UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

APPLICATIONS OF ITERATIVE DECODING TO MAGNETIC RECORDING CHANNELS

A Dissertation

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

degree of

Doctor of Philosophy

By

HONGXIN SONG
Norman, Oklahoma
2002

UMI Number: 3038030

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

APPLICATIONS OF ITERATIVE DECODING TO MAGNETIC RECORDING CHANNELS

A Dissertation APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY

Arthur M Breipohl

Kevin Grant

Fred D. Lee

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisor, Dr. J. R. Cruz, for his mentoring, guidance, support and encouragement throughout my Ph.D. studies. It is his technical expertise and wisdom that helped make this work possible. His valuable teaching and advice, both academic and non-academic, will benefit me forever.

Secondly, I would also like to thank Dr. Arthur Breipohl, Dr. John Cheung, Dr. Kevin Grasse and Dr. Fred Lee for their support and their time serving on my committee.

I also thank the fellow students in the Communication Signal Processing Laboratory (CSPLab) for the helpful discussions and the friendship.

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

To my wonderful wife Shuqing and our lovely son Jeffrey

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Magnetic recoding channels (MRCs) are subject to noise contamination and error-correcting codes (ECCs) are used to keep the integrity of the data. Conventionally, hard decoding of the ECCs is performed. In this dissertation, systems using soft iterative decoding techniques are presented and their improved performance is established.

Three coding schemes are investigated for magnetic recording systems. Firstly, block turbo codes, including product codes and parallel block turbo codes, are considered on MRCs. Product codes with other types of component codes are briefly discussed.

Secondly, binary low-density parity-check (LDPC) codes are proposed for MRCs. Random binary LDPC codes, finite-geometry LDPC codes and irregular LDPC codes are considered. With belief propagation decoding, LDPC systems are shown to have superior performance over current Reed-Solomon (RS) systems at the range possible for computer simulation. The issue of RS-LDPC concatenation is also addressed.

Finally, Q-ary LDPC (Q-LDPC) codes are considered for MRCs. Belief propagation decoding for binary LDPC codes is extended to Q-LDPC codes and a reduced-complexity decoding algorithm for Q-LDPC codes is developed. Q-LDPC coded systems perform very well with random noise as well as with burst erasures. Simulations show that Q-LDPC systems outperform RS systems.

1 INTRODUCTION TO MAGNETIC RECORDING SYSTEMS

In magnetic recording (MR) systems, the information is recorded on the media in the form of a remnant magnetic field, from which the information is retrieved. In this work, the interest is in digital MR systems, in which the information is discrete (0 or 1). For simplicity, digital MR systems are referred simply as MR systems in the following. Particular focus is on hard disk drives (HDDs), though the results can be easily applied to other MR systems. HDD applications require very reliable data retrieval, with typical failure rates below 10^{-14}.

In this chapter, the basics of MR systems are first introduced in Section 1.1, and coding requirements for magnetic recording channels are discussed in Section 1.2. The problems addressed in this dissertation are stated in Section 1.3. Finally, the organization of this dissertation is presented in Section 1.4.

1.1 Magnetic recording systems

1.1.1 Digital magnetic recording basics

The most fundamental components of an MR system are heads and media. When a current representing the data sequence passes through the head coil, it generates a magnetic flux pattern. The flux is used to magnetize the media. During the write process, the head and media are in relative movement, and the temporal signal pattern is recorded on the media in a spatial pattern. The media is magnetized along with the direction of the relative movement. During the replay process, the head picks up the flux change from the magnetized media with the same relative movement between the head and media [1]-[3].

The degree to which the media can be magnetized is in general continuous until it reaches saturation. However, in digital magnetic recording, saturation recording is used, i.e., the write magnetic flux is sufficiently strong that it saturates the media. Therefore, only two magnetization patterns, saturated magnetization along and opposite to the relative movement, are possible. The correspondence of the data, write current and the magnetization pattern on the media is illustrated in Figure 1.1.

As illustrated in Figure 1.1, the change of sign (transition) in the written data, either from +1 to −1 or from −1 to +1 (in non-return-zero format), corresponds to the transition of the magnetization direction on the media.

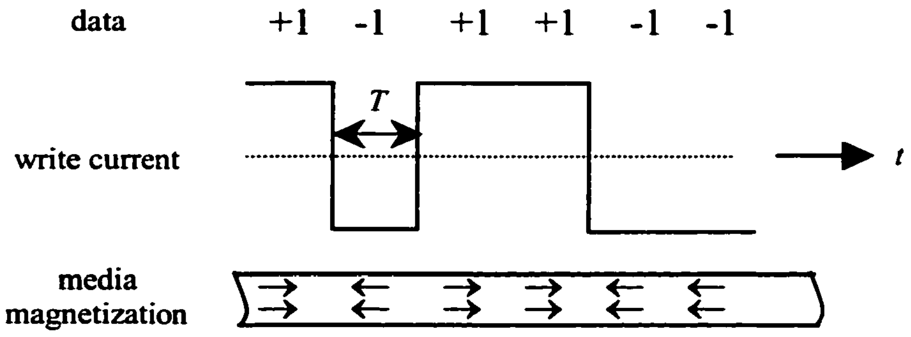


Figure 1.1. Illustration of data, write current and magnetization pattern.

A high-level description of conventional signal processing and coding for an MR system is shown in Figure 1.2 [1].

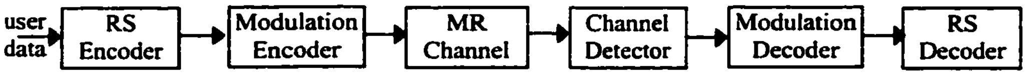


Figure 1.2. Conventional MR system.

In this system, the components shown are:

- MR channel. The MR channel includes the preprocessing before writing, the characteristics of the interaction of both write and read heads and media, and signal processing at the readback front end. The output of the MR channel is a symbol-rate sequence for the channel detector to estimate the recorded data.
- Channel detector. MR channels are often equalized to partial response (PR) targets, and partial-response maximum-likelihood (PRML) detection is often used to cope with the inter-symbol-interference (ISI). Conventionally, the PRML detector can be implemented by the Viterbi Algorithm (VA).
- Modulation coding. A modulation code is used to facilitate timing recovery and gain control. In addition, it may also improve the free distance [1].
- RS coding. Reed-Solomon (RS) codes are used to correct errors that may occur in the channel detector. Conventionally hard decoding is performed.

In addition to the components shown in Figure 1.2, a timing recovery system and a gain control system are necessary [1]. However, throughout this thesis, perfect timing and gain control is assumed.

In the sequel, the components shown in Figure 1.2 are discussed in detail.

1.1.2 Channel response

The write process and read back process can be characterized by the transition response (step response), which is the readback waveform when a single positive transition (from −1 to +1, at time 0) is written. The step response can be measured by averaging the read back signals of many isolated written rising (or falling) transitions. In hard disk recording systems, it is found that the Lorentzian pulse [1],

$$s(t) = \frac{V_0}{1+\left(\frac{2t}{PW50}\right)^2}, \tag{1.1}$$

where $PW50$ is the pulse width at half of the peak height and V_0 is a power normalization constant, is a good model for the step response.

Given the channel bit duration period T, as shown in Figure 1.1, the normalized channel density (or simply channel density) is defined as

$$S_c = \frac{PW50}{T}. \tag{1.2}$$

Normalized user density S_u can be defined similarly for user bit duration period T_u.

Because of its closeness to actual waveforms and its simplicity, the Lorentzian pulse is a popular step response model, and it will be used for the most part in this dissertation. In this dissertation, another step response model, the Lorentzian-Gaussian response [4][1],

[1] It is understood that the waveform in Eq. (4) in [4] is not Gaussian.

$$s(t)=\frac{1}{2}\left[\frac{V_0}{1+\left(\frac{2t}{PW50}\right)^2}+V_0\exp(-\frac{4\ln 2}{PW50^2}t^2)\right], \tag{1.3}$$

will also be considered. Figure 1.3 shows the Lorentzian and Lorentzian-Gaussian pulses, both at channel density 3.0.

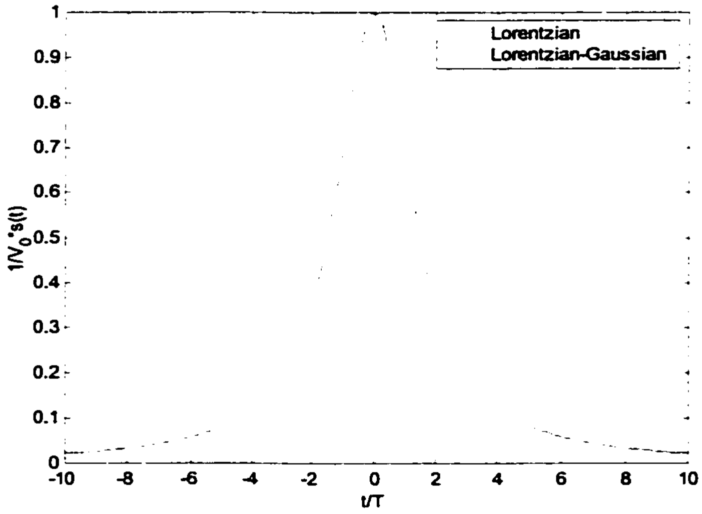


Figure 1.3. Lorentzian and Lorentzian-Gaussian waveforms at channel density 3.0.

The response of a positive transition (at time 0) followed by an immediate negative transition T seconds later, is denoted as the *dibit response* (also known as *pulse response*), and given by

$$h(t)=s(t)-s(t-T). \tag{1.4}$$

The dibit response is equivalent to the pulse response (continuous time) of the system. Shown in Figure 1.4 is the dibit response corresponding to a Lorentzian step response. It can be seen that the step response lasts well beyond the bit duration, resulting in ISI from neighboring bits.

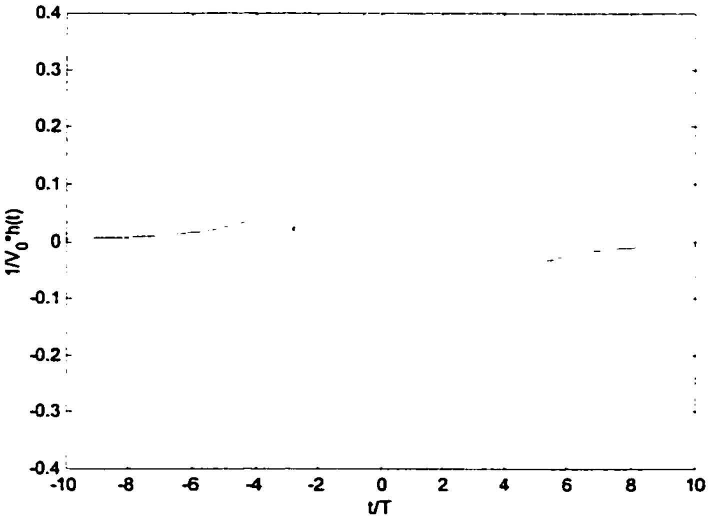


Figure 1.4. Pulse response at channel density 3.0 with Lorentzian step response.

The read back signal from the recorded data stream in which x_k is the k-th recorded data bit is

$$\sum_k x_k h(t-kT). \tag{1.5}$$

However, (1.5) is valid only when the channel is noise-free and without distortion, which is never true.

1.1.3 Noise

The noises that corrupt the read back signal fall into the following categories [1],[3].

- Electronic noise. Electronic noise is the thermal noise that is present in any communication system, and is usually assumed to be additive white Gaussian.

Electronic noise is assumed to be data independent, and in a different read the noise realization is different. This property makes it possible to improve the performance by re-reading.

- Media noise. Media noise is caused by the granularity of the magnetic material in the media, and can be divided into transition noise and particulate noise. The granularity results in non-smooth transition edges when transitions are written onto the media. For a given transition written on the media, the read back waveform may be shifted in time scale, resulting in position jitter, and the width of the read back waveform may also change, resulting in width variation. Transition noise only occurs at the locations of transitions. Any read back yields the same media noise. In addition, the granularity causes the discontinuity in the media, which results in particulate media noise.
- Nonlinear distortions and interferences.
- Abnormalities due to disk defects and thermal asperities. A disk defect results in the degradation or total loss of the signal, and a thermal asperity results in read back signal overflow.

In this dissertation, all nonlinear distortions are neglected, as well as the pulse width variation. Only electronic noise will be considered in Chapter 3, but position jitter noise, as well as disk defects and thermal asperities, will be considered in Chapters 4 and 5.

Position jitter noise

As mentioned above, position jitter is caused by the finite granularity of the media. This results in the randomness of the effective transition location, or in other words, the step response $s(t)$ that would be centered at $t = 0$ is now centered at $t = \Delta t$. Using a first-order approximation $s(t+\Delta t) \approx s(t) + \Delta t \frac{d}{dt} s(t)$, this shift can be modeled as shown in Figure 1.5. Δt is a random variable, whose probability density function (pdf) determines the jitter noise power per transition [5]. The pdf of each transition jitter will be assumed as independent and Gaussian.

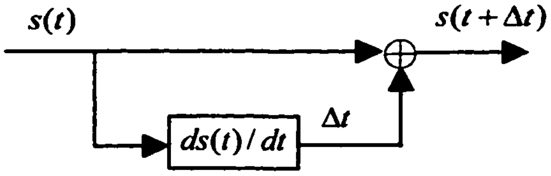


Figure 1.5. First-order position jitter model.

1.1.4 Channel model

As shown in Figure 1.4, the MR channel is an ISI channel. When the noise source is additive white Gaussian noise (AWGN) only, the channel is linear and the read back waveform is simply the superposition of the responses of the individual bits. It is well known that the optimal detector for this channel is a matched filter followed by a symbol rate sampler, which in turn is followed by a maximum likelihood sequence estimator [6]. The symbol rate sampler produces sufficient statistics. The maximum likelihood sequence estimator can be implemented by a Viterbi detector. The optimal detector is shown in Figure 1.6.

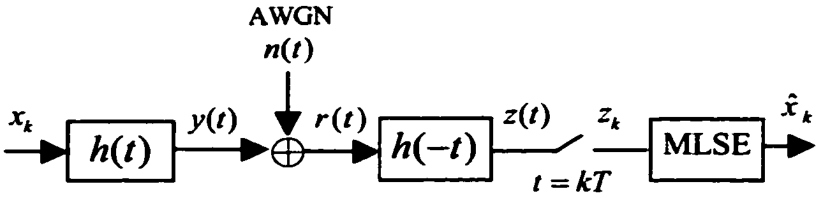


Figure 1.6. Optimal detector for ISI channel with AWGN.

This system yields the matched filter bound when the input is a single bit sequence (one shot). It is shown in [7] that the required VA needs 2^L-state (L is the length of ISI). Since the ISI is long (or even infinite) at high density, the complexity is very high.

In practice, sub-optimal detection is often used to trade-off complexity for performance. As shown in Figure 1.7, the matched filter $h(-t)$ is replace by a realizable low-pass filter $p(t)$ and the sampled output of the filtered dipulse response in (1.4) is equalized to some predetermined PR target with a finite-impulse-response (FIR) filter, and then a VA is used for detection [1],[8]. Noise correlation is simply neglected at the VA, which degrades the performance.

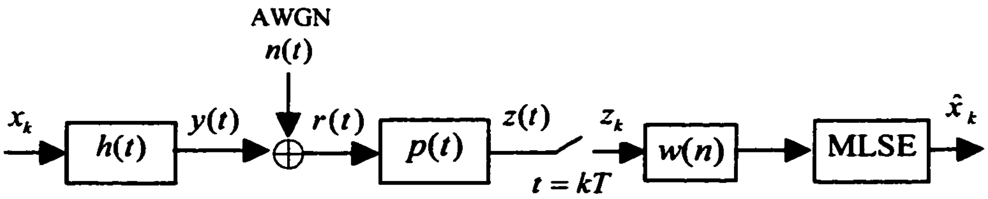


Figure 1.7. Sub-optimal detection with PR equalization.

The popular PR targets are PR4 with $H_{PR}(D) = (1-D)(1+D)$, EPR4 with $H_{PR}(D) = (1-D)(1+D)^2$, EEPR4 with $H_{PR}(D) = (1-D)(1+D)^3$, and some modified EEPR4 (ME^{2}PR4) such as the one with $H_{PR}(D) = (1-D)(1+D)(5+4D+2D^2)$ [9].

The discrete-time model of Figure 1.7 is shown in Figure 1.8, in which the sequence is expressed using D-transforms [1],[10].

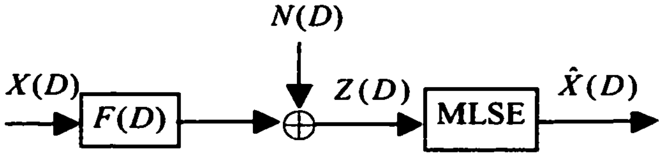


Figure 1.8. All discrete-time channel model.

In Figure 1.8, the D-transform of a sequence $\{a_k\}$ is defined as $A(D) = \sum_k a_k D^k$, where D denotes delay by one sample. Thus the following can be found,

$$F(D) = Q(D)W(D), \tag{1.6}$$

where $Q(D)$ is the D-transform of $\{q_k = h(t) * p(t)|_{t=kT}\}$. The correlation of the noise component at the MLSE is

$$R_n(D) = \sigma^2_{n(t)} R_{p_k}(D) W(D) W(D^{-1}), \tag{1.7}$$

where $R_{p_k}(D)$ is the D-transform of $\{r_{p_k} = r_{p(t),p(t)}(t)|_{t=kT}\}$. $R_n(D)$ is colored due to the low pass filter and the equalizer.

