UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

LOW-DENSITY PARITY-CHECK CODING FOR HIGH-DENSITY MAGNETIC RECORDING SYSTEMS

A Dissertation

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

degree of

Doctor of Philosophy

By

WEIJUN TAN
Norman, Oklahoma
2004

LOW-DENSITY PARITY-CHECK CODING FOR HIGH-DENSITY

MAGNETIC RECORDING SYSTEMS

A Dissertation APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY

[illegible]
J. R. Cruz

[illegible]
John Y. Cheung

[illegible]
Joseph P. Havlicek

[illegible]
Kevin A. Grasse

[illegible]
Tian-You Yu

Acknowledgements

I would like to express my sincere gratitude to my advisor, Dr. J. R. Cruz, for his insightful guidance and encouragement throughout my Ph.D. studies. It is his expertise and wisdom that led me into the area of magnetic recording, helped make this work possible and will benefit me forever.

I would also like to thank Dr. John Y. Cheung, Dr. Joseph P. Havlicek, Dr. Tian- You Yu, and Dr. Kevin A. Grasse for their support and time serving my Ph.D. program committee.

I am thankful to all the fellow students in the Communication Signal Processing Laboratory (CSPLab) for the helpful discussions and the friendship. Particularly, I am appreciative of Dr. Richard M. Todd, a former student of Dr. Cruz, who helped me with the C++ program for density evolution.

Finally, I would like to thank Hitachi, Ltd. for providing a graduate fellowship, which made this work possible.

To my parents and my wonderful wife Qiong

Table of Contents

List of Tables

List of Figures

Abstract

Magnetic recording channels (MRCs), including both longitudinal and perpendicular ones, are subject to a number of physical impairments, such as electronic/media noise, intersymbol interference (ISI), erasure, and intertrack interference (ITI). These impairments, if not appropriately handled, are barriers to achieving ultra-high densities. The goal of this dissertation is to study the impact of these multiple impairments on system performance, and to develop techniques to mitigate this impact such that the performance is as close to the theoretical limit of the channel as can be achieved by practical and implementable means.

Our strategy is to combine advanced signal processing techniques, the core of which is soft-decision iterative channel detection, with powerful low-density parity-check (LDPC) coding techniques.

Specifically, the performance of regular LDPC codes on MRCs is first evaluated. Both randomly and structurally constructed codes are considered. Secondly, density evolution is used to analyze and design LDPC codes for MRCs. Results show that better irregular codes can be obtained. Afterwards, this algorithm is modified to include erasures, and erasure detection algorithms are studied. Fourthly, an improved algorithm for LDPC decoding, called signal-to-noise ratio (SNR) mismatch is unveiled. This algorithm may be useful for future practical applications. Finally, a channel detection algorithm for handling ITI in perpendicular recording is optimized, the eventual goal of which is to maximize the attainable track density.

Chapter 1

Introduction to Magnetic Recording Systems

Magnetic recording data storage has been a key component of the information technology revolution that has swept the globe in the last decade of the 20th century. The internet and the world-wide-web would not be of any relevance today if it were not for the availability of massive amounts of storage capacity at incredibly low prices. These storage systems are built around hard disk drive units whose average size in the 1990's increased 100 times while the prices decreased dramatically.

This was made possible by technological advances in magnetic recording materials and devices used in longitudinal recording, These include the development of high-performance gigantic magneto-resistive (GMR) read heads, narrow pole-tip write heads, improved preamplifiers, low-noise thin film media, low flying height designs, and advances in the preparation of head and disk surfaces. Simultaneously, to maintain the system performance at increasingly higher areal densities it became also necessary to introduce advanced signal processing and coding techniques, such as more sophisticated equalization techniques using partial response (PR) targets and maximum likelihood (ML) detection with noise prediction, single parity-check codes, as well as more powerful error-correcting codes (ECCs).

1.1 Magnetic Recording

1.1.1 Digital Magnetic Recording Basics

The basic mechanism of magnetic recording is the interaction between a magnetic medium and a magnetic head in relative motion with respect to one another. The data is recorded on the medium in the form of a remnant magnetic field, from which the information is retrieved. In a digital magnetic recording systems, the information is discrete, i.e., either a -1 or a 1, representing the two patterns of the magnetization in saturation recording.

The two most important processes in magnetic recording are the write process and the read (or replay) process. During the write process, a signal current "modulated" by the recording data is applied to the write head. This current causes a flux pattern that follows the head path and magnetizes the medium. During the read process, the inductive or magneto-resistive read head senses the flux change from the magnetized medium. These two basic processes, more details of which can be found in [1], are illustrated in Fig. 1.1 for the two modes of magnetic recording, namely, the longitudinal recording and the perpendicular recoding.

1.1.2 Modes of Recording

Three modes of recording are defined based on the direction of the surface magnetization relative to the direction of the medium motion. These modes are longitudinal recording, perpendicular recording and transverse recording, the first two of which are considered in this dissertation and are shown in Fig. 1.1.

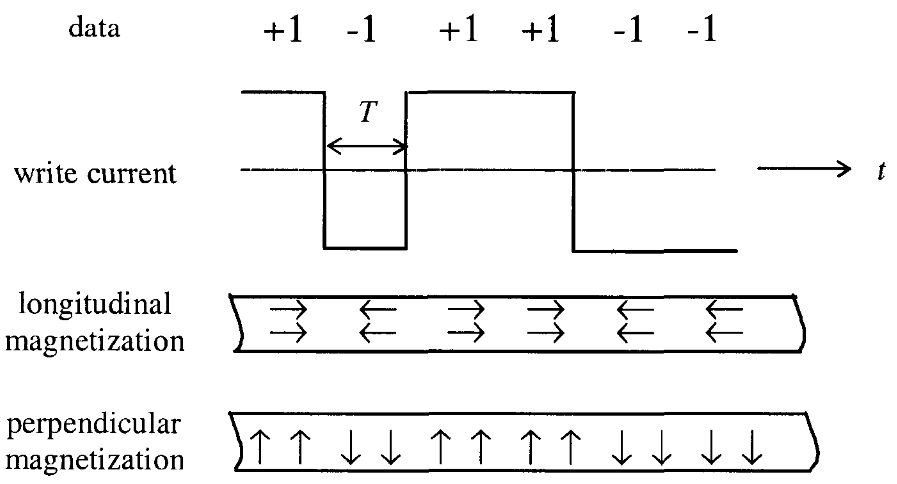


Fig 1.1. Illustration of the write and read processes for magnetic recording [2].

Longitudinal recording has been the standard mode of recording for the past fifty years due to its inherent advantages to achieve higher resolution offered by the architecture of a ring head with a narrow gap. However, the continuing demand for more storage capacity in recent years leads to areal densities approaching the superparamagnetic limit. This motivates more and more attention being paid to perpendicular recording [3].

Perpendicular recording has long been advocated as a means of achieving the highest areal densities. In particular, in the context of the 'superparamagnetic limit', perpendicular recording with a soft underlayer promises several key advantages [4]. These advantages include a higher coercivity, thicker media that should permit smaller diameter grains and higher signal-to-noise ratio (SNR). Also the sharper edge-writing will facilitate recording at very high track densities and a lower bit aspect ratio (BAR). It has been shown that perpendicular recording will deliver an increase in areal density between two to eight times higher than that achievable with longitudinal recording [4].

1.2 Continuous-Time Channel Model

The continuous-time magnetic recording channel (MRC) model, including channel response and physical impairments, is given in this section. Both longitudinal MRC (LMRC) and perpendicular MRC (PMRC) are considered. In many cases, LMRC is simply referred to as MRC if no ambiguity arises.

1.2.1 Channel Response

The read process can be characterized by the transition response (step response) $s(t)$, which is the response of the read head to an isolated positive transition. The width of half amplitude of $s(t)$ defines the resolution of the recording process, usually denoted by PW_{50} when measured spatially.

For a given mode of recording, $s(t)$ can be derived from electromagnetic theories; however, it can be more conveniently estimated from the read signal of a known pattern and modeled as a simple function.

For longitudinal recording, $s(t)$ is usually assumed to be a Lorentzian function,

$$s(t) = \frac{V_0}{1+(2t/PW_{50})^2} \tag{1.1}$$

or a Lorentzian-Gaussian function,

$$s(t) = \frac{1}{2}\left[\frac{V_0}{1+(2t/PW_{50})^2} + V_0\exp(-\frac{4\ln 2}{PW_{50}}t^2)\right], \tag{1.2}$$

where V_0 is the amplitude of the channel response. However, the Lorentzian function is more frequently used. The normalized channel density (or simply channel density) is defined as $S_c = PW_{50} / T$.

For perpendicular recording, the read back signal is dramatically different from that of longitudinal recording. Commonly used models for the transition response are the hyperbolic-tangent function [5],

$$s(t) = V_0 \tanh(\frac{\ln 3}{PW50} t) \tag{1.3}$$

and the arctangnent function [6],

$$s(t) = V_0 \arctan(\frac{2t}{PW50}). \tag{1.4}$$

In this dissertation the hyperbolic-tangent function (1.3) is used. Similarly the normalized channel density is defined as $S_c = PW_{50} / T$.

Shown in Fig. 1.2 are the transition functions, namely, the Lorentzian function for an LMRC with $S_c = 2.5$ and 3.0, and the hyperbolic-tangent function for a PMRC with $S_c = 1.5$ and 2.0. From Fig. 1.2 we see that these two transition functions are remarkably different. The one for PMRC is more like the integral of the one for LMRC.

Another frequently used channel response is the dibit response, also known as the pulse response,

$$h(t) = s(t) - s(t - T), \tag{1.5}$$

which is the response of a positive transition followed by an immediate negative transition.

Shown in Fig. 1.3 are the amplitude response $| H(e^{jwT}) |$ = |Fourier Transform of $h(t)$| for LMRC with $S_c = 2.5$ and 3.0, and for PMRC with $S_c = 1.5$ and 2.0. From Fig. 1.3

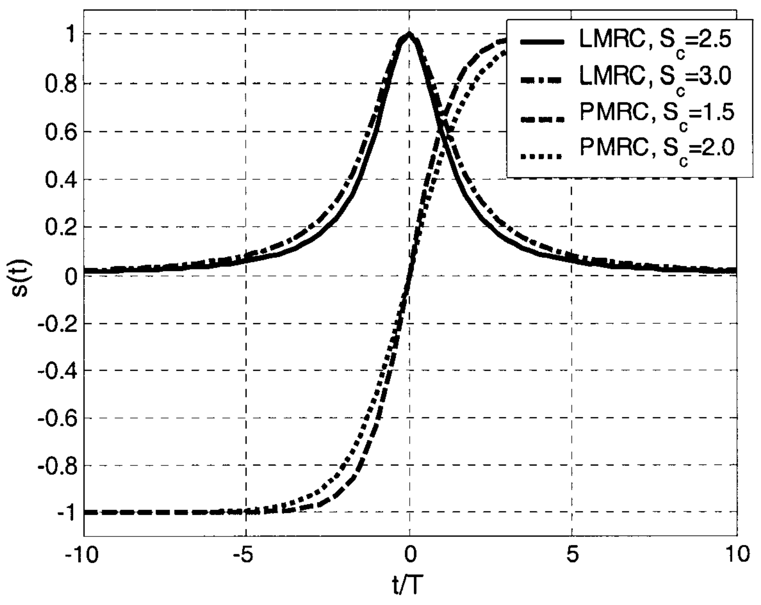


Fig. 1.2 Lorentzian function for LMRC and hyperbolic-tangent function for PMRC.

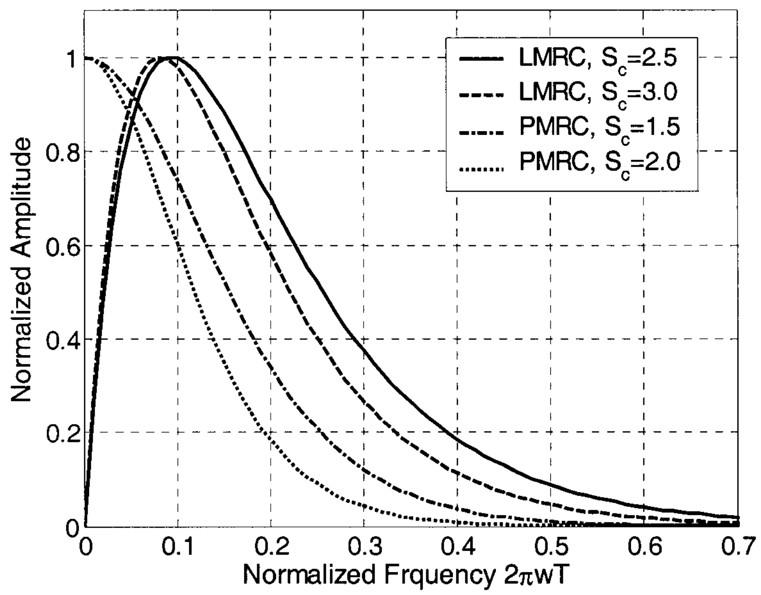


Fig. 1.3. Amplitude response $|H(e^{jwT})|$ for LMRC and PMRC.

it can be seen that these two channels have quite different frequency responses, particularly in the low-frequency range. Specifically, the LMRC has a null at DC while the PMRC has a peak at DC.

Let the recorded data sequence be $x_k \in \{+1,-1\}$, the read back signal can be expressed as,

$$r(t) = \sum_{k=-\infty}^{\infty}(x_k - x_{k-1}) \cdot s(t-kT) + n(t) \tag{1.6}$$

or in terms of $h(t)$,

$$r(t) = \sum_{k=-\infty}^{\infty} x_k \cdot h(t-kT) + n(t). \tag{1.7}$$

where $n(t)$ is the noise. If this noise is additive white Gaussian noise (AWGN), its power spectral density is $\sigma^2 = N_0 / 2$.

1.2.2 Physical Disturbances

There are many sources of physical disturbances in the write and read processes. Those in the read process are the main focus of this dissertation.

1. Noise

Noise originates in the medium, the head and the head preamplifier. The main two forms of noise are electronic noise and media noise.

Electronic noise is the thermal noise present in any communication system. It is usually assumed data independent and AWGN with normal distribution $\mathcal{N}(0,\sigma^2)$.

Media noise is caused by the granularity of the magnetic material in the medium, and can be divided into transition noise and particulate noise. The transition noise includes both the position jitter noise, and pulse-width variation noise. Media noise is generally neither additive nor stationary. In this dissertation, media noise is simplified as a first-order transition jitter noise $\Delta t \cdot \frac{ds(t)}{dt}$, where Δt is a random variable, which is usually assumed to have a normal distribution $\mathcal{N}(0, \sigma_J^2)$.

For an MRC with both AWGN $n_A(t)$ and position jitter noise $n_J(t)$, the noise in (1.7) is $n(t) = n_A(t) + n_J(t)$. The SNR is defined as

$$SNR = \frac{V_0^2}{2\left(\sigma^2 + \sigma_J^2 \int_{-\infty}^{\infty} \left|\frac{d}{dt} s(t)\right|^2 dt\right)}, \tag{1.8}$$

and the percentage of AWGN is $\sigma^2 \Big/ \left(\sigma^2 + \sigma_J^2 \int_{-\infty}^{\infty} \left|\frac{d}{dt} s(t)\right|^2 dt\right)$.

2. Erasure

Erasures are caused by a disturbance of the magnetization due to a variety of mechanisms. Two common causes are thermal asperities (TAs) and media defects (MDs), where the signals are subject to a DC-value saturation and an AC-value fading (called dropout), respectively. An additional mechanism in perpendicular recording is the strongly concentrated stray ambient fields in the gap between the soft underlayer and the read/write head. This field can possibly erase the signals on the adjacent tracks, which becomes more severe as the track pitch gets smaller. This erasure process is referred to as adjacent track erasure (ATE) and is one of the most challenging problems to overcome in high-density perpendicular recording systems. In all cases, no useful information is available at the channel output, and this is why these impairments are referred to as erasures.

3. Adjacent Track Interference (ATI)

Usually there is a guard band between two adjacent tracks. The reason for this is to avoid the read head from picking up some flux outside its nominal track width. However, some other sources may also result in ATI. For example, misregistration of the head, i.e., the deviation of the head from its perfect position due to servo error, produces some offtrack interference (OTI). In addition, in high-density magnetic recording, even when the head is perfectly positioned, intertrack interference (ITI) can occur. The ITI is due to the narrower track width and pitch, featured in perpendicular recording. The ITI can be caused by both side-writing and side-reading. The side-writing ITI arises when stray fields from the sides of the write head create interference on an adjacent track (it can also cause erasures, as discussed above). This stray field is asymmetric, much stronger on one side than on the other, due to the radially oriented magnetization of the soft-underlayer [7]. The side-reading ITI arises when the head reads not only the signal on the desired track, but also signals from adjacent tracks. The handling of ITI is a critical issue in this dissertation.

4. Nonlinear Distortions

Many other nonlinear distortions may be present in magnetic recording channels. However, they are not considered in this dissertation. For details please refer to [1].

1.3 Discrete-Time Channel Model

The discrete-time channel model is given in this section. First, an optimum channel detection model is discussed. Then a suboptimum channel detection based on PR equalization is presented. Finally, the selection of PR targets is addressed.

1.3.1 Optimum Channel Detection

The MRC can be viewed as a noisy communication channel with binary inputs. This channel is completely characterized by the channel pulse response $h(t)$. It can be easily seen from this $h(t)$ that the MRC is an intersymbol interference (ISI) channel. Therefore, when the noise $n(t)$ in (1.7) is AWGN, the optimal detector for this channel is a matched filter followed by a symbol rate sampler, a noise whitening filter, and a maximum likelihood sequence detector (MLSD). Let $r_{h,k} = [h(t) * h(-t)]_{t=kT}$ and $\eta_k = [n(t) * h(-t)]_{t=kT}$, then the signal s_k at the output of the sampler is

$$s_k = (x * r_h)_k + \eta_k . \qquad (1.9)$$

The noise whitening filter has transfer function $1/F(z^{-1})$ (in the z domain), where $R_h(z) = Z\{r_{h,k}\} = F(z)F(z^{-1})$. And the all-discrete channel model reads,

$$r_k = (x * f)_k + v_k , \qquad (1.10)$$

where v_k is η_k filtered by $1/F(z^{-1})$, which is AWGN . The memory length L in (1.10) is usually large.

The basic implementation of the MLSD is the Viterbi Algorithm (VA) or its soft version, the soft VA (SOVA) [8]. Another symbol-based channel detector is the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [9]. This algorithm aims at minimizing the symbol error rate, but has much higher complexity. For this reason it has been overlooked for several decades, until the emergence of turbo codes [10] has brought it back to people's attention. Both the VA and the BCJR algorithm need 2^L states, where L is the length of the ISI. Since the ISI is usually long at high density, the complexity is very high. Therefore, this ISI needs to be equalized to a shorter PR target.

1.3.2 Suboptimum Channel Detection

In practice, suboptimal detection is often used to trade-off complexity for performance. Firstly, the matched filter $h(-t)$ is replaced by a realizable low-pass filter $p(t)$. Then the sampled output of the filtered pulse response is equalized to some predetermined short-length PR target with a finite-impulse-response filter $\{w_i\}$, and finally a VA or BCJR detector is used for detection. The combination of PR equalization and MLSD is called PRML detection [1]. The PR equalized MRC model and the equivalent all discrete-time models are shown in Fig. 1.4, where $\{f_i\}$ is the PR target of a predetermined shorter length, and the equalizer $\{w_i\}$ is determined by the PR equalization. Then the all discrete-time channel model is expressed as in (1.10), except

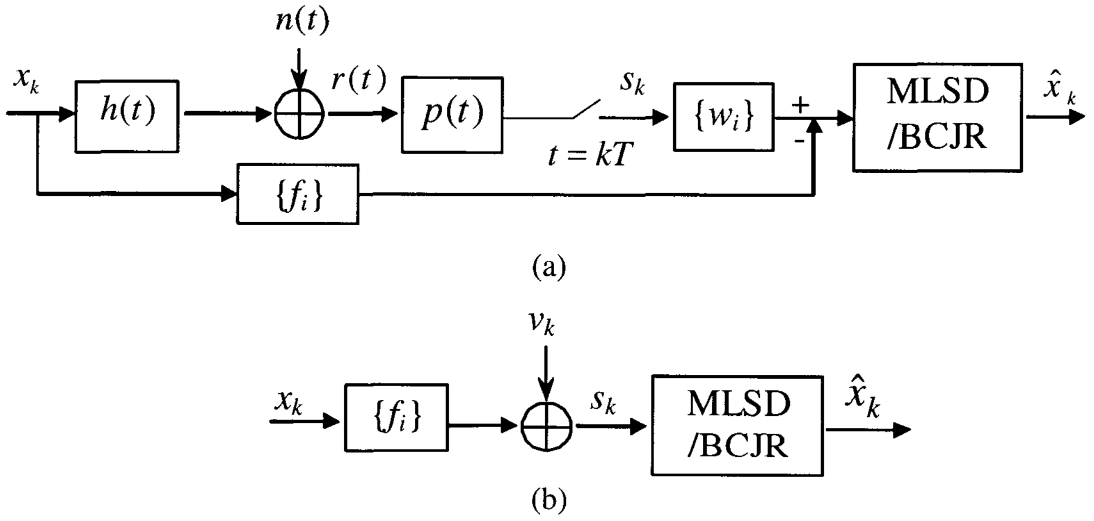


Fig.1.4. PR channel models. (a) PR equalized MRC model, (b) all discrete-time model.

that the memory length L is a predetermined small value, and that the noise v_n is no longer AWGN. This model is used throughout this dissertation, unless specified otherwise.

If the equalization can be perfectly achieved and the noise v_k is AWGN, we obtain another model – the ideal PR model. This model is also used occasionally in this dissertation. For clarity, the channel using an ideal PR model is called an ideal PR channel, and an MRC using an equalized PR model is called an MRC or simply a PR channel. For an ideal PR channel, the definition of SNR is,

$$SNR = \frac{\sum_{i=0}^{L} f_i^2}{2\sigma^2}, \tag{1.11}$$

where σ^2 is the variance of v_k.

1.3.3 Partial Response Equalization

The equalizer $\{w_k\}$ can be derived from the PR equalization. Let $r_{hp,k} = [h(t) * p(t)]_{t=kT}$ and $\zeta_k = [n(t) * p(t)]_{t=kT}$, the signal s_k at the output of the sampler is

$$s_k = (x * r_{hp})_k + \zeta_k. \tag{1.12}$$

The equalization error between the actual path and the targeted PR path, shown in the upper half and lower half in Fig. 1.4(a), is

$$e_k = (s * w)_k - (x * f)_k. \tag{1.13}$$

Different equalization criteria can be used upon e_k which results in a different equalizer

$\{w_k\}$. A simple example is the zero-force equalization; however, it is rarely used because of noise enhancement. The one widely used is the minimum mean square error (MMSE) equalization, which minimizes the expectation of the equalization error, i.e, $\frac{\partial E\{|e_k|^2\}}{\partial w_k} = 0$. After some manipulation, we have

$$\mathbf{R_s} \cdot \mathbf{w} = \mathbf{R_{s,x}} \cdot \mathbf{f}, \tag{1.14}$$

where $\mathbf{R_s}$ is the auto-correlation matrix of $\{s_k\}$, $\mathbf{R_{s,x}}$ is the cross-correlation matrix of $\{s_k\}$ and $\{x_k\}$.

1.3.4 Partial Response Targets

The issue of how to select good PR targets is addressed in this section. Both LMRC and PMRC are considered.

1. Longitudinal Recording

The popular PR targets for LMRCs are class-IV (PR4), extended PR4 (EPR4), extended EPR4(E^2PR4), and modified E^2PR4 (ME^2PR4), as listed in Table 1.1. Let us look at the amplitude responses of these targets, shown in Fig. 1.5, as well as the pulse responses for $S_c = 2.5$ and 3.0 for comparison. For LMRC at high densities, it is obvious that longer PR targets, e.g., E^2PR4 and ME^2PR4, are better than shorter ones, e.g., PR4, from the perspective of spectral match. However, as the target gets longer, the complexity of channel detector increases as well. Therefore, a good tradeoff must be made. Throughout this dissertation, two targets, namely, EPR4 and ME^2PR4, are used widely.

Table 1.1. PR targets for LMRCs.

Name	Target
PR4	[1 0 -1]
EPR4	[1 1 -1 -1]
E^2PR4	[1 2 0 -2 -1]
M E^2PR4	[2 2 -1 -2 -1], [5 4 -3 -4 -2]

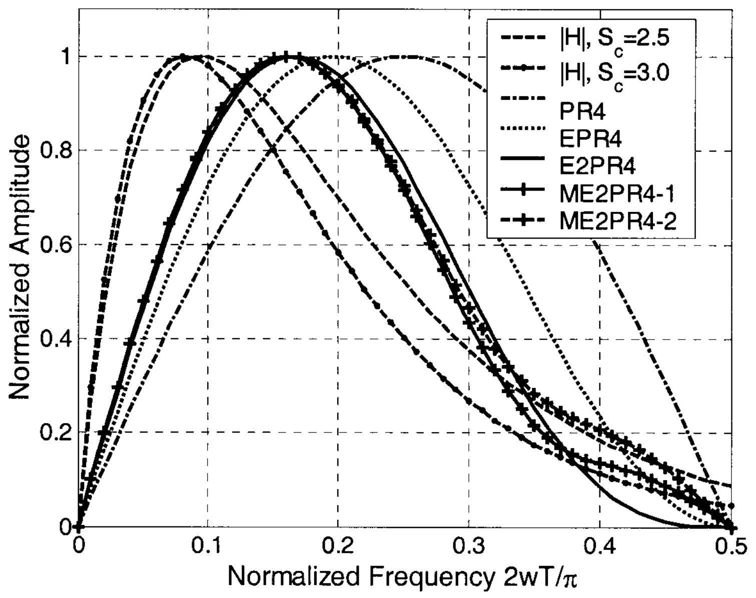


Fig. 1.5. Amplitude response of PR targets for LMRCs.

At high recording densities, generalized PR (GPR) targets with non-integer coefficients match the longitudinal recording channel more accurately than the conventional PR targets, leading to less noise enhancement at the equalizer output [11]. In particular, $F(D)=(1-D)p(D)$ or $F(D)=(1-D)^2 p(D)$, where

$p(D) = 1 + p_1 D + p_2 D^2 + \ldots + p_L D^L$ is a noise whitening filter, render the total noise at the input of the detector approximately white, provided the equalizer and the noise whitening filter are sufficiently long. This class of polynomials is significant in practice, and when combined with sequence detection, gives rise to noise-predictive maximum likelihood (NPML) systems [12].

2. Perpendicular Recording

The frequency response of the PMRC is quite different from that of the LMRC. Explicitly, the LMRC has a null at DC; however, the PMRC has a lot of DC content. It is this feature that results in different PR targets for PMRC.

PR targets for PMRCs are usually GPR ones with non-integer coefficients. Generally speaking, two criteria need to be considered in selecting a GPR target, namely, the spectral match of the target and the channel, and the handling of low-frequency disturbances including media noise and TAs. For spectral match, DC-full targets seem good choices. However, these targets allow low frequency disturbances to enter the read head, which can cause base-line wander and waveform distortion. On the other hand, if DC-free targets are selected, then low-frequency disturbances can be stopped, however, some performance penalty will occur due to spectral mismatch. Therefore, a tradeoff between DC-full and DC-free targets, referred to as DC-mix targets, is desirable.

There has been a great deal of attention paid to optimizing the selection of the PR target for perpendicular recording (e.g., [5], [6]). Let us follow the method in [5] and examine the PR target selection by analyzing the noise spectrum in the channel model of Fig. 1.2. First consider only white noise with variance σ^2. Then the power spectrum of the equalized noise is

$$R_n(D) = \frac{\sigma^2}{2} \frac{F(D)F(D^{-1})}{R_h(D)}. \tag{1.15}$$

The coefficients $\{f_k\}$ are obtained from

$$\frac{\partial}{\partial f_k}\left[\frac{\sigma^2}{2}\int_{-0.5}^{0.5} \frac{F(D)F(D^{-1})}{R_h(D)}\Big|_{D=e^{-j2\pi\Omega}} d\Omega\right] = 0. \tag{1.16}$$

The GPR targets obtained with this method have a lot of low-frequency content and are DC-full.

For media noise, the PR target is selected in the same way as for AWGN, while satisfying the DC-null constraint. The obtained targets are DC-free.

Furthermore, it has been further suggested in [5] to use a set of GPR3 targets,

$$F(D) = (1 - \alpha D)p(D), \tag{1.17}$$

where $p(D)$ is the noise whitening filter and α is to be optimized depending on media noise. This type of targets are DC-mix targets for $\alpha \neq 0, 1$.

Using these techniques, a number of GPR targets have been computed in [5]. As an example, targets for PMRC with $S_c = 1.4$ are listed in Table 1.2. The lengths of these targets are $L = 4$. Note that these three targets are DC-full, DC-free, and DC-mix, respectively.

The amplitude responses of the targets in Table 1.2, as well as those of the PMRCs with $S_c = 1.5$ and 2.0, are shown in Fig. 1.6. Strictly from a spectral match viewpoint, we see that the DC-full target is the best choice, and the DC-free one is the worst. However, when the media noise is accounted for, the DC-mix target may be better. More

Table 1.2. GPR targets for PMRC of $S_c = 1.4$.

DC Content	GPR Targets
DC-full	[1.0 1.72 1.14 0.33]
DC-free	[1.0 0.63 -0.87 -0.76]
DC-mix	[1.0 1.11 -0.07 -0.37]

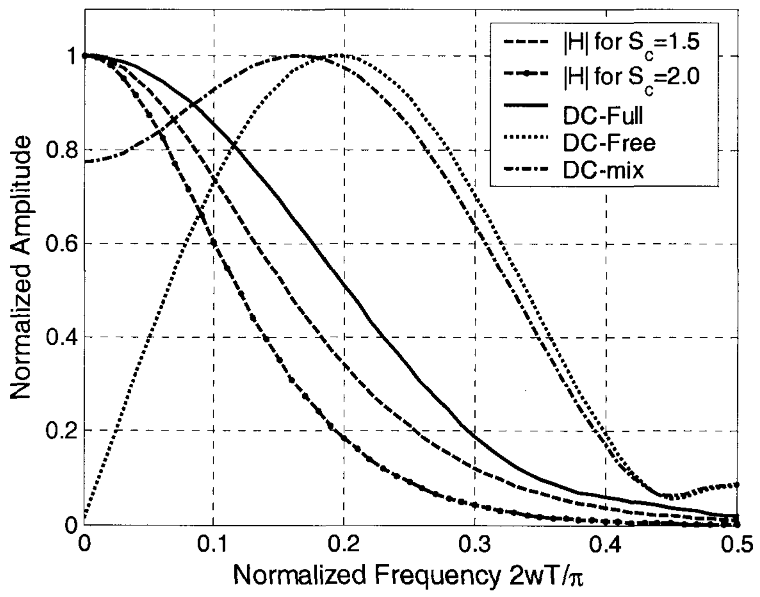


Fig.1.6. Amplitude response of GPR targets for PMRCs.

investigation on GPR target selection will be discussed in later chapters.

Another way to optimize the PR target is by using MMSE coupled with the Langrange method, as in [13]. Specifically, the PR target is computed as,

$$\mathbf{f} = \lambda\left(\mathbf{R}_{x,x} - \mathbf{R}_{s,x}^{T}\mathbf{R}_{s,s}^{-1}\mathbf{R}_{s,x}\right)^{-1}\cdot\mathbf{I}, \tag{1.18}$$

where $\mathbf{I} = [1,0,\cdots 0]^T$, $\mathbf{R}_{\mathbf{x}}$ is the autocorrelation of **x**, and

$$\lambda = \frac{1}{\mathbf{I}^T\left(\mathbf{R}_{\mathbf{x},\mathbf{x}} - \mathbf{R}_{\mathbf{s},\mathbf{x}}^T \mathbf{R}_{\mathbf{s},\mathbf{s}}^{-1} \mathbf{R}_{\mathbf{s},\mathbf{x}}\right)^{-1}\mathbf{I}}. \tag{1.19}$$

The equalizer **w** follows (1.14) with **f** determined by (1.18). This method has been shown effective for perpendicular recording with media noise [6].

1.4 Magnetic Recording Systems

A simple magnetic recording system model from the perspective of channel detection and coding is shown in Fig. 1.7. In addition to the components shown, a timing recovery and a gain control systems are necessary. However, perfect timing and gain control is assumed unless specified explicitly.

In this model, the MRC and the channel detector have already been explained. Let us explain briefly the functions of other components.

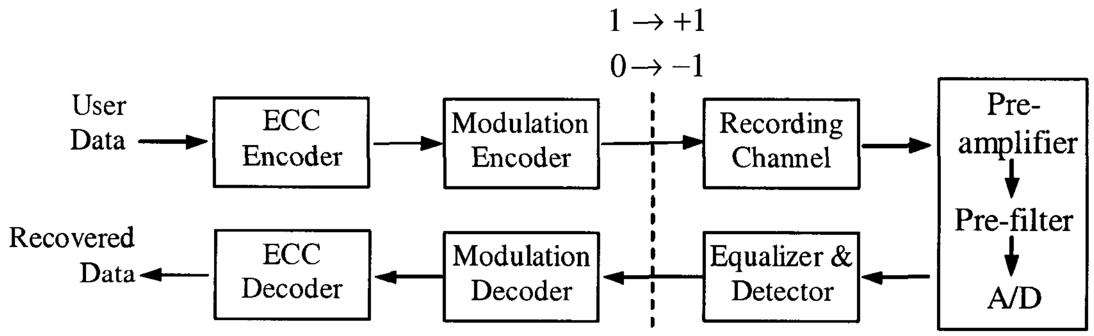


Fig. 1.7. Magnetic recording system model.

1.4.1 Error Correcting Code

ECC is one of the most effective approaches to combat physical impairments in a communication channel. We only consider linear codes in this dissertation. A binary code is usually denoted by (N, K) where N and K are the lengths in bit of the overall and user information. This code has $N-K$ redundant bits and the rate is $R = K / N$. Another important parameter of a linear code is the minimum distance $d_{\min}$, which is the minimum weight of the non-zero codewords.

Using an ECC of rate R, the normalized user density of a magnetic recording channel can be defined as $S_u = R \cdot S_c$, and therefore the introduction of coding reduces the density that can be used by user information. This is one reason why magnetic recording systems require a high-rate ECC. The second reason is, unlike transmission channels where the coding loss is linear with R, for a MRC it is proportional to R^2. The quadratic code rate loss implies that at code rate lower than some threshold the gain from lowering the code rate will not compensate for the code rate loss.

While Reed-Solomon (RS) codes are the *de facto* standard for current magnetic recording system, we are primarily interested in an alternative solution using low-density parity-check (LDPC) codes, which allow more advanced channel detection and decoding techniques such as turbo equalization and soft-decision iterative decoding.

1.4.2 Modulation Code

A modulation code is necessary in a magnetic recording system to facilitate gain control and timing recovery, although it may also improve the free Euclidean distance of the channel. As the read head only responds to the transition, the modulation code must have a constraint on the maximum run of transitions and/or non-transitions.

Two classes of modulation codes often used are run-length-limited (RLL) codes and maximum-transition-run (MTR) codes [1]. An RLL code generally has two constraints, d and k, where d is the minimum run length of non-transitions between two transitions and k is the maximal run length of non-transitions. These two parameters restrict the high and low frequency component of the channel signal, respectively. Another enhanced RLL code has one more parameter, G, which is the interleaved constraint to restrict the two-way interleaved maximum run length of non-transitions. This RLL code is denoted as a $(d,\ G|I)$ RLL code. For MTR codes, in addition to the minimum run length of non-transitions, the maximum run length of transitions is also constrained.

1.4.3 Precoder

The precoder is not shown in Fig. 1.7. But it is sometime a very important component in a magnetic recording system.

The basic function of a precoder is to convert the non-return-to-zero (NRZI) input data into return-to-zero (NRZ) format. Unless the code design is done in NRZ space, the precoder is necessary in the system.

However, the precoder may have great impact on the system performance. For example, in a serial concatenation of a convolutional code and a PR channel, the precoder can make the inner code recursive, thus providing possible interleaving gain. Another example is that the precoder affects the convergence of turbo equalization of a turbo coded PR channel [14]. However, we are mainly concerned with its impact on an LDPC coded magnetic recording system.

1.5 Outline of the Dissertation

This dissertation focuses on channel detection, which is the key signal processing issue, and LDPC coding for magnetic recording systems. Both longitudinal and perpendicular recording systems are considered. While Chapters 2 to 6 are primarily concerned with longitudinal recording, Chapter 7 is completely devoted to perpendicular recording. Other than generic techniques in Chapters 2 to 6, new techniques for handling ITI are stressed in Chapter 7.

Chapter 2 describes generic channel detection algorithms for PR channels. These include PRML, SOVA, BCJR, and forward-MAP – a continuously decodable variation of BCJR. The complexity and performance of these algorithms are evaluated and compared for LDPC coded PR channels.

LDPC codes and the application to PR channels are discussed in Chapter 3. First, the factor graph representation of an LDPC code is described. Then the message-passing (MP) algorithm for decoding LDPC code is presented. Finally, several types of LDPC codes, namely, random codes, finite geometry (FG) codes and array codes, are discussed.

Chapter 4 uses the density evolution algorithm to evaluate and design LDPC codes for PR channels. First a technique to compute the symmetric information rate of PR channels is reviewed. Then the framework for density evolution is established. This framework consists of the algorithm, its properties, and the estimation of the noise threshold and the bit error rate (BER).

Chapter 5 deals with erasures, one important and critical type of impairment in MRCs. After a model for erasure is developed and the BCJR-once information rate is computed, the density evolution algorithm in Chapter 4 is extended to PR channels with erasures.

This algorithm is used to evaluate the asymptotic performance of LDPC coded systems. However, asymptotic analysis is not sufficient. Erasure detection algorithms for practical systems are also considered. These algorithms explore amplitude in addition to sign of the BCJR log-likelihood ratios (LLRs), thus improving the erasure detection performance. LDPC coded systems with this erasure detector are evaluated.

Chapter 6 discusses the SNR mismatch for LDPC coded MRCs. This technique is expected to have potential practical applications because it can improve the system performance substantially. Simply speaking, the system performance can be enhanced by adjusting the SNR. This phenomenon is related to the spectrum of the noise, and can be analyzed by the density evolution algorithm. For a given practical system, the optimum value of SNR mismatch can be found by density evolution as well as by simulation.

Chapter 7 focuses on channel detection algorithms for PMRC with ITI. First of all, an ITI model is established, and two interpretations of it, namely, the single-track model and the joint-track model are described. Based on these two models, both single-track and joint-track channel detection algorithms are investigated, using information rate analysis as well as simulations.

Chapter 8 concludes the dissertation and lists the work for future study.

Chapter 2

Partial Response Channel Detection

In Chapter 1, we introduced a discrete-time PR model for an MRC, which is rewritten here for convenience,

$$r_k = \sum_{i=0}^{L} f_i x_{k-i} + n_k . \tag{2.1}$$

Generally speaking, the noise n_k is not AWGN; however, it is treated as AWGN for channel detection, or can be whitened somehow before channel detection.

The goal of channel detection is to detect the transmitted data sequence $\mathbf{x}$ given the received channel sequence $\mathbf{r}$. Basically, the MLSD finds a sequence $\hat{\mathbf{x}}$

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \Pr(\mathbf{x} \mid \mathbf{r}) = \arg\max_{\mathbf{x}} \Pr(\mathbf{r} \mid \mathbf{x}) \cdot \Pr(\mathbf{x}) / \Pr(\mathbf{r}) = \arg\max_{\mathbf{x}} \Pr(\mathbf{r} \mid \mathbf{x}) \tag{2.2}$$

since $\Pr(\mathbf{x})$ is equal for arbitrary $\mathbf{x}$. Symbol-based maximum *a posteriori* (MAP) detection finds x_k according to the LLR of the *a posteriori* probability (APP) $\Pr(x_k = 1 \mid \mathbf{r})$ and $\Pr(x_k = 0 \mid \mathbf{r})$,

$$\Lambda_k = \log \frac{\Pr(x_k = 1 \mid \mathbf{r})}{\Pr(x_k = 0 \mid \mathbf{r})} . \tag{2.3}$$

Both decisions in (2.2) and (2.3) can be hard, i.e., $x_k \in \{0,1\}$, or soft, i.e., the reliability of the decision. Channel detection is the key issue in channel signal processing, and is a central issue in this dissertation. Since the MRC is equalized to a PR target, the channel detection for MRC is called PR channel detection.

