COMMUNICATION SCIENCES INSTITUTE

Iterative Data Detection: Bounding Performance and Complexity Reduction

by

Kyuhyuk Chung

CSI-03-11-20

USC VITERBI SCHOOL OF ENGINEERING
UNIVERSITY OF SOUTHERN CALIFORNIA
ELECTRICAL ENGINEERING — SYSTEMS
LOS ANGELES, CA 90089-2565

ITERATIVE DATA DETECTION: BOUNDING PERFORMANCE AND COMPLEXITY REDUCTION

by

Kyuhyuk Chung

A Dissertation Presented to the

FACULTY OF THE GRADUATE SCHOOL

UNIVERSITY OF SOUTHERN CALIFORNIA

In Partial Fulfillment of the

Requirements for the Degree

DOCTOR OF PHILOSOPHY

(ELECTRICAL ENGINEERING)

November 2003

Dedication

To my Lord Jesus .....

Acknowledgements

First and foremost I am extremely grateful to my advisor, Dr. Keith M. Chugg, under whose guidance I have been able to prepare my dissertation. No amount of thanks can express my gratitude to him. My thanks also go to the members of my defense committee, Dr. Urbashi Mitra from the department of Electrical Engineering and Dr. Peter Baxendale from the department of Mathematics, for reading previous drafts of this dissertation and providing many valuable comments that improved this dissertation. I would like to thank the members of my qualifying committee, Dr. Zhen Zhang and Dr Antonio Ortega from the department of Electrical Engineering, for their help during my studies.

I am very grateful to my colleagues within Dr. Chugg's group, Jun Heo, Durai Thirupathi, Pansop Kim, Yuankai Wang, Mingrui Zhu, Jordan Melzer, Orhan Coskun, Ali Taha, Robert Golshan, Phunsak Thiennviboon, Tom Halford, Chunhsuan Kuo, ,for our many valuable discussions.

I am also thankful to my officemates, Terry Lewis, Jordan Melzer, and Meng-Hsuan Chung for helping to make thousands of hours productive and joyful.

I would like to show my sincere gratitude to two visiting scholars: Anders Hansson from Sweden for teaching me CPM and Gianluigi Ferrari from Italy for teaching me Italian, Ciao.

I am also very grateful for the CSI staffs: Milly Montenegro, Mayumi Thrasher, and Gerrielyn Ramos. Their help was essential to my study.

I am greatly indebted to my spiritual leader, Dr. Joshua Park for guiding me closer to God.

I will never find words enough to express the gratitude that I owe to my parents, brother and sister. Throughout my life they have always encouraged me.

Finally, I give all glory to my Lord Jesus Christ for all the work I have accomplished. I thank him for providing me the vision and clarity of thought during this program.

Contents

List Of Tables

List Of Figures

Abstract

Iterative detection (ID) or turbo decoding is a good approximation to the optimal decoder for a concatenated network of Finite State Machines (FSMs) constructed from constituent codes and ISI channel memory. Iterative detection was used successfully for the first time in the decoding of Parallel Concatenated Convolutional Codes (PCCCs or turbo codes). Applications that have been using iterative detection include Serially Concatenated Convolutional Codes (SCCCs), Trellis Coded Modulation (TCM) over interleaved, frequency-selective fading channels, and Serially Concatenated Continuous Phase Modulation (SCCPM).

The Soft Input Soft Output (SISO) module, which is a basic building block in iterative detection, implements an algorithm performing an update of the reliability information of the input and the output symbols of a FSM. For implementation, the complexity should be reduced. Reduced state SISO (RS-SISO) algorithms are used for complexity reduction. When perfect CSI is not available at the receiver, the SISO module should be able to deal with unknown, and possibly time-varying parameters. SISO modules that jointly estimate the parameters and generate soft information will be referred to as Adaptive SISO (A-SISO) modules. In this paper, the RS-SISO

algorithm is used for complexity reduction of the A-SISO module, namely RS-A-SISO algorithms. RS-A-SISO modules are applied to SCCPM over frequency-selective fading channels when perfect CSI is not available and trade-offs between performance and complexity are considered. RS-A-SISO algorithms are analyzed using the SNR evolution technique.

We present upper bounds for the maximum-likelihood decoding performance of turbo-like codes for specific interleavers. Previous research developed upper bounds for turbo-like codes using the uniformly interleaved assumption, which bounds the performance averaged over all interleavers. These upper bounds for specific interleavers are based on the simple bound and estimated weight distributions including the exact several smallest distance terms because if either estimated weight distributions on their own or the exact several smallest distance terms only are used, an accurate bound can not be obtained. The approximate weight distribution is verified by statistical reliability measure.

Chapter 1

Introduction

1.1 Iterative Detection

Iterative detection (ID) or turbo decoding is a good approximation to the optimal decoder for a concatenated network of Finite State Machines (FSMs) constructed from constituent codes and ISI channel memory. Iterative detection was used successfully for the first time in the decoding of Parallel Concatenated Convolutional Codes (PCCCs or turbo codes) [7]. Applications that have been using iterative detection include Serially Concatenated Convolutional Codes (SCCCs) [4], Trellis Coded Modulation (TCM) over interleaved, frequency-selective fading channels [1], and Serially Concatenated Continuous Phase Modulation (SCCPM) [28]. In Figure 1.1 performance of turbo codes is shown for 1st and 10th iterations. A great iteration gain is obtained by using iterative detection.

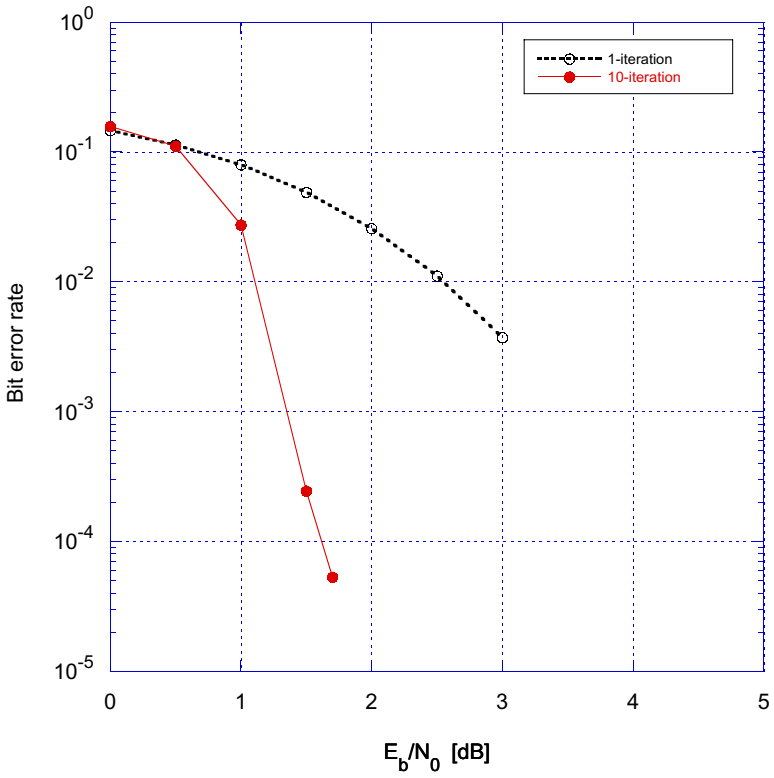

Figure 1.1: Performance of Turbo codes (Interleaver=2000, AWGN channel, BPSK)

1.2 Density Evolution Analysis of Iterative Detection

Density evolution [32] [31] [14] [21] [19] has recently been used to analyze iterative detection. This analysis explains many mysteries of iterative detection for turbo codes, SCCCs, and Low Density Parity Check (LDPC) codes. The evolution of the input and output density in iterative decoders is shown in Figure 1.2. The analytic capacity of LDPC codes was calculated in [32]. The density evolution technique was recursively used to track the density of extrinsic information between the variable nodes

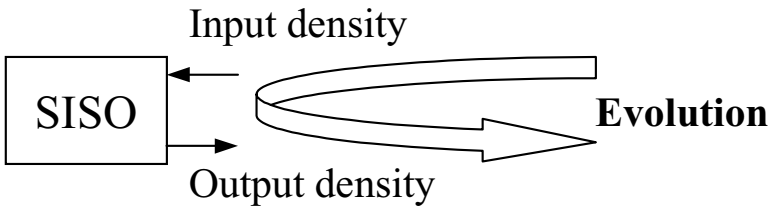

Figure 1.2: Density evolution in iterative decoders

and check nodes of LDPC code on various channel conditions. A simplified version of the density evolution technique was introduced with Gaussian approximation in [14]. The Gaussian approximation was based on the well known fact that the extrinsic information can be well represented as a Gaussian random variable as the number of iterations is increased. The most analytic research has been focused on LDPC codes because the LDPC decoding algorithm can be represented by a simple two-state trellis or parity-check equations. Similar attempts were made for Turbo codes based on the simulation to get the threshold which shows the asymptotic capacity of Turbo codes when the interleaver size and the number of iterations go to infinity [21]. The density evolution was used to explain many open problems of turbo codes and SCCCs, for example, the role of systematic bits and the role of recursive constitute codes. While the analytic density evolution methods [32] [14] are applied to LDPC codes, the simulation based density evolution methods [21] [19] are used for turbo codes and SCCCs.

1.3 Bounding on the Performance of Turbo-Like Codes

In [18], transfer function bounding techniques were applied to obtain upper bounds on the bit-error rate for maximum-likelihood decoding of turbo codes constructed with pseudo-random interleavers. Since union bounds are intractable for any particular pseudo-random interleaver, the transfer function bound is developed as a random coding bound based on the uniformly interleaved assumption, i.e., performance averaged over all possible interleaver permutations. Therefore, the transfer function bound can not be used to bound the performance of the turbo code with a particular interleaver.

Since the union bound cannot predict the performance above the cutoff rate, there is a great demand to have bounds on performance that are useful for rates above the cutoff rate. One of those bounds is the simple bound [17]. In [17], the simple bound is used with the uniformly interleaved assumption for turbo codes. Therefore, bounds on the maximum-likelihood decoding performance of the turbo-like codes with a particular interleaver are still unavailable. Note also that simulation performance of turbo-like codes using iterative detection is suboptimal because iterative detection is approximate to the optimal detection.

In this dissertation we solve intractability of upper bounds with any particular pseudo-random interleaver by using approximate input-output weight distributions. The performance of turbo-like codes at high SNR are well approximated by the expression of the union bound, truncated to the contribution of the several smallest non-zero distance terms [22]. In [22] branch and bound algorithms [26] were developed for finding the several smallest distances and their multiplicities. We include these exact distance spectrum results for several of the smallest non-zero distances into the approximate weight distributions. The reliability of the estimated upper bound is statistically evaluated.

1.4 Organization

In Chapter 2, iterative detection and density evolution analysis are introduced. The Soft Input Soft Output (SISO) module, which is a basic building block in iterative detection, is discussed. Reduced state SISO (RS-SISO) algorithms are explained for complexity reduction. When perfect CSI is not available at the receiver, the SISO module should be able to deal with unknown, and possibly time-varying parameters. SISO modules that jointly estimate the parameters and generate soft information are introduced as Adaptive SISO (A-SISO) modules.

In Chapter 3, the RS-SISO algorithm is used for complexity reduction of the A-SISO module, namely RS-A-SISO algorithms. RS-A-SISO modules are applied to

SCCPM over frequency-selective fading channels when perfect CSI is not available and trade-offs between performance and complexity are considered. RS-A-SISO algorithms are analyzed using the SNR evolution technique.

In Chapter 4, we obtain upper bounds on the performance of turbo-like code with any particular random interleaver by using an estimate of the probability density function of the input-output weight enumerator.

Chapter 2

Iterative Detection and Density Evolution Analysis

2.1 Iterative Detection

2.1.1 Soft Input Soft Output (SISO) Algorithms

The SISO module [5] implements an algorithm performing an update of the reliability information of the input and the output symbols of a FSM. A-Posteriori Probability (APP) forward-backward algorithms (FBAs) and Minimum Sequence Metric (MSM) FBAs [12] produce soft information that is thresholded to obtain the Maximum A-posteriori Probability (MAP) Symbol Detection and the MAP Sequence Detection respectively. By replacing min-sum operations by sum-product operations in the MSM FBA, one directly obtain the APP FBA. In the following we consider only the MSM FBA. MSM is defined by

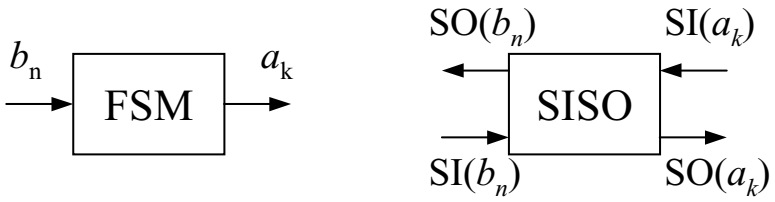

Figure 2.1: A block diagram of FSMs and SISO algorithms

$$\mathrm{MSM}(u) = \min_{\mathbf{a}:u}[-\ln(p(\mathbf{z}|\mathbf{a})p(\mathbf{a}))] \tag{2.1}$$

where $\mathbf{z}$ is an observation sequence, u is any quantity derived from the system input-output sequence pair $(\mathbf{a}, \mathbf{x}(\mathbf{a}))$, and the notation $\mathbf{a} : u$ means all $\mathbf{a}$ consistent with u. The forward-Add Compare Select(ACS) and backward-ACS recursions in the MSM algorithm are

$$\mathrm{MSM}_0^k(s_{k+1}) = \min_{t_k:s_{k+1}} [\mathrm{MSM}_0^{k-1}(s_k) + \mathrm{M}_k(t_k)] \tag{2.2}$$

$$\mathrm{MSM}_k^{K-1}(s_k) = \min_{t_k:s_k}[\mathrm{MSM}_{k+1}^{K-1}(s_{k+1}) + \mathrm{M}_k(t_k)] \tag{2.3}$$

where $\mathrm{MSM}_{k_1}^{k_2}(s_k)$ denotes soft information of a state at time k based on observation $\mathbf{z}_{k_1}^{k_2}$ and $\mathrm{M}_k(t_k)$ is the transition metric of t_k. The transition metric $\mathrm{M}_k(t_k)$ is defined as a sum of soft-inputs of the system input a_k and output $x_k(t_k)$,

$$\mathrm{M}_k(t_k) = \mathrm{MI}(x_k(t_k)) + \mathrm{MI}(a_k). \tag{2.4}$$

For the MSM version, the soft outputs are

$$\begin{aligned} \mathrm{MO}(u_k) &= \min_{t_k:u_k}[\mathrm{MSM}_0^{k-1}(s_k) + \mathrm{M}_k(t_k) \\ &\quad +\mathrm{MSM}_{k+1}^{K-1}(s_{k+1})] - \mathrm{MI}(u_k) \end{aligned} \tag{2.5}$$

where u_k can be a_k or x_k.