The transfer function $W(D)$ of the equalizer depends on the equalization criterion. For zero-force equalization,

$$W(D) = \frac{H_{PR}(D)}{Q(D)}. \tag{1.8}$$

For minimum mean square error (MMSE) equalization, the transfer function $W(D)$ is given by

$$\mathbf{R}_{z,z}\mathbf{w} = \mathbf{R}_{z,x}\mathbf{h}_{PR} \tag{1.9}$$

where $\mathbf{R}_{z,z}$ is the auto-correlation matrix of $\{z_k\}$ in Figure 1.7, $\mathbf{R}_{z,x}$ is the cross-correlation matrix between $\{z_k\}$ and $\{x_k\}$, $\mathbf{h}_{PR}$ represents the target PR response and $\mathbf{w}$ is the coefficients of the FIR equalization filter in vector form. When the noise is not data-dependent, $\mathbf{R}_{z,z}$ can be decomposed into two components, one due to the signal and the other due to the noise.

If the noise $n(t)$ is not white, the detection scheme shown in Figure 1.6 is not optimal. However, the sub-optimal detection scheme in Figure 1.7 is still used.

1.1.5 Modulation coding and precoding

MR systems need a modulation code to facilitate gain control and timing recovery, and may also be able to improve the free Euclidean distance [1],[11]. As the readback system only responds to the transition, the modulation code must have a constraint on the maximum run of non-transitions.

Two classes of modulation codes are often used, run-length-limited (RLL) codes and maximum-transition-run (MTR) codes [11]-[13]. In a *(d, k)* RLL code, the minimum run length of non-transitions between two transitions is *d* and the maximal run length of non-transitions is *k*. In a *(d, G/I)* RLL code, in addition to the constraint that the minimum non-transition run length between two transitions is *d*, the maximum run length of non-transitions is *G* in the global sequence and is *I* in both two-way interleaved sequences. For MTR codes, in addition to the minimum run length of non-transitions, the maximum run length of transitions is also constrained.

Typically, each code bit in these modulation codes is a transition marker (non-return-to-zero-inverted, NRZI). Bit 0 means a non-transition and bit 1 means a transition. The transition mark needs to be translated into non-transitions or transitions before the write current is generated to serve the designated purposes.

A precoder converts the NRZI input data into non-return-to-zero (NRZ) format. Unless the code design is done in NRZ space, the precoder is necessary in the system. Notice that the precoder has to match the code space. The following two examples illustrate this idea.

Exmaple 1. Suppose the modulation code is a (*d,k*) RLL code with $k=6$. A codeword section is a=[1 0 0 0 0 0 0 1], and the precoder $\frac{1}{1\oplus D\oplus D^3}$ is used. Suppose the precoder state is [1 0 0], then the output sequence b=[1 0 0 0 0 0 0 0 0 1], which has 0-run of length eight, violates the constraint.

Example 2. In a quasi-MTR code, the maximum length of 1-run is three. A codeword section in NRZI format is a=[0 0 0 1 1 1 0 1 0], and the precoder $\frac{1}{1\oplus D\oplus D^3}$ is used to transform this section into NRZ form. Suppose the precoder state is [0 0 0], then the output sequence b=[0 0 0 0 0 0 1 0 1 0 1 0 0], which has six consecutive transitions, which violates the constraint.

It is shown in [14]-[17] that the precoder has great impact on the performance. Notice that the assumption here is that we know how to design a constrained code that works with the precoder [14],[17].

1.1.6 Signal-to-noise ratio (SNR) definition

The proper definition of SNR is critical to compare the performance of a system with different coding schemes. In MR systems, the SNR definition should depend on the media-head characteristics and noise level, but invariant to coding [18],[19]. In other words, ideally, coding does not change the SNR with this definition.

In the following, SNR definitions for the AWGN channel and for channels with both AWGN and position jitter noise are discussed.

Assuming the step response $s(t)$ of the channel (Lorentzian, Gaussian, Lorentzian-Gaussian) is known, the SNR is defined as

$$SNR = \frac{V_0^2 / T}{\text{in-band additive noise power} + \text{jitter noise power (tone pattern)}}, \qquad (1.10)$$

where V_0 is the peak amplitude of $s(t)$. Assuming AWGN with single-sided power spectrum density N_0, the in-band additive noise power is $\frac{N_0}{2}\frac{1}{T}$, where T is the symbol duration. The jitter noise power under tone pattern is calculated as

$$\frac{1}{T}\int_{-\infty}^{+\infty} E\left\{n_j \frac{d}{dt}s(t)\right\}^2 dt = \frac{1}{T}\sigma_j^{\,2}\int_{-\infty}^{+\infty}\left|\frac{d}{dt}s(t)\right|^2 dt,$$

where n_j is a random variable describing the amplitude of the jitter and $\sigma_j^{\,2}$ is the variance of the jitter. Therefore (1.10) becomes

$$SNR = \frac{V_0^2}{\frac{N_0}{2} + \sigma_j^2 \int_{-\infty}^{+\infty} \left| \frac{d}{dt} s(t) \right|^2 dt}. \qquad (1.11)$$

Notice that the definition in (1.10) is in the form of $\frac{\text{energy per bit}}{\text{total noise power}}$, except that instead of using the energy in the dibit response $h(t)$, the energy in the step response $s(t)$ is used. This makes the definition invariant with the channel density, which changes when coding is applied.

Usually, the percentage of the jitter noise power in the total noise power is described as β, i.e.

$$\beta = \frac{\sigma_j^2 \int_{-\infty}^{+\infty} \left| \frac{d}{dt} s(t) \right|^2 dt}{\frac{N_0}{2} + \sigma_j^2 \int_{-\infty}^{+\infty} \left| \frac{d}{dt} s(t) \right|^2 dt}.$$

It can be seen from (1.11) that in order to determine σ_j^2 for a given SNR, the only thing to be calculated is $\int_{-\infty}^{+\infty} \left| \frac{d}{dt} s(t) \right|^2 dt$. For Lorentzian and Gaussian responses, closed forms can be found, but not for the Lorentzian-Gaussian response.

Lorentzian response

For the Lorentzian response, the integral can be solved in closed form as

$$\int_{-\infty}^{+\infty} \left| \frac{d}{dt} s(t) \right|^2 dt = V_0^2 \frac{\pi}{2} \frac{1}{PW50}.$$

The variance of jitter is given by $\sigma_j^2 = \beta \frac{N_0}{2} T \frac{1}{V_0^2} \frac{2}{\pi} PW50 = const \cdot PW50$. The first-order jitter noise model, which is the first-order derivative of $s(t)$ multiplied by a random variable with variance σ_j^2, only depends on $\frac{N_0}{2}$, β and $\frac{PW50}{T}$.

Gaussian response

A Gaussian pulse is given by $h(t) = V_0 \exp\left(\frac{-4\ln 2}{PW50^2} t^2\right)$. After a lengthy process the integral can be solved in closed form as

$$\int_{-\infty}^{\infty} \left|\frac{d}{dt} h(t)\right|^2 dt = V_0^2 \sqrt{2\pi \ln 2} \frac{1}{PW50}.$$

Again, the first-order jitter noise model is the first-order derivative of $s(t)$, multiplied by a random variable with variance σ_j^2. This model only depends on $\frac{N_0}{2}$, β and $\frac{PW50}{T}$.

Lorentzian-Gaussian response

A Lorentzian-Gaussian pulse is given by the average of a Lorentzian pulse and a Gaussian pulse. Unfortunately, no closed form result of the integral is available. However, it can be obtained by numerical integration. Although it cannot be proved, it is reasonable to believe that this integral also depends on $\frac{N_0}{2}$, β and $\frac{PW50}{T}$, but not on $PW50$ alone.

1.2 Coding requirement

Code rate R is defined as

$$R = \frac{S_u}{S_c}, \tag{1.12}$$

and therefore the introduction of coding increases the channel density S_c. Assuming AWGN, the effective SNR when the matched filter bound is achieved is $\frac{\int_{-\infty}^{+\infty} |h(t)|^2 \, dt}{N_0 / 2}$.

Codes for MR systems require a very high code rate for two reasons. First, for a given user density S_u, a low code rate R means high channel density S_c or equivalently high data rate need to be processed. Current HDDs operate at very high data rate, and further increased channel density makes the implementation difficult. The second reason lies on the peculiar coding penalty of MR channels.

In [19] and [20], it is shown that on Lorentzian channels with AWGN the effective SNR is

$$\frac{2\int_{-\infty}^{+\infty} |s(t)|^2 \, dt}{N_0 / 2} \frac{R^2}{S_u^2 + R^2} \approx \frac{2\int_{-\infty}^{+\infty} |s(t)|^2 \, dt}{N_0 / 2} \frac{R^2}{S_u^2}, \tag{1.13}$$

where the approximation holds for high S_u. This means the code rate loss is proportional to R^2. For comparison, the coding loss is linear with R for typical communication channels. The quadratic code rate loss implies that at code rate lower than some threshold the gain from lowering down the code rate will not compensate for the code rate loss.

1.3 Problem statement

With increasing areal densities, the noise and distortion present in MR systems become more severe. Apart from the efforts on improving the materials, such as the media and heads, error correction coding (ECC) has been a major focus of research [21],[22].

Possible ways to improve the system performance by coding techniques can be categorized into three groups. One approach is to keep the system architecture unchanged but to replace the channel detector and the RS decoder with soft decoding versions [23]-[25]. However the potential gain is limited. Another approach is to apply additional coding in the system [26]. The third approach is to replace the RS codes with other more powerful ECCs.

The breakthrough of iterative decoding [27]-[29] brought about a new generation of communication systems [21],[22]. The possibility of applying iterative decoding to magnetic recording is the issue that is investigated in this work. The following questions are to be answered: what kind of codes can be used, how do they perform in MR systems, and what is their complexity.

1.4 Overview of the dissertation

This dissertation considers applying turbo codes, block turbo codes, binary low-density parity-check (LDPC) codes, and Q-ary LDPC (Q-LDPC) codes to MR systems.

Chapter 2 gives an overview of soft iterative decoding for magnetic recording channels. Turbo coded systems and serially concatenated systems are briefly introduced to illustrate iterative decoding, and their shortcomings highlighted.

In Chapter 3, block turbo coded systems are considered. A product coded system with BCH codes as component codes is described. The channel detector is implemented with a soft-output Viterbi algorithm. The performance is compared with that of uncoded channels without considering RS coding. Product coded systems with other types of component codes are also discussed.

In Chapter 4, an LDPC coded system is proposed, in which the channel detector is a maximal *a posteriori* (MAP) detector and the LDPC decoder is a belief propagation (BP) decoder. Performance evaluation of the system without considering the RS coding suggests a substantial gain over uncoded systems. Concatenation of LDPC and RS codes does not improve the performance over the LDPC-only system, leading to the proposal of using the LDPC code to replace the RS code in Figure 1.2. LDPC systems in the presence of disk defects and thermal asperities (TAs) are considered. Also considered in this system are finite-geometry LDPC codes.

While realizing that binary LDPC codes provide little hope to cope with disk defects and thermal asperities, a Q-LDPC coded system is proposed in Chapter 5. The performance of a Q-LDPC coded system is evaluated in a pure electronic noise environment, as well as in environments with disk defects and thermal asperities. Media noise is also considered for this system. Furthermore, performance comparison between Q-LDPC systems and current RS coded systems is conducted. Array codes, which are non-binary codes, are also discussed.

Finally, conclusions are made in Chapter 6. Some discussion is also presented for further consideration of coding for MR systems.

2 ITERATIVE DECODING FOR MAGNETIC RECORDING CHANNELS

In this chapter, iterative decoding of concatenated codes is reviewed, as well as turbo equalization for MR channels. As examples, the application of turbo codes and serial turbo codes to MR channels is briefly introduced and discussed.

2.1 Iterative decoding on partial response (PR) channels

2.1.1 Coding for PR channels

A PR channel coded using an error-correcting code (ECC) C with information block length k, as shown in Figure 2.1, can be regarded in total as a system. Denote the noiseless channel output vectors as $\mathbf{y}$ and $\mathbf{y}'$ when the information vectors $\mathbf{x}$ and $\mathbf{x}'$ are sent, respectively. A union bound of the system error rate assuming AWGN is given in [31], as

$$\begin{aligned} P_e &\le \sum_{\mathbf{x}} \sum_{\mathbf{x}' \ne \mathbf{x}} Q\left(\sqrt{\frac{\|\mathbf{y}' - \mathbf{y}\|^2}{N_0/2}}\right)\left(\frac{1}{2}\right)^{d^H(\mathbf{x},\mathbf{x}')} \\ &\approx const \cdot Q\left(\sqrt{\frac{d_{\min}^2}{N_0/2}}\right) \end{aligned}, \qquad (2.1)$$

in which $d^H(\mathbf{x},\mathbf{x}')$ denotes the Hamming distance between the two binary vectors and $d_{\min}$ is the minimum Euclidean distance between any pair of $\mathbf{y}$ and $\mathbf{y}'$. The second equality is valid at high SNR. Therefore, to improve the ML system performance, one needs to improve $d_{\min}$.

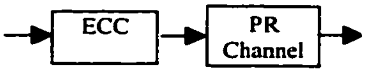


Figure 2.1. A PR channel with ECC.

On PR channels of the form $1-D^n$, it is proved that $d_{\min}^2 \geq 8\left\lfloor \frac{d_{\min}^H+1}{2} \right\rfloor$, in which $d_{\min}^H$ is the minimum Hamming distance of code C and $\lfloor \bullet \rfloor$ denotes the integer part [29]. A PR4 channel is of the form $1-D^2$. Therefore, to improve the ML system performance on these PR channels, one needs to improve $d_{\min}^H$, i.e., more powerful codes are needed. We assume that by improving $d_{\min}^H$ the ML system performance is also improved on other PR channels rather than of the form $1-D^n$.

2.1.2 Turbo equalization

Maximum likelihood sequence estimation (MLSE) of the system can be carried out if the ECC shown in Figure 2.1 has a well-defined trellis, e.g., a convolutional code, since the overall system has a well-defined trellis. Otherwise, MLSE is practically impossible, and sub-optimal decoding may be used.

One common implementation of a suboptimal estimation is as follows. Denote the received noisy vector as $\mathbf{r}$ for information vector $\mathbf{x}$ sent. The channel detector passes its decision to the ECC decoder, and the output from the ECC decoder is final. However, the ECC decision may help the channel detector to make a better estimation. Thus the feedback scheme in Figure 2.2 is introduced. Iterative decoding for PR channels is sometimes called *turbo equalization* [32].

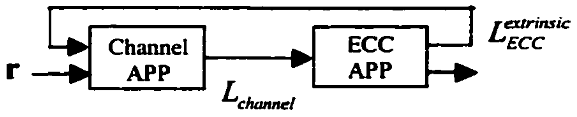


Figure 2.2. Turbo equalization.

To fully exploit the advantage of the feedback, the channel detector and the ECC decoder need to produce soft decisions (*soft information*). A device that is able to generate posterior probability information is called an *a posteriori* probability (APP) module. If for the i-th bit x_i in the transmitted codeword $\mathbf{x}$, we express all reliability information in terms of the log-likelihood-ratio (LLR) $\log\frac{P(x_i=+1)}{P(x_i=-1)}$, an APP module outputs posterior probability $L(x_i)$ based on the intrinsic LLR $L^{intrinsic}(x_i)$ and prior LLR $L^{prior}(x_i)$,

$$L(x_i) = L^{prior}(x_i) + L^{intrinsic}(x_i) + L^{extrinsic}(x_i). \tag{2.2}$$

The Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [34] and the soft output Viterbi algorithm (SOVA) [35] outlined in Appendix A can be used as APP modules for PR channels and convolutional codes.

The channel APP takes $\mathbf{r}$ and the prior LLR from the ECC APP, and outputs the posterior LLR $L_{channel}$. The ECC APP takes $L_{channel}$ as the prior LLR. The extrinsic LLR of the ECC APP, $L_{ECC}^{extrinsic}$, is sent back to the channel APP as prior LLR.

2.1.3 Iterative decoding

For a binary linear code (n,k), the number of codewords with weight i is denoted as w_i. For the AWGN channel, the probability of codeword error is bounded by the union bound,

$$p_e \leq \sum_{d_{\min}}^{n} \frac{w_i}{2^k} Q\left(\sqrt{\frac{E_b}{N_0/2}}\right), \tag{2.3}$$

where E_b is the energy per bit and N_0 is the single-sided power spectrum density of the white noise [6]. It can be seen that the performance depends not only on $d_{\min}^{H}$ but also on the weight distribution of the code.

It is well known that code concatenation of two or more codes produces good codes [36]. Each individual code in a concatenated code is referred to as a *component code*. One example of a serially concatenated code is shown in Figure 2.7 if the PR channel is regarded as a rate-one convolutional code [14]. A concatenated code is a *serially concatenated code* if the subsequent component encoder(s) rely on the parity checks from previous component encoders. Therefore, the encoding of each component code must be done in some order. A concatenated code is a *parallel concatenated code* if each component encoder relies on the message only. Encoding of each component code can be done in parallel. All the codes we are interested in this work, including turbo codes, product codes, low-density parity-check codes and Q-ary low-density parity codes are all concatenated codes.

ML decoding for most error correction codes (including concatenated codes) is difficult or impossible due to its complexity, which is proportional to 2^k, therefore sub-optimal decoding is required.