In this chapter, PR channel detection algorithms are presented. These include PRML, SOVA, BCJR, and forward-MAP. The first two are MLSD algorithms, while the other two are symbol-based algorithms.

2.1 Trellis Description of PR Channels

A PR channel can be viewed as a stationary finite-state machine, which can be described using a trellis.

Let the content in the memory of a PR channel be $s_{k-1} = (x_{k-L}, x_{k-L+1}, \cdots, x_{k-1})$. This s_{k-1} defines the state of the channel at time k-1. At time k with input x_k, the channel noiseless output z_k is determined by (2.1) and the new state is $s_k = (x_{k-L+1}, x_{k-L+2}, \cdots, x_k)$. This transition from state s_{k-1} at time k-1 to state s_k at time k can be denoted by $s_{k-1} \xrightarrow{x_k / z_k} s_k$. Plotting all such transitions together constitutes one section of the trellis for this particular k. Note that since the PR channel considered for the MRC is stationary, every section of the trellis is exactly the same. If a precoder is used before the PR channel, a combined trellis for the precoded PR channel can also be derived. As an example, the trellises for the EPR4 channel are shown in Fig. 2.1, where solid and dashed lines denote branches with $x_k = 0$ and 1, respectively. The eight states of these trellises are $S_0, S_1, \cdots, S_7$, corresponding to the content of the memory $(0,0,0), (0,0,1), \cdots, (1,1,1)$, respectively.

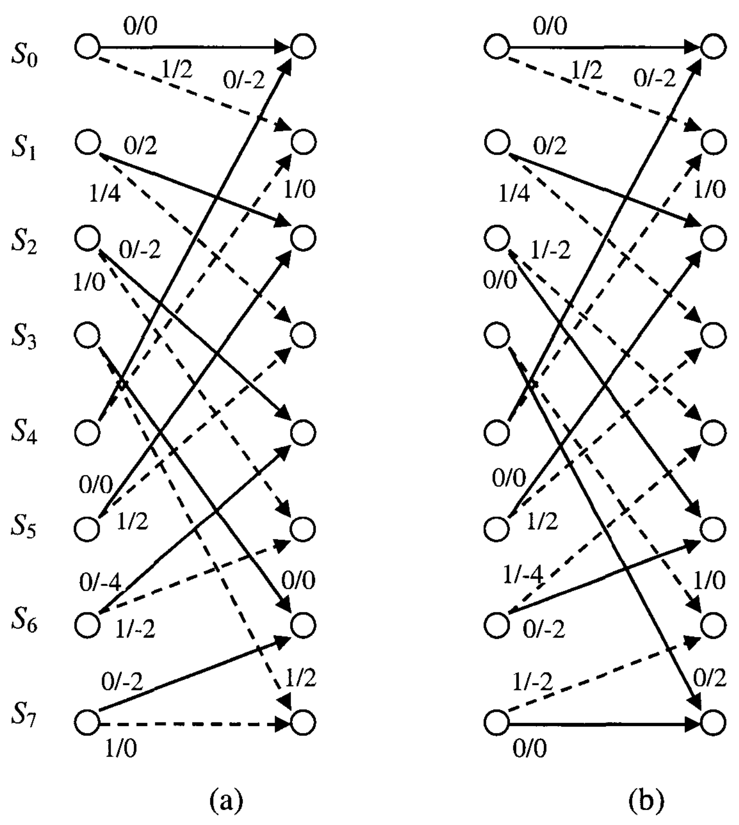


Fig. 2.1. Trellises for the (a) non-precoded and (b) $1/1 \oplus D^2$-precoded EPR4 channel.

2.2 PRML

PRML detection [1] is the state-of-the-art detection algorithm for current magnetic recording systems. PRML gained its popularity due to its good performance as well as low complexity and easy hardware implementation.

PRML detection is essentially a VA on a PR channel. The VA was first introduced for decoding convolutional codes and is one of the most successful algorithms in both theory and practice. Since then this algorithm has been a standard tool in communication receivers, fulfilling functions such as decoding, demodulation and equalization.

The VA selects the ML path through the trellis of a code or a channel. For this reason it is usually used interchangeably with MLSD. The detected data sequence $\hat{\mathbf{x}}$ of the ML path has the minimum distance from the actual transmitted data sequence $\mathbf{x}$. This distance can be a Hamming distance, or a Euclidean distance. For a PR channel, it is the Euclidean distance, and the VA finds

$$\hat{\mathbf{x}} = \min_{\mathbf{x}} \sum_{k=1}^{N} \left[r_k - (x_k + \sum_{i=1}^{L} f_i x_{k-i}) \right]^2 . \tag{2.4}$$

The task in (2.4) can be implemented as follows. Define the metric for the branch $s_{k-1} \xrightarrow{x_k / r_k} s_k$ as

$$\gamma_k(s_{k-1}, s_k) = -\left[r_k - (x_k + \sum_{i=1}^{L} f_i x_{k-i}) \right]^2 / N_0 . \tag{2.5}$$

Note that (2.5) is essentially the same as (2.4) except for a different sign and a normalization factor N_0, which can be omitted in practical implementation. For each state s_k at time k, consider all possible (typically two) states s_{k-1} at time k-1 entering state s_k at time k. A path up to time k with the maximum path metric is selected, and all others are discarded. This selected path is called the survivor path. The metric for this path with state s_k at time k is

$$M_k(s_k) = \max_{s_{k-1}} \{ M_{k-1}(s_{k-1}) + \gamma_k(s_{k-1}, s_k) \}, \tag{2.6}$$

where $M_0(S)$ is initialized as $M_0(s_0 = S_0) = 1, M_0(s_0 \neq S_0) = -\infty$.

This process can keep going until the final state at time N is reached. Since a convolutional code or a PR channel is usually terminated to state S_0 at time N, the VA can simply select the ML path and make a decision on $\hat{\mathbf{x}}$ by tracing back from state S_0 at time *N*. However, since all the 2^L survivor paths at time $k+D$ have merged at time k with high probability, decision $\hat{x}_k$ can be made with delay *D*, an empirical value for which is $D \geq 5L$.

2.3 SOVA

It can be easily seen that the factor N_0 is irrelevant and can be dropped in (2.4) if hard decisions are made. This is one advantage of PRML. However, if soft decisions are used, this factor must be known and kept in (2.4). Furthermore, the *a priori* information about x_k is not considered in PRML. We now modify PRML to a soft-decision version. This algorithm is called SOVA [8]. It is not called soft PRML, because a maximum APP (MAP) criterion is used. The SOVA was used for PR channel detection in [15].

The branch metric in SOVA is defined as,

$$\gamma_k(s_{k-1}, s_k) = -\left[r_k - (x_k + \sum_{i=1}^{L} f_i x_{k-i}) \right]^2 / N_0 + \log \Pr(x_k), \tag{2.7}$$

where $\log \Pr(x_k)$ is the *a priori* information, and can be computed as

$$\log \Pr(x_k) = \begin{cases} \Lambda_a(x_k) - \ln(1 + e^{\Lambda_a(x_k)}), if\ q(x_k = 1 \mid (s_{k-1}, s_k)) = 1 \\ -\ln(1 + e^{\Lambda_a(x_k)}), \qquad if\ q(x_k = 0 \mid (s_{k-1}, s_{k-1})) = 1 \end{cases}, \tag{2.8}$$

where $\Lambda_a(x_k)$ is the *a priori* LLR, provided by the extrinsic LLR in the previous iteration, and $q(x_k \mid (s_{k-1}, s_{k-1}))$ is the conditional probability of the instant transmitted bit. The path metric $M_k(s_k)$ and the ML path selection is exactly the same as in PRML, except that the soft reliability for every x_k is estimated. Again, decisions can be made with delay D.

Define the difference metric for state s_k at time k as

$$\Delta_k = \left| [M_{k-1}(s_{k-1}^{(1)}) + \gamma_k(s_{k-1}^{(1)}, s_k)] - [M_{k-1}(s_{k-1}^{(2)}) + \gamma_k(s_{k-1}^{(2)}, s_k)] \right|, \tag{2.9}$$

where $s_{k-1}^{(1,2)}$ are the two states at time k-1, with branches entering s_k at time k. It was shown in [15] that Δ_k represents the reliability that the path ending at s_k at time k is correct. Let us now track back from time $k+D$. First a hard decision $\hat{x}_k$ is made. Along the ML path, there are $D+1$ non-survivor paths that have been discarded, and each non-survivor path has a certain difference metric Δ_j for $k \le j \le k+D$. Finally, the LLR $\Lambda(x_k)$ is computed as [15],

$$\Lambda(x_k) = \hat{x}_k \cdot \min\{\Delta_k, \Delta_{k+1}, \cdots, \Delta_{k+D}\}. \tag{2.10}$$

2.4 BCJR

The BCJR algorithm [9] is an optimum symbol-by-symbol channel detection algorithm. It was first investigated for finite-state machines, but can be easily used for convolutional and turbo decoding and PR channel detection. The BCJR algorithm is an MAP algorithm. However, in this section it is described in the log domain, and is called a log-MAP algorithm.

Let us look back at (2.3) and rewrite it as

$$\Lambda(x_k) = \log \frac{\sum_{s_{k-1}\to s_k : x_k=1} \Pr(s_{k-1}, s_k, \mathbf{r}) / \Pr(\mathbf{r})}{\sum_{s_{k-1}\to s_k : x_k=0} \Pr(s_{k-1}, s_k, \mathbf{r}) / \Pr(\mathbf{r})}, \tag{2.11}$$

and note hat $\Pr(s_{k-1}, s_k, \mathbf{r})$ can be decomposed as follows,

$$\Pr(s_{k-1}, s_k, \mathbf{r}) = \Pr(s_{k-1}, \mathbf{r}_1^{k-1}) \cdot \Pr(s_k, r_k \mid s_{k-1}) \cdot \Pr(\mathbf{r}_{k+1}^N \mid s_k). \tag{2.12}$$

Define

$$\alpha_k(s_k) = \log\{\Pr(s_k, \mathbf{r}_1^k) / \Pr(\mathbf{r}_1^k)\} \tag{2.13}$$

$$\beta_k(s_k) = \log\{\Pr(\mathbf{r}_{k+1}^N \mid s_k) / \Pr(\mathbf{r}_{k+1}^N \mid \mathbf{r}_1^k)\} \tag{2.14}$$

$$\gamma_k(s_{k-1}, s_k) = \log \Pr(s_k, r_k \mid s_{k-1}), \tag{2.15}$$

and use the approximation

$$\log(e^{\delta_1} + e^{\delta_2}) \approx \max{}^*(\delta_1, \delta_2) = \max(\delta_1, \delta_2) + \log(1 + e^{-|\delta_1 - \delta_2|}), \tag{2.16}$$

then $\alpha_k(s_k)$ and $\beta_k(s_k)$ can be computed by the forward and the backward recursions,

$$\alpha_k(s_k) = \max_{s_{k-1}}{}^*\{\alpha_{k-1}(s_k) + \gamma_k(s_{k-1}, s_k)\} \tag{2.17}$$

$$\beta_{k-1}(s_{k-1}) = \max_{s_k}{}^*\{\beta_k(s_k) + \gamma_k(s_{k-1}, s_k)\}, \tag{2.18}$$

where $\gamma_k(s_{k-1}, s_k)$ is already defined in (2.7). The initializations for $\alpha_k(s_k)$ and $\beta_k(s_k)$ are $\alpha_0(s_0 = S_0) = 0, \alpha_0(s_0 \neq S_0) = -\infty$ and $\beta_0(s_0 = S_0) = 0$, $\beta_N(s_0 \neq S_0) = -\infty$. Finally the LLR $\Lambda(x_k)$ is computed as

$$\begin{aligned}\Lambda(x_k) = & \max^*_{s_{k-1}\to s_k : x_k=1} \{\alpha_{k-1}(s_{k-1}) + \gamma_k(s_{k-1}, s_k) + \beta_k(s_k)\} \\ & - \max^*_{s_{k-1}\to s_k : x_k=0} \{\alpha_{k-1}(s_{k-1}) + \gamma_k(s_{k-1}, s_k) + \beta_k(s_k)\}.\end{aligned} \tag{2.19}$$

This LLR consists of two parts, the *a priori* LLR, $\Lambda_a(x_k)$, and the extrinsic LLR, $\Lambda_e(x_k)$,

$$\Lambda_e(x_k) = \Lambda(x_k) - \Lambda_a(x_k), \tag{2.20}$$

which is used as the *a priori* LLR, $\Lambda_a(x_k)$, in next iteration of channel detection.

Note that the above BCJR algorithm using the approximation (2.16) is called a log-MAP algorithm. The second term of (2.16), namely, $\log(1+e^{-|\delta_1-\delta_2|})$, can be implemented by a look-up table. Another version, using the simplified approximation

$$\log(e^{\delta_1} + e^{\delta_2}) \approx \max(\delta_1, \delta_2), \tag{2.21}$$

is called a max-log-MAP algorithm [16]. The performance of the max-log-MAP is slightly worse than the log-MAP algorithm; however, its complexity is much lower.

2.5 Forward-MAP

Note that in the BCJR algorithm, all signals $\mathbf{r}$ are used to make decisions on $\hat{x}_k$, and implemented as both the forward and backward recursions. However, it is apparent from (2.1) that x_k only affects channel output signals, $r_k, r_{k+1}, \dots, r_{k+L}$, denoted by $\mathbf{r}_k^{k+L}$, from which $\hat{x}_k$ should be able to be decided with delay L. The MAP algorithm that uses $\mathbf{r}_1^{k+D}$ to detect $\hat{a}_k$ with a delay $D \geq L$ implemented using only the forward iteration is called a forward-MAP algorithm [17]. If the computation is carried out in the log-domain, the algorithm is called a log-forward-MAP algorithm. The complexity of this algorithm is the same as the sliding-window BCJR algorithm, and it can be decoded continuously.

The branch metric and the forward recursion of this algorithm are exactly the same as for the BCJR algorithm in the previous section. However, we do not need the backward recursion. Instead, another term

$$\theta(x_{k-D}, s_k) = \log \Pr(x_{k-D} \mid s_k, \mathbf{r}_1^k) \tag{2.22}$$

is defined, and a new recursion on $\theta(x_{k-D}, s_k)$ is needed,

$$\theta(x_{k-D}, s_k) = \max_{s_{k-1}}{}^* \left\{\theta(x_{k-D}, s_{k-1}) + \alpha_{k-1}(s_{k-1}) + \gamma_k(s_{k-1}, s_k)\right\} - \max_{s_{k-1}}{}^* \left\{\alpha_{k-1}(s_{k-1}) + \gamma_k(s_{k-1}, s_k)\right\}, \tag{2.23}$$

with initialization

$$\theta(x_{k-D}, s_{k-D+L}) = \alpha_{k-D+L-1}(s_{k-D+L-1}) + \gamma_{k-D+L}(s_{k-D+L-1}, s_{k-D+L}) - \alpha_{k-D+L}(s_{k-D+L}), \tag{2.24}$$

where $s_{k-D+L-1}$ is uniquely determined by x_{k-D} and s_{k-D+L}. Finally the LLR is computed as

$$\Lambda(x_{k-D}) = \max_{s_k}{}^* (\theta(x_{k-D} = 1, s_k) - \max_{s_k}{}^* (\theta(x_{k-D} = 0, s_k). \tag{2.25}$$

The delay D must be at least L. However, it has been shown in [17] that the longer D, the better the performance. From this perspective, the BCJR algorithm is just a special case of the forward-MAP algorithm, where $D = \max(N-k, L)$ for the k-th symbol, $k = 1, 2, \cdots, N$. Since the log-forward-MAP algorithm uses the MAP criterion, we can state that the BCJR algorithm is the best possible MAP algorithm.

2.6 Algorithm Comparison

The complexity as well as performance of the above described channel detection algorithms are compared in this section. In the sequel of this dissertation, the log-MAP BCJR algorithm is primarily used. However, other algorithms are also considered as specified.

2.6.1 Performance Comparison

Intuitively, the performance ranking, from the best to the worst, is as follows: log-MAP BCJR, log-forward-MAP, SOVA and PRML. More quantitative results can be obtained using simulations. Examples of such simulations are conducted for LDPC coded ideal EPR4 channels with AWGN. The reason for using an LDPC code is to demonstrate the superiority of soft-decision detection algorithms over hard-decision ones. The LDPC code used is a regular code of length $N = 4374$, rate $R = 8/9$, column weight $w_c = 4$ and row weight $w_r = 36$. The maximum number of LDPC decoding iterations is set at fifty. More details about LDPC codes can be found in Chapter 3.

Shown in Fig. 2.2 are the BER simulation results for the non-precoded ideal EPR4 channel with the channel detection algorithms running only once. Note that the rate of the LDPC code is included in the definition of SNR. The performance ranking of the four channel detection algorithms used is in agreement with expectations. More specifically, at BER=10^{-5}, the performance gains in terms of SNR, from right to left, are 1.2, 0.9, and 0.05 dB, respectively.

It is well known that channel detection with *a priori* information, in which the MAP criterion is used, can improve performance. This is why the SOVA and the BCJR algorithms are popular. However, to enhance this improvement, a soft ECC decoder is needed to iterate with them, in the turbo equalization mode. Although the LDPC decoder used in Fig 2.2 is such a decoder, more details must be presented to elaborate on the mechanism and effect of turbo equalization. For this reason, we leave this issue to Chapter 3.

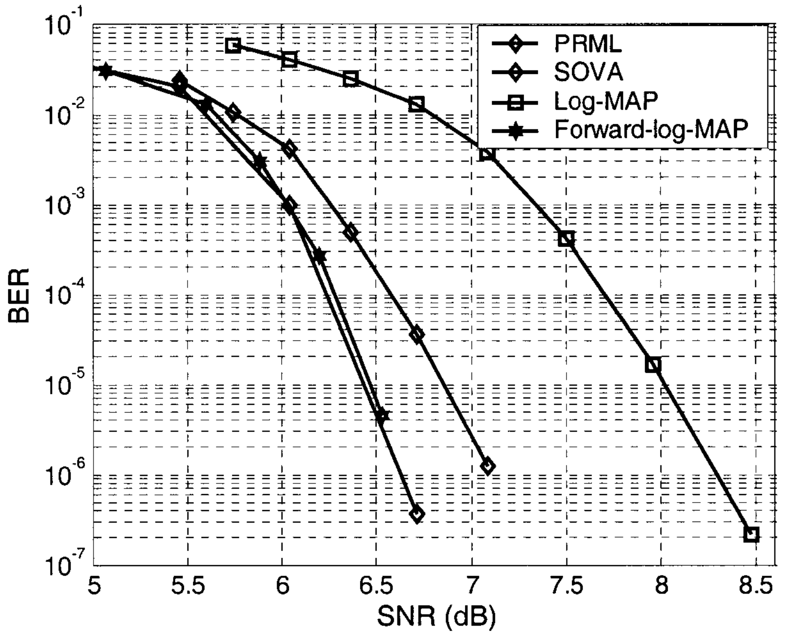


Fig. 2.2. Comparison of channel detection on ideal EPR4 channels.

2.6.2 Complexity Comparison

A complexity comparison between PRML, SOVA, log-MAP max-log-MAP and log-forward-MAP, per bit is illustrated in Table 2.1. Note that the complexities of SOVA and log-forward-MAP are for those with delay *D*.

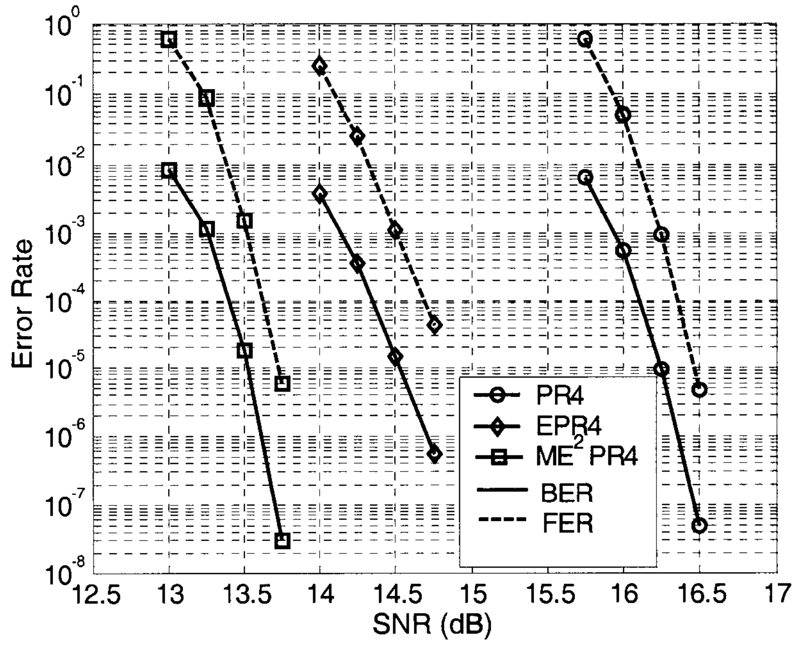
It can been easily seen from Table 2.1 that the complexity ranking of these algorithms is just the opposite of their performance ranking, except for the log-forward-MAP algorithm. The log-forward-MAP algorithm is a log-MAP algorithm, but its complexity is even higher than that of the standard log-MAP algorithm. Its advantage is the capability of continuous decoding, which is not an issue for PRML and SOVA, but is a problem for MAP-type algorithms.

Table 2.1. Complexity comparison of PR channel detection algorithms.

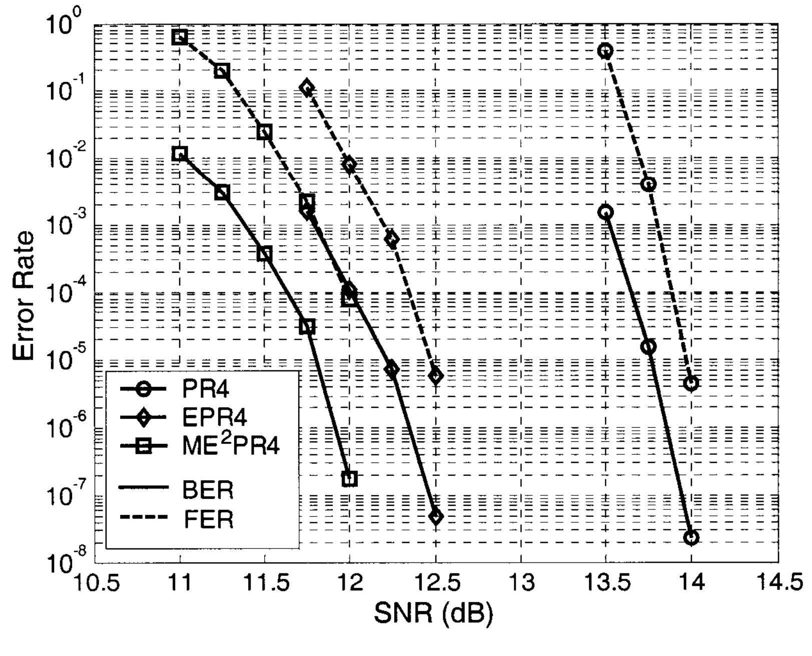
	PRML	SOVA	Log-MAP	Max-log-MAP	Log-forward-MAP
Addition	$4\cdot 2^L$	$4\cdot 2^L+3D$	$12\cdot 2^L$	$8\cdot 2^L$	$(4+6D)\cdot 2^L$
Multiply	$2\cdot 2^L$	$2\cdot 2^L$	$2\cdot 2^L$	$4\cdot 2^L$	$2\cdot 2^L$
Max/Min	$2\cdot 2^L$	$2\cdot 2^L+D-1$	$4\cdot 2^L$	$4\cdot 2^L$	$(3+2D)\cdot 2^L$
Look-ups			$4\cdot 2^L$		$(3+2D)\cdot 2^L$

2.7 Comparison of Longitudinal Recording Targets

In the rest of this chapter, we only consider the log-MAP BCJR algorithm for channel detection. Both LMRC and PMRC are studied, in this and the next section, respectively. The purpose is to compare different PR targets, as outlined in Section 1.3, where we discussed the two factors in target selection, spectral match and the handling of low-frequency disturbances. This discussion is checked using computer simulations.



(a)



(b)

Fig. 2.3. Performance comparison of LDPC coded LMRC with (a) AWGN, (b) 10% AWGN plus 90% media noise.

The following targets, for LMRC with $S_c = 2.995$ and AWGN or 10% AWGN plus 90% media noise, are considered: PR4, EPR4, and ME^{2}PR4. A random LDPC code (4608, 4096, 4, 36) is used as the ECC. The BCJR channel detector runs only once; and the MP LDPC decoder runs a maximum of fifty iterations. Shown in Fig. 2.3 are the simulation results in terms of BER as well as frame error rate (FER).

It is seen from Fig. 2.3 (a) that the ranking of these three targets, from the best to the worst, is ME^{2}PR4, EPR4, and PR4. This is not unexpected, agrees with Section 1.3 and [18], and can be explained by the extent of spectral match. However, the lengths of these three targets are $L = 5$, 4 and 3, respectively, and the complexity of the channel detection decreases exponentially.

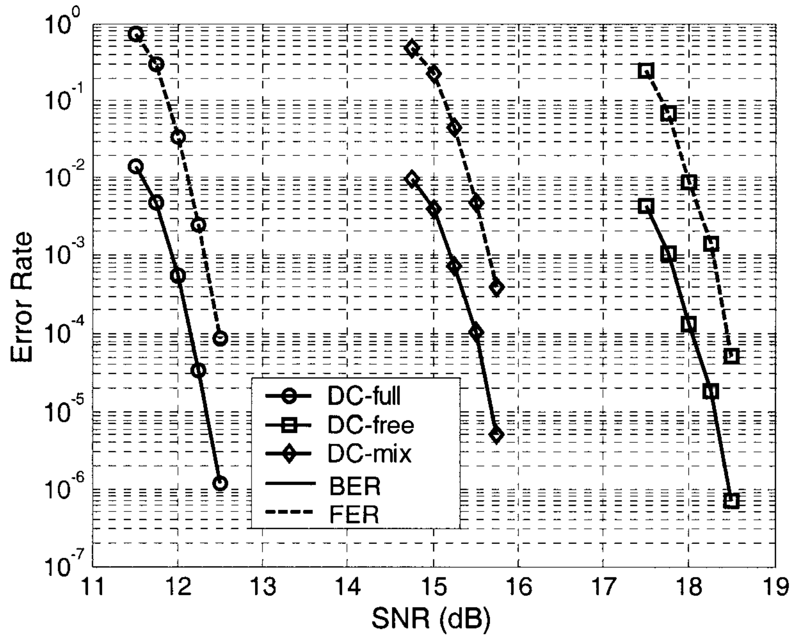
The results in Fig. 2.3 (b) agree with Fig. 2.3 (a). This indicates that the low-frequency disturbance is not as critical as the spectrum match. Unlike for PMRC, all the targets for LMRC have a null at DC. So the impact of low-frequency disturbance is about the same on all targets considered.

2.8 Comparison of Perpendicular Recording Targets

The three GPR targets in Table 1.2 are used in this section for a PMRC with $S_c = 1.4$, and either AWGN, or 10% AWGN plus 90% media noise.

First let us consider a PMRC with AWGN. In this case, since low-frequency disturbance is not a severe problem, it seems that DC-full targets are good choices, because their spectra match well with that of the channel response. On the other hand, both the DC-free and DC-mix targets suffer some SNR penalty.

These intuitive statements can be checked using simulations. The system model is



(a)

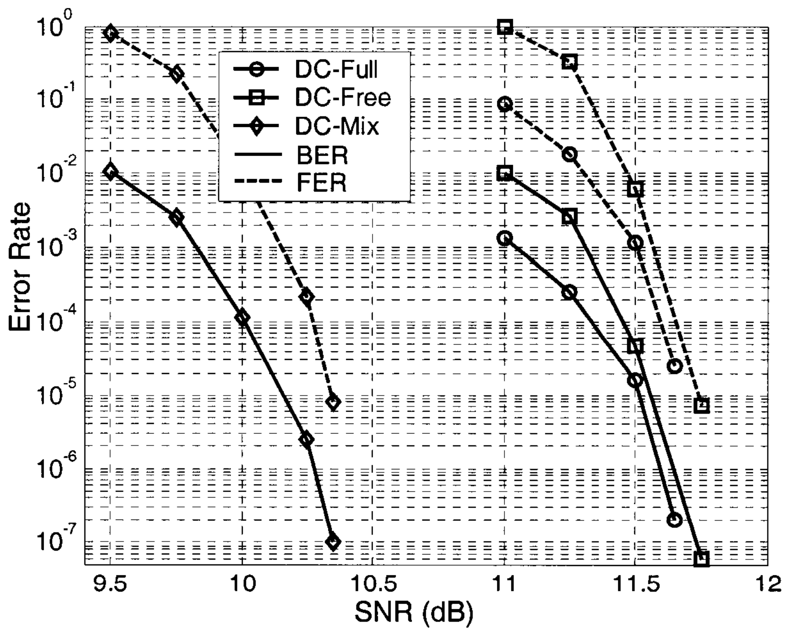


Fig. 2.4. Performance comparison of LDPC coded PMRC with (a) AWGN, (b)10% AWGN plus 90% media noise.

the same as for longitudinal recording, except that the PMRC channel is equalized to the targets listed in Table 1.2.

Plotted in Fig. 2.4 (a) are the simulation results. Apparently, the DC-full target is superior to the other two targets by 6.0 and 3.4 dB, respectively, at BER=10^{-5}. This observation is consistent with those of the PRML detection for uncoded PMRCs in [5] and indicates adequately that in the AWGN-dominant channel the spectral match is the key issue.

Next we consider PMRC with 10% AWGN plus 90% media noise. In contrast to the previous case, low-frequency disturbance is a major concern. So neither the DC-full target nor the DC-free target is a good choice. Instead, the DC-mix target may be a good tradeoff. It has been shown in [5] that the DC-mix target in Table 1.2 is the best choice. We redo this work for an LDPC coded PMRC with the same simulation settings as above and show the results in Fig. 2.4 (b).

From Fig. 2.4 (b), we see that this DC-mix target is better than the DC-full and DC-free targets by 1.35 and 1.40 dB, respectively, at BER=10^{-5}. Again these results are consistent with those of [5], and support the statement that both spectral match and low-frequency disturbance handling must be considered in GPR target selection.

Chapter 3

Low-Density Parity-Check Codes

LDPC codes [19], [20] and turbo codes [10] are two type of ECCs that can be decoded with iterative APP decoding algorithms. Both codes have excellent performance approaching the Shannon limit, thus attracting tremendous attention in recent years. However, LDPC codes have been shown to have close to or better performance than turbo codes on AWGN channels. For example, it was shown in [21] that a LDPC code can be 0.0045 dB away from the Shannon limit, although the convergence of this LDPC code is generally slower than turbo codes Furthermore, convolutional turbo codes usually have an error floor at about BER=10^{-6}; however, this error floor is rarely observed for LDPC codes. The decoding complexity of LDPC codes is much lower than turbo codes, and can be implemented using parallel architectures. Therefore, we are mainly investigating LDPC codes in this dissertation.

LDPC codes were initially introduced by Gallager in the 1960s [19]. However, due to limited computing capability, these codes had been forgotten for thirty years until turbo codes were invented in the early 1990s [10], which motivated MacKay *et al.* [22] to rediscover them. Since then, LDPC codes have been widely studied for many different communication channels. Typical examples include [20], [21] for AWGN channels and [23], [24] for MRCs.

This chapter investigates the application of LDPC codes to ideal EPR4 channels. Several types of LDPC codes, namely, non-structured (random) codes, and structured codes such as FG codes [25] and array codes [26], [27] are considered.

3.1 Introduction to LDPC Codes

A binary LDPC code is a linear ECC defined by a sparse parity-check matrix $\mathbf{H} = \{h_{i,j}\}_{N \times M}$, where N and M are the number of columns and rows, and $h_{i,j} \in \{0,1\}$. Each column of this matrix corresponds to a bit, and each row to a check. From this point of view, codewords are sequences of length N that satisfy a set of M binary parity checks. In other words,

$$\mathbf{h}_i \cdot \mathbf{x} = 0 \text{ for } i = 1,2,\cdots,N-1 \tag{3.1}$$

where $\mathbf{h}_i = (h_{i,1}, h_{i,2}, \cdots, h_{i,N})$ is the i-th row of $\mathbf{H}$ and $\mathbf{x} = (x_1, x_2, \cdots, x_N)^T$ is a codeword. More concisely,

$$\mathbf{H} \cdot \mathbf{x} = \mathbf{0}. \tag{3.2}$$

The number of user information bits is $K = N - M$, and the code rate is $R = K / N$ assuming that the matrix $\mathbf{H}$ is of full rank. A generator matrix $\mathbf{G}$ corresponding to $\mathbf{H}$ can be easily obtained by Gaussian elimination.

Other than the parity-check matrix, another way to represent an LDPC code is using an undirected bipartite graph. This graph is called a factor graph or a Tanner graph following its inventor's name [28]. The factor graph is more meaningful than the parity-check matrix itself because it shows the essence of the message-passing decoding algorithm for LDPC codes.

A factor graph of a parity-check matrix **H** consists of N variable (or bit) nodes and M check nodes, corresponding to the columns and rows of **H** one-to-one, and indicated by circles and squares, respectively. There is an edge between the j-th variable node and the i-th check node if and only if $h_{i,j} = 1$. For example, the factor graph for the parity-check matrix

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

is shown in Fig. 3.1, where $x_i, i = 1,2,\cdots,9$, and $\mathbf{h}_j, j = 1,2,\cdots,5$ denote variable nodes and check nodes, respectively.

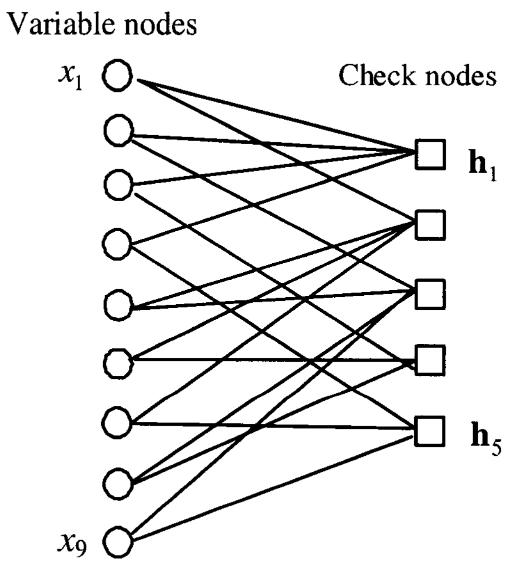


Fig. 3.1. An example of factor graph.

The degree of a node is defined as the number of edges incident on this node. LDPC codes with one single variable degree and one single check degree are called regular codes; otherwise, they are called irregular codes.

For regular LDPC codes, the variable degree is just the number of ones in each column of the parity-check matrix **H**, therefore, it is also called column weight; similarly, the check degree is also called row weight. These two weights are denoted by w_c and w_r, respectively, in this dissertation, and codes with these weights are denoted by (N, K, w_c, w_r). If the corresponding parity-check matrix is full-rank, the code rate $R = 1 - w_c / w_r = 1 - M / N$.

For irregular LDPC codes, the distribution of the variable degrees can be specified by the variable degree distribution function

$$\lambda(x) = \sum_i \lambda_i x^{i-1}, \tag{3.3}$$

where λ_i is the fraction of edges with variable degree i. Similarly, the check degree distribution function is

$$\rho(x) = \sum_i \rho_i x^{i-1} \tag{3.4}$$

where ρ_i is the fraction of edges with check degree i. The rate of this code is $R = 1 - \int_0^1 \rho(x)dx \Big/ \int_0^1 \lambda(x)dx$. Note that λ_i and ρ_i are defined with respect to edge. They can be converted to $\overline{\lambda}_i$ and $\overline{\rho}_i$ with respect to node as follows,

$$\overline{\lambda}_i = \frac{\lambda_i / i}{\sum_i \lambda_i / i} \tag{3.5}$$

$$\bar{\rho}_i = \frac{\rho_i / i}{\sum_i \rho_i / i}. \tag{3.6}$$

It will be seen in Chapter 4 that the degree distribution pair (λ, ρ) is very important in the context of optimizing LDPC codes.

3.2 Message-Passing Algorithm

LDPC codes are basically decoded using the MP algorithm (also called belief propagation algorithm or sum-product algorithm), which was first introduced in [19] and then elaborated in [20]. While there are some variations, the algorithm in [20] is used in this dissertation.

3.2.1 Basic MP Algorithm

Assume a codeword **x** being transmitted and the receiver computes the channel information,

$$p_j^b = \Pr(x_j = b). \tag{3.7}$$

for $b = 0, 1$. Let the set of bits participating in check i be $\Phi(i) = \{j : h_{i,j} = 1\}$ and the set of checks that bit j participates be $\Psi(j) = \{i : h_{i,j} = 1\}$. In each decoding iteration, the following two messages $q_{i,j}$ and $r_{i,j}$, called variable message and check message,

$$r_{i,j}^b = \Pr(\text{check } i \text{ is met} \mid x_j = b, \Pr(x_{j'}) = q_{i,j'} \text{ for } j' \in \Phi(i) \setminus \{i\}) \tag{3.8}$$

$$q_{i,j}^b = \Pr(x_j = b \mid \Psi(j) \setminus \{i\}) \tag{3.9}$$

are iteratively updated with messages passed from the neighbors of the underlying variable or check node through the horizontal step,

$$r_{i,j}^{b} = \frac{1}{2}\left[1+(-1)^{b}\prod_{j'\in\Phi(i)\setminus\{j\}}\delta q_{i,j'}\right], \tag{3.10}$$

where $q_{i,j'} = q_{i,j'}^{1} - q_{i,j'}^{0}$, and the vertical step,

$$q_{i,j}^{b} = \alpha_{i,j}p_{j}^{b}\prod_{i'\in\Psi(j)\setminus\{i\}} r_{i',j}^{b} \tag{3.11}$$

respectively, where $\alpha_{i,j}$ is a normalization factor such that $q_{i,j}^{0}+q_{i,j}^{1}=1$. Note that (3.11) is derived from (3.9) by assuming that the information from different checks is independent. Based on the updates in (3.10) and (3.11), the APPs $\Pr\left(x_j = b \mid \mathbf{p}^{b}\right)$, $b=0,1$, are estimated as

$$q_{j}^{b} = \alpha_{j}p_{j}^{b}\prod_{i\in\Psi(j)} r_{i,j}^{b} \tag{3.12}$$

where α_j is chosen such that $q_j^0 + q_j^1 = 1$.

The message passing processes of (3.10) and (3.11) can be illustrated in a more visual way, such as the one in Fig. 3.2, where Fig. 3.2 (a) corresponds to (3.11) and Fig. 3.2 (b)

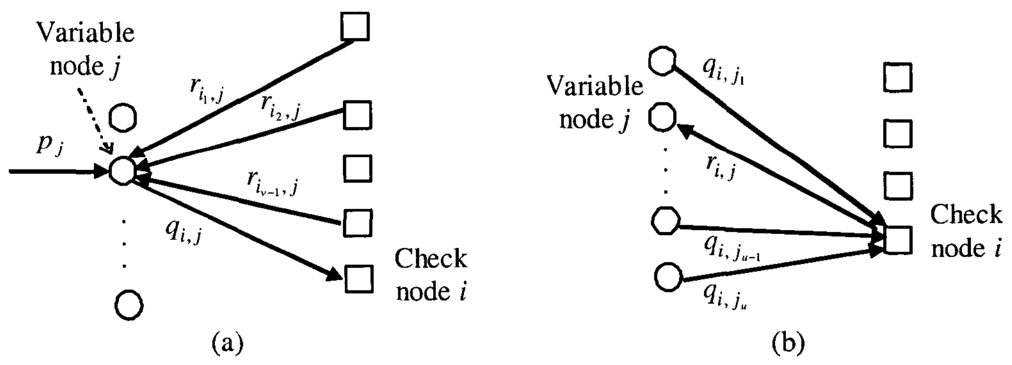


Fig. 3.2. Message passing processes in the MP algorithm.

corresponds to (3.10), respectively. For the j-th variable node in Fig 3.2 (a), the degree of this node is $v = |\Phi(j) \setminus \{i\}|$, while for the i-th check node in Fig 3.2 (b), the degree of this node is $u = |\Psi(i) \setminus \{j\}|$. The visual presentation of (3.12) can be similarly obtained.