2.1.2 Reduced State SISO (RS-SISO) Algorithms

RS-SISO algorithms have two options, namely forward decision feedback, backward decision feedback (FDF/BDF) structures and forward decision feedback (FDF) structures [9] [15]. In the FDF/BDF structure, survivors are constructed both in the forward and in the backward direction. In the FDF structure, survivors are constructed only in the forward direction, all transition metrics calculated in the forward recursion are saved, and the backward recursion is performed without survivors. The forward

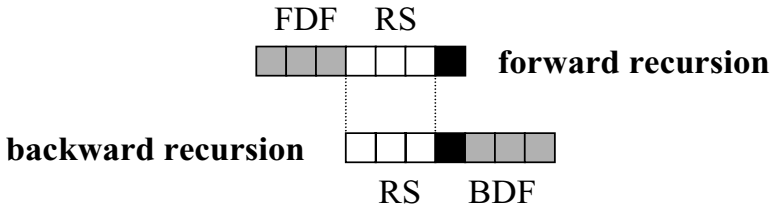

Figure 2.2: Forward and backward decision feedback in RS-SISO algorithms

and backward state metrics $\mathrm{F}(v_k)$ and $\mathrm{B}(v_k)$ are defined for the RS-SISO algorithm recursively as

$$\mathrm{F}(v_{k+1}) = \min_{v_k : v_{k+1}} [\mathrm{F}(v_k) + \mathrm{M}_k(\tau_k)] \tag{2.6}$$

$$\mathrm{B}(v_k) = \min_{v_{k+1} : v_k} [\mathrm{B}(v_{k+1}) + \mathrm{M}_k(\tau_k)] \tag{2.7}$$

where v_k is the truncated trellis state at time k and $\tau_k = (v_k, v_{k+1})$ is the corresponding truncated trellis transition at time k. Survivor paths that are associated with each trellis state are employed in the calculation of the transition metric $\mathrm{M}_k(\tau_k)$. After executing both the forward and backward recursions, the soft outputs for a_k can be obtained by

$$\begin{aligned} \mathrm{MO}(a_k) = \min_{v_{k+1} : a_k} [&\mathrm{F}(v_k) + \mathrm{M}_k(\tau_k) \\ &+ \mathrm{B}(v_{k+1})] - \mathrm{MI}(a_k). \end{aligned} \tag{2.8}$$

2.1.3 Adaptive SISO (A-SISO) Algorithms

When perfect CSI is not available at the receiver, the SISO module should be able to deal with unknown, and possibly time-varying parameters. SISO modules that jointly estimate the parameters and generate soft information will be referred to as Adaptive SISO (A-SISO) modules [1]. When we construct A-SISO algorithm whether Fixed-Interval (FI) or Fixed-Lag (FL), two options are available, namely

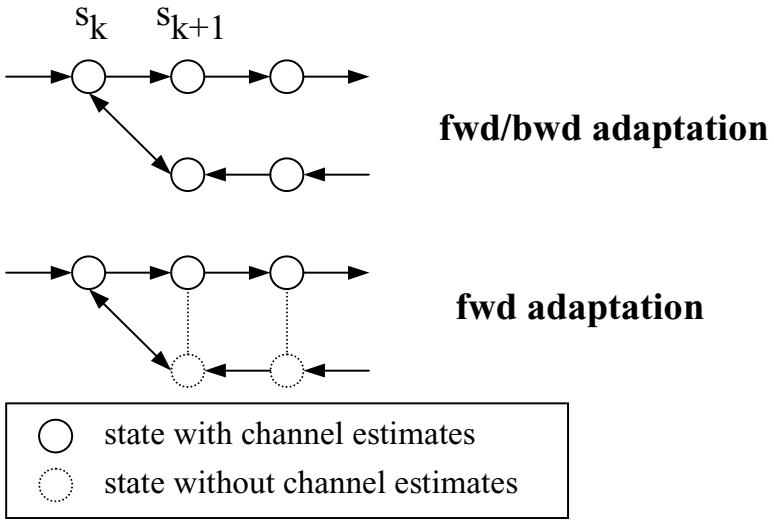

Figure 2.3: Forward and backward adaptation in A-SISO algorithms

forward adaptive, backward adaptive (FABA) structures and forward adaptive (FA) structures [12] [1] [25]. In the first one, adaptive processing is performed both in the forward and in the backward direction. In the second approach, adaptive processing is only performed in the forward direction and all transition metrics associated with the forward adaptive step are saved for the non-adaptive backward recursion. FA structure is used assuming a training sequence only at the beginning of data blocks. The Least Mean Square (LMS) algorithm is used for forward channel estimation.The LMS algorithm for updating a channel tap vector $\hat{\mathbf{f}}(s_k)$ in the forward recursion is given by

$$\hat{\mathbf{f}}(s_{k+1}) = \hat{\mathbf{f}}(s_k) + \beta[z_k - \hat{\mathbf{f}}^T(s_k) \cdot \mathbf{a}(t_k)]\mathbf{a}^*(t_k) \tag{2.9}$$

where $\mathbf{a}(t_k) = \{a_{k-i}\}_{i=0}^{N_c}$ are fractionally-spaced signal samples during an ISI-channel length (N_c+1), β is a suitable constant, and the forward channel estimate is initialized by running the Recursive Least Squares (RLS) algorithm on a training sequence.

2.2 Density Evolution Analysis

Density evolution [32] [31] has recently been used to analyze iterative detection. For short block size, the analysis of iterative detection is an open problem. Such analysis, however, is possible under certain assumptions for very large block size. The density evolution analysis tracks the probability density function (pdf) of the extrinsic information messages as this density evolves from iteration to iteration. In [32], the density evolution technique was recursively used to track the density of extrinsic message between the variable node and check nodes of Low Density Parity Check (LDPC) codes. The density evolution technique was simplified with a Gaussian approximation in [14]. The most analytic research have been focused on LDPC codes because the LDPC decoding algorithm can be represented by a simple two-state trellis or parity-check equations.

The SISO module in Turbo decoders was considered as a signal-to-noise ratio transformer in [21]. It was found in [19] that the Gaussian model based on the empirical mean only assuming that the variance is determined by the symmetry condition proposed in [31] gave closer agreement with the results obtained from actual density

evolution than the Gaussian model based on the empirically determined mean and variance as independent parameters. It was shown in [21] that the threshold is approximately matched the region where the bit error rate curves start to fall down. In [19], they used the density evolution to explain many mysteries of turbo codes and SCCCs, for example, the role of systematic bits and the role of recursive constitute codes etc.

Chapter 3

Reduced State Adaptive SISO Algorithms for Serially Concatenated CPM over Frequency-Selective Fading Channels and Density Evolution Analysis

3.1 Introduction

Iterative detection (ID) or turbo decoding [6] is a good approximation to the optimal decoder for a concatenated network of Finite State Machines (FSMs) constructed from constituent codes and intersymbol interference (ISI) channel memory.

In [36], Trellis Coded CPM (TCCPM) on multipath fading ISI channels was studied without iterative detection. It is shown in [29] that by exploiting the recursive

nature of CPM, TCCPM can have large interleaver gains similar to Serially Concatenated Convolutional Codes (SCCCs) [5]. Similar independent work with iterative detection was reported in [28]. None of papers above considers SCCPM [28] over fading ISI channels with iterative detection.

Adaptive SISO (A-SISO) modules [1] were developed as a natural extension of Soft-Input Soft-Output (SISO) modules [5] for use when perfect CSI is not available at the receiver. In [1], various A-SISO algorithms were defined and derived, which produces a family of suboptimal practical algorithms.

Since the complexity of SISO algorithms increases exponentially with constraint length, Reduced State (forward-backward) SISO (RS-SISO) algorithms were proposed independently for complexity reduction at the same time in [9] [15].

Density evolution [32] [31] has recently been used to analyze iterative detection. In [32], the density evolution technique was recursively used to track the density of extrinsic message between the variable node and check nodes of Low Density Parity Check (LDPC) codes. The density evolution technique was simplified with a Gaussian approximation in [14].

The SISO module in Turbo decoders was considered as a signal-to-noise ratio transformer in [21]. In [19], they used the density evolution to explain many mysteries of turbo codes and SCCCs, for example, the role of systematic bits and the role of recursive constitute codes etc.

In this paper, the RS-SISO algorithm is used for complexity reduction of the A-SISO module, namely RS-A-SISO algorithms [13]. RS-A-SISO modules are applied to SCCPM over frequency-selective fading channels when perfect CSI is not available and trade-offs between performance and complexity are considered. RS-SISO algorithms are of paramount importance in A-SISO algorithms because the complexity of A-SISO algorithms is higher than that of SISO algorithms due to channel estimation. When the input of a joint CPM-ISI trellis is binary, multiple-step RS-A-SISO algorithms [10] can be a design option for improved performance with slightly increased complexity. Thresholds of RS-A-SISO algorithms are obtained using the density evolution technique.

This paper organized as follows. Section 3.2 describes the system and channel model. In section 3.3 we explain various RS-A-SISO algorithms. Section 3.5 presents numerical results. Section 3.6 explains the density evolution technique. In section 3.7 we analyze RS-A-SISO algorithms using density evolution technique. Section 3.8 concludes the paper.

3.2 System Description and Channel Model

A block diagram of SCCPM on an ISI and AWGN channel is shown in Figure 3.1. After the source bits b_n are fed to a convolutional code (CC), the coded bits c_m are bit-interleaved and continuous-phase-modulated. The modulated signals $a(t)$ are

transmitted through an ISI and AWGN channel. The carrier phase is modulated by a positive normalized frequency pulse $g(t)$, containing no impulses and being non-zero for L symbol intervals [2]. Its corresponding phase pulse $q(t)$ is the integral of $g(t)$, normalized to $q(LT) = 1/2$ with a symbol time T. The modulation index is assumed to be rational and irreducible, $h = K/P$, and then the CPM modulator can be viewed as a continuous-phase encoder (CPE) followed by a memoryless modulator (MM) [33]. For practical purposes, we can assume that W is the bandwidth occupied by the real bandpass CPM signal. Then the band occupancy of the complex baseband equivalent lowpass CPM signal is $|f| \leq W/2$. A tapped-delay-line (TDL) model is usually used for a time-varying frequency-selective fading channel The lowpass impulse response for the channel is

$$f(t, \tau) = \sum_{i=0}^{N_c} f_i(t)\delta(\tau - iT_c) \tag{3.1}$$

where T_c is the channel resolution which satisfies $T_c \leq 1/W$, $f_i(t) = T_c f(t, iT_c)$ is the tap weight coefficient, and $(N_c + 1)$ is the length of the channel in terms of T_c.

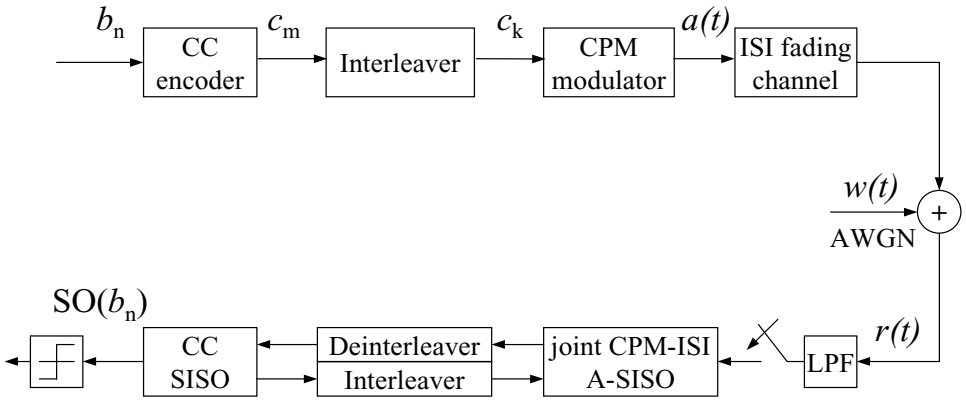

Figure 3.1: System model.

T_c is chosen such that symbol time $T = N_s T_c$ with N_s being an integer. In complex baseband representation, the received continuous-time signal is given by

$$\begin{aligned} r(t) &= x(t) + n(t) \\ &= \int_0^\infty f(t,\tau) a(t-\tau) d\tau + n(t) \\ &= \sum_{i=0}^{N_c} f_i(t) a(t - iT_c) + n(t) \end{aligned} \tag{3.2}$$

where n(t) is the AWGN. Conventionally, the received signal $r(t)$ is passed through a matched-filter $x^*(-t)$, followed by a sampler and a noise whitening filter. But since CPM is a nonlinear modulation, a bank of matched-filters has to be implemented. In [36] a low-pass filter (LPF) with two-sided bandwidth W is used, followed by fractionally-spaced sampling every T_c. Then the kth sample of $r(t)$ is

$$\begin{aligned} r_k \equiv r(kT_c) &= \sum_{i=0}^{N_c} f_i(kT_c) a(kT_c - iT_c) + \tilde{n}(kT_c) \\ &= \sum_{i=0}^{N_c} f_{i,k} a_{k-i} + n_k \end{aligned} \tag{3.3}$$

where $\tilde{n}(t)$ is filtered white noise and n_k is complex zero-mean Gaussian distributed with variance N_0. The coefficient $\{f_{i,k}\}_{i=0}^{N_c}$ is a time-varying tap vector at time k and $\{a_{k-i}\}_{i=0}^{N_c}$ are fractionally-spaced signal samples during an ISI-channel length (N_c+1). The equation (3.3) can be used as an equivalent Discrete-Time (DT) model for CPM on fading ISI channel [36].

The low-pass-filtered and sampled observations are fed to the inner joint CPM-ISI A-SISO module [1]. The outer CC SISO [5] and the inner A-SISO are connected by an interleaver and a deinterleaver to exchange soft information iteratively. The inner A-SISO re-estimates the ISI channel each iteration based on [1].

3.3 Reduced State Adaptive SISO Algorithms

The SISO module [5] implements an algorithm performing an update of the reliability information of the input and the output symbols of a FSM. A-Posteriori Probability (APP) forward-backward algorithms (FBAs) and Minimum Sequence Metric (MSM) FBAs [12] produce soft information that is thresholded to obtain the Maximum A-posteriori Probability (MAP) Symbol Detection and the MAP Sequence Detection respectively. By replacing min-sum operations by sum-product operations in the MSM FBA, one directly obtain the APP FBA. In the following we consider only the MSM FBA. MSM is defined by

$$\mathrm{MSM}(u) = \min_{\mathbf{a}:u}[-\ln(p(\mathbf{z}|\mathbf{a})p(\mathbf{a}))] \tag{3.4}$$

where $\mathbf{z}$ is an observation sequence, u is any quantity derived from the system input-output sequence pair $(\mathbf{a}, \mathbf{x}(\mathbf{a}))$, and the notation $\mathbf{a} : u$ means all $\mathbf{a}$ consistent with u.

The forward-Add Compare Select(ACS) and backward-ACS recursions in the MSM algorithm are

$$\mathrm{MSM}_0^k(s_{k+1}) = \min_{t_k:s_{k+1}} [\mathrm{MSM}_0^{k-1}(s_k) + \mathrm{M}_k(t_k)] \tag{3.5}$$

$$\mathrm{MSM}_k^{K-1}(s_k) = \min_{t_k:s_k} [\mathrm{MSM}_{k+1}^{K-1}(s_{k+1}) + \mathrm{M}_k(t_k)] \tag{3.6}$$

where $\mathrm{MSM}_{k_1}^{k_2}(s_k)$ denotes soft information of a state at time k based on observation $\mathbf{z}_{k_1}^{k_2}$ and $\mathrm{M}_k(t_k)$ is the transition metric of t_k.

When perfect CSI is not available at the receiver, the SISO module should be able to deal with unknown, and possibly time-varying parameters. SISO modules that jointly estimate the parameters and generate soft information will be referred to as Adaptive SISO (A-SISO) modules [1]. When we construct A-SISO algorithm whether Fixed-Interval (FI) or Fixed-Lag (FL), two options are available, namely forward adaptive, backward adaptive (FABA) structures and forward adaptive (FA) structures [12] [1] [25]. In the first one, adaptive processing is performed both in the forward and in the backward direction. In the second approach, adaptive processing is only performed in the forward direction and all transition metrics associated with the forward adaptive step are saved for the non-adaptive backward recursion. In this paper, FA structure is used assuming a training sequence only at the beginning of data blocks. The Least Mean Square (LMS) algorithm is used for forward channel

estimation.The LMS algorithm for updating a channel tap vector $\hat{\mathbf{f}}(s_k)$ in the forward recursion is given by

$$\hat{\mathbf{f}}(s_{k+1}) = \hat{\mathbf{f}}(s_k) + \beta[z_k - \hat{\mathbf{f}}^{\,T}(s_k) \cdot \mathbf{a}(t_k)]\mathbf{a}^*(t_k) \tag{3.7}$$

where $\mathbf{a}(t_k) = \{a_{k-i}\}_{i=0}^{N_c}$ are fractionally-spaced signal samples during an ISI-channel length (N_c+1), β is a suitable constant, and the forward channel estimate is initialized by running the Recursive Least Squares (RLS) algorithm on a training sequence.

