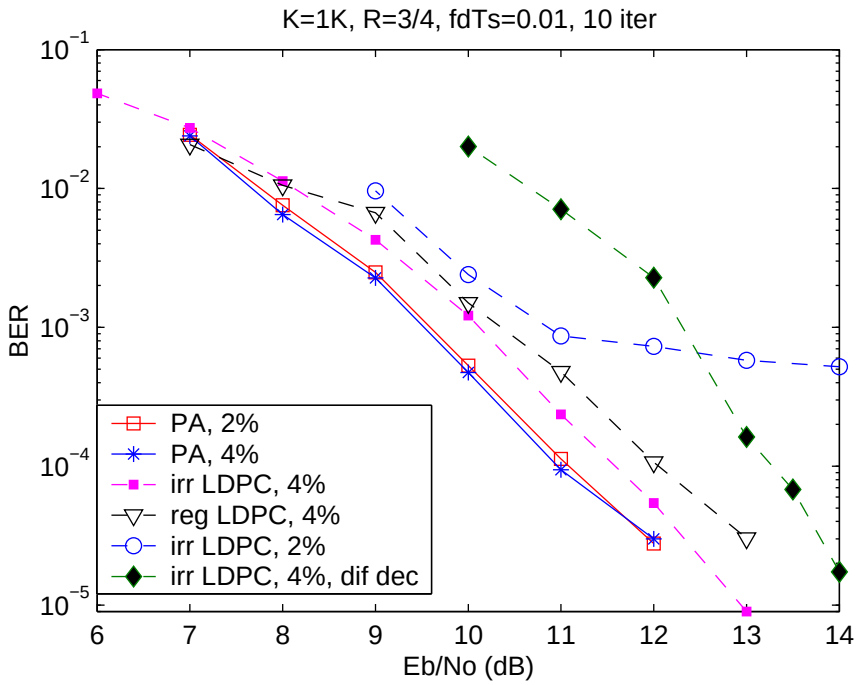


Fig. 46. Comparison of PA codes and LDPC codes on fast fading Rayleigh channels with noncoherent detection and decoding. (Solid line: PA codes, dashed lines: LDPC codes. Code rate 0.75, data block size 1K, filter length 65, normalized Doppler spread 0.01. The degree profiles of the regular and irregular LDPC codes are the same as in Figure. 43.)

that there is slight performance degradation caused by pilot symbols terminating the trellis, both in the coherent case and the noncoherent case, but the effect is not as drastic as the case in Figure 42. Second, due to the imperfect channel estimation, we see that the noncoherently detected codes are about 1 dB and 0.55 dB away from the ideal (coherent) detection at BER of 10^{-4} for block sizes of 48K and 1K, respectively. Considering that only 4% pilot symbols are inserted to tract the channel, and that the IDDD receiver is very low-complexity, this performance is quite satisfying.

Figure 46 compares the performance of PA codes and LDPC codes using noncoherent detection on Rayleigh channels with normalized Doppler spread of 0.01. All codes are of data block size 1K and code rate 3/4. Soft feedback is used and either 2%

or 4% pilot symbols are inserted. LDPC codes are evaluated either with or without a differential code and their degree profiles are the same as specified in Figure 43. From most power efficient to least power efficient, the curves shown are 1) PA codes with 4% of pilot symbols, 2) PA codes with 2% of pilot symbols, 3) BPSK-coded irregular LDPC codes with 4% of pilot symbols, 4) BPSK-coded regular LDPC codes with 4% of pilot symbols, 5) BPSK-coded irregular LDPC codes with 2% of pilot symbols, 5) differentially-coded (BDPSK-coded) irregular LDPC codes with 4% of pilot symbols. We see that with 4% of bandwidth expansion, BPSK-coded irregular and regular LDPC codes are about 0.5 and 1 dB worse than PA codes at BER of 10^{-4}, respectively, yet the differentially-coded irregular LDPC code is more than 2.2 dB worse. This confirms that (conventional) LDPC codes suffer a performance loss when used with a differential code. Further, while the performance gap between irregular LDPC codes and PA codes is acceptable (0.5 dB) with 4% of pilot symbols, it becomes quite significant when pilot symbols are reduced in half. For PA codes, 2% of pilot symbols are still sufficient to yield desirable performance. However, 2% of pilot symbols are insufficient for LDPC codes (without a differential code) to estimate the channel, thus causing a considerable performance loss and an error floor as high as BER of 10^{-3}. This shows the advantage of PA codes to (conventional) LDPC codes when noncoherent detection is required and when only limited bandwidth expansion is allowed.

We investigate how the number of pilot symbols and the length of the estimation filter affect the performance of noncoherent detection. Figure 47 illustrates the impact of the pilot spacing on the BER performance on fast fading channels where the normalized Doppler spread is either 0.05, 0.02 or 0.01. We observe the following: 1) The IDDD receiver is robust for different Doppler rates. 2) In any case, pilot spacing should be at least 6 symbols, since further increasing the number of pilot symbols

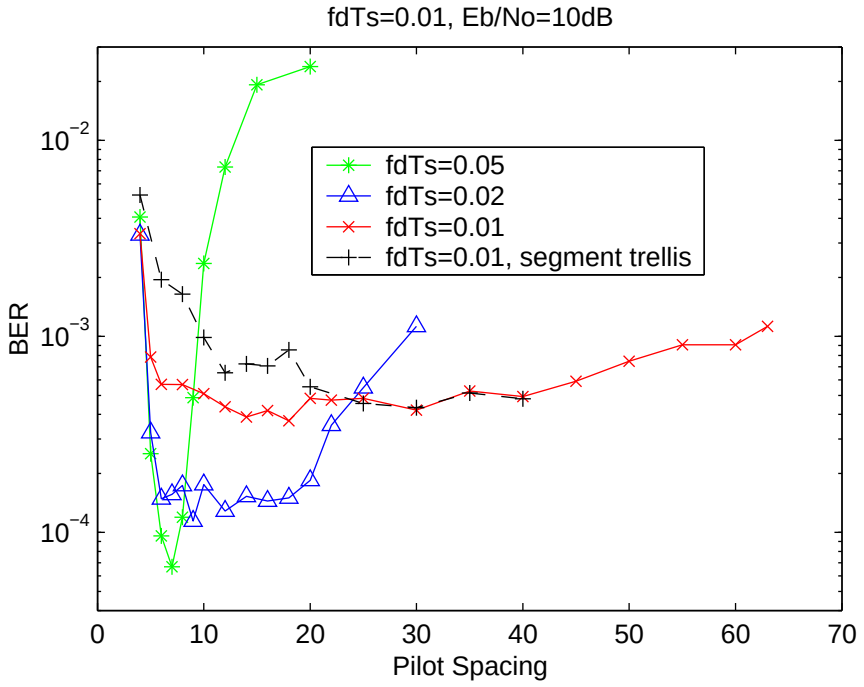


Fig. 47. Effect of the number of pilot symbols on the performance of noncoherent detected PA codes on correlated Rayleigh channels with f_dT_s=0.01. (Code rate 0.75, data block size 1K, filter length 65.)

would have consumed unnecessarily large portion of the overall energy without being able to compensate for it. 3) The code performance at high Doppler rates is more sensitive to pilot spacing than that at lower Doppler rate. At the normalized Doppler rate of 0.01 (although already fast fading), we see noncoherently detected PA codes can tolerate pilot spacing as small as 6 symbols and as large as 45 to 50 symbols (put aside the bandwidth issue); yet at extremely fast Doppler rate of 0.05, pilot spacing beyond 7-9 symbols will soon cause drastic performance degradation. Also shown in the plot is a case where pilot symbols periodically terminate the trellis (dashed line), and again we observe the performance loss caused by trellis segmentation. The effect of the length of the estimation filter is also studied. We observe that although the filter length also plays a role in the overall performance, the impact is limited

compared to pilot spacing. Similar results have been reported by other researchers [59] and, hence, we omit the plot here.

E. Code Design from the Convergence Property

1. Convergence-Constraint Design Method

The above EXIT chart analysis and computer simulations show that a conventional LDPC code is not suitable for differential coding, and that the outer code of PA codes seems to be a better match. This raises more interesting questions: How well does the outer code of PA codes match with the inner differential code? What types of degree files are good for differential coding and how to optimize them?

To the solution of the above questions, the work of Richardson, Urbanke, *et al.*, serve as the important ground stones. In [14], the nonlinear optimization problem of the irregular LDPC degree profiles is formulated on AWGN channels and the method of density evolution is introduced to solve the problem. In [32], a Gaussian approximation is applied to the density evolution method, which reduces the problem to be a linear optimization problem. The work of [75] [66] and [73] have further combined density evolution with differential evolution in the design of good LDPC ensembles for the erasure channel, the independent Rayleigh fading channel and the minimum shift keying (MSK) modulation, respectively. Observe that the above works on the optimization of LDPC ensembles have all aimed at the asymptotic threshold, where the cost function is set such that, for a fixed code rate, the corresponding SNR threshold is minimized, or equivalently, for a given SNR value, the code rate is maximized (call it "threshold-constraint" method). This is justified, since in all the aforementioned works, the only involvement of the channel is to provide the initial LLR information to trigger the start of the density evolution process. However, the

problem we consider here is somewhat different. Our goal is to design codes that can fully achieve the capacity provided by the given inner receiver. The key here is to control the interaction or the convergence behavior between the inner and outer code[3], which is reflected in the shapes and positions of the corresponding EXIT curves.

Below we propose a "convergence-constraint" method which is a useful extension of the conventional threshold-constraint method. The idea is to "sample" the inner EXIT curve and design an EXIT curve that matches with these sample points (or the "control points"). Mathematically, if we choose a set of M control points in the EXIT chart, denoted as (v_1, w_1), (v_2, w_2), $\cdots$, (v_M, w_M), and if we use $\mathcal{T}_o(\cdot)$ to denote the input-output mutual information transfer function of the outer LDPC code (exact expression of $\mathcal{T}_o$ will be defined later in (3.50)), the optimization problem can be formulated as

$$\max_{\sum_{i=1}^{D_v}\lambda_i=1,\ \sum_{j=2}^{D_c}\rho_j=1} \left\{ R = 1 - \frac{\sum_{j=2}^{D_c}\rho_j/j}{\sum_{i=1}^{D_v}\lambda_i/i} \;\middle|\; \mathcal{T}_o(w_k) \geq v_k,\ k = 1, 2, \cdots, M \right\}. \quad (3.38)$$

where R denotes the code rate, λ_i and ρ_i denote the fraction of edges in the bipartite graph that are connected to variable nodes and check nodes of degree i, respectively.

To ease the computation, we assume that the LLR messages are Gaussian or mixed Gaussian distributed, and that the output extrinsic mutual information of an irregular LDPC code is a linear combination of those from the regular codes. We note that whereas the Gaussian assumption for LLR messages is not far from reality on AWGN channels, it is less accurate for Rayleigh fading channels [66]. Nevertheless, Gaussian assumption is used since tracking the exact message pdf's involves tedious computation and may not render any feasible analytical expression. Further, the use

[3]It is true that whether or not the iterative process will converge successfully is also reflected in the final threshold value, but the primary concern here is on the interactions between the component codes.

of EXIT charts has already assumed a Gaussian approximation in the first place.

Let us first recall the main steps of code optimization using the conventional threshold-constraint density evolution with Gaussian assumption [32]:

Notation – Conforming to the notations and graphic framework presented in [32], we use $\lambda(x) = \sum_{i=1}^{D_v} \lambda_i x^{i-1}$ and $\rho(x) = \sum_{i=2}^{D_c} \rho_i x^{i-1}$ to describe the degree profiles from the edge perspective, where λ_i and ρ_i denote the fraction of edges in the bipartite graph that are connected to variable nodes and check nodes of degree i, respectively, and D_v and D_c denote the maximum variable node and check node degrees. Similarly, $\lambda'(x) = \sum_{i=1}^{D_v} \lambda_i' x^{i-1}$ and $\rho'(x) = \sum_{i=2}^{D_c} \rho_i' x^{i-1}$ are used to describe the degree profiles from the node perspective. We have the following relations (R is code rate)

$$\lambda_i' = \frac{\lambda_i / i}{\sum_{j=1}^{D_v} \lambda_j / j}, \tag{3.39}$$

$$\rho_i' = \frac{\rho_i / i}{\sum_{j=2}^{D_c} \rho_j / j}. \tag{3.40}$$

We use superscript (l) to denote the l_{th} LDPC decoding iteration, and subscript v and c to denote the quantities pertaining to variable nodes and check nodes, respectively. Further, we define two functions that will be useful and convenient for the discussion

$$\mathcal{I}(x) = 1 - \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi x}} e^{-\frac{(z-x)^2}{4x}} \log(1 + e^{-z}) dz, \tag{3.41}$$

$$\phi(x) = \begin{cases} 1 - \frac{1}{\sqrt{4\pi x}} \int \tanh\frac{z}{2}\, e^{-\frac{(z-x)^2}{4x}} dz, & x > 0, \\ 1, & x = 0. \end{cases} \tag{3.42}$$

Function $\mathcal{I}(x)$ maps the message mean x to the corresponding mutual information (under Gaussian assumption), and $\phi(x)$ is useful in describing how the message mean evolves in $\tanh(\frac{y}{2})$ operation, where y follows Gaussian distribution with mean x and variance $2x$.

The complete design process is in fact a dual constraint optimization process that

progressively optimizes variable node degree profile $\lambda(x)$ and check node degree profile $\rho(x)$ based on the other. It is interesting to note that although the optimization of $\lambda(x)$ and $\rho(x)$ are duals to each other, experiments show that the optimality of $\lambda(x)$ has a larger impact on the asymptotic performance than that of $\rho(x)$. In many cases, since the optimal check node degree profile are known to follow the concentration rule [32], i.e., $\rho(x) = \Delta x^k + (1-\Delta)x^{k+1}$, one can make a reasonable prediction on $\rho(x)$ and optimize for $\lambda(x)$ only. Below we focus our discussion on the optimization of $\lambda(x)$ for a given $\rho(x)$. The optimization of $\rho(x)$ for a given $\lambda(x)$ can be derived similarly.

Threshold-constraint method (optimizing $\lambda(x)$) – Under the assumption that the messages passed along all edges are i.i.d. and Gaussian distributed, the average messages variable nodes receive from their neighbors are mixed Gaussian distributed. From $(l-1)_{th}$ iteration to l_{th} local iteration (in the LDPC decoder), the mean of the messages associated with the variable node, m_v, evolves as

$$m_v^{(l)} = \sum_{i=2}^{D_v} \lambda_i \, \mathcal{N}(m_{v,i}^{(l)}, 2m_{v,i}^{(l)}), \tag{3.43}$$

$$= \sum_{i=2}^{D_v} \lambda_i \phi \left(m_0 + (i-1) \sum_{j=2}^{D_c} \rho_j \phi^{-1}(1-(1-m_v^{(l-1)})^{j-1} \right), \tag{3.44}$$

where m_0 denotes the mean of the initial messages received from the channel (or the inner code). Let us denote

$$h_i(m_0, r) \triangleq \phi\Big(m_0 + (i-1) \sum_{j=2}^{D_c} \rho_j \phi^{-1}\big(1-(1-r)^{j-1}\big)\Big), \tag{3.45}$$

$$h(m_0, r) \triangleq \sum_{i=2}^{D_v} \lambda_i h_i(m_0, r), \tag{3.46}$$

Equation (3.44) can then be rewritten as $r_l = h(m_0, r_{l-1}) = \sum_{i=2}^{D_v} \lambda_i h_i(m_0, r_{l-1})$. The conventional threshold-constraint density evolution ensures that the optimized

degree profile, if successful, will converge to the zero-error state (asymptotically) at the given initial message mean m_0. This is achieved by enforcing $r > h(m_0, r)$ for all $r \in (0, \phi(m_0)\,]$ [32]. Viewed from the EXIT chart, the threshold-constraint method has implicitly used a control point $(v, w) = (1, I(m_0))$ such that that resulting EXIT curve will stay below it.

Convergence-constraint method (optimizing $\lambda(x)$) – In the general case, a control point (v, w) can choose any value from 0 to 1; that is, for the given *a priori* mutual information w, we need the extrinsic mutual information at the output of the LDPC code to be better than v, but not necessarily to reach the point mass of 1. The constraint is correspondingly changed to $r > h(m_0, r)$ for all $r^* \in (0, \phi(m_0)\,]$ where $r^*(\geq 0)$ is the critical value to ensure that $\mathcal{T}_0(w) \geq v$ is satisfied. Formally, the problem can be formulated as follows: given a control point of (v, w), where $0 \leq v, w, \leq 1$, and check node degree profile $\rho(x)$,

$$\max_{\sum_{i=1}^{D_v} \lambda_i = 1} \quad \sum_{i=1}^{D_v} \lambda_i / i, \tag{3.47}$$

$$\text{subject to:} \quad (i) \quad \sum_{i=1}^{D_v} \lambda_i = 1, \tag{3.48}$$

$$(ii) \quad \sum_{i=1}^{D_v} \lambda_i \left(h_i(m_0, r) - r\right) < 0, \qquad \forall r \in (r^*, \phi(m_0)\,], \tag{3.49}$$

where $m_0 = \mathcal{I}^{-1}(w)$ and r^* satisfies

$$\mathcal{T}_o(w) \triangleq \sum_{i=1}^{D_v} \lambda_i' \, \mathcal{I}\Big(i \sum_{j=2}^{D_c} \rho_j \phi^{-1}\big(1 - (1 - r^*)^{j-1}\big)\Big) \geq v. \tag{3.50}$$

When $v = 1$, we see from (3.50) that $r^* = 0$, which is the special case of threshold constraint design.

Hence, for a set of M control points, (v_1, w_1), (v_2, w_2), $\cdots$, (v_M, w_M), where $0 \leq v_1 < v_2 < \cdots < v_M \leq 1$ and $0 \leq w_1 \leq w_2 \leq \cdots \leq w_M \leq 1$, we can combine

the constraints associated with each individual control point and perform a joint optimization on all of them. When the control points are properly chosen along the inner EXIT curve and when the initial $\rho(x)$ is properly set, we expect the resulting EXIT curve to stay closely below the inner EXIT curve. That it, the resulting (outer) code to match closely with the inner code/receiver in convergence behavior.

Linear programming – Note that the above constraint (ii) is a nonlinear function of λ_i's, and that the computation of r^* from (3.50) requires the knowledge of $\lambda(x)$, which is yet to be optimized. To get around with this, one possible approach is to consider an approximation of $\lambda(x)$ in (3.50) to compute r^*. Specifically, we consider only the two lowest degree variable nodes λ_{i_1} and λ_{i_2}, and approximate the degree profile as $\tilde{\lambda}(x) = \lambda_{i_1} x^{i_1-1} + \lambda_{i_2} x^{i_2-1} + O(\lambda_{i_2+1} x^{i_2}) \approx \lambda_{i_1} x^{i_1-1} + (1-\lambda_{i_1}) x^{i_1}$.

1. In a conventional LDPC ensemble, $i_1 = 2$, i.e., degree-1 nodes are not allowed, since the outbound messages from these variable will not improve in the message-passing decoding. In such cases, we consider only degree-2 and 3 nodes. We use an upper bound λ_2^* for degree-2 nodes, and treat all the rest as degree-3 nodes. The value of λ^* can be derived from the stability condition [14] [32]. Stability condition states that there exists a value $\xi > 0$ such that if density evolution is initialized with a symmetric message density P_0 satisfying $\int_{-\infty}^{0} P_0(x)dx < \xi$, then the necessary and sufficient condition for the density evolution to converge to the zero-error state is $\lambda^{'}(0)\rho^{'}(1) < e^{\gamma}$, where $\gamma \stackrel{\Delta}{=} -\log(\int_{-\infty}^{\infty} P_0(x)e^{-x/2}dx)$. Applying the stability condition on Gaussian messages[4] with initial mean value m_0, we get $\gamma = \frac{m_0}{4}$ and $\lambda_2^* = e^{m_0/4} / \sum_{j=2}^{D_c}(j-1)\rho_j$, or equivalently, $\lambda_2^*(w) = e^{\mathcal{I}^{-1}(w)/4} / \sum_{j=2}^{D_c}(j-1)\rho_j$. It should be noted that not

[4]The extrinsic information output from the inner differential code is in fact not Gaussian distributed, and we did not use an Gaussian approximation in computing its extrinsic mutual information. However, the outer LDPC code has no means of knowing the exact pdf of the mutual information passed from the inner code and, hence,

all values of w_k can be used in the above equation to compute λ_2^*. Recall that the stability condition is to ensure that the code will converge (asymptotically) to the zero-error state for the given input messages. Hence, $\lambda_2 \leq \lambda_2^*(w^*)$ is valid and is required only when the output mutual information will approach 1 at the input mutual information w^*. When we sample the inner EXIT curve, we can always choose one sample point, say the rightmost point to roughly satisfy the requirement, i.e., $(v_M, w_M) \approx (1, w_M)$. We can then use w_M to compute $\lambda_2^* = \lambda_2^*(w_M)$, and subsequently use $\tilde{\lambda}(x) \approx \lambda_2^* x + (1-\lambda_2^*)x^2$ to compute r^* from (3.50) for all control points from 1 to M.

2. In a nonconventional case when an LDPC code is used together with a differential code (or other inner code and/or modulation with memory), the inner code imposes another level of checks on all variable nodes. Hence, weight-1 nodes in the outer LDPC decoder will get extrinsic information from the inner code as the iteration progresses and their estimates will improve accordingly [73]. In this case, the first and the second nonzero λ_i's are λ_1 and λ_2. An analytical bound on λ_1' is difficult, but empirical results show that $\lambda_1' \leq 1-R$ is a reasonable assumption[5]. This is because, otherwise there are at least two degree-1 variable nodes, say the p_{th} node and the q_{th} node, connecting to the same check. When the LDPC code is considered alone, these two variable nodes are apparently wasteful and can be removed altogether. When the LDPC code is combined with the inner differential code, they result in a minimum distance of 4 for the overall codeword. As shown in Figure 48, when the four bits that are

---

it interprets it as Gaussian distributed. This discrepancy may lead to inaccuracy in the EXIT analysis.

[5]The exact code rate is dependent on the optimization result, but we know of the target code rate which is in the vicinity of the final code rate.

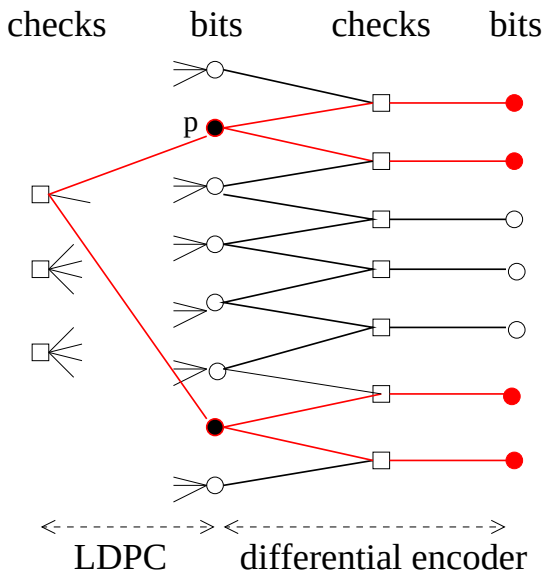


Fig. 48. Defect in code structure when $\lambda_1' > 1 - R$.

denoted by solid circles flip altogether, another valid codeword results and the decoder is unable to detect. In other words, for any finite length construction, the minimum distance of this LDPC ensemble is (at the most) 4, which is not desirable. Using the approximation $\tilde{\lambda}(x) = (1-R) + Rx$ in (3.50), we are able to compute (a lower bound of) r^* to be used in constraint (ii). Code design for differential coding is thus solvable using linear programming. Experiments show that the optimized EXIT curve has a shape as desired, but the position is slightly lower, i.e., code rate is slightly pessimistic. This can be compensated by pre-setting the control points slightly higher than we actually want them to be.

2. Optimization Results

In this subsection, we discuss some observations and findings from our optimization experiments.

First, we find that the LDPC ensemble optimal for differential encoding always contains degree-1 and degree-2 variable nodes. For high rate codes above 0.75, these nodes are dominant, or in some cases the only types of variable nodes in the degree profile. For medium rates around 0.5, there are also a good portion of high-degree variable nodes. Hence, getting back to the question how well the outer code of PA codes matches with differential encoding, it is fair to say that, at high rates, they are (near-)optimal, but at medium or low rates, they are not. Viewed from the EXIT chart, we see that at high rates, the area between the outer code of the PA code and the inner differential encoder is small (see Figure 43), leaving not much room for improvement. At rates around 0.5, however, the area is big (see Figure 49), which indicates that an optimized outer code could acquire more information rate for the same SNR threshold, or, for the same information rate, achieve a better SNR threshold.

The optimization result of the target rate 0.5 is shown in Figure 49. The resulting LDPC ensemble has code rate $R=0.5037$ and degree profile $\lambda(x) = 0.0672+0.4599x+0.0264x^8+0.0495x^9+0.0720x^{10}+0.0828x^{11}+0.0855x^{12}+0.0807x^{13}+0.0760x^{14}$ and $\rho(x) = x^5$. We see that it matches closely with an inner differential decoder operating at 0.25 dB on a $f_dT_s = 0.01$ Rayleigh fading channel using 10% of pilot symbols for noncoherent detection (Figure 49). Accounting for the rate of the outer code, we see that the resulting LDPC ensemble requires $0.25 - 10\log_{10}(0.5037) = 3.2283$ dB (asymptotically) in order for the iterative process to converge successfully. Compared to a rate 0.50 PA code which requires $1.26-10\log_{10}(0.5) = 4.2703$ dB (Figure 49), the optimized LDPC ensemble is about 1.04 dB better asymptotically. However, when the passage between the inner and outer curves is very narrow, more iterations are needed in order for the message-passing decoder to proceed to the zero-error state. This requires more computing complexity and processing time, which are the price

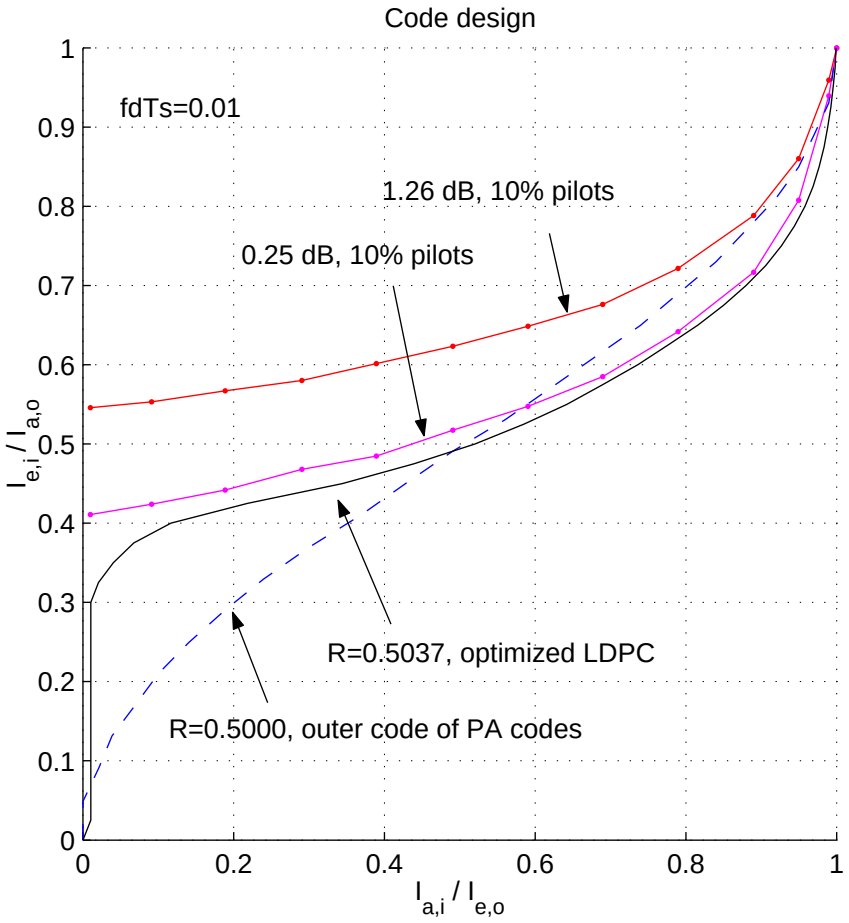


Fig. 49. EXIT chart of a rate 0.5 LDPC ensemble optimized using convergence-evolution for differential encoding. (Normalized Doppler rate 0.01, 10% of pilot symbols inserted to assist noncoherent differential detection. Degree profile of the optimized LDPC ensemble: $\rho(x) = x^5$, $\lambda(x) = 0.0672 + 0.4599x + 0.0264x^8 + 0.0495x^9 + 0.0720x^{10} + 0.0828x^{11} + 0.0855x^{12} + 0.0807x^{13} + 0.0760x^{14}$.)

we pay for stretching to the limit. Further, since the inner code is operating on a correlated fading channel using noncoherent detection, the messages it produces are approximations of the actual LLR information and, hence, the accuracy of the EXIT curve may be questionable. Nevertheless, simulation of a fairly long code shows a good agreement with the analytical result, which indicates that the design method is effective (Figure 50).