A hard decision $\hat{x}_j^b$ is made based on q_j^b, namely $\hat{x}_j = 1$ if $q_j^1 > 0.5$, otherwise $\hat{x}_j = 0$. The syndrome $\mathbf{H} \cdot \hat{\mathbf{x}}$ is checked and used as the criterion to determine if a valid codeword can be found. If $\mathbf{H} \cdot \hat{\mathbf{x}} = \mathbf{0}$ then the codeword $\hat{\mathbf{x}}$ is found valid, the decoding algorithm stops and output the codeword $\hat{\mathbf{x}}$; otherwise the algorithm keeps going until the maximum number of iterations is reached, in which a detected error is declared. No matter if a valid codeword can be found or not, the final APPs $\mathbf{q}^b$, b=0,1, are available for output and may be used for iterating with the channel detection.

3.2.2 Log-MP Algorithm

By representing the quantities $q_{i,j}$ and $r_{i,j}$ in the form of LLR, the MP algorithm can be expressed in the log-domain, and is referred to as the Log-MP algorithm [20].

Let us first define the LLRs, $\Lambda(p_j) = \log \frac{p_j^1}{p_j^0}$, $\Lambda(r_{i,j}) = \log \frac{r_{i,j}^1}{r_{i,j}^0}$, $\Lambda(q_{i,j}) = \log \frac{q_{i,j}^1}{q_{i,j}^0}$ and $\Lambda(q_j) = \log \frac{q_j^1}{q_j^0}$. Then the updates in (3.10)-(3.12) are modified to

$$\Lambda(r_{i,j}) = -2 \tan^{-1}\left(\prod_{j' \in \Phi(i) \setminus \{j\}} (-1) \tanh\left(\Lambda(q_{i,j'})/2\right) \right) \tag{3.13}$$

$$\Lambda(q_{i,j}) = \sum_{i' \in \Psi(j) \setminus \{i\}} \Lambda(r_{i',j}) + \Lambda(p_j) \tag{3.14}$$

$$\Lambda(q_j) = \sum_{i' \in \Psi(j)} \Lambda(r_{i',j}) + \Lambda(p_j). \tag{3.15}$$

Other elements, including the stopping criterion and hard decision, are the same as in the basic MP algorithm.

Since almost all arithmetic operations in the MP algorithm, either the basic one or the one using LLRs, are addition and multiplication, the MP algorithm is also referred to as sum-product algorithm.

Some simplified versions of the MP algorithm are available. One example is the max-product algorithm. More considerations from the perspective of reduced-complexity and reduced-storage implementation can be found, e.g., in [16] and [29].

3.2.3 Iterating with Channel Detection: Turbo Equalization

One advantage of the MP algorithm is that it outputs soft reliability information in the form of APPs or LLRs, which allows iterative channel detection and LDPC decoding, referred to as turbo equalization [30]. This is a unique property that soft-decision decoding algorithms have but hard-decision ones do not.

A diagram of this turbo equalization is shown in Fig. 3.3. The blocks with a dashed border are an interleaver (Π) and a deinterleaver (Π^{-1}), and the dashed border means that these blocks may or may not be present. The messages passed between the channel detector and the LDPC decoder are LLRs, where "ch/code" denotes the channel

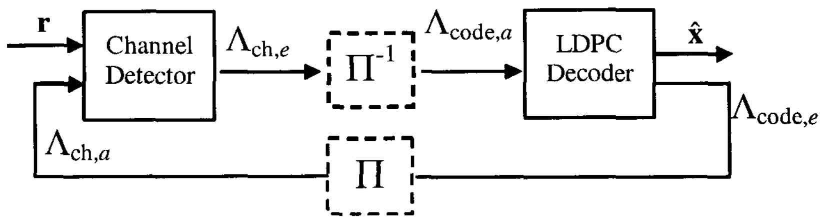


Fig. 3.3. Turbo equalization for an LDPC coded PR channel.

detector/LDPC decoder, and "*a*/*e*" denotes the *a priori*/extrinsic LLRs. Note that the channel detector extrinsic LLR, $\Lambda_{\mathrm{ch},e}$, can be obtained using the channel detection algorithms in Chapter 2 with the *a priori* LLR, $\Lambda_{\mathrm{ch},a}$, and the LDPC decoder extrinsic LLR, $\Lambda_{\mathrm{code},e}$ $(=\{\Lambda(q_j)\})$, can be obtained using the MP algorithm with the *a priori* LLR, $\Lambda_{\mathrm{code},a}$ $(=\{\Lambda(p_j)\})$.

Numerical examples of turbo equalization using the log-MAP BCJR algorithm for channel detection are shown in Fig. 3.4 for the non-precoded EPR4 channel. The same LDPC code as in Fig. 2.2 is used with the same decoding settings. The simulation results for one to five iterations are shown. The performance improvement by turbo equalization is clearly observed. Three iterations of the turbo equalization are enough to obtain most of the improvement.

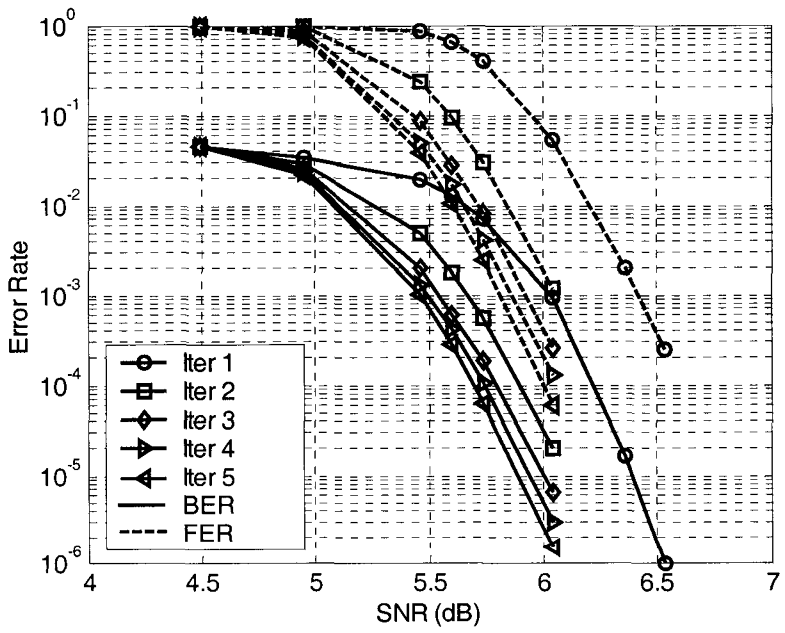


Fig. 3.4. Performance of turbo equalization.

3.3 Design of Regular LDPC Codes

One of the merits of LDPC codes is that high-rate codes are very easy to design. With the PR channel detection algorithms and the LDPC decoding algorithms explained, we are now ready to use LDPC codes for magnetic recording. First we consider regular codes. Irregular codes will be treated in Chapter 4.

All codes designed in this section are applied to the non-precoded EPR4 channel. The log-MAP BCJR algorithm and the MP algorithm are used for channel detection and LDPC decoding, respectively. The BCJR algorithm runs only once and the maximum number of iterations of the MP algorithm is set to fifty. In other words, no turbo equalization is used.

3.3.1 Random Codes

LDPC codes were rediscovered by first noting the excellent performance of random codes [20]. These codes were constructed using a random search, which has since been a very popular and successful approach.

A typical example of such a random search algorithm is the one in [20]. In this algorithm, a new column of the parity-check matrix is randomly created one by one until a matrix fulfilling all requirements is obtained. The restriction on this new column is that it has w_c ones and does not have cycles of length four with previous columns. The matrix obtained may not have a regular row weight w_r; however, the distribution can be made as uniform as possible. A collection of codes of a wide range of rates designed using this method can be found in [31].

Another algorithm creates a random mapping between codeword bits and parity-checks corresponding to a parity-check matrix and checks the mapping for cycles of length four [23]. If any such cycle is found, the elements of the mapping are randomly permuted until one without such cycle is obtained. This method can be used for designing irregular codes as well. Simulations show that random codes constructed using these two algorithms have similar performance.

One set of codes with different specifications is designed for demonstration. These codes have weights $(w_c, 10w_r), w_c = 3,4,5,6$ and rate $R = 0.9$. Shown in Fig. 3.5 are the simulation results for these codes. It is observed that for this set of codes, the smaller the column weight, the better the performance. On the other hand, the larger the column weight, the more complex the LDPC decoder. So we can state that regular codes with column weight $w_c = 3$ are good choices in terms both performance and complexity.

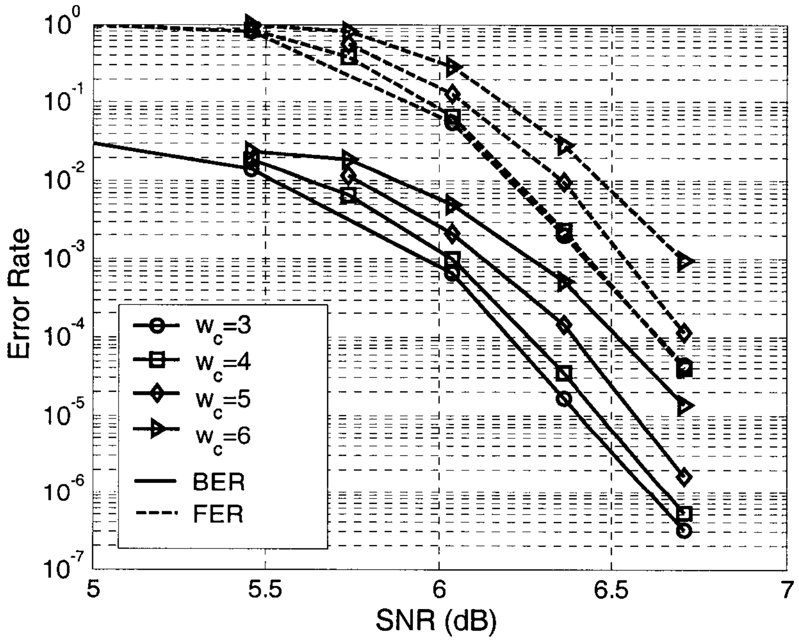


Fig 3.5. Random LDPC codes $(4374,3888, w_c, 10w_r)$, $w_c = 3,4,5$ and 6.

3.3.2 Finite-Geometry Codes

The encoder of a random code presents a challenge due to the large memory required to store the matrix and the often prohibitive complexity of the matrix multiplication. Although an efficient encoding method was introduced in [32] reducing the complexity by up to two orders of magnitude, the complexity is still very high when the code length is large. This motivated the search for cyclic or quasi-cyclic algebraic LDPC codes, which can be encoded using shift-register circuitry. One issue, however, is whether algebraic codes have as good performance as random codes. In this section we first consider one important type of structured LDPC codes based on finite geometry (FG). Another type of codes based on array codes will be considered next.

FG-LDPC codes were introduced by Kou *et al.* [25], using a code construction closely associated with FG. The basic idea is that the rows and columns of the parity-check matrix correspond to the points and lines of a FG. The geometries we are currently using are projective geometry (PG) and Euclidean geometry (EG), in which it is guaranteed that there are no cycles of length four.

Two type of FG codes are defined in [25], namely, the type-I and type-II codes. While the type-I codes use the incident matrix of the points on the lines in the FG as the parity-check matrix, the type-II codes use its transpose. We consider some type-I PG (PG-I) codes, type-II PG (PG-II) codes, type-I EG (EG-I) codes and type-II EG (EG-II) codes as listed in Table 3.1. Note that PG/EG(m,s) denotes the code based on geometry PG/EG$(m,2^s)$ in Table 3.1, where $d_{\min}$, w_c, w_r and R denote the minimum distances of the code, the column and row weight of the parity-check matrix and the code rate respectively.

Table 3.1. FG-LDPC Codes.

Code	m	s	N	K	R	$d_{\min}$	w_c	w_r
PG-I	2	5	1057	813	0.7692	34	33	33
PG-I	2	6	4161	3431	0.8246	66	65	65
PG-II	3	3	4745	4344	0.9155	10	9	73
PG-II	4	2	5797	5499	0.9486	6	5	85
PG-II	5	1	651	594	0.9124	4	3	31
PG-II	6	1	2667	2547	0.9550	4	3	63
EG-I	2	5	1023	781	0.7634	33	32	32
EG-I	2	6	4095	3367	0.8222	65	64	64
EG-II	3	3	4599	4227	0.9191	9	8	72
EG-II	4	2	5355	5121	0.9563	5	4	84

The FG codes have the following properties [33]:

- They are regular LDPC codes, with column weight w_c and row weight w_r. For PG-I code, $w_c = w_r = m^s + 1$, and for EG-I codes, $w_c = w_r = m^s + 1$.
- Type-I FG codes are cyclic and Type-II codes are quasi-cyclic. This is a very useful property that makes linear-time encoding with shift-register circuit hardware implementation possible.
- The lengths and rates of FG codes are not very flexible; however, they can be made more flexible by puncturing or splitting the parity-check matrix [25].

Several examples of FG codes, namely, PG-I(2,6), PG-II (3,3), and EG-II(4,2) in Table 3.1 are selected for simulations on the non-precoded EPR4 channel. Plotted in Fig.

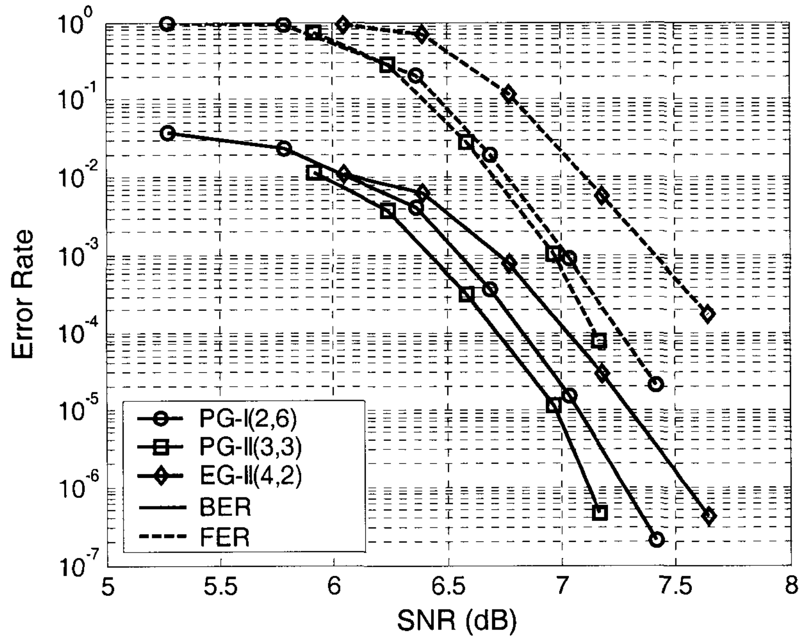


Fig. 3.6. PG-I(2,6), PG-II (3,3), and EG-II(4,2) codes.

3.6 are the simulation results. Since a PG-I/II (m,s) code has a structure very similar to an EG-I/II (m,s) code, it is not a surprise that they have similar performance, as shown in [25]. This is the reason why we do not include EG-I(2,6), EG-II(3,3) and PG-II(4,2) in Fig. 3.6. It can be seen that while the first code is just fair, its performance is worse than the higher-rate code PG-II(3,3), code PG-II(3,3) is a very good code, and its performance will be compared to other random and array codes in Section 3.3.4.

3.3.3 Array Codes

We review briefly array codes introduced in [26] in this section. Array codes refer to a class of codes defined on two-dimensional arrays. They have structure similar to RS codes except that they are defined over Galois rings instead of Galois fields. Since array codes are MDS codes in the ring, they have good error and erasure correction capability.

Algebraic decoding of array codes is similar to that for RS codes. Recently, array codes were found to have sparse binary parity-check matrices, and therefore are LDPC codes [26].

Let p be an odd prime, and γ an arbitrary integer such that $r \leq p$. The codewords of an array code can be defined as a square matrix $\mathbf{V}_{p\times p}$ such that for all $l = 0,1,...,r-1$ and $i = 0,1,..., p-1$,

$$\sum_{j=0}^{p-1} \mathbf{V}_{\langle i+jl \rangle_p, l} = \mathbf{0} \tag{3.16}$$

where $\langle g \rangle_p$ is the modulo-p residue. In other words, the code consists of all arrays whose entries along the p lines of slope $i = 0,1,..., p-1$ sum to zero.

It was reported in [26] that multiple algebraic decoders are available and an array code with r check symbols is able to correct one symbol error and r-2 symbol erasures, or r symbol erasures (one symbol has p bits).

Let $\mathbf{V}_{p\times p} = [\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_{p-1}]$, where each $\mathbf{v}_i$, $i = 0,1,..., p-1$, is a column vector and also a symbol in the array code. Arrange the columns of the array into a single vector $\mathbf{v}$, then we can rewrite the constraints in (3.16) in matrix form [26],

$$\mathbf{H} \cdot \mathbf{v} = \mathbf{0} \tag{3.17}$$

where

$$\mathbf{H} = \begin{bmatrix} \mathbf{I} & \mathbf{I} & \mathbf{I} & \dots & \mathbf{I} \\ \mathbf{I} & \boldsymbol{\sigma} & \boldsymbol{\sigma}^2 & \dots & \boldsymbol{\sigma}^{p-1} \\ \mathbf{I} & \boldsymbol{\sigma}^2 & \boldsymbol{\sigma}^4 & \dots & \boldsymbol{\sigma}^{2(p-1)} \\ \dots & \dots & \dots & \dots & \dots \\ \mathbf{I} & \boldsymbol{\sigma}^{r-1} & \boldsymbol{\sigma}^{2(r-1)} & \dots & \boldsymbol{\sigma}^{(r-1)(p-1)} \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} \mathbf{v}_0 \\ \mathbf{v}_1 \\ \mathbf{v}_2 \\ \dots \\ \mathbf{v}_{p-1} \end{bmatrix},$$

and $\boldsymbol{\sigma}$ is the $p \times p$ single-cyclic-shifting matrix. It is observed that $\mathbf{H} \cdot \mathbf{v} = \mathbf{0}$ is exactly the parity check equation of an LDPC code if both the matrix $\mathbf{H}$ and the vector $\mathbf{v}$ are represented in GF(2), i.e., the array code is an LDPC code over GF(2).

Note that $\mathbf{H}$ is determined by two parameters: p and r. The (i, j)-th $p \times p$ submatrix of $\mathbf{H}$ is $\mathbf{H}_{i,j} = \boldsymbol{\sigma}^{i \cdot j}$ for $i = 0,1,...r-1,\ j = 0,1,..., p-1$. That is, the exponents of $\mathbf{H}_{i,j}$ are $\alpha_i = i, \beta_j = j$. More generally, the matrix $\mathbf{H}$ can be defined by three parameters: p, r and $k < p$ (and the corresponding code is denoted by (p, r, k), and two exponent sequences α_i and β_j for $i = 0,1,...r-1, j = 0,1,..., k-1$, where the α_i's as well as the β_j's are not all equal [27]. Specifically,

$$\mathbf{H} = \begin{bmatrix} \boldsymbol{\sigma}^{\alpha_0\beta_0} & \boldsymbol{\sigma}^{\alpha_0\beta_1} & \boldsymbol{\sigma}^{\alpha_0\beta_2} & \cdots & \boldsymbol{\sigma}^{\alpha_0\beta_{k-1}} \\ \boldsymbol{\sigma}^{\alpha_1\beta_0} & \boldsymbol{\sigma}^{\alpha_1\beta_1} & \boldsymbol{\sigma}^{\alpha_1\beta_2} & \cdots & \boldsymbol{\sigma}^{\alpha_1\beta_{k-1}} \\ \boldsymbol{\sigma}^{\alpha_2\beta_0} & \boldsymbol{\sigma}^{\alpha_2\beta_1} & \boldsymbol{\sigma}^{\alpha_2\beta_2} & \cdots & \boldsymbol{\sigma}^{\alpha_2\beta_{k-1}} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \boldsymbol{\sigma}^{\alpha_{r-1}\beta_0} & \boldsymbol{\sigma}^{\alpha_{r-1}\beta_1} & \boldsymbol{\sigma}^{\alpha_{r-1}\beta_2} & \cdots & \boldsymbol{\sigma}^{\alpha_{r-1}\beta_{k-1}} \end{bmatrix}. \tag{3.18}$$

This matrix has column weight $w_c = r$, row weight $w_r = k$ and is free of cycles of length four. The code defined by this matrix has length $N = p \cdot k$, dimension $K = pk - r(p-1) - 1$ and code rate $R = 1 - \frac{r(p-1)+1}{p \cdot k}$. The parity-check matrix $\mathbf{H}$ in (3.17) is just a special case of (3.18) with $k = p$. Note that (3.18) is a generalization of the results in [26], [34]. Further considerations were suggested in [26] concerning the removal of short cycles and improving the minimum distance by additional coding of the column symbols.

The easiest way to construct an array-based LDPC code specified by (p,r,k) according to (3.18) is to set $\alpha_i = i$ and $\beta_j = j$ for $i = 0,1,...r-1, j = 0,1,...,k-1$. However, if cycles of length six are deemed detrimental to the performance of the MP algorithm, then short-cycle-removing techniques outlined in [26] can be used. By choosing appropriate values of (p,k,r), we can design regular LDPC codes for different applications. For example, the LPDC code (23,6,12) is a short code (276,133,6,12) of rate $R = 0.48$; the (127,4,40) code is a medium length code (5080, 4575, 4, 40) of rate

Table 3.2. High-rate array-based LDPC codes.

(p,r,k)	N	K	w_c	w_r	R
(63,6,54)	4482	3989	6	54	0.8900
(101,4,47)	4747	4346	4	47	0.9155
(101,5,45)	4545	4044	5	45	0.8898
(113,4,40)	5240	4071	4	40	0.9007
(127,4,36)	4572	4067	4	36	0.8895
(131,4,36)	4716	4195	4	36	0.8895
(131,4,40)	5240	4719	4	40	0.9006
(149,3,30)	4470	4025	3	30	0.9004
(167,3,27)	4509	4010	3	27	0.8893

$R = 0.9$. More high-rate codes appropriate for MRCs are listed in Table 3.2, some of which are used for demonstration using simulations, as shown in Fig. 3.7. The codes used in Fig 3.7 are of almost the same rate $R = 0.889$ but different column and row weights. Typically, we choose codes of column weight $w_c = 3,4,5$ and 6, respectively, similar to those in Fig. 3.5.

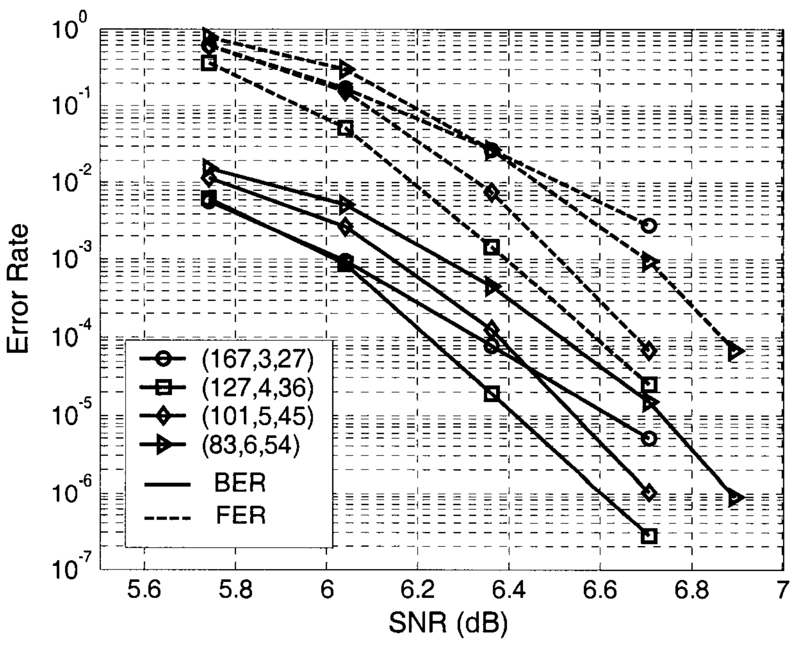


Fig. 3.7 Array-based codes (63,6,54), (101,5,45), (127,4,36) and (167,3,27).

3.3.4 Performance Comparison

As stated above, random codes usually have good performance, but the encoding complexity and storage is a challenge. On the other hand, structured LDPC codes usually have easy encoding implementation because they are either cyclic or quasi-cyclic. The question is whether structured codes are as good as random codes. As far the FG and array codes are concerned, it has been shown in [25] and [26] that these two types of structured codes are very good codes, at least for AWGN channels.

Now we compare these three types of codes on the non-precoded EPR4 channel using simulations. First, we must note that for comparison fairness, we must enforce the codes to have specifications, namely, code rate, length, column and row weights, as similar as possible. However, since some codes, e.g., FG codes, are not flexible, we have to be careful in selecting codes. The three codes we selected are:

- Random code: (4745,4338,3,35), $R = 0.9143$.
- FG code: PG-II(3,3) – (4745,4344,9,73), $R = 0.9155$..
- Array code: (101,4,47) – (4747,4346,4,47), $R = 0.9155$.

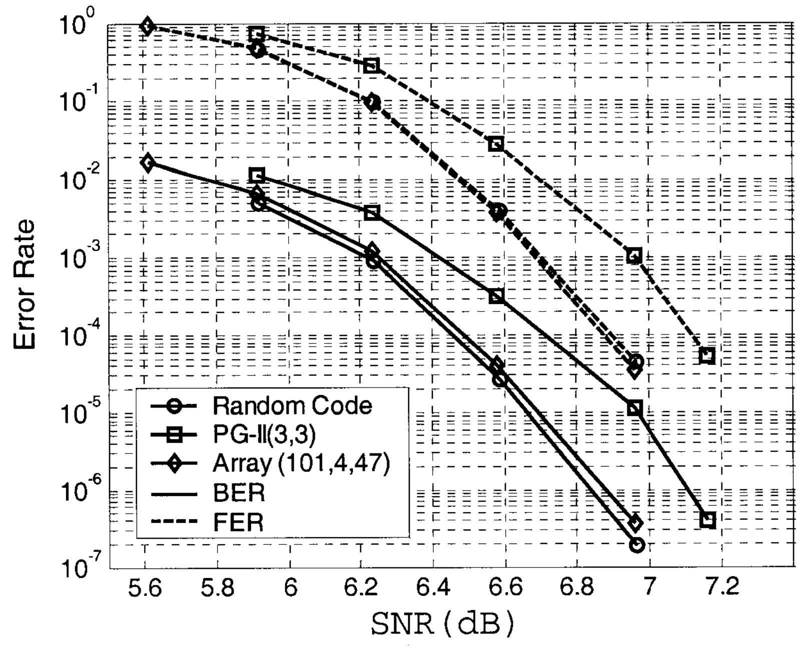


Fig. 3.8 Comparison of random code (4745, 4338, 3, 35), FG code FG-II(3, 3) and array code (101, 4, 47).

Plotted in Fig. 3.8 are the simulation results. It is observed that while the FG code is just fair, the array code is as good as the random code. This shows that this particular array code, in terms of both performance and encoding complexity, is possibly a good choice for practical magnetic recording application. Simulation results for more array codes, not shown here, indicate a similar trend.

Chapter 4

Density Evolution

Density evolution [38] is a newly introduced and powerful technique to model the behavior of iterative decoding for LDPC and turbo-like codes. This idea was first suggested by Gallager [19], and implemented by Luby *et al.* [39] for binary symmetric channels. Richardson and Urbanke [38] extended it to arbitrary binary-input symmetrical-output channels. Specifically, for a given class of LDPC codes, it provides a way to compute the probability density function (pdf) of the messages in the iterative decoding. One interesting aspect of this technique is the noise threshold phenomenon: if the channel noise is less than a certain threshold, the probability of error will go to zero asymptotically; otherwise, the probability is always bounded away from zero.

This technique was applied to PR channels in [40] with some modifications. Since the concatenation of a linear code and the PR channel is not a linear code any longer, we cannot use the all-zero codeword as reference. Another important point is that the consistency symmetry [38] does not hold for PR channels. So the Gaussian approximation simplification [41], in which only the mean of the messages is tracked, cannot be used, although the message densities are still very likely Gaussian. Instead, the discretized pdf of the messages is tracked in the density evolution [21].

Using this technique, Kavcic *et al.* [40] derived bounds for ensembles of LDPC codes on ISI channels given a particular codeword or assuming that an identical and uniformly distributed (i.u.d.) codeword is transmitted. However, these bounds are for message errors, i.e., the probability of erroneous variable messages in the MP decoding.

In this chapter, we extend it to the probability of symbol errors (bits in binary alphabets) [42] and modified the theories in [40] accordingly. We must note that although the system performances in terms of the probability of message errors have already reflected the essence of iterative decoding, system performance in terms of probability of symbol errors is a more commonly used better understood metric. More importantly, we can estimate the average BER of a given ensemble of LDPC codes on a PR channel using this probability of symbol errors, which can be justified by simulations.

4.1 Information Rate of PR Channels

Before proceeding to density evolution, we first discuss techniques to compute the symmetric information rate and the BCJR-once bound for a PR channel with AWGN, i.e., $r_k = \sum_i x_i f_{k-i} + n_k$. These two theoretical limits will be used as criteria to justify optimization of irregular LDPC codes.

4.1.1 Symmetric Information Rate

Capacity is the best theoretical limit for any channel. However, the computation of the capacity of a PR channel

$$C = \max_{p_{\mathbf{X}(\mathbf{x})}} I(\mathbf{X};\mathbf{R}) = \max_{p_{\mathbf{X}(\mathbf{x})}} \big(\mathcal{H}(\mathbf{R}) - \mathcal{H}(\mathbf{R} \mid \mathbf{X})\big), \tag{4.1}$$

where $\mathcal{H}(\,)$ is an entropy function, and $I(\mathbf{X};\mathbf{R})$ is the mutual information between $\mathbf{X}$ and

R, is a classical problem that has not been solved yet, although a number of (tight) bounds are available. Instead, we use the technique in [43] to estimate the symmetric information rate, i.e., the information rate of a PR channel with an i.u.d input sequence, i.e., $p_{\mathbf{X}}(\mathbf{x}) = 2^{-N}$ for any **x**. Let this information rate be denoted by

$$C_{\text{i.u.d.}} = \left[\mathcal{H}(\mathbf{R}) - \mathcal{H}(\mathbf{R} \mid \mathbf{X})\right]_{p_{\mathbf{X}}(\mathbf{x})=2^{-N}}. \tag{4.2}$$

Simply speaking, this information rate for the PR channel is computed in the following way. First, the differential entropies of **R** and **R** | **X** are defined as,

$$\mathcal{H}(\mathbf{R}) = \lim_{N\to\infty} \frac{1}{N} \mathcal{H}(\mathbf{r}) \tag{4.3}$$

$$\mathcal{H}(\mathbf{R} \mid \mathbf{X}) = \lim_{N\to\infty} \frac{1}{N} \mathcal{H}(\mathbf{r} \mid \mathbf{x}) = \lim_{N\to\infty} \frac{1}{N} \mathcal{H}(\mathbf{n}). \tag{4.4}$$

The differential entropy of the AWGN sequence **n** can be easily computed, actually, $\lim_{N\to\infty} \mathcal{H}(\mathbf{n})/N = 0.5\log_2(2\pi e\sigma^2)$. So the problem reduces to the computation of $\mathcal{H}(\mathbf{R})$, which can be expressed as,

$$\mathcal{H}(\mathbf{R}) = -\lim_{N\to\infty} \frac{1}{N} E[\log(p(\mathbf{r}))], \tag{4.5}$$

where $p(\mathbf{r})$ is the probability of the channel output sequence **r**. Fortunately, $p(\mathbf{r})$ can be computed by the forward recursion of the Baum-Welch/BCJR algorithm. For more details please refer to [43].

4.1.2 BCJR-Once Bound

Another theoretical limit for a PR channel is the BCJR-once bound, which is very useful to characterize losses in terms of achievable information rate if the BCJR-once channel detection is employed [40]. This is important because BCJR-once is usually used in practical systems for the sake of low complexity.

Assuming a transmitted i.u.d. sequence, the BCJR-once bound is defined as,

$$C_{\text{BCJR-once}} = \lim_{N\to\infty} \frac{1}{N} I(\mathbf{X};\Lambda)\,|_{p_{\mathbf{X}(\mathbf{x})}=2^{-N}}\,, \tag{4.6}$$

where Λ is the BCJR output LLR sequence. It can be easily proven that $C_{\text{BCJR-once}} \leq C_{\text{i.u.d.}} \leq C$.

The BCJR-once bound can be computed by simulation in a way similar to that for symmetrical information rate [40]. Namely, generate a very long sequence $\mathbf{x}$, transmit it through the PR channel, and obtain the channel output sequence $\mathbf{r}$. Then,

$$C_{\text{BCJR-once}} = 1 - \frac{1}{N}\sum_j \mathcal{H}\big(\Pr(x_j = 1 \,|\, \mathbf{r})\big). \tag{4.7}$$

4.2 LDPC Decoding Revisited

We consider an ensemble of LDPC codes specified by a pair of degree distribution functions (λ, ρ), as described in Section 4.1.

The decoder for this system consists of an MAP channel detector, which is the log-MAP BCJR algorithm in this chapter, and the MP LDPC decoder. Turbo equalization may or may not be used. The iterative decoding behavior of both decoders will be analyzed in terms of message passing and their density evolutions. A system without the turbo equalizer is straightforward.

There are several message-passing processes in the decoding system. In addition to the ones in the MP decoding, others are the messages from the channel to the variable nodes and from the variable nodes back to the channel. These message-passing processes can be represented clearly with a joint code/channel graph [40].

Given a transmitted codeword, we first compute the channel output extrinsic messages $\Lambda^{l}_{\mathrm{ch},e}$ in the l-th iteration by running the BCJR algorithm with the *a priori* message $\Lambda^{l}_{\mathrm{ch},a} = \Lambda^{l-1}_{\mathrm{code},e}$, which is available in the previous round, passed from the variable nodes. The message passing in the MP algorithm is exactly the same as that on the AWGN channel [21]. Two types of messages coming forward and backward between variable nodes and check nodes are updated iteratively. Let q^{l} be the message from a variable node of degree d_q to a check node in the l-th round, then q^{l+1} is equal to the sum of the channel message and the incoming messages from all its neighbors except the one that will get the message q^{l+1},

$$q^{l+1} = \Lambda^{l}_{\mathrm{ch},e} + \sum_{i=1}^{d_q-1} r_i^{l} . \tag{4.8}$$

The message update rule for check nodes is in the form of "tanh rule" [38]:

$$\tanh\frac{r^{l+1}}{2} = \prod_{j=1}^{d_r-1} \tanh\frac{q_j^{l+1}}{2}, \tag{4.9}$$

where the q_j's are the incoming messages from all neighbors of this check node except the one that will receive the message r^{l+1}. The last message is the extrinsic message $\Lambda^{l+1}_{\mathrm{code},e}$ passed from the variable node to the channel,

$$\Lambda^{l+1}_{\mathrm{code},e} = \sum_{i=1}^{d_q} r_i^{l+1} , \tag{4.10}$$

i.e., the sum of messages from all the neighbors of this variable node. This iteration process will continue unless a stopping criterion is met. If all checks are satisfied, all bits will be decided to either a digit one or minus one based on the sign of the message, i.e., $\hat{x} = sign(s^{l+1})$, where

$$s^{l+1} = \Lambda^{l}_{\mathrm{ch},e} + \Lambda^{l+1}_{\mathrm{code},e}, \tag{4.11}$$

otherwise, a decoding failure will be declared when the maximum iteration number is reached.

4.3 Density Evolution

After l such iterations, the algorithm would produce the exact LLR of all the bits if the bipartite graph contains no loops of length up to $2l$ [38]. If we assume that the graph is loop-free, we can analyze the decoding algorithm directly by tracking the evolution of the message densities in the iterations because the incoming messages to every node are independent. This algorithm is called density evolution in [38]. Also, by the general concentration theorem of [38], for almost all the graphs in a code ensemble (λ, ρ) and almost all inputs, the decoder performance on a binary-input symmetrical-output channel will converge to that of a corresponding loop-free graph as the codeword length approaches infinity. This theorem was extended to PR channels in [40]. It states that given a particular (or i.u.d.) transmitted codeword from a uniformly chosen ensemble code, the probability of erroneous messages can be approximated arbitrarily closely by the error concentration probability if the code length is larger than a certain value. This error concentration probability can be easily estimated from the density evolution. In this work, we extend this concept to erroneous symbols.

The density evolution for an LDPC code with distribution (λ, ρ) on a PR channel is exactly the same as in the decoding process. Let the probability density function of a random variable A be f_A. Define similarly as in [21], [40],

$$\lambda(f) = \sum_i \lambda_i \otimes^{i-1} f \tag{4.12}$$

$$\rho(f) = \sum_i \rho_i \, \mathcal{R}^{i-1} f \tag{4.13}$$

where $\otimes$ denotes convolution of pdf, and $\mathcal{R}^{i-1} f$ is symbolic notation for the average message pdf by evolving the pdf f through a check node of degree i,

$$\mathcal{R}^{i-1} f = \mathcal{R}(f, \mathcal{R}(f, \ldots \mathcal{R}(f, f), \ldots, f)), \tag{4.14}$$

where $\mathcal{R}(a,b) = 2\tanh^{-1}(\tanh(a/2)\cdot\tanh(b/2))$ [21]. Note that all these computations are in the discretized pdf domain. With these definitions, the density evolution corresponding to (4.8)-(4.11) can be written as

$$f^{l}_{\Lambda_{\text{ch},e}} = H(\mathbf{r}_{\text{i.u.d.}}, f^{l}_{\Lambda_{\text{ch},a}}, f_n) \tag{4.15}$$

$$f^{l+1}_{q} = f^{l}_{\Lambda_{\text{ch},e}} \otimes \lambda(f^{l}_{r}) \tag{4.16}$$

$$f^{l+1}_{r} = \rho(f^{l+1}_{q}) \tag{4.17}$$

$$f^{l+1}_{\Lambda_{\text{code},e}} = f^{l}_{\Lambda_{\text{ch},e}} \otimes \bar{\lambda}(f^{l+1}_{r}), \tag{4.18}$$

$$f^{l+1}_{s} = f^{l+1}_{\Lambda_{\text{code},e}} \otimes f^{l}_{\Lambda_{\text{ch},e}}, \tag{4.19}$$

where $\bar{\lambda}(f) = \sum_i \frac{\lambda_i}{i\int_0^1 \lambda(x)dx} \otimes^i f$ [40], f_n is the pdf of the channel noise, $\mathbf{r}_{\text{i.u.d.}}$ is the received channel signal corresponding to the transmitted i.u.d. sequence, and H denotes histogram of the BCJR channel detector output LLRs obtained by simulation, because there is no close-form expression. Note that the convolution of pdfs can be efficiently calculated in the Fourier domain. From f_s^l we can estimate the average decoding error.

In the case where turbo equalization is not used, the density evolution can be simplified straightforwardly where the channel detection is only processed once and the extrinsic message from the LDPC code is not fed back to the channel.

The use of an i.u.d. sequence in (4.15) is because the concatenation of a code and a PR channel is a nonlinear code and the all-zero codeword cannot be used for reference. This can be justified for high-rate linear codes. The length of the sequence must be long enough for us to be able to ignore the channel detection boundary effects. Since we do not know the codeword length in the concentration theorem, we will determine it experimentally.

4.4 Properties of Density Evolution

4.4.1 Concentration Theorems

Kavcic *et al.* [40] derived the concentration theorems for LDPC codes over ISI channels by analyzing the neighbors and trees in the joint channel-code graph. In their theory, for a given transmitted codeword, the probability of a variable-to-check message being erroneous after l iterations of the MP decoding algorithm is highly concentrated around the error concentration probability. Similar results apply to an i.u.d. codeword.

Instead of message error, we modify their theory to symbol error, which is more direct and meaningful. Note that with this modification, all related parameters are changed accordingly. But the computed value difference is small, because when the message error goes to zero, the symbol error will go to zero as well, and vice versa.

Using similar notation and derivation as in [40], we define π_θ^l as the probability that the tree of type θ and depth l delivers an incorrect symbol, i.e.,

$$\pi_\theta^l = \Pr(s^l \cdot x_\theta < 0 \mid \text{tree type } \theta). \qquad (4.20)$$

Then the symbol error concentration probability given a transmitted $\mathbf{x}$ is

$$P^l(\mathbf{x}) = \sum_{i=1}^{2^{N(l)}} \pi_{\theta_i}^l \Pr(\theta_i \mid \mathbf{x}). \qquad (4.21)$$

For an i.u.d. codeword, all possible $2^{N(l)}$ neighborhood types θ_i are equally probable,

$$P_{\text{i.u.d.}}^l = \sum_{i=1}^{2^{N(l)}} \pi_{\theta_i}^l \cdot 2^{-N(l)}. \qquad (4.22)$$

With these modifications, we can state the concentration theorems in terms of the symbol error probability in almost the same form as those in [40].