Similar to turbo equalization, iterative decoding may improve the decoding performance of a concatenated code. In fact, if we think of the PR channel as a rate-one code, then turbo equalization in Figure 2.2 is just an example of iterative ECC decoding.

2.1.4 Combined iterative decoding and turbo equalization

Both turbo equalization and iterative ECC decoding are possible on PR channels. Therefore two loops may be operating in an iterative decoded PR system, as shown in Figure 2.3. The iteration loop involving the turbo equalization is called the *outer loop*, and the iteration loop inside the ECC decoder is called the *inner loop*. The inner loop iteration may be performed more than once for each outer loop iteration.

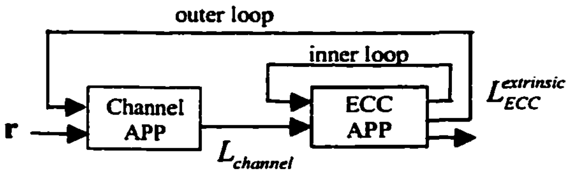


Figure 2.3. Outer and inner loops in an iterative decoding system.

2.1.5 Modulation codes for soft decoded magnetic recording systems

In a real MR system, a modulation code is needed. In Figure 1.2, the modulation code is placed immediately before the channel. This system diagram is in the top half of Figure 2.4, and the concatenation shown is called a *standard concatenation*. At the receiver side, a soft output modulation decoder is required to pass the soft information from the PR APP. Since the modulation code is typically non-systematic, the soft output modulation decoding is performed over the entire block.

To eliminate the need for a soft output modulation decoder, two *reversed concatenation* configurations can be used [37]. The first solution, shown in the lower half of Figure 2.4 is called the *modified concatenation with a second systematic coding technique*, in which two modulation codes are used [37],[39]. The message is first modulated by modulation code C_1, followed by the systematic ECC. The parity check bits of the ECC are then encoded by a systematic modulation code C_2. On the receiver side, the information pertinent to the parity check bits of C_2 is simply discarded. For a (0,k) type constraint, C_2 can be implemented by inserting a dummy bit for every k bits.

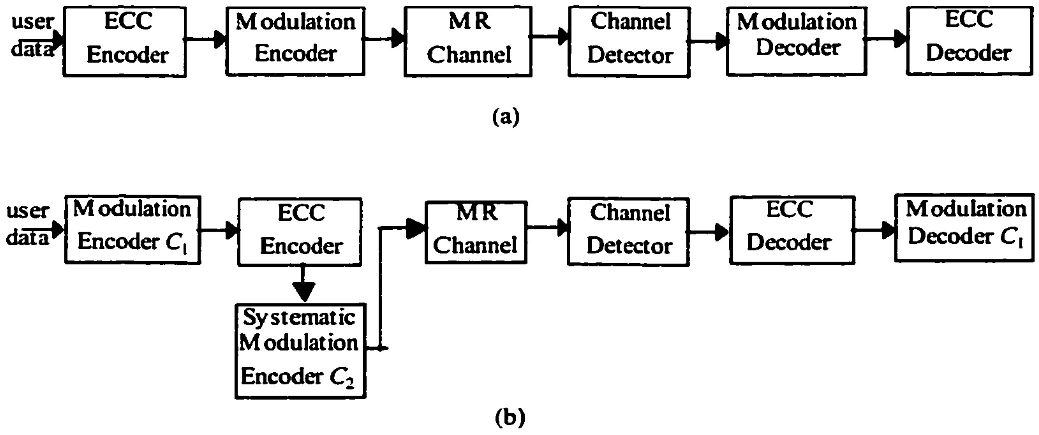


Figure 2.4. Concatenation of ECC and modulation code. (a) standard concatenation; (b) modified concatenation with a second systematic code.

An alternative way, the *bit insertion technique*, can be applied for a (0,*k*) type constraint [40]. The message is encoded by a (0,*k*-*a*) modulation code, where $a \geq 1$, followed by the ECC encoder. The parity check bits of the ECC are inserted into the output sequence of the modulation encoder. The resultant sequence still exhibits the (0,*k*) constraint.

For a certain range of ECC rates, the bit insertion technique yields higher overall code rate than the first technique [37]. However, the difference is minor. For instance, assume a capacity-achieving modulation code, a (0,6) constraint and a 16/17 ECC rate; the first technique yields an overall system code rate of 0.9266, while the bit insertion technique (with a=1) yields 0.9300.

2.2 Turbo codes for magnetic recording (MR) channels

2.2.1 Turbo codes

Conventional turbo codes are parallel concatenated codes with recursive systematic convolutional (RSC) codes as component codes (constituent codes) [27]. The structure is shown in Figure 2.5. The message sequence $\{x_k\}$ is directly passed through, and also encoded by encoder 1. The parity check sequence $\{x_k^{p_1}\}$ from encoder 1 is appended to the message. The message sequence $\{x_k\}$ is also encoded by encoder 2 after scrambled by an interleaver. The parity check sequence $\{x_k^{p_2}\}$

from encoder 2 is also appended to the message. The structure shown has code rate $1/3$. To improve the code rate, the parity check bits are punctured.

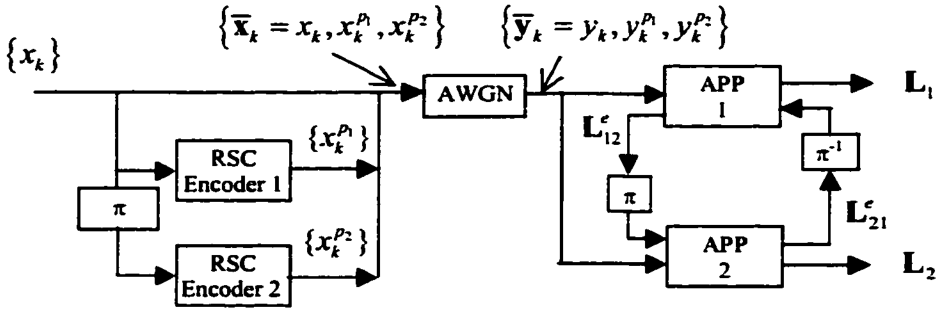


Figure 2.5. Turbo coding and decoding.

In order to have good distance properties, a random interleaver is needed. The resultant concatenated code has very few codewords with low weight, although the minimum distance $d_{\min}^{ff}$ is only two [41].

Suppose a codeword $\{\bar{\mathbf{x}}_k = x_k, x_k^{p_1}, x_k^{p_2}\}$ (± 1) is transmitted through an AWGN channel, and the received sequence is $\{\bar{\mathbf{y}}_k = y_k, y_k^{p_1}, y_k^{p_2}\}$, as shown in Figure 2.5. Decoding of the turbo code is accomplished in iterative fashion, and is also illustrated in Figure 2.5. Each component RSC code is decoded by an APP decoder. The BCJR algorithm is used for the APP decoders (see Appendix A).

The LLR of APP 1 is,

$$L_1(x_k) = L_c y_k + L_{21}^e(x_k) + L_{12}^e(x_k), \tag{2.4}$$

where L_c is a constant depending on the SNR. $L_c y_k$ is the intrinsic LLR. $L_{21}^e(x_k)$ is the prior LLR, which is obtained from APP 2. $L_{12}^e(x_k)$ is the extrinsic LLR, which will be sent to APP 2. Similarly for APP 2,

$$L_2(x_k) = L_c y_k + L_{12}^e(x_k) + L_{21}^e(x_k)\,. \tag{2.5}$$

The information exchange during iterative decoding is shown in Figure 2.5. The extrinsic information from one APP serves as the posterior information for the other. Turbo codes have been shown to have very good performance [27].

2.2.2 Turbo coded MR channels

With turbo equalization, the decoding of a turbo coded PR channel is shown in Figure 2.6 [22]. The feedback in the turbo equalization is $L_{ECC}^{extrinsic} = L_2 - L_{channel}$.

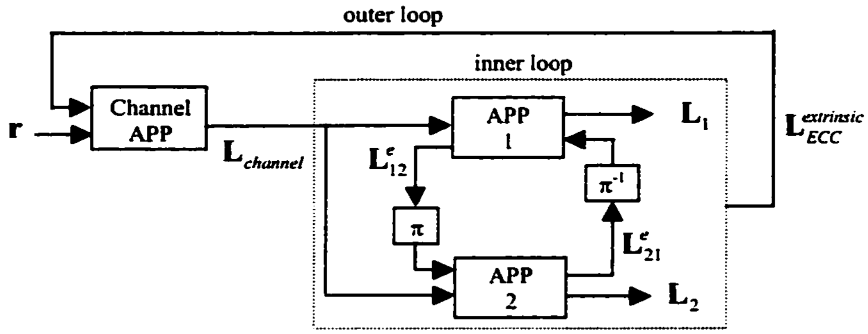


Figure 2.6. Iterative decoding of turbo coded PR channels.

It is shown in [21] and [22] that impressive gains can be obtained with turbo codes on MR channels with random noise [21]. In addition, turbo equalization is shown to provide about 0.5 dB gain [22]. However, turbo codes have some undesirable characteristics for an MR system:

1. High error floor. The minimum Hamming distance of turbo codes is two. The error floor due to the minimum weight codewords may show up at error rates higher than the operating rates of MR systems. To lower the error floor, codes with larger $d^H_{\min}$ are needed.
2. High decoding complexity. Turbo decoders use two BCJR APP decoders. The associated hardware complexity and computational complexity are very large, compared to some of the codes that are discussed in later chapters.
3. Ineffective for long burst errors. For long bursts due to disk defects and thermal asperities, a turbo coded system does not perform well [37]. The reason resides in that turbo codes are designed for random errors, not for burst errors.

2.3 Serial turbo codes for MR systems

Shown in Figure 2.7 is the turbo equalization between the PR channel and a recursive convolutional code separated by an interleaver [14]. The configuration is sometimes called *serial turbo coding*.

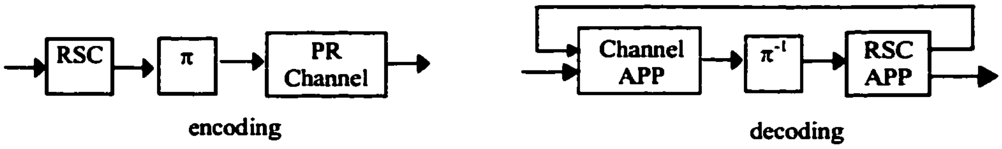


Figure 2.7. Serial concatenation of a PR channel and a convolutional code.

In this system, the turbo code shown in Figure 2.5, is replaced by a single convolutional code. Only the turbo equalization loop exists and is mandatory. Instead of three APPs needed in conventional turbo coded systems, only two are needed in this system.

The serial turbo system performs quite well at moderate bit error rates ($\sim 10^{-5}$) with random noise. However, it is shown in [37] that this system has higher error floor than turbo coded systems and the performance degrades rapidly with burst errors.

3 BLOCK TURBO CODES FOR MAGNETIC RECORDING CHANNELS

The component codes in the concatenated system shown in Figure 2.1 can be any codes in general. In the case of turbo codes, the component codes are recursive convolutional codes. Block turbo codes (BTC) are referred to as concatenated codes with block codes as component codes. Serially concatenated block codes are called *product codes* or *turbo product codes*; parallel concatenated block codes are called parallel BTCs.

Product codes have been shown to have near-optimum performance and lower decoding complexity than convolutional turbo codes for additive white Gaussian noise channels [28]. In this chapter we discuss the applicability of block turbo codes to partial-response equalized Lorentzian channels for magnetic recording. In particular, we consider the iterative decoding of product codes and parallel concatenated block codes. Simulation results show that both systems offer substantial gains over uncoded systems.

In Section 3.1, an introduction to BTCs is provided. The decoding of BTCs is described in Section 3.2. In Section 3.3, BTC coded MR systems with Hamming codes as component codes are investigated. In Section 3.4, other BTC coded MR systems are discussed and conclusions are made in Section 3.5.

3.1 Block turbo codes (BTCs)

3.1.1 Product codes

Given two linear systematic codes $C_1 = (N_1, K_1, d_1)$ and $C_2 = (N_2, K_2, d_2)$, where N_i, K_i, and d_i, $(i = 1,2)$ are the codeword length, number of information bits, and minimum Hamming distance, respectively. The construction of a product code with these codes as the component codes is shown in Figure 3.1.

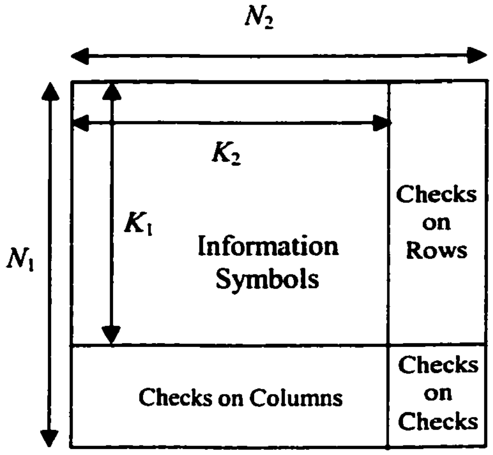


Figure 3.1. Product code.

The encoding can be carried out as follows: 1) place K_1K_2 information bits in an array of K_1 rows and K_2 columns, 2) encode the K_1 rows using C_2, and 3) encode the resultant N_2 columns using C_1. In the N_1 by N_2 array obtained, all the N_1 rows are codewords in C_2, and all the N_2 columns are codewords in C_1.

Encoding steps 2 and 3 cannot be performed in parallel. In other words, one has to be performed before the other. This is therefore a serial concatenation. The resultant product code $C_2 \otimes C_2$ is a $(N, K, d_{\min})$ code, where

$$\begin{aligned} N &= N_1 N_2 \\ K &= K_1 K_2 \\ d_{\min} &= d_1 d_2. \end{aligned} \tag{3.1}$$

The code rate R of the product is the product of the two component code rates, i.e.,

$$R = R_1 R_2 \tag{3.2}$$

where $R_i = K_i / N_i$, $i = 1,2$.

3.1.2 Parallel block turbo codes

As can be seen from (3.2), the code rate of a product code tends to be low. For a given block size, the code rate of a product code cannot be increased. To improve the code rate, parallel concatenated block codes can be used. The construction of parallel concatenated block codes is shown in Figure 3.2.

The encoding is also carried out in three steps. 1) Put $L_1 L_2$ message bits into a $L_1 \times L_2$ matrix form; 2) for the L_1 rows, encode each i_2 rows using C_1, which has $i_2 L_2$ information bits; 3) for the L_2 columns, encode each i_1 columns using C_2, which has $i_1 L_1$ information bits.

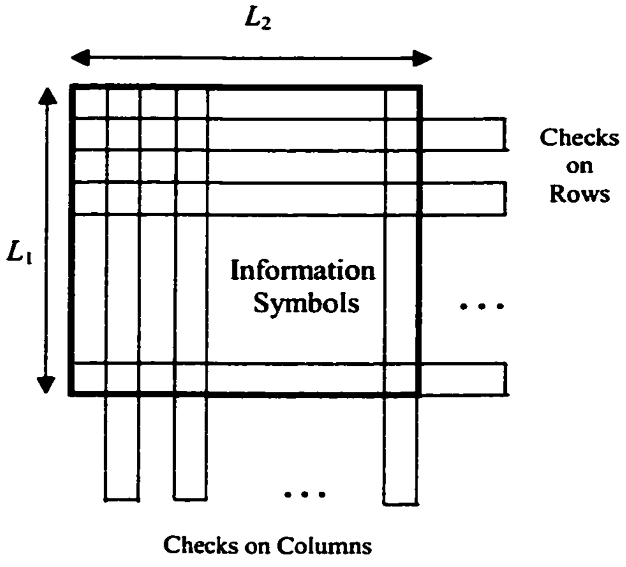


Figure 3.2. Construction of parallel concatenated block codes.

Steps 2 and 3 can be interchanged, and the resultant code is a parallel concatenated code. There is no check on checks in this code. The overall code rate R can be calculated as

$$R = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2} - 1} \tag{3.3}$$

3.2 Iterative soft decoding of block turbo codes

Iterative soft decoding of block turbo codes requires specifying the information exchange between the two component codes and specifying the soft decoding of each component code. In [28] an efficient iterative soft decoding is described, in which the component linear block codes are decoded based on the Chase algorithm [33].

3.2.1 Chase algorithm for block codes

For a linear block code $C = (N, K, d_{\min})$ on an inter-symbol-interference (ISI) free AWGN channel, the Chase algorithm (Chase II) [33] generates a set of codewords that are at a small Euclidean distance from the received vector.

Denote the transmitted codeword as $\mathbf{x}$ and the received vector as $\mathbf{r}$. The Chase algorithm is done in the following steps.

1. Generate a binary vector $\mathbf{h}$, such that h_i is the hard decision of r_i.
2. Find $p = \lfloor d_{\min}/2 \rfloor$ locations in $\mathbf{r}$ that have the least absolute, i.e., least reliable, values.
3. Generate 2^p binary vectors of length N in which the value can be 0 or 1 at the p bit locations found in step 2 and 0 at other bit locations. Each generated vector is called a test pattern.
4. For each test pattern $\mathbf{t}_k$ ($k = 1, \ldots, p$), hard decoding is performed on $\mathbf{h} + \mathbf{t}_k$. The output codeword is denoted as $\hat{\mathbf{x}}_k$. The set of all output codewords is denoted as Ω.
5. Find the codeword in Ω with smallest Euclidean distance from $\mathbf{r}$. This codeword is denoted as $\mathbf{d}$.