We point out that the optimized LDPC ensemble is good in the asymptotic sense, i.e., with infinite code length. In practice, we are also concerned with finite length implementation or individual code realization. From the concentration rule, we know that for long codes, all realizations perform nearly the same, and they tend to converge to the asymptotic threshold as length increases. For short block sizes, however, the variation in the code performance can be large, which means that there are good realizations and bad realizations. The reason why one realization is better than the others is a complicated problem that involves various factors that are not yet fully understood. One basic guideline for a good realization, however, is to avoid short cycles or to enlarge the girth (the length of the shortest cycle) in the code graph. In the actual construction, this can be a tedious task. A simpler practice is to focus on degree-2 variable nodes and try to eliminate (short) cycles among them. This is because experiments show that when degree-2 variable nodes form cycles among themselves, a deeply faded node would, with high possibility, flush all the edges in the cycle with erroneous messages, causing an "avalanche" effect that is difficult for the decoder to correct. Further, from the complexity point of view, if the portion of degree-1 and degree-2 variable nodes is close to or more than $1-R$ (which is like to be case for codes optimized for differential encoding), these degree-1 and 2 nodes can be lined up in a stair-case like manner to make the corresponding parity check matrix diagonal or sub-diagonal. This results in an realization that is linear time encodable

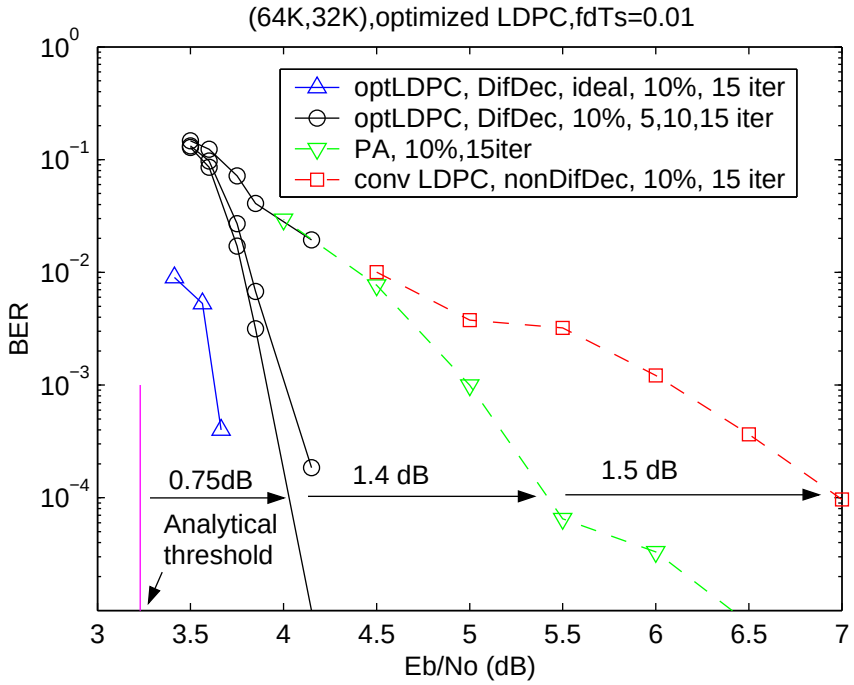


Fig. 50. Simulations of optimized LDPC code with differential encoding and iterative differential detection and decoding. (Code rate 0.5037, normalized Doppler rate 0.01, 10% pilot insertion, the degree profile of the optimized LDPC code is the same as in Figure. 50.)

using back substitution [33] [73].

Figure 50 shows the simulations of a rate 0.5037 differentially coded LDPC code with the aforementioned optimized degree profile on a correlated Rayleigh fading channel. We have chosen a long codeword length of $N = 64$K to test how well it matches with the analytical threshold. As mentioned before, a large number of iterations (like 100 iterations) are probably needed in order to fully deploy the code capacity. However, due to the limit in time and computing power, we simulated for only 15 iterations. In the figure, the leftmost curve correspond to the ideal cases (perfect knowledge on the fading phase and amplitude) with 10% pilot symbols. The circled curves correspond to the noncoherent performance using 10% of pilot

symbols and using iterative differential detection and decoding at the 5_{th}, 10_{th}, and 15_{th} iteration. For comparison purpose, we also plot the curves of noncoherently detected PA codes and conventional LDPC codes (not combined with a differential encoder) with the same parameters. The conventaional rate 0.5 LDPC code (rightmost curve) has degree profile of $\rho(x) = 0.0010x^6 + .9519x^7 + 0.0472x^8$ and $\lambda(x) = 0.2465x + 0.2306x^2 + 0.0020x^3 + 0.0465x^5 + 0.1502x^6 + 0.0353x^7 + 0.0048x^{18} + 0.2840x^{19}$, which is optimal for coherent detection (without a differential encoder) on Rayleigh fading channels [66]. We see that the performance of this differentially coded LDPC code using noncoherent detection is only about 0.3 dB worse than the coherent detection. Compared to the asymptotic threshold of 3.23 dB, the simulation performance of is about 0.75 dB away at BER of 10^{-4}, which is in plausible agreement. Further, the optimized differentially-coded LDPC code reveals an encouraging performance gain of about 1.4 dB over the PA code and 2.9 dB over the "conventionally optimal" LDPC code! This clearly shows the benefits of using differential encoding when noncoherent detection is performed, as well as the importance of designing codes matched to the specific receiver.

F. Conclusion

We have conducted a comprehensive investigation on the performance of product accumulate codes on flat Rayleigh fading channels with either coherent and noncoherent detection. Related issues concerning differential encoding and noncoherent differential detection, as well as code design to be matched with inner code/modulation are also studied. Below summarizes the major conclusions and contributions of this paper:

1. For coherent detection of PA codes, we have used Divsalar's simple bound to evaluate the performance of finite length codes and density evolution to compute

the thresholds of infinite length codes. Comparing PA codes with LDPC codes (for long lengths), we show that PA codes perform slightly better than regular LDPC codes but slightly worse than the optimized irregular LDPC codes with similar decoding complexity. Comparing PA codes with 16-state turbo codes (for short lengths), we show that PA codes are somewhat worse in performance, but are significantly lower in decoding complexity.

2. For noncoherent detection, we present and discuss a low-complexity iterative differential detection and decoding receiver. The IDDD receiver is shown to be robust for different Doppler rates and can perform within 1 dB from the coherent case on fast fading channels using little additional complexity and bandwidth. For normalized Doppler spreads of 0.01, 0.02 and 0.05, We show that as few as 2-2.5%, 5-6% and 12% pilot symbols are needed to achieve good channel estimation, respectively. We also show that the performance of noncoherent detection becomes more sensitive to pilot spacing when the channel fade rate becomes faster.

3. In terms of pilot insertion, we show that the popular practice of periodically terminating the trellis incurs an intrinsic loss in code capacity and is likely to cause high error floors and severe BER performance loss to the overall code performance. A better way of inserting pilot symbols is suggested which separates pilot symbols from the trellis structure.

4. It is well-known that coherently detected LDPC codes perform remarkably; however, little has been reported on the performance of noncoherently detected LDPC codes. This paper answers in part the above question. For noncoherent detection, a differential encoder may or may not be used. Through EXIT analysis and simulations, we show that conventional LDPC codes suffer a performance

loss when used with differential encoding and iterative detection/decoding, yet without differential encoding, they require a good number of pilot symbols in order to estimate the channel. Hence, it is fair to say that conventional LDPC codes using noncoherent detection do not perform as desirably as the coherent case (whereas PA codes perform equally well in both cases).

5. Finally, we propose a convergence-constraint method to design good LDPC ensembles matched with differential encoding/decoding (and in general any receiver). We observe that the LDPC ensemble optimal for differential encoding always contains degree-1 and 2 variable nodes, and that for high code rates, these nodes are dominant. The resulting optimal LDPC code shows a 1.04 dB gain over the existing PA code. It is worth mentioning that optimal differentially-coded LDPC codes are in fact (optimal) nonsystematic irregular repeat accumulate codes [22], but the proposed optimization procedure has a far-reaching implication and application since it can explicitly take into account the property and the imperfectness of the receiver.

We conclude by proposing product accumulate codes as a promising low-cost candidate for wireless applications. The advantages of PA codes include 1) they perform equally well with coherent and noncoherent detection (especially at high rates), 2) the performance is comparable to turbo and LDPC codes, yet PA codes require far less decoding complexity than turbo codes and far less encoding complexity and memory than random LDPC codes[6], and 3) the regular structure of PA codes makes it possible for low-cost implementation in hardware. However, we point out that the major drawback of PA codes is its relatively slow decoding convergence as compared

[6]The generator matrix of a random LDPC code is random and dense, and requires $K(N-K)$ storage space for a (N,K) code.

to turbo codes. Whereas turbo codes require 6 to 8 iterations to converge, PA codes require some 8 to 25 iterations to converge depending on channel conditions and application requirements. It would thus be a useful project to improve the convergence of PA codes. Possible solutions include to use PA-II codes rather than PA-I codes (which is what we typically refer to as PA codes) at high rates (see Chapter II) [24], and to use a set of interwoven short interleavers instead of one big interleaver.

CHAPTER IV

PRODUCT ACCUMULATE CODES FOR LONG-HAUL OPTICAL FIBER COMMUNICATIONS

A. Introduction

The wide distribution of Internet, multimedia transmission and various interactive services has generated a tremendous increase in data bandwidth to be transported through telecommunication networks, thus leading the research for high capacity and high data rate in optical fiber communications. While dense wavelength division multiplexing (DWDM) technology with 10 gigabits/second (Gb/s) channels has continued to enable multi-terabits/second (Tb/s) capacity over transoceanic distances, in oder to reduce the physical space needed at the system terminals and to make optimal use of long-haul amplifier bandwidth, data rates higher than 10 Gb/s are desirable. One of the major challenges is to effectively mitigate the fiber nonlinear effects and the increased sensitivity to dispersion and dispersion slope. Among the various technologies deployment in the fields of optical amplification, fiber, transmission bit rate and digital signal processing, forward error correction remains an important research focus. Most modern transoceanic systems use the Reed-Solomon (255,239) code, which yields about 6 dB of coding gain. However, to further increase amplifier spacing, transmission distance, data rate and system capacity for the next generation of DWDM transoceanic systems, more bandwidth- and power-efficient FEC codes are needed.

In addition to the hard decision decoding of RS codes [76] [77] [78], forward error correction codes that have been applied to or proposed for the long-haul OFC systems include concatenated RS/convolutional codes [79], concatenated RS/RS codes [80]

[81] [82], and soft-decision iterative decoding block turbo codes [80]. It shows the trend of improving the code performance by utilizing code concatenation, soft-decision decoding, and iterative decoding techniques. In particular, the breakthrough in the recent coding research has opened up possibility for potential capacity-approaching coding schemes like low density parity check codes, turbo codes, turbo product codes and product accumulate codes. These codes are able to achieve impressive coding gains at the cost of coding complexity and/or structural complexity.

Among the state-of-the-art advanced coding schemes, product accumulate codes, as a class of good high-rate codes with low-complexity[1], seem a more viable candidate than turbo or LDPC codes for potential use in optical fiber communications. As discussed in Chapter II and III, PA codes are capable of near capacity performance on AWGN channels and land-mobile Rayleigh fading channels. In this chapter we investigate the performance of soft-decision iterative-decoding PA codes based on different optical fiber channel models.

We consider optically amplified fiber communication systems using on-off keying modulation where the signal is modulated to be either 0 (also known as *space*) or an optical pulse of duration T_s (also known as *mark*). Under low-power operations, amplified spontaneous emission noise from optical amplifiers is the dominant source of noise in the system. The Chi-square model is by far the most accurate model of the ASE noise statistics, which gives closed form probability density functions for the marks and spaces after passing through a photodetector and an electrical filter [83] [84]. Since Gaussian densities can be handled more conveniently than Chi-square densities, the system model can be simplified by approximating the noise as Gaussian

[1]This is in comparison with turbo codes which require magnitude more decoding complexity than PA codes. However, compared to the hard-decoding of RS codes which can be implemented very efficiently in tap-delayed lines or linear shift register, the complexity of the soft-decoding of PA codes is still quite high.

distributed. In this work, we consider three memoryless channel models [85]: (1) asymmetric channels with uncorrelated Chi-square distributed noise, (2) asymmetric channels with Gaussian noise (an approximation to the Chi-square model), and (3) symmetric channels with Gaussian noise (i.e., AWGN) which is a further simplification and approximation of the channel model and which is widely employed in coding research.

At low signal-to-noise ratios, due to the lack of tight bounds, performance evaluation uses simulations of typical PA coding schemes. For high SNRs beyond simulation capabilities, we derive the pairwise error probability of the aforementioned channels and evaluate an average upper bound on the performance of the ensemble of PA codes. It is interesting to observe that AWGN channels, although fundamentally different from Chi-square channels, can serve as a convenient reference to approximate the performance of high-rate PA codes on Chi-square channels. In all the test cases, we assume that the decoder knows the perfect channel state information (i.e., the channel model and the relevant SNR parameters) and, hence, the soft information is set to match the particular channel model under investigation.

The rest of the chapter is organized as follows. Section B presents the three channel models under investigation. Section C discusses product accumulate codes and the iterative soft decoding that is used. Section D derives and computes the average union bounds for PA codes on different channels. Section E presents analytical and simulation results, and Section F summarizes the Chapter.

B. System Model

1. Channels with Chi-square Noise

Denote $M = B_o/B_e > 1$ as the number of modes per polarization state in the received optical spectrum, where B_o and B_e are the optical and electrical bandwidth of the system at the detector, respectively. As discussed in [84], prior to the square-law detection, the noise n_i can be mathematically represented as a Fourier series expansion with Fourier coefficients that are assumed to be independent Gaussian random variables with zero mean and variance $N_0/2$. After passing through the optical amplifier, the received signal (the integral of the output of the photodetector) over one bit interval is given by

$$x = \sum_{i=1}^{2M} (s_i + n_i)^2, \tag{4.1}$$

where s_i and n_i denote the signal and the ASE noise projected to $2M$ orthonormal basis. Signal energy is $\sum_{i=1}^{2M} s_i^2 = 2E_s$ for transmitting "1" and $\sum_{i=1}^{2M} s_i^2 = 0$ for transmitting "0", where E_s is the average energy of the transmitted signals (assuming equal probability of "1"s and "0"s).

Completing the square in the integral, the first order statistics of the optical channel (after the photodetector) can be modeled as the Chi- square distribution with $2M$ degrees of freedom [84] [83]. The closed-form probability density function of the received signal "1" and "0" after square-law detector are given by ($x > 0$) [84]

$$f_1(x) = \frac{1}{N_0}\Big(\frac{x}{2E_s}\Big)^{\frac{M-1}{2}} e^{-\frac{x+2E_s}{N_0}} I_{M-1}\Big(\frac{2\sqrt{2xE_s}}{N_0}\Big), \tag{4.2}$$

$$f_0(x) = \frac{1}{N_0}\frac{(x/N_0)^{M-1}\exp(-x/N_0)}{(M-1)!}, \tag{4.3}$$

where $I_{M-1}(\cdot)$ denotes the $(M-1)_{th}$ modified Bessel function of the first kind. The

means and variances of signal "1" and "0" can thus be derived as

$$\mu_1 = MN_0 + 2E_s, \qquad \sigma_1^2 = MN_0^2 + 4E_sN_0, \tag{4.4}$$

$$\mu_0 = MN_0, \qquad \sigma_0^2 = MN_0^2. \tag{4.5}$$

The (un-normalized) signal-to-noise ratio E_s/N_0 can be regarded as the number of signal photons at the input of the ideal high gain optical amplifier that produces the noise. However, it is conventional to use the Q factor to measure the channel condition where Q is defined as

$$Q = \frac{|\mu_1 - \mu_0|}{\sigma_1 + \sigma_0}. \tag{4.6}$$

This Q factor in dB ($20\log_{10} Q$) is sometimes referred to as the *gross* Q, since the coding overhead is not taken into consideration. Otherwise, we get the *net* Q_{net}, where $Q_{net} = Q - 10\log_{10}(R)$.

2. Asymmetric Channels with Gaussian Approximation

Observe that x is the sum of $2M$ independent random variables, the application of the central limit theorem (for large M) yields Gaussian approximation for both signal "1" and signal "0". Using the definition of Q factor as in (4.6), and normalizing N_0 in (4.2)-(4.5) to 1 , the noise parameters can then be rewritten as functions of the system parameters, B_o, B_e and Q, as [86]

$$\mu_1 = \tfrac{B_o}{B_e} + 2Q\sqrt{\tfrac{B_o}{B_e}} + 2Q^2, \qquad \sigma_1 = \sqrt{\frac{B_o}{B_e}} + 2Q, \tag{4.7}$$

$$\mu_0 = \tfrac{B_o}{B_e}, \qquad \sigma_0 = \sqrt{\frac{B_o}{B_e}}. \tag{4.8}$$

When the Chi-square distributed noise in (4.2) and (4.3) is modeled using a Gaussian approximation with the same mean and variance, we have the asymmetric

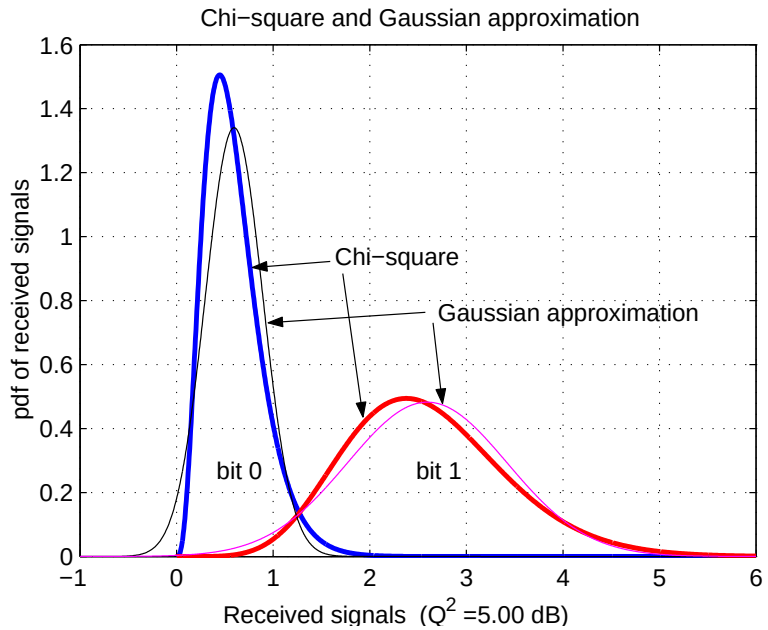


Fig. 51. Comparison of Chi-square and asymmetric Gaussian channels. (M=4, Q^2=5.0 dB.)

Gaussian channel model as follows

$$f_1(x) \quad = \mathcal{N}(\mu_1,\ \sigma_1^2) \quad = \mathcal{N}\left(\frac{B_o}{B_e} + 2Q\sqrt{\frac{B_o}{B_e}} + 2Q^2,\ \sqrt{\frac{B_o}{B_e}} + 2Q\right), \tag{4.9}$$

$$f_0(x) \quad = \mathcal{N}(\mu_0,\ \sigma_0^2) \quad = \mathcal{N}\left(\frac{B_o}{B_e},\ \sqrt{\frac{B_o}{B_e}}\right). \tag{4.10}$$

In Figure 51, we have plotted the pdf's of the received signals for the Chi-square channels and the asymmetric Gaussian channels with parameters $M = 4$ and $Q^2 = 5.00dB$. The curves give a feel of how close the Gaussian distribution has approximated the Chi-square distribution.

3. Symmetric Channels with Gaussian Approximation

Since AWGN channels are the most popular channel model in a conventional communication system, and since most of the coding research is done on this channel, we

have also included it in our study so as to facilitate comparison. Since on-off signaling is used in fiber communications rather than antipodal signaling (as is generally used in coding research), there is a 3 dB difference with the conventional results using BPSK modulation on AWGN channels.

C. Iterative Soft-Decoding for PA Codes on Optical Fiber Channel Models

For simple hard detection, an optimal threshold γ can be found numerically by letting $f_0(\gamma) = f_1(\gamma)$ in each received bit (see Figure 51). To obtain a feel of how much error correction power PA codes can provide, we consider in this work soft detection and decoding of PA codes. As discussed in Chapter II and III, the message-passing decoding has decoupled PA codes (the code structure/graph) from the channel model. For the aforementioned three channels, the initial LLR (from the channel) of a received signal x, defined as $L_{ch}(x) = \frac{\Pr(0|x)}{\Pr(1|x)}$, can be computed as (assuming equally probable "1"s and "0"s)

$$\text{Chi-square:} \quad L_{ch}(x) = \frac{(2xE_s)^{(M-1)/2}\exp(\frac{N_0}{2E_s})}{N_0^{M-1}(M-1)!\, I_{M-1}(\frac{\sqrt{8xE_s}}{N_0})}, \tag{4.11}$$

$$\text{Asymmetric Gaussian:} \quad L_{ch}(x) = \log\frac{\alpha}{\beta} - (\alpha-\beta)((\alpha+\beta)x - 2\alpha\beta^2)x, \tag{4.12}$$

$$\text{Symmetric Gaussian:} \quad L_{ch}(x) = \frac{4E_s - 4x\sqrt{E_s}}{N_0}, \tag{4.13}$$

where $\beta = \sqrt{\frac{B_0}{B_e}}$ and $\alpha = \beta + 2Q$. With this initial LLRs properly set, the iterative decoder can follow exactly the same steps as described in Chapter II and [24] to yield soft output information. We note that the symmetric Gaussian channel here uses on-off signaling instead of the conventional antipodal signaling (i.e., BPSK). A convenient alternative to (4.13) is to assume antipodal signaling with $L_{ch}(x) = \frac{4\sqrt{E_s}}{N_0}x$, and then to shift the performance curve rightward by 3 dB.

D. Analytical Bounds

1. Union Bounds

We evaluate the performance of a (N, K) PA code in optical fiber communications using the union bounds. The ensemble average bounds on the word error rate and bit error rate are given by

$$P_w \leq \sum_{h=h_{min}}^{N} \bar{A}_h P_2(h) = \sum_{h=h_{min}}^{N} \sum_{w=1}^{K} \bar{A}_{w,h} P_2(h), \tag{4.14}$$

$$P_b \leq \sum_{w=1}^{K} \frac{w}{K} \sum_{h=h_{min}}^{N} \bar{A}_{w,h} P_2(h), \tag{4.15}$$

where A_h is the output weight enumerator, $A_{w,h}$ is the input weight enumerator, and $P_2(h$ is the pair-wise error probability of the channel. The computation of the (ensemble average) IOWE of PA codes can be found in (2.56) in Chapter II.

2. Pair-Wise Error Probability $P_2(h)$

Pair-wise error probability $P_2(h)$ is in general a function of the channel characteristics, the modulation scheme and the decoding strategy. Below we derive the *average* pair-wise error probability PEP of the aforementioned channels, respectively. Average in the sense that we assume that equal probability of "1"s and "0"s are transmitted and that they are equally likely to be in error. Throughout the discussion, unless otherwise stated, we assume OOK modulation (signal energy either 0 or $2E_s$) and soft decoding.

a. Symmetric Gaussian Channel Model

For OOK signaling on symmetric Gaussian channels with noise variance $\sigma^2 = N_0^{AWGN}/2$, the average Euclidean distance of two codewords with Hamming distance h apart is

given by $\sqrt{2hE_s}$. It thus follows that the pair-wise error probability of soft decoding is

$$P_2(h) = \mathrm{Q}\left(\sqrt{\frac{hE_s}{N_0^{AWGN}}}\right), \tag{4.16}$$

where $\mathrm{Q}(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-x^2/2} dx$.

b. Asymmetric Gaussian Channel Model

With asymmetric Gaussian channels, the optimal decision threshold γ for a transmitted bit should satisfy $f_0(\gamma) = f_1(\gamma)$ in (4.9) and (4.10) [85]. To simplify computation, the threshold can be customarily set such that probabilities of bit "0" in error and bit "1" in error are the same (i.e., $P(1|0) = P(0|1)$) [84]. It is then convenient to approximate the two codewords within h Hamming distance apart as Gaussian distributed

$$f(\mathbf{c}_0) = \mathcal{N}(MN_0\sqrt{h},\, MN_0^2), \tag{4.17}$$

$$f(\mathbf{c}_h) = \mathcal{N}(MN_0\sqrt{h}+2\sqrt{h}E_s,\, MN_0^2+4E_sN_0). \tag{4.18}$$

The customary threshold γ^* for estimating codewords can be obtained by letting

$$\mathrm{Q}\left(\frac{\gamma^* - MN_0\sqrt{h}}{\sqrt{MN_0^2}}\right) = \mathrm{Q}\left(\frac{MN_0\sqrt{h} + 2\sqrt{h}E_s - \gamma^*}{\sqrt{MN_0^2 + 4E_sN_0}}\right), \tag{4.19}$$

which yields the threshold as

$$\gamma^* = MN_0\sqrt{h} + \frac{2\sqrt{h}E_s\sqrt{MN_0^2}}{\sqrt{MN_0^2} + \sqrt{MN_0^2 + 4E_sN_0}}, \tag{4.20}$$

and the corresponding pair-wise error probability $P_2(h)$ as

$$P_2(h) \approx \mathrm{Q}\left(\frac{2\sqrt{h}E_s/N_0}{\sqrt{M} + \sqrt{M + 4E_s/N_0}}\right). \tag{4.21}$$

It should be noted that we have simplified the computation of pair-wise error probability by evaluating only the typical all-zeros codeword and a weight h codeword. Although this is not exact due to the asymmetry of the channel [86], it is a reasonable approximation when the code space is linear, when all codewords are equally probable, and when the customary threshold criterion is used.

c. Chi-square Channel Model

Similar to the asymmetric Gaussian case, we use the all-zeros codeword and a weight h codeword to approximate the average pair-wise error probability $P_2(h)$ for Chi-square channels also. Unlike Gaussian distribution which is symmetric and which has a characteristic bell-shaped probability density curve, Chi-square distribution does not possess such properties to be conveniently exploited for a soft-decoding PEP. Thus we turn to hard-decoding PEP, which is an upper bound for a soft-decoding PEP.

For each noise-corrupted bit of energy 0 or $2E_s$, the receiver makes decision by comparing it with a threshold γ. The probabilities that a "1" is decided when a "0" is sent, and "0" is decided when a "1" is sent, are given by [84] [87] [88]

$$P(1|0) = \int_{\gamma}^{\infty} f_0(x)dx = e^{-\frac{\gamma}{N_0}} \sum_{k=0}^{M-1} \frac{1}{k!}(\frac{\gamma}{N_0})^k, \tag{4.22}$$

$$P(0|1) = \int_{0}^{\gamma} f_1(x)dx = 1 - \mathcal{Q}_M\Big(\sqrt{\frac{4E_s}{N_0}}, \sqrt{\frac{2\gamma}{N_0}}\Big), \tag{4.23}$$

where $\mathcal{Q}_M(a,b)$ is the generalized Marcum $\mathcal{Q}$ function of order M defined as

$$\mathcal{Q}_M(a,b) = \int_{b}^{\infty} \frac{x^M}{a^{M-1}} \exp(-\frac{x^2+a^2}{2}) I_{M-1}(ax)dx. \tag{4.24}$$

There is no simple, closed form expression for calculating the generalized Marcum $\mathcal{Q}$ function, but highly reliable and efficient numerical methods can be found in [87]

and the references therein. The optimum threshold γ can also be solved numerically by letting $f_0(x) = f_1(x)$ in (4.2) and (4.3), or

$$(2E_s\gamma)^{(M-1)/2} = e^{-2E_s/N_0} N_0^{M-1}(M-1)! I_{M-1}(2\sqrt{2E_s\gamma}/N_0). \tag{4.25}$$

Using the asymptotic expansion of $I_M(\cdot)$ reveals that the optimal normalized threshold approaches 1/4 for large $E_s/(N_0M^2)$ [83] [84]

$$\lim_{E_s/(N_0M^2)\to\infty} \frac{\gamma - MN_0}{2E_s} = \frac{1}{4}. \tag{4.26}$$

Under the assumption of equal prior probabilities, the average probability of a bit in error is given by

$$p = \frac{P(0|1) + P(1|0)}{2}. \tag{4.27}$$

Combining (4.22), (4.23) and (4.27), we get the pair-wise error probability for the Chi-square channel with two codewords of length N and Hamming distance h apart as (hard decoding)

$$P_2(h) = p^h(1-p)^{N-h} \approx p^h \tag{4.28}$$

where the approximation in (4.28) is justified for small p or large SNRs.