Theorem 1: For randomly chosen ensemble LDPC codes, let $Z^l(\mathbf{x})$ be the number of erroneous symbols after l iterations of the MP decoding when codeword $\mathbf{x}$ is transmitted, for arbitrarily small constant $\varepsilon > 0$, there exists a positive number β, such that if $N > 2\gamma / \varepsilon$, then

$$\Pr\left(\left| \frac{Z^l(x)}{N} - P_s^l \right| \geq \varepsilon \right) \leq e^{-\beta\varepsilon^2 N}. \qquad (4.23)$$

Theorem 2: For randomly chosen ensemble LDPC codes, let $\mathbf{x}$ be a random i.u.d. sequence and $Z^l(\mathbf{x})$ be the number of erroneous symbols after l iterations of the MP decoding when this i.u.d. sequence being transmitted, for arbitrarily small constant $\varepsilon > 0$, there exists a positive number β', such that if $N > 2\gamma / \varepsilon$, then

$$\Pr\left(\left|\frac{Z^l(\mathbf{x})}{N} - P^l_{\text{i.u.d.}}\right| \ge \varepsilon\right) \le 4e^{-\beta\varepsilon^2 N}. \tag{4.24}$$

The proofs in [7] also apply to these two theorems. So they are not repeated here.

4.4.2 Decoding Error Estimation

Using density evolution, we can compute the pdf of all the types of messages in the iterative decoding. Suppose an i.u.d. sequence is transmitted, the symbol concentration error probability is just the integral of the symbol message in error [42],

$$P^l_{\text{i.u.d.}} = \int_{-\infty}^{0} f_s^l(\xi)d\xi. \tag{4.25}$$

Note that in the computation of the average message densities for an ensemble of LDPC codes, the negative message (which corresponds to a modulated –1 signal) is flipped to be positive. From the concentration theorem, the probability of message error can be well approximated by this concentration error probability given the block length is greater than a certain number. Although we do not know what this number is for a specific ensemble of codes, we can evaluate it experimentally.

In fact, this approach was proposed in [41] to estimate the probability of symbol error on AWGN channels. In that case, the all-zero codeword is transmitted, and the pdf of the symbol message is approximated to be Gaussian and the probability of error is estimated from it. Because the coding system is linear, this approach works very well. However, since a linear code concatenated with a PR channel is not linear, we alternatively use an i.u.d. sequence as the reference in this system.

Here we simply use $P^l_{\text{i.u.d.}}$ as an approximation to the average decoding error for a high-rate ensemble of LDPC codes on PR channels, i.e., BER=$P^l_{\text{i.u.d.}}$. This approximation can be evaluated by simulation. The results given below will show that this estimate of the decoding error is very good when the block length is long enough.

4.4.3 Noise Threshold

The threshold phenomenon is observed in LDPC and turbo-like codes. When the channel parameter is above a certain value, the decoding error will go to zero if the iteration number goes to infinity. For channels with white Gaussian noise, this parameter is the variance of the noise. That is,

$$\sigma_{th} = \sup\left(\sigma : \lim_{l\to\infty} \lim_{n\to\infty} \text{BER} \to 0\right). \tag{4.26}$$

For linear codes on AWGN channels, we can assume the all-zero codeword is transmitted without loss of generality. However, for linear codes on PR channels, we take an i.u.d. sequence as the input to the system, i.e.,

$$\sigma_{th} = \sup\left(\sigma : \lim_{l\to\infty} \lim_{n\to\infty} P^l_{\text{i.u.d.}} \to 0\right) . \tag{4.27}$$

From the previous discussion, $P^l_{\text{i.u.d.}}$ can be computed as the integral of f^l_s, so

$$\sigma_{th} = \sup\left(\sigma : \lim_{l\to\infty} \lim_{n\to\infty} \int_{-\infty}^{0} f^l_s(\xi,\sigma)d\xi \to 0\right) . \tag{4.28}$$

The threshold phenomenon ensures that there exists a code for which the decoding error can be arbitrarily small if the noise variance is below the threshold. Thresholds can be used to compare codes as well. Codes with higher thresholds or lower SNR thresholds ($10\log_{10}\dfrac{1}{R\cdot 2\sigma^2_{th}}$) are better in general.

4.5 Performance of LDPC Coded PR Channels

Numerical results for both noise threshold and decoding error estimation are shown in this section. From the noise threshold, we can declare how close an LDPC code is to the theoretical limit.

4.5.1 Noise Threshold

With the assumption of an i.u.d. input sequence, we can compute the threshold of any LDPC code with a specified degree distribution, whether regular or irregular. Kavcic et al. [40] computed the thresholds for regular codes with variable degree $w_c = 3$ on the dicode channel and showed that the thresholds of high-rate regular codes are very close to their i.u.d. capacity. We recomputed the thresholds of these codes with the modifications in this chapter. Furthermore, we also computed them on other high-order PR channels. Because the memory in the dicode channel is only one, which will make the channel very likely to be memoryless with a random interleaver within it, we use an ME^{2}PR4 channel as an example of high-order channels.

Consider first regular codes (w_c, w_r). The computed thresholds for the codes $(3,w_r)$ where w_r is varying are shown in Figs. 4.1 and 4.2. Fig. 4.1 is the comparison of the thresholds of these codes with and without a turbo equalizer ("turbo" and "non-turbo" in the figures) on the normalized dicode channel with $f(D) = (1-D)/\sqrt{2}$ to the i.u.d. capacities. It can be seen that when the code rate is increasing from 0 to 1.0, the threshold is approaching the capacity curve whether a turbo equalizer is present or not. The difference is that there is about 1.0 dB performance loss for the system without a turbo equalizer when the code rate is greater than 0.25. This is also observed in [40],

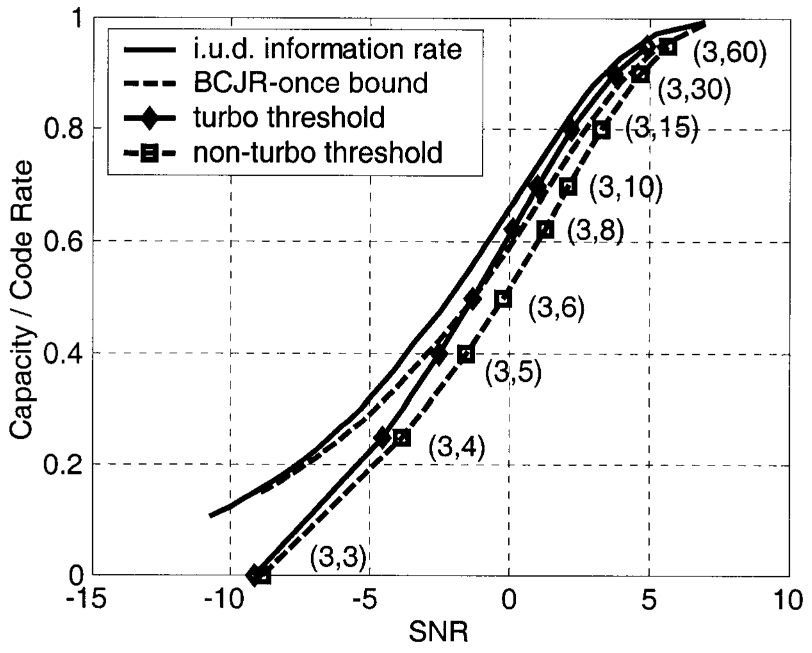


Fig. 4.1. Thresholds for the codes (3,*k*) on the dicode channel.

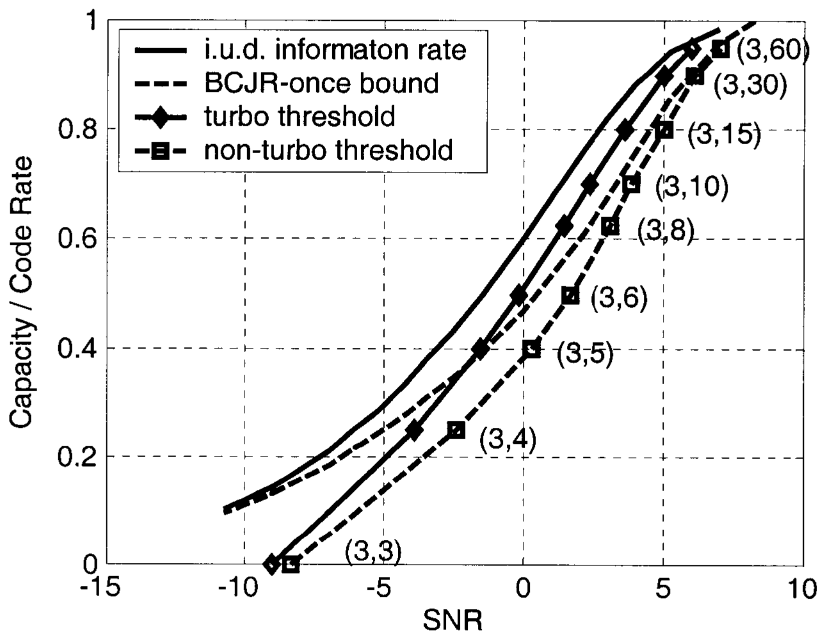


Fig. 4.2. Thresholds for the codes (3,*k*) on the ME^{2}PR4 channel.

where the definition of SNR is 3 dB offset from ours. Fig. 4.2 is the same comparison, but on a normalized ME^{2}PR4 channel with $f(D) = (2 + 2D - D^2 - 2D^3 - D^4) / \sqrt{14}$. We can draw similar conclusions from these figures that the technique proposed above works well for arbitrary PR channels.

However, the performance loss between the system with and without a turbo equalizer is larger than that on the dicode channel. But the codes that we are interested in are those of high-rate, where the code loss is approximately equal for these two channels. Note that the i.u.d. turbo and non-turbo capacities are computed using the method introduced in [43] and [40], where the former is called i.u.d. capacity while the latter is called the BCJR-once bound.

We also computed the thresholds for the codes (4, w_r) which are not shown. Furthermore, we computed the thresholds of the codes on a precoded PR channel. We chose $1/1 \oplus D^2$ as the precoder for the ME^{2}PR4 channel. The SNR threshold on this combined channel is larger than on the channel without the precoder, which means that the precoder in an LDPC coded system causes some performance loss.

One implication of these results is that the codes (3, w_r) of high rate are very good codes and may be optimal for PR channels. However, this is only an asymptotic result, in the sense that the code length and number of iterations go to infinity or are large enough. In practical applications, this needs to be evaluated and verified by simulations and/or experiments.

4.5.2 Decoding Error

Using the density evolution, we can estimate the average decoding BER as discussed above. Here we use the code (3,30) as an example. The PR targets considered are the dicode channel and the ME^2PR4 channel. In order for the density evolution to work properly, the code length is chosen to be large enough, we use $N = 10^6$. From [40] we know that density evolution does not depend on the code length. However, in the case of PR channels, we must use the BCJR simulation or density evolution to compute the channel-to-variable message, therefore a large number must be used to avoid the channel detection boundary effect [42]. The computed results are shown in Figs. 4.3 and 4.4. Since the coding gain from 50 to 500 iterations is very small, we use 50 as the maximum iteration number. We compare the codes (4,40) and (3,30), and code (3,30) with and without a precoder. All results are consistent with the information rate analysis.

In order to check the validity of the estimation of symbol error, we performed simulations. Since we do not know what the code length should be for density evolution to work properly, we use $N=10^4, 10^5$and 10^6. It can be seen from Figs. 4.3 and 4.4 that the decoding error estimation using density evolution works very well when the code length is larger than 10^5. The gap between the estimation and the simulation results for the code of length 10^6 is very small, less than 0.1 dB, and that for 10^5 is only 0.2 dB. As far as length 10^4 is concerned, the bit error estimation for the system with turbo equalizer works better than without it. At BER=10^{-5}, the gaps are about 0.5 dB. But the simulation curve for the system with a turbo equalizer is sharper. So for codes of length larger than 10^5, we can use the BER estimation by density evolution as the probability of decoding error down to a low BER, confidently.

Note that the codes of finite-length used in this chapter are randomly constructed. No attempt is made to remove short cycles. However, no cycle is assumed in the density evolution, so if short cycles, particularly cycles of length four are thoroughly removed

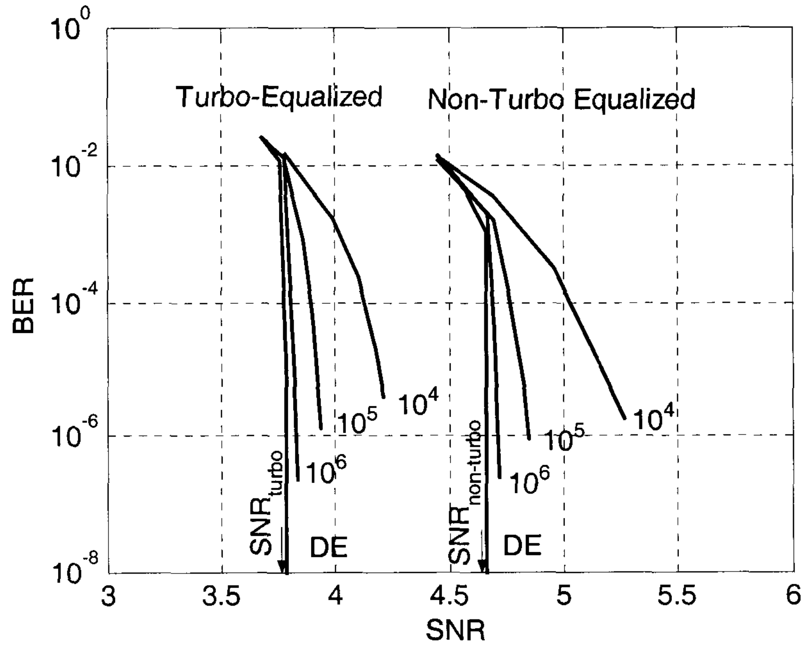


Fig. 4.3. Decoding error estimation using density evolution vs. simulation for the codes (3,30) on the dicode channel.

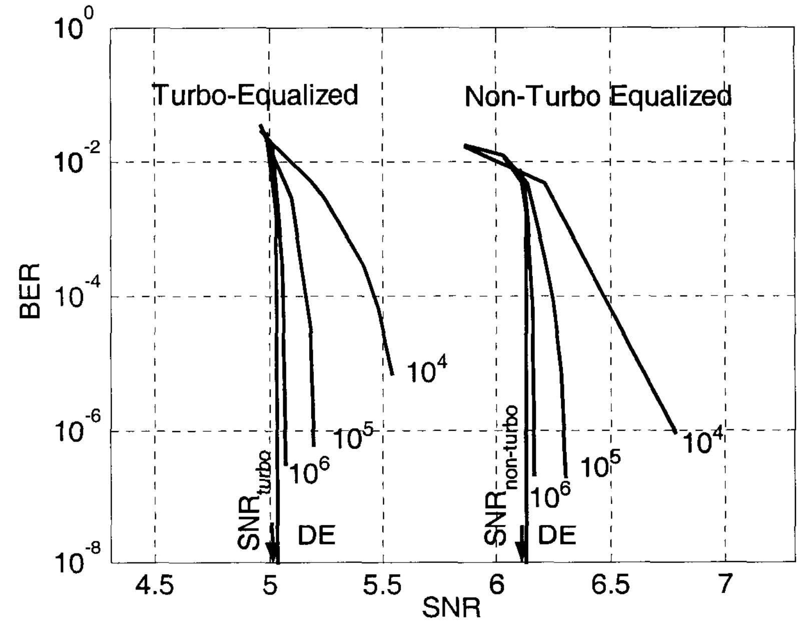


Fig. 4.4. Decoding error estimation using density evolution vs. simulation for the codes (3,30) on the ME^2PR4 channel.

from the codes, their performance will be even closer to the density evolution estimates. Some further results on this discussion will be given in Chapter 5, where density evolution is used for analyzing LDPC codes on PR channels with erasures.

4.6 Analyzing Precoding Effect

As indicated in Chapter 1, the precoder is a very important component in a magnetic recording system. In addition to converting NRZI data into NRZ data, it also affects system performance.

The precoding effect on system performance can be evaluated using density evolution asymptotically. As a follow up, simulation is used for finite-length codes. A number of precoders, listed in Table 4.1, are considered for the EPR4 channel. Also listed in Table 4.1 are the SNR thresholds for the LDPC (3,30) coded EPR4 channel. It is seen that no-precoder is optimal, and the performance losses due to precoding are from 0.2 to 0.7 dB. If a precoder has to be used, $1/1 \oplus D^2$ is the best choice, with only 0.22 dB loss.

Table 4.1. SNR thresholds and simulation results for LDPC (3,30) coded EPR4 channel. The simulations used a particular code of length 4374.

Precoder	SNR_{th} (dB)	SNR_{th} loss over no precoder (dB)	Simulated SNR loss over no precoder at BER=10^{-5} (dB)
No	5.71	0	0
$1/1 \oplus D$	6.29	0.48	0.65
$1/1 \oplus D^2$	5.93	0.22	0.06
$1/1 \oplus D \oplus D^2$	6.34	0.63	0.59
$1/1 \oplus D \oplus D^2 \oplus D^3$	6.37	0.66	0.68

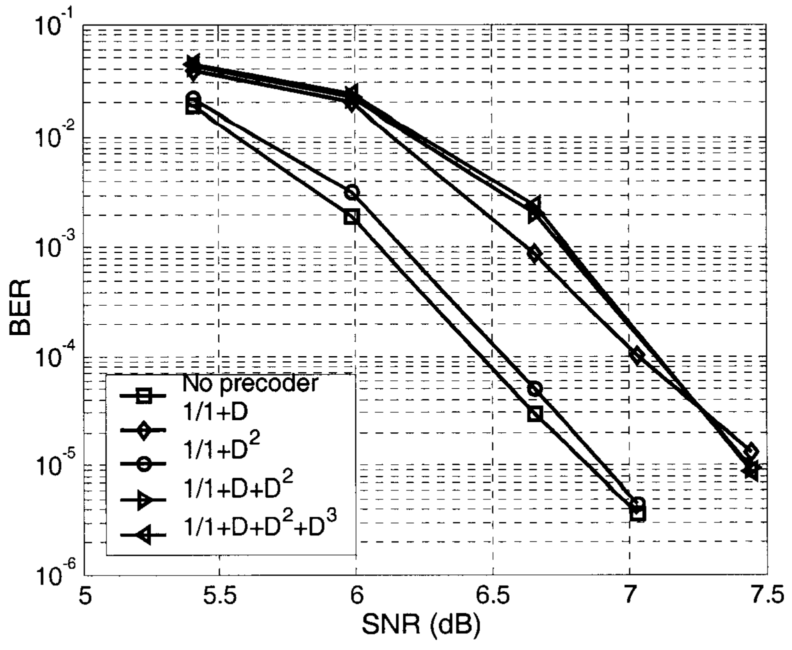


Fig. 4.5. LDPC (4374,3933,3,30) code on EPR4 with several precoders.

Shown in Fig 4.5 are the simulation results for a particular regular LDPC code (4370,3933,3,30) on the EPR4 channel. The performance losses caused by precoding, listed in the fourth column of Table 4.1, are about the same level as in density evolution for asymptotic and ensemble codes.

4.7 Designing Good LDPC Codes

Density evolution can be used to design good LDPC codes by optimizing the degree distribution functions (λ,ρ). The general idea is to search for (λ,ρ) that has the maximum noise threshold (or minimum SNR threshold).

The searching process is typically a nonlinear programming problem. Good examples of such process can be found in [21], [41], [44]-[48]. For completeness, the

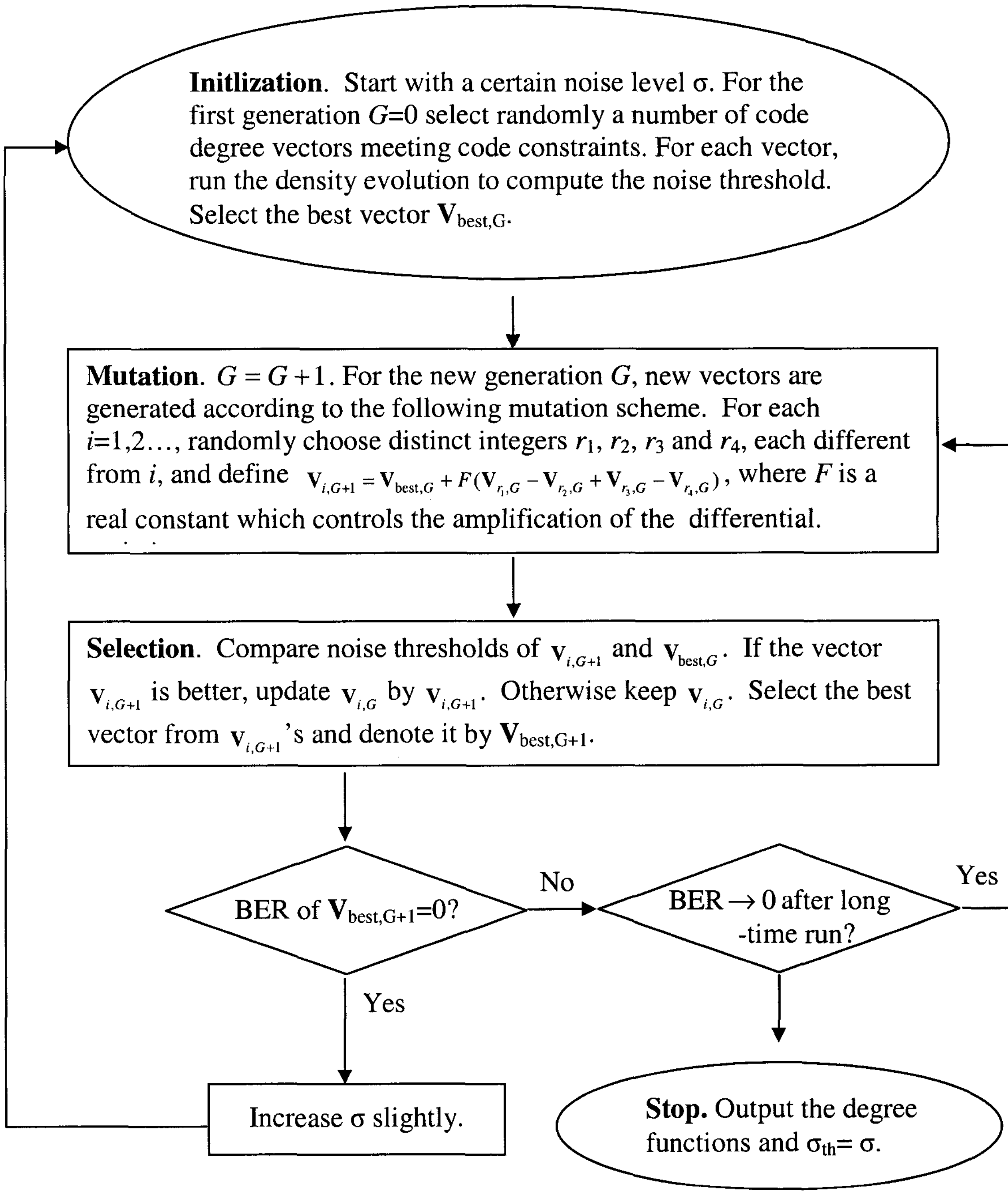

Initlization. Start with a certain noise level σ. For the first generation G=0 select randomly a number of code degree vectors meeting code constraints. For each vector, run the density evolution to compute the noise threshold. Select the best vector $\mathbf{V}_{\text{best,G}}$.
Mutation. $G = G+1$. For the new generation G, new vectors are generated according to the following mutation scheme. For each i=1,2..., randomly choose distinct integers r_1, r_2, r_3 and r_4, each different from i, and define $v_{i,G+1} = v_{\text{best},G} + F(v_{r_1,G} - v_{r_2,G} + v_{r_3,G} - v_{r_4,G})$, where F is a real constant which controls the amplification of the differential.
Selection. Compare noise thresholds of $v_{i,G+1}$ and $v_{\text{best},G}$. If the vector $v_{i,G+1}$ is better, update $v_{i,G}$ by $v_{i,G+1}$. Otherwise keep $v_{i,G}$. Select the best vector from $v_{i,G+1}$'s and denote it by $\mathbf{V}_{\text{best,G+1}}$.
BER of $\mathbf{V}_{\text{best,G+1}}$=0?
No
BER $\rightarrow$ 0 after long -time run?
Yes
Yes
Increase σ slightly.
Stop. Output the degree functions and $\sigma_{\text{th}} = \sigma$.

Fig. 4.6. Searching for good LDPC codes using density evolution.

process in [45] is summarized in the diagram in Fig. 4.6. This process can be used on any channel, including AWGN, fading, and PR channels.

However, many variations can be implemented in the mutation step. The one in Fig. 4.6 is just an example, where the variable and check degree functions are adjusted simultaneously. Other implementations are also possible. For example, the check degrees can be restricted to just a few values, or even fixed, in [21] and [46], thus do not need to be updated. Another example is [47], in which the variable and check degree functions are updated in a serial manner, i.e., the variable and check degree functions are updated alternately. Such algorithms have been shown very effective to design good LDPC codes for a number of channels. These include [21], [44] for AWGN and other binary-input symmetric-output memoryless channels, [45] for fading channels, [47] and [48] for PR channels.

Using these techniques, a number of irregular codes for PR channels have been designed in [47] and [48]. Since coset codes are considered in [47], i.e., $\mathbf{H} \cdot \mathbf{x} = \mathbf{c}$, where $\mathbf{c}$ can any binary vector of length K, codes of any rate can be designed. As an example, two codes of rate 0.7 are given in [47] for the dicode and the EPR4 channels respectively, and their noise thresholds computed. In [48], only high-rate codes are considered, i.e., $\mathbf{H} \cdot \mathbf{x} = \mathbf{0}$, and the rate close to 1.0. Several finite-length codes for the $1/1 \oplus D^2$-precoded EPR4 channel and the $1/1 \oplus D$-precoded ME^{2}PR4 channel are designed and simulated.

One issue hidden behind these two references is how to design finite-length practical codes using density evolution. As we know, density evolution can only search for good degree functions for asymptotic and ensemble of codes. It is still an open problem how to construct a particular code. Another issue is how to choose the parameters for the density evolution, namely, the minimum and maximum degrees of the variable and check nodes. This can substantially affect the searching results, and the possibility to construct finite-length codes without short cycles.

Two codes designed in [48] are used here to show the effectiveness of density evolution. These two codes are both (4835, 483), of rate $R \approx 0.9$. The first one is named code5 in [48] for the $1/1 \oplus D^2$-precoded EPR4 channel. The other one is named code7 in [48] for the $1/1 \oplus D$-precoded ME^{2}PR4 channel. Degree functions of these two codes are listed in Table 4.2.

Table 4.2. Degree functions of the code5 and code7.

	Code5		Code7	
i	λ_i	ρ_i	λ_i	ρ_i
3	0.7657		0.6980	
5	0.2343		0.3020	
33		0.9749		
34		0.0003		
37		0.0005		0.9991
39		0.0225		
46		0.0008		
51		0.0008		
59		0.0002		
78				0.0009

These codes are simulated on the turbo-equalized EPR4 and ME^{2}PR4 channels in [48]. Simulation results show that they are 0.5 and 0.3 dB, respectively, better than a random regular code of column weight $w_c = 3$ at BER=10^{-4}. These codes are simulated here, as shown in Fig. 4.7, but on the non-turbo-equalized ideal EPR4 and ME^{2}PR4 channels with AWGN.

The superiority of these two irregular codes is clearly observed in Fig. 4.7. Explicitly, code5 and code7 are both 0.12 dB better than the regular code of column weight $w_c = 3$ at BER=10^{-6}. Note that, the coding gains in Fig. 4.7 are a little bit smaller than those in [48]. This is because the regular codes used here and in [48] are different, and turbo equalization is used in [48].

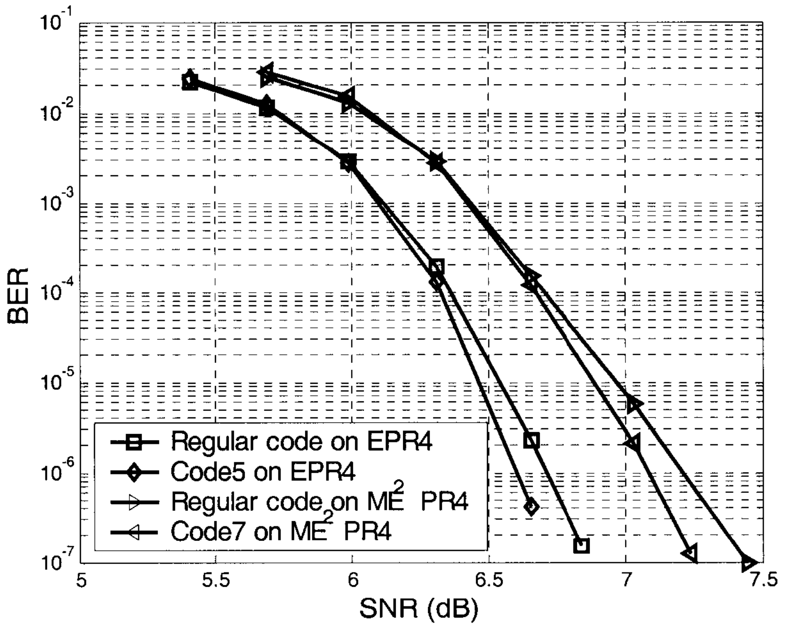


Fig. 4.7. Performance of code5 and code7 on ideal EPR4 and ME2PR4 channels.

Chapter 5

Erasures

Erasure is a common physical impairment in magnetic recording systems and must be handled appropriately. Otherwise, the system performance degrades dramatically. To address this issue we need to characterize the effects of these impairments on system performance and develop new signal processing and coding techniques to mitigate them.

One issue is that the erasure, whether caused by TA or MD, needs to be detected first. The need for erasure detection will be justified using noise threshold analysis. Only if erasure detection can be accomplished successfully, will channel detection and ECC techniques be effective in recovering the user information.

5.1 Erasure Model

Erasures in this paper refer to the sudden loss (including TA) or fading of the signal during read back, which we call full and partial erasures, respectively. Due to the nature of the erasure sources in magnetic recording, the bits in erasure are assumed contiguous. We further assume a rectangular window for the erasures, and use two parameters to model them: η is used to represent the fading depth, ranging from zero (no erasure) to one (full erasure), and the erasure length L_e is the number of bits in erasure out of all N bits in the codeword. Shown in Fig. 5.1 is the erasure model used in this work. We

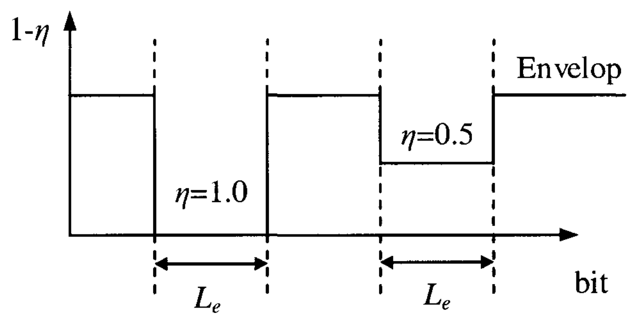


Fig. 5.1 Rectangular erasure model characterized by depth η and length L_e.

further assume that erasures have random starting locations and are present in every sector. Note that this erasure model has been also used in [27], [49], [50].

The basic way of handling detected erasures is erasure insertion, i.e., to clear their reliability information in the first iteration of the channel detection,

$$\Lambda_j = 0,\ j_e \le j \le j_e + L_e - 1, \tag{5.1}$$

where j_e and L_e are the starting location and length of the erasures. If two or more iterations of channel detection are used, e.g., iterative channel detection and ECC decoding, this clearing operation is only used in the first iteration, while in the other iterations, the extrinsic information of bits in erasure is not fed back for use in the next iteration. This erasure insertion has been shown very effective in [49]-[51] because it prevents the unreliability information of erasures from propagating to other bits and thus improves the system performance. In other words, an erasure detection algorithm is highly desirable in a practical system

Our goal in this chapter is to find out what erasures in terms of η and L_e should be detected, how to detect them, and furthermore, how to correct them.

5.2 BCJR-Once Information Rate Bound

The techniques in Section 4.1 for computing the information rate of PR channels can be easily modified for PR channels with erasures. However, we choose to use the BCJR-once bound because the erasure insertion is carried out only in the first iteration of the channel detection. Another reason for doing this is that we are more interested in a non-turbo-equalized channel.

From Section 4.4.2, the BCJR-once bound can be computed as

$$C_{\text{BCJR-once}} = 1 - \frac{1}{N}\sum_j \mathcal{H}\left(\Pr(x_j = 1 \mid \mathbf{r})\right), \tag{5.2}$$

by simulation in a way similar to that for symmetrical information rate. When a section of erasures is inserted, the bits in erasure have $\Lambda_j = 0$ according to (5.1), i.e.,

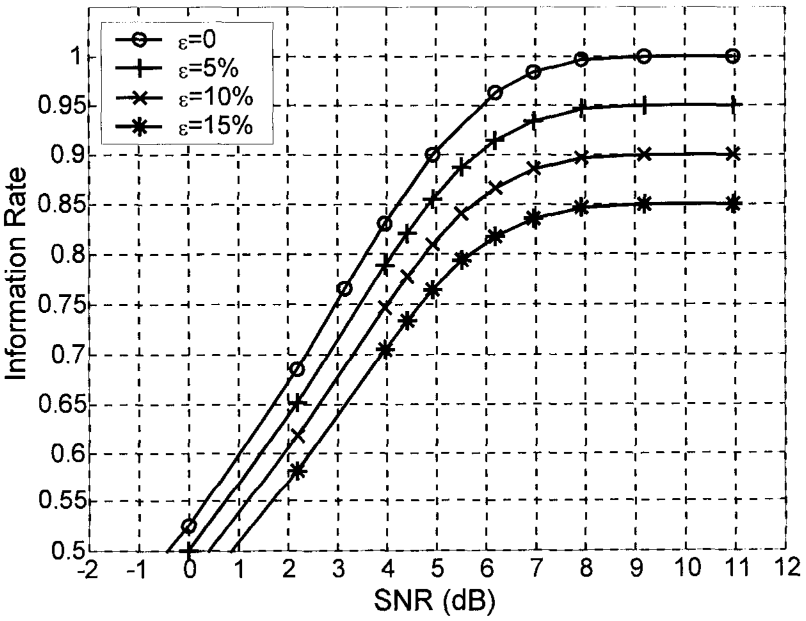


Fig. 5.2. BCJR-once bounds for the ideal EPR4 channel with AWGN and erasures.

$\Pr(x_j = 1 \mid \mathbf{r}) = \Pr(x_j = 0 \mid \mathbf{r}) = 0.5$, $j_e \le j \le j_e + L_e - 1$. Substituting into (5.2), the BCJR-once bound for erasure insertion can be easily computed as,

$$C^{\text{erasure}}_{\text{BCJR-once}} = 1 - \frac{1}{N}\left(\sum_{j<i_e, j>i_e+L_e} \mathcal{H}\left(\Pr(x_j = 1 \mid \mathbf{r})\right) + L_e \cdot h(0.5) \right). \tag{5.3}$$

Define $\varepsilon = L_e / N$, the probability of erasure if the erasures are scattered uniformly in the sector. Shown in Fig. 5.2 are the BCJR-once bounds for the ideal EPR4 channel with AWGN and erasures with $\varepsilon = 0, 5\%$, 10% and 15%.

5.3 Density Evolution for Erasures

Density evolution, a useful tool to analyze the performance of LDPC codes on PR channels, has been discussed in Chapter 4. However, the algorithm therein is valid only for standard PR channels with AWGN. In order to include erasures, some modifications must be made. In this chapter we first modify the algorithm, and then use it to analyze numerically LDPC codes on PR channels with erasures [51]. Turbo equalization is not considered. An ideal erasure detector is assumed if needed, i.e., the channel state information (CSI) about the erasures is known.

5.3.1 Modified Density Evolution

When erasures are present, the only change requested in the density evolution algorithm is the channel LLR, Λ_j,

$$\Lambda_j = \log \frac{P(x_j = +1 \mid \mathbf{r}, \eta, \varepsilon)}{P(x_j = 0 \mid \mathbf{r}, \eta, \varepsilon)}, \tag{5.4}$$

where the two parameters η and ε must be taken into consideration. Correspondingly, the pdf of the channel LLR, $f_{\Lambda,\mathrm{ch}}$, is modified to

$$f_{\Lambda,\mathrm{ch}} = H(\mathbf{r}_{\mathrm{i.u.d.}}, f_n, \eta, \varepsilon, e), \tag{5.5}$$

where e is a binary erasure flag. If e=1, which means that the signal fading depth is large and an erasure has been inserted, then $L_e = \varepsilon \cdot N$ bits of $\mathbf{r}$, are forced to zero, i.e., $r_j= n_j$ if e_j=1, and so are the LLRs, Λ_j=0 if e_j=1, and their pdfs become

$$f_{\Lambda_j} = \delta(\Lambda_j), \text{ if } e_j = 1 \tag{5.6}$$

for $j = j_e, j_e + 1, ..., j_e + L_e - 1$, and $\delta()$ is the delta function. The overall average $f_{\Lambda,\mathrm{ch}}$ is either (5.5) for a unflagged partial erasure or

$$f_{\Lambda,\mathrm{ch}} = (1-\varepsilon) \cdot H(\mathbf{r}_{\mathrm{i.u.d.}}, f_n, \eta, \varepsilon, e) + \varepsilon \cdot \delta(\Lambda), \tag{5.7}$$

for erasures with $e_j = 1$ for $j = j_e, j_e + 1, ..., j_e + L_e - 1$.

The $f_{\Lambda,\mathrm{ch}}$ in either (5.5) or (5.7) allows us to perform density evolution on PR channels with erasures, which can be straight-forwardly implemented as in Chapter 4. In other words, both the BER estimation and SNR threshold analysis can be carried out.

5.3.2 Characterizing Erasures

Two parameters to characterize erasure are the partial erasure fading depth η and the probability of erasure ε. Two values of these parameters, namely, the fading depth threshold η_{th} and the erasure correction capability $\varepsilon_{\max}$ can be determined using the density evolution.

For a partial erasure with probability ε, η_{th} is defined as,

$$\eta_{\text{th}} = \inf\{\eta, P_e(\varepsilon, SNR) < P_n(\varepsilon, \eta, SNR)\}, \tag{5.8}$$

where $P_e(\varepsilon, SNR)$ and $P_n(\varepsilon, \eta, SNR)$ denote the BERs for full and partial erasures at a given SNR. Note that η_{th} is a function of ε and the underlying code as well as the parameters of the MP decoder. From (5.8), all received signals with $\eta \geq \eta_{\text{th}}$ are determined unreliable and an erasure is flagged. This threshold can also be defined in terms of the SNR$_{\text{th}}$,

$$\eta_{\text{th}} = \inf\{\eta, SNR_{\text{th}}(\varepsilon, \text{erasure}) < SNR_{\text{th}}(\varepsilon, \eta)\}, \tag{5.9}$$

where $SNR_{\text{th}}(\varepsilon, \text{erasure})$ and $SNR_{\text{th}}(\varepsilon, \eta)$ are the SNR$_{\text{th}}$'s for the full and partial erasures, respectively.

As far as the parameter ε is concerned, $\varepsilon_{\max}$, is defined as the erasure correction capability of the LDPC code,

$$\varepsilon_{\max} = \sup(\varepsilon, P_e(\varepsilon, SNR) < P_0(SNR)), \tag{5.10}$$

where $P_0(SNR)$ is the required BER at a given SNR. Similarly to η_{th}, $\varepsilon_{\max}$ can be defined in terms of SNR$_{\text{th}}$ as well, equivalently in two forms,

$$\varepsilon_{\max} = \sup\{\varepsilon, SNR_{\text{th}}(\varepsilon, \text{erasure}) \text{ exists}\}, \tag{5.11}$$

$$\varepsilon_{\max} = \sup\left(\varepsilon, \lim_{SNR \to \infty} P_e(\varepsilon, SNR) \to 0\right), \tag{5.12}$$

due to the fact that $P_e \to 0$ for $SNR \geq SNR_{\text{th}}$.