In short, the Chase algorithm finds the most likely codeword from all codewords that are within Hamming distance $2\lfloor d_{\min}/2 \rfloor$ from the received hard decision vector. More importantly, it generates a set of codewords that can be used to calculate the soft information.

3.2.2 Iterative decoding based on the Chase algorithm

In [28], Pyndiah developed a sub-optimal algorithm for soft decoding of a linear block code based on the Chase algorithm.

For the k-th bit in the codeword, this algorithm consists in finding two codewords $\mathbf{c}^{-1(k)}$ and $\mathbf{c}^{+1(k)}$, where $\mathbf{c}^{-1(k)}$ is the codeword in C having a -1 for the k-th bit which is at the minimum Euclidean distance from the received vector $\mathbf{r}$. Similarly, $\mathbf{c}^{+1(k)}$ is the codeword in C having a $+1$ for the k-th bit which is at the minimum Euclidean distance from the received vector $\mathbf{r}$.

At least one of these codewords, $\mathbf{c}^{-1(k)}$ or $\mathbf{c}^{+1(k)}$, must be available from the Chase algorithm. If both of them are available, Pyndiah shows that the soft output LLR for the k-th bit with hard decision d_k is approximately,

$$LLR'(d_k) = \frac{1}{2\sigma^2}\left(\left|\mathbf{r}-\mathbf{C}^{-1(k)}\right|^2 - \left|\mathbf{r}-\mathbf{C}^{+1(k)}\right|^2\right). \tag{3.4}$$

If only one of them is available, an estimate of the soft output is used. Let us expand $LLR'(d_j)$ as

$$LLR'(d_k) = \frac{2}{\sigma^2}\left(r_k + \sum_{l=1,l\neq k}^{n} r_l c_l^{+1(k)} p_l\right) \tag{3.5}$$

where

$$p_l = \begin{cases} 0, & \text{if } c_l^{+1(k)} = c_l^{-1(k)} \\ 1, & \text{if } c_l^{+1(k)} \neq c_l^{-1(k)} \end{cases}.$$

The second term of $LLR'(d_j)$ in (3.5) can be thought as extrinsic information, denoted as $\Lambda^e(d_k)$, which can be expressed as

$$LLR^e(d_k) = LLR'(d_k) - \frac{2}{\sigma^2} r_k . \tag{3.6}$$

When the soft output needs to be estimated, the following is used:

$$LLR^e(d_k) = \beta \frac{2}{\sigma^2} d_k \text{, with } \beta \geq 0 \tag{3.7}$$

where β is a weight factor introduced in [28].

Suppose a block turbo codeword is transmitted and the received matrix is $\mathbf{R}$. The extrinsic information $LLR^e(d_j)$ from one decoder can be used as prior information for the other decoder. By adding the extrinsic information, the equivalent channel vectors for the inputs of the two decoders are:

$$\mathbf{R}_2 = \mathbf{R} + \alpha \frac{\sigma^2}{2} LLR_1^e \tag{3.8}$$

and

$$\mathbf{R}_1 = \mathbf{R} + \alpha \frac{\sigma^2}{2} LLR_2^e \tag{3.9}$$

where α is also a weight factor introduced in [28].

By combining (3.4)-(3.9), iterative decoding of a BTC can be accomplished. Each iteration consists of a row decoding and a column decoding. Iterative decoding of a BTC is very much the same as in a convolutional turbo code, except for the weight factors α and β.

3.3 Application of extended Hamming BTCs to magnetic recording channels

Two systems are considered using block turbo codes [42]. A product code is applied to the equalized EPR4 channel in the first system. In the second system, a parallel block turbo code is applied to an equalized ME^2PR4 channel. The code rate of the first system is higher than that of the second system. For complexity reason, SOVA is used for the channel APP and no turbo equalization is performed.

3.3.1 Hamming BTCs on AWGN channels

In current hard disk drives, each sector has 4096 information bits. In the following, block turbo codes are considered that can contain one, or more, sectors in each codeword. Pyndiah's algorithm is applied to these codes.

The first code $\text{ExBCH}(72,64,4)^2$ is a product code whose component codes are both the extended shortened BCH code $(72,64,4)$. The $(72,64,4)$ code is obtained by shortening the BCH code (127,120,3) to (71,64,3), followed by adding an even parity check bit. The resultant product code is a (5184,4096,16) code and has code rate 0.79. The simulation results are shown in Figure 3.3. With 8 iterations, a 6.4-dB gain is obtained at a bit error rate (BER) 10^{-5}.

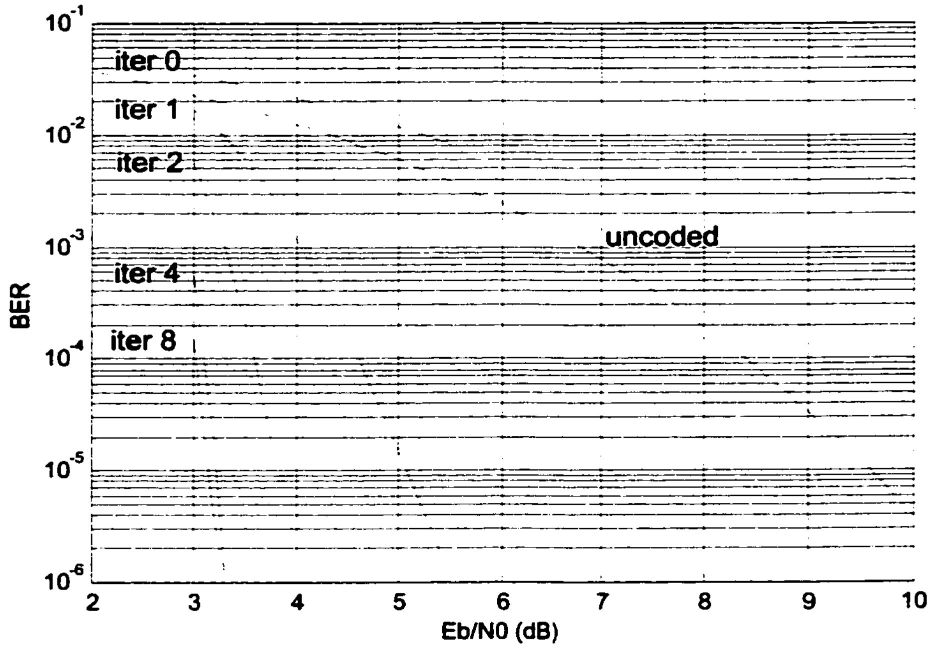


Figure 3.3. Performance of ExBCH$(72,64,4)^2$ on AWGN.

The second code ExBCH$(137,128,4)^2$ is also a product code whose component codes are both the extended shortened BCH code $(137,128,4)$. The $(137,128,4)$ code is obtained by shortening the BCH code (255,247,3) to (136,128,3), followed by adding an even parity check bit. The resultant product code has code rate 0.87. With 8 iterations, a 5.5-dB gain is obtained at a BER 10^{-5}.

The third code is a parallel concatenated code ExBCH$(207,198,4)^2$. The 4096 message bits are put into a 66×66 matrix, every three rows (or columns) are encoded by code (207,198,4) which is obtained by shortening BCH(255,247,3) to (206,198,3) followed by adding an even parity check bit. The resultant product code has code rate

0.92. The simulation results are shown in Figure 3.4. A 4.5-dB gain is obtained at a BER 10^{-5} with 4 iterations.

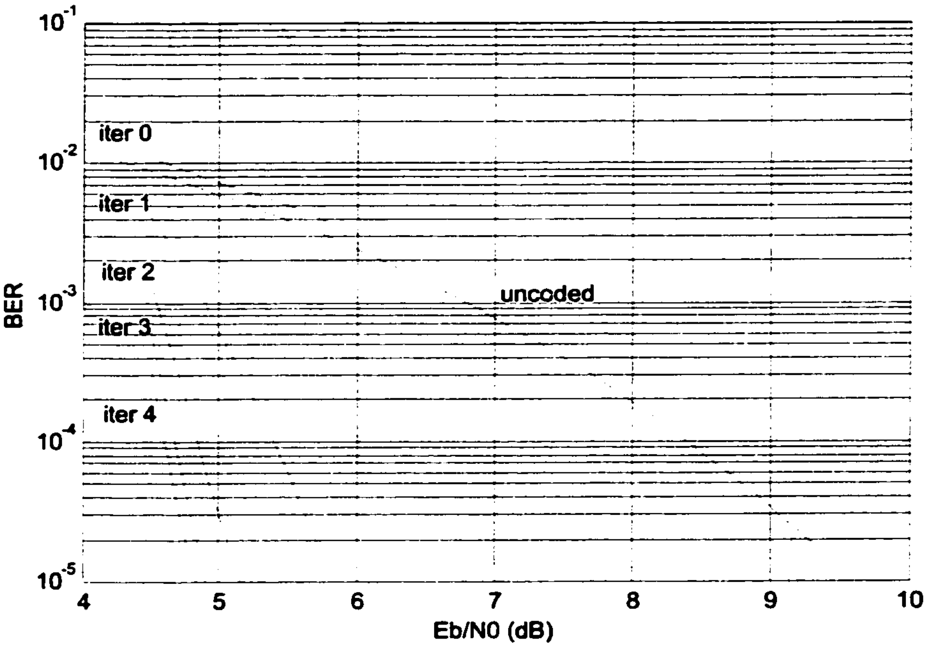


Figure 3.4. Performance of ExBCH(207,198,4)^2 on AWGN.

3.3.2 Product coded equalized EPR4 channel

The system diagram is depicted in Figure 3.5. The Lorentzian channel is equalized to the EPR4 channel response using an analog filter and a minimum mean-square error (MMSE) equalizer. The user data block is first encoded using a run-length-limited (RLL) code. After interleaving, the RLL coded block is further encoded using a BTC. Guard bits are inserted into the check bit sequence to enforce the RLL constraints before appending them to the RLL encoded user data block. On the decoder side, the interleaving is undone and guard bits are simply discarded. A rate 16/17 RLL code is used, and one guard bit (transition) is inserted every sixteen check bits.

The SOVA matches the $\frac{1}{1 \oplus D}$ precoded EPR4 channel response. The modified Chase algorithm II is used in our simulations. In the Chase algorithm II, for a received code vector only $\lfloor d_{\min}/2 \rfloor$ bit places are used to generate the test patterns, resulting in a total of $2^{\lfloor d_{\min}/2 \rfloor}$ patterns. In our simulations, we use $p = 4$ bit places to generate the test patterns, resulting in $2^p = 16$ patterns being searched for each received code vector. The parameters α's and β's in the iterative decoding are the same as in [28].

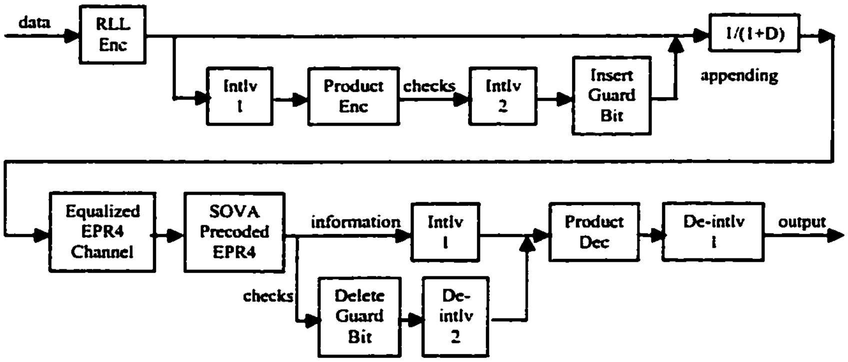


Figure 3.5. Diagram of an equalized EPR4 MR system with a product code.

User data block sizes 4096, 2×4096 and 3×4096 bits are considered, corresponding to one, two and three current standard sectors. The product codes used are listed in Table 3-I. All the component codes are obtained from the Hamming code (127,120,3) with a method similar to the one used in Section 3.3.1. Also listed are the overall system code rates (including the rate 16/17 RLL code).

TABLE 3-I

Product codes for MR systems and their overall system code rate

Sector size	Product code	System code rate (including rate 16/17 RLL code)	Coding gain after 8 iterations
1 sector per block	$(76,68,4)\otimes(72,64,4)$	0.74	3.4 dB
2 sectors per block	$(102,94,4)^2$	0.79	3.9 dB
3 sectors per block	$(126,118,4)^2$	0.81	4.0 dB

The systems are simulated at user density 2.22. Simulation results for the systems with block size one sector and two sectors are presented in Figure 3.6. BERs after 1, 2, 4 and 8 iterations are plotted. Also plotted is the BER for the uncoded EPR4-equalized channel (with a 16/17 RLL code).

For the one sector per block configuration, the product code provides a 3.4-dB gain over the uncoded channel at $\text{BER} = 10^{-5}$ after 8 iterations. With the two sectors per block configuration, the gain increases to 3.9 dB. With the three sectors per block configuration, the gain is 4.0 dB. These gains are also shown in Table 3-I.

These gains shown above are impressive. The code rates, however, are not high enough for MR systems, especially for the one sector per block configuration. Since the sector size is fixed, we have to seek code structures other than the product code. Parallel BTCs can achieve higher code rates.

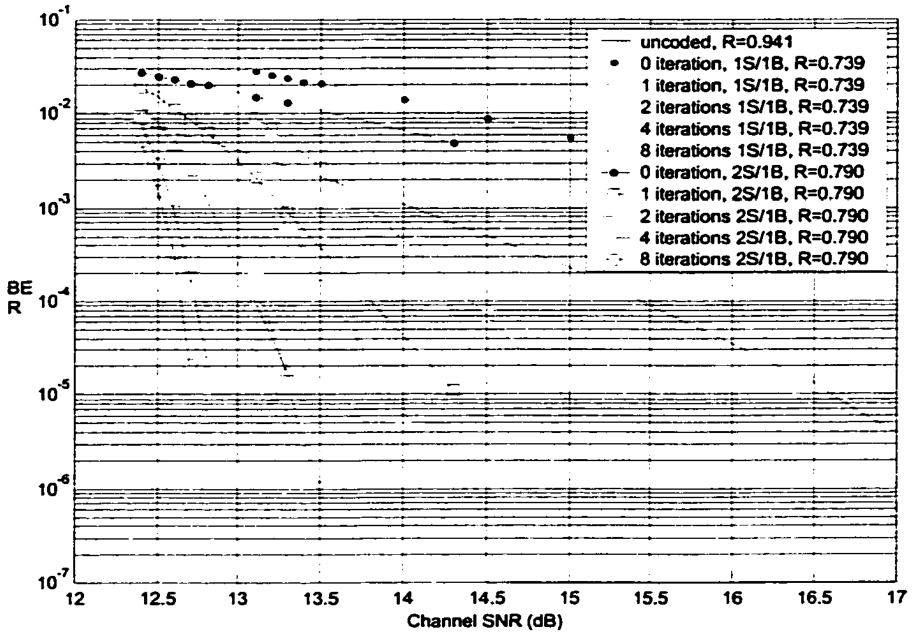


Figure 3.6. BER of equalized EPR4 with a BTC at user density of 2.22.

3.3.3 Parallel block turbo code on equalized ME^{2}PR4

Shown in Figure 3.7 is the proposed system diagram for an equalized ME^{2}PR4 channel with a parallel block turbo code. A quasi maximum-transition-run (MTR) code of rate 16/17 is used [9]. This code has a maximum of three consecutive transitions as well as an RLL constraint. The quasi MTR property reduces the number of occurrences of the dominant error events, $\pm(+1,-1,+1)$. To preserve the MTR property, guard bits are inserted into the check bits sequence.

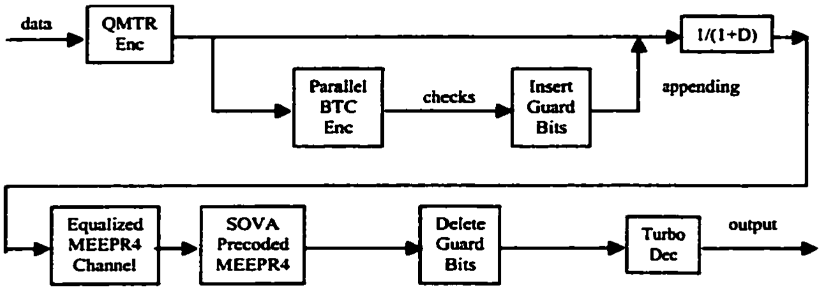


Figure 3.7. Diagram of an equalized ME^{2}PR4 MR system with a parallel BTC.

The block codes used here are the Hamming codes $(207,198,4)$ and $(216,207,4)$. They are obtained by first shortening the Hamming code $(255,247,3)$ and then extending them one bit by an even parity check. The size of the user data block is 4096 bits. The block after quasi-MTR encoding is placed into a 64×68 matrix. Three rows or three columns of the matrix are encoded using the above Hamming codes. One guard bit (transition) is inserted for every three check bits. The overall code rate for this system is 0.84. The soft-output Viterbi algorithm matched to the $\frac{1}{1\oplus D}$ precoded ME^{2}PR4 channel response is used as the soft channel decoder.

At a fixed user density of 2.51, after four iterations, the parallel block turbo coded system achieves 1.9-dB gain over QMTR coded ME^{2}PR4, or equivalently, the parallel BTC coded ME^{2}PR4 with QMTR code provides a 3.4-dB gain over EPR4 with a 16/17 RLL code.