E. Simulation and Analytical Results

As mentioned before, in all the simulations provided, we assume perfect channel knowledge on the receiver side; in other words, there is no concern about channel mismatch. Figures 52, 53 and 54 plot the simulations of a rate 0.8, block size 16K PA code on AWGN, asymmetric Gaussian and Chi-square noise channels, respectively. We use $M = 4$ and OOK signaling. Channel conditions are measured using gross

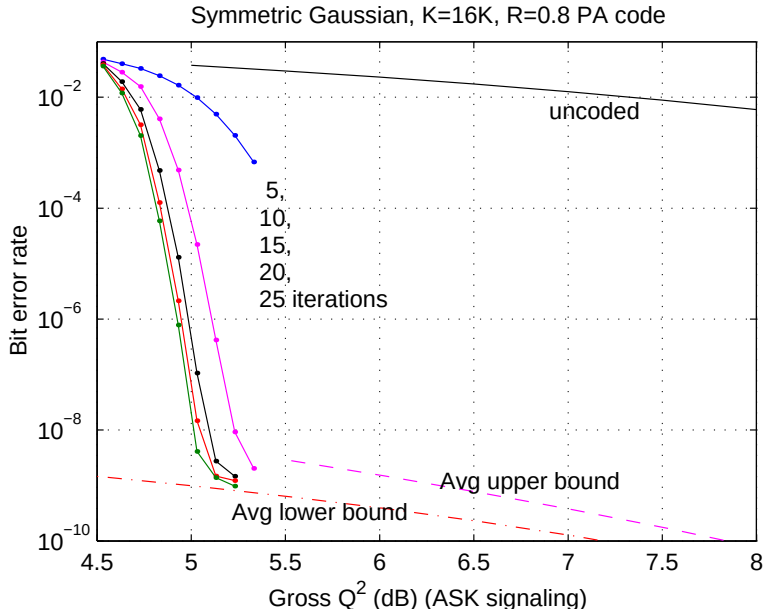


Fig. 52. Performance and bounds of high-rate PA codes on symmetric Gaussian channels with on-off signaling.

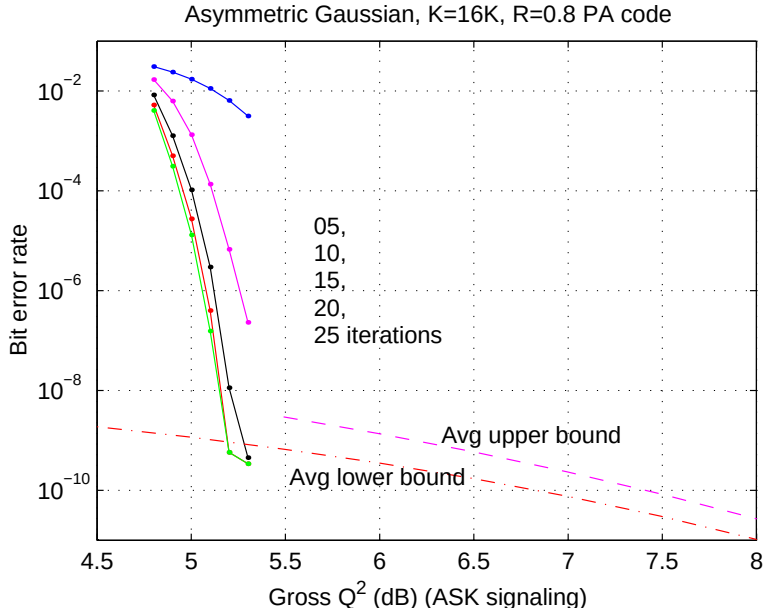


Fig. 53. Performance and bonds of high-rate PA codes on asymmetric Gaussian channels with on-off signaling.

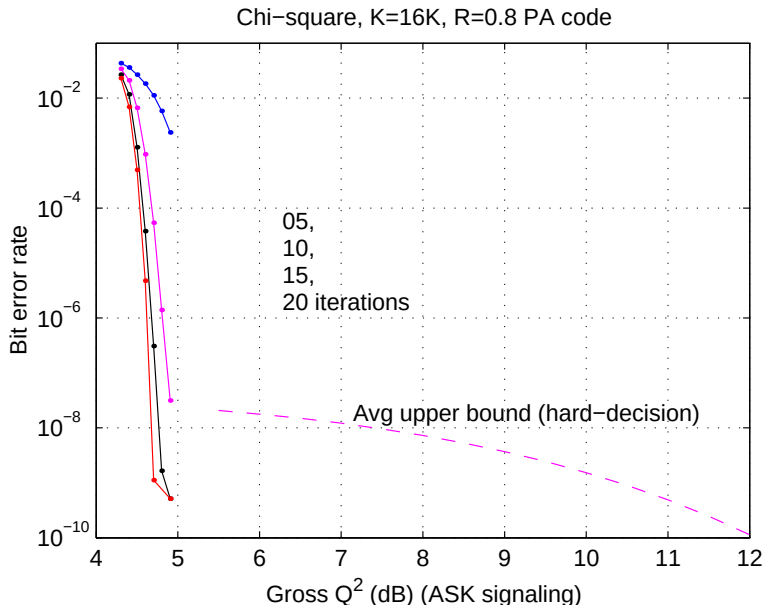


Fig. 54. Performance and bounds of high-rate PA codes on asymmetric Chi-square channels with on-off signaling.

Q^2 (in dB) as defined before. For AWGN channels, the conventional E_s/N_0 of BPSK signaling and the gross Q^2 in our simulations are off set by 3 dBs. The observations are made over 3×10^{11} bits for high SNRs, so the results are fairly reliable[2]. BER curves after 5, 10, 15, 20, 25 iterations are shown. As can be seen, PA codes yield impressive performance for all three channels, with error floors as low as BER of 10^{-9} to 10^{-10}. Comparing to the uncoded OOK systems which require Q^2 of 15 dB to achieve BER of 10^{-8} on AWGN channels, the rate 0.8 PA code can achieve as many as 9 dB gain (after the code rate penalty). Although not shown in the plots, comparison to turbo codes reveals that a 16-state turbo code with parameters $(21, 35)_{oct}$ and with the same code length and code rate at the 8_{th} iteration performs about 0.1 dB better than PA codes at the 20_{th} iteration. However, to achieve that gain, the turbo code requires

[2]Thanks to the powerful computing facility in Tyco Telecommunications (formerly known as TyCom) that we were able to run extensive simulations to benchmark the performance of PA codes.

about 12960 operations per data bit (log-domain BCJR decoding) [26], whereas the PA code requires only about 540 operations per data bit (log-domain massage-passing decoding) [24], which is less than 5% in complexity. We point out that it is a rough comparison though, since we have counted all operations which include min/max, table-lookup and addition/subtraction as of the same complexity, and have ignored the effect of interleaving. Nevertheless, how simple PA codes are as compared to turbo codes are obvious.

To facilitate the evaluation, analytical bounds are computed and presented along with the simulations. Since the bounds assume ML decoders (rather than the practical iterative decoders), and since they are averaged over all possible interleavers (thus may well be dominated by the interleavers that have the worst performance), the accuracy of the bound for a PA code with a specific interleaving scheme is questionable. In fact, the performances are seen to be slightly better than the average bounds. Nevertheless, they give useful indication on what to expect of PA codes in general for regions beyond the simulation capability.

Since Q^2 represents different channel conditions for different channel models, for a fair and accurate evaluation, we compare the performance on different channels in terms of BER-in vs BER-out. As shown in Figure 55, the performances on asymmetric Gaussian channels appear worse. It is interesting to observe that the performances on the Chi-square and AWGN channels match quite well at code rate 0.9 and show slight difference at code rate 0.8. This shows that a conventional AWGN channel can be used as a convenient reference to give indications of the performance of high-rate PA codes on a Chi-square channel, given the decoder is perfectly matched with the channel model.

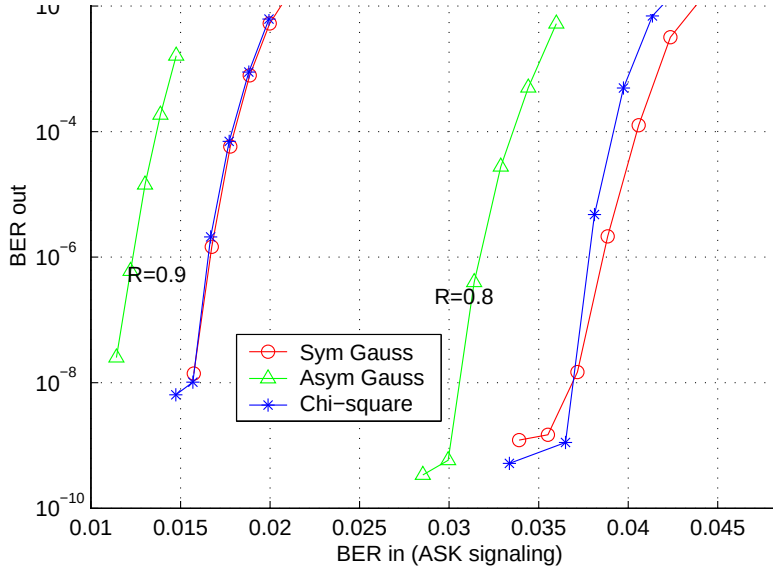


Fig. 55. Performance of high rate PA codes on different channels. (Rate $R = 0.8, 0.9$, data block size $K = 16$K, 20 turbo iterations.)

F. Summary

Product accumulate codes have been investigated with three different channel models for use in long-haul optical fiber communication systems. Extensive simulations down to quite low BERs provided benchmarks of the performance of high rate PA codes. Theoretical analysis provided insight into the regions that are beyond simulation capabilities. We have shown that a 9 dB gain over an uncoded system can be achieved for a rate 0.8, length 16K PA code, and that it can achieve essentially the same performance as turbo codes with less than 5% the decoding complexity. Hence, PA codes represent a promising prospect for error control coding for optical fiber communications, where high-rate and high-performance codes are needed and yet high complexity can not be afforded. We have also shown that the conventional AWGN channel can be used as a convenient reference to give indications of the code performance on Chi-square channels at high code rates. For future research, the rea-

son behind the observation that the PA code does not perform as well in asymmetric Gaussian channels as in Chi-square channels need to be investigated.

CHAPTER V

ITERATIVE DECODING FOR ISI CHANNELS WITH APPLICATIONS TO DIGITAL DATA STORAGE SYSTEMS*

A. Introduction

The increasing demand for vast, inexpensive and reliable data storage to satisfy the explosive growth of digitally stored information has resulted in vigorous research for ever increasing recording densities. Future generation of high-density digital magnetic recording has aimed at packing gigabits per squared inch, which requires the full technological deployment in recording media, head, servo and control, and signal processing and error correction coding. Toward satisfying this demand, in addition to technology advances in recording heads, media, and servo control, to mention a few, signal processing and coding play an important role. A digital magnetic recording channel can be modeled as a noisy, dispersive communication channel with colored noise where many of the advanced techniques in signal processing, telecommunication and information theory can be used. Magnetic recording channels present various challenges for telecommunication researchers due to many unique features, including

1. Binary input: Since the magnetic media requires that it be polarized entirely one way or the other (saturation recording), the channel input is constrained to be binary. Hence high-order modulation schemes cannot be used in magnetic recording systems to improve bandwidth efficiency.

---

[1]* © 2002 IEEE. Reprinted, with permission, from J. Li, K. R. Narayanan, E. Kurtas, and C. N. Georghiades, "On the performance of high-rate TPC/SPC codes and LDPC codes over partial response channels," *IEEE Transactions on Communications*, vol. 50, no. 5, pp. 723-734, May 2002, and J. Li, E. Kurtas, K. R. Narayanan, and C. N. Georghiades, "On the performance of turbo product codes over partial response channels," *IEEE Transactions on Magnetics*, vol. 37, no. 4, pp. 1932-1934, July 2001.

2. Heavy inter-symbol interference: There is significant amount of inter-symbol interference in the channel, especially in high-density recording systems where read-back pulses are extremely close to one another. In partial response maximum likelihood (PRML) systems, the channel is equalized to a well-chosen PR polynomial to approximate the spectrum of the channel. Non-linearities in the inter-symbol interference may require special handling of the channel equalizer. Conventional techniques like signal-space approaches and decision-feedback equalization (DFE), and the more recent approaches like turbo equalization (also known as iterative decoding and equalization), are to be exploited.

3. High code rate and short code length: Future high-density recording systems require high-rate short-length codes. The downsides of lower code rates as affecting most transmission channels include increased bandwidth and clock rates. In addition, magnetic recording channels are particularly sensitive to the code rate R in that coding overhead also leads to a substantial increase of ISI. It has been shown that the rate loss is of the order of $10\log_{10}(R^2)$ rather than $10\log_{10}(R)$ as in AWGN channels [89]. Particularly in high-density recording systems, ISI could be magnified to the point that it overwhelms any coding gain offered by the code. Further, due to the limitation on the recording sector, the code length is constrained to be no more than 512 bytes or 4096 bits.

4. Extremely low error probability: Data storage systems typically require bit error rates to be of 10^{-14} or lower. Since the majority of codes reach an error floor far above this point, such low error rates are realized by serial concatenation of two error correction codes, where the outer code is to clear up the residual errors left over by the inner. A typical implementation of the outer code is an RS code (known as RS-ECC) working on the byte level, capable of correcting

a few tens of errors, and typically interleaved to avoid error bursts.

5. Colored noise: With the existence of electronic noise, media noise and other impairments in the channel, such as timing jitter and inter-track interference from off-track reading, the noise presented in a magnetic recording channel is not white, and in fact, not even Gaussian.

6. Bursty errors: Due to thermal asperities, errors in a digital magnetic recording channel tend to come in large bursts which may easily exceed the capability of the error correction code and thus lead to loss of the entire block data.

7. Other issues to be taken care of include typical error pattern elimination, parallelization, complexity and latency, run-length limit (RLL) constraint and timing recovery, direct current (DC) balance, and physical restrictions.

The recent breakthroughs in the design of error correction codes which use code concatenation, random interleavers and iterative probabilistic decoding have had great impact on a number of applications including high-capacity digital data storage systems. Soft iterative decoding is being seriously considered for application in future digital magnetic recording systems. After being precoded, filtered and equalized to some simple partial response target, the magnetic recording channel appears much like an intersymbol interference channel to an outer code and, hence, many of the techniques used in concatenated coding systems can be adopted. In particular, the observation that an ISI channel can be effectively viewed as a rate-1 convolutional code leads to the natural format of a serial concatenated system where the ISI channel is considered as the inner code and an error correction code like an LDPC code or a punctured convolutional code acts as the outer code. With reasonable complexity, iterative decoding and equalization can be used to obtain good performance

gains (due to the interleaving gain). This is the motivation for considering iteratively soft-decodable codes.

Several researchers have shown that turbo codes based on punctured recursive systematic convolutional codes [90] [91] [92] [93] and low density parity check codes [94] [95] can provide about 4-5 dB of coding gain over uncoded partial response maximum likelihood systems at bit error rates of around 10^{-5} or 10^{-6}. However, since the actual BERs that are of interest in magnetic recording applications are of the order of 10^{-14}, and since the performance of these codes cannot be easily evaluated at such low BERs, significant coding gains cannot be guaranteed at such low BERs. Therefore, a t-error correcting Reed-Solomon error correction code (RS-ECC) is typically assumed in addition to the LDPC code or turbo code, which is to clear up the residual errors. As such, it is important to ensure that the output of the LDPC or turbo decoder will not contain more than t byte errors that may cause the RS-ECC decoder to fail.

Due to the high decoding complexity of turbo codes, current research focuses on lower complexity solutions that are easily implementable in hardware. Iterative decoding of turbo product codes, also referred to as block turbo codes [18] [19] [39] [38], and low density parity check codes in particular [9] [94] [95], seem to be potential solutions. An LDPC code exhibits similar performance to that of a turbo code, yet with considerably less decoding complexity (about 1/10 that of a turbo decoder). A randomly constructed LDPC code has quadratic encoding complexity in the length N of the code ($O(N^2)$). It has been shown in [33] that several greedy algorithms can be applied to triangulate matrices (preprocessing) to reduce encoding complexity, where the required amount of preprocessing is of order at most $N^{3/2}$. With the exception of a few LDPC codes that have cyclic or quasi-cyclic structures [34], large memory is generally required (for storage of generator and/or parity check matrices), which

could be a concern in hardware implementation. Furthermore, it has been reported that bursty errors tend to occur with LDPC codes [94] [95], which may cause failure of the outer RS-ECC code.

Single-parity check turbo product codes are a very simple class of turbo product codes, which possess many desirable properties for data storage systems, such as high-rate, linear encoding/decoding complexity and a highly parallelizable encoding/decoding process. While turbo codes and LDPC codes have been under extensive investigation for use in digital magnetic recording, little has been reported about TPC/SPC codes in this area. In this paper, we undertake a comprehensive study of the properties of high rate TPC/SPC codes and their applicability to digital magnetic recording using precoded partial response channels.

We first show that although TPC/SPC codes have a very small minimum distance, if several codewords are combined and used with an interleaver and a *precoded* PR channel, the distance spectrum improves significantly due to the interleaving gain. This makes the performance of TPC/SPC codes comparable to LDPC codes of the same rate while maintaining the advantage of a slightly lower decoding complexity and linear encoding complexity. Next, we compute the thresholds for iterative decoding of TPC/SPC codes (and LDPC and turbo codes) using density evolution [14] [12] [40] [32] [42] [11]. Finally, we study the distribution of errors at the output of the decoder (i.e., at the input to the RS-ECC decoder) and show that TPC/SPC codes have better error distribution (than that of the LDPC codes), making them more in harmony with the recording system (in the presence of an outer RS-ECC code).

We have also included (random) LDPC codes in the analysis and evaluation for two reasons. First, from the graph perspective a TPC/SPC code can be viewed as a structured LDPC code. Second, with their known powerful error correction capability, LDPC codes serve as a good benchmark in judging the performance of

a code. Through the comparison in decoding complexity, bit error rate and error statistics, we show that structured TPC/SPC codes seem a better candidate than random LDPC codes for use in future digital recording devices.

Although PR channels (or more generally ISI channels) are a widely used channel model for recording systems especially at the first stage of coding research, for practical purpose, a more realistic Lorentzian channel model with electronic circuitry noise as the dominant noise is also studied. In this situation, the performance of the code is not only a subject of code rate or block size, but is also affected by the normalized areal recording density and the equalized PR target. Several practical issues are investigated concerning the choice of the PR target, the code rate and the performance to provide a guideline for the choice of the TPC/SPC code.

The rest of the chapter is organized as follows. The system model is presented in Section B, where both the ideal PR channel model and the more realistic Lorentzian channel model are discussed. Section C analyzes the distance spectrum of the TPC/SPC-coded PR system. Section D calculates the thresholds of both TPC/SPC, LDPC systems (as well as turbo systems) using density evolution with Gaussian approximation. Certain interesting issues in the optimization of the decoding process are addressed in Section E. Section F evaluates the performance of both systems, including bit error rate and bit/byte error statistics on PR channels. Finally, Section G summarizes the chapter with a discussion of future work in this area.

B. System Model

1. PR Channel Model

In this ideal PR channel model, the channel impulse is modeled as a perfectly equalized partial response polynomial with additive white Gaussian noise. As shown in Fig-

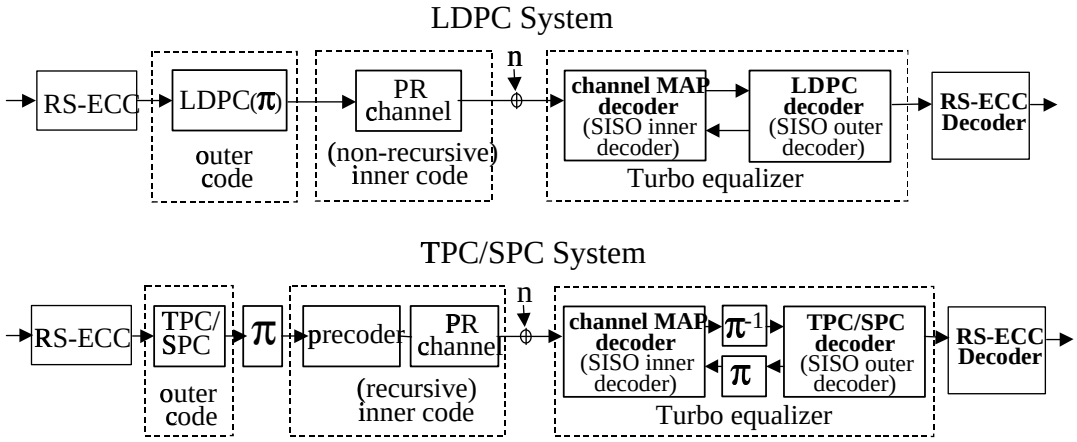


Fig. 56. System model of LDPC and TPC/SPC codes over PR channels.

ure 56, in the ideal PR channel model, the data is first encoded using a Reed-Solomon code, which is referred to as the *error correction code* or *RS-ECC*. The output of the RS-ECC code is encoded using an outer code. We consider TPC/SPC codes, LDPC codes, and punctured convolutional codes as *outer codes*. The reason for referring to these codes as outer codes is that we consider the ISI channel as the *inner code* in the concatenated scheme. When TPC/SPC codes or punctured convolutional codes are considered as the outer codes, the outer codewords are interleaved and then encoded by a rate-1 recursive precoder before being recorded onto the disk. The random interleaver in the above systems works to break the correlation among neighboring bits, to eliminate error bursts, and (in conjunction with the precoder) to improve the overall distance spectrum by mapping low-weight error events to high-weight ones (spectrum thinning). Since the LDPC codes we investigated are constructed randomly using the computer (i.e. there is an embedded random interleaver within the code), an explicit random interleaver is thus not necessary. (Although not shown, simulations show that adding a random interleaver does not improve the performance of our LDPC systems.) Further, for LDPC codes which have quite good distance spectrum, no

effective spectrum thinning results by concatenating a rate-1 inner code. As has been shown in [74], no precoding represents the best case for LDPC codes. In fact, the use of precoder results in about 1 dB loss on EPR4 channels with the suboptimal iterative decoding. The channel is modeled as an ISI channel with AWGN. The impulse response of the ISI channel is assumed to be a partial response polynomial with additive white Gaussian noise (Figure 56)

$$r_k = \sum_{i=0}^{L-1} h_i x_{k-i} + n_k. \tag{5.1}$$

In this study, we primarily consider the PR4 channel (channel polynomial $H(D) = 1 - D^2$), EPR4 channel ($H(D) = 1 + D - D^2 - D^3$), E^{2}PR4 channel ($H(D) = 5+4D-3D^2-4D^3-2D^5$) and ME^{2}PR4 channel ($H(D) = 5+4D-3D^2-4D^3-2D^5$), all of which are widely used PR targets for Magnetic recording.

Since an overall maximum likelihood decoding and equalization of the system is prohibitively complex, the practical, yet effective way, is to use turbo equalization to iterate soft outputs between the outer decoder and the equalizer, and then feed the hard decision decoding to the RS-ECC code. Illustration of message flow is shown in Figure 57.

A TPC/SPC code, as introduced in Chapter I, is a multi-dimensional turbo product code whose component codes are single parity check codes. Recall that magnetic recording systems require a high code rate since for recording systems code rate loss (in dB) is of the order of $10\log_{10}(R^2)$ rather than $10\log_{10}(R)$ as in an AWGN channel [89] (R is the code rate). Hence, we focus on 2-D TPC/SPC codes with code rate $R = (K_0/(K_0+1))^2$. As discussed in Chapter I, the encoding operation involves adding a single-parity check bit in each row and column which eliminate the need for an explicit storage of the (dense) generator matrix (as opposed to random LDPC codes) and, hence, is extremely simple. The decoding process of the 2-D TPC/SPC

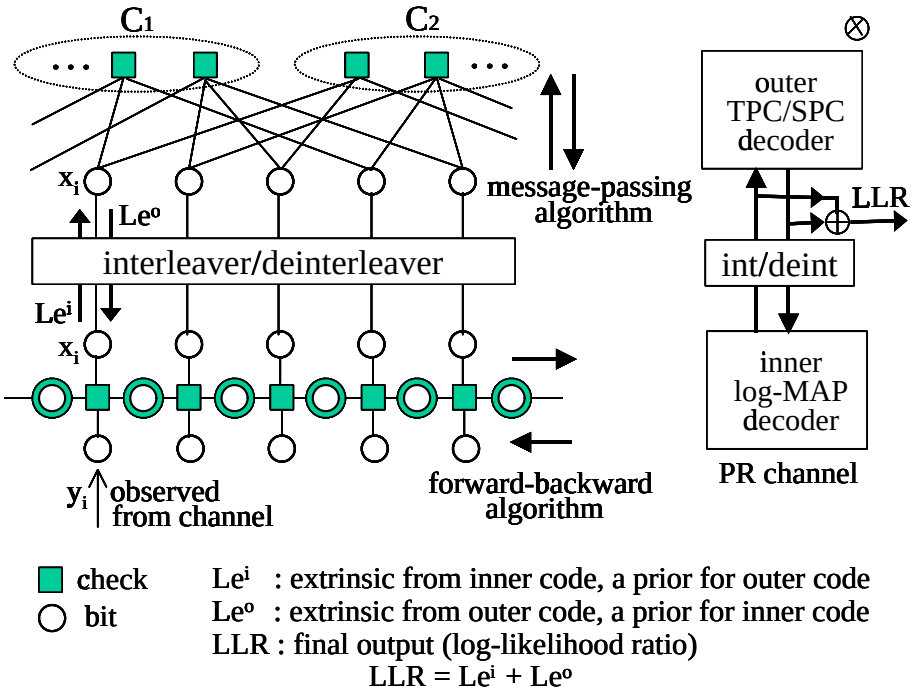


Fig. 57. Illustration of message flow in the iterative decoding of LDPC-coded PR system.

is essentially the same as that of LDPC codes except that for LDPC codes all checks are simultaneously updated, whereas for TPC/SPC codes checks are grouped into two groups (corresponding to component codes $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively, see Figure 5) and updated in turn (i.e. serially). Table IV compares the complexity of TPC/SPC, LDPC and MAP decoders implementing the BCJR algorithm in the log domain [26] (assuming that $\log(\tanh(\frac{x}{2}))$ and its reverse function $2\tanh^{-1}(e^x)$ are implemented through table lookup, and that multiplications are converted to additions in the log-domain). We can see that the decoding algorithm for a 2-D TPC/SPC code requires about 2/3 the complexity and about 1/3 the storage space of the decoding algorithm for a regular column-weight-3 LDPC code in each decoding iteration. The decoding algorithm for a punctured convolutional code is considerably higher, although the actual number of iterations needed would be lower.

Table IV. Decoding complexity of TPC/SPC- and LDPC-coded PR channels (number of operations per bit per iteration). (d: dimensions of TPC/SPC code. s: average column weight of LDPC code. m: memory size of the convolutional code.)

Operations	**TPC/SPC**	**LDPC**	**MAP for PR channel**
addition	$3d$	$4s$	$15 \cdot 2^m + 9$
min/max			$5 \cdot 2^m - 2$
table lookup	$2d$	$2s$	$5 \cdot 2^m - 2$

2. Equalized Lorentzian Channel Model

A transition sequence in magnetic recording systems is described by a non-return-to-zero inverted (NRZI) waveform, where bit "1" corresponds to a transition and "0" no transition. The response of the head to a transition in magnetization along the track is modeled as a step function or transition response with a Lorentzian pulse $s(t)$

$$s(t) = \frac{1}{1 + (2t/PW_{50})^2}, \tag{5.2}$$

where PW_{50} is the width of the pulse at 50% of its peak value. Since $s(t)$ is a response to NRZI dibit, the continuous time channel response is characterized by the dipulse

$$h(t) = \frac{1}{2}[s(t) - s(t - T)], \tag{5.3}$$

where T is the channel bit duration.

The dipulse $h(t)$ has a Fourier transform $H(v)$ with a spectral null at $v = 0$ and when normalized linear densities $D_{norm} = PW_{50}/T$ is greater than 2, most of its spectral energy is within the frequency band $[-1/2T, 1/2T]$. This DC-free spectrum and strong high-frequency attenuation validate the characterization of the channel in its discrete time domain by sampling every T seconds, ie: $h_k = h(kT)$.