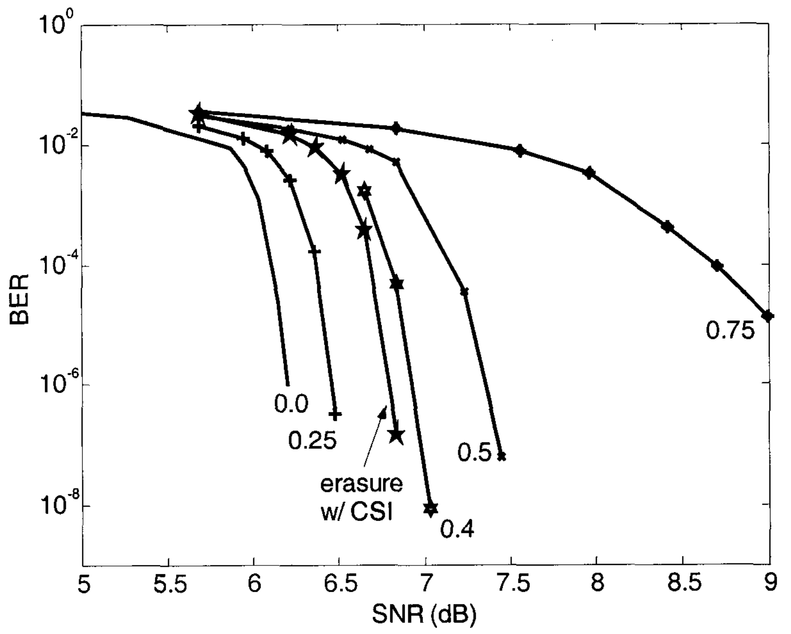
Note that the values of η_{th} and $\varepsilon_{\max}$ defined in terms of the SNR$_{\text{th}}$, and computed by density evolution are always over-estimated with respect to a finite-length code. Hence for a given practical system, a thorough investigation needs to be done to determine them reliably.

5.3.3 Numerical Results

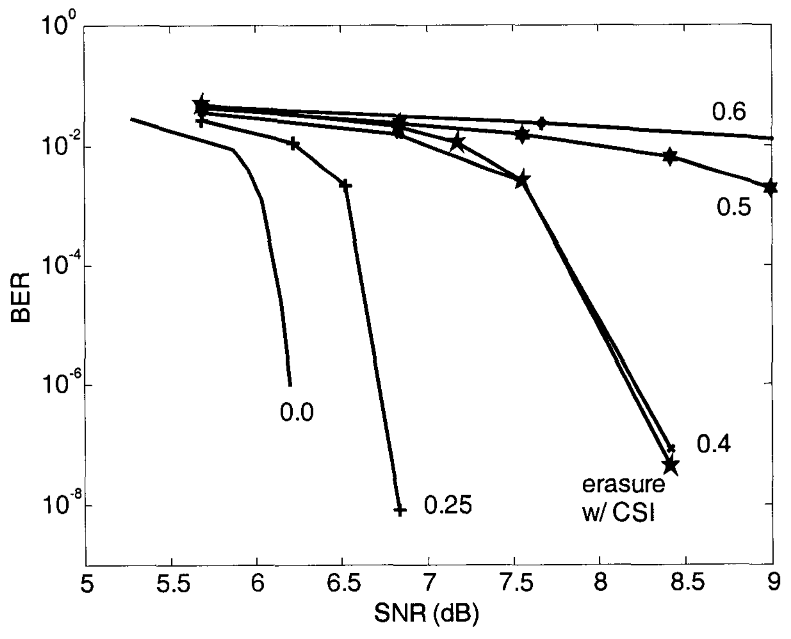
In this chapter we only consider regular codes (w_c, w_r) on the $1/1 \oplus D^2$-precoded EPR4 channel as an example. The extension to irregular codes and other PR channels is straightforward. The maximum number of iterations for the MP decoding of LDPC codes for the BER and SNR threshold evaluations is ten and fifty, respectively, if not specified otherwise.

Partial erasures without CSI are first considered. Intuitively, the fading depth should substantially affect the performance; the lower the fading depth, the better the system performance. For partial erasures with a large fading depth, the availability of CSI should be helpful.

First we estimate the BER using density evolution. The estimated BERs for an LDPC (3,30) coded system with $\varepsilon = 3\%$ and 6% (corresponding to 123 and 246 out of 4096) bits in partial erasure with different fading depths are shown in Fig. 5.3. In Fig. 5.3(a), the curves from left to right correspond to fading depths $\eta = 0$ (no fading), 0.25, 0.40, 0.50, and 0.75, except that the third curve from the left corresponds to a full erasure with CSI. The coding losses of the five curves, from left to right, compared to no fading, at BER=10^{-6}, are 0.25, 0.6, 0.7, 1.2 and larger than 4 dB. Similarly, the same curve sequence is observed in Fig. 5.3(b) for $\eta = 0$, 0.25, 0.40, 0.50, and 0.60, except that the third curve from the left corresponds to a full erasure with CSI but the coding losses are much larger. This shows that the system is quite sensitive to ε even at low fading depths.



(a)



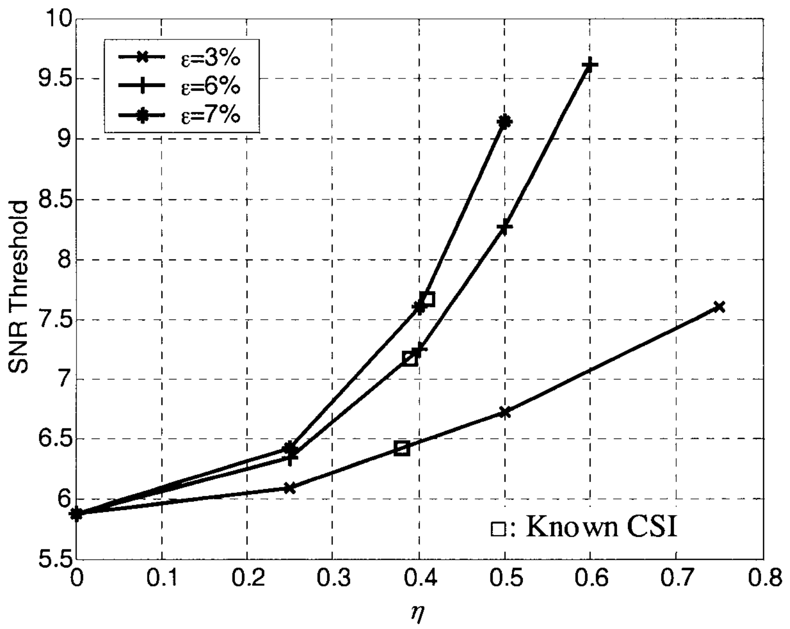
(b)

Fig. 5.3. BER estimates for LDPC code (3,30) on the precoded EPR4 channel with (a) $\varepsilon = 3\%$, (b) $\varepsilon = 6\%$ and η as a parameter.

From Fig. 5.3, we see that partial erasures with fading depth $\eta > 0.75$ and 0.5 for $\varepsilon = 3\%$ and 6% are quite detrimental to performance.

From both Fig. 5.3 (a) and (b), the BER for partial erasures with $\eta_{\rm th}$ =0.40 is slightly worse than that for a full erasure with CSI. This shows that if the erasure detector can detect partial erasures with η>0.40, the system performance will be improved. In other words, we have $\eta_{\rm th} \approx 0.40$ by the definition in (5.9) for SNR in the range of 6~7 dB. We also observe that $\eta_{\rm th}$ $(\varepsilon = 3\%)$ is slightly smaller than $\eta_{\rm th}$ $(\varepsilon = 6\%)$.

Note that the number of iterations in these evaluations is equal to ten. If this number is increased, say to fifty, the performance will be improved, especially for large fading



(b)

Fig. 5.4. SNR thresholds as a function of η for LDPC code (3,30) on the EPR4 channel with partial erasure of probability ε (unknown CSI).

depths. For example, the BER for $\varepsilon = 6\%$ and $\eta = 0.5$ drops from 10^{-3} to 10^{-7} when the number of iterations increases from ten to twelve. This indicates that η_{th} depends on the number of iterations of the MP decoding.

Next we use SNR threshold as the criterion. We compute SNR_{th} as a function of the fading depth η for partial erasures with $\varepsilon = 3\%$, 6% and 7%, as plotted in Fig. 5.4 with $e = 0$. Also plotted are the SNR_{th}'s for full erasures with CSI, denoted by "□" in Fig. 5.4. From this figure, we can observe a similar tendency of the coding loss as in the BER curves, but the absolute values are smaller. Using the definition of η_{th} in (17), we have $\eta_{\text{th}} \approx 0.385$, 0.390 and 0.410 for $\varepsilon = 3\%$, 6% and 7%.

Next we consider full erasures with CSI. As discussed in [49], the recoverable erasure length of a random regular LDPC code with check node degree w_r is about

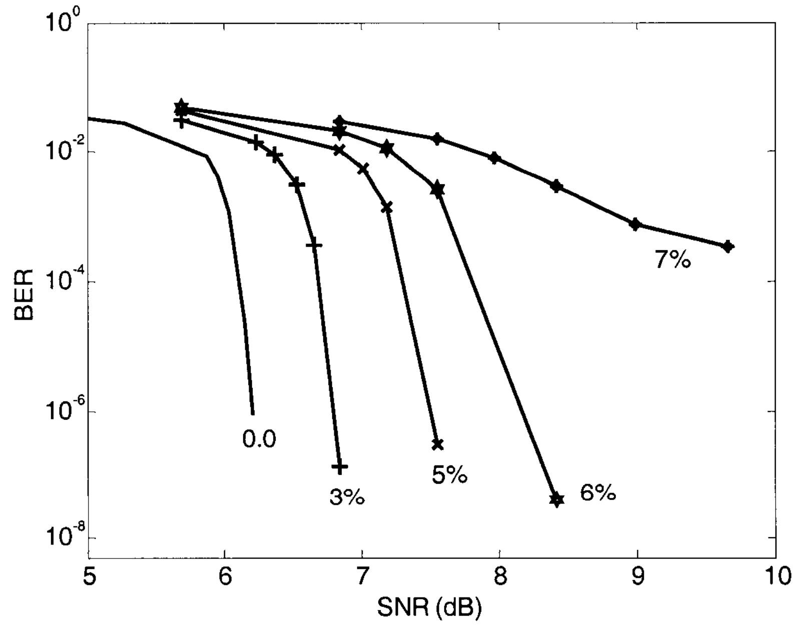


Fig. 5.5. BER estimate for LDPC code (3,30) on the precoded EPR4 channel with full erasures with CSI, with ε as a parameter.

$2N / w_r$. Equivalently, the erasure recovery capability is $\varepsilon_{\max} = (2 / d_c \times 100)\%$. However, this is only a rough estimate, probably valid for regular codes with $w_c \geq 3$. For the code (3,30), this number is $\varepsilon_{\max} = 6.67\%$. So if it is a valid estimate, all erasures with $\varepsilon < 6.67\%$ should be recoverable.

Let us first estimate the BERs, shown in Fig. 5.5 for the code (3,30) with $\varepsilon = 3\%$ through 7%. The curve for no erasure is also plotted as a reference. The coding losses for $\varepsilon = 3\%$, 5% and 6% at BER=10^{-6} are 0.6, 1.3, and 2.0 dB. More importantly, the coding loss for $\varepsilon = 7\%$ is quite large, showing that erasures with $\varepsilon = 7\% > \varepsilon_{\max}$ are not recoverable. Note again that the maximum number of iterations for this analysis is ten. If it is increased to fifty, a larger ε, perhaps up to 7%, may be recoverable, i.e., $\varepsilon_{\max}$ may be slightly larger than 7%.

The erasure recovery capability can also be evaluated in terms of the SNR threshold. The SNR_{th} as a function of the fading depth η and erasure fraction ε is shown in Fig. 5.6. Using a step of 1% for ε in the numerical computation, SNR_{th}'s for $\varepsilon = 8\%$ and 7% are computable while for $\varepsilon = 9\%$ and 8% they are not, respectively, for the codes (3,30) and (4,40). This shows that $\varepsilon_{\max} \approx 8\%$ and 7% for these two codes, respectively, which are larger than those obtained using the BER criterion. This can be explained by the different maximum number of iterations used by the MP decoder, ten and fifty for BER and SNR threshold analysis, respectively. If we increase this number for the BER estimate to fifty, we should get similar values as for the SNR_{th}'s. However, we do not use such large number of iterations, because ten iterations is a more meaningful value for evaluating the BER of practical systems.

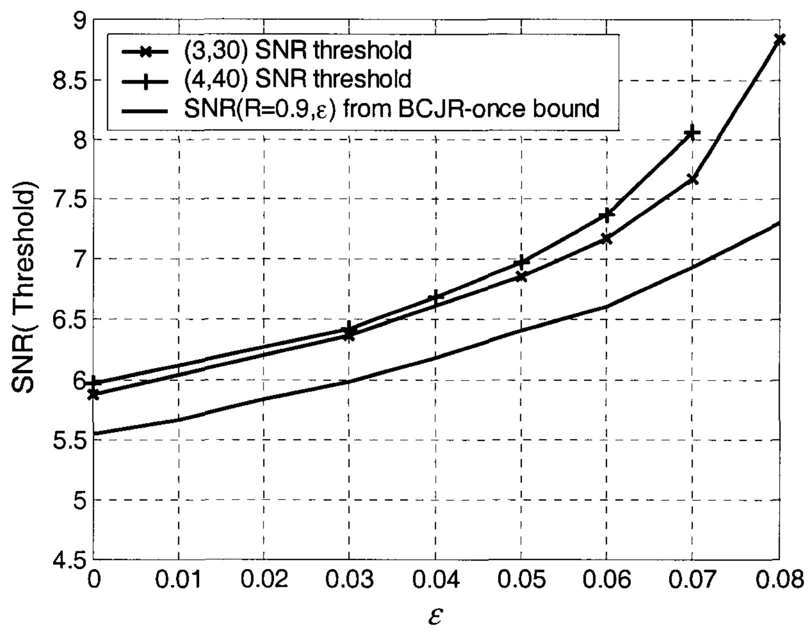


Fig. 5.6. SNR thresholds for LDPC codes (3,30) and (4,40) on the precoded EPR4 channel with known full erasures.

Since $\mathrm{SNR_{th}}$'s for the code (4,40) are always larger than for code (3,30), we conclude that the code (3,30) is asymptotically better than the code (4,40).

Also shown in Fig. 5.6 are the SNRs corresponding to the BCJR-once bound at rate $R = 0.9$, i.e., $\mathrm{SNR}= C^{-1}_{\text{BCJR-once}}(R,\varepsilon)$. Apparently, this SNR curve is quite close to the $\mathrm{SNR_{th}}$ curve for the code (3,30). This fact shows that the code (3,30) is very good for erasure correction. Although we have tried to use density evolution to search for better codes, we did not yet succeed.

5.3.4 Validation of BER Estimation

As we stated in Chapter 4, if the code length is large enough, the actual BER approaches the density evolution BER estimate. This has been validated in [42] and in

Chapter 4 for the case of no erasures. We repeat this work here for the case of erasures and compare the simulation results with the estimated ones.

First we do simulations using similar configurations of Fig. 5.3(a), i.e., ε=3% in view of the signal fading threshold. However, only BER curves for $\eta = 0$ (no fading), $\eta = 0.4$ without CSI, $\eta = 1.0$ with CSI are plotted in Fig. 5.7. Two finite-length codes are used, one of length $N = 10^5$, the other of $N = 5\times10^3$. Both codes are constructed randomly and are free of short cycles (of length two and four).

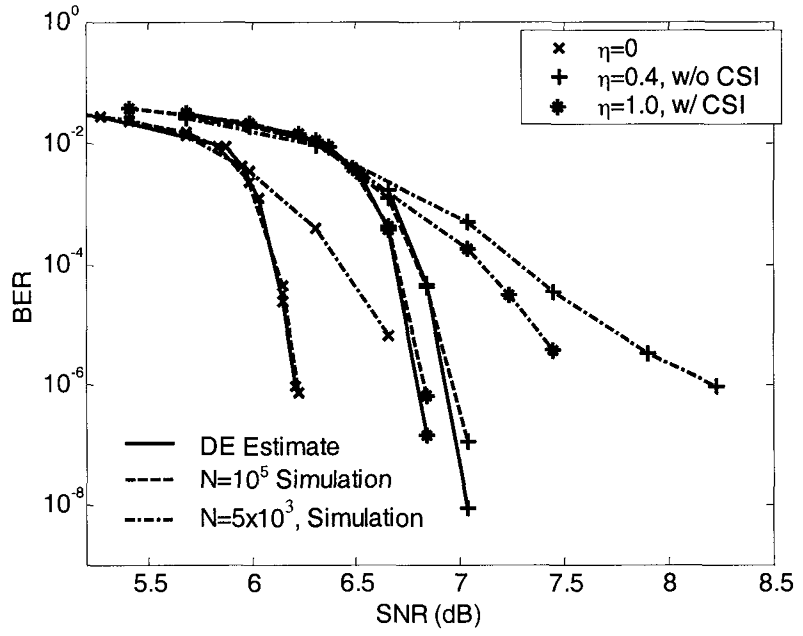


Fig. 5.7. Comparison of BER estimates using density evolution and simulation for LDPC (3,30) coded EPR4 channel with partial erasures.

It is not surprising that the simulated BERs for the code with N=10^5 agree very well with the density evolution ones. This is consistent with [42] where density evolution was shown to provide good BER estimates for codes with N=10^5 and N=10^6. On the other hand, the simulated BERs for the code with N=5x10^3 deviate from the density evolution estimates, because the length is too small, and the estimate of the tails of the channel LLR pdfs are not accurate.

However, these observations do not affect the conclusion in Section 5.3.3, namely, that η_{th} is a little bit less than 0.4. This illustrates that density evolution results can be used as a good reference to estimate the signal fading threshold.

Next we do simulations as in Fig. 5.5 using the same two codes. The BERs for ε=5%, 6% and 7% are shown in Fig. 5.8. Again, the simulated BERs for the code with N=10^5 agree well with the density evolution estimates for ε=5% and 6%, but not 7%. In fact, the simulated BERs for 7% are one order better than the density evolution estimates at SNR of 10~11 dB. This implies that this particular code is better than the ensemble for

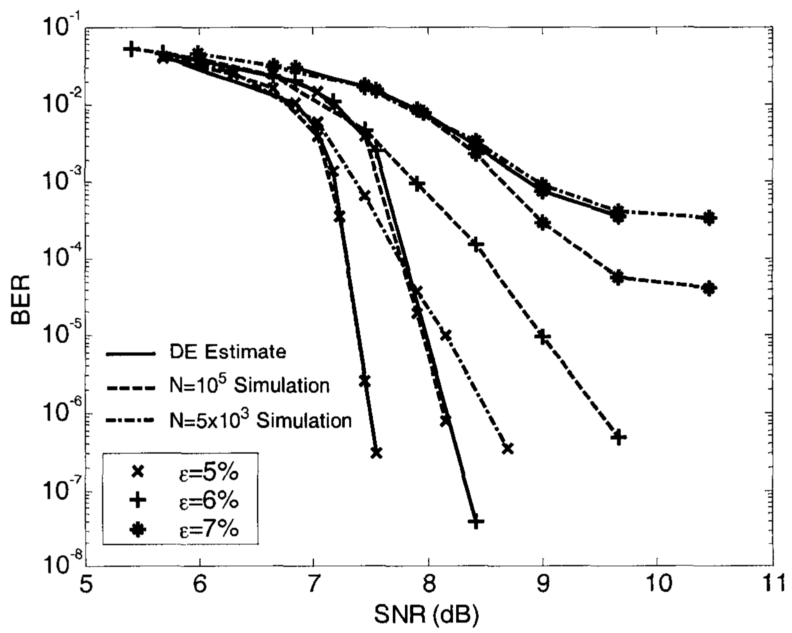


Fig. 5.8. Comparison of BER estimates using density evolution and simulation for LDPC (3,30) coded EPR4 channel with full erasures and CSI.

large ε. On the other hand, the simulated BERs for the code with $N=5\times10^3$ deviate from the density evolution estimates, as expected.

The most important thing is that all the BER curves for $\varepsilon=7\%$ show up error floors, which means that their BER performances cannot meet the requirement $P_0(SNR)$ by the definition of $\varepsilon_{\max}$ in (5.10). However, the BER curves for $\varepsilon=6\%$ do not exhibit such error floors, at least at the simulated BER levels. This validates the statement that $\varepsilon_{\max}<7\%$. This illustrates that the density evolution results can be used to estimate the erasure correction capability.

One byproduct of Figs. 5.7 and 5.8 is the validation of BER estimates using density evolution for no erasure. This has been done in Chapter 4 and is redone here, but with a code free of cycles of length four. It is observed that when cycles of length four are removed, the simulation results are almost identical to the density evolution estimates, which confirms the discussion in Section 4.5.2.

5.4 Erasure Detection Algorithms

Until now we assumed an ideal erasure detector. However, this assumption is not realizable. In this section we present practical erasure detectors for magnetic recording systems.

The erasure detection algorithm in [52] uses the RLL k-constraint violation to flag erasures for a (3, 16) MTR-coded MRC equalized to an ME^{2}PR4 target. Specifically, in the first decoding iteration, the max-log-MAP implementation of the BCJR channel detector makes tentative decisions from the sign of the BCJR LLRs, finds violations of the k-constraint, and generates an erasure flag. This algorithm was shown to be effective for the given system by simulations for deep erasures, whose flagging as full erasures improves performance. However, this algorithm is not general, e.g., it does not work for the ME^{2}PR4 channel without a precoder, where neither the *d*- nor the *k*-constraint may be violated when erasures occur. Further details on this example are given in the next section.

Another problem with the algorithm in [52] is that it cannot detect partial erasures effectively. For example, assuming probability of erasure ε=3%, partial erasures of fading depth η>0.4 should be detected and flagged as erasures [51], however, the algorithm in [52] fails to do so.

The new algorithm in [53] overcomes this difficulty. Observing that the BCJR LLRs are attenuated due to partial erasures, [53] suggests to use the amplitude in addition to the sign of the LLRs to make the tentative decisions. By thresholding the amplitude of the LLRs, partial erasures can be detected effectively. Comparison shows the superiority of the algorithm in [53] to the one in [52]. However, the algorithm in [53] requires the use of a particular precoder for a given PR channel, which makes it unusable in practical systems without precoders. This is an issue to be addressed in this chapter.

5.4.1 BCJR Analysis for Full Erasures

In this chapter we analyze the BCJR algorithm for detection of full erasures on PR channels with RLL (0, *k*) or (0, $G|I$) code. The PR targets considered are the EPR4 and ME^{2}PR4 for longitudinal recording.

For erasures of depth η=1.0 and length L_e, the read back signal is $r_j = n_j$, and the BCJR branch metric becomes

$$\gamma_j(s_{j-1},s_j) = -(n_j - z_j)^2 / 2\sigma^2 + \log \Pr(x_j), \tag{5.13}$$

for $j = j_e, j_e+1, \cdots, j_e + L_e - 1$. For every pair of branches $s_{j-1} \xrightarrow{x_j / z_j^{(1,2)}} s_j^{(1,2)}$, the one with smaller distance $|z_j|$ usually has larger $\gamma_j(s_{j-1},s_j)$. Of particular interest are the branches with $z_j = 0$ because signals in erasure are equivalent to them. The probability of $\gamma_j(s_{j-1},s_j,z_j^{(1)}=0 \mid r_j)$ being less than $\gamma_j(s_{j-1},s_j,z_j^{(2)} \neq 0 \mid r_j)$ for the given PR channel is

$$\Pr\left(n_j^2 > (n_j - z_j^{(2)})^2\right) = \Pr(n_j > |z_j^{(2)}|/2) = Q\left(\sqrt{(z_j^{(2)})^2 SNR/2}\right), \tag{5.14}$$

where $SNR = \sum_i f_i^2 / 2\sigma^2$ is defined in Chapter 1. For example, in the normalized EPR4 channel, $|z_j^{(2)}| \geq 1$, assuming $SNR = 15$ dB, $Q(\sqrt{(z_j^{(2)})^2 SNR/2}) \leq 3.5\times10^{-5}$. This fact makes the braches with $z_j = 0$ play a major role in the detection of full erasures.

Let us first consider the non-precoded EPR4 channel, whose trellis is shown in Fig. 2.1 (a). Denote $S_A = \{S_0, S_1, \cdots, S_7\}$. Consider the four branches with $z_j = 0$: $S_0 \xrightarrow{0/0} S_0$, $S_2 \xrightarrow{0/0} S_5$, $S_5 \xrightarrow{1/0} S_2$ and $S_7 \xrightarrow{1/0} S_7$. It is easy to see that if a state $s_{j_e-1} \in \{S_0, S_2, S_5, S_7\}$ has the largest α_{j_e-1}, then from (2.17) a state $s_{j_e} \in \{S_0, S_2, S_5, S_7\}$ continues to have the largest α_j for $j = j_e, j_e+1, \cdots,\ j_e + L_e - 1$ along the above four braches. For example, assume $\alpha_{j_e-1}(S_2) > \alpha_{j_e-1}(S_A \backslash S_2)$, then γ_j's along the four branches are equal, while greater than all other γ_j's with probability $Q(\sqrt{SNR/2}) << 1$ by (5.14) given high enough SNR. We call such states $\{S_0, S_2, S_5, S_7\}$

stable states because they have the largest α_j's for most of the $j_e \le j \le j_e + L_e - 1$.

Consider the case when a state $s_{j_e - 1} \notin \{S_0, S_2, S_5, S_7\}$ has the largest $\alpha_{j_e - 1}$. Let us look at the α recursion. At each step of this iteration, a negative value $-(n_j - z_j)^2 / 2\sigma^2$ for some r_j is added to α. For the paths $S_0 \to S_0 \to \cdots \to S_0$, $S_7 \to S_7 \to \cdots \to S_7$, and $(S_2 \to) S_5 \to S_2 \to \cdots \to S_5$ beginning at the $j_e - 1$-th bit, this value is $-n_j^2 / 2\sigma^2$, and its accumulation is

$$\Gamma_0 = -\sum_i n_i^2 / 2\sigma^2 . \tag{5.15}$$

However, for other paths with most of $z_j \neq 0$, the accumulation is

$$\Gamma_1 = \sum_i (n_i - z_i)^2 / 2\sigma^2 . \tag{5.16}$$

Since it is very likely that $n_j^2 < (n_j - z_j)^2$, for $z_j \neq 0$ at high SNR, especially for those z_j's with large absolute values, it is highly probable that $\Gamma_0 < \Gamma_1$, i.e.,

$$\Pr(\Gamma_0 < \Gamma_1) \approx 1.0 . \tag{5.17}$$

Moreover, the longer the recursion goes, the larger will be the difference between Γ_0 and Γ_1. This difference will finally cancel the difference between $\alpha_{j_e - 1}(s_{j_e - 1})$ and $\alpha_{j_e - 1}(\{S_0, S_2, S_5, S_7\})$, and one of the states in $\{S_0, S_2, S_5, S_7\}$ begins to have the largest α_j for some $j = j_e + l_e$, where l_e is the number of steps of α recursions needed for this to occur. From here on, $\{S_0, S_2, S_5, S_7\}$ are stable states for $j_e + l_e \le j \le j_e + L_e - 1$. Although we cannot provide a rigorous proof for the above argument, it is supported adequately by simulation results.

Similar arguments apply to the β recursions, leading to the same stable states $\{S_0, S_2, S_5, S_7\}$ for $j_e + L_e - l_e' \geq j \geq j_e$, where l_e' is a small positive integer.

Consider the portion of erasure where $\{S_0, S_2, S_5, S_7\}$ are stable states for both α and β recursions, i.e., for $j = j_e + l_e, j_e + l_e + 1, \cdots, j_e + L_e - l_e'$. Then,

- If $S_0 \to S_0 \to \cdots \to S_0$ is the selected path with respect to both α and β recursions, then $\Lambda_j < 0$ continuously.
- If $S_7 \to S_7 \to \cdots \to S_7$ is the selected path with respect to both α and β recursions, then $\Lambda_j > 0$ continuously.
- If $(S_2 \to) S_5 \to S_2 \to \cdots \to S_5$ is the selected path with respect to both α and β recursions, then $\Lambda_j < 0$ and $\Lambda_j > 0$ alternately

These three cases are *exactly* the same as normal signals with zero-output sequence. If hard decisions are made tentatively at the BCJR output as,

$$\hat{x}_j = \begin{cases} +1, \text{if } \Lambda_j > 0 \\ 0, \text{if } \Lambda_j \leq 0 \end{cases}, \tag{5.18}$$

then the decided sequences in the first two cases are $00\cdots00$ and $11\cdots11$, respectively, both of which can be detected by monitoring the RLL *k*-constraint, if the length of runs of 0's or 1's is larger than RLL(*k*). However, the detected sequence in the third case is $0101\cdots01$, which does not violate either the $d = 0$ and the *k*- constraint, and cannot be detected by the RLL(*d*=0, *k*) code.

Now consider the $1/1 \oplus D^2$-precoded EPR4 channel. Doing the same α and β

recursions focusing on erasures, we can obtain the following cases of selected path,

- If $S_0 \to S_0 \to \cdots \to S_0$, $S_7 \to S_7 \to \cdots \to S_7$, or $(S_2 \to)S_5 \to S_2 \to \cdots \to S_5$ is the selected path with respect to both α and β recursions, then $\Lambda_j < 0$ continuously.

In all these three cases, the tentative decided sequences are $00\cdots 00$, which can be detected by the RLL *k*-constraint, if the length of runs of 0's is larger than RLL(*k*).

Other than above three combinations of paths with respect to α and β recursions, there are also other possibilities. Here we give only one example for the non-precoded EPR4 channel, others can be analyzed similarly. Let $S_0 \to S_0 \to \cdots \to S_0$ and $S_7 \to S_7 \to \cdots \to S_7$ be the selected paths with respect to α and β iterations, respectively. In this case, since

$$\beta_j(S_7) - \beta_j(S_0) \approx \beta_{j+L_e-l_e'}(S_7) - \beta_{j+L_e-l_e'}(S_0), \tag{5.19}$$

$$\alpha_{j-1}(S_0) - \alpha_{j-1}(S_7) \approx \alpha_{j_e+l_e}(S_0) - \alpha_{j_e+l_e}(S_7), \tag{5.20}$$

it is very likely (with probability close to one) that

$$\Lambda_j = \Lambda_j(S_7, S_7) - \Lambda_j(S_0, S_0) = \left(\beta_j(S_7) - \beta_j(S_0)\right) - \left(\sigma_{j-1}(S_0) - \alpha_{j-1}(S_7)\right). \tag{5.21}$$

Therefore, the signs of Λ_j 's are preserved and determined by the relativity of $\beta_{j+L_e-l_e'}(S_7) - \beta_{j+L_e-l_e'}(S_0)$ and $\alpha_{j_e+l_e}(S_0) - \alpha_{j_e+l_e}(S_7)$, and the tentative decided sequence is $00\cdots 00$ or $11\cdots 11$, which can be detected by the RLL *k*-constraint.

5.4.2 BCJR Analysis for Partial Erasures

The BCJR algorithm analysis for detection of partial erasures is not easy, because we do not have a particular received signal sequence, like the zero-output sequence for full erasures. Therefore we only give some intuitive explanations and use a numerical method. The EPR4 channel is used as an example. Partial erasures of typical depth $\eta \approx 0.5$ are discussed particularly, because $\eta \approx 1.0\,(>0.7)$ and $\eta \approx 0.0\,(<0.3)$ can be viewed as full erasures and no erasures, approximately.

The first fact we note is that unlike for full erasures, there are no stable states for partial erasures. This can be analyzed similarly as for full erasures and is frequently observed in simulations. As a result, the amplitudes of the LLRs, i.e., $|\Lambda_j|$, $j = j_e, j_e+1, \cdots, j_e+L_e-1$, are small in contrast to those subject to no erasures.

Another reason why $|\Lambda_j|$'s are small is that it is almost impossible to select a path throughout these partial erasures. Although the BCJR algorithm is not a sequence-based channel detection algorithm, usually a sequence can still be selected given high enough

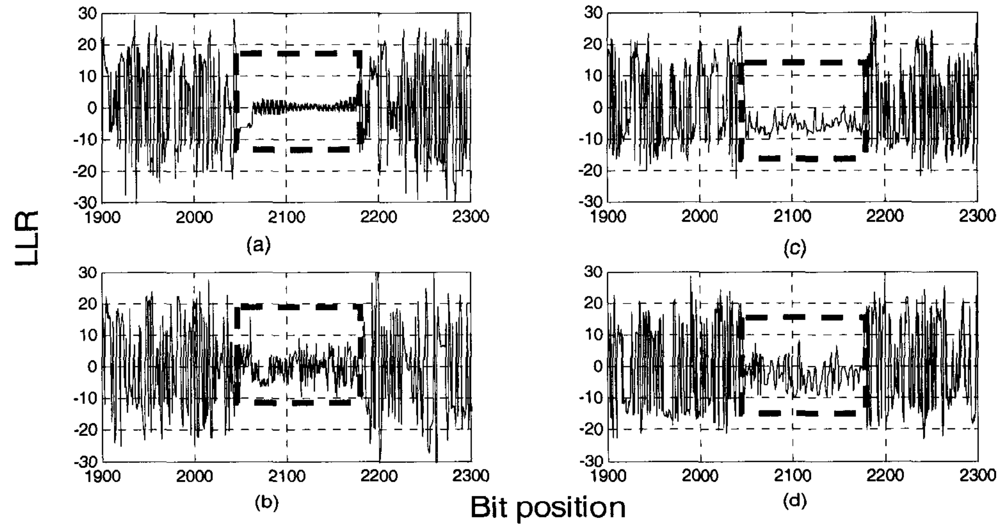


Fig. 5.9 LLRs for the EPR4 channel at SNR=7 dB. (a) non-precoded, $\eta = 1.0$, (b) non-precoded, $\eta = 0.5$ (c) $1/1\oplus D^2$-precoded, $\eta = 1.0$, (d) $1/1\oplus D^2$-precoded, $\eta = 0.5$. The erasures are indicated by a dashed-line box.

SNR. However, in the case of partial erasure, the $(1-\eta)$ scaling factor due to erasure disturbs the path of the original transmitted sequence, and makes it possible for many paths to have comparable likelihoods.

Let us consider both the non-precoded and the $1/1 \oplus D^2$-precoded EPR4 channels with AWGN and SNR = 7 dB. A set of simulations are conducted for $\eta = 1.0$ (full erasure) and $\eta = 0.5$ (partial erasure). Shown in Fig. 5.10 are the LLRs, from which the following facts are observed:

- The amplitudes of LLRs for partial erasures of $\eta = 0.5$ are smaller relative to those for non-erasure, whether the precoder $1/1 \oplus D^2$ is used or not. This fact can be used for detection of partial erasures.
- For full erasures on the $1/1 \oplus D^2$-precoded EPR4 channel, $\Lambda_j < 0$ continuously for erasures, which agrees with the discussion in Section 5.5.1. The amplitudes of Λ_j's are smaller than those for non-erasure.
- For full erasures on the non-precoded EPR4 channel, $\Lambda_j < 0$ and $\Lambda_j > 0$ alternatively for erasures, which also agrees with the discussion in Section 5.5.1. The amplitudes of Λ_j's are very small, $\Lambda_j \approx 0$.

These facts indicate that in addition to the sign of the LLR, its amplitude can be further used to detect erasures. This is particularly useful for detection of partial erasures because no other effective way is available to accomplish this task.

5.4.3 Erasure Detection Algorithms

Based on the previous BCJR algorithm analysis for both full and partial erasures,

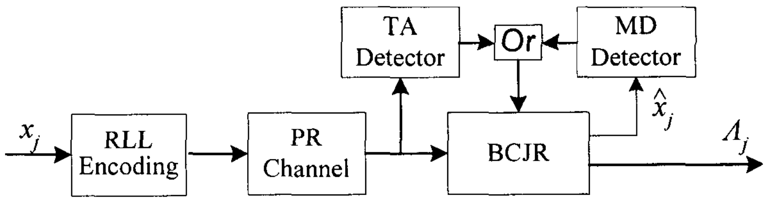


Fig. 5.10. Diagram of MRC with erasure detection. A DC-thresholding-based TA detector is also shown.

several erasure detection algorithms are suggested. An MRC channel model using erasure detection is shown in Fig. 5.10. Since the detection of TA is trivial, the erasure detection algorithms presented below are mainly for MD.

Algorithm A is an extended version of the one in [52]. In Section 5.5.1, we have shown that in the non-precoded EPR4 channel one case of tentative decided sequences at the BCJR output is $1010\cdots10$ and cannot be detected by the RLL *k*-constraint. To overcome this difficulty, the precoder $1/1\oplus D^2$ can be used. Another way is to use an RLL with one enhanced interleaved constraint, i.e., to use an RLL(d,*G*|*I*) code. Specifically, the tentative decision is made according to (5.9), and then both the *G*- and *I*-constraints are checked. If any violation is found, a section of erasures is declared. In other words,

$$\text{Erasures are declared if } \begin{cases} \text{runs of } \hat{x}_j = +1\,(\text{or } 0) > G' \\ \text{runs of interleaved } \hat{x}_j = +1\,(\text{or } 0) > I' \end{cases}, \tag{5.22}$$

where $G' \geq G$ and $I' \geq I$ are two constants.

Algorithm B use amplitude in addition to sign of the LLR to make tentative decisions, i.e., in addition to (5.18), the following decision is made,

$$\hat{x}'_j = u \;\text{ if } |\Lambda_j| < \Lambda_{th}, \tag{5.23}$$

where Λ_{th} is an experimentally determined threshold, and u represents an unreliable decision. An empirical value for Λ_{th} is $\Lambda_{\text{th}} = \text{mean}(\Lambda) - \sqrt{\text{var}(\Lambda)/3}$ where Λ is the LLR value in the absence of erasure. Any violation of the RLL constraints triggers a declaration of a section of erasures. In other words:

$$\text{Erasures are declared if} \begin{cases} \text{runs of } \hat{x}_j = +1\,(\text{or } 0) > G' \\ \text{runs of interleaved } \hat{x}_j = +1\,(\text{or } 0) > I' \\ \text{runs of } \hat{x}'_j = u > G' \end{cases}. \tag{5.24}$$

Note that both G- and I-constraints are used in (5.24). If the I-constraint is not used, i.e., an RLL(d,k) code is used, then the second condition in (5.24) disappears.

Algorithm C is a reduced version of Algorithm B. In this algorithm only the amplitude of LLR is used to make the tentative decision, as in (5.23), and erasures are declared if runs of $\hat{x}'_j = u > G'$, i.e., the third condition in (5.24).

This algorithm is particularly useful for PMRC targets, e.g., the ones in Table 1.2. The first two conditions in (5.24) can be dropped for two reasons. First, the tentative sequence patterns $00\cdots00$, $11\cdots11$ or $1010\cdots10$ very rarely occur in the DC-full or the DC-mix channels. Although they can happen in the DC-free channel, it has been shown in Chapter 2 that DC-free targets will not be used in practice. Secondly, closer observation reveals that the amplitude of the LLR for these sequences, if they can happen, is very small. This fact makes Algorithm C possible.

These three algorithms are checked in uncoded LMRC or PMRC systems by observing the statistics of detected erasures. First, Algorithms A and B are used on the EPR4 channel. Simulation results show that Algorithm A can detect effectively full erasures, while missing most of the partial erasures of $\eta = 0.5$. This is consistent with the previous BCJR algorithm analysis. However, Algorithm B can detect effectively both full erasures and partial erasures of depth $\eta = 0.5$. The improvement is due to the additional use of the amplitude thresholding of the BCJR algorithm LLRs. Secondly, Algorithm C is used on DC-full and DC-mix PMRCs. Its effectiveness is not unexpected.

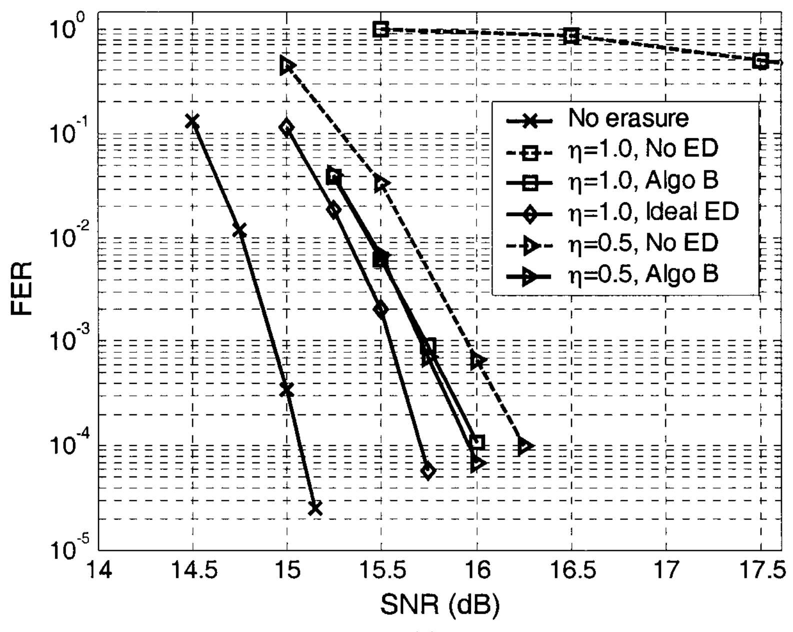
5.5 LDPC Erasure Correction Performance

LDPC codes are used for erasure correction in LMRC and PMRC. The Algorithms B and C described above are used for erasure detection. The LDPC code used is a regular code (4608, 4096, 4,36). Other settings are the same as in Fig. 5.9. An RLL code (0,28|18) is used and $G' = 50, I' = 18$.