RS-SISO algorithms also have two options, namely forward decision feedback, backward decision feedback (FDF/BDF) structures and forward decision feedback (FDF) structures [9] [15]. In the FDF/BDF structure, survivors are constructed both in the forward and in the backward direction. In the FDF structure, survivors are constructed only in the forward direction, all transition metrics calculated in the forward recursion are saved, and the backward recursion is performed without survivors.

In this paper, the FDF structure is used. RS-SISO with the FDF structure is consistent with the FA structure in A-SISO. The forward and backward state metrics $\mathrm{F}(v_k)$ and $\mathrm{B}(v_k)$ are defined for the RS-SISO algorithm recursively as

$$\mathrm{F}(v_{k+1}) = \min_{v_k : v_{k+1}} [\mathrm{F}(v_k) + \mathrm{M}_k(\tau_k)] \tag{3.8}$$

$$\mathrm{B}(v_k) = \min_{v_{k+1} : v_k} [\mathrm{B}(v_{k+1}) + \mathrm{M}_k(\tau_k)] \tag{3.9}$$

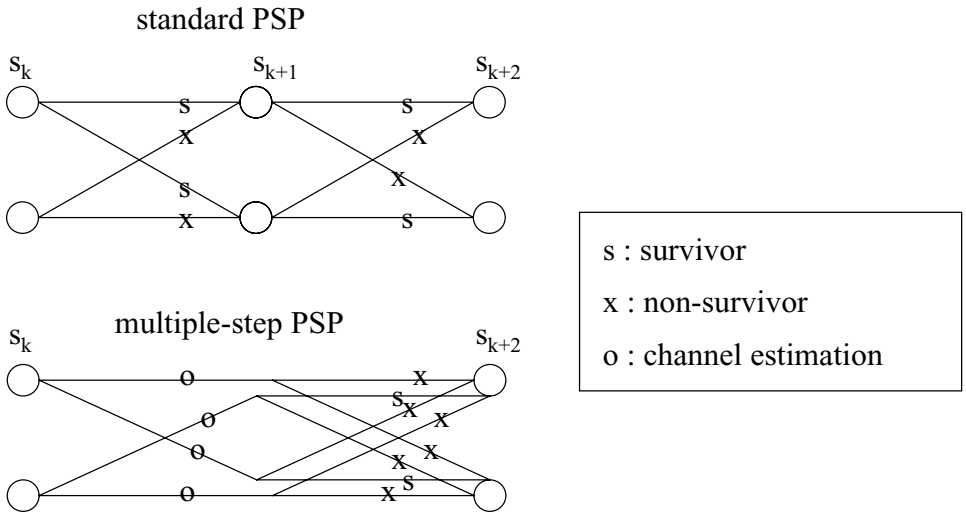

Figure 3.2: Standard PSP and multiple-step PSP.

where v_k is the truncated trellis state at time k and $\tau_k = (v_k, v_{k+1})$ is the corresponding truncated trellis transition at time k. After executing both the forward and backward recursions, the soft outputs for a_k can be obtained by

$$\begin{aligned} \mathrm{MO}(a_k) &= \min_{v_{k+1}:a_k} [\mathrm{F}(v_k) + \mathrm{M}_k(\tau_k) \\ &\quad + \mathrm{B}(v_{k+1})] - \mathrm{MI}(a_k). \end{aligned} \tag{3.10}$$

Table 3.1: Complexity Comparison: S = number of states, M-ary input

Algorithm	Number of transitions	Channel updates
ME-RS-A-SISO	S×M	S
SE-RS-A-SISO	S×M	1
MS-RS-A-SISO	S×M^2×1/2	(S×M+S)×1/2

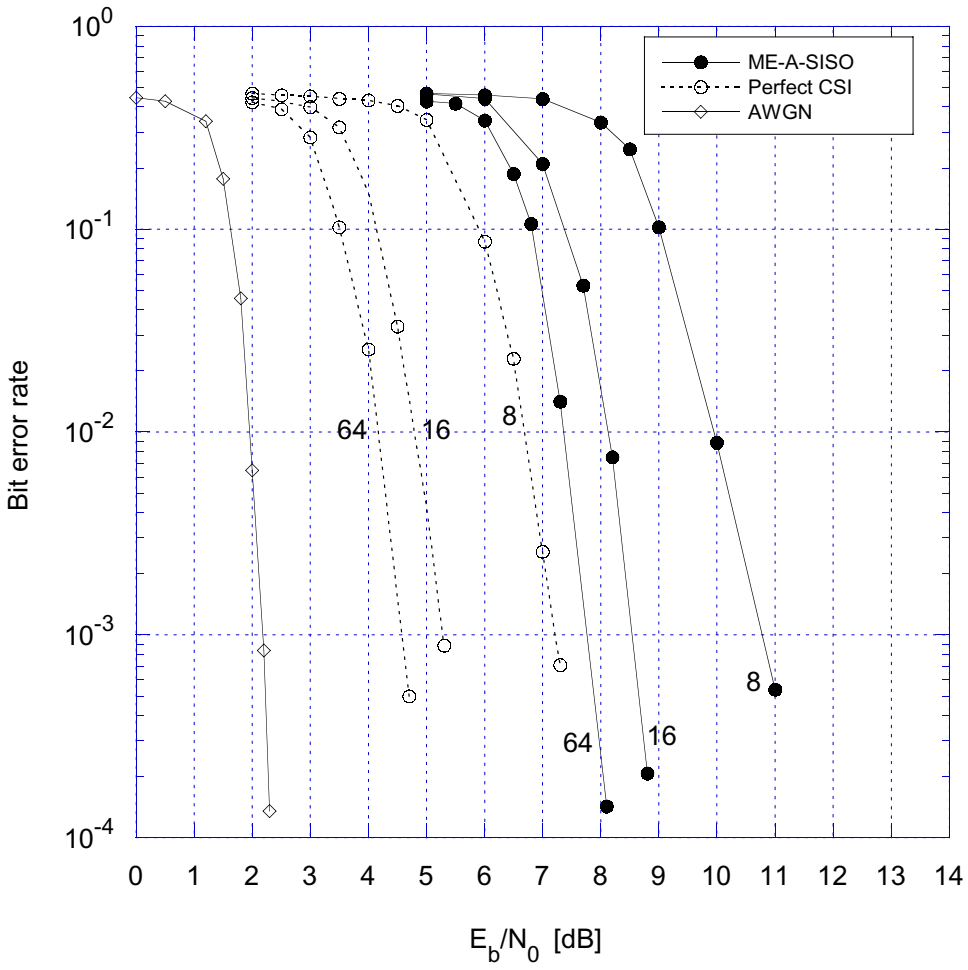

Figure 3.3: Performance of AWGN channel, perfect CSI, and multiple-estimator A-SISO ($\nu_d = 0.002$). (For AWGN channel, the 8-state CPM SISO is used. For perfect CSI and ME-A-SISO, 64 is full state and 16 or 8 are reduced states.)

RS-SISO algorithms can be used in A-SISO algorithms to produce soft-outputs for iterative detection with reduced complexity when perfect CSI is not available. These combined SISO algorithms will be referred to as RS-A-SISO algorithms. In RS-A-SISO algorithms the state-survivor paths selected for updating channel tap vectors are used for calculating transition metrics. In the following subsections various RS-A-SISO algorithms are described.

3.3.1 Multiple-Estimator (ME) RS-A-SISO Algorithm

In the ME-RS-A-SISO each state has a estimated channel tap vector $\hat{\mathbf{f}}(v_k)$ that is updated in the Per-Survivor Processing (PSP) manner [30]. The reduced number of states means reduced channel-estimation complexity as well as ACS complexity. For the calculation of the truncated trellis transition metric both the estimated channel tap vector $\hat{\mathbf{f}}(v_k)$ and the state-survivor corresponding to the transition are required. The same state-survivors are also used in updating channel tap vectors.

3.3.2 Single-Estimator (SE) RS-A-SISO Algorithm

In the SE-RS-A-SISO every state has the common estimated channel tap vector $\hat{\mathbf{f}}_k$ that is updated employing the decision-feedback (DF) assumption introduced in [34]. For updating the estimated channel tap vector $\hat{\mathbf{f}}_k$ the state with minimum metric is first chosen, the state-survivor path is traced back as many steps as tentative delay d, and then $\hat{\mathbf{f}}_k$ is updated with the truncated trellis transition τ_{k-d} that has the corresponding state-survivor path. The single estimator structure reduces estimator complexity more than the multiple estimator structure.

3.3.3 Multiple-Step (MS) RS-A-SISO Algorithm

The A-SISO algorithm based on a trellis in a fading channel is suboptimal due to a forced-folding of a tree [11]. But it is almost impossible to use a tree structure A-SISO for detection of large size sequences so that trellis structures are used in practice. The MS-RS-A-SISO uses a tree structure during two trellis steps instead of a trellis structure [10]. The channel adaptation scheme in MS-RS-A-SISO is shown in Figure 3.2. The channel updates are executed on all transitions instead of survivor paths in the first step of each multiple step, which means that non-survivors in standard trellis-based PSP can be a candidate for a survivor depending on the decision of the next step. In the second step of each multiple step, the channel updates are executed only on survivor paths. Complexity increases slightly with a relatively large performance gain. Complexity comparison is described in Table 3.1 using the number of transitions and channel updates per trellis step.

3.4 Complexity Comparison of Joint Trellis and Separate Trellises

The M-ary, $h = K/P$, L-partial response CPM trellis produces one of $P \cdot M^L$ signals and has $P \cdot M^{L-1}$ states. The ISI channel of length L_c can be described separately by $(P \cdot M^L)^{L_c-1}$ states. If we describe the CPM and the ISI by a joint trellis, only

$P \cdot M^{L-1} \times M^{L_c-1}$ states are needed because the input to the joint trellis is M-ary. For $M = 2$, $P = 4$, $L = 2$, and $L_c = 4$, the CPM trellis produces one of $P \cdot M^L = 16$ signals and has $P \cdot M^{L-1} = 8$ states. The ISI channel can be described separately by $(P \cdot M^L)^{L_c-1} = 16 \times 16 \times 16 = 4096$ states, but this number of states is infeasible for SISO algorithms. If we describe the CPM and the ISI by a joint trellis, only $P \cdot M^{L-1} \times M^{L_c-1} = 64$ states are needed. Note that although the separate processing is much more complex, the performance of the separate processing is worse than that of the joint processing because the joint processing is locally optimal and the separate processing uses the suboptimal iterative decoding.

In Table 3.2, we compare complexity of joint trellis and separate trellis for various CPM schemes and ISI channel length. Note that even for the simplest case, such as Minimum Shifting Keying (MSK) and $L_c = 2$ ISI in the second row in Table 3.2, complexity of the separate processing is greater than that of the joint processing.

Table 3.2: Complexity Comparison of Joint Trellis and Separate Trellis: M-ary, $h = K/P$, L-partial response CPM, L_c=ISI channel length

CPM	ISI	number of states for separate		number of states for joint
		CPM	ISI	
M=2, h=1/2, L=1	L_c=2	2	4	4
M=2, h=3/4, L=2	L_c=4	8	4096	64
M=4, h=1/3, L=2	L_c=3	12	2304	192
M=4, h=1/3, L=3	L_c=3	48	36864	768

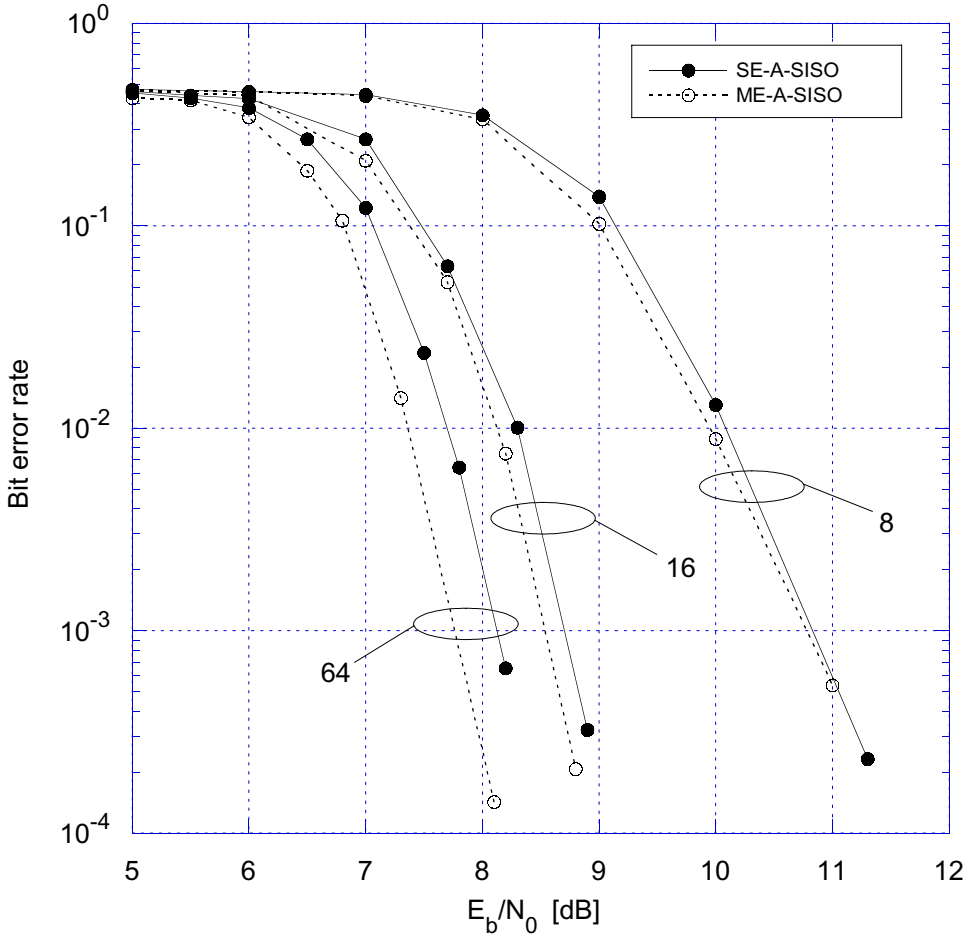

Figure 3.4: Performance of single-estimator A-SISO and multiple-estimator A-SISO ($\nu_d = 0.002$). (64 is full state and 16 or 8 are reduced states)

3.5 Numerical Results

In this section we present simulation results for the system described in Section 3.2. The outer convolutional code of rate 1/2 and constraint-length 4 is used. The code-bits of the convolutional encoder are bit-interleaved using an interleaver with size $1800 = 30 \times 60$. Each block of 60 interleaved code-bits with an additional training sequence is referred as a burst. Bursts are continuous-phase-modulated with an $M =$