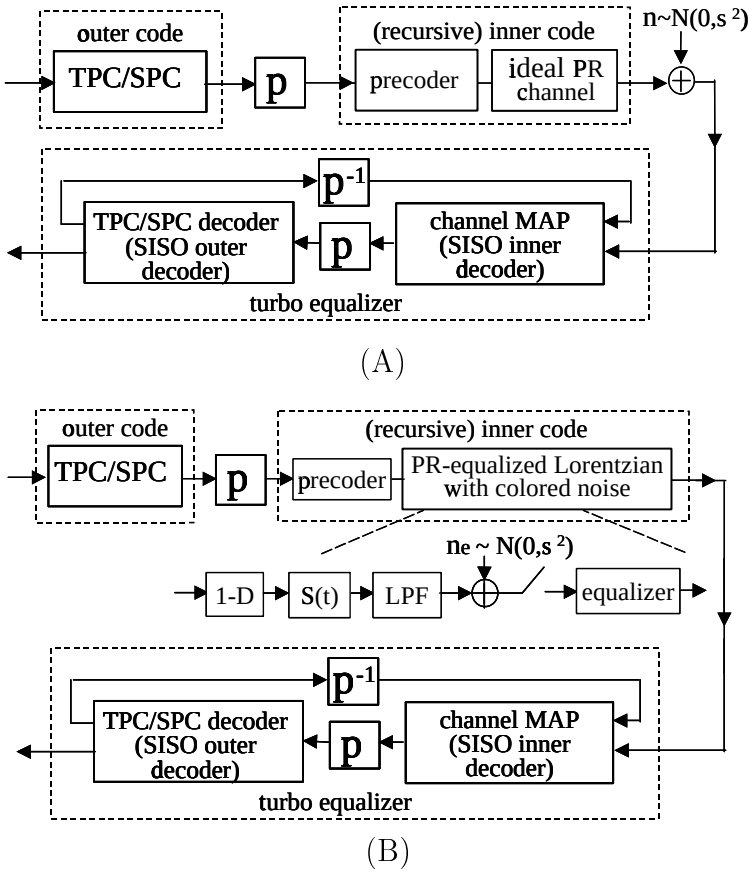


Fig. 58. System model for magnetic recording channels. (A) Ideal partial response channels. (B) PR-equalized Lorentzian channels.

We consider electronic noise caused by the head circuitry in our system (denoted by n_e in Figure 58), which is modeled as white Gaussian noise with uniform two-sided spectral density $N_0/2$. The read-back data is therefore a sampled sequence from a linear dispersive channel with additive white Gaussian noise

$$r(t) = \left(\sum_i s_i \cdot p_s(t - iT) \right) * h(t) + n_e(t), \tag{5.4}$$

where s_i denotes the binary input, $p_s(t)$ the write current pulse, $n_e(t)$ the electronic noise, and $*$ denotes the convolution operation.

An optimal receiver would consist of a whitened matched filter followed by a

Viterbi detector performing maximum-likelihood sequence estimation on the inter-symbol interference trellis. For channels with long impulse responses, the complexity of such receivers becomes formidable and thus leads to the development of suboptimal receivers with substantially lower complexity.

As a practical treatment in the magnetic recording systems, a linear equalizer, which is a finite impulse response filter, is adopted to first shape the channel response to a desired target of acceptably short duration and with amplitude-frequency characteristics closely matched to those of the channel. A Viterbi detector with greatly reduced states is then applied to decode ISI. Reasonable PR targets take the form of $H(D) = (1-D)P(D)$, where $P(D) = 1 + p_1 D + p_2 D^2 + \cdots + p_L D^L$. Although it is expected that a generalized PR monic polynomial with real coefficients has a more accurate match of the recording channel with less noise enhancement, in the correlative coding literature, the coefficients are usually restricted to integer values with a greatest common divisor equal to one. Some popular targets are the partial response class IV family of the form $H(D) = (1-D)(1-D)^L$, where $L = 1$ is called PR4, $L = 2$ EPR4, $L = 3$ E^{2}PR4, etc.

In this work, we adopt an adaptive FIR with blind start up using least means square (LMS) training as the front-end equalizer to shape the channel. When the read-back sequence sampled every T seconds, $r_k = r(kT)$, is passed through this FIR equalizer, correlation is introduced and the white electronic noise becomes colored. Since the number of the coefficients is limited and the training sequence is also limited, the output channel response shaped by this FIR equalizer is not perfectly matched to the PR targets as in the ideal model, but this more realistic model allows us to capture the effect of recording density dependencies, imperfect channel shaping as well as noise coloring.

Although the successful introduction of the Viterbi algorithm for decoding ISI

(i.e. PRML), among other factors, has led to great improvements in areal density for the past decades, the current trend towards a further increase in storage densities and reliabilities indicates an insufficiency of the use of PRML, for its hard outputs fail to take advantage of the newly advancement in coding techniques such as soft decoding and iterative approaches. Hence in this study, we use SISO MAP detector implementing BCJR algorithm instead of Viterbi detector to decode the inter-symbol interference. The MAP decoder is designed to match to the ideal channel PR target as in the previous model, which will enable soft information to iterate between the channel decoder and the outer TPC/SPC decoder.

Here are a few comments on this model: (1) In our systems, we do not take into account the material dependent media noise, interference from adjacent tracks, off-tack head registration and other non-linear factors. We consider only the electronic noise, which is dominant in magnetic recoding systems; (2) In a practical system, modulation constraints should be complied to ensure the sequence of bits recorded satisfies certain restrictions for the purpose of timing recovery and synchronization. This is usually done by a run-length limit code. Due to the SISO decoding feature and the random interleaver in between, the conventional constraint coding structure, in particular, the interface between the RLL code and the error correction code, is not immediately applicable. This paper will not address modulation constraints, rather, readers are directed to [96] [97] and the references therein, where new system configurations are proposed for coping with the interleaved serial concatenated architecture.

In summary, with this Lorentzian channel model, we will investigate the performance of TPC/SPC codes where LMS FIR equalization and iterative decoding of code and channel are employed. Such factors as the channel recording density effects, the FIR equalizer length effects, the equalized PR target pattern, and the noise coloring caused by the equalizer are to be evaluated. For a comprehensive performance

evaluation, not only the bit error rate, but the bit/byte error statistics will be examined to assess the overall error rate (or block failure rate) after the functioning of the out-most RS-ECC code (RS-ECC is not shown in the system model).

C. Analysis of Distance Spectrum

The minimum distance of a randomly-constructed regular LDPC code with column weight $j \geq 3$, depends on the actual construction and is hard to determine, but with high probability increases linearly with block length N, especially for large N. Hence it possesses good error detection capability and the decoding algorithm rarely converges to a wrong codeword. On the other hand, the distance spectrum of a TPC/SPC code is characterizable (see (2.1) in Chapter II). Since a 2-D TPC/SPC code has many codeword pairs that have minimum distance of 4, it therefore encounters many undetectable errors. In particular, all rectangular error patterns are undetectable. Therefore, TPC/SPC codes by themselves are quite weak in error correction capability, compared to LDPC codes of the same rate. To enhance the performance, instead of increasing the dimensionality and/or adding more parities [45] which will incur undesirable rate loss, we group P TPC/SPC codewords together and interleave them before encoding by the precoder. Recall that this is the same philosophy that leads to the invention of product accumulate codes (see Chapter II). As expected, this will lead to significant improvement in distance spectrum and performance without sacrificing code rate. The spectrum analysis of such a system[1] has much similarity with that of the PA codes. However, since an ISI channel is real-valued output and since different input sequences result in different output Euclidean distance spaces (i.e. the

[1]As a clarification of our notation, we use "TPC/SPC codes" to mean plain TPC/SPC codes (w.r.t. AWGN channels), and use "TPC/SPC systems" to mean the combination of TPC/SPC codes and PR channels (which forms a serial concatenated code). Similar terms are used for "LDPC codes" and "LDPC systems".

all-zero sequence cannot be used as a reference for ISI channels), the evaluation of its distance spectrum is nevertheless worth a separate discussion.

Below we compute the distance spectrum of a TPC/SPC system with a precoded PR4 channel using the ideas in [25] [98] [99]. Since the precoder is a rate-1 recursive convolutional code, the combination of the ISI channel and the precoder is a recursive ISI channel. One approach is to consider the overall system as the concatenation of the outer code and the precoded ISI channel (which acts as a recursive inner code). Then, we can compute the distance spectrum of such a system over the ensemble of all possible interleavers such as in [25] [98]. We show that caution should be exercised in extrapolating the results of Benedetto *et al.*, since the results are somewhat unexpected. Hence, it is worth pursuing this exercise.

Let N denote the length of each codeword (effective block size) formed by grouping P TPC/SPC codewords of length $(N/P) = (K_0 + 1)^2$ each and interleaving them. This length-N codeword is then passed through a precoded PR channel. Each TPC/SPC code has $\sqrt{N/P}$ rows and columns. Let A_l^o denote the number of outer codewords (TPC/SPC) of output Hamming weight l, and A_{l,d_E}^i denote the number of inner codewords (precoded PR channel) of input Hamming weight l and output Euclidean weight d_E. Assuming a uniform interleaver, the average number of codewords of Euclidean weight d_E, $A_{d_E}^c$, over the ensemble of interleavers is

$$A_{d_E}^c = \sum_{l \geq 4,\ l\ even} \frac{A_l^o \times A_{l,d_E}^i}{\binom{N}{l}}. \tag{5.5}$$

The lower limit for the sum is $l = 4$ because the minimum distance of the TPC/SPC code is 4 and only even terms are considered since all codewords of the TPC/SPC are

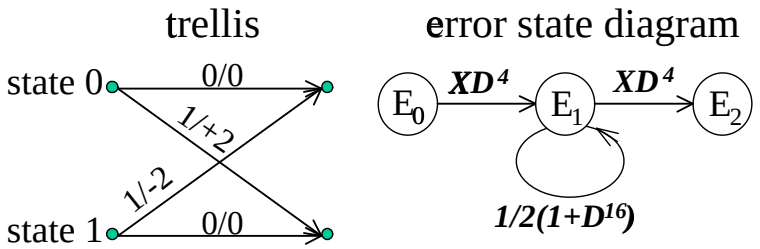


Fig. 59. Equivalent trellis for even/odd bits of precoded PR4 channels ($\frac{1-D^2}{1\oplus D^2}$).

of even weight. The average word error rate is upper-bounded by the union bound

$$P_w \leq \sum_{d_E} A^c_{d_E} \cdot Q\left(\frac{d_E}{2\sigma}\right), \tag{5.6}$$

where σ^2 is the variance of the noise. To argue that TPC/SPC codes are capable of interleaving gain on precoded PR channels, we need to show that $A^c_{d_E}$ for small d_E decreases with an increase in interleaver size, which in turn provides a reduction in error rate. Since a precoded PR channel is in general non-linear, the all-zeros codeword cannot be treated as the reference codeword. However, a full accounting of error events pertaining to A^i_{l,d_E} is prohibitively complex for an exact analysis. To simplify this, Öberg and Siegel have made the assumption that the input to the precoded channel is an i.i.d. (independent and identically distributed) sequence [98]. This assumption makes it easier to compute the transfer function of the precoded channel, since an i.i.d. sequence of zeros and ones can be treated as the reference sequence. In the following, we use this assumption to analyze the distance spectrum of the combination of the TPC/SPC outer code and the precoded channel.

Let us consider a precoded PR4 channel as an example. The equivalent trellis corresponding to odd/even bits of the precoded PR4 channel ($\frac{1-D^2}{1\oplus D^2}$) is shown in Figure 59. Following similar derivations as in [99], the average error enumerating

function, where the average is taken over all possible input sequences, is given by

$$\begin{aligned} T(X,D) &= \frac{X^2D^8}{1-\frac{1}{2}(1+D^{16})}, && (5.7)\\ &= X^2D^8\big[1+\frac{1}{2}(1+D^{16})+\cdots+\frac{1}{2^k}(1+D^{16})^k+\cdots\big], && (5.8)\\ &= X^2D^8\big[(1+\frac{1}{2}+\frac{1}{2^2}+\cdots)+D^{16}(\frac{1}{2}+\frac{2}{2^2}+\frac{3}{2^3}+\cdots)+O(D^{32})\big], && (5.9)\\ &= X^2D^8\big[2+2D^{16}+O(D^{32})\big]. && (5.10) \end{aligned}$$

where the exponent of X is the input Hamming weight of the error sequence, and the exponent of D is the output squared Euclidean distance of the error sequence. The fractional terms in the branch weight enumerator such as $1/2(1+D^{16})$ (Figure 59) are a direct consequence of the assumption that the input corresponding to that branch can be a 0 or 1 with equal probability 1/2 [99].

For the precoded PR4 channel, the independent (i.e., unconcatenated) input error sequence always has input weight 2. This can be seen from the transfer function since every term corresponds to X^2. Specifically, all input error sequences of the form $1+D^{2j}$ result in an error event. The minimum Euclidean distance over all such error events occurs when $j=1$ and the minimum Euclidean distance is 8 (assuming i.i.d inputs). Every finite weight codeword is the concatenation of k weight-2 input error events for some k. For large N, let $T_N(X^{2k},D)$ denote the truncated weight enumerator truncated to length N, where each error event is the result of k input error sequences each of weight-2. Then,

$$T_N(X^{2k},D) \propto \binom{N}{k} X^{2k}D^{8k}\left[2+2D^{16}+O(D^{32})\right]^k, \qquad (5.11)$$

since there are approximately $\binom{N}{k}$ ways to arrange k error events in a block of length N. For the least non-zero l in the TPC/SPC system, namely $l=4$ (i.e., $k=2$ in (5.11)), we see that $A^i_{l=4,d_E=4} \approx 4\binom{N}{2}$, and $A^o_{l=4} \approx P[\binom{\sqrt{N/P}}{2}]^2$, (there are $[\binom{\sqrt{N/P}}{2}]^2$

ways in which we can arrange a block of weight 4 within a TPC/SPC and there are P blocks in a codeword of length N.) Substituting them into (5.5) and using the approximation $\binom{N}{n} \approx N^n/n!$ for large N, we have

$$A^c_{d_E=4} \propto \frac{\frac{N^2}{P} \cdot N^2}{N^4} \propto P^{-1}. \tag{5.12}$$

It should be observed from (5.12) that the reduction in word error rate is proportional to the number of blocks P of the TPC/SPC that form a codeword, rather than N, as would be expected from Benedetto *et al.*'s analysis [25]. This is especially important for finite block lengths, since this means that an interleaving gain is limited to the number of codewords of the outer code that are concatenated. Although we have only discussed the error event corresponding to the least non-zero l (i.e., $l = 4$), it can be shown that for other values of small l, similar arguments hold. Similar results can be shown for EPR4 or other ISI channels also. To handle ISI channels with more number of states, it is convenient to consider the precoder separately from the channel. That is, we treat the concatenation of the TPC/SPC and the precoder as a code whose codewords are passed through the ISI channel. Since the interleaving gain is dependent only on the recursive nature of the inner code, an interleaving gain will result regardless of the type of ISI channel. This idea will be further addressed later for the optimization of TPC/SPC systems.

It is important to note that the fact that the least non-zero l is 4 (i.e., $d_{min} = 4$ for the outer code) is crucial to the result in (5.12). It is shown by Benedetto *et al.* [25] that the outer code should have a d_{min} of at least 3 in order to obtain an interleaving gain in the word error rate. *The key advantage of TPC/SPC codes is that for any rate and any codeword length, $d_{min} = 4$, which enables an interleaving gain.* On the contrary, for punctured convolutional codes of high rate (0.9 or higher, such as what is of interest), the constraint length of the code must be very large to

obtain a minimum distance of 3 or higher. For example, even a 16-state punctured convolutional code with generator polynomials $(31, 33)_8$ of rate 0.9 has a d_{min} of only 2. The obvious disadvantage is that the decoding complexity increases exponentially with the constraint length. Therefore, TPC/SPC codes are a computationally efficient choice for constructing a good class of high-rate outer codes which guarantee an interleaving gain.

D. Threshold Analysis Using Density Evolution

1. Introduction to Density Evolution and Gaussian Approximation

Although distance spectrum analysis shows that TPC/SPC codes concatenated with precoded ISI channels possess good distance spectra, the analysis is useful only if a maximum likelihood decoder is used. For the analysis to be more precise and convincing, we extend density evolution to compute the thresholds of the coded- PR systems. The idea of general method of density evolution was discussed in Chapter II as well as in [14] [12] [40] [32] [42] [11]. Here, we go through the critical points in the application of density evolution to TPC/SPC and LDPC systems [42]. For comparison purposes, we extend it to include serial turbo systems (with punctured convolutional codes) also.

2. Problem Formulation

The systems under investigation have a unified architecture in that the (precoded) PR channel is modeled as an inner rate-1 convolutional code, with the outer code being an LDPC code, a TPC/SPC code or a (punctured) convolutional code. A turbo equalizer is used to iterate messages between the inner and outer decoders. During the q_{th} iteration, the outer decoder generates extrinsic information on the j_{th} coded bit

a_j, denoted by $L_o^{(q)}(a_j)$, and passes it to the inner decoder. The inner MAP decoder then uses this extrinsic information (treated as *a priori*) with the received signal and generates extrinsic information, $L_i^{(q+1)}(a_j)$. The extrinsic information $L_o^{(q)}(a_j)$ is a random variable and, for an infinite block size, the random variables $L_o^{(q)}(a_j)$ are i.i.d. $\forall j$. The idea in density evolution is to examine the probability density function of $L_o^{(q)}(a_j)$ during the q_{th} iteration, denoted by $f_{L_o{}^{(q)}}(x)$, and to compute the threshold such that

$$\eta = \inf_{SNR} \left\{ SNR : \lim_{q\to\infty} \lim_{N\to\infty} \int_{-\infty}^{0} f_{L_o{}^{(q)}}(x|\bar{\mathbf{y}}_N, SNR)) \to 0 \right\}, \tag{5.13}$$

where $\bar{\mathbf{y}}_N$ denotes the observed sequence of length N, superscript (q) denotes the q_{th} iteration, and subscripts i and o denote quantities pertaining to the inner and outer code, respectively.

Since it is quite difficult to analytically evaluate $f_{L_o^{(q)}}(x)$ for all q, we simplify the computation by approximating $f_{L_o^{(q)}}(x)$ to be Gaussian of mean m_o^q and variance $2m_o^{(q)}$. Under i.i.d. and Gaussian assumptions, the mean of the messages, $m_o^{(q)}$ then serves as the sufficient statistic of the message density. The problem thus reduces to

$$\eta = \inf_{SNR} \left\{ SNR : \lim_{q\to\infty} \lim_{N\to\infty} m_o^{(q)} \to \infty \right\}. \tag{5.14}$$

3. Message Flow Within the Channel MAP Decoder

To evaluate the concatenated systems using density evolution, we need to examine the message flow within the outer decoder, the inner decoder as well as in between the two. Specifically, we need to evaluate $m_o^{(q)}$ as a function of $m_i^{(q)}$ and vice-versa. For the inner MAP decoder (equalizer), since it is not straight-forward to derive $m_i^{(q+1)}$ as a function of $m_o^{(q)}$, Monte Carlo simulations are used to simulate the behavior of

the MAP decoder and determine a relationship between $m_i^{(q+1)}$ and $m_o^{(q)}$, denoted by

$$m_i^{(q+1)} = \gamma_i(m_o^{(q)}). \tag{5.15}$$

The mean of the message $m_i^{(q+1)}$ is evaluated at the output of the inner MAP decoder given the input *a priori* information is i.i.d. and Gaussian with mean $m_o^{(q)}$ and variance $2m_o^{(q)}$. Since ISI channels are generally non-linear, the input sequence is not assumed to be all zeros, rather a sequence of i.i.d. bits. Detailed description and figures of Monte Carlo simulation technique for computing γ_i can be found in [74].

4. Message Flow Within the Outer Code

This section describes how to compute $m_o^{(q)}$ as a function of $m_i^{(q)}$ for different outer codes.

a. LDPC Codes

The LDPC decoder itself is an iterative decoder which uses L iterations to update extrinsic information passed between bits and checks. Since turbo equalization is also an iterative process, we use superscript (q) and (l) to denote quantities during the q_{th} iteration of turbo equalization (outer loop) and l_{th} iteration within the LDPC decoder (local loop). Let $\mathcal{E}(b_k)$ and $\mathcal{E}(c_j)$ denote the set of all checks connected to bit b_k, and the set of all bits connected to check c_j, respectively, in the LDPC code. Assuming regular LDPC codes with $|\mathcal{E}(b_k)| = t,\ \forall k$, and $|\mathcal{E}(c_j)| = s,\ \forall j$, we have a code rate $R = 1 - t/s$. Message flow on the code graph is a two-way procedure, namely, bit updates and check updates, which correspond to the summation in the real domain and the so-called check-sum operation or *tanh* rule [32] [40], [42]. After L local iterations of message exchange, the message passed over to the inner MAP decoder is the LLR of the bit in the L_{th} iteration after subtracting $L_i^{(q)}$ which was

obtained from the inner code and was used as *a priori* information.

Under the Gaussian assumption, we are interested in tracking the means of $L_b^{(q,l)}$ and $L_c^{(q,l)}$ given by $m_b^{(q,l)}$ and $m_c^{(q,l)}$, respectively. Treating extrinsic information as independent, the means of the extrinsic information at each iteration can be shown to be [32]

$$\text{bit-to-check:} \qquad m_b^{(q,l)} = m_o^q + (s-1) \cdot m_c^{(q,l-1)}, \tag{5.16}$$

$$\text{check-to-bit:} \qquad m_c^{(q,l)} = \psi^{-1}\left(\left[\psi(m_b^{(q,l-1)})\right]^{t-1}\right), \tag{5.17}$$

$$\text{LDPC-to-MAP:} \qquad m_o^{(q)} = s \cdot m_c^{(q,L)}, \tag{5.18}$$

where $\psi(x)$ is the expected value of $\tanh(\frac{u}{2})$, and u follows a Gaussian distribution with mean x and variance $2x$. $\psi(x)$ is given by

$$\psi(x) = \begin{cases} \frac{1}{\sqrt{4\pi x}} \int_{-\infty}^{\infty} \tanh(\frac{u}{2}) e^{-\frac{(u-x)^2}{4x}} du, & x > 0, \\ 0, & x = 0. \end{cases} \tag{5.19}$$

Function $\psi(x)$ is continuous and monotonically increasing on $[0, \infty)$ with $\psi(0) = 0$ and $\psi(\infty) = 1$. The initial condition is $m_b^{(q,0)} = m_c^{(q,0)} = 0$. When x is large (corresponding to low error probability), $(1-\psi(x))$ is shown to be proportional to the error probability [32]. The above derivation is essentially an extension of Chung *et al*'s work [32] to the case of turbo equalization. For more detailed and thorough understanding, readers are directed to [14] [12] [40] [32] [11] and the references therein. For the turbo equalization case, after q (big) iterations between the outer and inner decoder (where each big iteration includes L local iterations within the LDPC decoder), the capacity is evaluated as

$$\eta_{LDPC} = \inf_{SNR} \left\{ SNR : \lim_{q \to \infty} s \cdot m_c^{(q,L)} \to \infty \right\}. \tag{5.20}$$

It is instructive to note that L is to be carefully chosen, since it affects the capacity

of the resulting code[2]. Of particular practical interest is to find the best trade-off between the resulting threshold and complexity, as will be addressed in a later section.

b. TPC/SPC Codes

Although a TPC/SPC code can be viewed as a special type of LDPC code, the DE procedure cannot be applied directly. This is because density evolution assumes that there are no cycles in the code graph. For TPC/SPC codes, even as the length of the code becomes very large, there are always cycles of length $4(k+1)$, where k is any integer. As mentioned before, this is due to the fact that a rectangular error pattern always results in a loop of length 8. Consequently, the assumption that messages being passed within the code are independent (loop-free operation) is no longer valid.

For this reason, we propose and discuss a slightly modified procedure. If the number of local iterations within the TPC/SPC code is restricted to be small, then, the density evolution method would have operated on cycle-free subgraphs of TPC/SPC codes. Put another way, the messages exchanged within TPC/SPC codes along each step are statistically independent as long as the cycles have not "closed". Here, we restrict the number of local iterations within TPC/SPC codes to be one row update and one column update. Any more updates in either direction will either pass information to its source or pass duplicate information to the same node, which is unacceptable. On the other side, due to the (perfect) random interleaver, infinite number of turbo iterations can be performed between the inner and outer decoders if the messages within the outer TPC/SPC code are reset to zero in every new turbo iteration. In order to improve the convergence of the decoding algorithm, we consider

[2]The decoding strategy is considered part of the "code", since different decoding parameters lead to varying performance.

Table V. Summary of density evolution procedure for (conventional) TPC/SPC codes (upper bound).

Initialization:

$m_o^{(0)} = 0;$

Density Evolution:

for $q = 1, 2, \cdots$

compute $m_i^{(q)} = \gamma_i(\bar{\mathbf{r}}, m_o^{(q-1)});$

$m_{c_2}^{(q)} = 0;$

row-wise: bit to check: $m_{b_1}^{(q)} = m_i^{(q)};$

check to bit: $m_{c_1}^{(q)} = \psi^{-1}(\,[\psi(m_{b_1}^{(q)})]^{K_1});$

col-wise: bit to check: $m_{b_2}^{(q)} = m_i^{(q)} + m_{c_1}^{(q)};$

check to bit: $m_{c_2}^{(q)} = \psi^{-1}(\,[\psi(m_{b_2}^{(q)})]^{K_2});$

$m_o^{(q)} = m_{c_1}^{(q)} + m_{c_2}^{(q)};$

end;

Target:

$$\eta_{TPC/SPC} = \inf_{SNR}\,\{SNR : \lim_{q\to\infty} m_o^{(q)} \to \infty\}$$
$$= \inf_{SNR}\,\{SNR : \lim_{q\to\infty} m_{c_1}^{(q)} + m_{c_2}^{(q)} \to \infty\}$$

a serial update - that is, the row update and the column update are not performed simultaneously. Rather, the row update is performed first and the extrinsic information from the row checks is passed to bits and later used in the column updates. The resulting procedure to compute the densities can then be summarized as in Table V.

For an exact threshold, the density evolution procedure should, in addition to avoiding looping messages, also ensure completeness in the sense that every bit should have utilized all the messages (through dependencies) from all the checks. The pro-

cedure discussed in the previous paragraph and tabulated in Table V, although stemming naturally from the decoding procedure, is unfortunately not complete. This is because only one row update followed by one column update is performed, which is not sufficient to exploit the information from all the checks [40]. Hence, the resulting threshold is an upper bound[3].

c. Serial Turbo Systems

In the serial turbo system, the outer code is a punctured convolutional code with a moderate constraint length. Treating it much the same way we treat the inner convolutional code (the PR channel), a MAP decoder implementing the BCJR algorithm is used and the same Monte Carlo method is adopted to track the mean of the extrinsic information (of the outer code), $m_o^{(q)}$, during the q_{th} iteration. The capacity is computed using

$$\eta_{serial} = \inf_{SNR} \left\{ SNR : \lim_{q \to \infty} m_o^{(q)} \to \infty \right\}. \tag{5.21}$$

5. Thresholds

The upper bound on the threshold for TPC/SPC codes, and the thresholds for LDPC codes and punctured convolutional codes are shown in Figure 60 for PR4 and EPR4 channels. We consider regular LDPC codes with column weight 3, since regular LDPC codes have slight advantage over irregular LDPC codes for short block sizes and high rates as in data storage applications [35]. It can be seen that the *upper bound* for TPC/SPC codes is about 0.5 dB away from that of LDPC codes for a code rate of

[3]By upper bound, we mean that the exact thresholds of TPC/SPC system should be better than this. In other words, for a given dB, the achievable code rate (bandwidth efficiency) could be higher, or equivalently, for a given rate, the required SNR could be smaller.

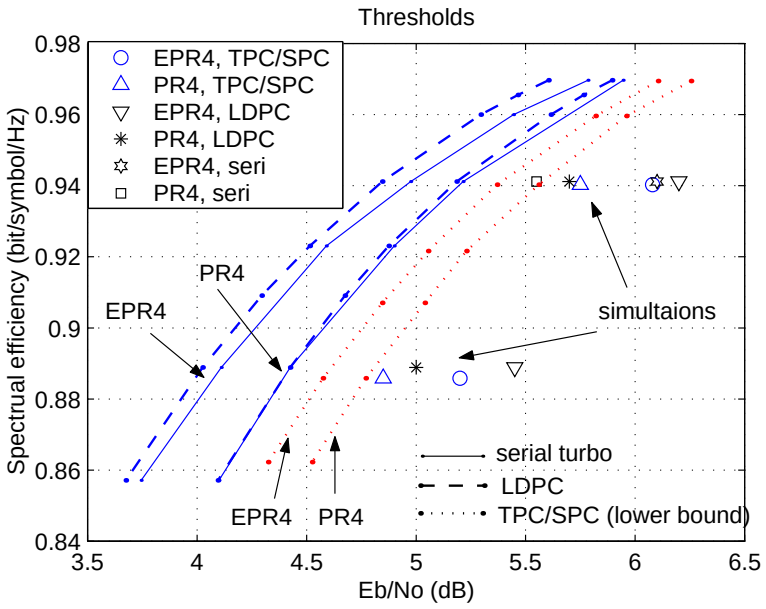


Fig. 60. Thresholds of TPC/SPC, LDPC and serial turbo systems over ideal PR channels. (The outer code in the serial turbo system is a systematic recursive convolutional code with generating polynomial $(31, 33)_{oct}$, and the LDPC code is regular with column weight 3.)