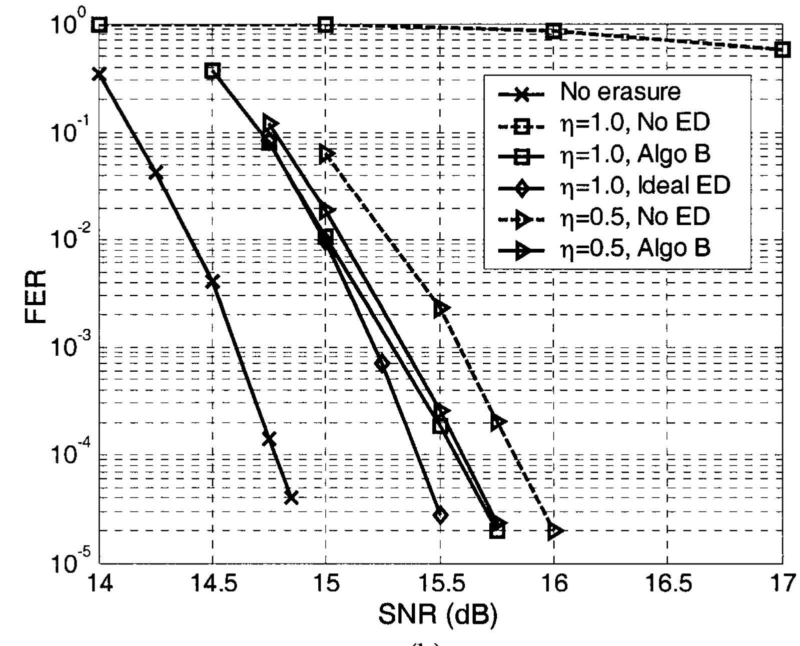
5.5.1 Longitudinal Recording

Since Algorithm A does not work well, we do not use this algorithm in the simulations. Instead, Algorithm B is used for an equalized EPR4 channel.

Shown in Fig. 5.11 are the FER simulation results for the LMRC with $S_c = 2.995$ and with $L_e = 128$-bit erasures of depth $\eta = 1.0$ or 0.5 and different compositions of noise, namely, AWGN, and 40% AWGN plus 60% media noise. In both cases, Algorithm B is working effectively, for detection of both full and partial erasures. This is supported by the fact that the system performance with Algorithm B is very close to that with ideal detection. It is also observed that erasures need to be detected, because the system performance using erasure detection is much better than that using no erasure detection. However, even if ideal erasure detection is available, the performance loss due to erasure is still about 0.8 dB at FER = 10^{-4}. Therefore, if erasures have to be corrected, a certain

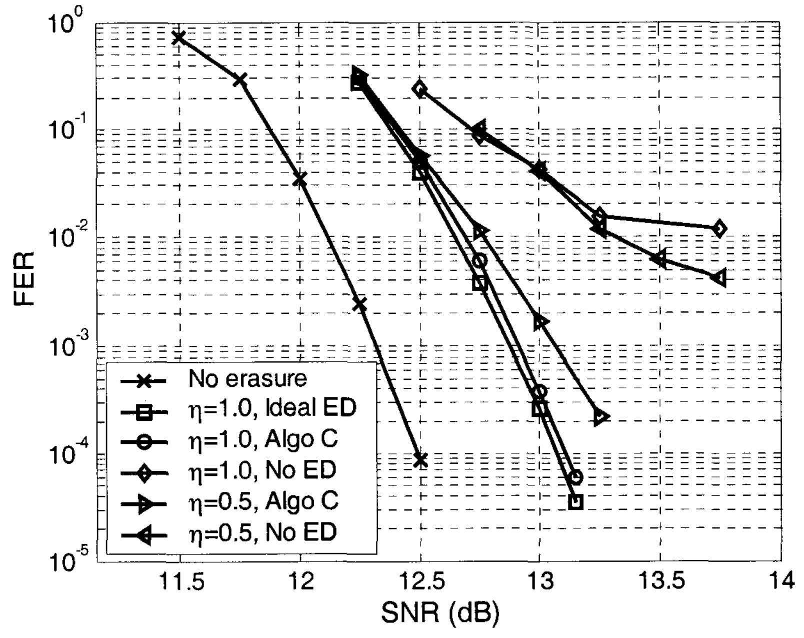


(a)

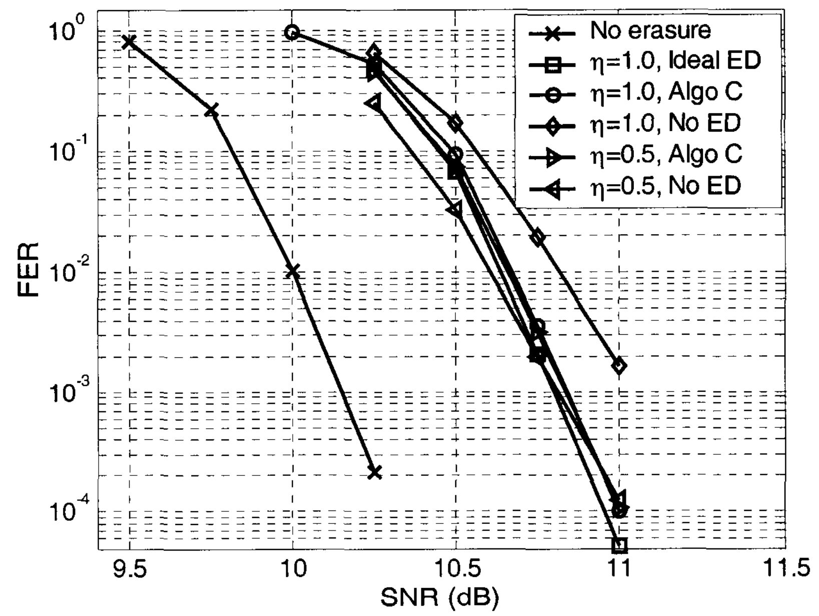


(b)

Fig. 5.11. LDPC (4608,4096,4,36) coded LMRCs with $L_e = 128$ -bit erasures and (a) AWGN and (b) 40% AWGN and 60% media noise.



(a)



(b)

Fig. 5.12. LDPC coded PMRCs with L_e = 128-bit erasures. (a) DC-full channel with AWGN, (b) DC-mix channel with 10% AWGN plus 90% media noise.

margin of SNR must be maintained to ensure the required system performance.

5.5.2 Perpendicular Recording

Algorithm C is used for PMRC in this section. Since the DC-free target has poor performance, only the DC-full and DC-mix targets in Table 1.2 are used. Shown in Fig. 5.12 are the FER simulation results for a PMRC with $S_c = 1.4$ and either AWGN, or 10% AWGN plus 90% media noise. All results show that erasures of depth $\eta \geq 0.5$ need to be detected and the detection algorithms are working effectively for both full erasures and partial erasures of $\eta = 0.5$.

Fig. 5.12 (b) needs additional explanation. In this case, the ideal detector and Algorithm C for full and half erasures all have nearly identical performance. This shows that half erasures do not need to be detected. This is surprising but helpful, and can be explained by the channel trellis. In terms of LLRs, the amplitude for bits in erasure is very small, nearly zero. This is why clearing the LLRs does not help the performance, as much as in Fig. 5.11 (a). This property is advantageous and can be used for future work to combat erasures.

Chapter 6

SNR Mismatch

The decoding system for an LDPC coded MRC consists of a channel detector using the log-MAP BCJR algorithm and an LDPC decoder using the MP algorithm. One requirement for using these algorithms is the knowledge of the variance of the noise. However, this variance is not always known, although there are means to estimate it. The problem is further compounded by the fact that even if the variance of the noise were perfectly known, the system performance may not be optimum. More specifically, if the variance of the noise used by the APP decoders is changed by some amount, either up or down, the system performance may improve. This phenomenon is referred to as the SNR mismatch in the literature [54], [55] and has been observed in our simulations for LDPC coded MRCs [56].

The SNR mismatch effect for turbo codes on AWGN channels was discussed in [54]. It was observed that an SNR mismatch of -2~6 dB does not have much impact on system performance; however, a large negative SNR mismatch, beyond -2 dB, almost disables the system. It was further reported in [55] that an SNR mismatch has different effects when different implementations of the BCJR algorithm are used. They studied the SNR mismatch effects for both the basic BCJR algorithm, the log-MAP algorithm, and the simplified max-log-MAP algorithm. Very interestingly, they proved that the max-log-

MAP algorithm is not affected by the SNR mismatch by showing that the variance of the noise can be factored out from the APPs, thus making the hard decoding BER unchanged. Although this is not true for the log-MAP algorithm, the SNR mismatch does not change the BER much, if it is kept within the range of -2~6 dB.

The SNR mismatch issue for LDPC codes was first raised in [2] on MRCs with erasures. It was observed that when a certain number of bits are erased, the codeword cannot be corrected unless the SNR is mismatched, and furthermore, with an appropriate mismatch the performance can be dramatically improved. This observation motivated us to look into the underlying mechanism and how to address this problem in a practical system.

Let the noise variance used by the decoder be $\kappa\sigma^2$, where κ is the SNR mismatch factor, and the corresponding SNR mismatch is defined as

$$\Delta SNR = -10\log_{10}\kappa . \tag{6.1}$$

If $\Delta SNR > 0$, the SNR has been over estimated, otherwise we refer to it as being under estimated.

The issues with SNR mismatch are two-fold: what causes the SNR mismatch effect, and how can we make use of it to improve system performance. We will use two approaches to tackle these issues. First we will use density evolution to illustrate the asymptotic SNR mismatch effect, and then use simulations to verify our results. The density evolution has already been discussed in Chapter 4. The only modification is that σ^2 in (4.15) is replaced by $\kappa\sigma^2$.

6.1 SNR Mismatch for Channel Detection

Before we consider the density evolution analysis of the SNR mismatch effect on LDPC codes, we first consider its influence on channel detection using the BCJR algorithm.

6.1.1 Theory

It was proved in [55] that when the max* operation, typically used in a log-MAP algorithm, is replaced by the max operation, used in a max-log-MAP algorithm, the variance of the noise σ^2 can be factored out from all three components of the APPs, namely, the forward and backward accumulated path metrics α_k, β_k and the instant path metric γ_k. So if hard-decision decoding is used, the BER performance of the channel detection is independent of σ^2.

Next, we will prove that the max-log-MAP or non-turbo-equalized (running only once) log-MAP algorithm is also independent of the SNR mismatch following a similar derivation as in [55].

For a PR channel with $f(D) = 1 + \sum_{i=1}^{L} f_i D^i$, the BCJR algorithm is described in Section 2.4 with γ_k defined in Section 2.3. From (2.5), we see clearly that the computation of the APPs depends on the variance of the noise. However, if we use the max to replace the max* and use

$$\log(1 + e^{\Lambda_a(x_k)}) = \max(0, \Lambda_a(x_k)) \tag{6.2}$$

in (2.8), $\log \Pr(x_k)$ becomes either 0, $-\Lambda_a(x_k)$ or $\Lambda_a(x_k)$. Thus, if it is assumed that

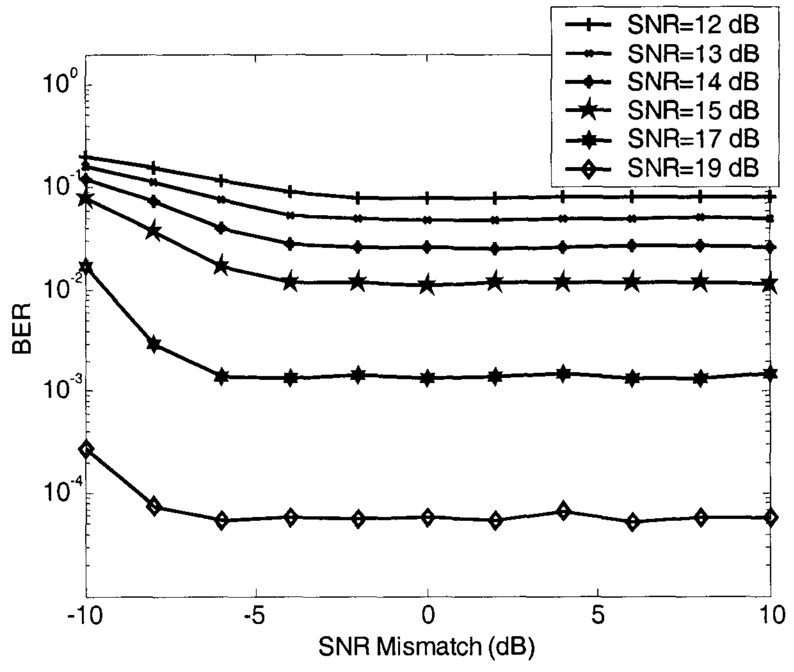
$\Lambda_a(x_k) \propto 1/\sigma^2$, then $\gamma_k(s_{k-1}, s_k) \propto 1/\sigma^2$, so do $\alpha_k(s_k)$ and $\beta_k(s_k)$. Eventually, $\Lambda(x_k) \propto 1/\sigma^2$ and σ^2 can be factored out from $\Lambda(x_k)$. In other words, the SNR mismatch ΔSNR is simply scaling the LLR. However, this does not imply that the performance of a system employing the BCJR algorithm for channel detection is independent of the SNR mismatch because differently scaled LLR may result in quite different performance downstream.

Since $\Lambda_a(x_k)$ is the extrinsic information of the previous iteration in the context of BCJR decoding, it can be easily seen that $\Lambda_a(x_k) \propto 1/\sigma^2$ by initializing $\Lambda_a(x_k)=0$ in the first iteration. This finishes the proof that the max-log-MAP algorithm is independent of the SNR mismatch.

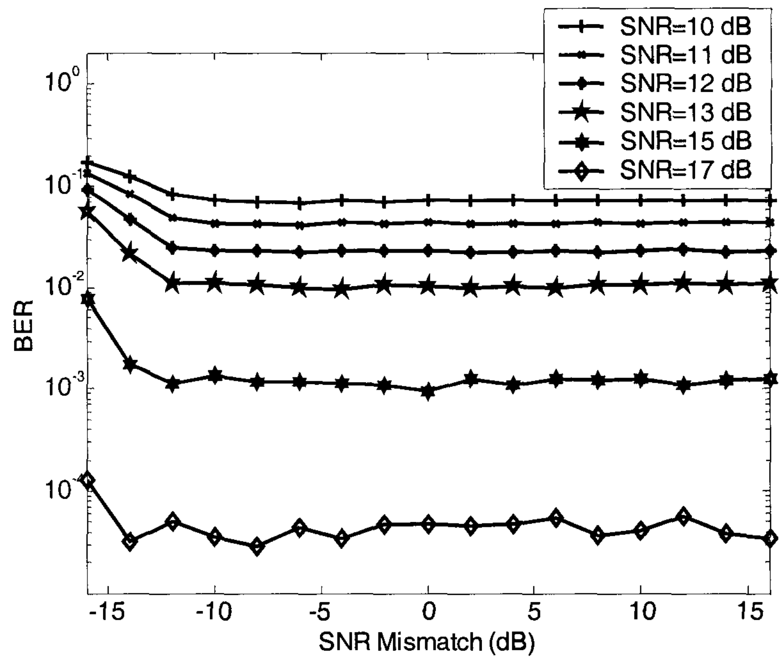
If the log-MAP algorithm is used, the above derivation of $\Lambda(x_k) \propto 1/\sigma^2$ does not hold in general due to (2.8). This shows that a log-MAP algorithm depends on the SNR mismatch. However, for a PR channel with the log-MAP algorithm running only once, which is called a non-turbo-equalized log-MAP algorithm, $\Lambda_a(x_k)=0$, hence σ^2 can be factored out from α_k, β_k and γ_k, and finally from $\Lambda(x_k)$. This proves that the non-turbo-equalized log-MAP algorithm is still independent of the SNR mismatch.

6.1.2 Simulations of Uncoded MRCs

The proof given assures us that the hard decision max-log-MAP implementation of the BCJR algorithm is independent of the SNR mismatch, regardless of the number of iterations. For the log-MAP implementation, since we only run it once in our application, it is also independent of the SNR mismatch.



(a)



(b)

Fig. 6.1. Simulations of uncoded MRC with $S_c = 2.995$ equalized to EPR4 with (a) AWGN (b) 10% AWGN and 90% media noise.

Two sets of simulations are carried out on an MRC with $S_c = 2.995$ equalized to an EPR4 target, in which the log-MAP algorithm is used and either AWGN or media noise are dominant. In both cases, several SNRs are used. The simulation results are shown in Fig. 6.1, from which it is found that the BER is independent of the SNR mismatch over a large range. This is in close agreement with the theoretical results.

However, the independence of the BCJR algorithm on the SNR mismatch does not imply that a system employing the BCJR channel detection is insensitive to the SNR mismatch because differently scaled LLRs may result in quite different performance of the LDPC decoder.

6.2 SNR Mismatch for LDPC Decoding

Given the insensitivity of the BCJR channel detection to SNR mismatch, we can conclude that it is the LDPC MP decoding algorithm that is mostly affected by the SNR mismatch. However, we must emphasize again that this does not mean that the channel detection is irrelevant. Actually the LDPC decoder is sensitive to the input channel message produced by the soft channel detector.

6.2.1 Gaussian Channels

First let us look at LDPC coded AWGN channels. For this linear system, we can assume that the all-zero codeword is transmitted without loss of generality. It is well known that the pdf of the channel message, in terms of the LLR, is a Gaussian function with mean $2/\sigma^2$ and variance $4/\sigma^2$, i.e.,

$$f_{\Lambda_e} = \mathcal{N}(2/\sigma^2, 4/\sigma^2). \tag{6.3}$$

In summary, the channel pdf has two important properties:

- It is Gaussian distributed.
- Its variance is twice the mean. Or, in other words, the variance-to-mean ratio (VMR) equals two.

The combination of these two properties is called the consistency condition in [38]. Simulations have shown that for this system any SNR mismatch is detrimental. This motivated us to investigate whether the violation of either or both of these two properties is responsible for the SNR mismatch effect.

Now we consider channels that have a perfect Gaussian pdf. For the baseline AWGN channel, we assume the channel pdf to be exactly known with Gaussian density

$$f_{\Lambda_e} = \mathcal{N}(2/\sigma^2, 2m/\sigma^2), \tag{6.4}$$

where m is the VMR. Note that such channel may not exist in practice, and that this assumption is used only for investigation purposes.

For such channels with different VMR, we use a density evolution similar to the one described in Section III to compute the SNR threshold. Since this threshold is a typical parameter that features the asymptotic behavior of an LDPC coded system, we believe that it can highlight some essential properties of this system. For m=1.7~2.3, we computed their SNR thresholds, as plotted in Fig. 6.2. The curve for the standard AWGN channel with $m = 2.0$ confirms the observation that LDPC codes on AWGN channels should not use any SNR mismatch. For $m = 1.9$ and 2.1, despite the fact that the SNR threshold is minimum when no SNR mismatch is used, the differences between no SNR mismatch and a ΔSNR=1 dB for $m = 1.9$ and a ΔSNR=-1 dB for $m = 2.1$, compared to those for $m = 2.0$, are small. For $m = 1.7$ and 2.3, it is clear that ΔSNR=1 and -1 dB

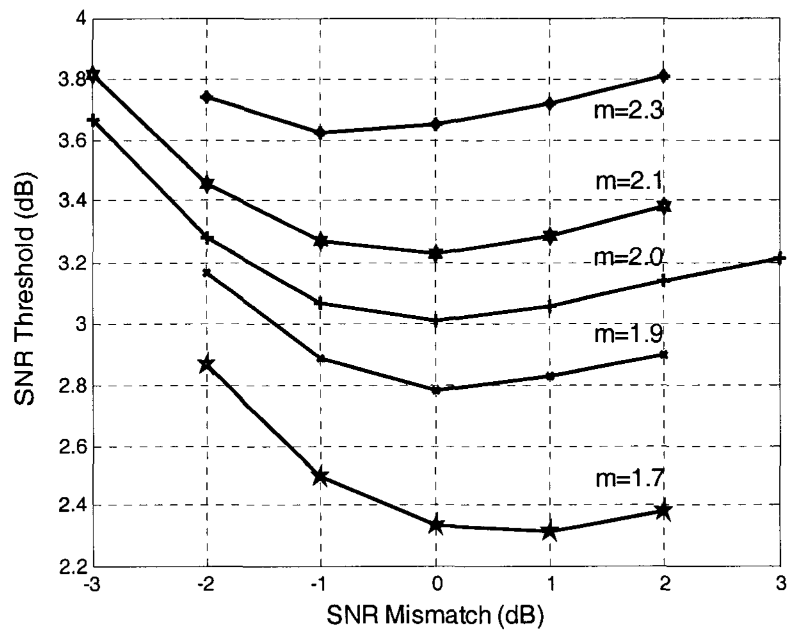


Fig. 6.2. SNR thresholds for channels with Gaussian density given in (6.3).

improve the SNR threshold slightly.

From the above analysis, we can conclude that the VMR is one of the major contributing factors to the SNR mismatch effect. Over-estimation or under-estimation of the SNR, respectively, improves the SNR thresholds when the VMR is larger or smaller than two. Furthermore, the more the VMR deviates from two, the larger the SNR mismatch. We observed numerically that the following value for the SNR mismatch,

$$\kappa = \mathrm{VMR} / 2 \tag{6.5}$$

is optimum if the channel message is approximately Gaussian distributed. Although we cannot prove it rigorously, we can check it using both the density evolution analysis and simulations.

6.2.2 Ideal EPR4 Channel

Next we use the ideal EPR4 channel as our target, which is of more interest to us. Instead of using only AWGN noise as in the preceding subsection, both white and correlated additive Gaussian noise are considered. For the purpose of investigating the influence of correlated noise on the SNR mismatch, we use a very simple third-order low pass FIR filter on the white noise to simulate correlated noise. The density evolution described in Chapter 4 is used to compute the SNR thresholds for the (4,36) LDPC coded EPR4 channel. The results are shown in Fig. 6.3. We observe that, for white noise, no SNR mismatch is supposed to be used, while for correlated noise, a -1-dB mismatch is helpful. It can also be seen that the coding loss due to the noise correlation is about 0.8 dB when no SNR mismatch is used. This indicates, as expected, that the BCJR algorithm is not optimum for correlated noise, and that some modification might be needed.

Recall that in this SNR threshold analysis of an LDPC coded PR channel, the channel pdf is obtained by approximating the histogram of the BCJR APPs. A commonly used assumption for this channel pdf is that it is Gaussian distributed. It is sometimes further assumed that the VMR equals two [57]. In our simulations, we observed that this channel pdf is approximately Gaussian; however, the VMRs are not equal to two, as

Table 6.1. VMRs of channel LLR pdf densities for ideal EPR4 with AWGN and correlated noise.

SNR	White noise		Correlated noise	
(dB)	mean	variance	mean	variance
7	1.905	2.81×10^{-5}	2.302	3.76×10^{-5}
5.2	2.037	2.82×10^{-5}	2.331	3.26×10^{-5}

shown in Table 6.1.

From Table 6.1, we see that for white noise, the VMRs are very close to two, such that $\kappa \approx 1$ and ΔSNR ≈ 0, while for correlated noise, the VMRs are slightly larger than two, therefore $\kappa = 1.15$ and ΔSNR=-0.67 dB. Regardless of whether the channel pdf is Gaussian or not, the SNR mismatch effects due to the VMRs are consistent with the results in the preceding subsection. This means that the VMR produces a similar SNR mismatch effect on PR channels as in AWGN channels.

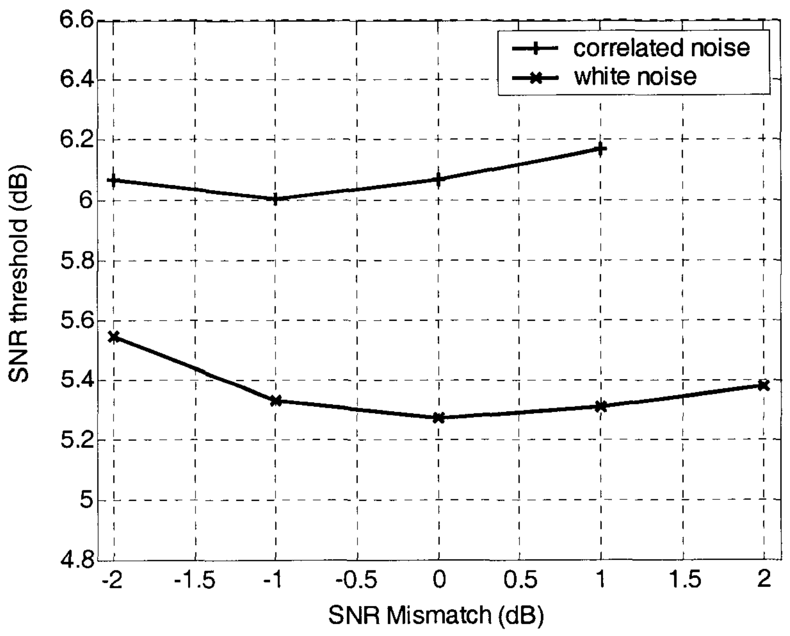


Fig. 6.3. SNR thresholds for LDPC (4,36) coded ideal EPR4 channel with white and correlated noise.

6.2.3 Ideal EPR4 Channel with Erasures

Next we attempt to use a non-Gaussian channel pdf to see how it affects the SNR mismatch. Since the channel pdf of PR channels with AWGN is usually approximately

Gaussian with VMR>2, we will use the EPR4 channel corrupted by erasures as an example of a non-Gaussian channel pdf. In addition, we will consider a channel pdf with a VMR<2 to exclude the SNR under-estimation effect observed when VMR>2.

Erasures have been discussed in Chapter 5. It is assumed that a certain percentage of the bits in the codeword are erased. First we assume that these erasures are not detected, i.e., the location of the erasure is unknown and the noise is still present. From the standpoint of the BCJR LLRs, the erased part of the codeword generates a section of APPs that carry no useful information, and the average channel pdf is no longer Gaussian, instead, it has a small side pedestal on the left tail of the distribution.

In this work, 3% of the bits are assumed erased and only AWGN is present. As

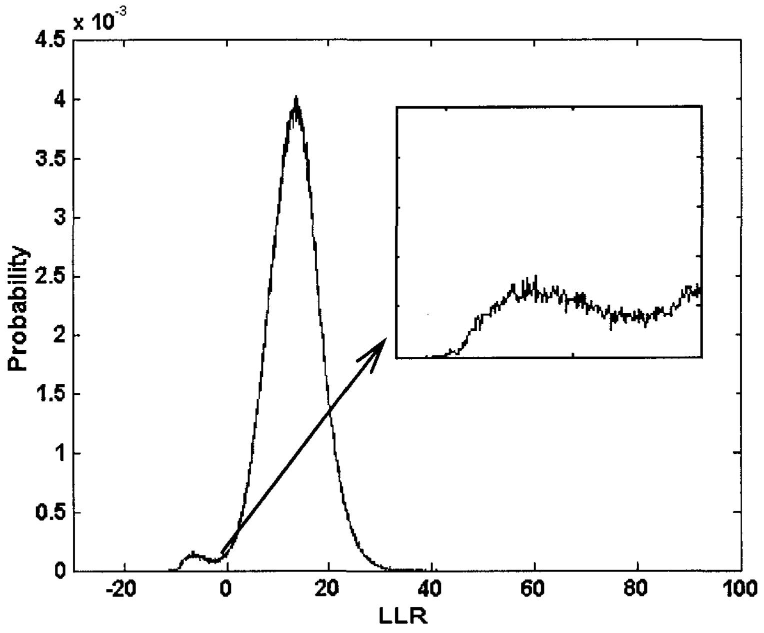


Fig. 6.4. Channel pdf density for the ideal EPR4 channel with undetected erasures at SNR=7 dB.

Table 6.2. VMRs of channel LLR pdf densities for ideal EPR4 with undetected erasures.

ΔSNR (dB)	SNR=7 dB		SNR=5.2 dB	
	mean	variance	mean	variance
0	2.381	1.70×10^{-3}	2.321	8.00×10^{-4}
-1	1.881	2.63×10^{-5}	1.829	2.52×10^{-4}

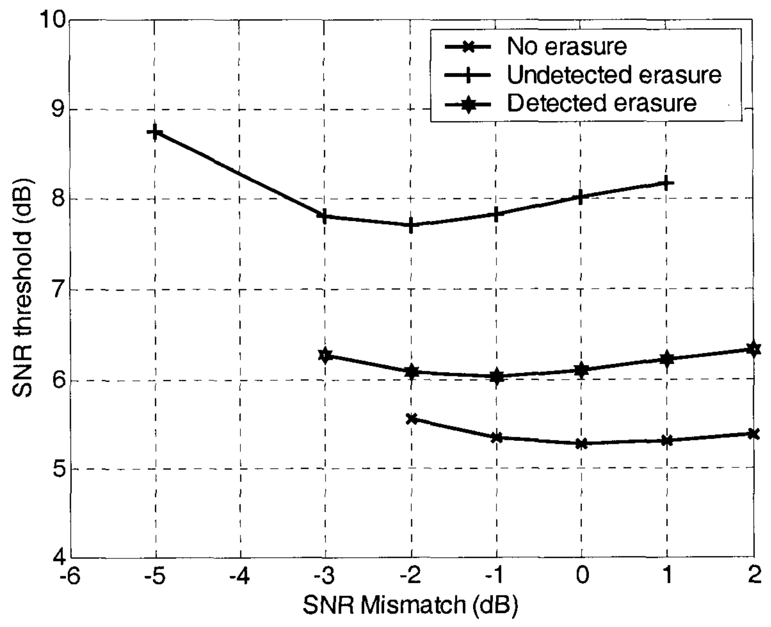


Fig. 6.5. SNR thresholds for LDPC (4,36) coded ideal EPR4 channel with 3% of the bits in erasure.

described in Chapter 5 we use a Monte Carlo simulation to obtain the average channel LLR pdf. Plotted in Fig. 6.4 is an example of such a channel LLR pdf. There is an obvious side pedestal due to the erasures. We use such channel pdf as input to the LDPC decoder to find out the influence of non-Gaussianity on the SNR mismatch effect.

Let us first look at the VMRs of the channel pdfs listed in Table 6.2. Two SNRs, 7 and 5.2 dB are used. When no SNR mismatch is used, i.e., ΔSNR=0 dB, VMR>2. However, when ΔSNR = -1 dB, VMR<2. So we use this channel pdf as test example to exclude the SNR mismatch effect observed for VMR>2. This is equivalent to a hypothetical channel that has this pdf. Using these channel pdfs, the SNR thresholds for an LDPC (4,36) coded system are shown in Fig. 6.5. Note that since we have the hypothetical channel as our target, the SNR mismatch has a 1-dB shift from the original channel. Also shown in Fig. 6.5 are the SNR thresholds for ideal EPR4 without erasures and with detected erasures.

From Fig. 6.5, we see that for this hypothetical channel, a ΔSNR=-2 dB improves the performance by 0.5 dB. This highlights the fact that the violation of the Gaussian property of the channel pdf is one of the causes of the SNR mismatch effect. Also seen is that the coding loss due to this 3% erasure is as large as about 3 dB. Even if the optimum SNR mismatch can be used, the coding loss is still larger than 2 dB, which might not be acceptable. This implies that erasure detection is necessary. The SNR thresholds, already shown in Fig. 6.5, for such detected erasures are also computed using the density evolution algorithm. As expected, the SNR thresholds are reduce by about 2 dB. Also seen is that a ΔSNR=-1 dB is helpful.

6.3 Finding Optimum Values for SNR Mismatch

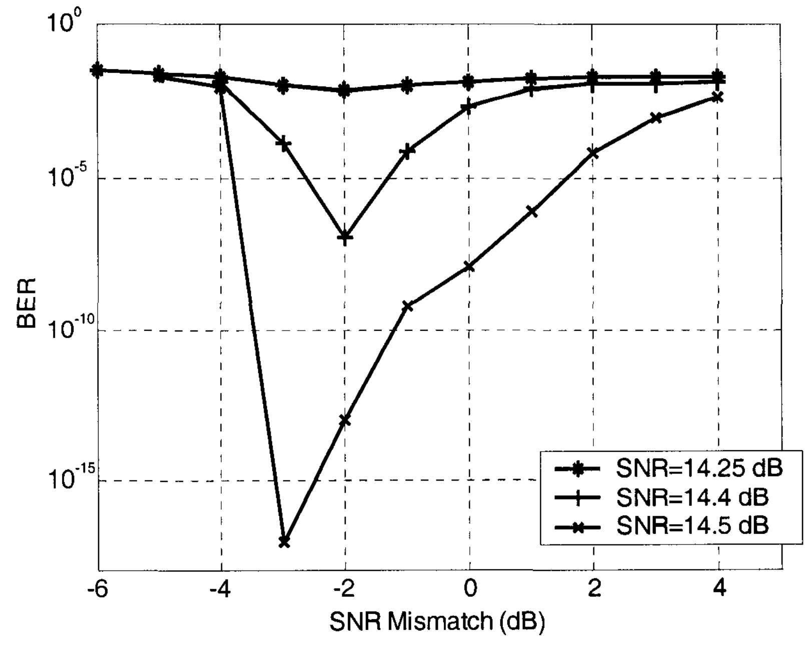
From the discussion in Section 6.2, the Gaussianity and the VMR of the channel LLR pdf are the causes of the SNR mismatch effect. The combination of these two properties makes things even more complex. In fact, for a practical system, e.g., a magnetic recording system, we need to carefully analyze these two properties to find an optimum configuration.

Instead of using SNR thresholds, we use the BER as the performance criterion in this section. The asymptotical BER is estimated using density evolution.

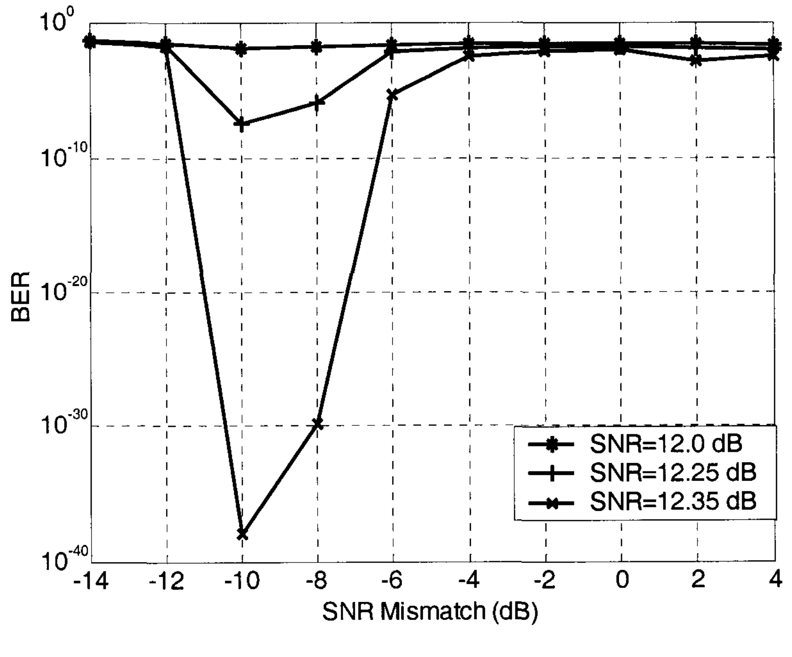
6.3.1 MRCs without Erasures

Before estimating the BERs, we first look at the VMRs and the Gaussianity of the channel pdfs. Although not shown in this work, we found that for practical MRCs, especially for those with dominant jitter noise, the VMR>>2. More specifically, for the cases with pure AWGN VMR $\approx$ 3.0; and for those with 10% AWGN and 90% media noise VMR $\approx$ 12.0. As far as the Gaussianity is concerned, there are no side pedestals as in the erasure-corrupted PR channels, but the shape of the distribution function is asymmetric. In other words, both properties described above are not met, which makes the SNR mismatch effect more prominent.

Shown in Fig. 6.6 are the estimated BERs for different configurations in which either AWGN or media noise is dominant. For each case, three SNRs are chosen. It is easily seen that for AWGN, ΔSNR = -3 to -2 dB is optimum. Particularly, for MRCs that usually work at high SNR, ΔSNR = -3 dB is the best choice. Similarly, for media noise, ΔSNR = -10 to -8 dB of SNR mismatch improved the system performance the most, and ΔSNR = -10 dB is the best choice for high SNRs. At low SNRs, e.g., for SNR=14.2 and 12.0 dB the SNR mismatch is not so obvious, but it actually improves the system to some extent. From these figures, we can see the importance of SNR mismatch in reducing the BERs, especially at high SNRs.

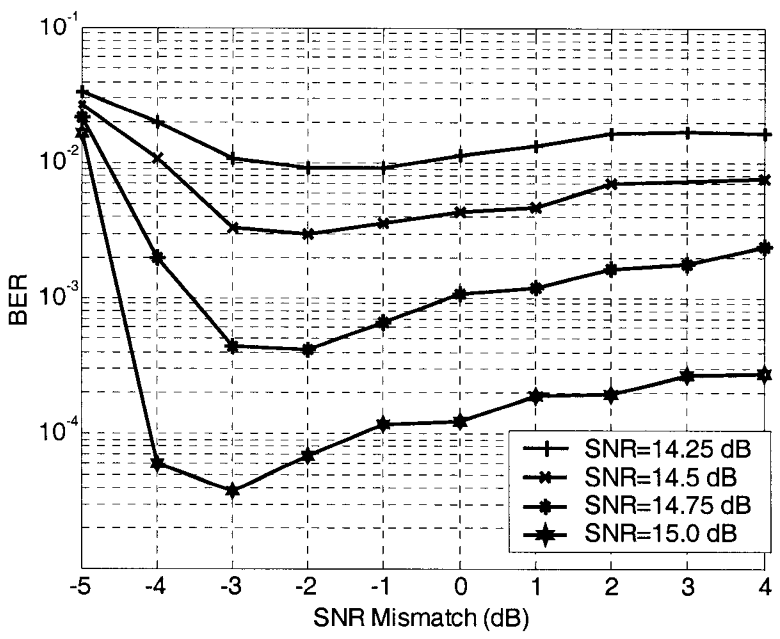


(a)

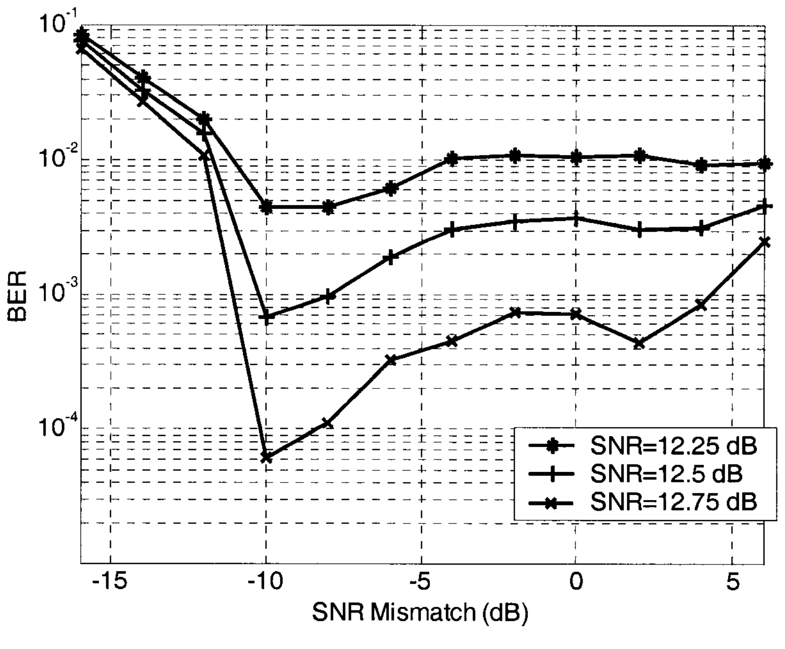


(b)

Fig. 6.6. Estimated BER for LDPC (4,36) coded MRC with (a) AWGN, and (b) 10% AWGN and 90% media noise.