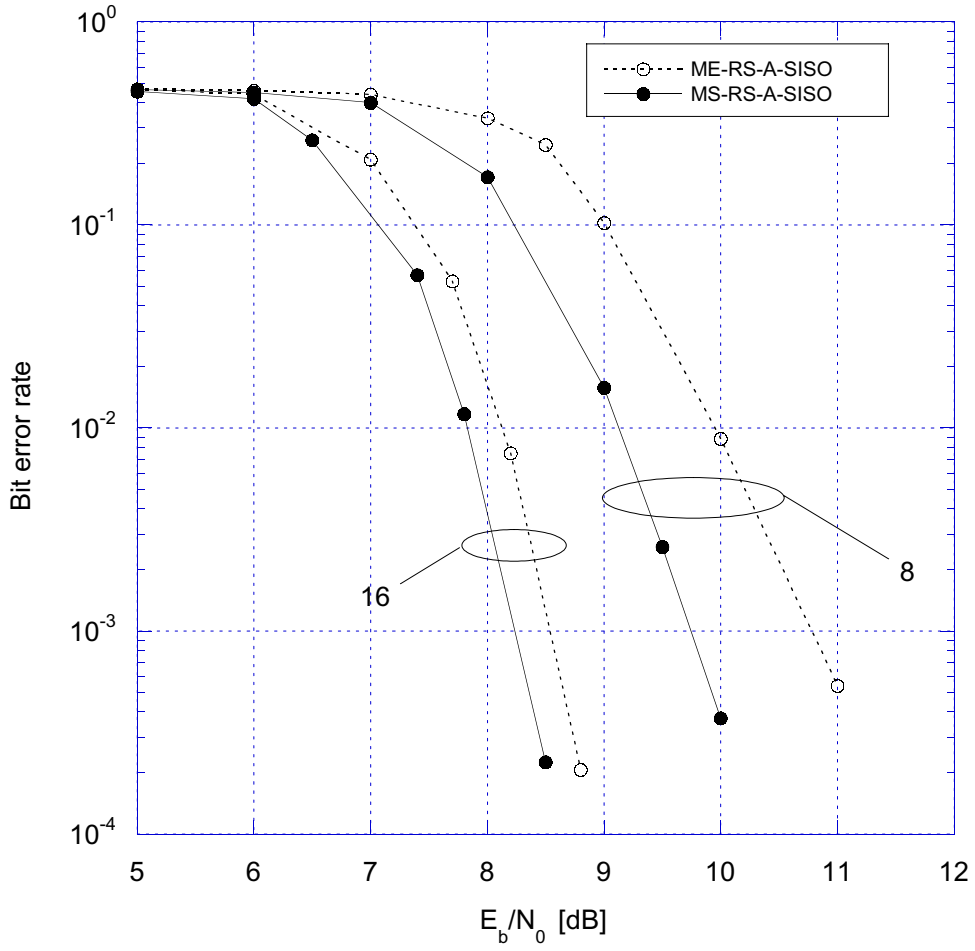

Figure 3.5: Performance of multiple-step RS-A-SISO and multiple-estimator RS-A-SISO ($\nu_d = 0.002$). (16 or 8 are reduced states)

2, L = 2-RC, $h = K/P = 3/4$ partial response CPM scheme ($L > 1$), which has the following frequency pulse function

$$g(t) = \begin{cases} \frac{1}{2LT}[1 - \cos(\frac{2\pi t}{LT})], & 0 \le t \le LT \\ 0, & \text{otherwise} \end{cases} \tag{3.11}$$

where T is a symbol time. The CPM signals of each burst are sent over $N_c + 1 = 7$ tap equal power Rayleigh fading channels with maximum normalized Doppler spreads

$\nu_d = f_d T = 0.002$ and $\nu_d = 0.01$. The taps of the DT channel model are generated under the wide sense stationary, uncorrelated scatter (WSSUS) assumption from an independent Gaussian process with the Clarke spectrum

$$R_f(m) = J_0(2\pi\nu_d m) \tag{3.12}$$

where $J_0(\cdot)$ is the zero-order Bessel function of the first kind. The channel resolution T_c is chosen as $T/2$, which means $N_s = 2$ samples per symbol [1]. Then the length of the channel ISI in terms of the symbol time T is $L_c = \lfloor N_c/N_s \rfloor + 1 = 4$. In Figure 3.3 the performance of the adaptive iterative detection using ME-A-SISO algorithms is shown for the fifth iteration over the ISI channel with $\nu_d = 0.002$. If we decompose the CPM into the CPE and the MM [33], then the CPE has $P \cdot M^{L-1} = 8$ states and the MM produces one of $P \cdot M^L = 16$ signals. The ISI channel can be described separately by $(P \cdot M^L)^{L_c-1} = 16 \times 16 \times 16 = 4096$ states, but this number of states is infeasible for SISO algorithms. If we describe the CPM and the ISI by a joint trellis, only $P \cdot M^{L-1} \times M^{L_c-1} = 64$ states are needed. A joint CPM-ISI A-SISO is used based on the 64-state joint trellis. The joint CPM-ISI FSM is recursive because the CPE FSM is recursive. The state of recursive FSM includes the accumulation (memory) of all the previous input symbols. When the number of states of the joint CPM-ISI FSM is reduced, survivors must have the accumulated value. For every E_b/N_0 we use

[1] The normalized double-sided bandwidth of this CPM signal is 1.165 [3]. This means that 2 samples per symbol are sufficient.

fixed values of the step size $\beta = 0.011$ and $\beta = 0.04$ for $\nu_d = 0.002$ and $\nu_d = 0.01$ respectively assuming the receiver doesn't estimate signal-to-noise ratio (SNR). The optimized tentative delays found via simulation in the single-estimator scheme are $d = 5$ and $d = 3$ for $\nu_d = 0.002$ and $\nu_d = 0.01$ respectively. In this section, each system was identified by the number of states in the joint CPM-ISI A-SISO, i.e.,

- 64 is full-state,
- 16 or 8 are reduced-state.

This figure also includes a perfect CSI case and an AWGN channel case for comparison. For the AWGN channel, the 8-state CPM-SISO is used. Note that ME-A-SISO algorithms show steep slopes as an AWGN channel and a perfect CSI channel.

In Figure 3.4 the comparison of the adaptive iterative detection using SE-RS-A-SISO algorithms and ME-RS-A-SISO algorithms is shown for the fifth iteration. As the number of states is reduced, the performance of the single estimator schemes becomes closer to that of the multiple estimator schemes. At low Doppler spread with $\nu_d = 0.002$, the performance of SE-RS-A-SISO algorithms is quite close to that of ME-RS-A-SISO algorithms. It is shown in Figure 3.6, however, that for more rapidly varying channels with $\nu_d = 0.01$ the SE-A-SISO with 64 full-states reaches an error floor above a BER 10^{-2}. At this Doppler spread $\nu_d = 0.01$ no error floor was observed for the ME-A-SISO with 64 full-states while the ME-RS-A-SISO with 16 reduced-states hits an error floor below a BER 8×10^{-4}.

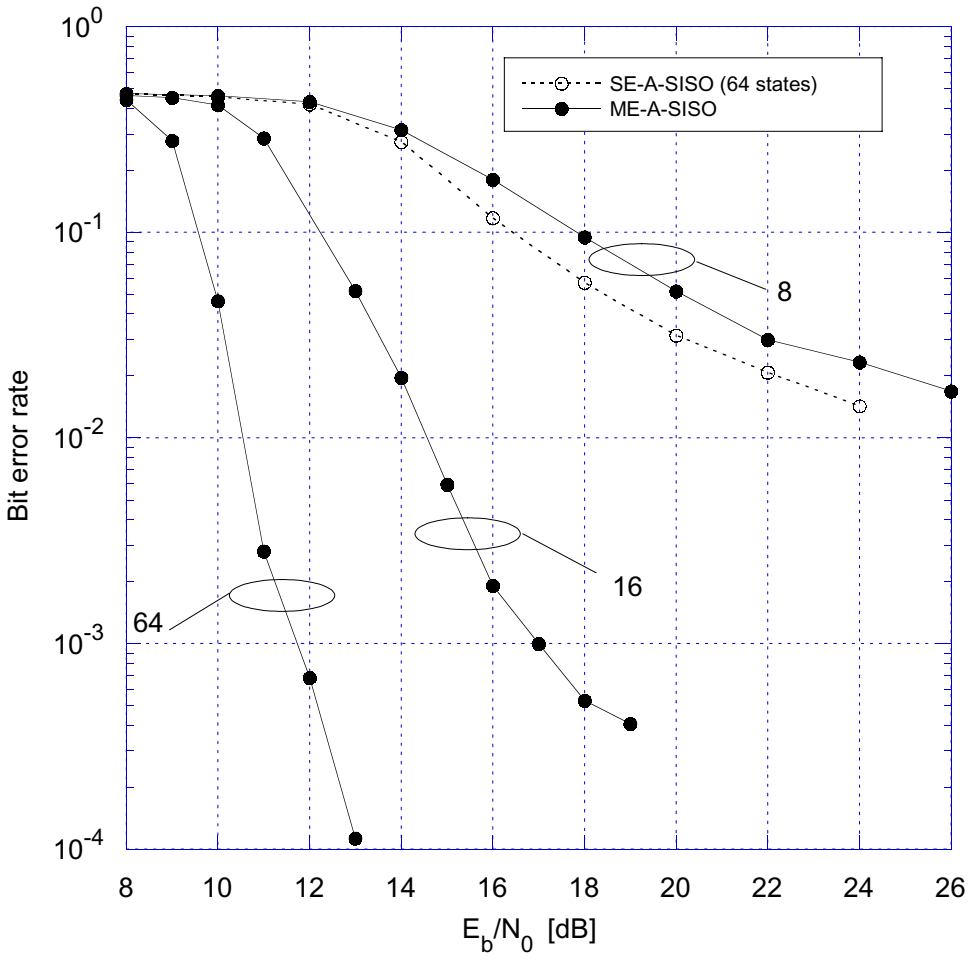

Figure 3.6: Performance of single-estimator A-SISO and multiple-estimator A-SISO ($\nu_d = 0.01$). (64 is full state and 16 or 8 are reduced states)

The comparisons of MS-RS-A-SISO algorithms and ME-RS-A-SISO algorithms are shown for the fifth iteration in Figure 3.5 and Figure 3.7 for $\nu_d = 0.002$ and $\nu_d = 0.01$ respectively. Note that the gain of MS-RS-A-SISO algorithms over ME-RS-A-SISO algorithms increases as the number of states is reduced. The gains at BER 10^{-3} are 0.5 dB and 1 dB for a 16 reduced-state and an 8 reduced-state scheme respectively at $\nu_d = 0.002$ and 2 dB for a 16 reduced-state scheme at $\nu_d = 0.01$.

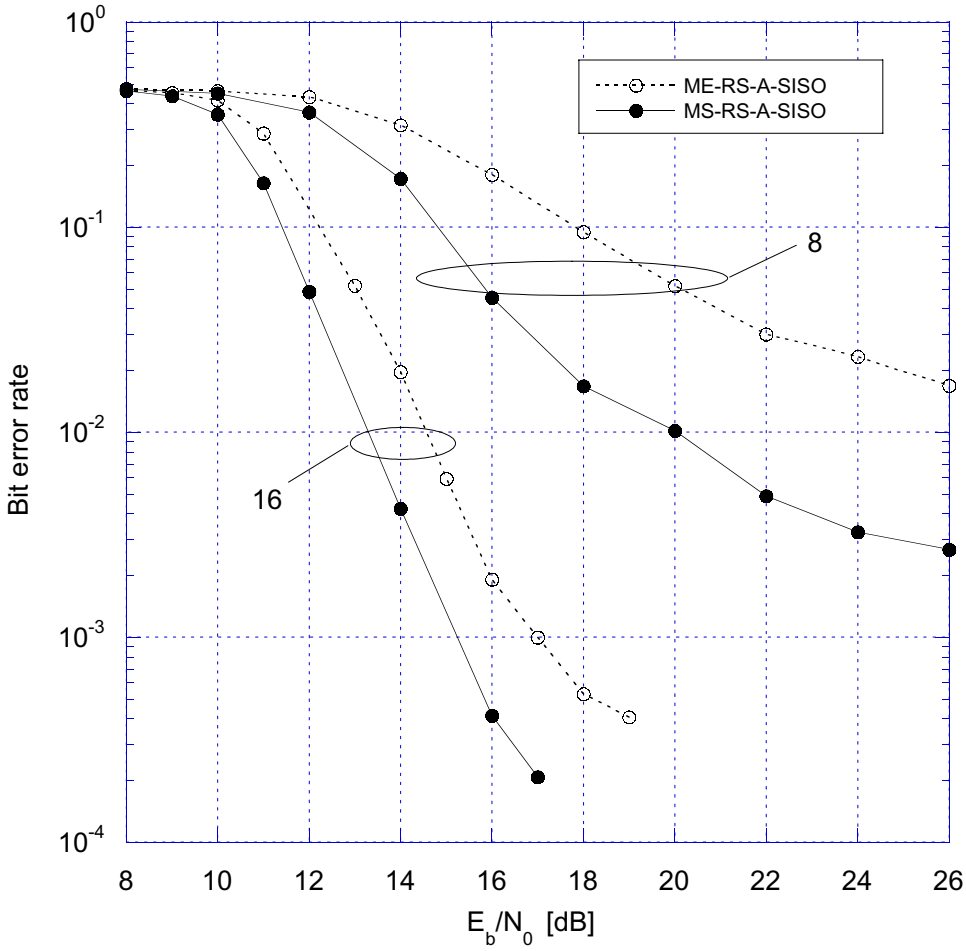

Figure 3.7: Performance of multiple-step RS-A-SISO and multiple-estimator RS-A-SISO ($\nu_d = 0.01$). (16 or 8 are reduced states)

3.6 Density Evolution Analysis of Iterative Detection

Density evolution [32] [31] has recently been used to analyze iterative detection. For short block size, the analysis of iterative detection is an open problem, because the iid assumption for exchanged messages is not satisfied. Such analysis, however, is possible under certain assumptions for very large block size. The density evolution analysis

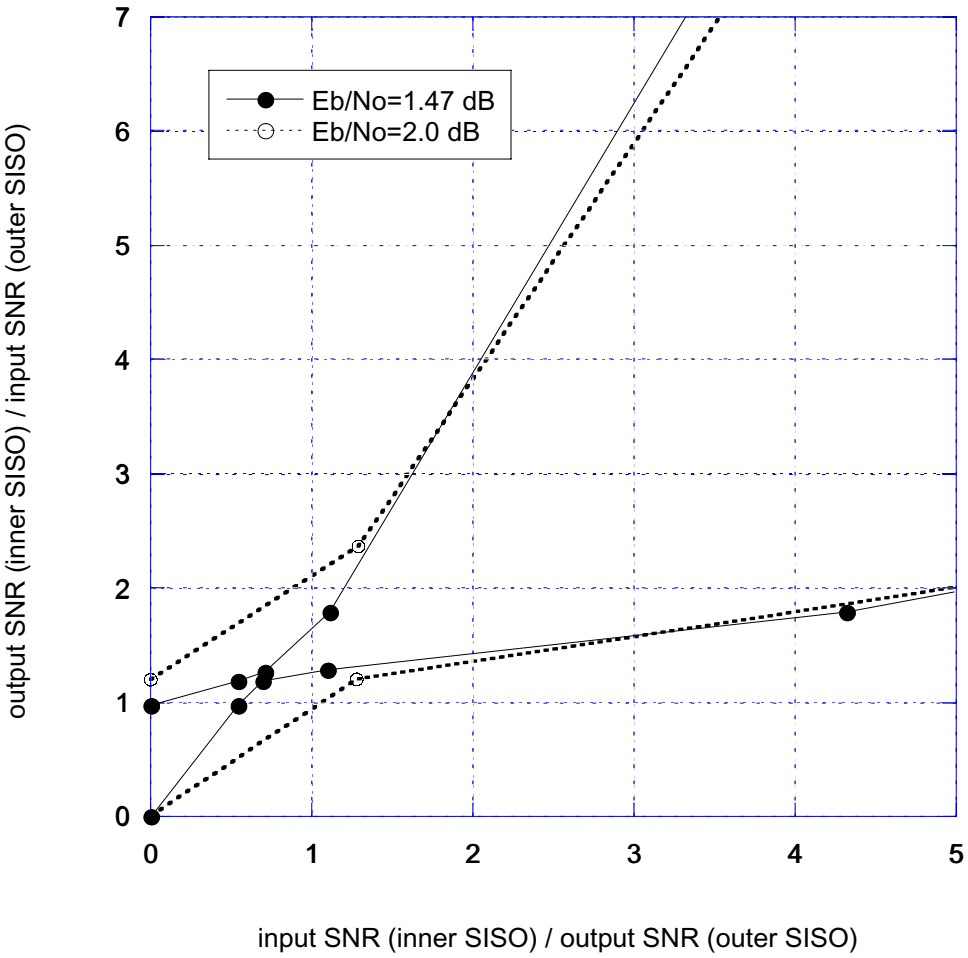

Figure 3.8: Actual SNR evolution of AWGN channel at the threshold, i.e., Eb/No=1.47 dB (for comparison, actual SNR evolution at Eb/No=2 dB is included)

tracks the probability density function (pdf) of the extrinsic information messages as this density evolves from iteration to iteration. In [32], the density evolution technique was recursively used to track the density of extrinsic message between the variable node and check nodes of Low Density Parity Check (LDPC) codes. The density evolution technique was simplified with a Gaussian approximation in [14]. Most analytic research has been focused on LDPC codes because the LDPC decoding algorithm can be represented by a simple two-state trellis or parity-check equations.