0.94. This shows that the performance of TPC/SPC is expected to be within a few tenths of a dB from that of LDPC codes. Further, the thresholds for LDPC codes are comparable to those of a serial concatenated code with a 16-state convolutional code. Since the decoding complexity of LDPC codes and TPC/SPC codes is significantly lower than that of 16-state convolutional codes for high rates, there seems to be little advantage in using punctured convolutional codes.

Also presented are the corresponding simulation results that are evaluated at a BER of 10^{-5}, with block size of 4K user data bits. It can be seen that for practical block sizes the performance of TPC/SPC codes is actually slightly better than that of LDPC codes for EPR4 channels and is comparable to that of LDPC codes for PR4 channels. Due to the finite block size, the simulations are around 0.5 to 1 dB away from the bounds. Nevertheless, this presents a reasonable match and indicates

density evolution as a useful tool in the threshold analysis of LDPC codes, TPC/SPC codes and serial concatenated codes for PR channels.

E. Optimization of Decoding Process

1. LDPC Systems

Each turbo iteration (outer loop) involves a pass of forward-backward decoding of the inner MAP decoder (BCJR algorithm) followed by L rounds of bit-check/check-bit updates (small loop) of the LDPC decoder. As mentioned above, with the assumption of an infinite block size and a perfect random interleaver, the girth (shortest cycles) of a LDPC code is unbounded, and thus L can be infinitely large. Perceivably, the resulting thresholds are non-decreasing with L, but overly large L is computationally inefficient. Hence it would be of practical interest to investigate how the value of L affects performance, and in particular, to find an optimal balancing point where best performance is achieved at the least decoding complexity. This can be done by calculating the thresholds of LDPC systems using density evolution with different values for L. We examined a rate-16/17 and a rate-8/9 regular LDPC code (column weight 3) over PR4 and EPR4 channels respectively. As shown in Figure 61, increasing L beyond a point brings only marginal improvement in the thresholds. Further, it is interesting to observe that the optimal value of L is slightly different with different channel coefficients. Whereas $L = 4$ or 5 seems a good trade-off on EPR4 channels, $L = 7$ to 8 seems better for PR4 channels. Extensive simulation experiments show that somewhere around 5 to 8 seems a good choice for L, corroborating this result. It is also worth mentioning that the above results are for the LDPC code ensemble where the column weights are uniformly 3 and the row weights follow the concentration rule (as uniform as possible). The optimal value of L might differ slightly for

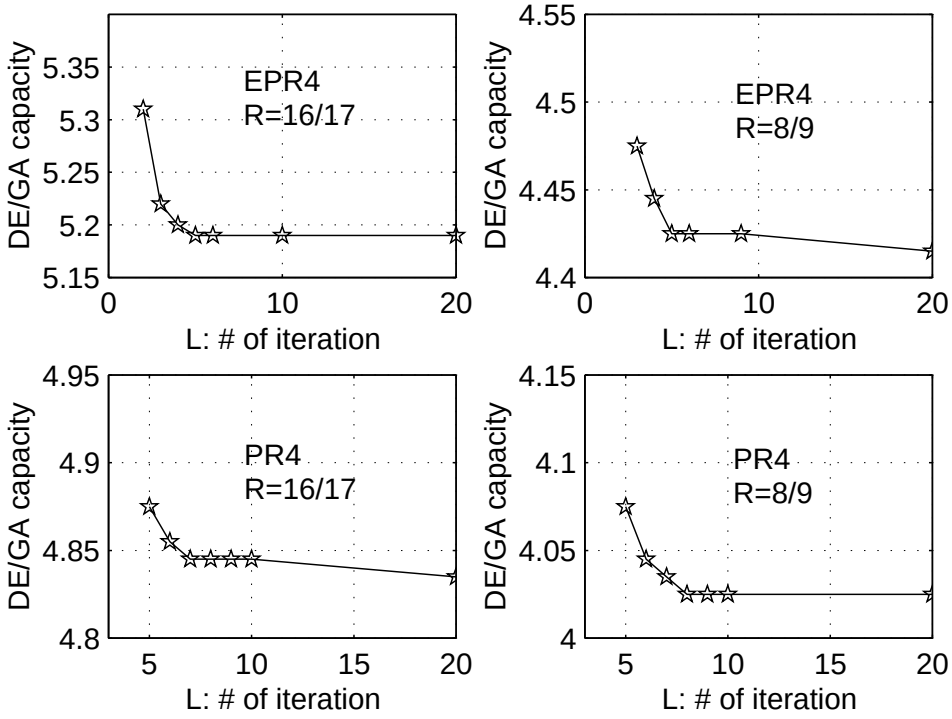


Fig. 61. Thresholds vs L in LDPC systems.

different designs of LDPC codes, but the difference should be small. Further, for a fixed complexity, the value of L may be lower than the ones reported (which is for unconstrained complexity).

2. TPC/SPC Systems

Most work on turbo equalization of partial response channels treats the combination of the precoder and the ISI channel as the inner code [90][91] [92] [27] [94]. Therefore, each iteration in the turbo equalization process involves decoding of the outer code followed by a BCJR decoder for the precoded channel. Since most of the complexity comes from the inner MAP decoder (Table IV), it is desirable to reduce the number of iterations q involving the MAP decoder and, hence, to devise a decoding strategy which minimizes q with a fairly small sacrifice in performance. This is of particular in-

terest to high-density recording systems where the appropriate PR targets correspond to 8-state (like EPR4) or even 16-state trellis (like E^2PR4 or ME^2PR4).

Although it is important to exploit the memory in the ISI channel and the recursiveness introduced by the precoder, the interleaving gain is dependent only on the recursiveness of the inner code. With this observation, we propose an efficient and effective modified receiver structure where the combination of precoder and the TPC/SPC code is considered as an outer code and the non-precoded PR channel is the inner code. As such, MAP equalization need not be performed at every iteration stage. Rather, it can be done after every s iterations between the TPC/SPC code and the precoder, as illustrated in Figure 62. The key advantage here is that the precoder is often of the form $1/(1 \oplus D)$ or $1/(1 \oplus D^2)$, which can be represented by a 2-state trellis rather than an 8- or 16-state trellis for an EPR4 or E^2PR4 channel and therefore saving considerable complexity without sacrifice in performance. The complexity can be further reduced by using the sum-product algorithm on the graph of the precoder [24] [54]. When the precoder is of the form $1/(1 \oplus D^m)$ (m an integer), its corresponding code graph alone has no cycles and therefore sum-product decoding is optimal. In particular, using the *tanh* implementation of the sum-product algorithm results in approximately 1/5 the complexity of a conventional 2-state BCJR algorithm for the precoder $1/(1 \oplus D^m)$ (altogether 5 additions and 5 lookups per encoded bit) [24] [54].

Given this set up, we now address two important questions in a practical implementation:

1. Given an overall allowable complexity, what are the optimum values of s (the number of local iterations in TPC/SPC decoder) and q (the number of turbo iterations between the channel and the outer code)?

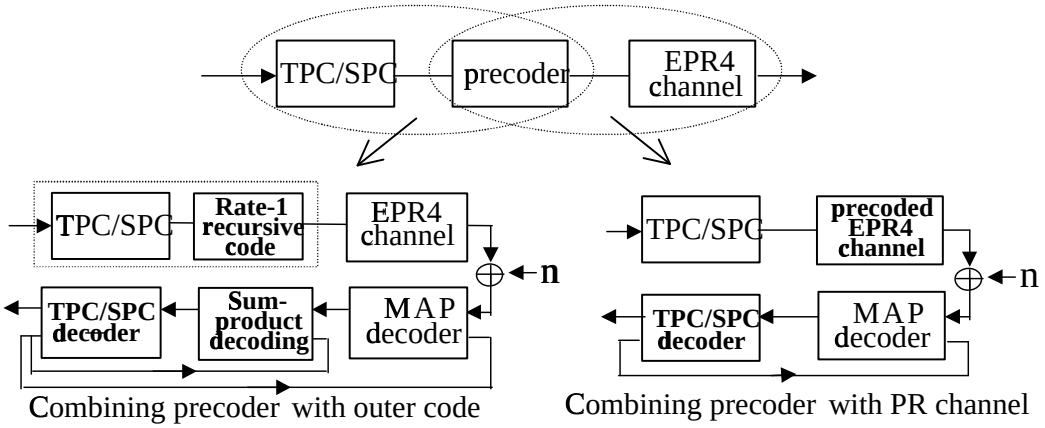


Fig. 62. Different view-stands of the serial concatenated system.

2. What is the trade-off in performance vs overall complexity, given that we can optimize the performance by answering question 1.

We answer these questions using density evolution with some modifications. For a given q and s, let $\Delta(q,s)$ denote the overall complexity, including additions, max operations and lookups (see Table IV). Since we are interested in finite complexity and, hence, finite number of iterations, we first reformulate the thresholds as the SNR for which the mean reaches a threshold (η_{thresh}) (a positive number serving as the practical infinite point). For a given q and s, the new threshold is thus given by

$$\eta(q,s) = \inf_{SNR} \{SNR : m_o^{(q)} \geq \eta_{thresh}\}. \tag{5.22}$$

When the value of η_{thresh} is set large enough, the difference from the actual threshold will be negligible. For a given overall complexity Δ_0, different values of q and s will produce different thresholds and we are interested in the best (least) value $\eta^*(\Delta_0)$ given by

$$\eta^*(\Delta_0) = \min_{q,s}\{\eta(q,s) : \Delta(q,s) \leq \Delta_0\}. \tag{5.23}$$

The cost function $\Delta(q, s)$ depends on the outer code, the actual channel and the precoder. For a TPC/SPC code on an EPR4 channel, we have (see Table IV)

$$\Delta(q, s) = ((10 + 10)\, s + 25 \cdot 2^3 + 5)\; q = (20\, s + 205)\; q, \tag{5.24}$$

where 205 is the number of operations per encoded bit for a BCJR decoding of the 8-state EPR4 channel, and 20 is the number of operations per encoded bit for one small iteration between the TPC/SPC code and the precoder.

Figure 63 shows a plot of $\eta^*(\Delta_0)$ versus Δ_0 for various values of s for a rate 0.94 TPC/SPC code over EPR4 channels, where $\eta_{thresh} = 30$. Obvious from the figure is that, for a given Δ_0, the value of s has a significant impact on the resulting thresholds. Also seen from the figure is that setting s around 3 optimizes the thresholds and complexity consistently and, hence, is a good choice. This means that the equalization procedure (with respect to the channel MAP) is used only once every 3 iterations and, hence, results in complexity savings. It is interesting to note that the performance of $s = 40$ is quite poor. That is, for a fixed complexity, if s is increased beyond 5, due to the few stages of turbo equalization that are possible, the resulting thresholds are weak. Depending on the exact complexity Δ_0 that can be allowed, the procedure can be repeated over that range to optimize s and q.

F. Simulation Results

1. Simulation Parameters

To be applicable to present day data storage systems, the 2-D TPC/SPC codes we investigated have rate 0.89 and rate 0.94 which are formed from (17,16) and (33,32) TPC codes, respectively. We combine sixteen $(17, 16)^2$ TPC/SPC codewords and four $(33, 32)^2$ TPC/SPC codewords respectively to form an effective data block size of 4K

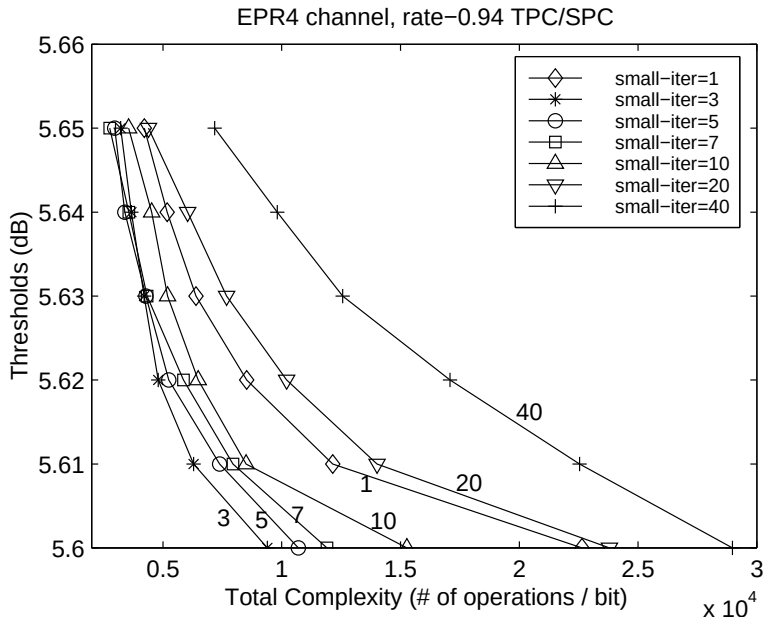


Fig. 63. Optimization of TPC/SPC systems.

bits. The channel models we test are PR4 and EPR4 magnetic recording channels. For comparison purposes, also presented are the results of a rate $8/9 = 0.89$ and $16/17 = 0.94$ regular LDPC codes with column weight 3 and data block size 4K. It should be noted here that irregular LDPC codes of such high rates have been seen to perform slightly worse than regular codes [35] and, hence, this represents the best case for LDPC codes.

Being aware of the high data-rate and low cost required by data storage systems, complexity and latency is to be carefully watched for. At the current stage of commercial hardware implementation, more than 10 turbo iterations are considered intolerably slow and therefore impractical. Hence, unless otherwise stated, performance curves presented employ no more than 3-8 iterations. In each iteration, the turbo equalization procedure includes a forward-backward process in the channel MAP decoder followed by 2 rounds of bit-check updates in the outer TPC/SPC de-

coder or 4 rounds of bit-check updates in the outer LDPC decoder. Although this leads to a decoding complexity of LDPC codes a bit higher than TPC/SPC codes, it is a good compromise of complexity and performance for both codes.

For a fair comparison of the various schemes at different rates and normalized linear densities D_{norm}, adjustment of the normalized density for the rate loss is needed. In other words, the physical recording density for a rate-R code is $D_{user} = D_{norm}/R$.

2. SNR definition

For the ideal PR channels with additive while Gaussian noise, we define the signal to noise ratio as

$$SNR = 10\ \log_{10}\left(\frac{E_b}{N_o}\right) = 10 \cdot \log_{10}\left(\frac{E_s}{2R\sigma^2}\right), \tag{5.25}$$

where E_s is the symbol energy, $\sigma^2 = N0/2$ is the noise variance, and R is the code rate.

For the equalized Lorentzian channel model, we define SNR as

$$SNR = 10\ \log_{10}\left(\frac{\bar{S}}{\bar{N}}\right), \tag{5.26}$$

where $\bar{S}$ and $\bar{N}$ are the mean-square signal and noise values measured at the input to the equalizer, respectively.

3. Results on PR Channels

Figure 64 shows the BER performance of LDPC codes and TPC/SPC codes over PR4 and EPR4 channels. It can be seen that gains of 4.4 to 5 dB over uncoded partial response maximum likelihood systems are obtained for TPC/SPC codes at a BER of 10^{-5}, comparable to those of LDPC codes. All TPC/SPC codes are precoded with $1/(1 \oplus D^2)$ which is the best for PR4/EPR4 channels, as shown analytically in [74]

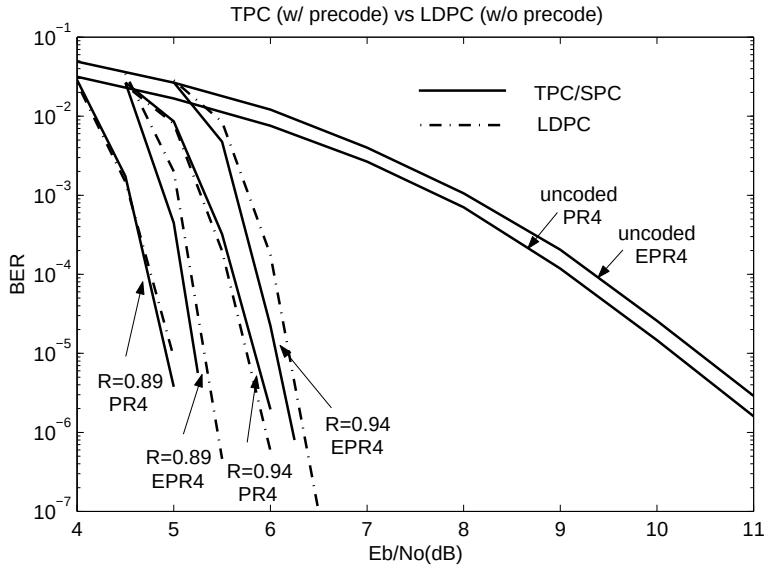


Fig. 64. Performance of high-rate TPC/SPC codes and LDPC codes over ideal PR4 and EPR4 channels.

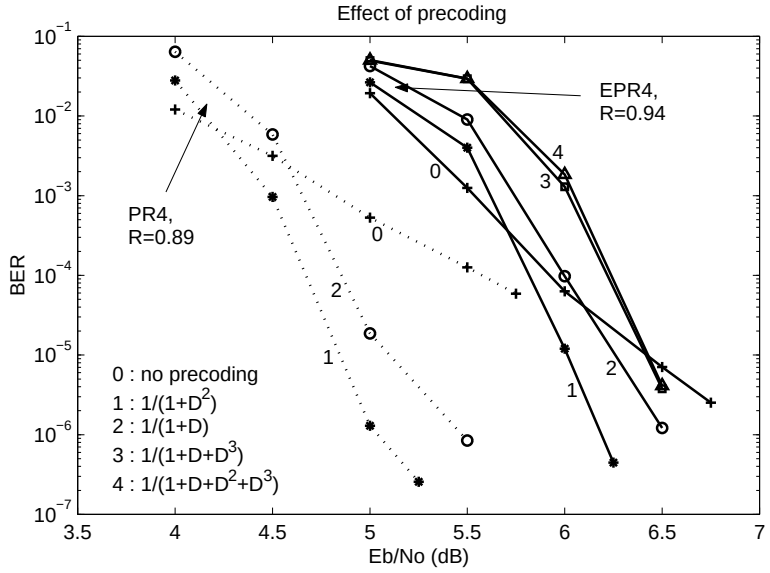


Fig. 65. Effect of precoding in TPC/SPC systems.

and as shown empirically in Figure 65. LDPC codes are not precoded, for, as shown in [74], their performances are better without precoding. We have confirmed this through simulations also. Hence, the comparison is fair as it represents the best cases for both codes.

Although both TPC/SPC and LDPC codes seem to offer significant coding gains when the average BER is of the order of 10^{-7}, it is still unclear whether LDPC codes and TPC/SPC codes may suffer from an error floor. Therefore, the conventional use of RS-ECC is still necessary to reduce the BER to 10^{-15} as is targeted for recording systems. The RS-ECC code works on the byte level, capable of correcting up to t byte errors in each data block of size 4K bits or 512 bytes (t is usually around 10 to 20). Hence, the maximum number of uncorrected errors left over in each block after TPC/SPC or LDPC decoding has to be relatively small to guarantee the proper functioning of the RS-ECC code. In other words, block error statistics are crucial and closely relate to the overall system performance. Unfortunately, this has been largely neglected in most of the previous work.

Figure 66 and Figure 67 plot the histograms of the number of bit/byte errors for an effective block size $4K$, rate 0.94 LDPC code and TPC/SPC code over EPR4 channels, respectively. The left column plots bit error histograms and the right byte error histograms. The statistics are collected over more than $100,000$ blocks of data size 4K bits. At an SNR of 6.5 dB and after the 10_{th} iteration (outer loop), the maximum number of symbol errors observed in a single block is less than 10 for TPC/SPC codes (which would be corrected by the RS-ECC code), but around 50 for LDPC codes. We attribute this to the fact that TPC/SPC codes have quite small minimum distance. It is perceived that when an error occurs, the decoder is mostly likely to decode it to its nearest neighbor which fortunately is not different in too many bit positions. If further iterations are allowed, error bursts in LDPC

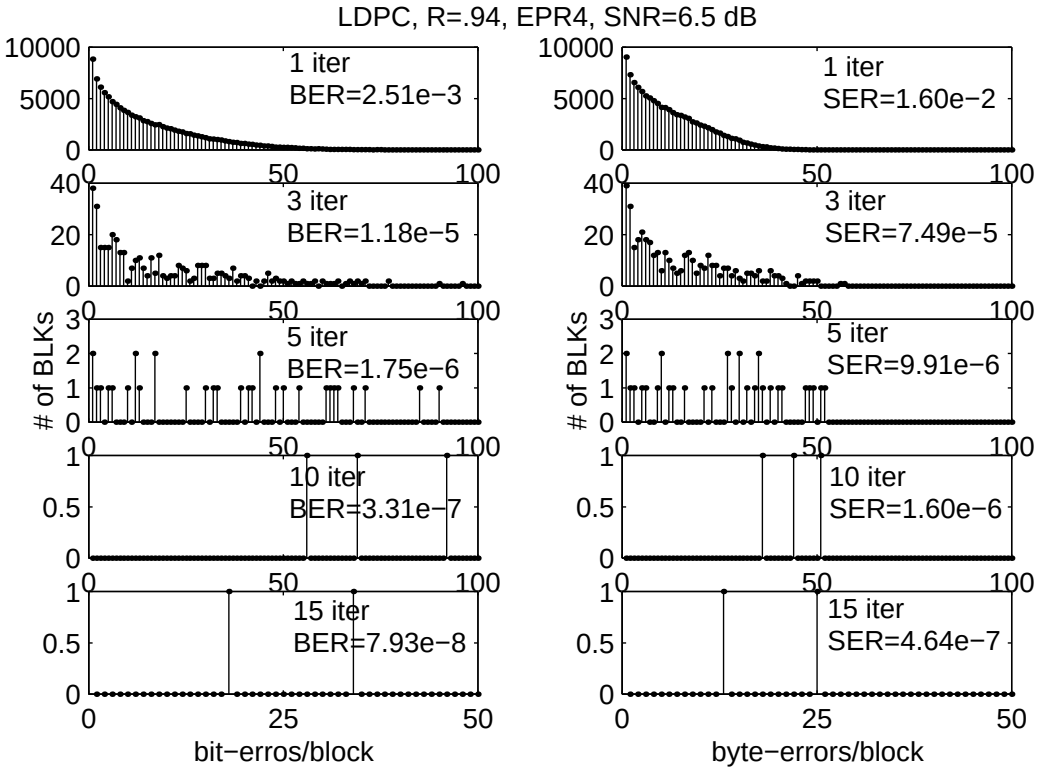


Fig. 66. Error statistics of LDPC codes over EPR4 channels. (Code rate $R = 0.94$, $E_b/N_0 = 6.5$ dB, collected over 100,000 blocks, x-axis: maximum number of errors observed within a block, y-axis: occurrence of such blocks.)

codes are alleviated. Nevertheless a block containing 25 symbol errors is observed after 15 turbo iterations and this may still cause the RS-ECC code to fail. Unless a more powerful RS-ECC is employed, LDPC codes are prone to block failure, where all data in that block are presumed lost. It should be noted that although what we have observed suggests that TPC/SPC codes may be more compatible with magnetic recording systems than LDPC codes, the statistics are nonetheless insufficient. Due to the random interleaver as well as the suboptimal iterative decoding, hardware tests may still be needed before a convincing argument can be made.

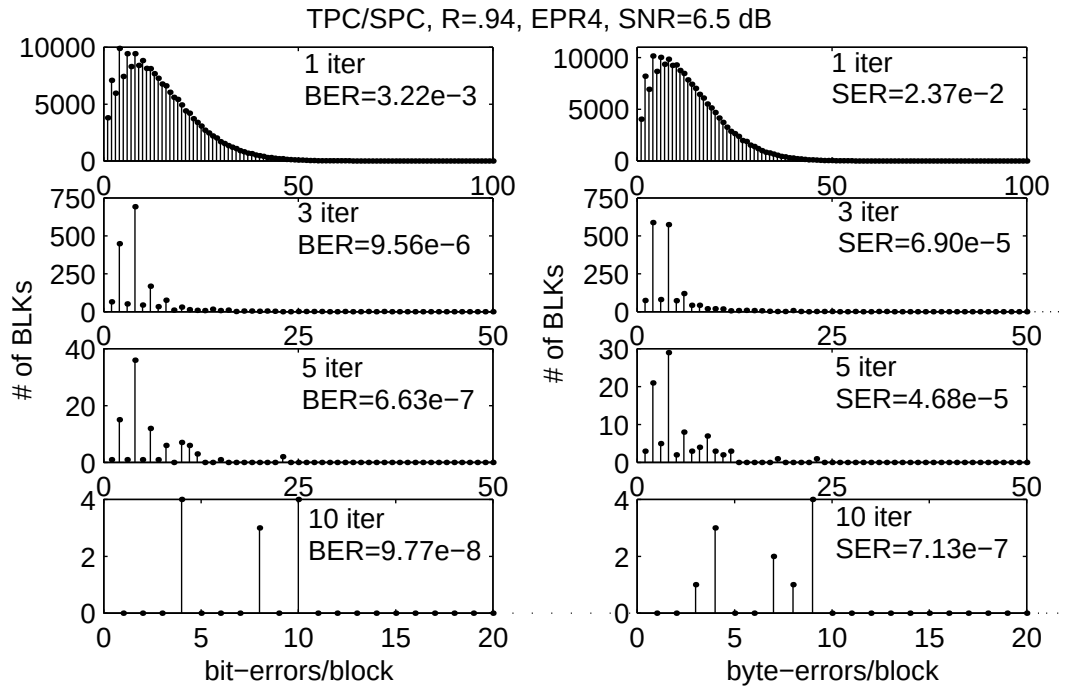


Fig. 67. Error statistics of TPC/SPC codes over EPR4 channels. (Code rate $R = 0.94$, $E_b/N_0 = 6.5$ dB, collected over 165,000 blocks.)

4. Results on PR-Equalized Lorentzian Channels

Figure 68 plots the simulation results of rate-0.94 TPC/SPC codes over EPR4-equalized Lorentzian channels at several normalized densities from low to high.

To examine the trade-off between code rate loss and coding gain, we compare a rate-0.89 and rate-0.94 TPC/SPC codes at several normalized densities over a EPR4-equalized Lorentzian channel. Note for a given normalized density in a given channel, lower rate codes provide more error correction capabilities but encounter more ISI, whereas higher rate codes are intrinsically weaker but encounter less ISI. Hence a balance in choosing code rate for different areal densities must be maintained for a best hit. The curves in Figure 69 clearly indicate that lower rate codes work better at low densities, while higher rate codes are better at high densities.

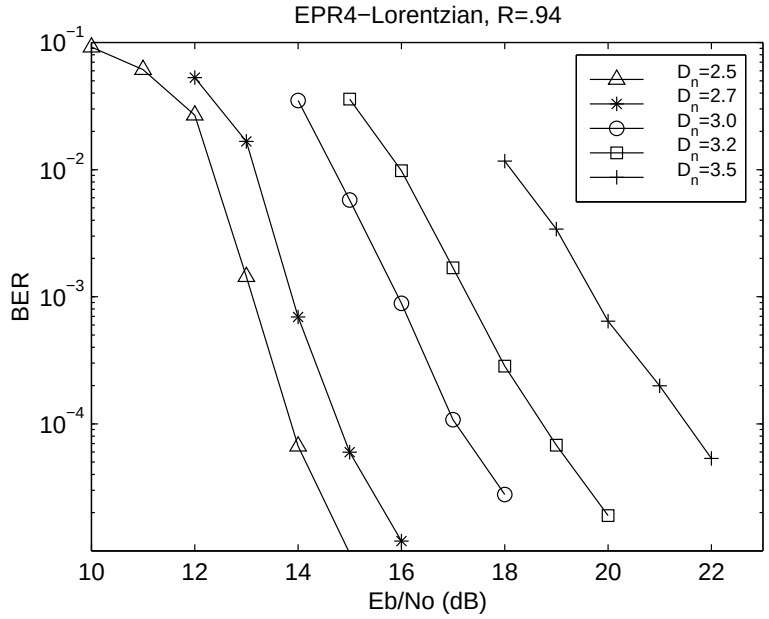


Fig. 68. Performance of TPC/SPC codes over Lorentzian channels. (EPR4 target, TPC/SPC code rate $R = 0.94$, 3 turbo equalizations.)