The performance gain obtained from this parallel BTC is roughly the same as that from the one sector per block product code. Therefore, in terms of code rate, the parallel BTC is preferable to the product code. However, the disadvantage of a parallel BTC, compared with a product code, is its small minimum Hamming distance [36]. Therefore, parallel BTCs should have higher error floor than product codes.

The parameter p in the modified Chase algorithm II is also four, and the same sets of α's and β's as in the previous example are used. However, if $p = 6$ is used, an additional 0.4-dB gain can be obtained, showing that for long block codes, the Chase algorithm with $p = 4$ is far from optimum.

3.4 Application of other product codes to magnetic recording channels

In the previous section, BTCs with extended Hamming codes as component codes are considered. There have been attempts to use product codes with other types of component codes on MRCs. The component codes chosen tend to be either very weak or very powerful codes.

3.4.1 MR system with Reed-Solomon (RS) product codes

Product codes using RS codes as component codes are considered in [72]. Obtaining high code rates is more difficult than with Hamming product codes. Also, RS codes are not effective at correcting the random error events occurring on MR channels. The overall system performance is inferior to the extended Hamming product coded systems in Section 3.3.2.

3.4.2 MR systems with single-parity product codes

The product codes considered in [15] are the simplest ones, with single-parity check (SPC) codes as component codes. An SPC code can be described by a two-state trellis and soft-input-soft-output (SISO) APP decoding, e.g., BCJR or SOVA, can be easily implemented with high efficiency and good performance. Either BCJR or SOVA is superior to the Chase II algorithm. Also, SPC product codes are actually column weight 2 low-density parity-codes (see Chapter 4), and more efficient decoding algorithms (belief propagation algorithm) can be used instead of the iterative decoding algorithm described in Section 3.2.2.

The minimum distance of an SPC product code, however, is only 4 regardless of the code size. In addition, the multiplicity of minimum weight codewords is quite large. To improve the performance, multiple codewords per sector were used in [15]. This, however, does not improve the minimum distance.

SPC product codes were shown to provide large gain in MR systems over uncoded systems, better than the extended Hamming product codes in Section 3.2.2. This is largely due to the effective soft decoding of the component codes of the SPC product codes. The small minimum distance and large multiplicity cause noticeable error floor.

3.5 Conclusion and discussion

It was shown that extended Hamming product codes provide more than 3 dB gain over uncoded systems on MRCs, and a parallel BTC with extended Hamming codes as component codes provides similar coding gain but higher code rate. It is observed that the Chase II algorithm is far from the capacities of the component codes and more so as the component codes become more powerful. Therefore, using BTCs with more powerful component codes is not a good option for the following reasons. First, achieving high code rates is more difficult, and secondly, the decoding of the component codes using Chase II is less effective.

The option left is to use simple codes as component codes, as is done in SPC product codes. The possibility of simple and effective decoding and therefore good performance make this option even more attractive. However, the high error floor associated with SPC product codes is a disadvantage. In addition, as will be seen in next chapter, SPC product codes are surpassed by other low-density parity-check codes with essentially the same decoding complexity.

4 LOW-DENSITY PARITY-CHECK CODES FOR MAGNETIC RECORDING CHANNELS

Low-density parity-check (LDPC) codes were investigated initially by Gallager [43] and later by MacKay [29], [44]. They were shown to have performance close to or better than convolutional turbo codes on AWGN channels [45]-[47]. The following reasons make LDPC codes especially suitable for magnetic recording channels:

1. High-rate LDPC codes can be designed with very good performance.
2. Decoding complexity of LDPC codes is lower than that of turbo codes, and can be implemented in a highly parallel fashion. In addition, the iterative decoding algorithm has a built-in self-stopping mechanism.
3. LDPC codes have large minimum Hamming distance, hence low error floor.
4. The interleaver necessary for a turbo code system is not needed for an LDPC system.

In this chapter the use of LDPC codes on magnetic recording channels is investigated. Random regular LDPC codes, finite-geometry LDPC codes and irregular LDPC codes are considered. RS-LDPC concatenation is also investigated. Coding gain, decoding behavior and error distributions are reported.

This chapter is organized as follows. In Section 4.1 LDPC codes and their decoding are introduced. In Section 4.2 systems with random regular LDPC codes are described. In Section 4.3 finite-geometry LDPC coded systems are investigated. In Section 4.4 an irregular LDPC coded system is described and its performance evaluated. Conclusions are given in Section 4.5.

The LDPC codes investigated in this chapter are binary codes; the application of non-binary LDPC codes will be investigated in the following chapter.

4.1 Low-density parity-check (LDPC) codes

4.1.1 Code description

There are two equivalent ways to describe an LDPC code: matrix form and graph representation. An $M \times N$ matrix $\mathbf{H}$ with the elements from $\{0, 1\}$, defines a binary LDPC code as the N-tuple null space of $\mathbf{H}$, if it has sparse 1's.

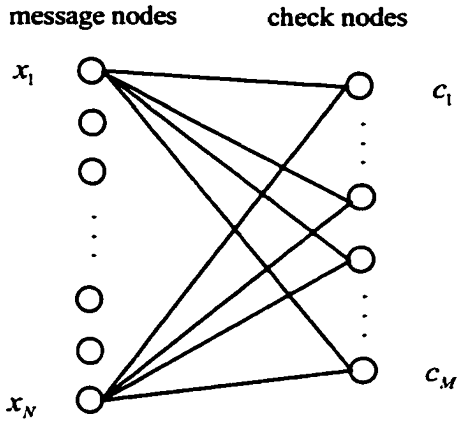


Figure 4.1. Graph representation of an LDPC code.

Alternatively, an LDPC code can be described by a bipartite graph [48], as shown in Figure 4.1. Each code bit is represented by a *message node* on the left, while each row of $\mathbf{H}$ is represented by a *check node* on the right. Message node x_i and check node c_j are connected by an edge if $H_{ji} = 1$.

The LDPC code defined above can be seen as a collection of M sub-codes [53]. Each sub-code, constrained by a row of $\mathbf{H}$, simply has even parity check. Decoding of LDPC codes relies on the decoding of the sub-codes.

4.1.2 Regular and irregular LDPC codes

For a regular LDPC code, the associated parity check matrix $\mathbf{H}$ has uniform column Hamming weights W_c and uniform row Hamming weights W_r. An $M \times N$ parity check matrix $\mathbf{H}$ with W_c and W_r defines a length N, rate $\frac{W_r - W_c}{W_r}$ code, provided that $\mathbf{H}$ is of full rank [43],[29].

For effective decoding, it is required that there be no cycles of length four in the graph [44], or equivalently, the overlap of any two columns of $\mathbf{H}$ is at most one. With this condition, there is a limit on N, the code length, for a given M and W_c [44],

$$N \leq \frac{M(M-1)}{W_c(W_c-1)}. \tag{4.1}$$

A simple explanation of (4.1) is as follows. Each column of $\mathbf{H}$ has W_c locations with ones. Choose two locations from the W_c locations, and form a weight two column vector. There can be $\binom{W_c}{2}$ different weight two column vectors. From N

columns of $\mathbf{H}$, there can be $N\binom{W_c}{2}$ distinct weight two column vectors. Since the maximum number of weight two column vectors of length M is $\binom{M}{2}$, we have $N\binom{W_c}{2} \le \binom{M}{2}$ and hence (4.1).

Equation (4.1) sets an upper limit on the code rate for a given number of information bits and a column weight W_c. Figure 4.2 shows the upper bound of code rate for regular LDPC codes with column weight three and four.

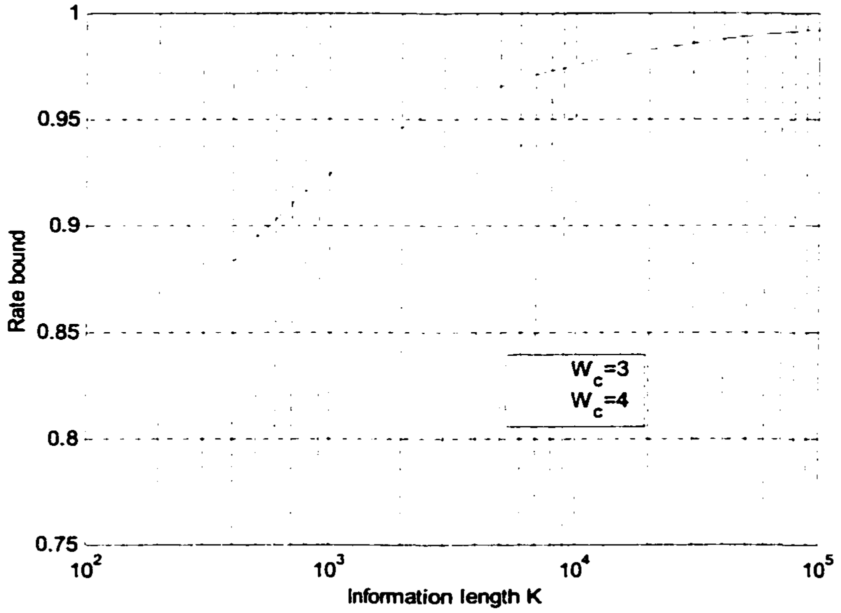


Figure 4.2. Upper bound on the code rate R for regular LDPC codes.

A special example of regular LDPC codes is the SPC product codes, discussed in Section 3.4.2. SPC product codes are LDPC codes with $W_c = 2$.

For an irregular LDPC code, the associated parity check matrix **H** has non-uniform row weights and/or column weights. In terms of the bipartite graph, as shown in Figure 4.1, the message nodes have different degrees and/or the check nodes have different degrees. A convenient way of describing the code graph is defining the *degree distribution polynomial pair*

$$\lambda(x) = \sum_i \lambda_i x^{i-1} \tag{4.2}$$

and

$$\rho(x) = \sum_i \rho_i x^{i-1} \tag{4.3}$$

where λ_i is the fraction of edges having left degree i and ρ_i is the fraction of edges having right degree i [45],[46].

Finding the weight enumerator function of an LDPC code is not trivial due to the high dimensionality. The ensemble average weight enumerating function, instead, is more practical to deal with. For regular LDPC codes, the ensemble average minimum distance increases linearly with N for fixed rate and W_c [43].

4.1.3 Belief propagation decoding algorithm

LDPC codes may be decoded by a majority-decoding method [50], which is a hard decoding technique. This is, however, very ineffective, since only $\left\lfloor \frac{W_c - 1}{2} \right\rfloor$ bit errors are correctable. Gallager's bit-flipping decoding [43] performs better, but is still far from the capacity.

BP algorithm

The belief propagation (BP) algorithm (also called *message passing* algorithm) can be used with an LDPC code [44]. It is an iterative SISO decoding method. McEliece *et al.* have shown that the BP algorithm and the turbo decoding algorithm are essentially the same algorithm [51]. The following description of the BP algorithm is based on [44].

Suppose a codeword $\mathbf{x} = [x_1, x_2, \cdots]$ such that **Hx=0** is transmitted and the receiver receives the following probabilities,

$$f_n^i = P(x_n = i), i = 0,1. \tag{4.4}$$

The set of bits participating in sub-code m is denoted by $\Phi(m) = \{n : H_{mn} = 1\}$. Similarly the set of sub-codes that bit n participates is denoted by $\Psi(n) = \{m : H_{mn} = 1\}$. In each decoding iteration, two main alternating steps are carried out, in which quantities q_{mn} and r_{mn} are iteratively updated:

$$r_{mn}^i = P(\text{sub-code } m \text{ is satisfied} | x_n = i, P(x_{n'}) = q_{mn'} \text{ for } n' \in \Phi(m) \backslash \{n\}) \tag{4.5}$$

$$q_{mn}^i = P(x_n = i | \Psi(n) \backslash \{m\}) \tag{4.6}$$

In (4.5), r_{mn}^i is the probability that sub-code m is satisfied conditioned on $x_n = i$ and $P(x_{n'}) = q_{mn'}$ for $n' \in \Phi(m) \backslash \{n\}$. In (4.6), q_{mn}^i is the probability that $x_n = i$, given the information obtained from $\Psi(n) \backslash \{m\}$.

Since each x_n is binary, $P(x_n = 1) - P(x_n = 0)$ contains all information of $P(x_n)$. Equation (4.5) can be calculated as

$$r^i_{mn} = \frac{1}{2}\left[1 + (-1)^i \prod_{n' \in \Phi(m) \backslash \{n\}} \delta q_{mn'}\right], \tag{4.7}$$

where

$$\delta q_{mn'} = q^1_{mn'} - q^0_{mn'}.$$

The BP algorithm using (4.7) is sometimes called *difference BP* algorithm [53]. Evaluating (4.5) is simply the SISO decoding of an even-parity sub-code. If the sum of code bits up to the current stage is defined as the state variable, this sub-code can be described with a two-state trellis diagram, originating from and converging to the zero state, as shown in Figure 4.3. SISO decoding, such as the MAP decoding algorithm, can be carried out on this trellis diagram, and the various decoding algorithms originate from the various SISO decoding methods.

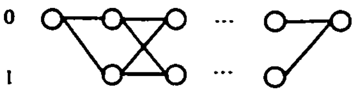


Figure 4.3. Trellis diagram of a sub-code.

In (4.6), it is always assumed that the information from different sub-codes is independent. Therefore, (4.6) can be rewritten as

$$q^i_{mn} = \alpha_{mn} f^i_n \prod_{m' \in \Psi(n) \backslash \{m\}} r^i_{m'n} \tag{4.8}$$

where α_{mn} is chosen such that $q^0_{mn} + q^1_{mn} = 1$.

The posterior probabilities $P(x_n = i \mid \mathbf{f}^i, \mathbf{Hx} = 0)$ are estimated as

$$q^i_n = \alpha_n f^i_n \prod_{m \in \Psi(n)} r^i_{mn} \tag{4.9}$$

where α_n is chosen such that $q_n^0 + q_n^1 = 1$.

A hard decision $\hat{x}_n$ is made based on q_n^i, namely $\hat{x}_n = 1$ if $q_n^1 > 0.5$. The syndrome $\mathbf{H}\hat{\mathbf{x}}$ is calculated. If $\mathbf{H}\hat{\mathbf{x}} = \mathbf{0}$ then the decoding algorithm halts and a valid codeword $\hat{\mathbf{x}}$ is obtained. Otherwise the algorithm repeats from the horizontal step for at most a certain prescribed number of iterations. If the maximum number of iterations are performed without a valid decoding, the algorithm declares an error. Whether or not the decoding is valid, the final $\mathbf{q}^i = [q_1^i, q_2^i, \ldots]$ is always available as the soft estimate of $\mathbf{x}$, and it may be used when turbo equalization is implemented.

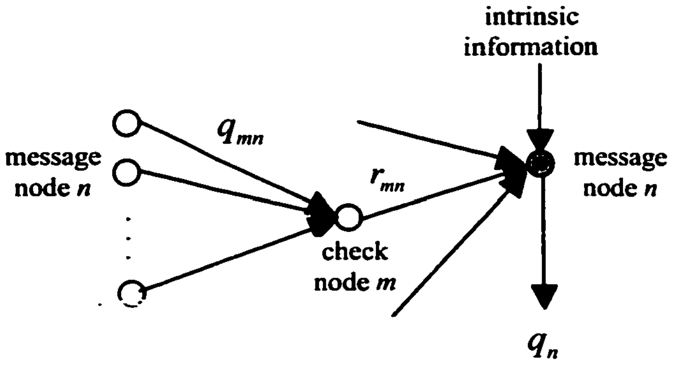


Figure 4.4. Message flow in the BP algorithm.

The message passing inside the BP algorithm is best illustrated using the code graph, as in Figure 4.4. For clarity, message node n is split into two. r_{mn}^i in (4.5) corresponds to the message passed from message node n to check node m. It is the extrinsic information of bit n from the sub-code m and is broadcasted to $\Psi(n) \setminus \{m\}$. q_{mn}^i in (4.6) corresponds to the message passed from check node m to message node

n. It is the estimation of bit n, which is the intrinsic information plus the extrinsic information from $\Psi(n)\setminus\{m\}$, and is broadcasted to $\Phi(m)\setminus\{n\}$.

BP algorithm using LLR

By representing the probabilities in log-likelihood ratio (LLR) form, the BP algorithm may be expressed in the logarithmic domain, and is referred to as the Log-BP algorithm [53].

Define the following LLRs:

$$LLR(f_n) = \log\frac{f_n^1}{f_n^0}$$

$$LLR(r_{mn}) = \log\frac{r_{mn}^1}{r_{mn}^0}$$

$$LLR(q_{mn}) = \log\frac{q_{mn}^1}{q_{mn}^0}$$

and

$$LLR(q_n) = \log\frac{q_n^1}{q_n^0}.$$

The recursive updating steps (4.7)-(4.9) are given by

$$LLR(r_{mn}) = -2\tan^{-1}\left(\prod_{n'\in\Phi(m)\setminus\{n\}} -\tanh\left(\frac{1}{2}LLR(q_{mn})\right)\right) \tag{4.10}$$

$$LLR(q_{mn}) = \sum_{m'\in\Psi(n)\setminus\{m\}} LLR(r_{m'n}) + LLR(f_n) \tag{4.11}$$

and

$$LLR(q_n) = \sum_{m' \in \Psi(n)} LLR(r_{m'n}) + LLR(f_n). \qquad (4.12)$$

Simplified implementation of the BP algorithm

The computational complexity of the BP algorithm lies largely on the MAP decoding of the sub-codes, and is usually large. Also, each r_{mn} and q_{mn} need to be stored. The simplification of the BP algorithm is therefore achieved in two different steps. First, sub-optimal decoding methods can be used to decode the sub-codes. Equations (4.7) and (4.10) are replaced by MAX-approximation [52]. Secondly, q_n may be used to approximate q_{mn}, reducing the storage requirement [54]. Further storage reduction can be achieved by using a "staggered" decoding schedule [55].