The SISO module in Turbo decoders was considered as a signal-to-noise ratio transformer in [21]. It was found in [19] that the Gaussian model based on the empirical mean only assuming that the variance is determined by the symmetry condition proposed in [31] gave closer agreement with the results obtained from actual density evolution than the Gaussian model based on the empirically determined mean and variance as independent parameters. It was shown in [21] that the threshold is approximately matched to the region where the bit error rate curves start to fall down. In [19], they used the density evolution to explain many mysteries of turbo codes and SCCCs, for example, the role of systematic bits and the role of recursive constitute codes etc. As an another application of these DE techniques, the effect of soft information scaling was explained using both analytic and simulation based DE in [24]. In Table 3.3 density evolution analysis is summarized.

Table 3.3: Classification of Density Evolution (DE) analysis (GA: Gaussian Approximation)

	Analytic DE	Simulation based DE
output	threshold	SNR evolution curves
tracking	pdf (w/o GA), mean (w/ GA)	mean (w/ GA)
system	regular/irregular LDPC codes	Turbo codes, SCCCs

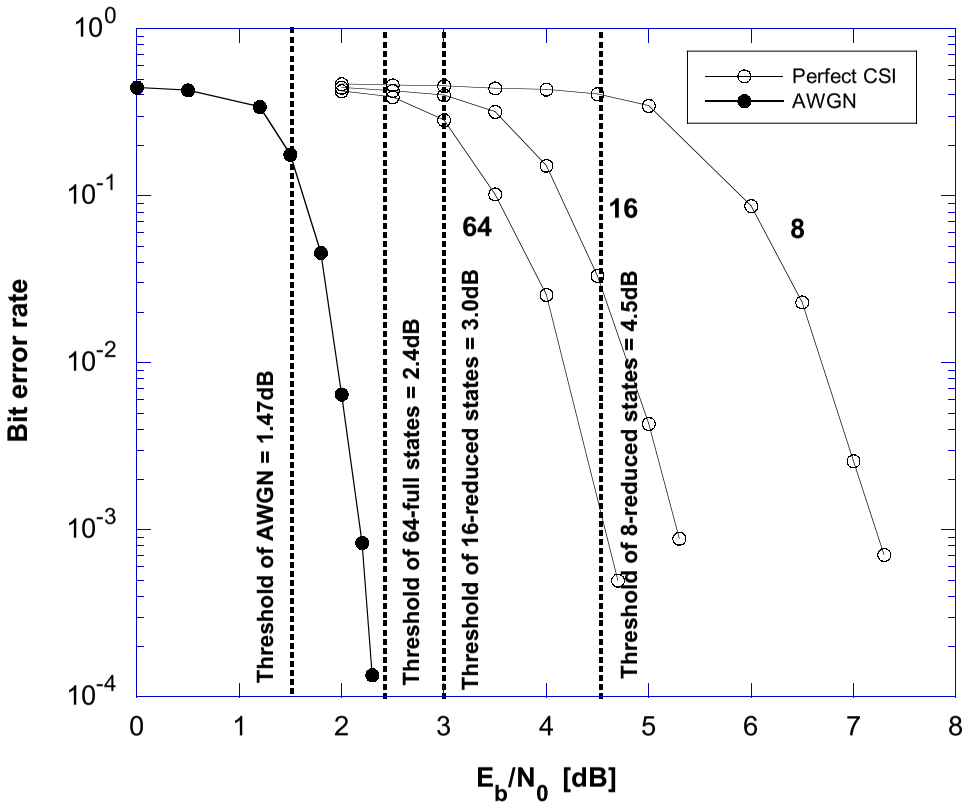

Figure 3.9: Thresholds of full-state and reduced-state PCSI systems ($\nu_d = 0.002$).(64 is full state and 16 or 8 are reduced states)

3.7 Density Evolution Analysis of Reduced State Adaptive SISO Algorithms

When it is difficult to apply the analytic density evolution, such as turbo codes and SCCCs, the SNR evolution is the only way to find thresholds. The SNR [19] is defined as $\mu/2$, where μ is the mean of extrinsic soft information. The SNR evolution is obtained by plotting the output SNR versus the input SNR for one component decoder and the input SNR versus the output SNR for the other component decoder.

If the two curves do not intersect, then the iterative decoder converges. The iterative decoding tunnel will be closed at the SNR where the two curves touch. If E_b/N_0 is greater than this threshold, the decoder converges and the BER goes to zero as the iterations increase and the block size goes to infinity. In Figure 3.8 the actual SNR evolution curves of the AWGN channel are shown at E_b/N_o=1.47 dB. The actual SNR evolution curves of the AWGN channel are at the threshold because the tunnel of two curves is close. For comparison the actual SNR evolution curves of the AWGN channel at E_b/N_o=2.0 dB are shown. Since the threshold E_b/N_0=1.47 dB is far from E_b/N_0=2.0 dB, the actual density evolution curves at E_b/N_0=2.0 dB are open wide.

In Figure 3.9 thresholds of full-state and reduced-state PCSI systems are shown. This figure also includes threshold of the AWGN channel. The performance degradation between the 64-full state system and the 16-reduced state system is about 0.6 dB. This can be compared with the threshold degradation 0.6 dB. When the number of states is reduced to 8 states, the SNR loss at BER 10^{-2} is about 1.8 dB. This SNR loss is approximately consistent with the threshold loss 1.5 dB. The reason that the degradation of the thresholds is less than that of simulations might be using finite interleaver and iteration for simulations. In Figure 3.10 thresholds of full-state and reduced-state ME-A-SISO systems with $\nu_d = 0.002$ are shown. Similarly this figure can be explained as in case of PCSI.

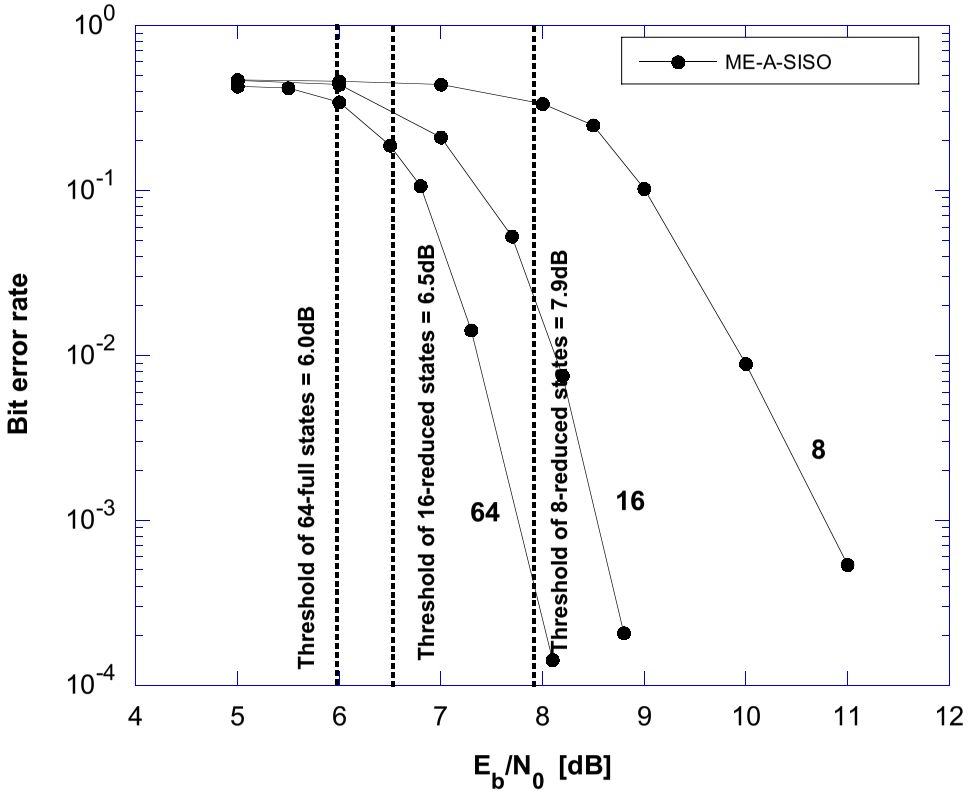

Figure 3.10: Thresholds of full-state and reduced-state ME-A-SISO systems (ν_d = 0.002). (64 is full state and 16 or 8 are reduced states)

3.8 Conclusions

SCCPM with AID showed turbo-like performance over fading ISI channels. RS-A-SISO algorithms were proposed and various design options for RS-A-SISO algorithms were evaluated for SCCPM over frequency-selective fading channels. RS-A-SISO systems also showed large iteration gains. As the number of states used decreases, the performance of the SE-RS-A-SISO algorithms approaches that of the ME-RS-A-SISO algorithms at low Doppler spread. For more rapidly varying channels full-state ME-A-SISO algorithms did not show an error floor but full-state SE-A-SISO

algorithms showed an error floor. MS-RS-A-SISO algorithms can be a design option to improve performance. Using density evolution technique, RA-A-SISO algorithms were analyzed. We showed that density evolution technique that is usually used for AWGN systems is also a good analysis tool for RS-A-SISO systems over frequency-selective fading channels.

Chapter 4

A Simple Stopping Criteria for the MIN-SUM Iterative Decoding Algorithm

4.1 Introduction

Complexity reduction is one of major goal when an iterative decoding is implemented. To reduce the average number of iteration, several stopping criteria have been suggested in the literature. Those stopping conditions are based on a fact that, in relatively high SNR region, many of iteration is executed with little performance improvement. To know a proper instant of this convergence, a stopping rule based on the CE was shown as an effective method in [23]. However, the CE criteria is not realistic due to its complexity. In [35], two simplified version of CE criteria were suggested for Turbo decoding. One is checking the ratio of sign changes of extrinsic information during consecutive iterations. The ratio of sigh changes is compared to

a empirical threshold to determine the moment when iterative decoding is stopped. For example, the iteration stops when

$$T(l) \leq (0.005 \sim 0.3)N \tag{4.1}$$

where $T(l)$ is the number of sign changes in extrinsic information $L_e^{(l)}(u_k)$ at l-th iteration and N is the input block size. This algorithm was named as sign-change-ratio(SCR). The other is comparison of hard decision during consecutive iteration named as hard-decision-aided(HDA). If current hard decision is completely matched to the previous hard decision, the iterative decoding is stopped.

$$\hat{u}_k^{(l)} = \hat{u}_k^{(l-1)}, \forall k, 1 \leq k \leq N \tag{4.2}$$

The effectiveness of both algorithms were shown by sum-product algorithm on a Turbo decoder. The conclusion was that the HDA criteria had less complexity (average number of iterations) at low and mid SNR region and the SCR criteria showed less complexity at high SNR.

In this work, a new simple stopping criteria is proposed with better complexity reduction (i.e., less average number of iterations) and less required memory. Based on the fact that a min-sum algorithm is a sequence decoding not a symbol decoding, the new stopping criteria checks if a decoded sequence is valid on the encoded trellis structure with the extrinsic information. Depending on the encoded system, each

state at time k can be mapped into only two possible states at time $k+1$ in a binary input system and when a sequence completely follows this mapping structure we say it is valid (i.e., it is a valid transmitted codeword C).

$$\hat{\mathbf{u}}_1^{N(l)} \in C \tag{4.3}$$

Note that the intrinsic information of the min-sum algorithm always gives a valid decoded sequence. A simple loop-up table describing one transition of a trellis can be used for this valid sequence check. If an entire decoded sequence is a valid codeword, the iterative decoding is stopped. Unlike the CE and SCR criteria, there is no need to optimize the threshold empirically.

4.2 Simulation results

A serially concatenated convolutional codes (SCCC) is considered with the input block size $N = 1024$. Each of constituent code is $\frac{1}{2}$-rate systematic recursive convolutional code and the BPSK modulation is considered. The stopping algorithms are applied based on the output of outer Soft-In-Soft-Output (SISO) decoder at every iteration.

In Fig. 4.2, the performance of different stopping rules are shown with the performance of fixed iteration (10). The HDA and sequence check criteria (SEQ) show the same performance as that of fixed-number iteration decoding. In other words,

when a stop criteria determine to stop an iterative decoding, the actual bit error at the corresponding iteration is zero. However, when the actual bit error is zero, the stop criteria may determine to continue the iterative decoding. This increases the required number of iteration (i.e., decoding complexity). For the SCR criteria, the performance and required number of iteration are design parameter. We chose the threshold $T(l) = 0.03N$ which gives the same performance as that of fixed iteration. When stop criteria has the same performance as that of fixed-number iteration, the effectiveness of a stopping rule can be shown with the average number of iteration. In Fig. 4.3 the average number of iteration was shown for the HDA, SCR, and SEQ algorithms. The proposed SEQ algorithm has about $\frac{1}{2}$ number of iteration reduction compared to both HDA and SCR algorithms at intermediate and high SNR region for both block size. It means that the SEQ requires less complexity than the HDA or SCR does. In addition, since the HDA and SCR algorithms need to compare hard decision of current iteration with those of previous iterations, it requires to save the previous decision values. In the mean time, the SEQ uses only current hard decision values and it does not need to save any decision value. Moreover, while the entire block should always be checked for the next iteration in the HDA and SCR rule, the SEQ is only executed by the point where the invalid sequence check occurs. This invalid sequence check usually occurs in the middle of a block except the last iteration where the SEQ stops the iteration. These advantage of the SEQ algorithm save memory and complexity.

Although the results were shown for SCCC case, the extension for Turbo code is trivial. Because Turbo code has different characteristic of soft information evolution with that of SCCC, the effectiveness of stopping rules may different in Turbo decoding.

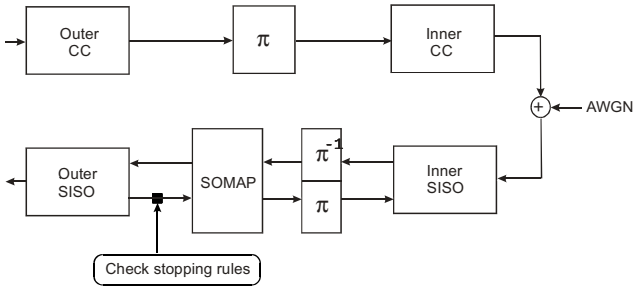

Figure 4.1: Block diagram of SCCC encoder and decoder with stopping rules

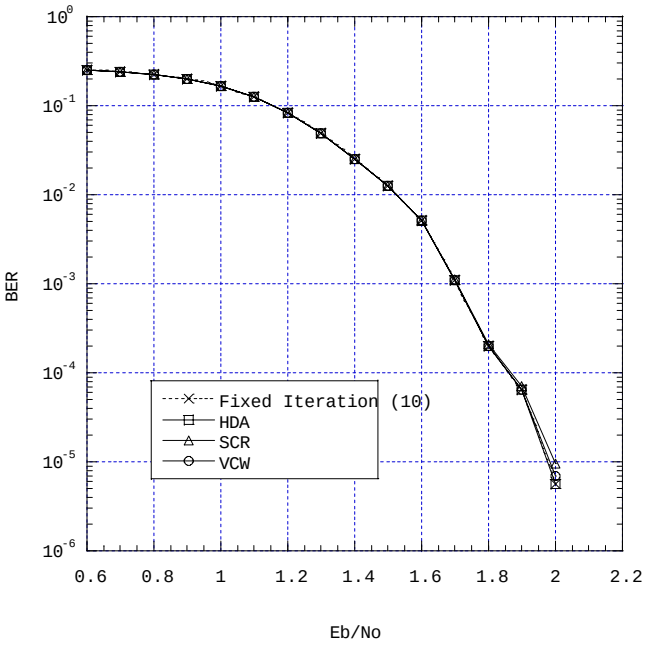

Figure 4.2: Performance of different stopping criteria with the fixed-number iterative decoding for a input block length N=1024

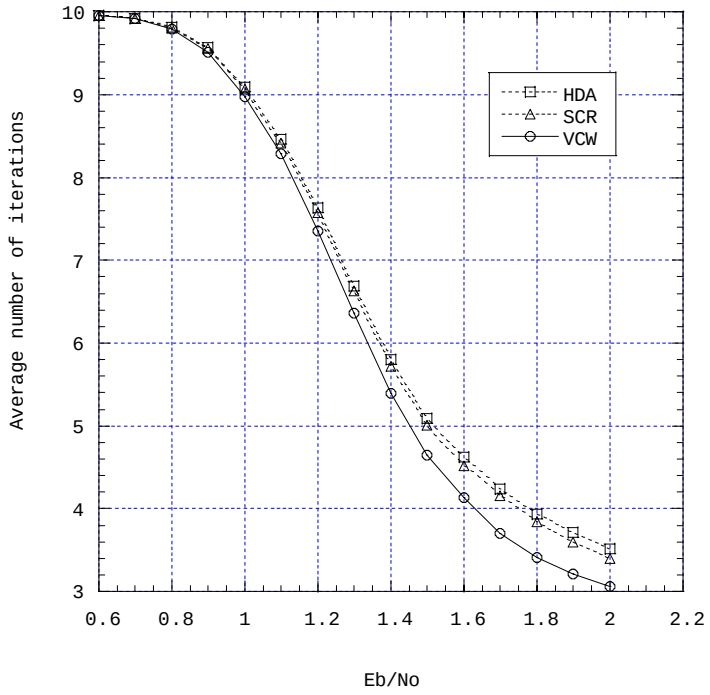

Figure 4.3: Average number of iterations for different stopping criteria for a input block length N=1024

4.3 Conclusion

In this work, we proposed the VCW stopping rule for min-sum iterative decoding algorithms. The performance and computational complexity are compared with the simplified cross entropy-based SCR and hard decision algorithm HDA. The proposed VCW algorithm requires less computational complexity and less memory while there is no performance degradation compared to the iterative decoding algorithm using a fixed number of iterations and there is no need for empirical parameters.