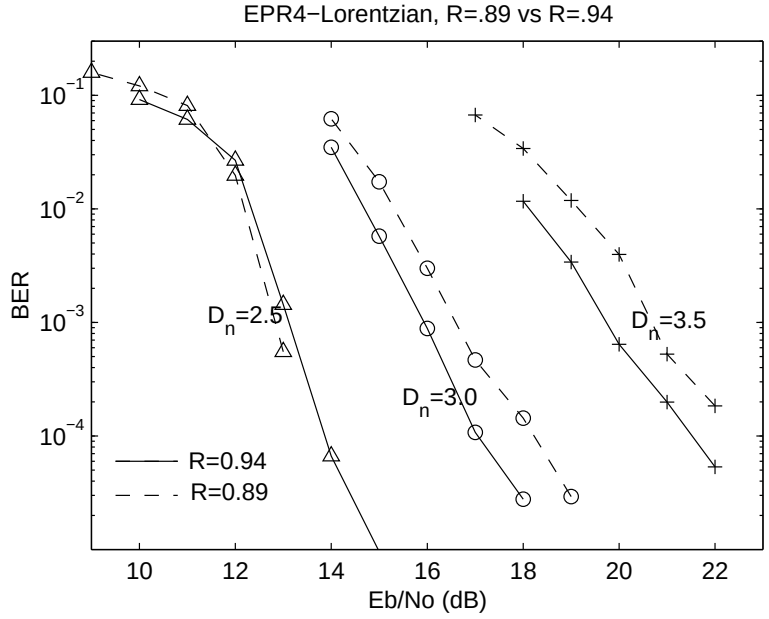


Fig. 69. Code rate loss vs coding gain on PR-equalized Lorentzian channels. (EPR4-target, TPC/SPC code rate $R = 0.89$, 0.94, normalized density $D_n = 2.5$, 3.0, 3.5, 3 turbo equalizations.)

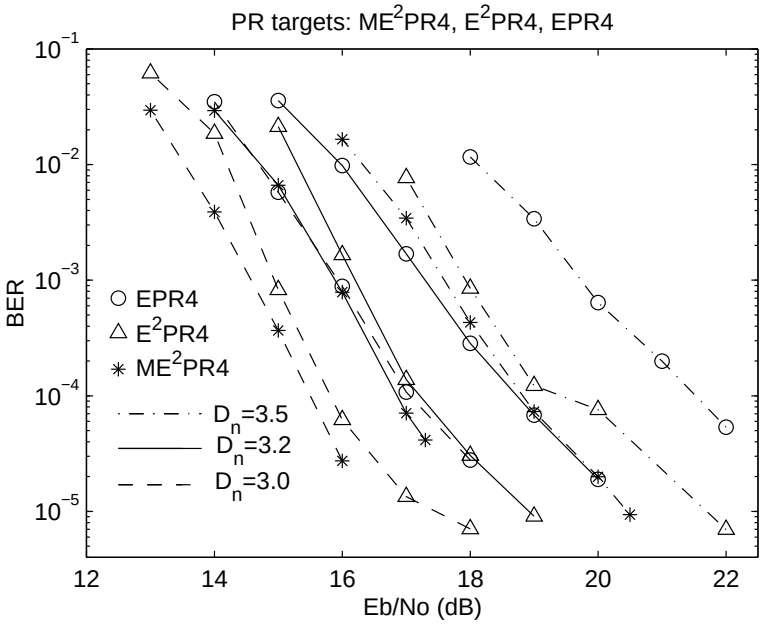


Fig. 70. Effect of PR targets at different normalized densities on Lorentzian channels. (TPC/SPC code rate $R = 0.94$, normalized density $D_n = 3.0, 3.0, 3.5$, 3 turbo equalizations.)

For a given normalized areal density, the BER performance of the code is contingent upon the equalized PR targets among other factors. Research work has indicated that the conventional PR4 targets ($H(D) = 1 - D^2$) are ineffective in shaping the channel at high densities, introducing much noise enhancement. Higher order forms, like EPR4, E^{2}PR4 and ME^{2}PR4, are proposed as better targets under different conditions. From a more systematic point of view, this optimization problem should not be limited to searching for the best PR target that will match the recording channel more accurately with less noise enhancement. Rather, the factors involved should also include the specific error correction codes, in addition to PR targets and areal densities. The argument comes from the observation that an error correction code could be more sensitive to one type of channel than the other. An example of this can be found in [94], where it is shown that while TPC/SPC codes perform almost the

same as LDPC codes over PR1 channels, they offer a substantially better performance over PR2 channels. From this aspect, we investigate the performance of a rate-0.94 TPC/SPC code over different PR targets at different normalized densities. From the plot (Figure 70), we can see that E^2PR4 seems to be a better target than EPR4 for TPC/SPC codes at normalized densities of $D_{norm} = 3.0, 3.2, 3.5$, and ME^2PR4 seems to offer even larger gains. The higher the density, the more the gain. However, be aware that the gains come at the expense of increased complexity. In the E^2PR4 and ME^2PR4 cases, the channel trellis involves 16 states, doubling that of the EPR4 model.

Error statistics on ideal PR channels have revealed that TPC/SPC codes seem better in error bursts. To facilitate the understanding of the effect of colored noise and imperfect channel shaping, error statistics are examined for Lorentzian channels also. As shown in Figure 71, the same good news prevails. After the 3_{rd} iteration, no blocks containing more than 14 symbols[4] over 100,000 blocks transmitted. Although not shown, we also examined a E^2PR4-equalized Lorentzian channel. A similar phenomenon is observed. This confirms that (1) TPC/SPC codes are quite insensitive to colored noise (due to the random interleaver) and (2) TPC/SPC codes are able to work in harmony with the out-most RS-ECC codes, maximizing the overall capacity.

G. Summary

This chapter investigates the potential of applying TPC/SPC codes to magnetic recording systems, with LDPC codes as a comparison study. The main results from this chapter can be summarized as follows:

1. In the application of TPC/SPC codes to PR magnetic recording channels, con-

[4]Here a symbol is equivalent to a byte, which contains 8 consecutive bits.

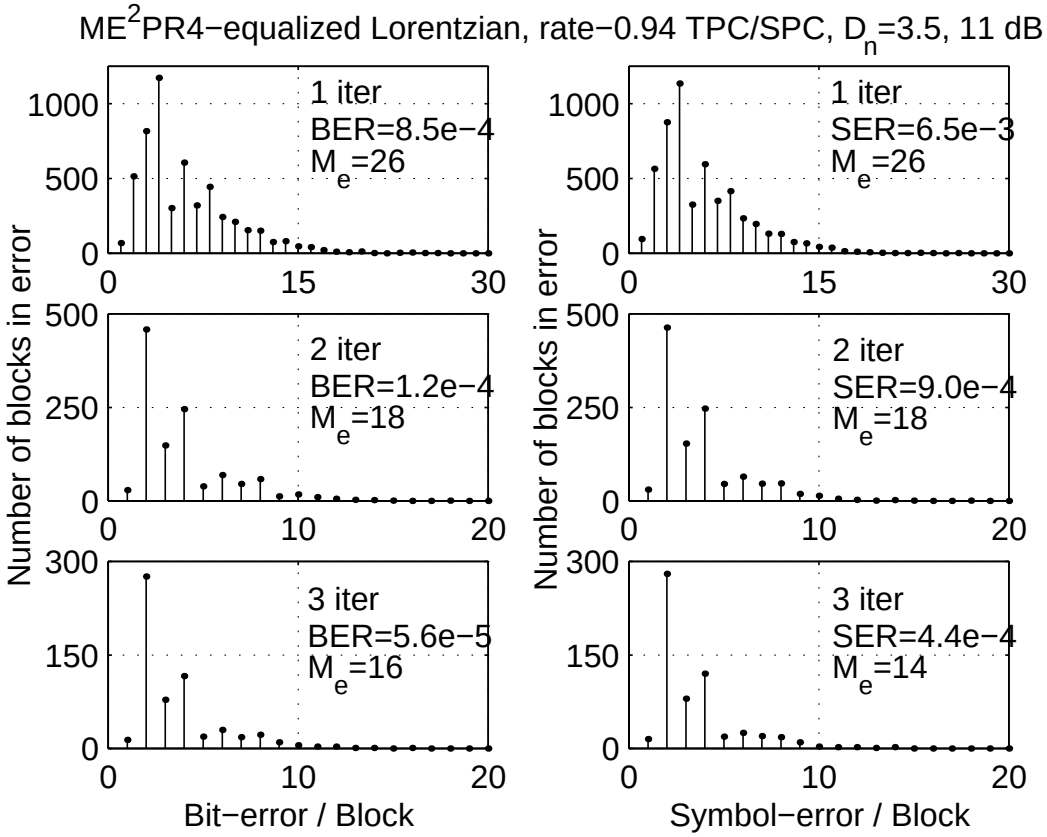


Fig. 71. Bit and byte error statistics of TPC/SPC code over Lorentzian channels. (ME^{2}PR4 target, TPC/SPC code rate 0.94, normalized density $D_n = 3.0$, SNR=16 dB. The statistics are made from observation of 100,000 blocks, each of data block size 4K bits. In the plot, "BER" stands for bit error rate, "SER" stands for symbol/byte error rate and "M_e" stands for the maximum number of errors observed within a block.)

siderable coding gains can be achieved by combining several blocks of TPC/SPC together (since interleaving gain is proportional to the number of TPC/SPC blocks combined in a codeword) and by choosing a proper precoder for the channel. In particular, gains of more than 4.4 dB over uncoded systems are observed on ideal PR4 and EPR4 channels, revealing a performance comparable to that of LDPC codes. On Lorentzian channels, it is observed that the gains seen for the ideal case are reduced somewhat, but they are still substantial, especially with a more properly equalization target such as E^2PR4 and ME^2PR4 at high densities.

2. While the decoding complexity is slightly smaller than that of LDPC codes, TPC/SPC codes are linear time encodable. Further, they are well-structured and do not require large storage for the parity check and generator matrices. The interleaving pattern should be stored - however, algebraic interleavers which can be generated "on the fly" can be used which demonstrate reasonably good "randomness" and which save precious storage in hardware implementation [24] [53].

3. In contrast to LDPC codes whose large error bursts are beyond the capacity of the outer RS-ECC codes, TPC/SPC codes demonstrate error statistics favorable to RS-ECC codes, which assures a consistent and quality performance of the whole system.

4. Density evolution is an effective tool in the analysis of iterative decoding processes by taking into consideration both the code structure and the iterative feature of the decoding algorithm. Through its use in the calculation of thresholds for TPC/SPC, LDPC and serial turbo systems, we demonstrate a framework under which this useful method can be exploited for a variety of concatenated

systems where iterative approaches are used.

To summarize, our work has indicated TPC/SPC codes as a promising candidate in the application of future magnetic recording systems. However, further experiments need to be conducted over more realistic channel models, like Lorentzian channels and, hopefully, on real data collected in the lab. Other interesting problems include how to achieve a good compromise among iterations, performance, complexity and delay in a practical setting, as well as how to incorporate the run-length limit constraint without affecting much complexity and performance.

CHAPTER VI

RATE-COMPATIBLE LDPC CODES FOR USE IN HYBRID ARQ SYSTEMS IN PACKET DATA NETWORKS

A. Introduction

Flexible rate is desired in the design of practical error control systems, especially on time-varying channels, or in applications where adaptive error correction or unequal error protection is required (like speech or image compression). Rate-compatible codes are a family of nested codes where the codeword bits from the higher-rate codes are embedded in the lower-rate codes and, hence, can be encoded and decoded using a single encoder/decoder pair. They are of particular interest in packet data systems that allow for retransmission requests such as automatic repeat request with forward error correction systems to achieve desired throughput efficiency with a high degree of flexibility.

Key elements concerning the throughput efficiency of an ARQ/FEC system include a wise ARQ strategy and, perhaps more importantly, a powerful rate-compatible code. Successful attempts in creating rate-compatible codes have used BCH codes [100], convolutional codes [101] [102] and turbo codes [43] [44] [103] [104]. BCH codes and convolutional codes are easily implementable but cannot provide near capacity performance. Turbo codes have demonstrated impressive performance, but their high decoding complexity results in high cost in a commercial system.

This paper focuses on low density parity check codes [9] [11] [10] [14] which have been shown to provide performance comparable to turbo codes, yet with less decoding complexity. In the search for simple constructions of efficient rate compatible LDPC codes, two approaches are investigated: the conventional technique of puncturing and

a special approach of extending. Through analysis on code ensemble and asymptotic performance using density evolution (DE), we show that the efficiency of puncturing is limited only to high rate range where the amount of puncturing is small. To extend the dynamic range to low rates, we resort to the technique of extending. We propose a special code structure where extending is exploited to design efficient RC-LDPC codes at low rates. Combining both techniques, a systematic model is presented to construct RC-LDPC codes that are potentially powerful to provide strong error correction ability and high system throughput at a wide range of code rates.

A few examples of the proposed RC-LDPC codes are presented and their remarkable performances (through computer simulations) have opened the possibility for capacity-approaching throughput. A type II hybrid ARQ/FEC system using RC-LDPC codes is investigated and is shown to achieve throughput at about 1 dB from the capacity, which is comparable to turbo-ARQ systems in [43] [44], yet with less decoding complexity.

The rest of the Chapter is organized as follows. Section B analyzes a type II hybrid LDPC-ARQ system using code combining and packet combining and pinpoints the importance of a strong FEC code. Section C discusses the construction of efficient RC-LDPC codes exploiting both puncturing and extending. Simulations are provided along with some discussion. Section D provides concluding remarks.

B. Hybrid ARQ/FEC Using RC-LDPC Codes

1. ARQ System Using RC-LDPC Codes

A typical ARQ/FEC system uses both error correction codes and error detection codes (EDC, such as a cyclic redundancy check (CRC)). After the transmitted codeword is decoded by ECC, it is examined by EDC. If the decoding is deemed in error, a

negative acknowledgment (NACK) is sent as a retransmission request. When LDPC codes are used in an ARQ protocol, their strong error detection ability enables them to act as both error correction and error detection codes. This obviates the need for another EDC, and thereby reduces the overhead. Whether the error detection capability of LDPC codes can match that of a CRC is not fully discussed in this paper, but nevertheless it is assumed that the error detection capability provided by the LDPC code is sufficient for the application at hand.

The type-II hybrid ARQ/FEC system under investigation exploits the rate compatibility of RC-LDPC codes and uses code-combining and packet-combining to maximize the throughput efficiency and transmission reliability. A packet is first transmitted using the highest rate code. If it is not deemed correctly decoded, a NACK is fed-back and a new set of parity bits is provided by the transmitter (i.e. incremental retransmission). This new set of parity bits, combined with all previous transmissions, is treated as a codeword of a lower rate code in the family (code combining) which provides stronger error correction capability [105]. This procedure continues, until all supplemental parity bits are used up, and then the procedure restarts with another "initial transmission". When a new copy of the same coded bits (either data bits or parity bits) are received, old copies are not discarded. Rather, they are all combined together to facilitate decoding (packet combining). In general, packet combining is done by averaging the soft decision values from the multiple copies, and on AWGN channels, this is equivalent to maximum-likelihood diversity combining [106]. Specifically for LDPC codes with a soft message-passing decoder, the input message to the decoder of a bit s_i is obtained by $\sum_{j=1}^{k} 2r_i^{(j)}/\sigma^2$, where $r_i^{(1)}, r_i^{(2)}, \cdots, r_i^{(k)}$ are the multiple copies received for the same bit s_i. The above strategy is optimal for achieving high throughput either in stop-and-wait ARQ or selective-repeat ARQ systems, under the assumption that the feedback channel is noiseless, that the buffer

size is infinite, and that the transmission latency, the feedback channel traffic and the decoding complexity are not a concern.

2. Throughput Analysis

A standard measure for the efficiency of an ARQ scheme is throughput, which is defined as the average number of coded and modulated symbols that need to be transmitted for a single data bit to reach the destination error-free. We define $p_i, i = 0, 1, \cdots$, as the probability that the decoder succeeds after the i_{th} retransmission but fails at all previous attempts (the initial transmission is considered 0_{th} retransmission); K_0 as the number of data bits in a frame/codeword; N_0 is the packet size of the initial transmission; and $M_i, i = 1, 2, \cdots$, are the packet size of the i_{th} retransmissions. The throughput, η, is then given by

$$\eta = K_0/(N_0 + \sum_{i=1}^{\infty} p_i M_i), \tag{6.1}$$

where

$$p_i = (1 - F_i) \sum_{j=0}^{i-1} F_j, \quad i = 1, 2, \cdots, \tag{6.2}$$

where F_i is the word/frame error rate after the i_{th} retransmission. Substituting (6.2) into (6.1), we have:

$$\eta = R_0 \Big/ \Big(1 + F_0 \sum_{i=1}^{\infty} \frac{M_i}{N_0} (1 - F_i) \sum_{j=1}^{i-1} F_j \Big), \tag{6.3}$$

where $R_0 = K_0/N_0$ is the code rate of the initial transmission. It is apparent from (6.3) that the error rate of the initial transmission, F_0, plays an important role in the throughput efficiency, since subsequent transmissions occur only when the initial transmission fails. If the initial word error rate is very small, in particular, if $F_0 \to 0$, then the throughput will reach R_0, the highest possible rate in the system. Further,

observe that the term $\frac{M_i}{N_0}$, which represents the granularity of the retransmission, also has an impact on the throughput. Smaller retransmission size M_i means finer adjustment, which will improve the throughput efficiency and smoothen the throughput curve, but may incur more delay and decoding complexity in practical systems.

Since the word error rate, F_i, of the FEC code plays the key role in achieving ARQ throughput efficiency, it is crucial to choose a strong code. The next section focuses on the construction of good rate-compatible LDPC codes.

C. Constructing RC-LDPC Codes

A regular LDPC code has parameters (N, K, t, s), which denote the codeword length, data block size, column weight and row weight of the parity check matrix, respectively. We use a sequential design, the bit filling method, to obtain column-weight-3 regular LDPC codes as the mother code for the RC-LDPC code family. To ensure decent performance of the mother code, we have enforced the constraint that the girth (the length of the shortest cycle in the code graph) be at least 6 in the construction. We would like to mention that irregular LDPC codes (with nonuniform column/row weights) have been shown to outperform regular LDPC codes in bit error rate [11] [14], but the difference is very marginal for short to moderate code lengths (a few hundred to a few thousand bits). Further, whether they are also better in *word error rate* (which is the determining factor of ARQ throughput as shown in (6.3)) is less known and needs to be bench marked. We note that the word error rate and error detection capability depend on the minimum distance, d_m, of the code. For regular LDPC codes with column weight $t \geq 3$, the ensemble average minimum distance increases linearly with code length N [9]. However, this may not be true with irregular LDPC codes. Since we would like LDPC codes to assume the dual role of error correction

and error detection, it is thus desirable to start with a regular code that typically has large minimum distance (recall the number of errors a code can detect is $\leq d_{min}-1$). Additionally, unlike irregular LDPC codes, regular LDPC codes have uniform row weights and column weights and, hence, may be exploited for parallelization in the decoder implementation.

1. Puncturing

Puncturing has been widely used in BCH codes, each of of which is a subcode of the mother code and each of which is encoded and decoded 'convolutional codes and turbo codes to achieve rate flexibility [43]-[104]. It is also applicable to LDPC codes. Through proper puncturing, a series of higher rate codes are obtained from the low rate mother code. The encoder generates the full set of parity bits, but some parities are not transmitted (punctured). The decoder inserts erasures to where parity bits are punctured and then performs the decoding algorithm as in a non-punctured case.

An LDPC code can be viewed as a parallel concatenation where each row in the parity check matrix H acts as a simple component code (a parity check). Consider an (N, K) LDPC code where L (parity) bits/columns are punctured. Those rows/checks that happen to have "1"s in the positions of the punctured bits are treated as being erased. To see how puncturing impairs the code performance, we examine the effect of puncturing on the ensemble of the LDPC codes and study the asymptotic performance of the punctured codes.

a. Code Ensemble

Consider the ensemble of (N, K, t, s) LDPC codes. Randomly pick a parity check matrix from the ensemble and puncture $L = \rho N$ columns, where $\rho = L/N$ is defined as the puncturing rate. Assuming all rows are independent, the average portion of

rows being affected by at least one erasure, $\lambda_1(\rho)$, and by multiple (two or more) erasures, $\lambda_2(\rho)$, are given by

$$\lambda_1(\rho) = 1 - \frac{\binom{N-\rho N}{s}}{\binom{N}{s}}, \tag{6.4}$$

$$\lambda_2(\rho) = 1 - \frac{\binom{N-\rho N}{s} + \binom{\rho N}{1}\binom{N-\rho N}{s-1}}{\binom{N}{s}}. \tag{6.5}$$

We observe that large value of $\lambda_2(\rho)$ has a destructive affect on the decoder performance. To see this, we need to get back to the message-passing decoding algorithm of LDPC codes [10]. Suppose bits $j_1, j_2, \cdots, j_s$ participate in check j, the extrinsic information of bit j_k (in log likelihood ratio or LLR form), denoted as $L_{e,j_k}(x)$, to be obtained from check j is given by

$$L_{e,j_k}(x) = \boxplus_{i=1, i\neq k}^{s} L_{j_i}(x), \quad k = 1, 2, \cdots, s, \tag{6.6}$$

where $L_{j_i}(x)$ denotes the LLR message content of bit j_i, and operation $\boxplus$ is defined as $\gamma = \alpha \boxplus \beta = log((1 + e^{\alpha}e^{\beta})/(e^{\alpha} + e^{\beta}))$. An erasure in position j_i means its initial message content is 0, i.e. $L_{j_i}(x) = 0$. When multiple erasures present in one check, at least one term on the right hand side of (6.6) is 0. Since $\alpha \boxplus 0 = 0$, this leads to $L_{e,j_k}(x) = 0, \forall k \in \{1, 2, \cdots, w\}$. In other words, no information is exchanged/obtained from this row/check. When the percentage of such rows is large, message exchange becomes quite inefficient and ineffective. Through simulations, we observe that, in such cases, the decoding algorithm may get stuck in a "zero-trapping" state, leading to poor performance.

Solid lines and dashed lines in Figure 72 plot $\lambda_1(\rho)$ and $\lambda_2(\rho)$ vs ρ for different code rates. It can be clearly seen that puncturing has a larger adverse impact when the mother code is of low rate than when the mother code is of high rate (the punctured code having the same code rate), which matches our simulations.

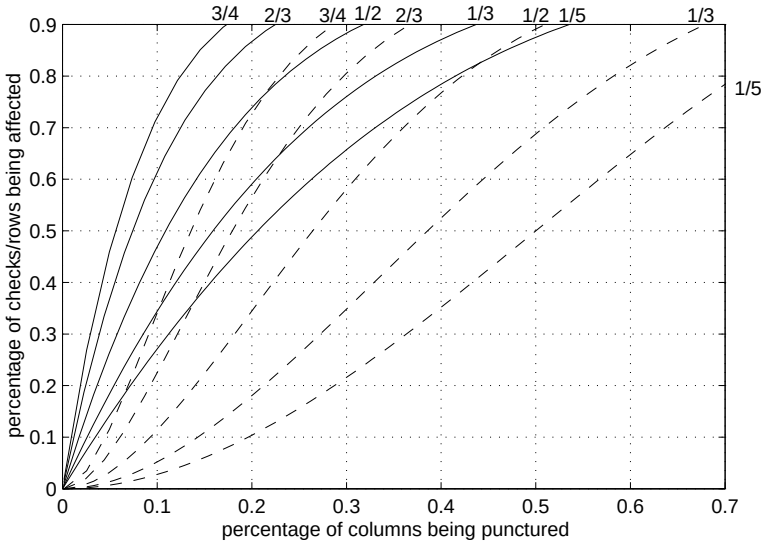


Fig. 72. Effect of puncturing on column-weight-3 LDPC code ensemble. (Solid lines: percentage of rows affected by at least one erasure: from left to right: rate 4/5, 2/3, 1/2, 1/3, 1/5. Dashed lines: percentage of rows affected by multiple erasures: from left to right: rate 4/5, 2/3, 1/2, 1/3, 1/5.)

b. Asymptotic Performance

To further understand the effect of puncturing, we use density evolution (DE) to examine the asymptotic performance of punctured LDPC codes and to quantify the performance loss caused by puncturing. Here asymptotic refers to an infinite code length N and an infinite iteration number l in the message-passing iterative decoding process. The idea of density evolution is to track the distribution of the messages passed along the code graph in each step, and to examine the portion of the incorrect messages (i.e. messages leading to the wrong decision). Details of density evolution on (non-punctured) LDPC codes can be found in [14]. Here we focus on the difference in the computation between the punctured and the non-punctured case.

Assuming AWGN channels with noise variance σ^2 and antipodal signaling with unit energy (± 1), the density of the initial messages (from the channel) of a non-

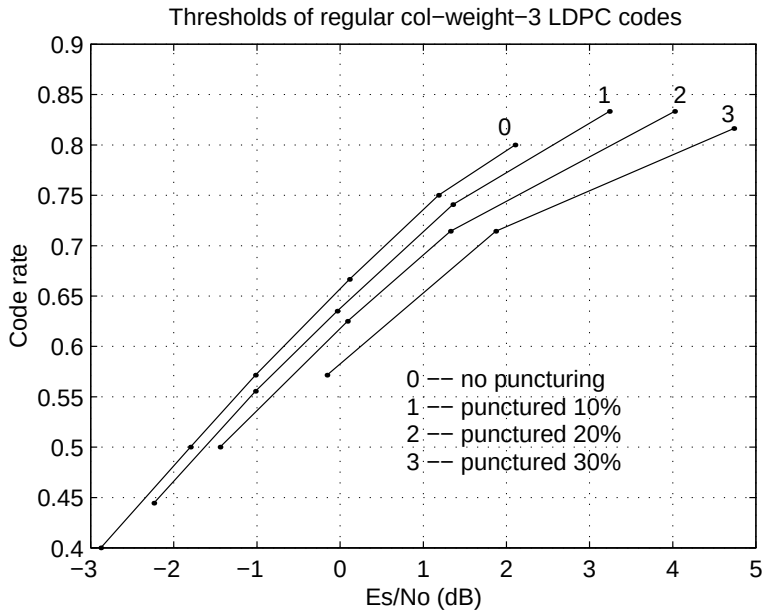


Fig. 73. Thresholds of punctured/non-punctured LDPC codes using density evolution. (regular LDPC codes, column weight 3.)

punctured LDPC code is given by a Gaussian distribution

$$f_{o,nonpunc}(x) = \frac{\sigma}{2\sqrt{2\pi}}\ e^{-\frac{(x-2/\sigma^2)^2}{8/\sigma^2}} = \mathcal{N}(\frac{2}{\sigma^2}, \frac{4}{\sigma^2}). \tag{6.7}$$

In the punctured case, the decoder inserts erasures (whose message content are 0) to where the bits are punctured. With puncturing rate ρ, the density of the messages observed by the decoder is in the form of mixed Gaussian and Kronecker delta function

$$f_{o,punc}(x) = (1-\rho)\cdot\mathcal{N}(\frac{2}{\sigma^2}, \frac{4}{\sigma^2}) + \rho\cdot\delta(x). \tag{6.8}$$

The rest of the procedure is the same for both cases and can be found in [14]. Figure 73 compares the thresholds (asymptotic performance) computed using density evolution of regular (non-punctured) LDPC codes and punctured LDPC codes with puncturing rate $\rho = 10\%, 20\%, 30\%$ (all with column weight 3). Apparently,

punctured codes suffer performance loss, and the effect is more evident as ρ increases. That is, for a fixed desired rate (after puncturing), it is desirable to choose the mother code such that the percentage of puncturing is as small as possible. However, this will result in a limited range of achievable code rates. It can be seen from the plot that if the desired code rate (after puncturing) is high (which is typically the case), the performance loss suffered by picking a lower rate mother code is very large and increases as the desired code rate increases. Since for an ARQ system, the probability of error during the first transmission is very important, low rate mother codes are not a good choice.

From the analysis of code ensemble and computation of the asymptotic performance, we conclude that puncturing provides a viable solution to produce RC-LDPC codes but the efficiency is limited at high rate range where the amount of puncturing is not large. This is also confirmed by computer simulations.

2. Extending

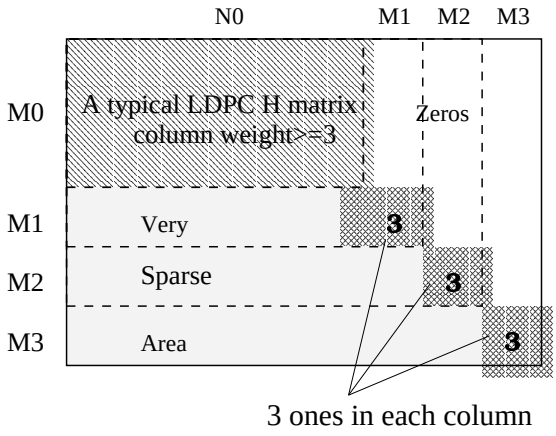
Just opposite to puncturing, the method of extending builds RC codes from high rates to low rates through the addition of more parity bits. A strong motivation for extending comes from the observation that the quality of the initial transmission is the most important to achieve high throughput in ARQ systems. In the proposed RC-LDPC code structure, the initial transmission corresponds to an *non-punctured* LDPC code. This optimizes the error probability during the first transmission F_0 in (6.3). Then, additional parity bits are added to reduce the rate in such a way that the extended code provides sufficiently good performance at the lower rate. Another motivation for using extending to build RC-LDPC codes is the decoding complexity. Unlike puncturing where all parity bits are generated at the encoder regardless of whether they will be used, extending allows bits to be generated only as needed, thus

avoiding unnecessary computations at the encoder and the decoder.