4.1.4 Encoding of random LDPC codes

The straightforward way of encoding an LDPC code is to use the generator matrix. Suppose $\mathbf{H}$ is of full rank, it can be rearranged into the following form by column permutation,

$$\mathbf{H} = \left[\mathbf{H}_1 | \mathbf{H}_2\right]$$

where $\mathbf{H}_2$ is a non-singular $M \times M$ square matrix. The generator matrix $\mathbf{G}$ can be found as $\mathbf{G} = \left[\frac{\mathbf{I}_K}{\mathbf{H}_2^{-1}\mathbf{H}_1}\right]$, and the encoding can be done by $\mathbf{G} \cdot \mathbf{s}$ for an information vector $\mathbf{s}$. In general, $\mathbf{H}_2^{-1}\mathbf{H}_1$ is not sparse, therefore the encoding complexity is $R(1-R)N^2$, even though $\mathbf{H}$ is sparse.

An efficient encoding method described in [49] avoids the matrix multiplication $\mathbf{G} \cdot \mathbf{s}$ and in fact avoids solving for the explicit generator matrix $\mathbf{G}$ altogether. Instead, this encoding method takes advantage of the sparseness of the parity check matrix $\mathbf{H}$, and finds the parity check bits in two steps. The resultant encoding complexity is dependent on the code rate and the erasure threshold of the given code graph α^{*} (see 4.4.1), and concentrates at $(1-R-\alpha^{*})^{2} N^{2}+O(N)$. For a rate 8/9 regular LDPC code with $W_{c}=3$, the encoding complexity is $0.019^{2} N^{2}+O(N)$, which is considerably lower than $0.099 N^{2}$ for the straightforward encoding, by a factor of over 200. For a rate 16/17 regular LDPC code with $W_{c}=3$, the encoding complexity is $0.011^{2} N^{2}+O(N)$, compared to $0.055 N^{2}$, a factor of 455.

4.2 MRC with regular LDPC codes

In this section, a MR system with a regular LDPC code is proposed and investigated. A regular LDPC coded system configuration has been considered in [70], in which the channel is assume to be perfectly equalized but AWGN was assumed after equalization. Also, the performance of LDPC codes for a generalized PR channel was studied in [53], with reduced-complexity decoding for LDPC codes.

4.2.1 System description

The proposed system is shown in Figure 4.5. The Lorentzian channel response is equalized into an ME^{2}PR4, with $H(D)=5+4D-3D^{2}-4D^{3}-2D^{2}$. The QMTR code is used to meet the run-length constraint, as well as to reduce the channel errors.

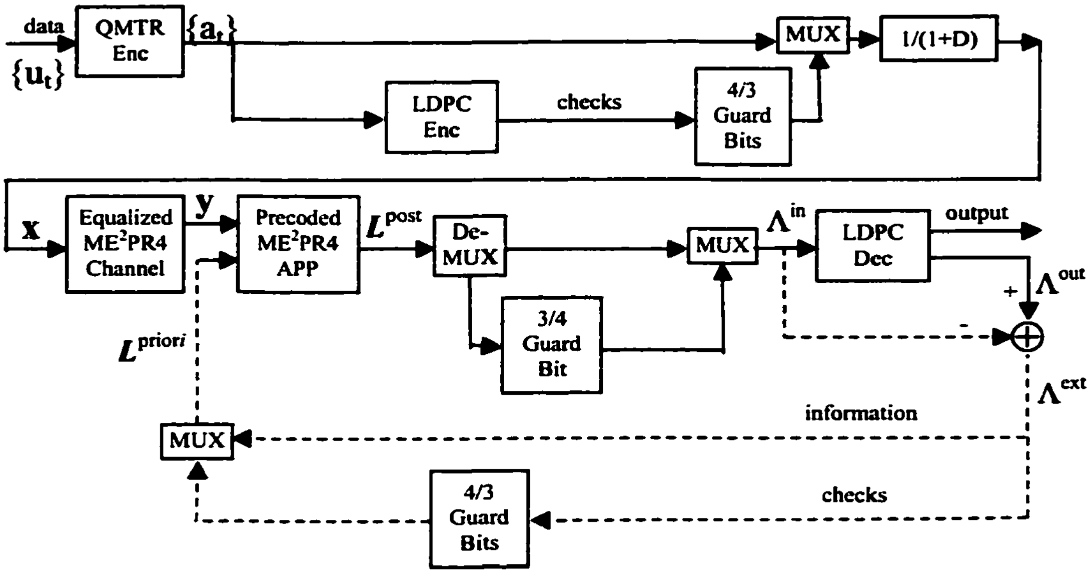


Figure 4.5. Block diagram of a PR channel with an LDPC code.

To avoid soft decoding of the QMTR code, in contrast to current recording systems, the modulation code and the error-correcting code are reversed in order in this system. This scheme can be traced to [37]. On the recording end, the incoming user data block is first encoded by a rate 16/17 QMTR code [9]. The resultant block from the QMTR encoder is further encoded by a high-rate LDPC code. Before recording, the sequence of LDPC check bits is inserted with guard bits so that both run-length conditions are satisfied. The $\frac{1}{1 \oplus D}$ precoder is used to translate the ones into transitions.

On the reading end, a MAP decoder matched to the precoded ME^{2}PR4 is used for the PR channel. The MAP decoder takes the channel vector **y** and the prior LLR vector $\mathbf{L}^{prior}$ with $L_n^{prior} = \log\frac{P(x_n = 1)}{P(x_n = 0)}$, and computes the posterior LLR vector $\mathbf{L}^{posterior}$ with $L_n^{posterior} = \log\frac{P(x_n = 1|\mathbf{y}, \mathbf{L}^{prior})}{P(x_n = 0|\mathbf{y}, \mathbf{L}^{prior})}$.

The LDPC decoder takes the posterior LLR of the channel decoder as input $\Lambda^{in}$and outputs the pseudo-posterior LLR Λ^{out}. From Λ^{in} and Λ^{out}, define $\Lambda^{ext} = \Lambda^{out} - \Lambda^{in}$ as the extrinsic LLR. Λ^{ext} is fed back to the PR channel decoder as the prior LLR. Therefore turbo decoding between the channel and the LDPC code is made possible.

The decoding process has two iteration loops, as shown in the flow chart of Figure 4.6. One is the LDPC loop, which is within the LDPC decoder. After each LDPC iteration, the decoder checks the syndrome $\mathbf{H}\hat{\mathbf{x}}$. If a valid codeword is found, the LDPC decoding is finished, and the whole decoding process stops. The other iteration loop is the channel loop. It is the turbo equalization between the PR channel and the LDPC code [32]. The channel loop iteration takes place only when the maximum number of LDPC loop iterations is reached without finding a valid codeword. Therefore the decoding process halts when either a valid LDPC codeword is found or the maximum number of channel loop iterations is reached.

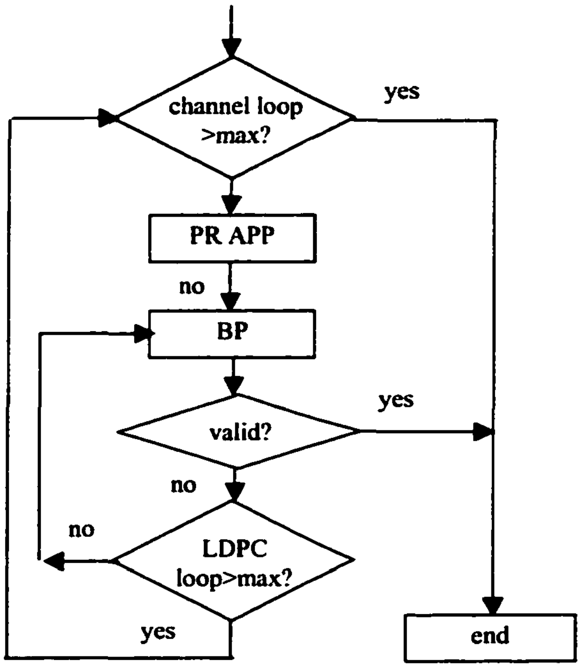


Figure 4.6. The flow chart for iterative decoding.

4.2.2 Performance gains

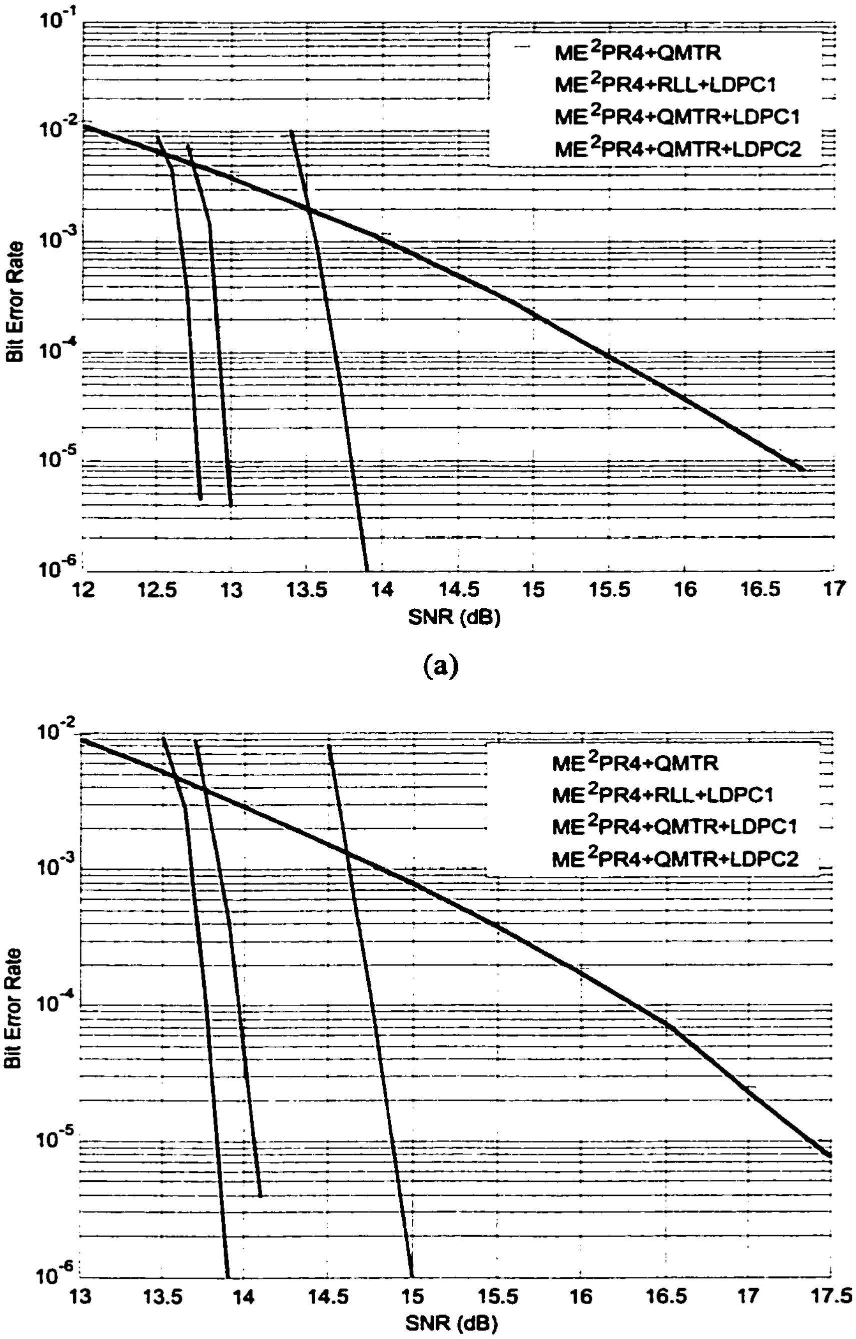
Several LDPC codes are investigated. LDPC1 is a rate 0.9358 code with block length 4376 and 4095 information bits, given in [61], with column weight $W_c = 3$. LDPC2 is designed with rate 0.9402, block length 4629 and 4352 information bits, also with column $W_c = 3$ [62],[68].

The proposed system with LDPC2 has overall code rate 0.8674 and user block size 4096 bits, whereas the system with LDPC1 has code rate 0.8622 and user block size 3854 bits. A maximum of 50 LDPC iterations are performed before the algorithm branches back to the channel detector and a maximum of 100 channel iterations are allowed. The system is evaluated at two different user densities. Figure

4.7(a) shows the performance at user density 2.505 and Figure 4.7(b) shows the performance at user density 2.8. Also plotted in these figures are the performance of the rate 16/17 RLL coded ME^{2}PR4 channel, the QMTR coded ME^{2}PR4 channel, and the LDPC1 and QMTR coded ME^{2}PR4 channel, all at the same user density. The channel signal-to-noise ratio is defined in (1.10).

The simulation results show that the proposed system achieved a significant gain. At user density 2.505, LDPC2 provides an additional 4-dB gain over the QMTR code at a bit error rate of 10^{-5}. At user density 2.8 LDPC2 provides a 3.5-dB gain.

As can be seen later in this section, there are many bit errors (twenty or more in some instances) when a block error occurs, while this number is smaller for the uncoded channel. Therefore, larger performance gains would be obtained if block error rates were used for performance comparison.



(a)

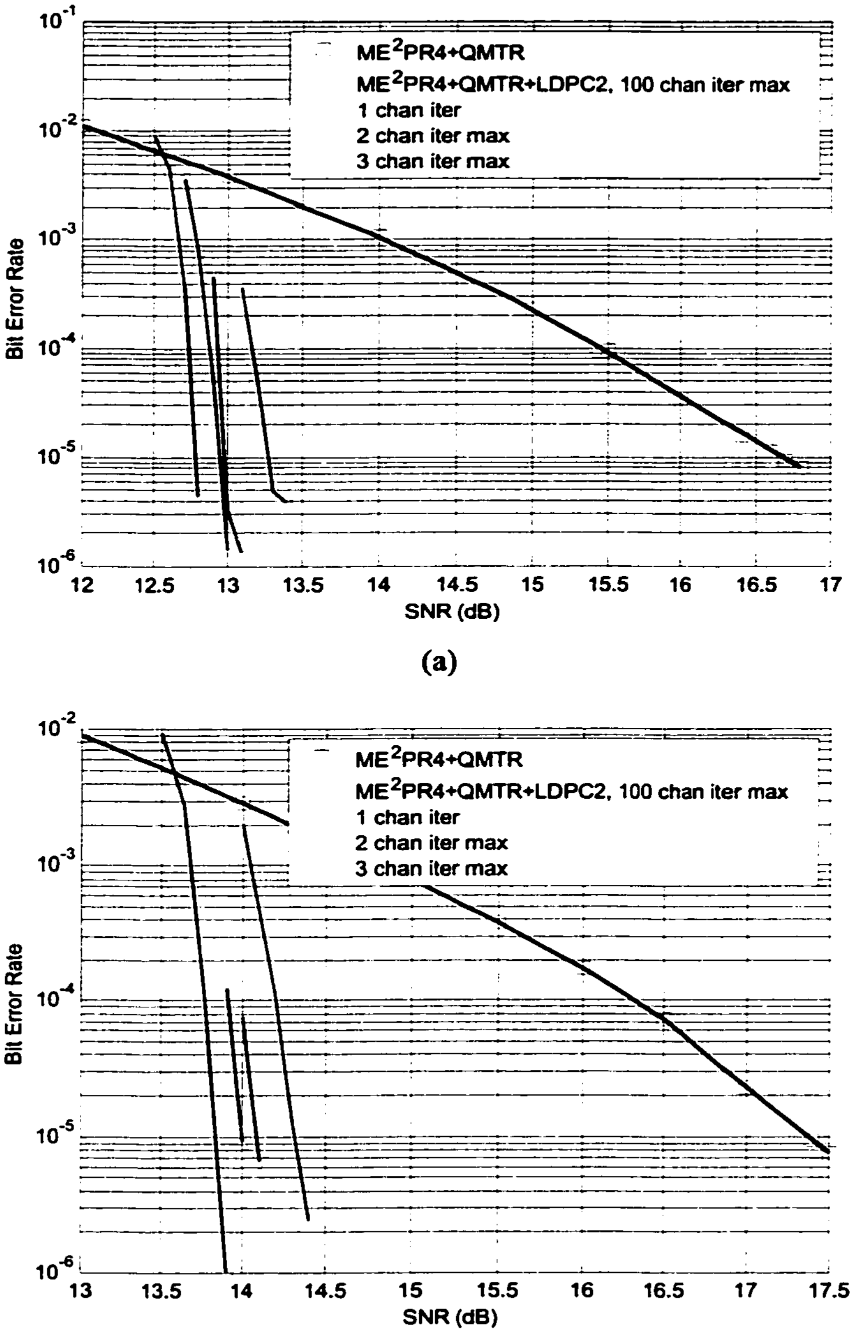
(b)

Figure 4.7. LDPC code on PR channels: (a) S_u=2.505 and (b) S_u =2.8.