Chapter 5

Upper Bounds on the Performance of Turbo-Like Codes for Specific Interleavers

5.1 Introduction

Iterative detection for concatenated codes with interleavers represents a great advancement in communications theory because of their excellent performance. Parallel concatenated convolutional codes (PCCCs, or turbo codes) with interleavers, introduced in [6], consist of simple binary convolutional codes connected in parallel through an interleaver.

In [18], transfer function bounding techniques were applied to obtain upper bounds on the bit-error rate for maximum-likelihood decoding of turbo codes constructed with pseudo-random interleavers. Since union bounds are intractable for any particular pseudo-random interleaver, the transfer function bound is developed as a random

coding bound based on the uniformly interleaved assumption, i.e., performance averaged over all possible interleaver permutations. Therefore, the transfer function bound can not be used to bound the performance of the turbo code with a particular interleaver.

Since the union bound cannot predict the performance above the cutoff rate, there is a great demand to have bounds on performance that are useful for rates above the cutoff rate. One of those bounds is the simple bound [17]. In [17], the simple bound is used with the uniformly interleaved assumption for turbo codes. Therefore, bounds on the maximum-likelihood decoding performance of the turbo-like codes with a particular interleaver are still unavailable. Note also that simulation performance of turbo-like codes using iterative detection is suboptimal because iterative detection is approximate to the optimal detection.

In this paper we solve intractability of upper bounds with any particular pseudo-random interleaver by using approximate input-output weight distributions. The performance of turbo-like codes at high SNR are well approximated by the expression of the union bound, truncated to the contribution of the several smallest non-zero distance terms [22]. In [22] branch and bound algorithms [26] were developed for finding the several smallest distances and their multiplicities. We include these exact distance spectrum results for several of the smallest non-zero distances into the approximate weight distributions. The reliability of the estimated upper bound is statistically

evaluated. We conjecture that the thresholds of the simple bound of turbo-like codes for specific optimized spread interleavers are approximately the channel capacity.

The paper is organized as follows. A review of transfer function bounds is given in Section II. Section III summarizes a simple tight bound. In Section IV we compare transfer function bounds, simple bounds, and simulation results. Free-distance truncated union bounds is explained in Section V. In Section VI upper bounds for specific interleavers are presented. Reliability measures for the upper bounds is given in Section VII. Section VIII concludes the paper.

5.2 Transfer Function Bounds on Word and Bit-Error Probabilities

For maximum-likelihood decoder a union bound on the probability of word error and bit error over an additive white Gaussian noise channel with channel symbol signal-to-noise ratio E_s/N_o requires the input-output weight distribution. For the overall (N, K) code C with code rate $r = K/N$, $A_{w,d}$ denote the number of codewords for input sequence weight w and output codeword weight d. Then the conditional

probability of producing a codeword of weight d given an input sequence of weight w is

$$p(d|w) = \frac{A_{w,d}}{\sum_{d'} A_{w,d'}} = \frac{A_{w,d}}{\binom{K}{w}}. \tag{5.1}$$

The conditional probability distribution $p(d|w)$ for turbo codes is obtained by using the uniformly interleaved assumption [18]. The conditional probability that a maximum-likelihood decoder will choose a particular codeword of total weight d to the all-zero codeword is $Q(\sqrt{2dE_s/N_0})$, where $Q(\cdot)$ is the complementary unit variance Gaussian distribution function. Then, the codeword error probability P_w is upper bounded as follows:

$$\begin{aligned} P_w &= \sum_{w=1}^{K} \Pr\{\text{error event of input weight } w\} \\ &\leq \sum_{w=1}^{K} \binom{K}{w} \mathrm{E}_{d|w} \left\{ Q\left(\sqrt{\frac{2dE_s}{N_0}}\right) \right\} \\ &\leq \sum_{w=1}^{K} \binom{K}{w} \left[\sum_{d} p(d|w) \left\{ Q\left(\sqrt{\frac{2dE_s}{N_0}}\right) \right\} \right] \end{aligned} \tag{5.2}$$

where the conditional expectation $\mathrm{E}_{d|w}\{\cdot\}$ is over the probability distribution $p(d|w)$. Similarly, the information bit-error probability P_b is upper bounded by

$$\begin{aligned} P_b &= \sum_{w=1}^{K} \frac{w}{K} \Pr\{\text{error event of input weight } w\} \\ &\leq \sum_{w=1}^{K} \frac{w}{K} \binom{K}{w} \mathrm{E}_{d|w} \left\{ Q\left(\sqrt{\frac{2dE_s}{N_0}}\right) \right\} \\ &\leq \sum_{w=1}^{K} \frac{w}{K} \binom{K}{w} \sum_{d} p(d|w) \left\{ Q\left(\sqrt{\frac{2dE_s}{N_0}}\right) \right\} \end{aligned} \tag{5.3}$$

The divergence properties of the transfer function bounds for turbo codes is observed above the cutoff rate [18].

5.3 A Simple Tight Bound

The performance of turbo-like codes is close to Shannon's channel capacity limit for moderate to large block sizes, so there is a need for bounds on performance that are useful for rates above the cutoff rate. In [17] such a simple bound on the probability of decoding error for block codes is derived in closed form. This bound is based on the bounding technique developed by Gallager in 1963 [20]. This bound is simple because it does not require any integration or optimization in its final version.

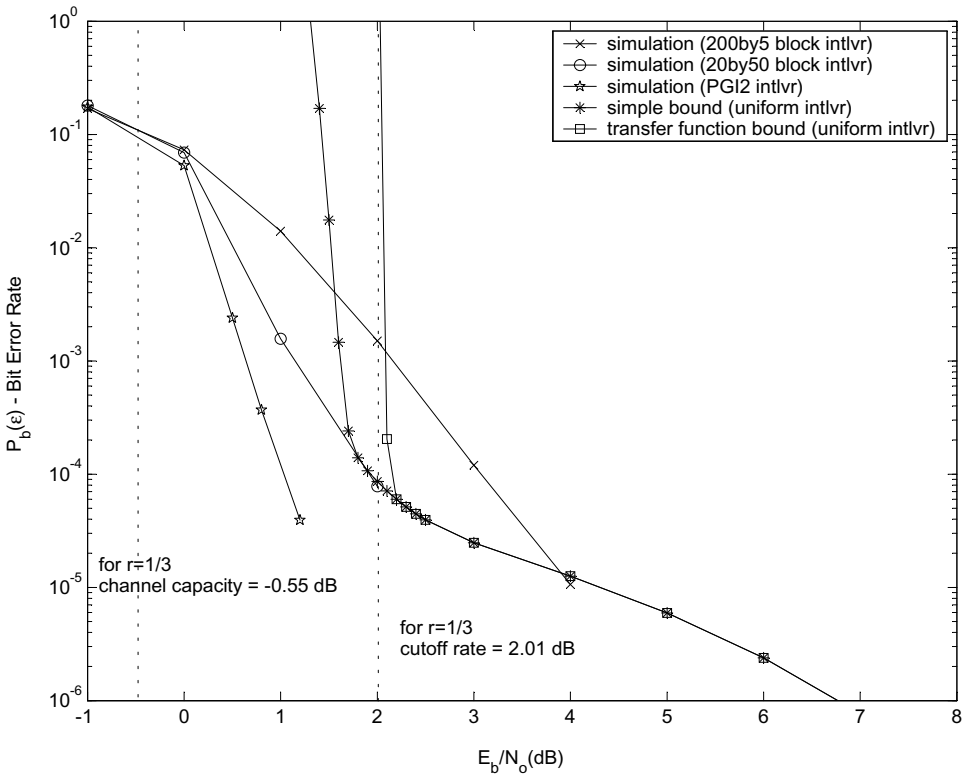

Figure 5.1: Upper bounds with uniform interleavers and simulations with various interleavers for turbo code (r=1/3, K=1000)

Consider a linear binary (N, K) block code C with code rate $r = K/N$. For a given code d is the Hamming weight of a codeword. The upper bound on the error rate with maximum likelihood codeword decoding is given by

$$P \leq \sum_{d=d_{min}}^{N-K+1} \min\left\{ e^{-nE(c,d)}, e^{ng(\delta)} Q\left(\sqrt{2ch}\right)\right\} \tag{5.4}$$

where

$$\text{if} \quad c_0(\delta) < c < \frac{e^{2g(\delta)} - 1}{2\delta(2-\delta)}, \tag{5.5}$$

$$E(c,d) = \frac{1}{2}\ln[1 - 2c_0(\delta)f(c,\delta)] + \frac{c\, f(c,\delta)}{1 + f(c,\delta)}, \tag{5.6}$$

$$\text{otherwise} \qquad E(c,h) = -g(\delta) + \delta\, c \tag{5.7}$$

$\delta = d/N, \quad c = R_c(E_b/N_0)$, and

$$f(c,\delta) = \sqrt{\frac{c}{c_0} + 2c + c^2} - c - 1 \tag{5.8}$$

and for the bit-error rate

$$g(\delta) = \frac{1}{N}\ln\left\{\sum_w \frac{w}{K} A_{w,d}\right\} \tag{5.9}$$

where $A_{w,d}$ is the input-output weight distribution. Specifically $A_{w,d}$ is the number of codewords with the Hamming weight d for input sequences of the Hamming weight w. The frame-error rate can be upper bounded using

$$g(\delta) = \frac{1}{N}\ln A_d \tag{5.10}$$

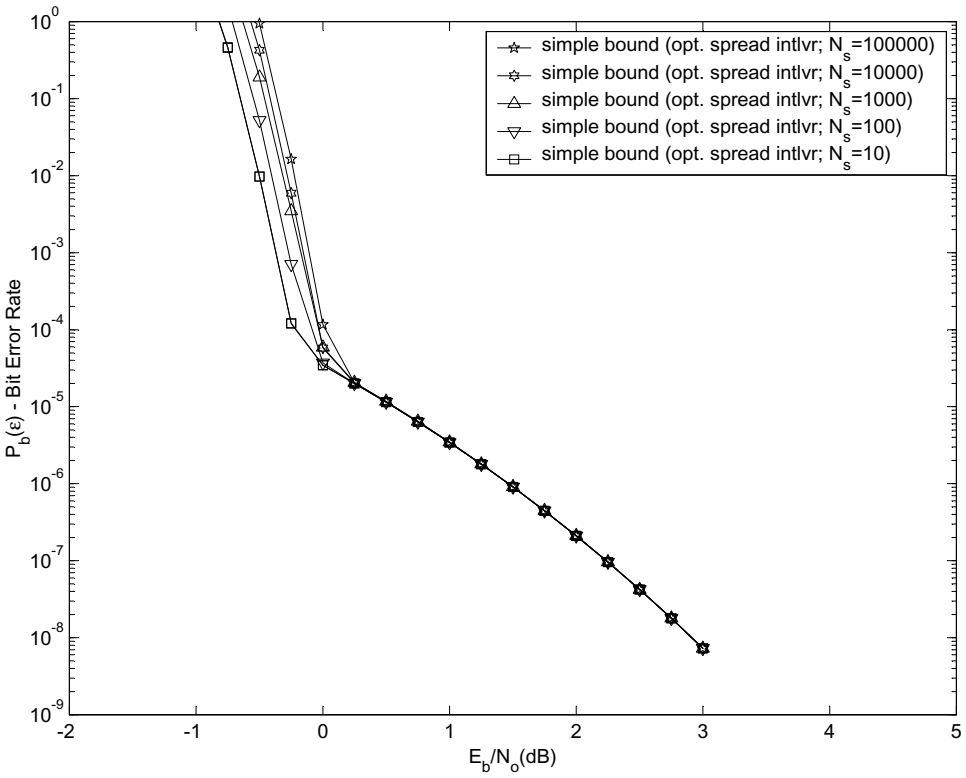

Figure 5.2: Simple bounds for different N_s number of generated sample codewords

where A_d is the output weight distribution. Specifically A_d is the number of codewords with the Hamming weight d.

5.4 Comparison of Transfer Function Bounds, Simple Bounds, and Simulation Results

Figure 5.1 compares upper bounds with an uniformly interleaved assumption (transfer function bounds and simple bounds) with simulation results for various interleavers. The turbo code uses constituent convolutional codes with the generator polynomial

$(1 + D)/(1 + D + D^2)$. The rate of the turbo code is 1/3. The three simulation results are obtained from three different interleavers with length $K = 1000$. The first is an optimized spread interleaver [16], which shows the best performance among the three interleavers. The second is a block interleaver, which reads bits in 20 by 50 rectangular array row-wise and reads out column-wise. The third is an block interleaver with 200 by 5 rectangular array. While the transfer function bound uses the uniform interleaver assumption, these results show that performance depends on the specific interleaver. Note that the performance of the iterative decoder can be worse than the upper bound on the maximum-likelihood decoding performance because iterative detection is suboptimal.

Figure 5.1 also compares the simple bound with the transfer function bound. We observe that the simple bound is tighter than the transfer function bound at the low range of SNR. At a BER of 10^{-1} the simple bound is about 0.7 dB tighter than the transfer function bound. It is well known that the transfer function bound diverges above the cutoff rate. The divergence of the transfer function bound that is observed in the low range of SNR is not observed for the simple bound until $E_b/N_0 = 1.5$ dB. The cutoff rate corresponds to $E_b/N_0 = 2.01$ dB for rate 1/3 codes.