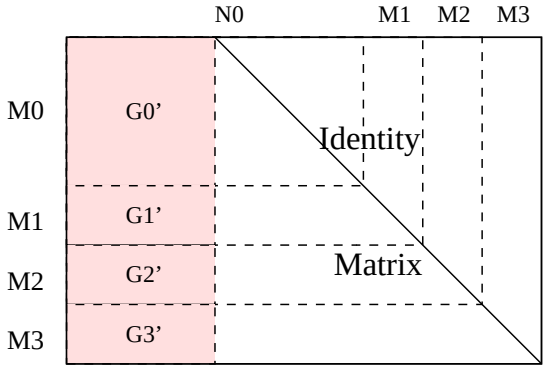
Whereas this observation is not particularly new, it is not apparent how extending can be used to realize rate compatibility for most of the block codes and trellis codes. Nonetheless, researchers have successfully used repetition, which may be deemed as the simplest type of extending, to construct rate-compatible convolutional codes [107] [108]. Fortunately, the intrinsic randomness in an LDPC code makes it possible to exploit the technique of extending. Figure 74(A) presents the proposed structure for building RC-LDPC codes using extending. The parity check matrix of the highest rate code has $M_0 = N_0 - K$ rows and N_0 columns with column weight $t \geq 3$ (upper-left part in Figure 74(A)). The parity check matrix of each lower rate code is constructed by padding M_i rows and M_i columns, until finally reaching a $(N_0 + \sum_{i=1}^{L} M_i, K)$ code after L levels of padding. A family of RC-LDPC codes of rates $R_0 > R_1 > \cdots > R_L$ thus results, where $R_i = K/(N_0 + \sum_{j=1}^{i} M_j), 1 \leq i \leq L$. To embed higher rate codewords in lower rate codewords, the upper-right part of each padding must be "0"s as shown. The squares in the bottom-right part (see Figure 74(A)) have column weight 3 to ensure the resulting parity check matrix also has column weight of at least 3. The bottom-left area is made reasonably sparse to ease the construction and to save the decoding complexity, but at least one "1" is needed for each row in order to build sufficient dependencies between the code bits of the mother code and the newly added parity bits.

The encoder structure is shown in Figure 74(B)(C). Like a conventional LDPC code, Gaussian elimination is used to derive generator matrix from the parity check matrix. It is easy to see that the generator matrices corresponding to the initial and subsequent transmissions take the form:

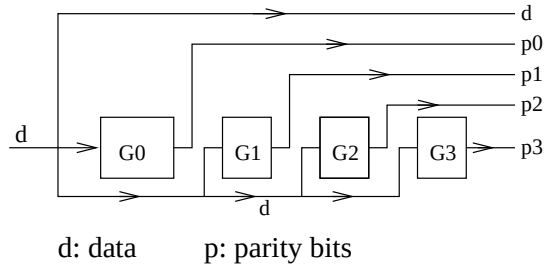
$$\Big[\,\mathbf{I}\,|\,\mathbf{G_0}\,\Big],\ \Big[\,\mathbf{I}\,|\,\mathbf{G_0}\,|\,\mathbf{G_1}\,\Big],\ \cdots,\ \Big[\,\mathbf{I}\,|\,\mathbf{G_0}\,|\,\mathbf{G_1}\,|\cdots|\,\mathbf{G_L}\,\Big]. \tag{6.9}$$



(A)



(B)



(C)

Fig. 74. Illustration of RC-LDPC codes by extending. (A). Format of parity check matrix. (B). Parity check matrix in its systematic form. (B). Encoder structure.

Table VI. Complexity comparison: puncturing vs extending.

Encoding		
	puncturing	extending
XOR	$(K^2-K)(1/R_0-1)$	$(K^2-K)(1/R_i-1)$
OR	$K^2(1/R_0-1)$	$K^2(1/R_i-1)$
Decoding (per iteration)		
	puncturing	extending
addition	$3tK/R_0$	$3t'K/R_i$
table-lookup	$2tK/R_0$	$2t'K/R_i$

$t=3, \quad 3\leq t'\leq 3.3, \quad R_i=K/(N_0+\sum_{j=1}^{i} M_j)$.

where $\mathbf{I}$ is the identity matrix of size K, and $\mathbf{G}_i$ has dimensionality $K \times M_i$. This is important since this means that groups of parity bits can be generated independently and parity bits from the ith group is not required to decode the any $j_{th} < i_{th}$ transmission.

The decoder of RC-LDPC codes (constructed by extending) implements the structure of H_{all} matrix. When decoding the subcode with rate $R_j = K/(N_0 + \sum_{i=1}^{j} M_i), j = 0, 1, \cdots, L$, only the relevant bits and checks (i.e., the top $\sum_{i=0}^{k} M_i$ rows and the left $N_0 + \sum_{i=1}^{k} M_i$ columns) will be used for (extrinsic) information exchange. As rate decreases, more parity checks and bits join the message exchange process, and, through the increased information and enhanced dependencies, offer a stronger error correction capability.

As mentioned above, one advantage of extending is its low complexity. Table VI compares the encoding and decoding complexity of RC-LDPC codes constructed using puncturing and extending. As can be seen, puncturing requires a fixed complexity

regardless of channel conditions, whereas the complexity of extending reduces as channel conditions improve. Most of the time extending involves less decoding complexity than puncturing.

Figures 75 and 76 show the BER and WER performance of RC-LDPC codes constructed using extending and puncturing, respectively. Simulation results match well with the analytical result that puncturing constructs efficient RC-LDPC codes at high rates and extending at low rates. In Figure 75, the performance of a set of RC-LDPC codes of rates $\frac{4}{8}\backslash\frac{4}{9}\backslash\frac{4}{10}\backslash\frac{4}{11}\backslash\frac{4}{12}$ constructed by extending are plotted in solid lines. The mother code has rate 1/2 and column weight 3, and performs on par with any rate-1/2 regular LDPC code of the same parameters. The lowest rate code in this case is an irregular LDPC code with average column weight 3.3. As rate reduces, the performance improves steadily, until finally reaching the capacity of the specific rate-$\frac{1}{3}$ irregular LDPC code with an average column weight of 3.3. Comparatively, a rate $\frac{1}{2}$ LDPC code constructed through puncturing from a rate $\frac{1}{3}$ mother code (dashed line) performs about 0.8 dB worse than its peer in the "extending family". This shows that extending should be used to construct good RC-LDPC codes at low rates. The situation is just the opposite at high rates As shown in Figure 76, a family of $\frac{16}{20}\backslash\frac{16}{21}\backslash\frac{16}{22}\backslash\frac{16}{23}\backslash\frac{16}{24}$ RC-LDPC codes constructed using puncturing (solid lines) demonstrates encouraging performance, where those constructed using extending (dashed lines) fail to offer incremental improvement in performance. We observe that. at high rates, the parity check matrix of the "extending family" has fairly large weights in the first M_0 rows (corresponding to the mother code), but very low weights in the padded rows. It is shown that the row weight profile of an LDPC code should be "concentrated", i.e. all rows should have almost the same weights, in order to achieve good performance [14]. This huge discrepancy among row weights may be the explanation why extending leads to only marginal performance

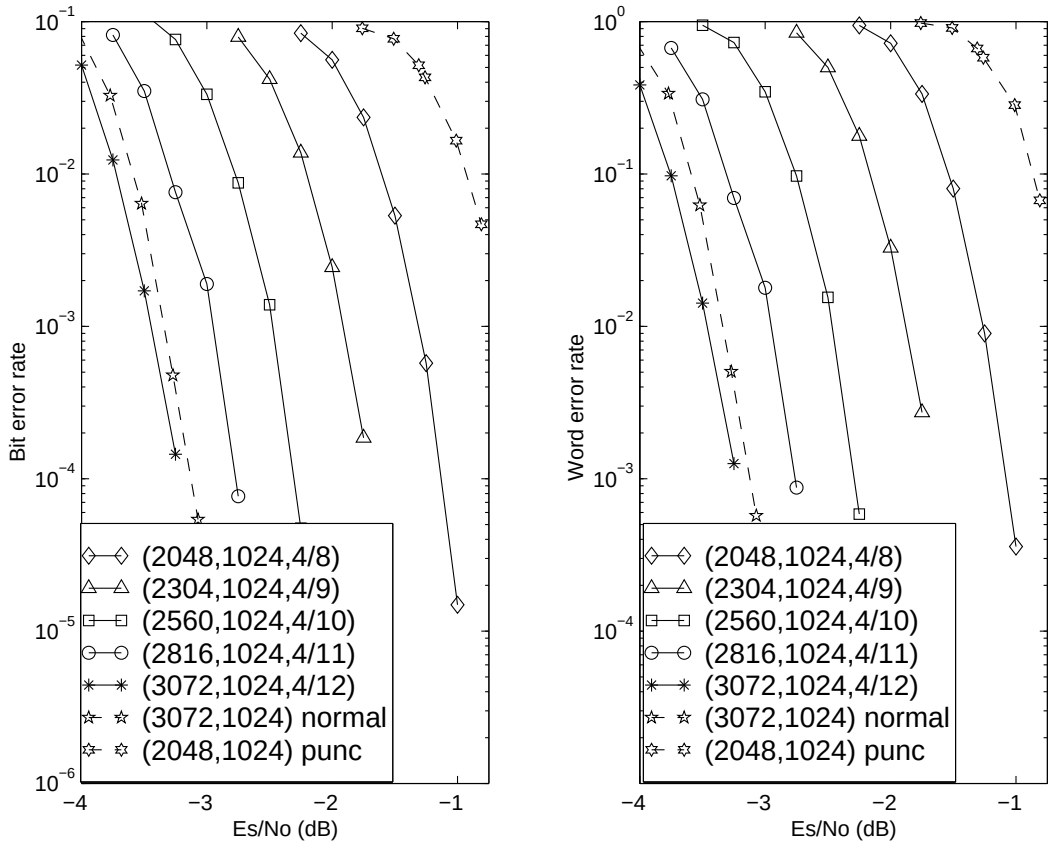


Fig. 75. Efficient RC-LDPC codes constructed through extending (at low rate range). (Solid lines: RC-LDPC codes by extending; from right to left: a regular column-weight-3 (2048, 1024, 4/8) LDPC mother code, 256 parities extended, 512 parities extended, 768 parities extended, 1024 parities extended (final average column weight 3.3). Dashed lines: RC-LDPC codes by puncturing (for comparison); from left to right: a normal column-weight-3 (3072, 1024, 4/12) LDPC mother code, a (2048, 1024, 4/8) code by puncturing 1024 parities out of the mother code.)

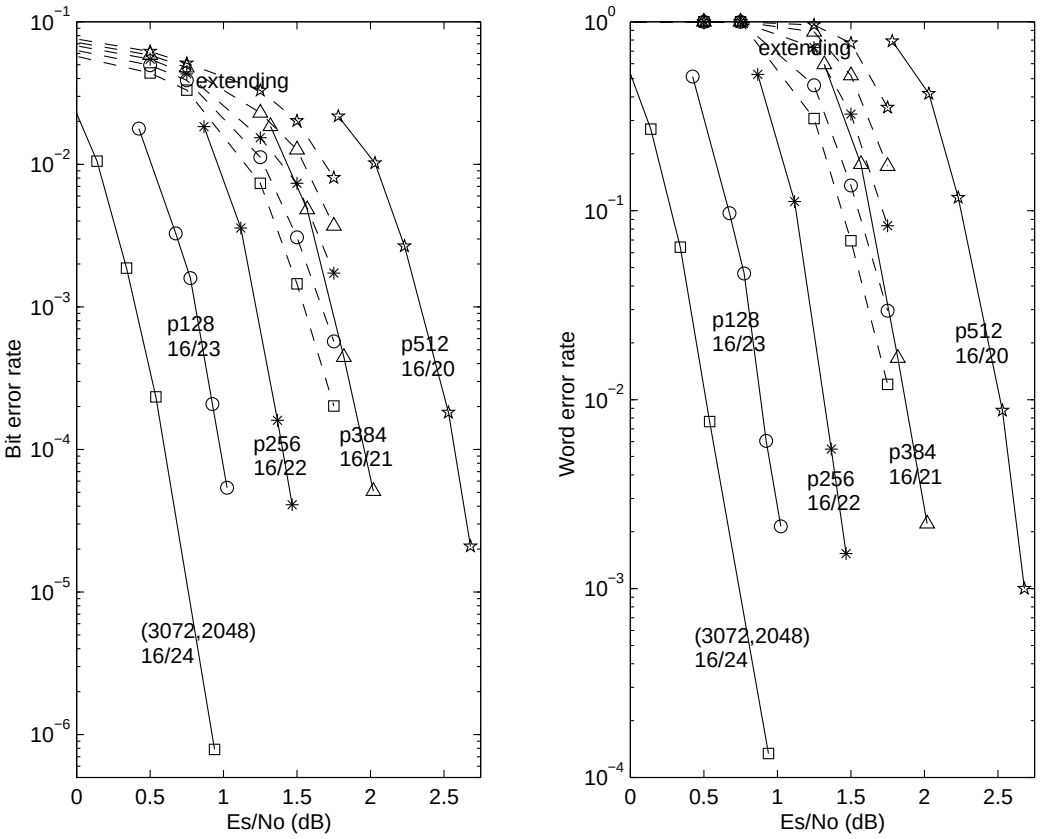


Fig. 76. Efficient RC-LDPC codes constructed through puncturing (at high rate range). (Solid lines: RC-LDPC codes by puncturing; from left to right: a regular column-weight-3 (3072, 2048, 16/24) LDPC mother code, 128 parities punctured, 256 parities punctured, 384 parities punctured, 512 parities punctured. Dashed lines: RC-LDPC codes by extending (for comparison); from right to left: a normal column-weight-3 (2560, 2048, 16/20) LDPC mother code, 128 parities extended, 256 parities extended, 384 parities extended, 512 parities extended.)

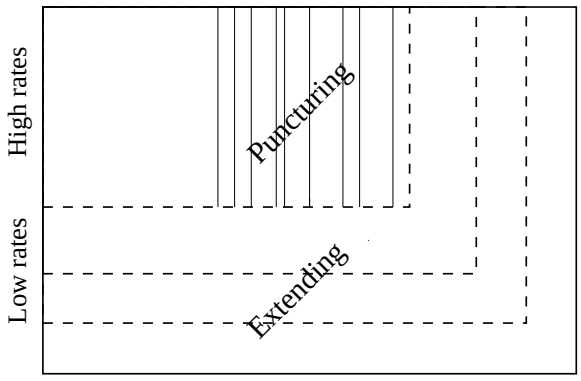


Fig. 77. Overall system model for RC-LDPC codes using both puncturing and extending.

improvement, since the padded rows apparently have little influence on the overall code.

3. Overall Structure of RC-LDPC Codes

With the above discussion and simulations, it follows naturally that efficient RC-LDPC codes could take advantage of both approaches. Figure 77 presents the overall system structure, where the proper boundary rate guiding which of the two techniques to use should be somewhere between rate-$\frac{1}{2}$ and $\frac{2}{3}$. As an example, we construct a family of $K = 1024$ RC-LDPC codes with rates ranging from $\frac{8}{11}$ to $\frac{8}{20}$ (Figure 78). The mother code is a column weight 3, rate-$\frac{8}{14}$ regular LDPC code. Puncturing is used to to get rates $\frac{8}{13}\backslash\frac{8}{12}\backslash\frac{8}{11}$ and extending to get rates $\frac{8}{15}\backslash\frac{8}{16}\backslash\frac{8}{17}\backslash\frac{8}{18}\backslash\frac{8}{19}\backslash\frac{8}{20}$. As can be seen, each individual code provides good error correction capability, and they collectively offer a steady improvement in performance with code compatibility. For comparison purpose, also presented are conventional LDPC codes of rate $\frac{8}{12}$ and $\frac{8}{20}$ (dashed lines). As can be seen, each code in the family provide powerful error correction capability and the performance increases steadily at each retransmission. In the extending case, since the structure is devised aiming at rate-compatibility, no specific effort is made

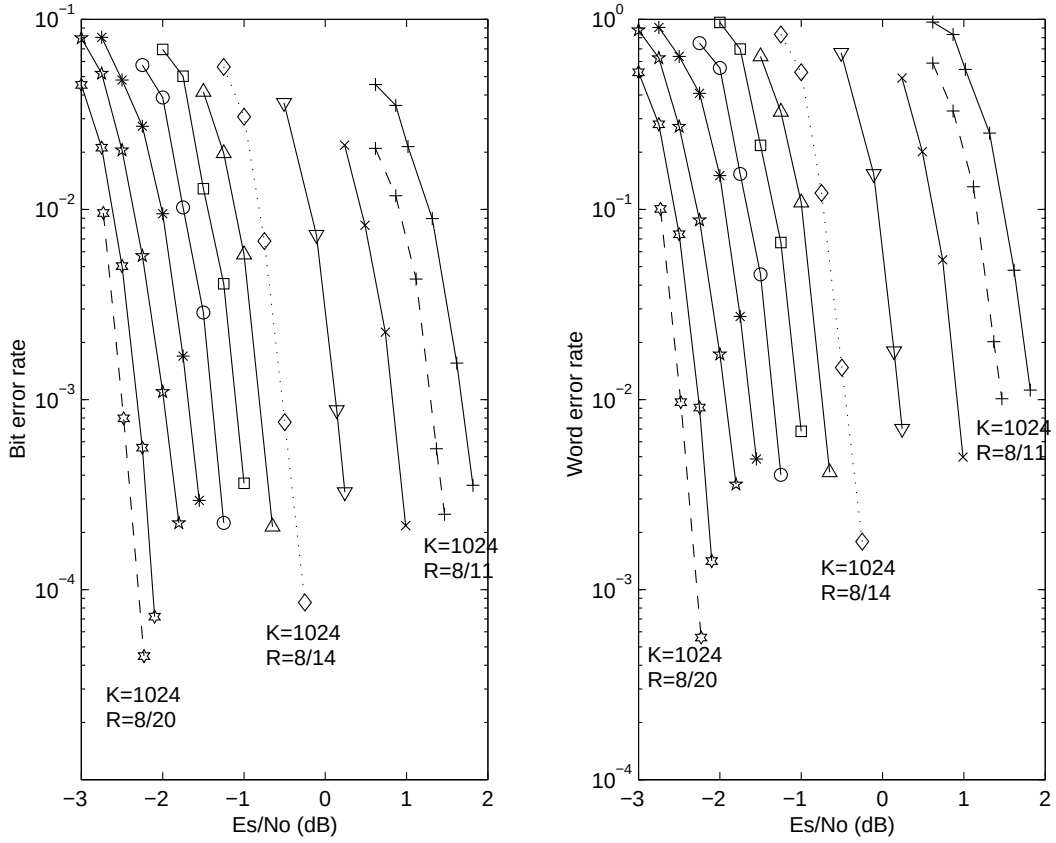


Fig. 78. Performance of RC-LDPC codes using both puncturing and extending. (User data block size K=1024. Dotted line: a regular rate 8/14 LDPC code (mother code). Solid lines to the left of the dotted line are codes constructed by extending; from right to left: rate 8/15, 8/16, 8/17, 8/18, 8/19, 8/20. Solid lines to the right of the dotted line are codes constructed by puncturing; from left to right: rate 8/13, 8/12, 8/11. Dashed lines: rate 8/20 and 8/11 regular (non-punctured) LDPC codes for comparison purpose.)

to optimize the performance of lower rates. Nonetheless, simulations indicate that the performance compromise is very reasonable. As shown in Figure 78, after 6 layers of extending, the performance of the rate-8/20 LDPC code constructed by extending (leftmost solid line) is only less than 0.2 dB worse than a normal LDPC code of the same parameters (dashed line).

The encouraging performance of RC-LDPC codes has opened possibility for capacity-approaching ARQ/FEC systems. As an example, we evaluate the proposed ARQ/FEC system using the family of RC-LDPC codes presented in Figure 78. The throughput as defined in (6.1) is computed and plotted in Figure 79. Also shown is the throughput of ARQ systems using rate-compatible turbo codes in [43] [44]. We see that the proposed LDPC-ARQ system has throughput efficiency around 1 dB away from the capacity limit, which is on par with turbo-ARQ systems. Yet, rate compatible LDPC codes have less decoding complexity than rate compatible turbo codes. In our example, the retransmission packet size is pretty small (128 bits), hence the reduction in rate required for adaptation is more gradual than most of the incremental redundancy ARQ systems in literature, and the capacity line is therefore quite smooth rather than "staircase-like". This is desired for maximizing throughput. However, in real applications, the retransmission sizes need to be balanced against the overhead of decoding complexity, the volume of traffic on the feedback channel and the delay caused by subsequent retransmissions. In case of need, several small retransmission packets may be combined as one to speed the process.

D. Summary

Efficient rate compatibility and adaptivity can be achieved from LDPC codes if the family of codes is carefully designed. The main result in this paper is that in order to

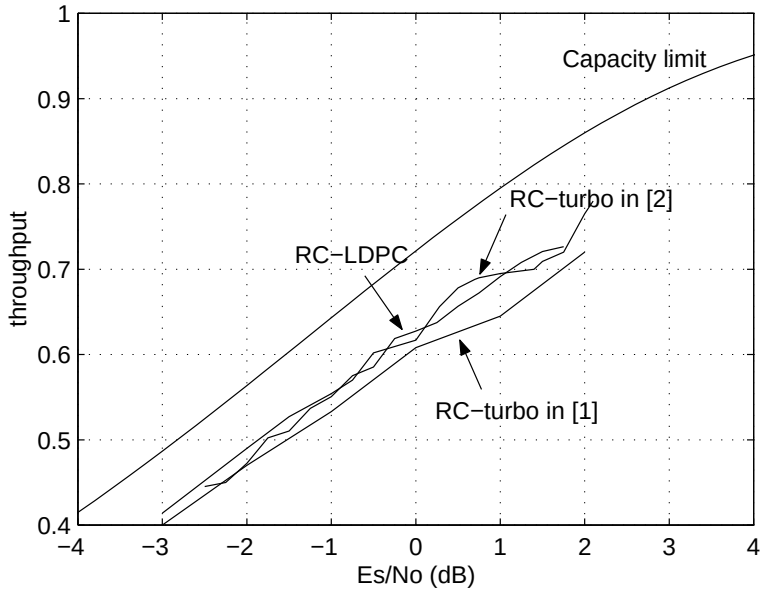


Fig. 79. Throughput of the proposed ARQ system using RC-LDPC codes (with comparison to turbo-ARQ systems). (Throughput are presented at 0.25 dB increment of E_s/N_0 for RC-LDPC, 0.5 dB increment for turbo-ARQ in [44] and 1 dB increment for turbo-ARQ [43].)

obtain a good RC-LDPC code with a wide range of rates $R_1 < R_2 < \ldots R_M$, it is not a wise strategy to use a mother code with rate R_1 and puncture to obtain the other rates (which is the conventional practice with BCH codes, convolutional codes and turbo codes). A special construction for LDPC codes has been proposed which uses a mother code of rate R_j (closer to R_M); higher rates are obtained via puncturing and lower rates through a novel extending technique that has been discussed. The proposed LDPC-ARQ system can achieve near capacity throughput, and is appealing to a wide variety of packet data applications, where powerful codes are required, feedback channels are available and latency due to retransmission overhead is acceptable.

CHAPTER VII

CONCLUSION

This dissertation is dedicated to the design, analysis and evaluation of high-performance and low-complexity error correction codes. We are primarily interested in regular codes whose structures can be represented in Tanner graphs and which can be decoded using soft iterative message-passing algorithms.

The first part of the dissertation is focused on coding theory and code design. Specifically, we have proposed, analyzed and bench-marked a class of simple, bandwidth- and power-efficient codes named product accumulate codes or PA codes, and have extended them to the generalized product accumulate codes or GPA codes (Chapter II).

The second part of the dissertation is devoted to the application and evaluation of PA codes to other high-performance codes like LDPC codes, TPC/SPC codes in a variety of practical scenario covering wireless communications, optical fiber communications, digital data storage systems and packet data networks. For land mobile wireless communications, we have investigated the coherent and noncoherent detection of PA codes on flat Rayleigh fading channels, and have extended the discussion to the general case where the outer code can be any LDPC code. A simple joint differential detection and decoding receiver is discussed which is shown to perform within 1 dB from the coherent case with very little additional complexity and bandwidth expansion. We conduct EXIT analysis on a couple of interesting issues concerning differential coding. Furthermore, we have proposed a convergence-constraint method that is useful for designing good LDPC ensemble matched to the differential code (or in general any receiver) (Chapter III). For long-haul optical fiber communications, we have investigated high-rate, long PA codes with parallel turbo codes as a compar-

ison study. On-off signaling on three types of channel models is investigated, where analytical bounds and simulations show that the bit error rate of PA codes is very impressive with error floors as low as 10^{-9} (Chapter IV). For high-density digital data storage systems, we have investigated single parity check turbo product codes and LDPC codes on ideal partial response and PR-equalized Lorentzian channels. A number of issues concerning binary precoding, iterative decoding scheduling and optimization, asymptotic thresholds of coded ISI channels, and recording density and PR targets are investigated. Complexity, bit error rate and error statistics are evaluated which shows that the simple, regular TPC/SPC codes seem a promising candidate for future magnetic recording devices (Chapter V). For packet data networks, we have investigated a hybrid automatic repeat request system using LDPC codes. We have constructed efficient rate compatible LDPC codes using both puncturing and extending, and have shown that the resulting RC-LDPC coded ARQ system can achieve throughput around 1 dB from the capacity, which is as good as turbo-ARQ system yet with less decoding complexity (Chapter VI).

The major contributions of this work include:

1. We have invented a class of low-complexity, high-performance, high-rate, provably "good", regular codes, named product accumulate codes, which are simple to construct, simple to implement and simple to analyze. Computer simulations show that this class of codes perform comparable to turbo or LDPC codes on a number of communication channels including AWGN, flat Rayleigh fading, Chi-square and asymmetric Gaussian channels. Analytical bounds and thresholds show that these codes are capable of performance within a few tenths a dB from the Shannon limit on AWGN and flat Rayleigh fading channels. In this, we have shown PA codes to be a low-cost alternative to turbo or LDPC codes

for a variety communication applications. We have also extended PA codes to generalized product accumulate codes or GPA codes, and show that GPA codes are most desirable in systems that require rate adaptivity.

2. We have proposed a convergence-constraint method to design good LDPC ensembles matched with differential coding (and in general any receiver). We show that the proposed method is efficient and effective. The proposed method is a useful extension of the conventional threshold-constraint design method, but has a far-reaching implication and application since it can explicitly take into account the property and the imperfectness of the receiver.

3. We have proposed a graph-based message-passing algorithm (or the sum-product algorithm) for decoding the rate-1 recursive convolutional code $1/(1+D)$. We have shown that using a specific serial update procedure, the proposed sum-product algorithm is equivalent to the conventional BCJR algorithm and that its low-complexity approximation, the min-sum algorithm, is equivalent to the Max-log-MAP algorithm, yet require only about one tenth the complexity.

4. We have proposed a systematic way of constructing rate compatible LDPC codes for use in hybrid ARQ systems. We show efficient RC-LDPC codes of large rate range can be constructed and, when used with smart ARQ systems, can achieve near capacity throughput.

5. We have revealed that the popular practice of inserting pilot symbols to periodically terminate the trellis incurs an intrinsic loss in code capacity and is likely to cause high error floors and severe BER performance loss to the overall code performance. A better way of inserting pilot symbols is suggested which separates pilot symbols from the trellis structure.

REFERENCES

[1] R. W. Hamming, "Error detecting and correction codes," *Bell Sys. Tech. Journ.*, vol. 29, pp. 147–160, 1950.

[2] A. Hocquenghem, "Codes correcteurs d'erreurs," *Chiffres*, vol. 2, pp. 147–156, 1959.

[3] B. C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Information and Control*, vol. 3, pp. 68–79, Mar. 1960.

[4] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *SIAM Journ. Applied Math.*, vol. 8, no. 300-304, pp. 1960, 1960.

[5] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 260–269, Apr. 1989.

[6] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284–287, Mar. 1974.

[7] G. D. Forney, *Concatenated Codes*, MIT Press, Cambridge, MA, 1966.

[8] C. Berrou, A. Glavieux, and P. Thitimasjshima, "Near Shannon limit error-correcting coding and decoding: turbo-codes (1)," in *Proc. IEEE Intl. Conf. Commun.*, Geneva, Switzerland, May 1993, pp. 1064–1070.

[9] R. G. Gallager, *Low-density parity-check codes*, MIT Press, Cambridge, MA, 1963.

[10] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, No. 2, pp. 399–431, Mar. 1999.

[11] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Analysis of low density codes and improved designs using irregular graphs," in *Proc. ACM Symposium*, 1998, pp. 249–258.

[12] T. J. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, pp. 599–618, Feb. 2001.

[13] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: turbo-codes," *IEEE Trans. Commun.*, vol. COM-44, pp. 1261–1271, Oct. 1996.