4.2.3 Iterative performance

The impact of the maximum number of channel iterations is investigated. If this number is set to one, the channel decoder works once and no turbo equalization is performed. Figure 4.8 shows the performance of the LDPC2 coded system with 1, 2, 3 and 100 maximum channel iterations. Additionally, the performance of the uncoded ME^{2}PR4 system at the same user density is also shown as a comparison.

The total number of LDPC iterations for a block is the sum of the LDPC iterations in each channel iteration. Although the maximum number of LDPC iterations is 50 in the simulations above, the average total number of LDPC iterations is much smaller. Figure 4.9 shows the average number of channel iterations and LDPC iterations actually performed if the maximum number of channel iterations is 1, 2 and 3 for user density 2.8. At bit error rate 10^{-5}, the average number of channel iterations is 1, 1.2 and 1.5, or equivalently the turbo equalization is performed 0, 0.2 and 0.5 times per block on average. The average number of LDPC iterations is about 5, 20 and 35 respectively. From Figure 4.8, it can be seen that at bit error rate 10^{-5}, the gain for a maximum of three channel iterations is about 0.3 dB over a single channel iteration or in other words without turbo equalization. Assuming that the computation time is proportional to the average number of LDPC iterations, it will take a factor of seven increase in computation to obtain this small gain.



(a)

(b)

Figure 4.8. Performance with few channel iterations: (a) S_u=2.505 and (b) S_u=2.8.

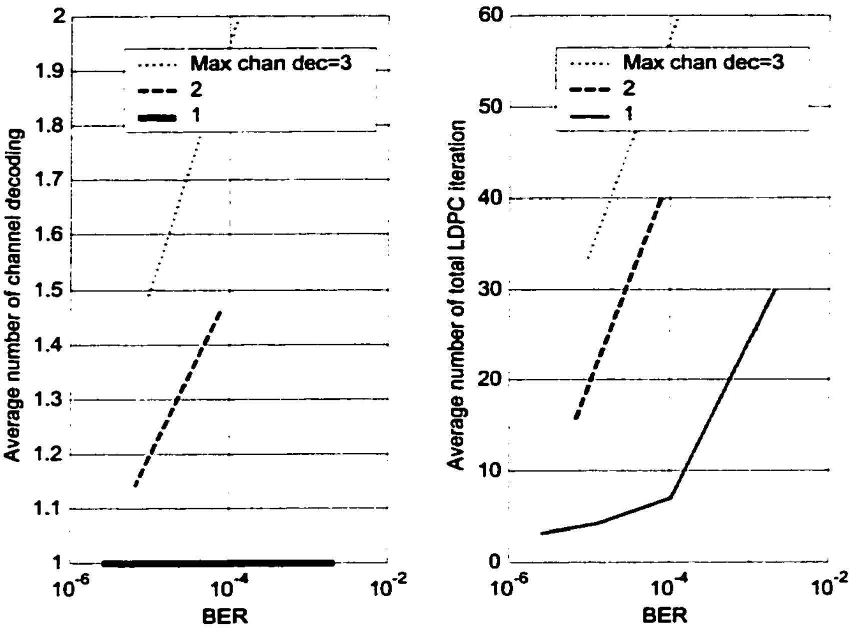


Figure 4.9. Average number of iterations actually performed.

If turbo equalization is not performed and the PR APP output is assumed to be independent Gaussian, LDPC codes that perform well on the AWGN channel will also perform well on the MR channels.

4.2.4 Error distribution

The histogram of the number of bit errors in a block is shown in Figure 4.10. The user density is 2.8 and the bit error rate is around 10^{-5}. No turbo equalization is performed. Notice that when a block has errors, there are many bit errors.

Figure 4.11 shows the error burst statistics of the system without turbo equalization at user density 2.8 and bit error rate around 10^{-5}. An error event delimiter of eight bits is assumed. It can be seen that long error bursts are rare.

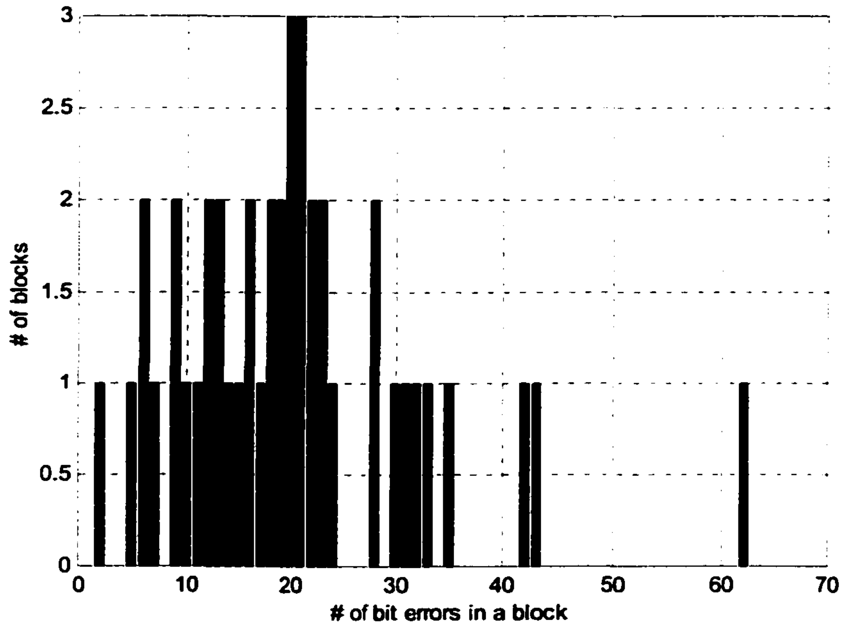


Figure 4.10. Histogram of the number of bit errors in a block.

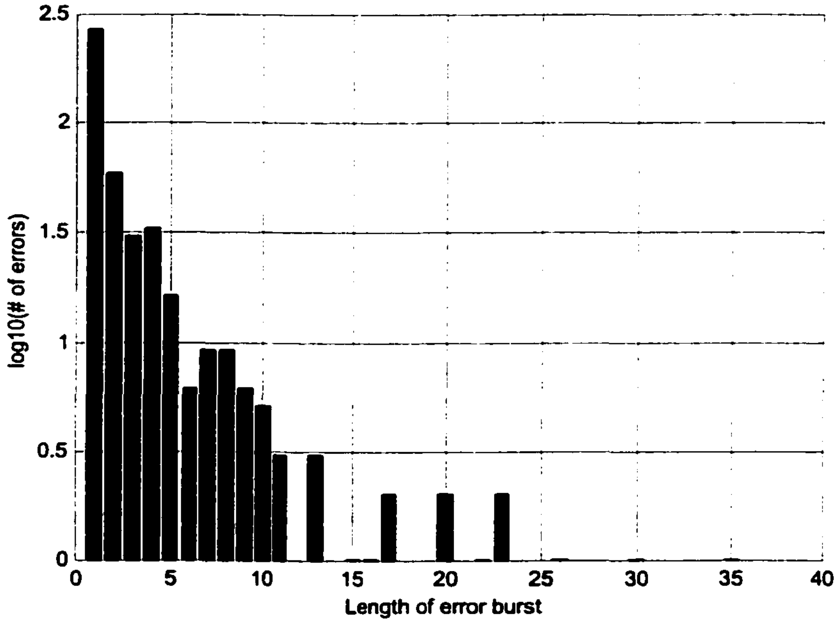


Figure 4.11. Error burst statistics.

4.2.5 Concatenation with RS codes

It is difficult or even impossible to determine the performance of the proposed system at the error rate where HDD systems operate. This fact prompts the consideration of concatenating a RS code with an LDPC code in a system [67]. Shown in Figure 4.12 is the system to be investigated. Enclosed in the dashed box is simply the system shown in Figure 4.5.

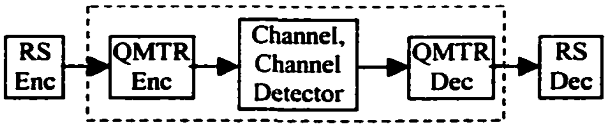


Figure 4.12. Diagram of RS-LDPC system.

In the simulation, the QMTR decoding is skipped, and only pseudo-RS coding is performed in that there is actually no RS encoding, but the outputs of the LDPC decoder are assumed to be codewords of some RS code. To determine the error, the assumed codewords are compared with the correct codewords. If the number of symbol errors exceeds the error correcting capability, an error is declared.

RS code (136, 126) on GF(2^8) is assumed to be used, and codewords are 4-way interleaved. Therefore, the information bits of an LDPC codeword can be decomposed into four RS codewords.

The system is simulated at $S_u = 2.8$. First assume the RS code is always in the system, therefore no further code rate loss is considered. The simulation results are shown in Figure 4.13. The gain of the RS-LDPC system over the LDPC system is very limited.

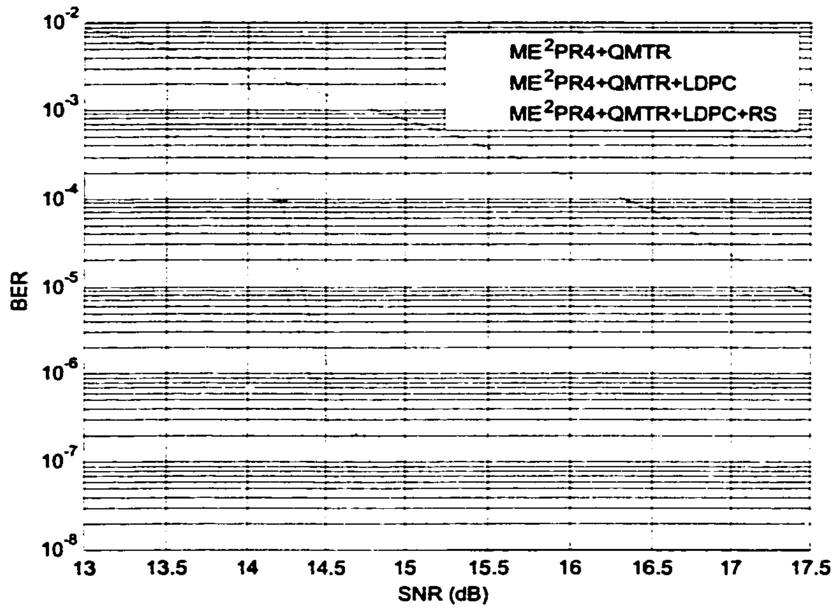


Figure 4.13. RS error correction on an LDPC coded system.

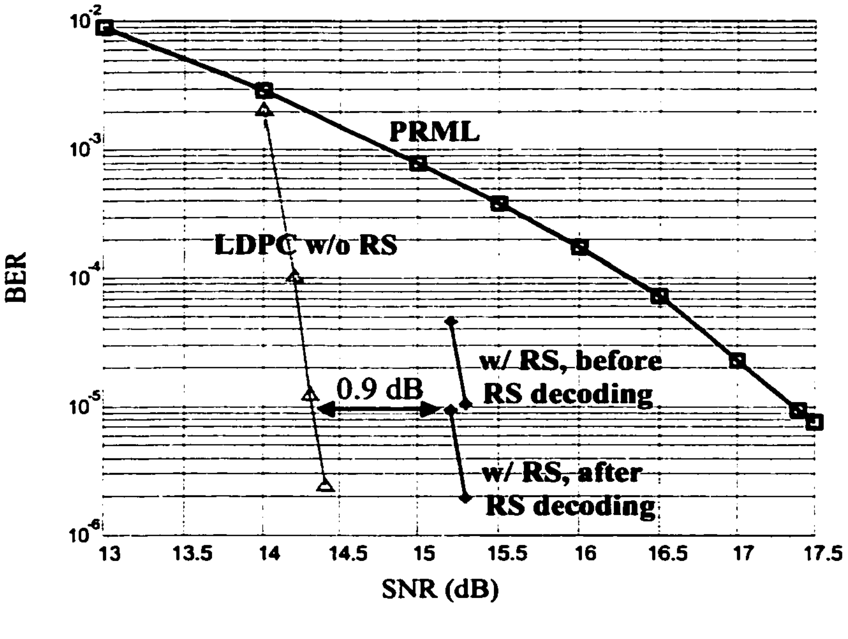


Figure 4.14. Performance of RS-LDPC system taking into account the RS coding penalty.

If the coding penalty of the RS code is considered, then the performance of RS-LDPC code is actually not as good as the LDPC system. The loss is about 0.9 dB.

The reason why the RS code is not very effective in this system lies in the following facts. The average number of bit errors in a failed block is 19.4. When a block fails it usually has many bit errors, as shown in Figure 4.10. What is worse, most errors are single bit errors, as shown in Figure 4.11, so there are many error events in a failed block, making RS decoding difficult. However, on MR channels with very few random errors, RS-LDPC concatenation might be beneficial. In this case, very-high rate LDPC codes should be used to minimize the coding penalty.

4.3 Finite-geometry LDPC codes for MR systems

Finite-geometry LDPC codes are a family of codes that have a geometric structure, as well as low-density parity-check matrices.

4.3.1 Finite geometry LDPC codes

All the LDPC codes described above are randomly generated. However, there are LDPC codes that can be generated by a finite geometry [57]. It is noted in [57] that Euclidean geometry (EG) codes and projective geometry codes [58] are LDPC codes. They are referred to as Euclidean geometry LDPC (EG-LDPC) codes and projective geometry LDPC codes [57]. These two classes of codes are very similar, and therefore only EG-LDPC codes are investigated.

On GF(2^{ms}), let α be a primitive element and all the 2^{ms} elements be $\{0,1,\alpha,\cdots,\alpha^{2^{ms}-2}\}$. Let β be a sub-field $GF(2^s)$. For any $a_0, a \in GF(2^{ms})$, $a_0 + a\beta$ is defined as a line. There are $\left(2^{(m-1)s}-1\right)\frac{2^{ms}-1}{2^s-1}$ distinct lines that do not cross the origin. The incidence vector of a line is defined as a row vector **v** such that $v_i = 1$ if α^i is on the line and $v_i = 0$ otherwise, where $i = 0,\cdots,2^{ms}-2$. Denote as **H** the matrix that has all incidence vectors as its rows. **H** is a $\left(2^{(m-1)s}-1\right)\frac{2^{ms}-1}{2^s-1}$ by $2^{ms}-1$ sparse matrix with $W_c = \frac{2^{ms}-1}{2^s-1}-1$ and $W_r = 2^s$. The overlap between any pair of columns is at most one. The code with **H** as parity check matrix is an EG code, denoted as EG$(m, 2^s)$, and is an LDPC code free of cycles of length four.

EG-LDPC codes cyclic and can be encoded using shift-register circuitry, which is extremely simple. However, EG-LDPC codes with moderate length have low code rates. In order to obtain high-rate codes for MR systems, an extension of EG-LDPC codes is done by modifying $\mathbf{H}$, resulting in split EG-LDPC codes and companion EG-LDPC codes [57].

4.3.2 Split EG-LDPC codes

For simplicity, only EG$(2,2^s)$ (m=2) codes are considered. Therefore **H** is a $\left(2^{2s}-1\right)\times\left(2^{2s}-1\right)$ matrix. One can generate a $\left(2^{2s}-1\right)\times k\left(2^{2s}-1\right)$ matrix $\mathbf{H}_{split}$ through k-fold column splitting. Each column vector $\mathbf{H}_i$ is split into k column vectors of $\mathbf{H}_{split}$, denoted as $\mathbf{H}_{i,1}$, $\mathbf{H}_{i,2}$, and $\mathbf{H}_{i,k}$. The first 1 in $\mathbf{H}_i$ goes to $\mathbf{H}_{i,1}$ at the same place, the second in $\mathbf{H}_i$ goes to $\mathbf{H}_{i,2}$, and so on. If $k \mid W_c$ where W_c is the column weight of $\mathbf{H}$, then $\mathbf{H}_{split}$ can be rearranged by column permutation into the following form,

$$\mathbf{H}_{split} = \left[\mathbf{H}_{split,1} \quad \mathbf{H}_{split,2} \quad \cdots \quad \mathbf{H}_{split,k}\right]$$

where each $\mathbf{H}_{split,i}$ is a circulant matrix. The split $\mathrm{EG}(2,2^s)$ code is defined as the null space of the row space of $\mathbf{H}$ and therefore quasi-cyclic [57]. The generator matrix of the split $\mathrm{EG}(2,2^s)$ is easy to find and the encoding can be done by shift-register circuits [59],[60].

Split EG-LDPC codes have smaller minimum distances than the original EG-LDPC codes, and it is expected that the performance of a split EG-LDPC code might be inferior to a random LDPC code with the same code rate.

A split EG-LDPC coded EPR4 system is considered in [60] and compared with RS coded system. It is shown that this system combats 128-bit burst erasure very well, with the assumption that the erasure is detected.

4.3.3 Companion $\mathrm{EG}(3,2^3)$ codes

The parity check matrix $\mathbf{H}'$ of a companion EG-LDPC (C-EG-LDPC) code is obtained by transposing the parity check matrix $\mathbf{H}$ of the original EG-LDPC code. In this section, we are only interested in the companion $\mathrm{EG}(3,2^3)$ (C-$\mathrm{EG}(3,2^3)$) code.

The matrix $\mathbf{H}$ for $\mathrm{EG}(3,2^3)$ is 4599×511, and in the following form,

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 & \mathbf{H}_2 & \cdots & \mathbf{H}_9 \end{bmatrix}^t$$

where each $\mathbf{H}_i$ is a circulant square matrix, i.e., circularly shifting any row results into another row. C-EG$(3, 2^3)$ has $\mathbf{H}'$ as parity check matrix,

$$\mathbf{H}' = \begin{bmatrix} \mathbf{H}_1 & \mathbf{H}_2 & \cdots & \mathbf{H}_9 \end{bmatrix}.$$

Therefore, the companion EG$(3, 2^3)$ code is quasi-cyclic, whose codeword, when circularly shifted by 9 bits, is another codeword.