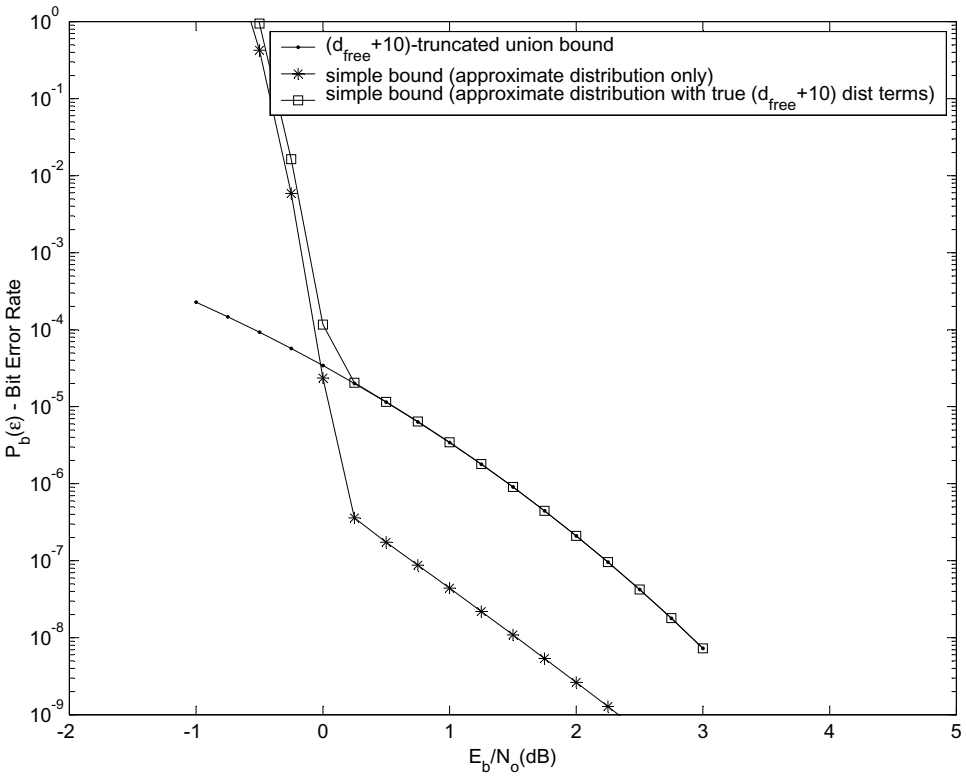

Figure 5.3: Comparison of upper bounds of the true several smallest distance terms, the approximate distribution only, and the approximate distribution with the true several smallest distance terms (opt. spread interleaver, r=1/3, K=1000)

5.5 Free-distance Truncated Union Bounds

For a linear binary code $C(N, K)$ (N is the codeword length and K the information frame length) with free distance d_{free}, we will denote by N_{free} its multiplicity (the number of codewords with weight d_{free}), and by w_{free} its information bit multiplicity (defined as the sum of the Hamming weights of the N_{free} information frames generating the codewords with weight w_{free}). For very high values of E_b/N_0, where E_b is

the energy per information bit and N_0 the one-sided noise spectral density, we can write

$$\mathrm{FER} \simeq N_{free}\mathrm{Q}\left(\sqrt{\frac{2E_b}{N_0}\cdot\frac{K}{N}\cdot d_{free}}\right) \tag{5.11}$$

and

$$\mathrm{BER} \simeq \frac{w_{free}}{K}\mathrm{Q}\left(\sqrt{\frac{2E_b}{N_0}\cdot\frac{K}{N}\cdot d_{free}}\right). \tag{5.12}$$

For turbo-like codes, a better approximation can be obtained by including other terms of the distance spectrum [22]. By the symbol UB(j) we will denote the union bound expression, truncated to the contribute of the j-th nonzero distance,

$$\mathrm{UB}(j) = \sum_{i=1}^{j}\frac{w(i)}{K}\mathrm{Q}\left(\sqrt{\frac{2E_b}{N_0}\cdot\frac{K}{N}\cdot d(i)}\right) \tag{5.13}$$

where $d(i)$ is the i-th nonzero distance and $N(i)$ and $w(i)$ are its multiplicities (as a consequence, $d(1)/N(1)/w(1) = d_{free}/N_{free}/w_{free}$). In [22], branch and bound algorithms [26] for finding the several smallest distances and their multiplicities were developed allowing performance of turbo codes to be approximated by UB(j) at the high SNRs. But since these algorithms are based on branch-and-bound method [26], complexity for finding the whole weight distribution is intractable.

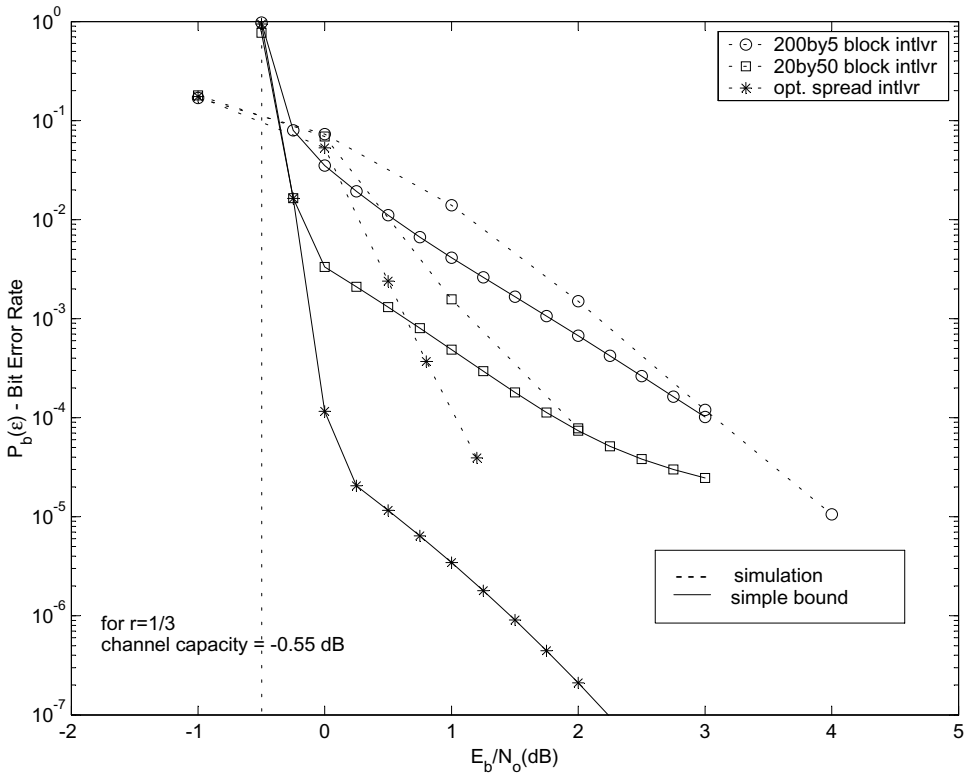

Figure 5.4: Simple bounds and simulations for various interleavers (r=1/3, K=1000)

5.6 Upper Bounds for Specific Interleavers

The simple bound is the tightest closed-form upper bound on decoding error rate [17]. We use the simple bound for specific interleavers. To use the simple bound we need the conditional probability $p(d|w)$ that is defined in Equation (5.1) as

$$p(d|w) = \frac{A_{w,d}}{\binom{K}{w}}. \tag{5.14}$$

But it is intractable to obtain $p(d|w)$ because of complexity. So we want to obtain the maximum-likelihood estimator $\widehat{p}_{\mathrm{MLE}}$ of $p(d|w)$. This problem is the same as

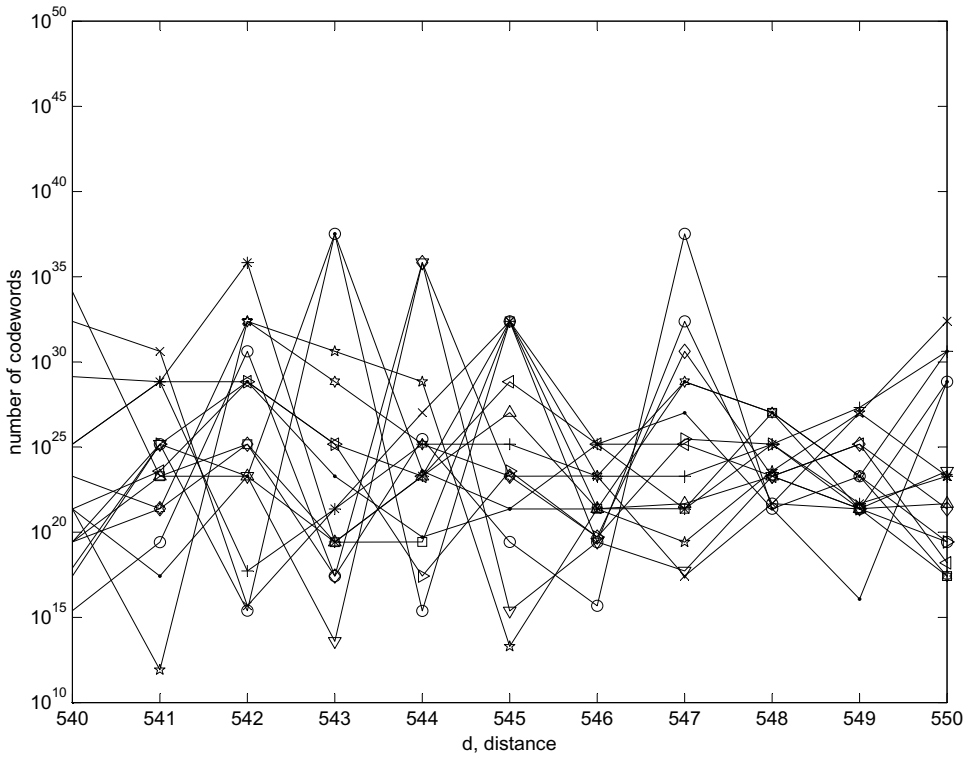

Figure 5.5: $n = 15$ approximate output weight distributions ($N_s = 10000$)

estimating the probability of white balls when a ball is drawn with replacement from an urn that contains $A_{w,d}$ white balls and $[\binom{K}{w} - A_{w,d}]$ black balls. The indicator function $I(c)$ is defined by

$$I(c) = \begin{cases} 1 & \text{if } c \text{ is a codeword with } w \text{ and } d \\ 0 & \text{otherwise.} \end{cases} \tag{5.15}$$

Then $I(c)$ is the Bernoulli random variable with $p = p(d|w)$ [27]. If we let k be the number of codewords with the Hamming weight d for input sequences of the Hamming weight w among N_s generated sample codewords, k is the sum of the Bernoulli random

variables associated with each of the N_s independent trial. Then k is the binomial random variable with the following probability mass function

$$P(k|p) = \begin{pmatrix} N_s \\ k \end{pmatrix} p^k (1-p)^{N_s - k} \tag{5.16}$$

for $k = 0, 1, \cdots, N_s$. In order to obtain the maximum-likelihood estimator $\widehat{p}_{\mathrm{MLE}}$, we maximize the likelihood function $P(k|p)$,

$$\begin{aligned} \widehat{p}_{\mathrm{MLE}} &= \max_p \ P(k|p) && (5.17) \\ &\equiv \max_p \ p^k (1-p)^{N_s - k}. && (5.18) \end{aligned}$$

Differentiating the argument and setting the result equal to 0 give the solution,

$$\widehat{p}_{\mathrm{MLE}} = \frac{k}{N_s}. \tag{5.19}$$

It is also straightforward to verify that this solution is the global maximum. We approximate $p(d|w)$ by $\widehat{p}_{\mathrm{MLE}}$. Then

$$\begin{aligned} p(d|w) &\simeq \widehat{p}_{\mathrm{MLE}} && (5.20) \\ \frac{A_{w,d}}{\binom{K}{w}} &\simeq \frac{k}{N_s} && (5.21) \end{aligned}$$

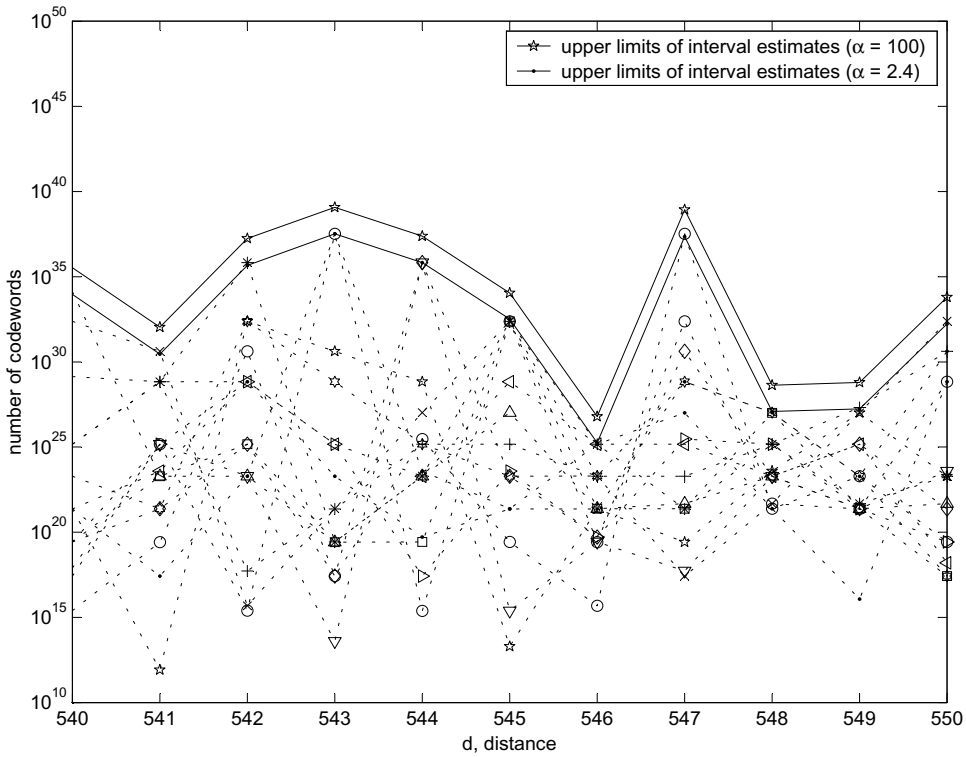

Figure 5.6: Upper limits of interval estimates for different α

$$\frac{\widehat{A}_{w,d}}{\binom{K}{w}} = \frac{k}{N_s} \tag{5.22}$$

$$\widehat{A}_{w,d} = \binom{K}{w} \frac{k}{N_s}. \tag{5.23}$$

In Figure 5.2 simple bounds for different N_s values for each input weight w are shown. We choose $N_s = 10000$ because the thresholds of simple bounds for $N_s = 1000, 10000$, and 100000 were found to be similar. In order to obtain $\widehat{A}_{w,d}$ we generate $N_s = 10000$ codewords randomly for each input weight w, calculate k

from the generated codewords, and obtain the estimated weight distribution $\widehat{A}_{w,d}$ using (5.23) . The true distribution of several smallest distances $\{A_{w,d}\}_{d=d_{min}}^{d=d_{min}+10}$ is included in $A_{w,d}$ using the algorithm in [22] because for very high values of E_b/N_0 these several smallest distance terms are dominant. In Figure 5.3 we compare three cases, i.e., upper bounds including the true several smallest distance terms only, the approximate distributions on their own, or the approximate distributions with the true several smallest distance terms. By including $\{A_{w,d}\}_{d=d_{min}}^{d=d_{min}+10}$, we approximate better error floor region at medium to high SNR.

Figure 5.4 compares simulation results and upper bounds for various interleavers. We obtain three different accurate upper bounds for each interleaver.We also observe that at the low range of SNR the thresholds of the simple bound are approximately the channel capacity of rate $r = 1/3$, i.e., -0.55 dB.