[14] T. J. Richardson, M. A. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory*, pp. 619–637, Feb. 2001.

[15] S. Benedetto and G. Montorsi, "Unveiling turbo codes: some results on parallel concatenated coding schemes," *IEEE Trans. Inform. Theory*, vol. IT-42, pp. 409–428, Mar. 1996.

[16] S. Benedetto, G. Montorsi, D. Divsalar, and F. Pollara, "Serial concatenation of interleaved codes: performance analysis, design, and iterative decoding," *JPL TDA Progress Report*, vol. 42-126, Aug. 1996.

[17] D. Divsalar and F. Pollara, "Serial and hybrid concatenation codes with applications," in *Proc. Intl. Symp. on Turbo Codes and Related Topics*, Brest, France, Sept. 1997, pp. 80–87.

[18] P. Elias, "Error-free coding," *IRE Trans. Inform. Theory*, pp. 29–37, Sept. 1954.

[19] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inform. Theory*, vol. IT-42, pp. 429–445, Mar. 1996.

[20] G. Caire and C. Taricco, "Weight distribution and performance of the iterated product of single-parity-check codes," in *Proc. GLOBECOM*, 1994, pp. 206–211.

[21] D. Divsalar, H. Jin, and R. J. McEliece, "Coding theorems for "turbo-like" codes," in *Proc. Allerton Conf. Commun. and Control*, Sept. 1998, pp. 201–210.

[22] H. Jin, A. Khandekar, and R. J. McEliece, "Irregular repeat-accumulate codes," in *Proc. 2nd Intl. Symp. on Turbo Codes and Related Topics*, Brest, France, Sept. 2000, pp. 1–5.

[23] J. Li, K. R. Narayanan, and C. N. Georghiades, "A class of linear-complexity, soft-decodable, high-rate, 'good' codes: construction, properties and performance," in *Proc. Intl. Symp. Inform. Theory*, Washington DC, June 2001, p. 122.

[24] J. Li, K. R. Narayanan, and C. N. Georghiades, "Product accumulate codes: a class of capacity-approaching, low-complexity codes," submitted to *IEEE Trans. Inform. Theory*, 2001.

[25] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: performance analysis, design, and iterative decoding," *IEEE*

Trans. Inform. Theory, vol. 44, No. 3, pp. 909–926, May 1998.

[26] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proc. BLOBECOM*, 1995, pp. 1009–1013.

[27] J. Li, E. Kurtas, K. R. Narayanan, and C. N. Georghiades, "On the performance of turbo product codes over partial response channels," *IEEE Trans. Magnetics*, pp. 1932–1934, July 2001.

[28] A. Glaviex, C. Laot, and J. Labat, "Turbo equalization over a frequency selective channel," in *Proc. 1st Intl. Symp. on Turbo Codes and Related Topics*, Brest, France, Sept. 1997, pp. 96–102.

[29] K. R. Narayanan and G. L. Stuber, "A serial concatenation approach to iterative demodulation and decoding," *IEEE Trans. Commun.*, vol. 47, pp. 956–961, July 1999.

[30] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 533–547, Sept. 1981.

[31] N. Wiberg, "Codes and decoding on general graph," Ph.D. dissertation, Linkoping University, Linkoping, Sweden, 1996.

[32] 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. Inform. Theory*, pp. 657–670, Feb. 2001.

[33] T. J. Richardson and R. L. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, No. 2, pp. 638–656, Feb. 2001.

[34] Y. Kou, S. Lin, and M. Fossorier, "Low-density parity-check codes based on finite geometries: a rediscorty and new results," *IEEE Trans. Inform. Theory*, pp. 2711–2736, Nov. 2001.

[35] D. J. C. MacKay and M. C. Davey, "Evaluation of Gallager codes for short block length and high rate applications," in *Proc. IMA Workshop on Codes, Systems and Graphical Models*, 1999, pp. 113-130; also available http://www.keck.ucsf.edu/∼mackay/seagate.ps.gz.

[36] S. J. Johnson and S. R. Weller, "Construction of low-density parity-check codes from Kirkman triple systems," in *Proc. GLOBECOM*, San Antonio, Nov. 2001, pp. 770–774.

[37] B. Basic, "Structured iteratively decodable codes based on Steiner systems and their application in magnetic recording," in *Proc. BLOBECOM*, San Antonio, Nov. 2001, pp. 2954–2960.

[38] R. M. Pyndiah, "Near-optimum decoding of product codes: block turbo codes," *IEEE Trans. Commun.*, pp. 1003–1010, Aug. 1998.

[39] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inform. Theory*, vol. 18, pp. 170–182, Jan. 1972.

[40] J. Li, K. R. Narayanan, E. Kurtas, and C. N. Georghiades, "On the performance of high-rate TPC/SPC and LDPC codes over partial response channels," *IEEE Trans. Commun.*, May 2001.

[41] D. Divsalar, "A simple tight bound on error probability of block codes with application to turbo codes," *TMO Progress Report 42-139*, Nov. 1999.

[42] J. Li, K. R. Narayanan, C. N. Georghiades, and E. Kurtas, "Thresholds for iterative equalization of partial response channels using density evolution," in *Proc. Intl. Symp. Inform. Theory*, Washington DC, June 2001, p. 73.

[43] D. N. Rowitch and L. B. Milstein, "Rate compatible punctured turbo (RCPT) codes in a hybrid FER/ARQ system," in *Proc. BLOBECOM*, Phoenix, AZ, Nov. 1997, pp. 55–59.

[44] R. Mantha and F. R. Kschischang, "A capacity-approaching hybrid ARQ scheme using turbo codes," in *Proc. BLOBECOM*, 1999, pp. 2341–2345.

[45] P.-P. Sauvé, A. Hunt, S. Crozier, and P. Guinand, "Hyper-codes: high-performance, low-complexity codes," in *Proc. 2nd Intl. Symp. on Turbo Codes and Related Topics*, Brest, France, Sept. 2000, pp. 121–124.

[46] M. Peleg, I.Sason, S. Shamai, and A. Elia, "On interleaved, differentially encoded convolutional codes," *IEEE Trans. Inform. Theory*, vol. 45, pp. 2572–2582, Nov. 1999.

[47] B. J. Frey, *Graphical models for machine learning and digital communication*, MIT Press, Cambridge, MA, 1998.

[48] Jr G. D. Forney, "Codes on graphs: normal realizations," *IEEE Trans. Inform. Theory*, pp. 520–548, Feb. 2001.

[49] G. Poltyrev, "Bounds on the decoding error probability of binary linear codes via their spectra," *IEEE Trans. Inform. Theory*, pp. 1284–1292, July 1994.

[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," in *Proc. IEEE Intl. Symp. Inform. Theory*, Cambridge, MA, Aug 1998, p. 29.

[51] T. M. Duman and M. Salehi, "New performance bounds for turbo codes," *IEEE Trans. Commun.*, vol. 46, pp. 717–723, June 1998.

[52] B. Hughes, "On the error probability of signals in additive white Gaussian noise," *IEEE Trans. Inform. Theory*, vol. 37, pp. 151–155, Jan. 1991.

[53] Jr. G. C. Clark and J. B. Cain, *Error-correction coding for digital communications*, Plenum Press, New York, 1981.

[54] K. R. Narayanan, J. Li, and C. N. Georghiades, "Product accumulate codes: properties and performance," in *Proc. IEEE Inform. Theory Workshop*, Cairns, Australia, 2001, pp. 21–23.

[55] H. Jin and R. J. McEliece, "RA codes achieve AWGN channel capacity," in *Proc. 13th Intl. Symp. Applied Algebra, Algebraic Algorithms, and Eorror-Correcting Codes*, 1999, pp. 10–18.

[56] N. Kahale and R. Urbanke, "On the minimum distance of parallel and serially concatenated codes," in *Proc. Intl. Symp. Inform. Thoery*, Cambridge, MA, Aug. 1998, p. 31.

[57] L. Ping and K. Y. Wu, "Concatenated tree codes: a low-complexity, high-performance approach," *IEEE Trans. Inform. Theory*, pp. 791–799, Feb. 2001.

[58] D. Divsalar and E. Biglieri, "Upper bounds to error probabilities of coded systems beyond the cutoff rate," in *Proc. Intl. Symp. Inform. Theory*, Sorrento, Italy, June 2000, p. 288.

[59] M. C. Valenti and B. D. Moerner, "Iterative channel estimation and decoding of pilot symbol assisted turbo codes over flat-fading channels," *IEEE Jour. on Selected Areas in Commun*, vol. 19, pp. 1697–1705, Sept. 2001.

[60] P. Hoeher and J. Lodge, ""turbo dpsk": iterative differential psk demodulation and channel decoding," *IEEE Trans. Commun.*, vol. 47, No. 6, pp. 837–843, 1999.

[61] B. Zhao and M. C. Valenti, "Per-survivor based detection of DPSK modulated high rate turbo codes over rayleigh fading channels," in *Proc. Asilomar Conf. on Signals, Systems and Computers*, 2001, pp. 1026–1030.

[62] D. Divsalar and M. K. Simon, "Maximum-likelihood differential detection of uncoded and trellis coded amplitude phase modulation over AWGN and fading channels-metrics and performance," *IEEE Trans. Commun.*, pp. 76–89, Jan. 1994.

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

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

[65] P. Ho and D. Fung, "Error performance of multiple symbol differential detection of PSK signals transmitted over correlated Rayleigh fading channels," *IEEE Trans. Commun.*, vol. 40, pp. 1566–1569, Oct. 1992.

[66] J. Hou, P. H. Siegel, and L. B. Milstein, "Performance analysis and code optimization of low density parity-check codes on rayleigh fading channels," *IEEE Jour. Sele. Area Commun.*, vol. 19, No. 2, pp. 924–934, May 2001.

[67] I. S. Gradshteyn and I. M. Ryzhik, *Tables of Integrals, Series and Products*, Academic, New York, NY, 1980.

[68] G. L. Stuber, *Principles of mobile communication*, Kluwer Academic Publishers, Cambridge, MA, 1996.

[69] M. K. Simon and M. S. Alouini, *Digital communication over fading channels*, John Wiley and Sons, New York, NY, 2000.

[70] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: a model and two properties," in *Proc. Conf. Inform. Sciences and Systems*, Princeton, NJ, March 2002.

[71] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, No. 10, pp. 1727–1737, Oct. 2001.

[72] "Help for LdpcOpt – AWGN channels," http:// lthcwww.epfl.ch/ research/ ldpcopt/awgnhelp.php, accessed on October 28, 2002.

[73] K. R. Narayanan, I. Altunbas, and R. Narayanaswami, "On the design of LDPC codes for MSK," in *Proc. IEEE GLOBECOM*, San Antonio, TX, Nov. 2001, pp. 1011–1015.

[74] K. R. Narayanan, "Effect of precoding on the convergence of turbo equalization for partial response channels," *IEEE Jour. on Sele. Area Commun.*, vol. 19, pp. 686–698, April 2001.

[75] A. Shokrollahi and R. Storn, "Design of efficient erasure codes with differential evolution," in *Proc. IEEE Intl. Symp. Inform. Theory*, Sorrento, Italy, June 2000, p. 5.

[76] J. L. Pamart, E. Lefranc, S. Morin, G. Balland, Y. C. Chen, T. M. Kissell, and J. L. Miller, "Forward error correction in a 5 gbit/s 6400 km edfa based system," *IEEE Electronics Letters*, vol. 30, No. 4, pp. 342–343, Feb. 1994.

[77] S. Yamamoto, H. Takahira, and M. Tanaka, "5 gbps optical transmission terminal equipment using forward error correction code and optical amplifier," *IEEE Electronics Letters*, vol. 30, No. 3, pp. 254–255, Feb 1994.

[78] H. Kidorf, N. Ramanujam, I. Hayee, M. Nissov, J. Cai, B. Pedersen, A. Puc, and C. Rivers, "Performance improvement in high capacity, ultra-long distance, wdm systems using forward error correction codes," in *OFC/IOOC Tech. Digest*, Baltimore, Mar. 2000, pp. ThS3–1–ThS3–3.

[79] A. Puc, F. Kerfoot, A. Simons, and D. L. Wilson, "Concatenated fec experiment over 5000 km long straight line wdm test bed," in *OFC/IOOC Tech. Digest*, San Diego, Feb. 1999, pp. ThQ6–1–THQ6–3.

[80] O. Ait Sab, H. Yamauchi, T. Inoue, , and K. Goto, "Performance improvement of highly nonlinear long-distance optical fiber transmission system using novel high gain forward error correcting code," in *OFC/IOOC Tech. Digest*, Baltimore, Mar. 2000, pp. ThS5–1–ThS5–3.

[81] H. Taga, H. Yamauchi, T. Inoue, and K. Goto, "Performance improvement of highly nonlinear long-distance optical fiber transmission system using novel high gain forward error correcting code," in *OFC/IOOC Tech. Digest*, Anaheim, Mar. 2001, pp. TuF3–1–TuF3–3.

[82] O. Ait Sab, "Fec techniques in submarine transmission systems," in *OFC/IOOC Tech. Digest*, Anaheim, Mar. 2001, pp. TuF1–1–TuF1–3.

[83] D. Marcuse, "Derivation of analytical expressions for the bit-error probability in lightwave systems with optical amplifiers," *Jour. Lightwave Tech.*, vol. 8, pp. 1816–1823, Dec. 1990.

[84] P. A. Humblet and M. Azizoğlu, "On the bit error rate of lightwave systems with optical amplifiers," *Jour. Lightwave technology*, vol. 9, pp. 1576–1582, Nov. 1991.

[85] Y. Cai, J. M. Morris, T. Adali, and C. R. Menyuk, "On turbo code decoder performance in optical fiber communication systems with dominating ASE noise," submitted to *Jour. Lightwave Tech.*

[86] Y. Cai, N. Ramanujam, J. M. Morris, T. Adali, G. Lenner, A. B. Puc, and A. Pilipetskii, "Performance limit of forward error correction codes in optical fiber communications," in *OFC/IOOC Tech. Digest*, Anaheim, Mar. 2001, p. TuF2.

[87] D. A. Shnidman, "The calculation of the probability of detection and the generalized Marcum Q-function," *IEEE Trans. Inform. Theory*, vol. 35, pp. 389–400, Mar. 1989.

[88] O. K. Tonguz and L. G. Kazovsky, "Theory of direct-detection lightwave receivers using optical amplifiers," *Jour. Lightwave Tech.*, vol. 9, pp. 174–181, Feb. 1991.

[89] K. Immink, "Coding techniques for the noisy magnetic recording channel: a state-of-the-art report," *IEEE Trans. Comm*, pp. 413–419, May 1989.

[90] T. M. Duman and E. Kurtas, "Comprehensive performance investigation of

turbo codes over high density magnetic recording channels," in *Proc. GLOBECOM*, 1999, pp. 744–748.

[91] T. Souvignier, A. Friedmann, P. Siegeland R. Swanson M. Oberg, and J. Wolf, "Turbo decoding for PR4: parallel vs. serial concatenation," in *Proc. Intl. Conf. Commun.*, Brest, France, June 1999, pp. 1638–1642.

[92] W. Ryan, L. McPheters, and S. McLaughlin, "Combined turbo coding and turbo equalization for PR4-equalized lorentzian channels," in *Proc. Conf. Inform. Sci. and Sys.*, Princeton, NJ, Mar 1998, pp. 489–493.

[93] M. Oberg and P. H. Siegel, "Performance analysis of turbo-equalized dicode partial response channel," in *Proc. Allerton Conf. Commun., Control and Computing*, Sept. 1998, pp. 230–239.

[94] J. Li, E. Kurtas, K. R. Narayanan, and C. N. Georghiades, "On the performance of turbo product and LDPC codes over partial-response channels," in *Proc. Intl. Conf. Commun.*, Helsinki, Finland, June 2001, pp. 2176–2183.

[95] H. Song, R. M. Todd, and J. R. Cruz, "Performance of low-density parity-check codes on magnetic recording channels," in *Proc. 2nd Intl. Symp. on Turbo Codes and Related Topics*, Brest, France, Sept. 2000, pp. 395–398.

[96] K. A. S. Immink, P. H. Siegel, and J. K. Wolf, "Codes for digital recorders," *IEEE Trans. Inform. Thoery*, vol. 44, No. 6, pp. 2260–2299, Oct. 1998.

[97] J. L. Fan and J. M. Cioffi, "Constrained coding techniques for soft decoders," in *Proc. GLOBECOM*, 2000, pp. 723–727.

[98] M. Oberg and P. H. Siegel, "Parity check codes for partial response channels," in *Proc. GLOBECOM*, 1999, pp. 2341–2345.

[99] L. McPheters, S. McLaughlin, and K. Narayanan, "Precoded PRML, serial concatenation, and iterative (turbo) decoding for digital magnetic recording," *IEEE Trans. Magn.*, pp. 2325–2327, Sept 1999.

[100] S. Lin and P. S. Yu, "Hybrid ARQ scheme with parity retransmission for error control of satellite channels," *IEEE Trans. Commun.*, vol. 30, pp. 1701–1719, July 1982.

[101] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Trans. Commun.*, vol. Com-36, pp. 389–400, 1988.

[102] J. B. Cain, G. C. Clark Jr., and J. M. Geist, "Punctured convolutional codes of rate (n-1)/n and simplified maximum likelihood decoding," *IEEE Trans. Inform. Theory*, vol. IT-25, pp. 97–100, Jan. 1996.

[103] A. S. Barbulescu and S. S.Pietrobon, "Rate compatible turbo codes," *Electronic Letters*, vol. 31, pp. 535–536, Mar. 1995.

[104] T. Ji and W. E. Stark, "Turbo-coded ARQ schemes for DS-CDMA data networks over fading and shadowing channels: throughput, delay and energy efficiency," *IEEE Jour. Sele. Areas Commun.*, pp. 1355–1364, Aug 2000.

[105] D. Chase, "Code combining - a maximum-likelihood decoding approach for combining an arbitrary number of noisy packets," *IEEE Trans. Inform. Theory*, vol. COM-33, pp. 385–393, May 1985.

[106] B. A. Harvey and S. B. Wicker, "Packet combining systems based on the Viterbi decoder," *IEEE Trans. Commun.*, vol. 42, pp. 1544–1557, Feb./Mar./Apr. 1994.

[107] S. Kallel and D. Haccoun, "Generalized type II hybrid ARQ scheme using punctured convolutional coding," *IEEE. Trans. Commun.*, vol. 38, No. 11, pp. 1938–1990, Nov. 1990.

[108] Z. Lin and A. Svensson, "New rate-compatible repetition convolutional codes," *IEEE Trans. Inform. Theory*, vol. 46, No 7, pp. 2651–1657, Nov. 2000.

APPENDIX A

DECODING ALGORITHM FOR TPC/SPC CODES

Assuming even-parity check codes, BPSK modulation ($0 \rightarrow +1$, $1 \rightarrow -1$) and AWGN channels, a 2-D TPC/SPC code formed from $(N_1, N_1 - 1) \otimes (N_2, N_2 - 1)$ has the following SISO decoding algorithm (Tab. VII), where $r_{i,j}$ denotes the bits received from the channel, $Li_{i,j}$ denotes the *a priori* information (obtained from the channel or the inner code in a concatenated scheme), $LLR_{i,j}$ denotes the log-likelihood ratio, and $Le^{(1)}_{i,j}$ and $Le^{(2)}_{i,j}$ denotes the extrinsic information associated with component code $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively.

Table VII. Decoding algorithm for 2-D TPC/SPC codes.

Initialization:

for $i = 1$ to N_2, for $j = 1$ to N_1,

$Li_{i,j} = \frac{2}{\sigma^2} r_{ij}$,

$Le^{(1)}_{i,j} = Le^{(2)}_{i,j} = 0$,

Iterations:

Decoding row code $\mathcal{C}_1$: for $i = 1$ to N_2, for $j = 1$ to N_1,

$Le^{(1)}_{i,j} = 2 \tanh^{-1}(\prod_{1 \le t \le N_1, t \ne j} \tanh(\frac{Li_{i,t} + Le^{(2)}_{i,t}}{2}))$,

Decoding column code $\mathcal{C}_2$: for $j = 1$ to N_1, for $i = 1$ to N_2,

$Le^{(2)}_{i,j} = 2 \tanh^{-1}(\prod_{1 \le t \le N_2, t \ne i} \tanh(\frac{Li_{t,j} + Le^{(1)}_{t,j}}{2}))$,

Soft output and decision:

for $i = 1$ to N_2, for $j = 1$ to N_1,

$LLR_{i,j} = Li_{i,j} + Le^{(1)}_{i,j} + Le^{(2)}_{i,j}$,

$\hat{s}_{i,j} = LLR_{i,j} > 0 \ ? \ 0 : 1;$

Iteration stop criteria:

Success: All rows and columns add up (modulo-2) to 0.

Fail: A max number of iteration is reached.

APPENDIX B

COMPUTATION OF THRESHOLDS FOR PRODUCT ACCUMULATOR (PA-I) CODES

Notation:

- $f_{L_{ch,y}}$ – pdf of the messages of the received signals y obtained from the channel.
- $f^{(k)}_{L_{o,x}}$ – pdf of the (*a prior*) messages of the input x to the inner $1/(1+D)$ code in the k_{th} iteration (obtained from the outer code in the $k-1_{th}$ iteration).
- $f^{(k)}_{Le,x}$ – pdf of the (extrinsic) messages passed from the inner code to the outer code in the k_{th} iteration.
- $f^{(k)}_{L_{e1,(\cdot)}}$ and $f^{(k)}_{L_{e2,(\cdot)}}$ – pdf's of the extrinsic information computed from the upper and lower branch of the outer code in the k_{th} iteration, respectively. Subscripts d and p denote data and parity bit, respectively. Obviously, $f^{(0)}_{L_{e2,d}} = f^{(0)}_{L_{e2,p}} = \delta(0)$, where $\delta(\cdot)$ is the Kronecker delta function.
- $\mathcal{Q}$ – the quantization operation (on the messages).
- $\gamma = \alpha \boxplus \beta$ – (discretized) check operation defined as

$$\gamma = \mathcal{Q}(2\tanh^{-1}(\tanh\frac{\alpha}{2}\tanh\frac{\beta}{2})). \tag{B.1}$$

- $f_\gamma = \mathcal{R}(f_\alpha, f_\beta)$ – the operation of the resulting pdf from the check operation, which can be computed as

$$f_\gamma[k] = \sum_{(i,j):\, k\Delta = i\Delta \boxplus j\Delta} f_\alpha[i] \cdot f_\beta[j]. \tag{B.2}$$

Further, for convenient notation, denote

$$\mathcal{R}^k(f_\alpha) \triangleq \underbrace{\mathcal{R}(f_\alpha, (\mathcal{R}(f_\alpha, \cdots, \mathcal{R}(f_\alpha, f_\alpha)\cdots))}_{k-1}. \tag{B.3}$$

It then follows from [24] that the density evolution of a rate $t/(t+2)$ PA code proceeds as:

$$\begin{aligned}
\text{Initialization:} \quad & f^{(0)}_{L_{o,x}} = f^{(0)}_{L_{e,y}} = f^{(0)}_{L_{e1},d} = f^{(0)}_{L_{e2},d} = \delta(0), && \text{(B.4)}\\
\text{Inner Code:} \quad & f^{(k)}_{L_{e,y}} = \mathcal{R}\big(f^{(k-1)}_{L_{o,x}},\, f_{L_{ch,y}} * f^{(k-1)}_{L_{e,y}}\big). && \text{(B.5)}\\
& f^{(k)}_{Le,x} = \mathcal{R}^2\big(f_{L_{ch,y}} * f^{(k)}_{L_{e,y}}\big), && \text{(B.6)}\\
\text{Inner-to-Outer:} \quad & f^{(k)}_{L_o,d} = f^{(k)}_{L_{e,x}}, && \text{(B.7)}\\
& f^{(k)}_{L_o,p} = f^{(k)}_{L_{e,x}}, && \text{(B.8)}\\
\text{Outer Code:} \quad & f^{(k)}_{L_{e1},d} = \mathcal{R}\big(f^{(k)}_{L_o,p},\, \mathcal{R}^{(t-1)}(f^{(k)}_{L_o,d} * f^{(k-1)}_{Le2,d})\big), && \text{(B.9)}\\
& f^{(k)}_{L_{e1},p} = \mathcal{R}^t\left(f^{(k)}_{L_o,d} * f^{(k-1)}_{L_{e2},d}\right), && \text{(B.10)}\\
& f^{(k)}_{L_{e2},d} = \mathcal{R}\left(f^{(k)}_{L_o,p},\, \mathcal{R}^{(t-1)}\big(f^{(k)}_{L_o,d} * f^{(k)}_{L_{e1},d}\big)\right), && \text{(B.11)}\\
& f^{(k)}_{L_{e2},p} = \mathcal{R}^t\left(f^{(k)}_{L_o,d} * f^{(k)}_{L_{e1},d}\right), && \text{(B.12)}\\
\text{Outer-to-Inner:} \quad & f^{(k+1)}_{L_{o,x}} = \frac{t\big(f^{(k)}_{L_{e1},d} + f^{(k)}_{L_{e2},d}\big)}{t+2} + \frac{f^{(k)}_{L_{e1},p} + f^{(k)}_{L_{e2},p}}{2t+2}. && \text{(B.13)}
\end{aligned}$$

It is instructive to note that although the outer code (the parallel concatenation of 2 single parity check codes) can be viewed as an LDPC code with column weight 2, it is desirable to take a serial update procedure as described above rather than a parallel one as in a conventional LDPC code. In this way, the checks corresponding to SPC1 and SPC2 take turns to update, which is expected to have a faster convergence [24].

APPENDIX C

ABBREVIATIONS

APP	*a posteriori* probability
ARQ	automatic repeat request
ASE	amplifier spontaneous emission
AWGN	additive white Gaussian noise
BCH	Bose-Chaudhuri-Hocquenghem
BCJR	Bahl-Cocke-Jelinek-Raviv
B-DPSK	binary differential phase shift keying
BER	bit error rate
BPSK	binary phase shift keying
BTC	block turbo code
CRC	cyclic redundancy check
CSI	channel state information
CT	concatenated tree
DC	direct current
DE	density evolution
DPSK	differential phase shift keying
DFE	decision feedback equalization
DWDM	dense wavelength division multiplexing

ECC	error correction coding
EDC	error detection coding
EXIT	extrinsic information transfer
FEC	forward error correction
FER	frame error rate
FFT	fast Fourier transform
FIR	finite impulse response
GPA	generalized product accumulate
HCCC	hybrid concatenated convolutional codes
IDDD	iterative differential detection and decoding
IDE	iterative decoding and equalization
IIR	infinite impulse response
IOWE	input-output weight enumerator
IOWTP	input-output weight transfer probability
IRA	irregular repeat accumulate
ISI	inter-symbol interference
LDPC	low density parity check
LLR	log-likelihood ratio
LMS	least mean square
MAP	maximum *a posteriori*
ML	maximum likelihood
MMSE	minimum mean square error
MSK	minimum shift key

NASK	negative acknowledgment
NRZI	non-return-to-zero inverted
OFC	optical fiber communications
OOK	on-off keying
OWE	output weight enumerator
PA	product accumulate
PEP	pair-wise error probability
PCCC	parallel concatenated convolutional codes
pdf	probability density function
PR	partial response
PRML	partial response maximum likelihood
RA	repeat accumulate
RC	rate compatible
RLL	run-length limit
RS	Reed-Solomon
RSC	recursive systematic convolutional
SCCC	serially concatenated convolutional codes
SISO	soft-in soft-out
SNR	signal-to-noise ratio
SOVA	soft-output Viterbi algorithm
SPC	single-parity check
TPC	turbo product code
WER	word error rate

VITA

Jing Li (Tiffany) received the B.Sc. degree in computer science (with honor) from Peking University, China, in 1997, and the M.Eng. degree and the Ph.D. degree in electrical engineering from Texas A&M University, College Station, TX, in 1999 and 2002, respectively.

From 1998 to 1999, She was a Graduate Assistant with Texas Transportation Institute. From 1999 to 2002, she was a Research/Teaching Assistant with the Department of Electrical Engineering, Texas A&M University. She was a research intern with Seagate Research Laboratory, Pittsburgh, PA, and with Tyco Telecommunications Laboratory, Eatontown, NJ, in the summer of 2000 and 2001, respectively. She is currently with the Electrical and Computer Engineering Department in Lehigh University. Her research interests fall in the general area of telecommunications and digital signal processing, with specific focus on communication and coding theory, advanced coding techniques for wireless/wireline, optical fiber communications as well as data storage systems.

Ms Li is the recipient of the Ethel Ashworth-Tsutsui Memorial Award for Research (2001), the J. W. Van Dyke Memorial Scholarship for Academic Excellence (2001) and the TxTEC (Texas Telecommunication Engineering Consortium) Scholarship (2000). Her contact is:

Tiffany Jing Li

Electrical and Computer Engineering Department

Lehigh University

Bethlehem, PA 18015, USA

The typist for this thesis was Jing Li (Tiffany).