(a)



(b)

Fig. 6.7. Simulated BER for LDPC (4608, 4096, 4,36) coded MRC with (a) AWGN, and (b) 10% AWGN and 90% media noise.

Next we use simulation to find the SNR mismatch effect for a practical system with an LDPC code with appropriate length and rate. Here we still use the (4,36) code, but the length is 4608. We denote it as (4608,4,36). We also choose several sample SNRs and use different SNR mismatches. The simulated BERs vs. SNR mismatch are depicted in Fig. 6.7.

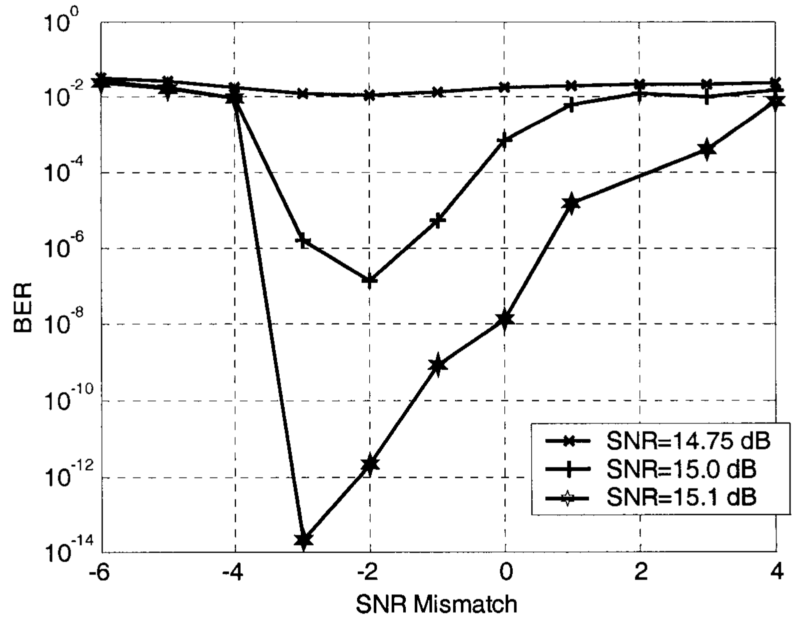
Interestingly, we found almost the same optimum values for SNR mismatch as predicted by density evolution, ΔSNR = -3 to -2 dB and -10 to -8 dB for the two cases. This confirms that the value computed using density evolution can be reliably used in a practical system. In summary, either of these two approaches is available for finding the optimum configuration for a given system.

6.3.2 MRCs with Detected Erasures

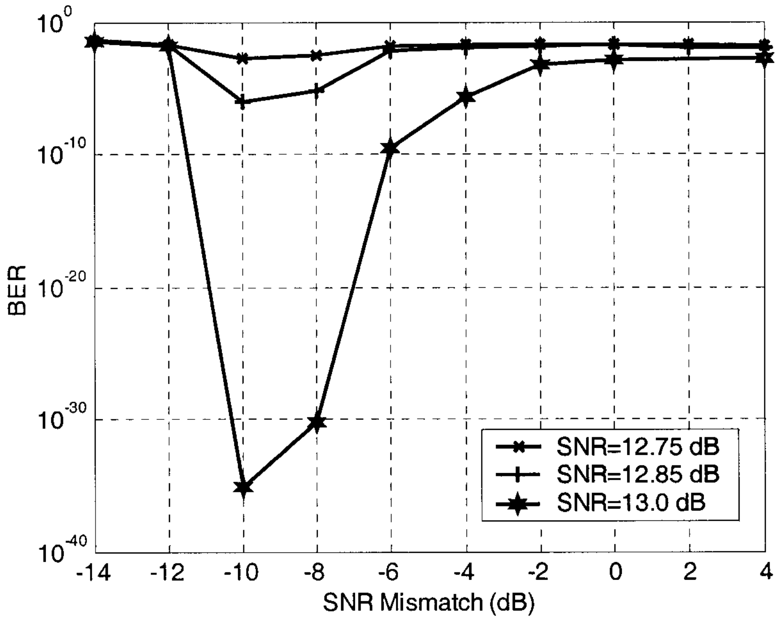
We perform similar BER estimation and simulation for MRCs with erasures. From the discussion of Chapter 5 and Section 6.2, an erasure insertion algorithm is necessary. Here we assume that all erasures are detected.

Similarly, we first look at the channel pdfs. Since the erasures are flagged, the corresponding channel LLRs are set to zero. So the channel pdf is a combination of a smooth Gaussian-like function and a delta function. As far as the VMRs are concerned, VMR $\approx$ 3 and 12, respectively for MRCs with AWGN and with 10% AWGN and 90% media noise, which are similar to those for the examples without erasures.

The estimated BERs for MRCs with 3% of the bits in erasure are shown in Fig. 6.8, while the simulated BERs for the same (4608,4,36) code are given in Fig. 6.9. Note that in both figures, (a) and (b) are AWGN and media noise dominated, respectively. The optimum values of SNR mismatch were predicted reliably and agreed with the simulation

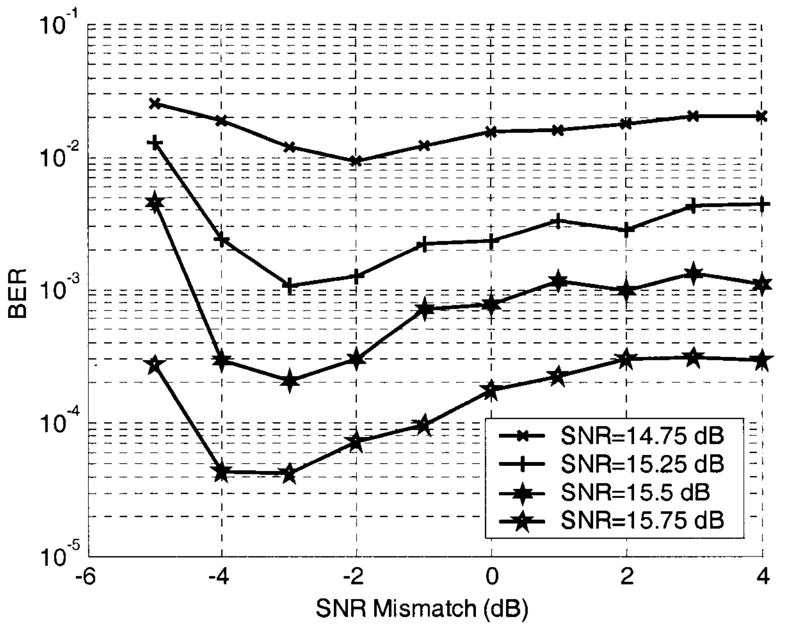


(a)

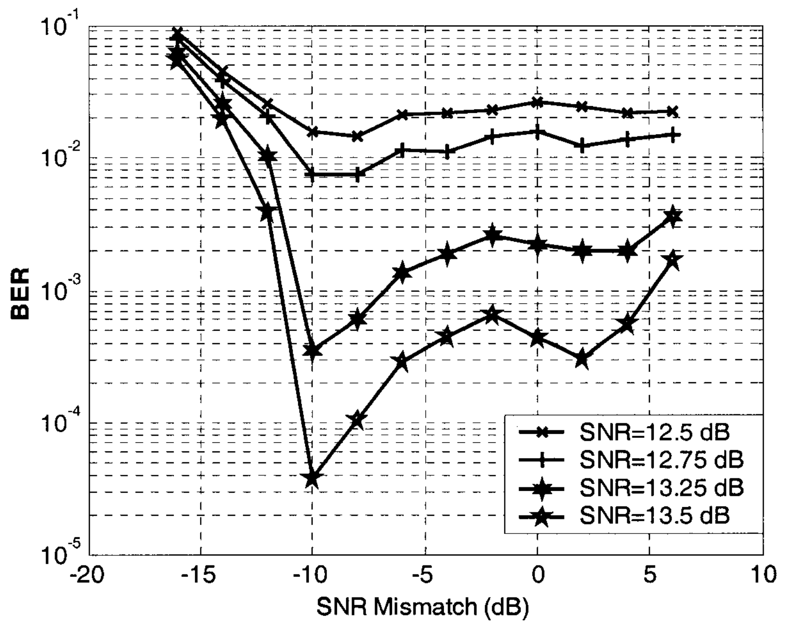


(b)

Fig. 6.8. Estimated BER for LDPC (4,36) coded MRC with 3% erased bits and (a) AWGN, and (b) 10% AWGN and 90% media noise.



(a)



(b)

Fig. 6.9. Simulated BER with ten MP iterations for LDPC (4608,4,36) coded MRC with 138- bit erasures and (a) AWGN, and (b) 10% AWGN and 90% media noise.

results; they are ΔSNR = -3 to -2 dB and -10 to -8 dB for AWGN and media noise dominated MRCs. More interestingly, we observe that the shapes of the BER curves are very alike and the optimum values of SNR mismatch are quite consistent for MRCs with and without erasures, which is very important from the viewpoint of practical implementation. The coding losses due to the presence of the 3% erasures are 0.5 to 0.7 dB.

6.4 Application to Practical Systems

In Section 6.3, we found the optimum values of SNR mismatch for the given system. However, we must note that these values are only valid for the system specified therein. Generally speaking, SNR mismatch is closely dependent on the channel model. Model parameters, including the PR target, channel density, noise composition, equalization technique, and so on, must be specified before we determine the optimum SNR mismatch. For instance, if we use a ME^{2}PR4 target, the SNR mismatch value will change, or if a post-equalization noise whitening technique is used, the SNR mismatch effect is expected to be much smaller.

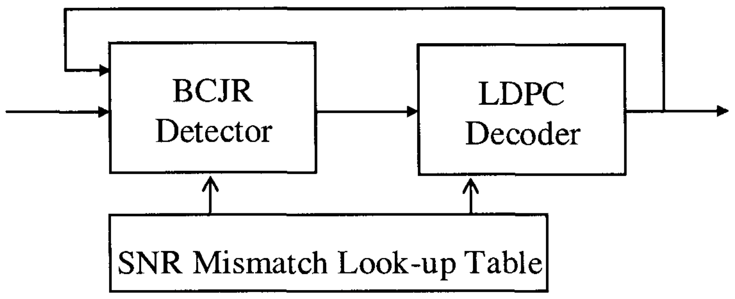


Fig. 6.10. A practical scheme employing SNR mismatch.

These optimum values for SNR mismatch can be employed in a practical system, in order to substantially improve its performance. The values can be determined by an appropriate calibration test of the given unit and stored for the decoder's use. Such a scheme is shown in Fig. 6.10. Note that the application of SNR mismatch can be combined with the erasure detector described in Chapter 5.

Chapter 7

Channel Detection for Perpendicular Recording With Intertrack Interference

The next-generation information storage systems will rely on perpendicular recording to deliver areal densities approaching the fundamental limits of magnetic recording. However, before this promise can be fulfilled, a number of challenging signal processing and coding problems must be solved.

One of the significant problems, perhaps the most critical one, is the presence of large amounts of ITI in the read channel, as well as OTI. Although ITI can be alleviated by using multi-track multi-head recording, these systems are of limited practical interest, and are not considered here. The focus of this work is on a single-head system, the current state-of-the-art for hard-disk drives.

Although there are quite a number of references on channel detection with OTI, e.g., [58]-[60], the issue of ITI has received much less attention, but may become particularly important in the future. This is due to the rapid advances in perpendicular recording research in recent years. The issue of ITI in longitudinal recording was considered and modeled in [61]. While the OTI can be mitigated to some extent by channel equalization [58], joint-track detection seems to be a more effective method [59], [60]. The first formulation of joint track detection, the joint-PRML detection, was presented in [59].

However, other algorithms are yet to be studied.

In this chapter, we investigate the channel detection algorithms for PMRC with ITI [62], [63]. These algorithms can be straightforwardly extended to OTI. An LDPC code is usually used but is not the focus of this chapter. The goal is to optimize the channel detection to attain the best possible performance, to find out what levels of ITI are acceptable in a practical system, and eventually, to optimize the track density of perpendicular recording systems.

7.1 ITI model

In this chapter we only consider an ITI model with one interfering track. Extension to multiple interfering tracks is straightforward.

Shown in Fig. 7.1 is a model for PMRC with ITI. Let the pulse response of the desired and interfering tracks be $h(t)$ and $g(t) = \lambda \cdot h(t - \Delta t)$, where λ and Δt are the interfering factor and phase difference between the two adjacent tracks. However, we only consider $0 \leq \Delta t \leq 0.5T$ in the sequel, because the results for $\Delta t < 0$ are similar. Denote the recorded data sequences on the main and interfering tracks as $\{x_k\}$ and $\{y_k\}$, then the received signal is given by

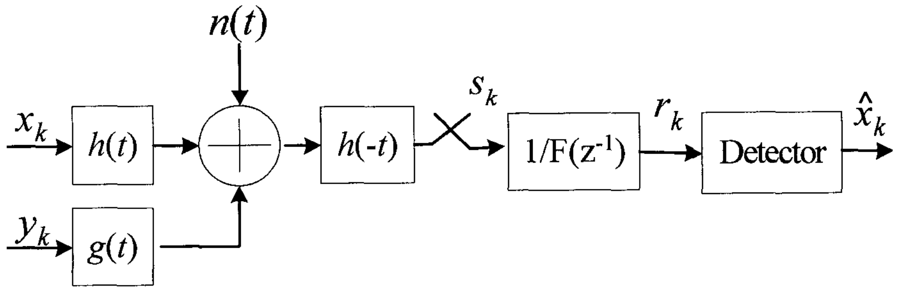


Fig. 7.1 Diagram of PMRC with ITI. Only one interfering track is considered.

$$r(t) = \sum_k [x_k h(t-kT) + y_k g(t-kT)] + n(t) \tag{7.1}$$

This model is similar to the one used in [60], except that the coefficients for $h(t)$ and $h(t-\Delta t)$ are different, and the phase difference Δt is assumed to be zero in [60].

The optimal detector for (7.1) includes a filter matched to the channel response of the main track, namely, $h(-t)$. It is well known that the samples of the matched filter output provide sufficient statistics for optimal data processing if no ITI is present. Here we assume that this still holds in the presence of ITI. This can be justified by the fact that we are only interested in detecting the desired track sequence.

Similarly to the simple channel model in Section 1.3, let $r_{h,k} = [h(t) * h(-t)]_{t=kT}$, $r_{g,k} = [g(t) * h(-t)]_{t=kT}$ and $\eta_k = [n(t) * h(-t)]_{t=kT}$, then the signal s_k at the output of the sampler is

$$s_k = (x * r_h)_k + (y * r_g)_k + \eta_k . \tag{7.2}$$

The channel described by (7.2) is simply an ISI channel with ITI and colored noise. Let $R_h(z) = z\{r_{h,k}\}$ and $R_g(z) = z\{r_{g,k}\}$. To whiten the noise, a filter $1/F(z^{-1})$ (in the z domain) is used, where $R_h(z) = F(z)F(z^{-1})$. At the output of this filter, we still have sufficient statistics for data detection,

$$r_k = (x * f)_k + (y * g)_k + v_k \tag{7.3}$$

where $\{f_n\}$ are the coefficients of $F(z) = R_h(z)/F(z^{-1})$, and $\{g_n\}$ are those of $G(z) = R_g(z)/F(z^{-1})$, an FIR approximation, in the least squares error sense, of $\| R_g(z) - F(z^{-1})G(z) \|$, and v_k is the AWGN.

Table 7.1: Coefficients $\{g_n/\lambda\}$ for PMRC of $S_c = 1.0$ and 1.4.

S_c	$\Delta t/T$	$\{g_n/\lambda\}, n=0,1,\ldots L'-1$
1.0	0	[0.74, 0.83, 0.33, 0.08, 0.01]
	0.25	[1.02, 0.61, 0.30, 0.02, 0.03, -0.01, 0.01]
	0.5	[1.27, 0.33, 0.32, -0.05, 0.06, -0.03, 0.02]
1.4	0	[0.40, 0.71, 0.53, 0.24, 0.09, 0.03, 0.01]
	0.25	[0.61, 0.61, 0.14, 0.10, -0.00, 0.02, -0.01}
	0.5	[0.86, 0.42, 0.56, 0.01, 0.14, -0.05, 0.04, -0.02, 0.01]

The ISI coefficients $\{f_n\}$ and $\{g_n\}$ for PMRCs with $S_c = 1.0$ and 1.4 are computed by first setting the length very large and then truncating the coefficients that are too small. Listed in Table 7.1 are the $\{g_n/\lambda\}$ for $\Delta t \geq 0$ and f_n=g$_n$(Δt =0)/λ. It is observed that $\{f_n\}$ and $\{g_n\}$ are generally of different length L and L'. This is in good agreement with [60]. Since these $\{g_n\}$ and $\{f_n\}$ are too long, they must be equalized to some target of shorter length, e.g., length three to six.

7.2 Information Rate of PMRC with ITI

Our objective in this section is *not* to develop a technique to compute the information rate for a PMRC with ITI. Instead, we use it as a tool in determining what channel detection is the right approach for handling ITI in perpendicular recording. From a practical perspective, it also allows us to quantify how close we can get to the fundamental limit of the channel with a practical system.

Two methods to view the model (7.3) are considered, namely, the single-track model and the joint-track model.

7.2.1 Single-Track Model

First we view (7.3) as a single-track channel with ITI treated as additive noise. Define

$$w_k = \sum_j y_j g_{k-j} + v_j, \tag{7.4}$$

then (7.3) can be rewritten as

$$r_k = \sum_j a_j f_{k-j} + w_k . \tag{7.5}$$

The noise in (7.4) is usually treated as AWGN, e.g., in the MLSD or the BCJR channel detection. Then the information rate can be computed as

$$R \le \lim_{N\to\infty} [\mathcal{H}(\mathbf{r}) - \mathcal{H}(\mathbf{w})] / N . \tag{7.6}$$

However, it is very critical to note that in this single-track model, the received data **r** is contaminated with the noise **w**, but has no knowledge of the structure of the interfering track. Only the structure of the desired track is known. For this reason, we denote the information rate for this model as

$$R \le \lim_{N\to\infty} \mathcal{H}(\mathbf{r} \,|\, \mathbf{f}) / N - 0.5 \log_2 (2\pi e \sigma_w^2) , \tag{7.7}$$

where the variance of **w**, $\sigma_w^2 = \sigma_v^2 + \text{var(ITI)}$, is assumed known or can be measured experimentally.

7.2.2. Joint-Track Model

Another model for (7.3) is the joint-track model. Looking back at (7.6), the information rate for this model is still defined as $R \le \lim_{N\to\infty} [\mathcal{H}(\mathbf{r}) - \mathcal{H}(\mathbf{w})]/N$. However, a unique character of this joint-track model is that the structures of both the desired and the interfering tracks are known to the receiver. Therefore the received data $\mathbf{r}$ is viewed as joint-track data instead of single-track, and $\mathbf{w}$ is viewed as the sum of ITI and noise instead of only noise. This is the essential difference between the joint-track model and the single-track model. For this same reason, the information rate for the joint-track model is denoted as

$$R \le \lim_{N\to\infty} \{\mathcal{H}(\mathbf{r} \,|\, \mathbf{f}, \mathbf{g})/N - \mathcal{H}(\mathbf{w} \,|\, \mathbf{g})/N\}. \tag{7.8}$$

Both $\lim_{N\to\infty} \mathcal{H}(\mathbf{r} \,|\, \mathbf{f}, \mathbf{g})/N$ and $\lim_{N\to\infty} \mathcal{H}(\mathbf{w} \,|\, \mathbf{g})/N$ can be estimated using the techniques in Chapter 4, on the joint-track trellis of **f** and **g**, and the single-track trellis of **g**, respectively [64].

7.2.3 Numerical Results

First let us define the SNR as follows,

$$SNR = \sum\nolimits_k f_k^2 / 2\sigma_v^2 . \tag{7.9}$$

for both the single-track and joint-track models. Note that another definition of SNR, particularly meaningful for the single-track model is $SNR = \sum_k f_k^2 / 2\sigma_w^2$. From a practical standpoint, our purpose is to retrieve the desired-track data from the channel output impaired by noise and ITI, no matter how ITI is handled. Therefore, no matter what definition of SNR is used, we must use a consistent one for both the single-track and joint-track models. Otherwise, the comparison will be unfair.

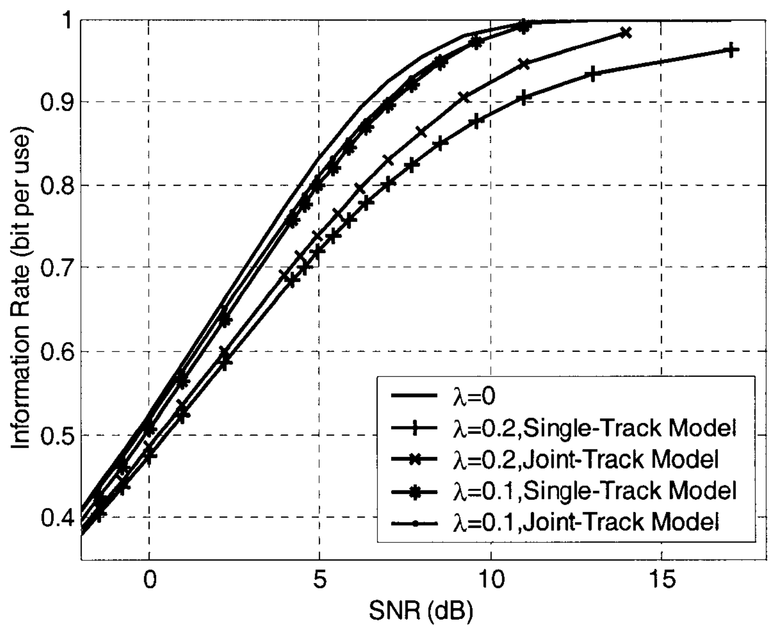


Fig. 7.2 Information rate for PMRC with **f** = [0.40, 0.71, 0.53, 0.24, 0.09, 0.03, 0.01] and **g**=λ**f**.

Using the technique outlined above, we computed the information rates for PMRCs with ITI using both the single-track and the joint–track models as in (7.7) and (7.8). Shown in Fig. 7.2 are the results for $S_c = 1.4$ with $\Delta t/T$=0 whose **f** and **g** can be found from Table 7.1. Results for $\Delta t/T$ = 0.25 and 0.5 are similar and they are not shown. The information rate for λ = 0, i.e., no ITI, is used as a reference. It can be seen from this figure that the SNR losses due to ITI are much smaller using the joint-track model than using the single-track model, particularly when ITI is severe, e.g., λ=0.2. This indicates that the joint-track model should be used in this case. On the other hand, the differences between the two models for λ=0.1 are small, implying that either the joint-track or the single-track model can be used.

7.3 Single-Track Detection

In this and the following sections, we consider channel detection algorithms for handling ITI. Channel detection algorithms corresponding to the single-track and joint-track models are called single-track and joint-track detection.

Note that the PR targets listed in Table 7.1 are normally too long for practical channel detection. Therefore, we use equalized and shorter targets in this section. Since we are mainly concerned about which detection should be used, we use ideal GPR targets with AWGN in this and the following sections. Specifically, we use the target **f** = [1, 1.72, 1.15, 0.33], the DC-full target listed in Table 1.2, and $\mathbf{g}=\lambda\cdot\mathbf{f}$ for PMRC with $S_c = 1.4$ and $\Delta t/T=0$ in the sequel. The results for other cases are similar. Also note that the definition for SNR in (7.9) is still used but the rate of the LDPC code is included.

The simplest detection scheme for the model in (7.3) is single-track detection, in which the ITI due to the interfering data is treated as additive noise. For the single-track detection, the following algorithms presented in Chapter 2 are employed: SOVA, BCJR (log-MAP), forward-MAP. The joint-track detection will be investigated in the following section.

The advantage of the single-track detection is its simplicity, and the fact that there is no need to modify the algorithms currently used in commercial products. However, the disadvantage of this scheme is apparent and probably critical for the future development of ultra-high density products. First, the noise power may increase dramatically if the ITI is treated as additive noise. Secondly, since the ITI and the desired–track signals are in exactly the same frequency range, no linear filtering techniques can be used effectively to prevent the ITI from entering the receiver. Thirdly, conventional detection algorithms are optimum for AWGN. Even when a noise whitening technique is used, the total noise spectrum (including ITI) is not white due to the ITI. Therefore, the detection algorithms are not optimum any more.

SNR mismatch can mitigate this effect to some extent. It has been shown in [56] that it is a very effective method to improve the performance of a LDPC coded system when the noise at the input of the BCJR channel detection is not white. By adjusting the noise variance by a certain value, the distribution function of the soft reliability information of the BCJR channel detection is optimized for the downstream LDPC code to perform adequately.

Shown in Fig. 7.3 are the simulation results for $\lambda = 0$ (no ITI), 0.1 and 0.2. For $\lambda = 0$, since no ITI is present, no SNR mismatch is used. For $\lambda = 0.1$ and 0.2, it is found

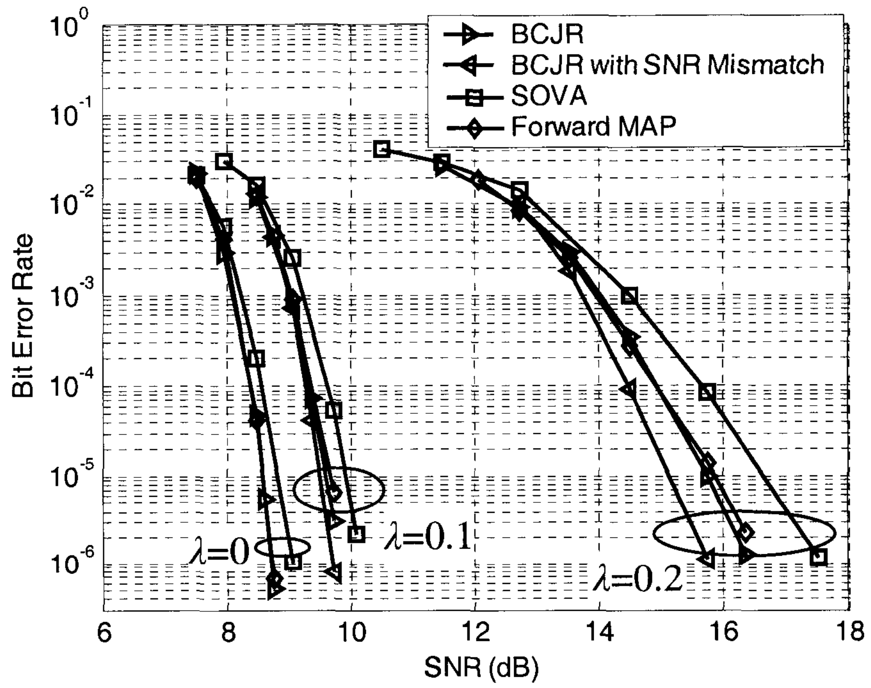


Fig. 7.3. BER simulation results for single-track detection of channel with **f** = [1, 1.72, 1.15, 0.33] and **g**=λ**f**.

that ΔSNR = -0.98 and -2.04 dB, respectively, produces the best results. Especially in the latter case, the SNR mismatch can improve the system performance by 0.8 dB at a bit error rate (BER) of 10^{-4}. These results will be compared to those of joint-track detection in the next subsection.

7.4 Joint-Track Detection

ITI is essentially a signal rather than noise. Therefore, unlike random noise, there is inherent structure in this signal, and this structure can be exploited to improve the channel detection performance. Channel detection algorithms using the structure of two independent PR channels are referred to as joint-track detection algorithms. Such an algorithm for a joint-PRML channel was first presented in [60] for handling OTI.

Based on the assumption that the channel pulse responses of the desired and interfering tracks are known, the whole channel is equivalent to a combination of two PR channels, a more complex joint-PR channel of length $L+L'$.

When the interfering track is present, both the input sequences **x** and **y** must be taken into account in the joint channel trellis. Define the channel state in this case as $S=[(x_{k-1},x_{k-2},\cdots,x_{k-L}),(y_{k-1},y_{k-2},\cdots,y_{k-L'})]$, the state transition becomes $S\xrightarrow{(x_k,x_k)/z_k}S_{new}$. With this joint channel trellis, the MLSD and BCJR algorithms can be implemented straightforwardly. The algorithm steps remain essentially the same as for a single PR channel, except that the number of states is now $2^{L+L'}$.

7.4.1 Joint - MLSD

Both joint-MLSD and joint-MAP symbol-based detection algorithms work on the joint-track trellis very much in the same fashion as the single-track algorithms on a single-track trellis. However, one subtle difference lies in the computation of the LLR in the joint-MAP algorithms, which results in the optimality of the joint-MAP algorithms over the joint-MLSD ones. For this reason, we choose to repeat the algorithms here.

Basically speaking, the joint-MLSD finds the following joint sequence,

$$(\hat{\mathbf{x}},\hat{\mathbf{y}}) = \max_{\mathbf{x},\mathbf{y}} \Pr(\mathbf{x},\mathbf{y} \mid \mathbf{r}) = \max_{\mathbf{x},\mathbf{y}} \Pr(\mathbf{r} \mid \mathbf{x},\mathbf{y}), \tag{7.10}$$

where we assume $\mathbf{x}$ and $\mathbf{y}$ are equally probable and independent. Let $M_k(s)$ be the metric of a path up to time k at state s, then the metric update is as follows,

$$M_k(s) = \max_{s'}\{M_{k-1}(s') + \gamma_k(s',s)\}, \quad M_0(0) = 0, M_0(s \neq 0) = -\infty, \tag{7.11}$$

where the metric for the branch from state s' to state s, with inputs (a_k, b_k) and output $\sum_i h_i a_{k-i} + \sum_i g_i b_{k-i}$ is

$$\gamma_k(s',s) = -\left[y_k - (\textstyle\sum_i h_i a_{k-i} + \sum_i g_i b_{k-i})\right]^2 / 2\sigma^2 + \log \Pr(a_k). \tag{7.12}$$

In the above equation $\log \Pr(a_k)$ is the priori information and $\log \Pr(a_k)$=0 for the first iteration. This joint-MLSD finally selects the sequence $\hat{\mathbf{x}}$ with the maximum path metric $M_N(s_0)$, where the final state is forced back to s_0 by trellis termination.

We must note the difference between the joint-MLSD and the algorithm aimed at detecting the desired-track sequence $\mathbf{x}$. The joint-MLSD maximizes $\Pr(\mathbf{x},\mathbf{y} \mid \mathbf{r})$ while the latter maximizes $\Pr(\mathbf{x} \mid \mathbf{r})$. In other words, the joint-MLSD is detecting both track sequences ($\mathbf{x}$,$\mathbf{y}$), while the one we seek should only detect the desired-track sequence $\mathbf{x}$. This is essentially the same problem outlined in Section 7.2 but in different form.

Let us look more closely at this difference. The maximum probability $\max_{\mathbf{x}} P(\mathbf{x} \mid \mathbf{r})$ for the desired track data **x** can be expanded as,

$$\max_{\mathbf{x}} \Pr(\mathbf{x} \mid \mathbf{r}) = \max_{\mathbf{x}} \sum_{\mathbf{y}} \Pr(\mathbf{x},\mathbf{y} \mid \mathbf{r}) . \tag{7.13}$$

The complexity of (7.13) is proportional to the number of distinct **y**'s, which is prohibitively large. However, if, at high SNRs, the conditional densities involved in the summation are dominated by that of one particular **y**, the following approximation can be made,

$$\max_{\mathbf{x},\mathbf{y}} \Pr(\mathbf{x},\mathbf{y} \mid \mathbf{r}) \approx \max_{\mathbf{x}} \sum_{\mathbf{y}} \Pr(\mathbf{x},\mathbf{y} \mid \mathbf{r}) . \tag{7.14}$$

Note that the term on the left side of (7.14) is exactly the operation of the joint-MLSD algorithm, which finds (**x**,**y**) simultaneously. Thus, the joint-MLSD can be viewed as a possible high-SNR approximation to the sequence detection aiming to select the most probable desired-track data **x**. However, this approximation becomes less and less accurate as the length of **y** increases and no one particular sequence **y** dominates the summation in (7.14). From this standpoint, the joint-MLSD is not optimum for sequence detection of the desired-track data.

The above discussion was first put forward in [60], which we found useful not only for this purpose, but also for the comparison of the joint-MLSD and joint-BCJR algorithms. From this discussion, we may have some clues to improve the joint-MLSD, by selecting several, instead of one, sequences of (**x**,**y**), which makes the approximation in (7.14) better. This is left for future work.

7.4.2 Joint-BCJR

The joint-MAP symbol detection algorithms can be implemented in a variety of ways. Typical examples are the joint-BCJR algorithm and the joint-forward-MAP algorithm. In this section we first present the joint-BCJR algorithm, which is a joint-forward-backward MAP algorithm. The continuously decodable joint-forward-MAP algorithm will be presented afterwards.

The forward and backward recursions of the joint-BCJR are expressed as

$$\alpha_k(s) = \ln \sum_{s'} \{\alpha_{k-1}(s') + \gamma_k(s', s)\},\ \alpha_0(0) = 0, \alpha_0(S \neq 0) = -\infty \tag{7.15}$$

$$\beta_{k-1}(s') = \ln \sum_{s} \{\beta_k(s) + \gamma_k(s', s)\},\ \beta_N(0) = 0, \beta_N(S \neq 0) = -\infty \tag{7.16}$$

where the branch metric $\gamma_k(s', s)$ is defined in (7.12). Then the APP $\Pr((x_k, y_k) = \rho \mid \mathbf{r})$, were $\rho \in \{(0,0), (0,1), (1,0), (1,1)\}$ is computed as,

$$\Pr((x_k, y_k) = \rho \mid \mathbf{r}) = \sum_{s',s:(x_k,y_k)=\rho} \{\alpha_{k-1}(s') \cdot \beta_k(s) \cdot \gamma_k(s', s)\}, \tag{7.17}$$

and the APP $\Pr(x_k \in \{0,1\} \mid \mathbf{r})$ is

$$\Pr(x_k \mid \mathbf{r}) = \Pr((x_k, y_k = 0) \mid \mathbf{r}) + \Pr((x_k, b_k = 1) \mid \mathbf{r}). \tag{7.18}$$

Finally, the LLR is computed as,

$$\Lambda_k = \ln \frac{\Pr(x_k = 1 \mid \mathbf{r})}{\Pr(x_k = 0 \mid \mathbf{r})} = \ln \frac{\sum_{\mathbf{x}:x_k=1} \Pr(\mathbf{x} \mid \mathbf{r})}{\sum_{\mathbf{x}:x_k=0} \Pr(\mathbf{x} \mid \mathbf{r})} = \ln \frac{\sum_{\mathbf{x}:x_k=1} \sum_{\mathbf{y}} \Pr(\mathbf{x},\mathbf{y} \mid \mathbf{r})}{\sum_{\mathbf{a}:a_k=-1} \sum_{\mathbf{y}} \Pr(\mathbf{x},\mathbf{y} \mid \mathbf{r})}. \tag{7.19}$$

Let us take a closer look at (7.10) and (7.19), we can observe a unique property that the join-BCJR algorithm has but the joint-MLSD does not. The joint-MLSD provides the $\max_{\mathbf{x},\mathbf{y}} \Pr(\mathbf{x},\mathbf{y} \mid \mathbf{r})$, but what we are really looking for is the $\max_{\mathbf{x}} \sum_{\mathbf{y}} \Pr(\mathbf{x},\mathbf{y} \mid \mathbf{r})$. The difference is that the operation finds one particular **y** versus the summation of all possible **y**'s. However, in (7.19), if we ignore the time index k, it is obvious that all possible **y**'s are considered in the computation. This is exactly what the detection algorithm for **x** is seeking. From this standpoint, the joint-BCJR algorithm is superior to the joint-MLSD, not only in minimizing BER, but also in aiming at detecting the desired-track data only. This discussion applies to any joint- MAP symbol-based algorithm, including the joint-forward-MAP algorithm.

7.4.3 Joint-Forward-MAP

The joint-forward-MAP algorithm is extension of the single-track forward-MAP algorithm. The merit of this algorithm is that it is continuously decodable.

The branch metric and the forward recursion of this algorithm are exactly the same as for the joint-BCJR algorithm. However, we do not need the backward recursion. Instead we perform the following iteration,

$$\Pr((x_{k-D}, y_{k-D}) = \rho \mid S_k = s, \mathbf{r}_1^k) = \frac{\sum_{s'} \Pr((x_{k-D}, y_{k-D}) = \rho \mid S_{k-1} = s', \mathbf{r}_1^{k-1}) \alpha_{k-1}(s') \gamma_k(s', s)}{\sum_{s'} \alpha_{k-1}(s') \gamma_k(s', s)} \tag{7.20}$$

with initialization $\Pr((x_{k-D}, y_{k-D}) = \rho \mid S_{k-D} = s, \mathbf{r}_1^{k-D})$ being 0 or 1 determined by S_{k-D} and ρ. Then the APP $\Pr((x_{k-D}, y_{k-D}) \mid \mathbf{r}_1^k)$ is computed as,

$$\Pr((x_{k-D}, r_{k-D}) = \rho \mid \mathbf{r}_1^k) = \sum_{s} \Pr((x_{k-D}, y_{k-D}) = \rho \mid S_k = s, \mathbf{r}_1^k). \tag{7.21}$$

Finally, the APP $\Pr(x_{k-D} \mid \mathbf{r}_1^k)$ and the LLR $\Lambda_{k-D} = \ln \frac{\Pr(x_{k-D} = 1 \mid \mathbf{r}_1^k)}{\Pr(x_{k-D} = 0 \mid \mathbf{r}_1^k)}$ can be easily computed, as in (7.19) and (7.20).

The delay D must be at least L. However, it has been shown in [14] for the single PR channel that the longer D is, the better the performance. From this perspective, the joint-BCJR algorithm can be viewed as a special case of the joint-forward-MAP algorithm, where $D = \max(N-k, L)$ for $k = 1,2,\cdots N$. Since the joint-forward-MAP algorithm uses the MAP criterion for the desired-track data sequence, we can state that the joint-BCJR algorithm is the best possible algorithm for detection of the desired-track sequence.

7.4.4 Numerical Results

We perform BER simulations for the same channel used previously for $\lambda = 0$ (no ITI), 0.1 and 0.2. To illustrate the unique difference between the joint-MLSD and the joint-MAP algorithm, as discussed previously, we use the joint-SOVA as an instance of a joint-MLSD algorithm and the joint-BCJR and joint-forward-MAP as instances of joint-MAP algorithms. For single-track detection, the best attainable performance of the BCJR algorithm with SNR mismatch is presented.

The simulation results are shown in Fig. 7.4. From this figure, the superiority of the joint-track detection over the single-track detection is evident, particularly when ITI is severe, e.g., $\lambda = 0.2$. This is in good agreement with the information rate analysis. Also observed is that the performance of the joint-BCJR detection is much better than that of the joint-SOVA detection. This is not unexpected for the reason outlined above. The forward-MAP algorithm has similar performance to the BCJR algorithm, in all scenarios.

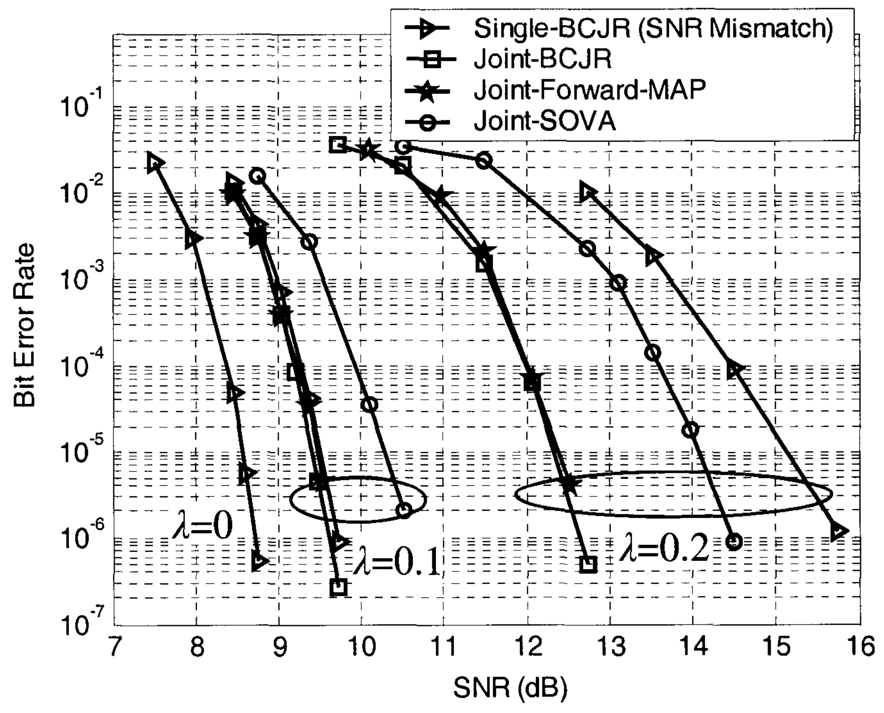


Fig. 7.4. BER simulation results for joint-track detection of channel with **f** = [1, 1.72, 1.15, 0.33] and **g**=λ**f**.