5.7 Reliability Measure of Upper Bounds

In Figure 5.4 we use N_s randomly generated sample codewords to obtain an approximate input-output weight distribution $\widehat{A}_{w,d}$. A natural question arises: What happens to upper bounds if we use a different set of randomly generated sample codewords? Now we answer this question. First we obtain many different sets of randomly generated sample codewords, i.e., $n = 15$, where n is the number of sets of randomly

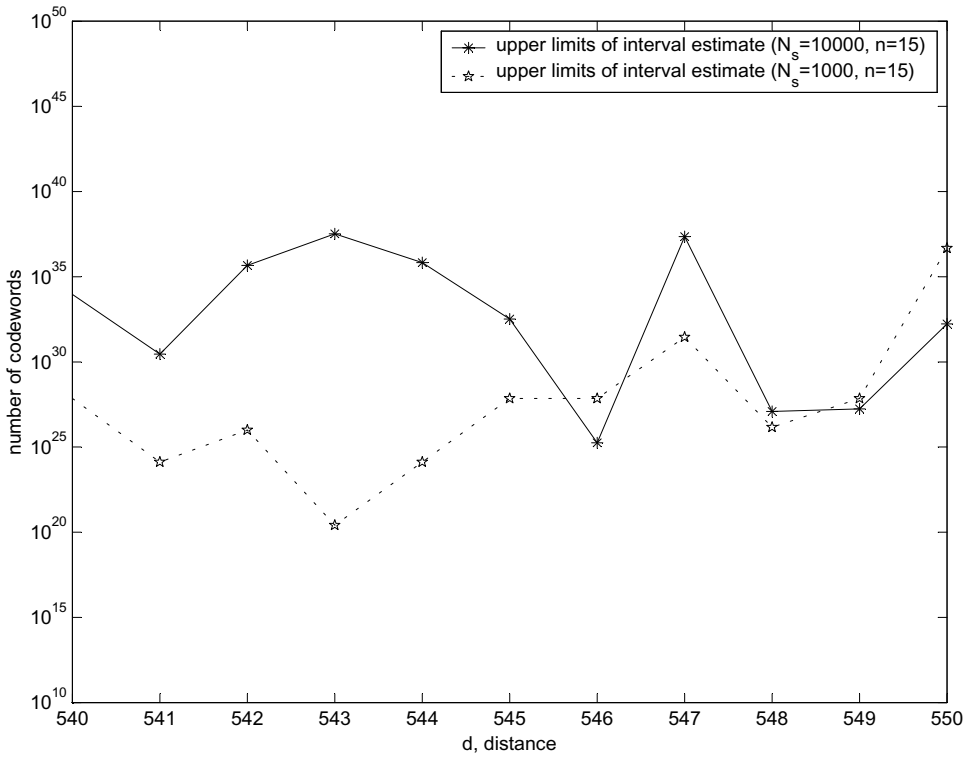

Figure 5.7: Upper limits of interval estimates for different N_s

generated sample codewords. In Figure 5.5 we show these approximate output weight distributions $\widehat{A}_d$, using

$$\widehat{A}_d = \sum_{w=1}^{K} \widehat{A}_{w,d} \tag{5.24}$$

We observe from Figure 5.5 that these approximate weight distributions are different from each other. Note that the larger $\widehat{A}_d$ is, the higher the upper bound is. Our approach is to choose the worst case statistically from the given approximate output weight distributions $\widehat{A}_d$. In order to do this we use one-sided interval estimates [8]. In [8] an interval estimate of a real-valued parameter μ is defined as a function

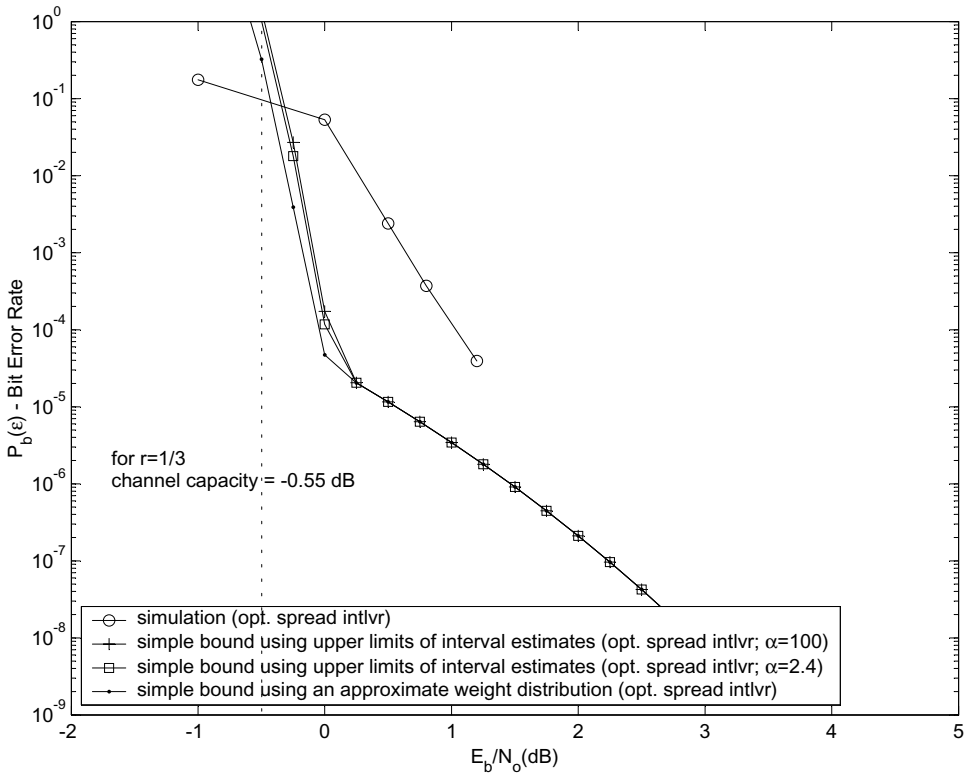

Figure 5.8: Simple bounds using upper limits of interval estimates for an optimal spread interleaver

$U(c_1, c_2, \cdots, c_n)$ of a sample $\mathbf{c} = \{c_i\}_{i=1}^{i=n}$. If $\mathbf{C} = \mathbf{c}$ is observed, the inference $\mu \leq U(\mathbf{c})$ is made, where $\mathbf{C}$ is a random vector and $\mathbf{c}$ is a realization. The random interval $[-\infty,\ \ U(\mathbf{C})]$ is called an interval estimator. Since an interval estimator is a random variable, the estimate is a function of a given sample. Even so, our target is to choose the worst case statistically from given approximate output weight distributions $\widehat{A}_d$. With n estimated weight distributions, we obtain n random samples $\{c_i\}_{i=1}^{i=n}$ for each distance d. Then we calculate the sample means and sample variances for each distance d to obtain one-sided interval estimates assuming that these samples for each distance d are Gaussian distributed. The sample mean is the arithmetic average of

the values in a random sample. It is usually denoted by

$$\overline{c} = \frac{1}{n}\sum_{i=1}^{n} c_i \tag{5.25}$$

where n is the size of a random sample. The sample variance is the statistic defined by

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(c_i - \overline{c})^2. \tag{5.26}$$

For a sample $c_1, c_2, \cdots, c_n$ the probability that the population mean μ belongs to the interval $[-\infty,\ \ \overline{c} + \alpha S]$ can be calculated as

$$P\{\mu \leq \overline{c} + \alpha S\} = P\{\frac{\overline{c} - \mu}{S} \geq -\alpha\} = 1 - Q(\alpha) \tag{5.27}$$

where $Q(\alpha)$ is tail integral of a unit-Gaussian probability density function, and is defined as

$$Q(\alpha) = \int_{\alpha}^{\infty} \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}} dx. \tag{5.28}$$

In Figure 5.6 we show these upper limits of interval estimates for $\alpha = 2.4$ and $\alpha = 100$. We observe that these upper limits of interval estimates with $\alpha = 2.4$ coincide approximately with highest points for each d and that the upper limits of interval estimates with $\alpha = 100$ is higher than that with $\alpha = 2.4$. Therefore we obtain the

worst case statistically from given approximate output weight distributions $\widehat{A}_d$. In Figure 5.7 we compare the upper limits of interval estimates for $N_s = 10000$ and $N_s = 1000$ with $\alpha = 2.4$. We observe that two curves were found to be similar. In Figure 5.8 we compare the upper bound using upper limits of interval estimates and the upper bound using an approximate weight distribution. We observe that the upper bound using upper limits of interval estimates is above the upper bound using an approximate weight distribution. But these bounds are close to each other.

5.8 Conclusion

In this chapter, we studied upper bounds on the performance of turbo-like codes for non-uniform specific interleaver. Upper bounds for specific interleavers was proposed. These bounds make it possible to estimate the performance of optimal detection for specific interleavers, which is better than that of suboptimal iterative detection. The reliability measure of estimated upper bounds was presented.

Chapter 6

Conclusions and Future Work

In this dissertation, complexity reduction techniques and bounding performance were developed. In Chapter 3, RS-A-SISO algorithms were proposed and various design options for RS-A-SISO algorithms were evaluated for SCCPM over frequency-selective fading channels. RS-A-SISO systems also showed large iteration gains. As the number of states used decreases, the performance of the SE-RS-A-SISO algorithms approaches that of the ME-RS-A-SISO algorithms at low Doppler spread. For more rapidly varying channels full-state ME-A-SISO algorithms did not show an error floor but full-state SE-A-SISO algorithms showed an error floor. MS-RS-A-SISO algorithms can be a design option to improve performance. Using density evolution technique, RA-A-SISO algorithms were analyzed. We showed that density evolution technique that is usually used for AWGN systems is also a good analysis tool for RS-A-SISO systems over frequency-selective fading channels. In Chapter 4, we studied upper bounds on the performance of turbo-like codes for specific interleaver. Upper bounds

for specific interleavers was proposed. These bounds make it possible to estimate the performance of optimal detection for specific interleavers, which is better than that of suboptimal iterative detection. The reliability measure of upper bounds was presented.

We found that density evolution technique is also a good analysis tool for RS-A-SISO systems over frequency-selective fading channels. More mathematical explanation about these results, such as Gaussian approximation assumption in AWGN channels, will be a future research topic.

Reference List

[1] A. Anastasopoulos and K. M. Chugg. Adaptive soft-input soft-output algorithms for iterative detection with parametric uncertainty. *IEEE Trans. Commun.*, vol. 48, pp. 1638–1649, October 2000.

[2] J.B. Andersona, T. Aulin, and C.-E. Sundberg. *Digital Phase Modulation*. Plenum Press, New York, 1986.

[3] T. Aulin and C.-E. Sundberg. An easy way to calculate power spectra for digital fm. IEE Proc. Part F,, vol. 130, pp. 519–526, October 1983.

[4] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara. Serial concatenation of interleaved codes: performance analysis, design, and iterative decoding. *IEEE Trans. Inform. Theory*, , no. 3, pp. 909–926, May 1998.

[5] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara. Soft-input soft-output modules for the construction and distributed iterative decoding of code networks. *European Trans. Commun.*, , no. 2, March/April 1998.

[6] C. Berrou and A. Glavieux. Near optimum error correcting coding and decoding: turbo-codes. *IEEE Trans. Commun.*, vol. 44, no. 10, pp. 1261–1271, October 1996.

[7] C. Berrou, A. Glavieux, and P. Thitmajshima. Near shannon limit error-correcting coding and decoding: turbo-codes. In *International Conference on Communications*, pages 1064–1070, Geneva, Switzerland, May 1993.

[8] G. Casella and R. L. Berger. *Statistical Inference*. Duxbury, 2002.

[9] X. Chen and K. M. Chugg. Reduced state soft-in/soft-out for complexity reduction in iterative and non-iterative data detection. In *Proc. International Conf. Communications*, pages 6–10, 2000.

[10] K. M. Chugg. *Sequence Estimation in the Presence of Parametric Uncertainty*. PhD thesis, University of Southern California, Los Angeles, CA, August 1995.

[11] K. M. Chugg. The condition for the applicability of the viterbi algorithm with implications for fading channel mlsd. *IEEE Trans. Commun.*, vol. 46, pp. 1112–1116, September 1998.

[12] K. M. Chugg, A. Anastasopoulos, and X. Chen. *Iterative Detection.* Kluwer Academic Press, 2001.

[13] Kyuhyuk Chung, J. Heo, and K. M. Chugg. Reduced stateadaptive siso algorithms for serially concatenated cpm over frequency-selective fading channels. In *Proc. Globecom Conf.*, pages 1162–1166, San Antonio, Texas, 2001.

[14] S.Y. Chung, T. Richardson, and R. Urbanke. Analysis of sum-product decoding of low-density parity-check codes using a gaussian aaproximation. *IEEE Trans. Inform. Theory*, vol. 47, pp. 657–670, February 2001.

[15] G. Colavolpe, G. Ferrari, and R. Raheli. Reduced state bcjr-type algorithms. In *Proc. International Conf. Communications*, pages 460–464, 2000.

[16] S. Crozier. New high-spread high-distance interleavers for turbo-codes. *20th Biennial Symposium on Communications, Kingston, Ontario, Canada*, pages 3–7, May 2000.

[17] D. Divsalar. A simple tight bound on error probability of block codes with application to turbo codes. *TMO Progress Reports 42-139., Jet Propulsion Lab*, November 1999.

[18] D. Divsalar, S. Dolinar, and F. Pollara. Transfer function bounds on the performance of turbo codes. *TDA Progress Reports 42-122., Jet Propulsion Lab*, August 1995.

[19] D. Divsalar, S. Dolinar, and F. Pollara. Iterative turbo decoder analysis based on density evolution. *jsac*, vol. 19, pp. 891–907, May 2001.

[20] R. Gallager. *Low Density Parity Check Codes.* MIT press, 1963.

[21] H. El Gamal and A. R. Hammons Jr. Analyzing the turbo decoder using the gaussian approximation. *IEEE Trans. Inform. Theory*, vol. 47, pp. 617–686, February 2001.

[22] R. Garello, P. Pierleoni, and S. Benedetto. Computing the free distance of turbo codes and serially concatenated codes with interleavers: Algorithms and applications. *IEEE J. Select. Areas Commun.*, vol. 19, no. 5, pp. 800–812, May 2001.

[23] J. Hagenauer, E. Offer, and L. Papke. Iterative decoding of binary block and convolutional codes. *IEEE Trans. Inform. Theory*, vol. 42, no. 2, pp. 429–445, March 1996.

[24] J. Heo and K. M. Chugg. Analysis of sccc and ldpc code based on density evolution using a gaussian approximation. submitted *IEEE Trans. Commun.*, 2002.

[25] J. Heo, K. M. Chugg, and A. Anastasopoulos. A comparison of forward-only and bi-directional fixed-lag adaptive sisos. In *Proc. International Conf. Communications*, pages 1660–1664, 2000. (Comm. Theory Mini-Conf.).

[26] E. Horowitz and S. Sahni. *Fundamentals of Computer Algorithms.* Computer Science Press, 1978.

[27] A. Leon-Garcia. *Probability and Random Processes for Electrical Engineering.* Addison Wesley, 1994.

[28] P. Moqvist and T. Aulin. Power and bandwidth efficient serially concatenated cpm with iterative decoding. In *Proc. Globecom Conf.*, (San Fransico, CA), 2000.

[29] K. R. Narayanan and G. L. Stüber. Performance of trellis coded cpm with iterative demodulation and decoding. In *Proc. Globecom Conf.*, (Rio de Janeiro, Brazil), 1999.

[30] R. Raheli, A. Polydoros, and C-K. Tzou. Per-survivor processing: A general approach to MLSE in uncertain environments. *IEEE Trans. Commun.*, vol. 43, pp. 354–364, Feb–Apr. 1995.

[31] T. Richardson, A. Shorkrollahi, and R. Urbanke. Design of capacity-approaching irregular low-density parity check codes. *IEEE Trans. Inform. Theory*, vol. 47, pp. 619–637, February 2001.

[32] T. Richardson and R. Urbanke. The capacity of low density parity check codes under message passing decoding. *IEEE Trans. Inform. Theory*, vol. 47, pp. 599–618, February 2001.

[33] B. Rimoldi. A decomposition approach to cpm. *IEEE Trans. Inform. Theory*, vol. 34, pp. 260–270, March 1988.

[34] J. P. Seymour and M. P. Fitz. Near-optimal symbol-by-symbol detection schemes for flat rayleigh fading. *IEEE Trans. Commun.*, vol. 43, pp. 1525–1533, March/April 1995.

[35] R. Shao, S. Lin, and M. Fossorier. Two simple stopping criteria for turbo decoding. *IEEE Trans. Commun.*, vol. 47, pp. 1117–1120, August 1998.

[36] L. Yiin and G. L. Stüber. Mlse and soft-output equalization for trellis-coded continuous phase modulation. *IEEE Trans. Commun.*, vol. 45, pp. 651–659, June 1997.