7.5 Single-Track Equalization

In the previous two sections we assumed ideal GPR channels with AWGN. However, this is only for theoretical analysis and is not practical. Similarly to Section 1.3, a realizable low-pass filter $p(t)$ and a FIR equalizer $\{w_k\}$ must be used to equalize the channel to a predetermined GPR target. GPR equalization techniques for single-track detection and joint-track detection are studied in this and the next section, respectively. Both AWGN and media noise dominant channels are considered.

In the single-track model, the equalization is exactly the same as for a simple PR channel, as described in Section 1.3, except that the signal at the output of the sampler s_k is different. Specifically, (1.14), (1.18) and (1.19) can be used to compute the desired track target **f** and the FIR equalizer **w**. For completeness, these three equations are repeated here,

$$\lambda = \frac{1}{\mathbf{I}^T\left(\mathbf{R}_{\mathbf{x}} - \mathbf{R}_{\mathbf{s,x}}^T\mathbf{R}_{\mathbf{s}}^{-1}\mathbf{R}_{\mathbf{s,x}}\right)^{-1}\mathbf{I}} \tag{7.21}$$

$$\mathbf{f} = \lambda\left(\mathbf{R}_{\mathbf{x,x}} - \mathbf{R}_{\mathbf{s,x}}^T\mathbf{R}_{\mathbf{s,s}}^{-1}\mathbf{R}_{\mathbf{s,x}}\right)^{-1} \cdot \mathbf{I} \tag{7.22}$$

$$\mathbf{w} = \mathbf{R}_{\mathbf{s}}^{-1} \cdot \mathbf{R}_{\mathbf{s,x}} \cdot \mathbf{f}\,. \tag{7.23}$$

Note that these equations are very general, can be used for any single-track equalization, as long as s_k is modified as needed. For the issue of equalization for PMRCs with ITI, the interfering signal picked up from an adjacent track must be included in s_k. Another example is media noise. When modeled as a first-order transition jitter noise, the media noise can be considered into s_k straightforwardly.

7.5.1 AWGN Channels

First we consider PMRCs with AWGN. As discussed in Chapter 1, good targets for these channels are DC-full ones, whose spectra match those of the channel very well. Let us consider PMRCs with channel density $S_c = 1.5$. Listed in Table 7.2 are the computed GPR targets **f**'s of length L=3 and 4 for $\lambda = 0$, 0.1, 0.15 and 0.2. It can be easily seen from (7.21)-(7.23) that the equalization is closely dependent on the SNR. Therefore, the working SNR must be specified to optimize the equalization. Another observation is that Δt has almost no impact on the computed targets. This is why it does not show up in Table 7.2. However, this is not true for joint-track detection.

Almost all targets in Table 7.2 are DC-full ones, particularly those for small λ's, e.g.,

Table 7.2. Single-track targets for PMRCs of $S_c = 1.5$ with ITI and AWGN.

SNR (dB)	λ	**f** (L=3)	**f** (L=4)
39	0	[1.0 1.55 0.67]	[1.0 1.85 1.30 0.37]
21	0	[1.0 1.30 0.60]	[1.0 1.50 1.04 0.33]
15	0	[1.0 1.07 0.52]	[1.0 1.21 0.81 0.28]
21	0.1	[1.0 1.12 0.44]	[1.0 1.14 0.50 0.06]
15	0.1	[1.0 01.00 0.45]	[1.0 1.07 0.62 0.17]
21	0.15	[1.0 0.96 0.30]	[1.0 0.947 0.255 -0.047]
15	0.15	[1.0 0.92 0.385]	[1.0 0.955 0.465 0.088]
21	0.2	[1.0 0.82 0.18]	[1.0 0.801 0.105 -0.090]
15	0.2	[1.0 0.835 0.31]	[1.0 0.842 0.329 0.022]

[1.0 1.85 1.30 0.37] for $\lambda = 0$. Very interestingly, this target is almost identical to the target [1.0 1.85 1.33 0.40] obtained by ideal noise whitening in [5]. However, the target [1.0 1.85 1.30 0.37] is optimized for SNR=39 dB. In the simulation SNR range of 12-13 dB, it has been shown that other two targets [1.0 1.21 0.81 0.28] optimized for SNR=21 dB and [1.0 1.50 1.04 0.33] optimized for SNR=15 dB perform better. This can be easily explained by the deviation of the target SNRs from the simulation ones.

Now we use the target [1.0 1.50 1.04 0.33] as a reference. In other words, we assume it is an optimized target, which is true at least for no ITI and actual SNR$\approx$15 dB. Then the question is whether this target is still good for an ITI-channel. An intuitive answer to this question is, probably yes, if there is no knowledge about the presence and level of ITI. For this purpose, we use this target in all scenarios, i.e., for λ =0, 0.1, 0.15, and 0.2 in

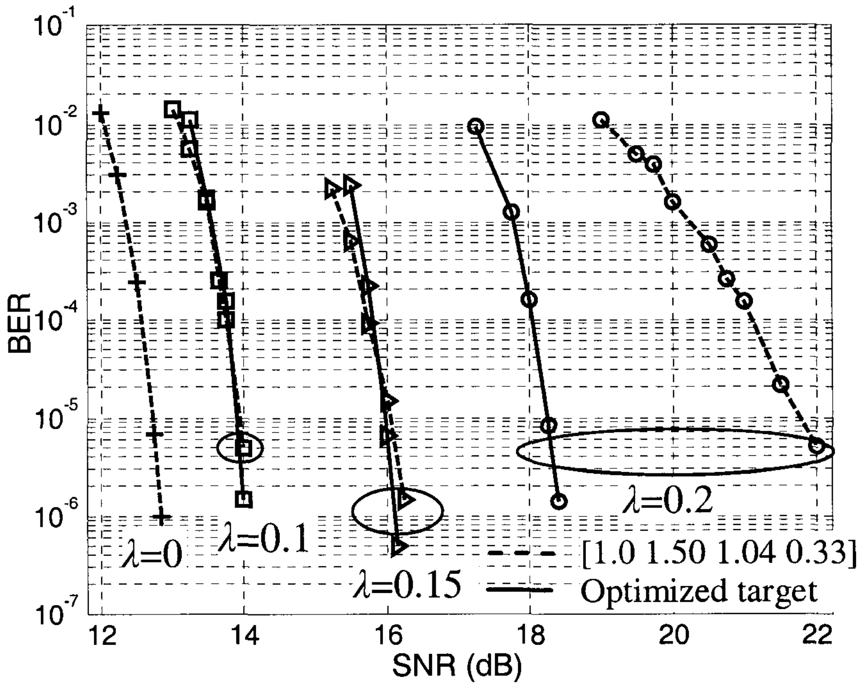


Fig. 7.5. BER simulation results for single-track detection of ITI-channels with AWGN using different targets.

the simulations. In addition, the computed targets for known ITI, as listed in Table 7.2, are also used in simulations.

Shown in Fig. 7.5 are the simulation results, where the BCJR (log-MAP) algorithm and the same LDPC code as in Section 7.3 are used. It is apparent that the ITI-oriented targets are better than the one for no ITI. Although the improvements for $\lambda = 0.1$ and 0.15 are negligible, for $\lambda = 0.2$ it is as large as 3.6 dB at BER=10^{-5}. This supports our expectation that good targets for no ITI are no longer good for ITI-channels. Therefore, ITI, particularly for severe ITI channels, must be detected and handled appropriately in the channel detection. An example for such detection is given in [13], where an adaptive technique is employed to track the OTI level and to adapt the PR target.

7.5.2 Media Noise Channels

In this section we redo the work in the previous section, but for PMRCs with 10% AWGN plus 90% media noise.

Table 7.3. Single-track targets for PMRCs with $S_c = 1.5$, ITI and 10% AWGN plus 90% media noise.

SNR (dB)	λ	**f** (L=2)	**f** (L=3)	**f** (L=4)
21	0	[1.0 0.814]	[1.0 0.95 0.16]	[1.0 0.93 0.07 -0.11]
15	0	[1.0 0.719]	[1.0 0.80 0.11]	[1.0 0.79 0.04 -0.09]
21	0.1	[1.0 0.727]	[1.0 0.70 -0.03]	[1.0 0.71 -0.09 -0.09]
15	0.1	[1.0 0.684]	[1.0 0.717 0.048]	[1.0 0.713 -0.015 -0.089]
21	0.15	[1.0 0.640]	[1.0 0.565 -0.117]	[1.0 0.569 -0.140 -0.041]
15	0.15	[1.0 0.644]	[1.0 0.642 -0.004]	[1.0 0.64 -0.054 -0.079]
21	0.2	[1.0 0.546]	[1.0 0.460 -0.157]	[1.0 0.460 -0.157 0.001]
15	0.2	[1.0 0.595]	[1.0 0.565 -0.050]	[1.0 0.569 -0.085 -0.062]

Recalling the discussion in Chapter 1, good targets for these channels are DC-mix ones. Both DC-full and DC-free targets are generally considered inferior. However, this is not exactly true, as can be seen from the computed targets listed in Table 7.3 for PMRCs with $S_c = 1.5$. For example, the target [1.0 0.93 0.07 -0.11] is almost a DC-full one, however, its performance is better than the DC-mix target [1.0 1.07 -0.09 -0.35], which can be computed using the technique of [5]. While the targets of length L=3 and 4 have some negative terms, targets of length L=2 have all positive terms, thus are DC-full ones. However, the DC components of these targets are much smaller than those in

Table 7.2, i.e., $f_1 < f_0$.

Note that unlike in Table 7.2, we consider additionally targets of length L=2. This is because for targets of length L=3 and 4, f_2 and/or f_3 are much smaller than f_0 and f_1, which implies that short targets of length L=2 may work fine as well. This is a great advantage, making the complexity of a practical channel detector low. We are particularly interested in this fact for joint-track detection and will take advantage of it in next section.

Now we use the target [1.0 0.93 0.06 -0.11] as a reference. Again we ask the question: how is this target working compared with other computed targets for known ITI, as listed in Table 7.3?

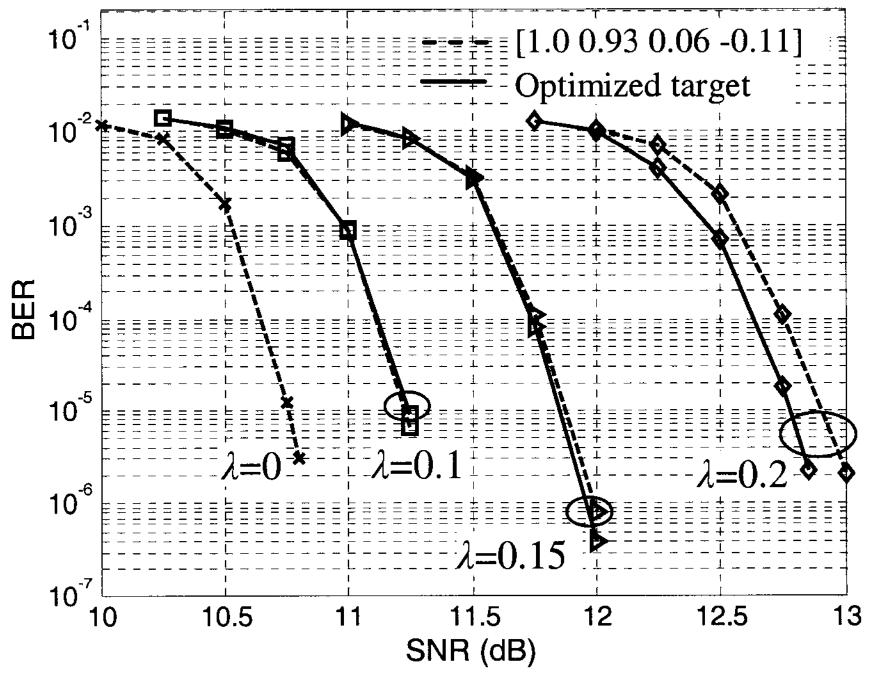


Fig. 7.6. BER simulation results for single-track detection of ITI-channels with 10% AWGN plus 90% media noise using different targets.

Shown in Fig. 7.6 are the simulation results. Similarly as for AWGN channel, the ITI-oriented targets are better than the one for no ITI. This highlights again that ITI needs to be detected and handled in the channel detection. However, the improvement by the ITI-oriented target for ITI with $\lambda = 0.2$ is only 0.2 dB at BER=10^{-5} in contrast to 3.6 dB in the AWGN channel. This shows that the target [1.0 0.93 0.06 -0.11] is not too bad for ITI with $\lambda = 0.2$.

7.6 Joint-Track Equalization

In the joint-track model, the equalizer aims at minimizing the joint mean square error of $e_k = (s * w)_k - (x * f)_k - (y * g)_k$. Using similar techniques as in Sections 1.3 and 7.5, the desired track target **f**, the interfering track target **g**, and the FIR equalizer **w** can be computed as follows,

$$\lambda = \frac{1}{\mathbf{I}^T \left(\mathbf{R}_\mathrm{x} - \mathbf{R}_\mathrm{s,x}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,x} - \frac{\left(\mathbf{R}_\mathrm{x,y} - \mathbf{R}_\mathrm{s,x}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,y}\right)\left(\mathbf{R}_\mathrm{x,y}^T - \mathbf{R}_\mathrm{s,y}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,x}\right)}{\mathbf{R}_\mathrm{y} - \mathbf{R}_\mathrm{s,y}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,y}} \right)^{-1} \mathbf{I}} \tag{7.24}$$

$$\mathbf{f} = \lambda \left(\mathbf{R}_\mathrm{x} - \mathbf{R}_\mathrm{s,x}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,x} - \frac{\left(\mathbf{R}_\mathrm{x,y} - \mathbf{R}_\mathrm{s,x}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,y}\right)\left(\mathbf{R}_\mathrm{x,y}^T - \mathbf{R}_\mathrm{s,y}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,x}\right)}{\mathbf{R}_\mathrm{y} - \mathbf{R}_\mathrm{s,y}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,y}} \right)^{-1} \cdot \mathbf{I} \tag{7.25}$$

$$\mathbf{g} = \lambda \left(\mathbf{R}_\mathrm{x,y} - \mathbf{R}_\mathrm{s,x}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,y} - \frac{\left(\mathbf{R}_\mathrm{x} - \mathbf{R}_\mathrm{s,x}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,x}\right)\left(\mathbf{R}_\mathrm{y}^T - \mathbf{R}_\mathrm{s,y}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,y}\right)}{\mathbf{R}_\mathrm{x,y}^T - \mathbf{R}_\mathrm{s,y}^T \mathbf{R}_\mathrm{s}^{-1} \mathbf{R}_\mathrm{s,x}} \right)^{-1} \cdot \mathbf{I} \tag{7.26}$$

$$\mathbf{w} = \mathbf{R}_\mathrm{s}^{-1} \cdot (\mathbf{R}_\mathrm{s,x} \cdot \mathbf{f} + \mathbf{R}_\mathrm{s,y} \cdot \mathbf{g}). \tag{7.27}$$

It is easy to see that this equalization is general regarding the composition of noise, i.e., whether it is AWGN dominated or media noise dominated. Another fact is that the equalization is closely dependent on the SNR. Therefore, the working SNR needs to be specified. One characteristic unique to joint-track equalization is that the target is affected by the phase difference Δt. This is different from single-track equalization, and Δt may need to be detected too.

We consider PMRCs with $S_c = 1.5$ and $\lambda = 0.2$ ITI. Other values of λ can be evaluated straightforwardly, however, are not implemented here. SNR = 21 and 15 dB and $\Delta t = 0$, $0.25T$ and $0.5T$ are used in the equalization. Both λ and Δt are assumed known in the channel, and the targets and equalizer are optimized.

7.6.1 AWGN Channels

As for single-track equalization, we first consider PMRCs with AWGN, then 10% AWGN plus 90% media noise.

Table 7.4. Joint-track targets for PMRCs with $S_c = 1.5$, $\lambda = 0.2$ and AWGN.

SNR (dB)	Δt	$\mathbf{f}$ ($L=4$)	$\mathbf{g}$ ($L'=4$)
21	0	[1.0 1.506 1.041 0.335]	[0.191 0.301 0.208 0.067]
21	$0.25T$	[1.0 1.518 1.056 0.341]	[0.237 0.296 0.173 0.043]
21	$0.5T$	[1.0 1.529 1.073 0.348]	[0.274 0.280 0.137 0.024]
15	0	[1.0 1.216 0.814 0.277]	[0.176 0.243 0.163 0.055]
15	$0.25T$	[1.0 1.219 0.820 0.279]	[0.207 0.234 0.135 0.037]
15	$0.5T$	[1.0 1.218 0.822 0.281]	[0.230 0.217 0.106 0.021]

Listed in Table 7.4 are the joint-tract targets for AWGN channels. It is observed that the targets $\mathbf{f}$'s for Δt=0 are almost identical to the single-track targets in Table 7.2, and are DC-full ones. Although Δt has some impact on the target **f**, this impact is just slight. On the other hand, the impact on the target **g** is relatively large.

We use in simulations the targets computed at SNR = 21dB in Table 7.4. However, we also consider another case, in which Δt is fixed but unknown to the equalizer. To make things simple, we use $\Delta t' = 0$ in the equalization, and denote it by $(\Delta t, \Delta t')$. Note that the targets in Table 7.4 correspond to $\Delta t' = \Delta t$.

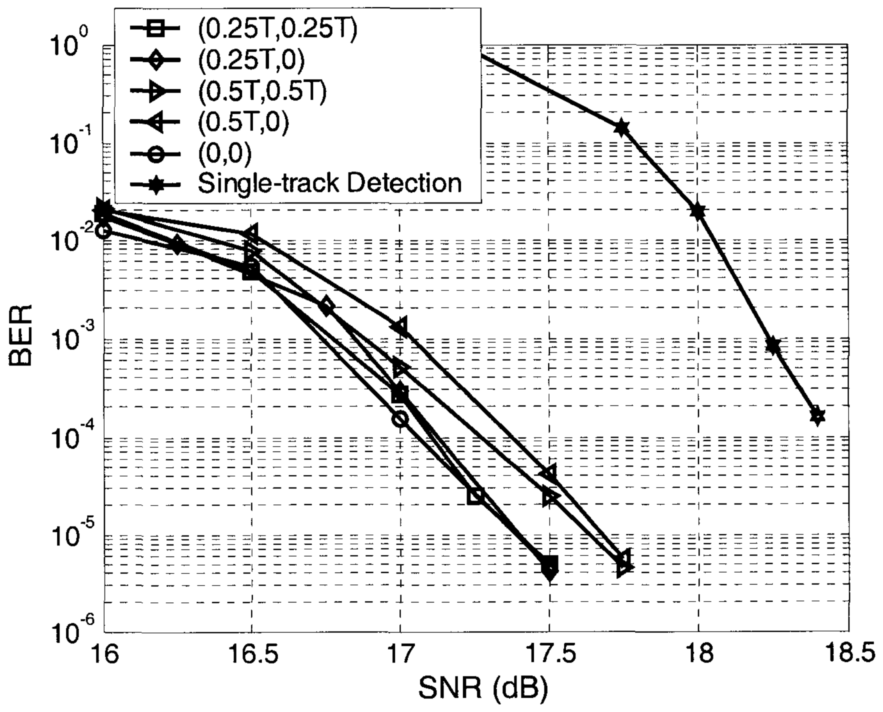


Fig. 7.7. BER simulation results for joint-track detection of ITI-channels with $\lambda = 0.2$ and AWGN. A number of $(\Delta t, \Delta t')$ are considered.

Plotted in Fig. 7.7 are the simulation results. The single-track detection performance using **f** = [1.0 0.801 0.105 -0.090] is also shown. It is observed that Δt does cause some performance loss, and this loss can be as large as 0.25 dB at BER=10^{-5}. Another loss caused by $\Delta t' \neq \Delta t$ is much smaller, implying that the equalizer does not have to know the value of Δt, and can simply use $\Delta t' = 0$ in the equalization. Furthermore, the performance loss caused by $\Delta t \neq 0$ is negligible compared with that caused by ITI. Therefore, we can simply ignore its effect and assume $\Delta t = 0$.

7.6.2 Media Noise Channels

We redo the work in the previous subsection but for media noise dominant channels with 10% AWGN plus 90% media noise.

Table 7.5. Joint-track targets for PMRCs with $S_c = 1.5$, $\lambda = 0.2$ and 10% AWGN plus 90% media noise.

SNR (dB)	Δt	**f** ($L = 4$ or 2)	**g** ($L = 4$ or 2)
21	0	[1.0 0.930 0.061 -0.106]	[0.195 0.186 0.012 -0.021]
21	0.25T	[1.0 0.957 0.073 -0.108]	[0.227 0.146 -0.010 -0.015]
21	0.5T	[1.0 1.005 0.101 -0.111]	[0.242 0.106 -0.021 -0.009]
21	0	[1.0 0.814]	[0.194 0.163]
21	0.25T	[1.0 0.828]	[0.222 0.117]
21	0.5T	[1.0 0.849]	[0.227 0.071]

Listed in Table 7.5 are the joint-tract targets optimized at SNR = 21 dB. Again we see that Δt has a slight impact on the target **f**, but a relatively larger impact on the target **g**. The reason we also consider targets of length two is because $f_2, f_3 << f_0, f_1$ in targets of length four. This has been observed in the previous section, and is elaborated herein. As presented in Section 7.4, the complexity of the joint-track channel detection algorithm is

$O(2^{L+L'})$. Although it is too high for a channel with $L = L' = 4$, it is low for a channel with $L = L' = 2$, as low as that of a single-track detection for a channel with $L = 4$.

We first use the targets of length four in simulations. In addition, two other cases $(\Delta t, \Delta t')$ = $(0.25T,0)$ and $(0.5T,0)$ are also considered. Plotted in Fig. 7.8 are the simulation results. Performance losses caused by $(\Delta t, \Delta t')$ are smaller than those for

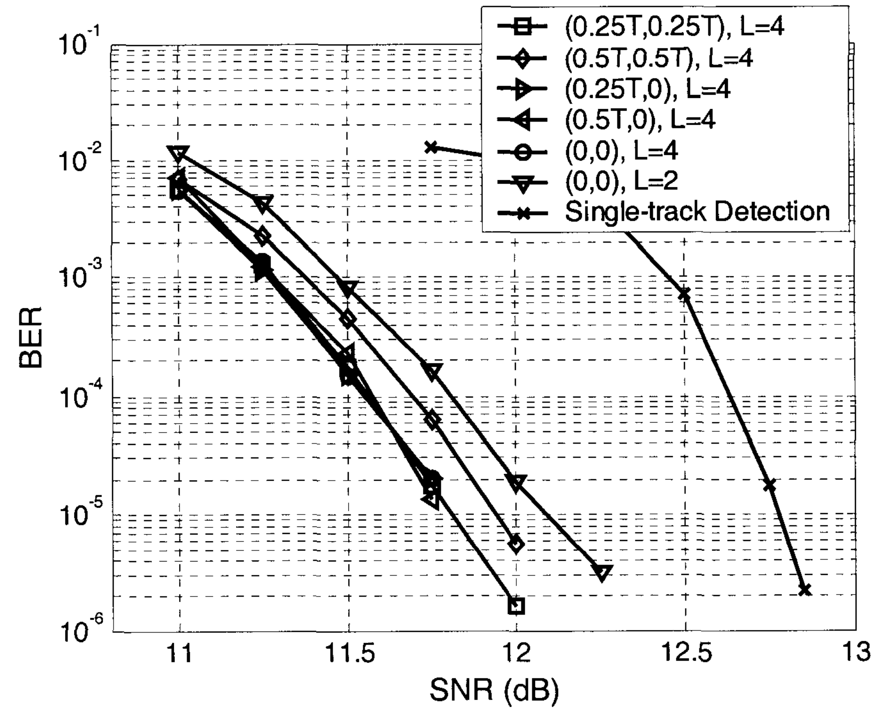


Fig. 7.8. BER simulation results for joint-track detection of ITI-channels with $\lambda = 0.2$ and 10% AWGN plus 90% media noise.

AWGN channels, therefore, $\Delta t'$ can be ignored and it is assumed zero for equalization. For this same reason, we only consider targets of length two with $(\Delta t, \Delta t') = (0,0)$ in the simulations. More interesting, it is seen that targets of length two perform just slightly worse, about 0.25 dB, than targets of length four. This is a very important fact, making the joint-track detection realizable.

Chapter 8

Conclusions and Future Work

Signal processing and LDPC coding techniques aiming at mitigating physical impairments, such as electronic/media noise, ISI, erasure and ITI, were investigated in this dissertation.

8.1 Conclusions

First, only ISI and electronic noise were considered. The goal is to design good LDPC codes for such channels. Two approaches are available – simulation and density evolution. While simulation can be used for any particular finite-length code, density evolution is only valid asymptotically for an ensemble of codes.

In this dissertation, we first compared several types of regular codes, namely, random codes and structured codes, e.g., FG codes and array codes. Generally speaking, random codes have very good performance. However, some structured codes, if selected carefully, can also perform well, and furthermore, they have much lower encoding complexity.

Density evolution provides a more systematic way of designing LDPC codes. First a good degree function is searched using a nonlinear process; then a particular code with this degree function is constructed. Numerical results indicated that irregular codes better than regular codes can be obtained.

Next, we considered erasures in magnetic recording, which can be caused by either TA or MD. It was shown by several means, including information rate analysis, density evolution and simulation, that such erasures must be detected. Therefore, erasure detection algorithms were studied and used in LDPC coded channels. Moreover, these erasures can be corrected by LDPC codes. The correction capability of LDPC codes was analyzed with a modified density evolution algorithm.

Thirdly, a new technique, called SNR mismatch, was introduced. This technique, as analyzed by density evolution as well as by simulation, can improve the performance of LDPC coded MRCs substantially. For a given practical system, an optimum value for SNR mismatch can be determined by simulation and used in practical applications. This SNR mismatch was used throughout this dissertation for LDPC coded MRCs but not for LDPC coded ideal PR channels.

Finally, special attention was paid to ITI, a very critical issue for high-density PMRCs. If this ITI can be mitigated effectively, higher track density can be achieved. Two models, the single-track model and the joint-track model, were used. It turned out that joint-track detection algorithms, especially the joint-BCJR detection algorithm, can improve performance when the ITI is severe.

This dissertation includes the following original contributions.

- Modification of the density evolution algorithm to make it applicable to LDPC coded PR channels. SNR threshold can be computed and BER can be estimated.
- Further modification of the density evolution algorithm to include other impairments,

such as erasures, detected or undetected.

- Improvement and proposal of several erasure detection algorithms, for both LMRCs and PMRCs.
- Investigation of the SNR mismatch. This investigation figures out why the SNR mismatch is happening and how to use it. The SNR mismatch technique has potential for use in practical systems.
- Study of channel detection for ITI in PMRC. This study intends to find out the best attainable performance of a practical system and to optimize the track density.

8.2 Future Work

There are still a number of questions to be explored.

- Evaluation of LDPC codes on MRCs at a very small BER. Practical systems require BER of 10^{-15}. But such small BER is beyond the capability of computer simulation. Density evolution can estimate such low level BER, however, it is only valid for asymptotic and ensemble of codes.
- Design of good finite-length LDPC codes for MRCs. Density evolution only begins to address this issue. A possible approach is to combine density evolution and the extrinsic information transfer (EXIT) chart [65].
- Design of LDPC codes for long erasures. How to balance the scenarios of no erasure and long erasures is an open problem.
- Physical modeling of ITI in PMRCs. This has to be done for connecting the ITI model with the physical parameters of practical systems.
- Reduced-complexity implementation of the joint-track detection. The joint-BCJR

detection algorithm is too complex. A tradeoff of acceptable performance and complexity must be made. Iterative soft MMSE equalization (replacing the BCJR channel detection algorithm), e.g., the turbo aided equalization [66], is low in complexity; however it cannot be used because of its poor performance. One possible scheme is an improved joint-MLSD algorithm, in which multiple most probable paths are selected such that the approximation in (7.23) is more accurate.

- Design of ECCs, particularly LDPC codes, for PMRC with ITI. The ITI may change the channel characteristics of the PMRC. Therefore, new ECCs may need to be designed.

Bibliography

[1] J. Bergmans, *Digital Baseband Transmission and Recording*. Boston, MA: Kluwer Academic Publishers, 1996.

[2] H. Song, *Applications of Iterative Decoding to Magnetic Recording Channels*, Ph.D. Dissertation, The University of Oklahoma, Norman, OK, 2002.

[3] R.W. Wood, J. Miles, and T. Olson, "Recording technologies for terabit per square inch systems," *IEEE Trans. Magn.*, vol. 38, pp. 1711-1718, July 2002.

[4] R. Wood, Y. Sonobe, Z. Jin, and B. Wilson, "Perpendicular recording: the promise and problems," *J. Magnetism Magn. Materials*, vol. 235, pp. 1-9, Oct. 2001.

[5] H. Sawaguchi, Y. Nishida, H. Takano, H. Aoi, "Performance analysis of modified PRML channels for perpendicular recording systems," *J. Magnetism Magn. Materials*, vol. 235, pp. 265-272, Oct. 2001.

[6] P. Kovintavewat, I. Ozgunes, E. Kurtas, J. Barry, and S.W. McLaughlin, "Generalized partial-response targets for perpendicular recording with jitter noise," *IEEE Trans. Magn.*, vol. 38, pp. 2340-2342, Sept. 2002.

[7] W. Jiang, M. Williams, N. Smith, W. Weresin, K. Kuroki, Y. Ikeda, K. Takano, G. Khera and R. Wood, "Investigations on adjacent-track interference in perpendicular recording systems," *IEEE Trans. Magn.*, vol. 39, pp. 1891-1896, July 2002.

[8] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," in *Proc. IEEE Global Telecommun. Conf.*, 1989, 1690-1686.

[9] 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. 20, pp. 284-287, Mar. 1974.

[10] C. Berrou, A. Glavieux, and P. Thitimahshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. IEEE Int. Conf. Commun.*, 1993, pp. 1064-1070.

[11] T. Mittelholzer, A. Dholakia, and E. Eleftheriou, "Reduced-complexity decoding of low density parity check codes for generalized partial response channels," *IEEE Trans. Magn.*, vol. 31, pp. 721-728, Mar. 2001.

[12] J. D. Coker, J.D, E. Eleftheriou, R. L. Galbraith, and W. Hirt, "Noise-predictive maximum likelihood (NPML) detection", *IEEE Trans. Magn.*, vol. 34, pp. 110-117, Jan. 1998.

[13] S. K. Nair, H. Shafiee, and J. Moon, "Equalization and detection in storage channels," *IEEE Trans. Magn.*, vol. 32, pp. 5206-5217, Sept. 1996.

[14] K. R. Narayanan, "Effect of precoding on the convergence of turbo equalization for partial response channels," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 686-698, Apr. 2001.

[15] A. Ghrayeb and W. E. Ryan, "Concatenated coding and iterative SOVA decoding with PR4 signaling," in *Proc. IEEE Int. Conf. Commun.*, 2000, pp. 697-601.

[16] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proc. IEEE Int. Conf. Commun.*, 1995, pp. 1009-1013.

[17] Y. Li, B. Vucetic, and Y. Sato, "Optimum soft-output detection for channels with

intersymbol interference," *IEEE Trans. Inform. Theory*, vol. 41, pp. 704-713, May 1995.

[18] T. Nishiya, K. Tsukano, T. Hirai, T. Nara, and S. Mita, "Turbo-EEPRML: An EEPR4 channel with an error-correcting post-processor designed for 16/17 rate quasi-MTR code," in *Proc. IEEE Global Telecommun. Conf.*, 1998, pp. 2706-2711.

[19] R. G. Gallager, *Low-Density Parity-Check Codes*, Cambridge, Massachusetts: MIT Press, 1963.

[20] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 46, pp. 399-431, Mar. 1999.

[21] S.-Y. Chung, G. D. Forney,Jr., T. J. Richardson and R. L. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, pp. 58-60, Feb. 2001.

[22] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electron. Lett.*, vol. 33, pp. 457-458, Mar. 1997.

[23] H. Song, R. M. Todd, and J. R. Cruz, "Applications of low-density parity-check codes to magnetic recording channels," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 918-923, May 2001.

[24] J. Fan, A. Friedmann, E. Kurtas, and S. McLaughlin, "Low density parity check codes for magnetic recording," in *Proc. Thirty-Seventh Allerton Conf. Commun., Control, and Computing,* 1999.

[25] Y. Kou, S. Lin and M. Fossorier, "Low-density parity-check codes based on finite geometries: a rediscovery and new results," *IEEE Trans. Inform. Theory.* vol. 47, pp. 2711-2736, Nov. 2001.

[26] J. Fan, "Array codes as LDPC codes," in *Proc. 2nd Int. Symp. Turbo Codes and Related Topics*, 2000, pp. 543-546.

[27] W. Tan and J. R. Cruz, "Array codes for erasure correction in magnetic recording channels," *IEEE Trans. Magn.*, vol. 39, pp. 2579-2581, Sept. 2002.

[28] M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. 27, pp. 533-547, Sept. 1981.

[29] M. P. C. Fossorier, M. Mihaljevic and H. Imai, "Reduced complexity iterative decoding of low-density parity-check codes based on belief propagation," *IEEE Trans. Commun.,* vol. 47, pp. 673-680, May 1999.

[30] C. Douillard, M. Jezequel, C. Berrou, A. Picart, P. Didier, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," *European Trans. Telecommun.*, vol. 6, Sept./Oct. 1995, pp. 507-511.

[31] D. J. C. MacKay's web site, http://wol.ra.phy.cam.ac.uk/mackay/codes/EN/C/.

[32] T. J. Richardson and R. L. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 638-656, Feb. 2001.

[33] W. Tan, R. M. Todd and J. R. Cruz, "Bounds for LDPC codes over partial response channels," *IEEE Trans. Magn.*, vol. 38, pp. 2310-2312, Sept. 2002.

[34] E. Eleftheriou and S. Olcer, "G.gen: LDPC codes for G.dmt.bis and G.lite.bis," *ITU Telecommunication Standardization Sector*, Study Group 15, Temporary Document CF-060, Clearwater, Florida, Jan. 2001.

[35] M. C. Davey and D. J. C. MacKay, "Low density parity check codes over GF(q)," *IEEE Commun. Letters*, vol. 2, No. 6, pp. 165-167, June 1998.

[36] M. C. Davey and D. J. C. MacKay, "Low density parity check codes over GF(q)," in *Proc. IEEE Information Theory Workshop*, June 1998, pp. 70-71.

[37] H. Song and J. R. Cruz, "Reduced complexity decoding of q-ary LDPC codes for magnetic recording channels," *IEEE Trans. Magn.,* vol. 39, 1081-1087, Mar. 2003.

[38] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity -check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599-618, Feb. 2001.

[39] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi and D. A. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. Inform. Theory*, vol. 47, pp. 585-598, Feb. 2001.

[40] A. Kavcic, X. Ma and M. Mitzenmacher, "Binary intersymbol interference channels: Gallager codes, density evolution and code performance bounds", *IEEE Trans. Inform. Theory*, vol. 49, pp. 1636-1652, July 2003.

[41] S.-Y. Chung, T. J. Richardson and R. L. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. Inform. Theory*, vol. 47, pp. 657-670, Feb. 2001.

[42] W. Tan and J. R. Cruz, "Performance evaluation of low-density parity-check codes on partial response channels using density evolution," to appear in *IEEE Trans. Commun.*, Aug. 2004.

[43] D. Arnold and H. A. Loeliger, "On the information rate of binary-input channels with memory," in *Proc. IEEE Int. Conf. Commun.*, 2001, pp. 2692-2695.

[44] T. J. Richardson, M. A. Shokrollahi and R. L. Urbanke, "Design of capacity-approaching low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 619-637, Feb. 2001.

[45] 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 J. Select. Areas Commun.*, vol. 19, pp. 924-934, May 2001.

[46] EPFL Communications Theory Lab website, http://lthcwww.epfl.ch/research /index.php.

[47] N. Varnica and A. Kavcic, "Optimized LDPC codes for partial response channels," *IEEE Commun. Lett.,* vol. 7, pp. 168-170, Apr. 2003.

[48] R. M. Todd, *Design of Low-Density Parity-Check Codes for Magnetic Recording Channels*, Ph.D. Dissertation, The University of Oklahoma, Norman, OK, 2003.

[49] M. Yang and W. E. Ryan, "Performance of (quasi-)cyclic LDPC codes in noise bursts on the EPR4 channel," in *Proc. IEEE Global Telecommun. Conf.*, 2001, vol. 5, pp. 2961-2965.

[50] H. Xia and J. R. Cruz, "On the performance of soft Reed-Solomon decoding for magnetic recording channels with erasures," *IEEE Trans. Magn.,* vol. 39, pp. 2576-2578, Sept. 2003.

[51] W. Tan and J. R. Cruz, "Analyzing LDPC codes for partial response channels with erasures using density evolution," to appear in *IEEE Trans. Magn.*

[52] T. Morita, Y. Sato and T. Sugawara, "ECC-less LDPC coding for magnetic recording channels," *IEEE Trans. Magn.*, vol. 38, pp. 2304-2306, Sept. 2002.

[53] W. Tan, H. Xia, and J. R. Cruz, "Erasure detection algorithms for magnetic recording channels," to appear in *IEEE Trans. Magn.,* July 2004.

[54] T. A. Summers and S. G. Wilson, "SNR mismatch and online estimation in turbo decoding," *IEEE Trans. Commun*, vol. 46, pp. 421-423, April 1998.

[55] A. Worm, P. Hoeher, and N. Wehn, "Turbo-decoding without SNR estimation," *IEEE Commun. Lett.*, vol. 4, pp. 193-195, June 2000.

[56] W. Tan and J. R. Cruz, "Signal-to-noise mismatch for low-density parity-check coded magnetic recording channels," *IEEE Trans. Magn.*, vol. 40, pp. 498-506, Mar. 2004.

[57] A. Thangaraj and S. W. McLaughlin, "Thresholds and scheduling for LDPC-coded partial response channels," *IEEE Trans. Magn.*, vol. 38, pp. 2307-2309, Sept. 2002.

[58] W. L. Abbott, J. M. Cioffi, and H. K. Thapar, "Performance of digital magnetic recording with equalization and offtrack interference," *IEEE Trans. Magn.*, vol. 27, pp. 705-716, Jan. 1991.

[59] B. G. Roh, S. U. Lee, J. Moon, and Y. Chen, "Single-head detection in the presence of offtrack interference," in *Proc. IEEE Global Telecommun. Conf.*, 1999, pp.955-959.

[60] B. G. Roh, S. U. Lee, J. Moon, and Y. Chen, "Single-head/single-track detection in interfering tracks," *IEEE Trans. Magn.*, vol. 38, pp.1830-1838, July 2002.

[61] M. P. Vea and J. M. F. Moura, "Magnetic recording channel with intertrack interference," *IEEE Trans. Magn.*, vol. 27, pp.4834-4836, Nov. 1991.

[62] W. Tan and J. R. Cruz, "Detection algorithms for perpendicular recording channels with intertrack interference," in *Digests 7th Perpendicular Magn. Recording Conf.*, May/June 2004, Sendai, Japan.

[63] W. Tan and J. R. Cruz, "Optimizing perpendicular recording channel detection for intertrack interference," submitted to *IEEE Global Telecommun. Conf.*, 2004.

[64] Z. Zhang, T. M. Duman, and M. E. Kurtas, "On information rates of single-track and muiti-track magnetic recording channels with intertrack interference," in Proc. *IEEE Int. Symp. Inform. Theory*, p. 163, 2003.

[65] S. ten Brink, "Convergence of iterative decoding," *Electron. Lett.*, vol. 35, pp. 806-808, May 1999.

[66] Z.-N. Wu, *Coding and Iterative Detection for Magnetic Recording Channels.* New York, N.Y.: Kluwer Academic Publishers, 2000.