The rank of $\mathbf{H}'$ is 372 and C-EG$(3, 2^3)$ is a $(N = 4599, K = 4227)$ LDPC code. The redundant rows of $\mathbf{H}'$ can be eliminated, resulting in a parity check matrix with maximum $W_c = 8$. C-EG$(3, 2^3)$ is simulated on an AWGN channel, in one case $\mathbf{H}'$ is used as the parity check matrix and in another case the reduced $\mathbf{H}'$ is used as parity check matrix. The simulation results are shown in Figure 4.15. Both the total word error rate (WER) and undetected WER are shown.

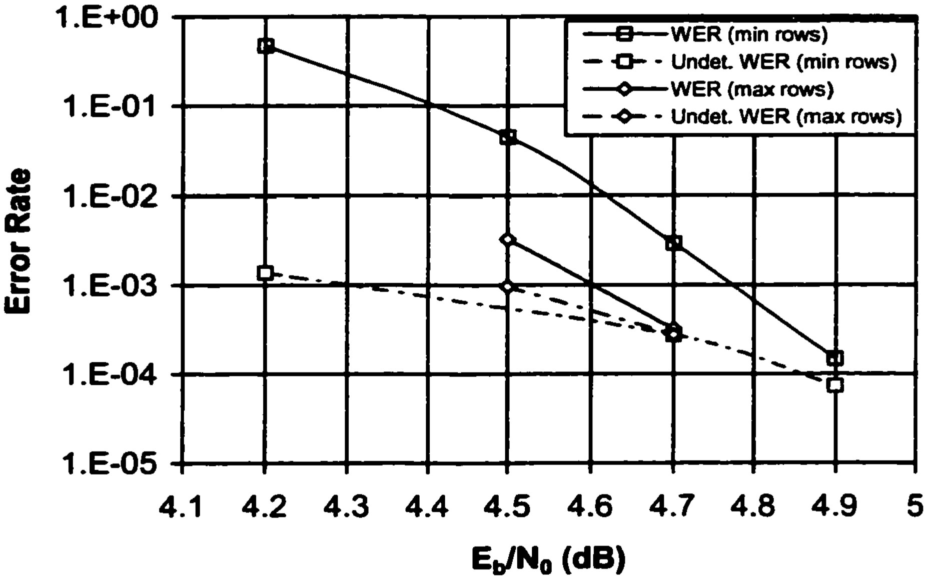


Figure 4.15. Performance of C-EG$(3,2^3)$ on AWGN, with different redundancy parity check matrices. Maximum 50 iterations.

At WER 10^{-3}, decoding using $\mathbf{H}'$ has less than 0.2 dB gain over that using redundancy-reduced $\mathbf{H}'$, with 1.37 times of complexity though. Also, when decoded with $\mathbf{H}'$, the majority of the errors are undetected at $E_b / N_0 = 4.7$ dB. This indicates two things: 1) The performance shown is very close to the maximum-likelihood bound and 2) The multiplicities of the low weight codewords are very large.

4.3.4 Turbo equalization schedules

Different turbo equalization *schedules*, i.e., the number of LDPC iterations in the LDPC loop, are possible for the iteration between the channel detector and the LDPC decoder. Mittelholzer *et al.* studied the optimal schedule for a particular configuration [53]. On EPR4 with an LDPC code of length N=508 and R=0.8839, it was found that the schedule with just one iteration in the inner LDPC loop yields best performance for the same total number of LDPC iterations.

However, their conclusion seems not to be universal. For the C-EG(3, 2^3) coded system, different schedules were tested, with a total of 100 LDPC iterations performed in each schedule. The simulation results are shown in Figure 4.16, in which both BER and sector error rate (SER) curves are plotted. The best schedule is the one with 5 LDPC iterations in the inner loop.

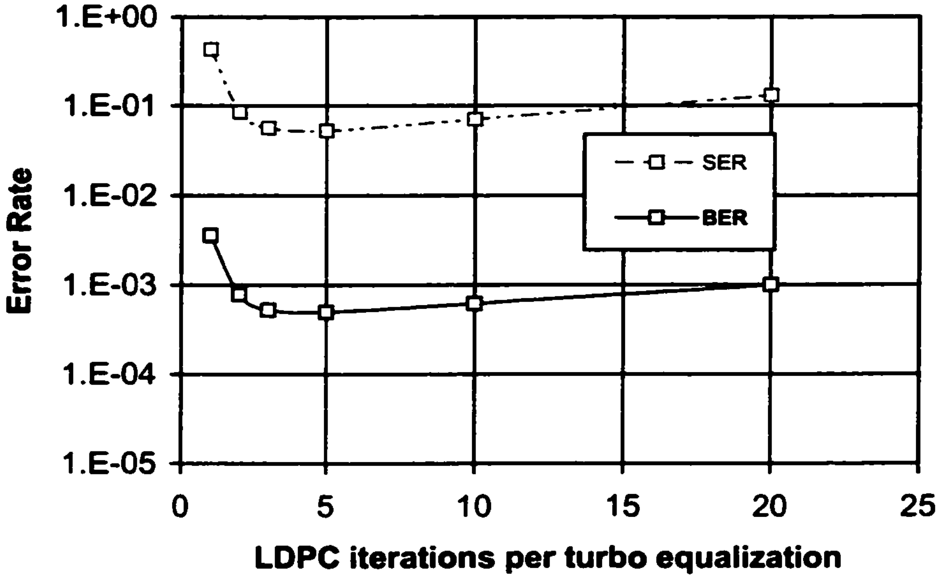


Figure 4.16. Comparison of turbo equalization schedules.

The best turbo equalization must be dependent on the LDPC code. Suppose that the LDPC code graph is free of cycles below L, then for the first $\frac{L}{2}$ BP iterations, the information passed in the graph satisfies the independent assumption in (4.8). Intuitively, larger L and fewer short cycles allow more LDPC iterations in the inner loop. Since lower rate codes have larger L or fewer short cycles, more LDPC iterations per turbo equalization is a better schedule.

Also, given that the complexity of the channel detector is higher than that of the LDPC decoder, it is justified that only schedules with many iterations in the LDPC loop be considered.

4.3.5 System performance

Considered here is the C-EG-LDPC(3,2^3) in a system diagram shown in Figure 4.5. This system has overall code rate $R = 0.842$, and is simulated at $S_u = 2.72$ under AWGN.

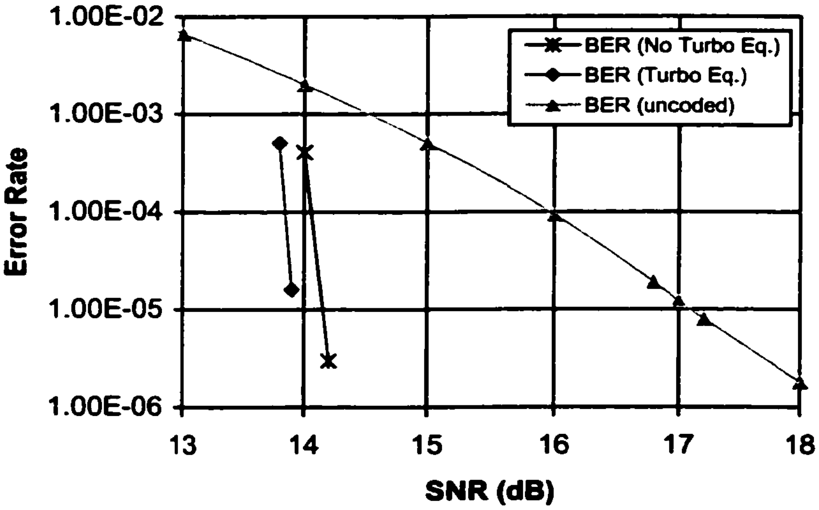


Figure 4.17. Performance of C-EG-LDPC coded system.

This system is also simulated under AWGN with 48-bit disk defect. At SNR=19.2 dB, the total sector error rate (SER) is 0.0051, and the undetected SER is 0.0041. Again, the high undetected SER is probably due to the larger multiplicities of low weight codewords.

4.4 Irregular LDPC codes for MR systems

In this section, an irregular LDPC code for an MR system is described and simulated. The performance is compared with an RS coded system.

4.4.1 Capacity and optimization of LDPC codes

Given an LDPC code with a certain length and the degree distribution pair $\lambda(x)$ and $\rho(x)$ as defined in (4.2) and (4.3), to find its performance is difficult. Two techniques look at the capacity of the infinitely long LDPC code with tree-like cycle-free code graph and the same degree distribution pair $\lambda(x)$ and $\rho(x)$ under BP decoding.

On a binary erasure channel, each bit is erased with probability α. The erasure *threshold* of the infinitely long LDPC code is denoted by α^*. Any erasure fraction $\alpha < \alpha^*$ is recoverable. α^* is shown as to be the supreme of α such that [45]

$$\lim_{i \to \infty} \alpha_i = 0 \text{ with } \alpha_i = \alpha\lambda\left(1-\rho\left(1-\alpha_{i-1}\right)\right) \text{ and } \alpha_0 = \alpha. \tag{4.13}$$

For example, the rate 8/9 LDPC code with $W_c = 3$ has capacity $\alpha^* = 0.092$ and the rate 16/17 LDPC code with $W_c = 3$ has capacity $\alpha^* = 0.048$.

On binary input AWGN channels, the capacity of the infinitely long LDPC code can be found by the density evolution technique [56].

By assuming the capacities obtained above are also the figures of merit of finite-length LDPC codes with the same degree distribution pair, these two techniques can be used to optimize the degree distribution pair for best performance [46],[47].

4.4.2 Irregular LDPC coded system

Using density evolution techniques [56], an irregular LDPC code was optimized for the ME^{2}PR4 channel [63]. This code has length 4835, rate 0.9 and average column weight 6.41. Considering the average column weight, the computational complexity (per bit per iteration) is more than twice of a $W_c = 3$ regular LDPC code.

A system diagram using this code is shown in Figure 4.18, and has overall code rate $R = 0.8410$. In the simulation, the block in the dashed box is substituted by random data.

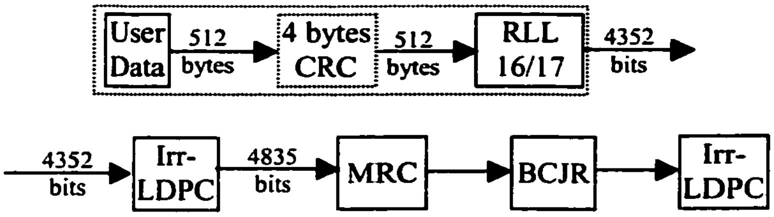


Figure 4.18. System diagram using an irregular LDPC code.

4.4.3 Performance comparison with RS coded system

On a Lorentzian-Gaussian channel with position jitter, the performance of this LDPC system is simulated and compared with a 4-way interleaved RS system shown in Figure 4.19 at $S_u = 2.5$. No turbo equalization is performed and 50 LDPC iterations are performed at most. The RS code is over GF(2^8) and capable to correct seven errors.

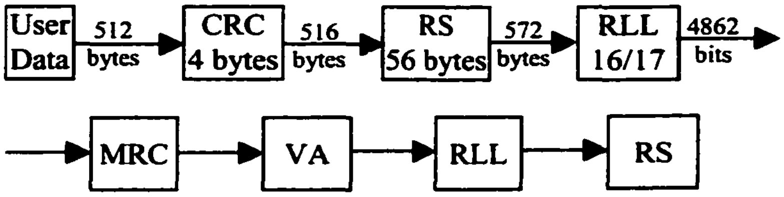
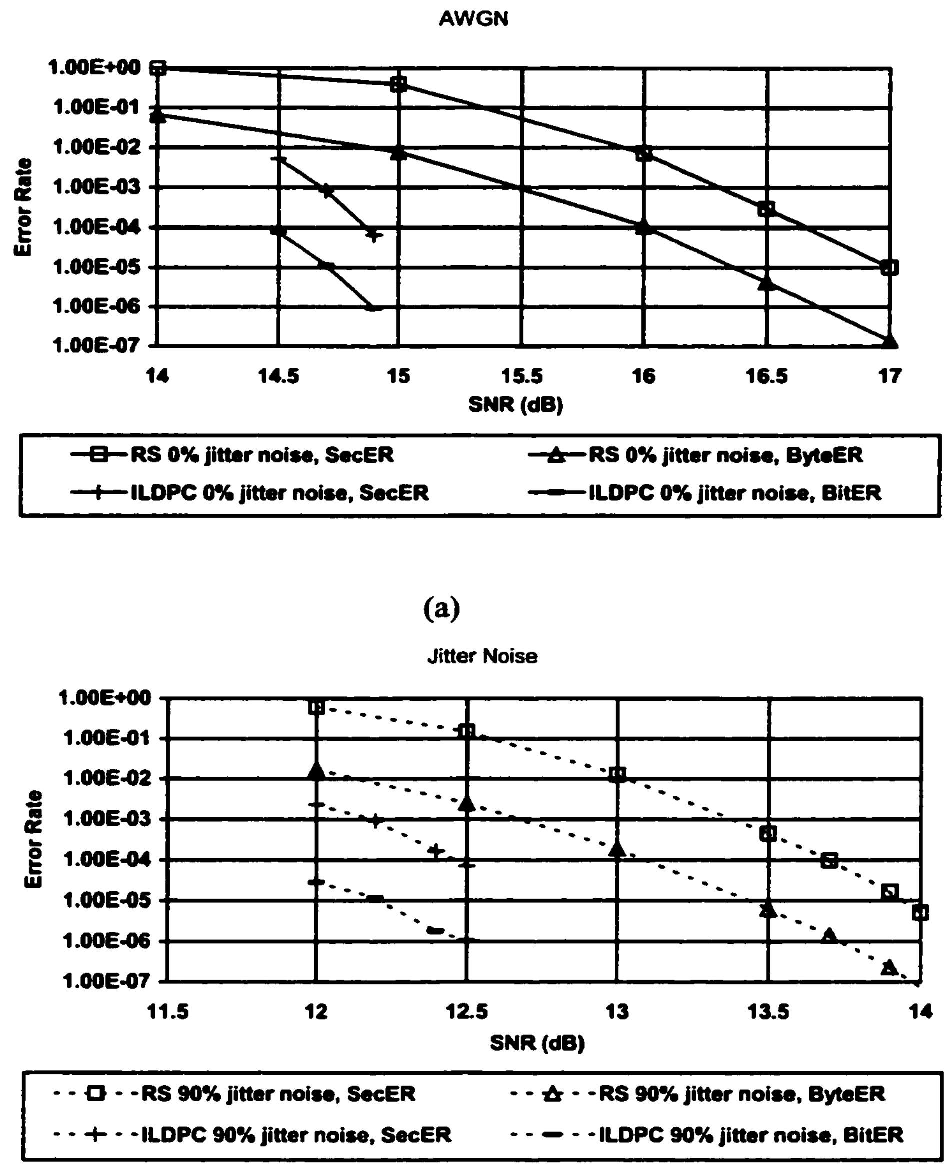


Figure 4.19. System diagram using an RS code.

Shown in Figure 4.20(a) and (b) are the performance of the irregular LDPC coded system with 0% and 90% jitter noise, respectively. With 0% jitter noise, 1.7 dB gain is obtained at SER=10^{-4}; with 90% jitter noise power, 1.3 dB gain is obtained at the same SER.



(a)

(b)

Figure 4.20. Performance of irregular LDPC coded systems. (a) 0% jitter noise; (b) 90% jitter noise.

4.5 Conclusion

In this chapter, MR systems with different LDPC codes were investigated. LDPC coded systems show substantial gains over uncoded channels or conventional RS systems with AWGN or jitter noise. Turbo equalization brings only several tenths of a dB gain. At moderately low error rate, very few iterations are needed.

All LDPC codes, random codes and finite-geometry LDPC codes, regular codes and irregular codes, provide similar coding gains. Finite-geometry LDPC codes have the advantage of low encoding complexity but the disadvantage of high decoding complexity. Irregular LDPC codes also have high decoding complexity. In conclusion, the $W_c = 3$ random regular LDPC codes seem to be the best overall choice.

RS-LDPC concatenation is not effective under random noise. But on MRCs where burst noise is the main source of errors, it might be necessary. In this case, a very high-rate LDPC code should be used to correct occasional large number of bit errors in a sector, leaving the task of correcting long bursts to the RS code.

5 Q-ARY LDPC CODES FOR MAGNETIC RECORDING CHANNELS

The LDPC codes considered in the previous chapter are all over GF(2). To emphasize this, they shall be called binary LDPC (B-LDPC) codes. It is shown that B-LDPC codes provide impressive gains on MRCs with random noise. However, MRCs are subject to long erasure bursts, and B-LDPC codes cannot combat long bursts very well. In this chapter, non-binary LDPC (or Q-ary LDPC, Q-LDPC) codes are proposed for MRCs.

In Section 5.1, the basics of Q-LDPC codes are provided. In Section 5.2, a decoding method for Q-LDPC codes is described, and its decoding complexity analyzed. A code design method is developed in Section 5.3 to improve the erasure correction capability. Performance of Q-LDPC coded MRCs is presented in Section 5.4. Array codes are addressed and compared to Q-LDPC codes in Section 5.5.

5.1 Q-ary LDPC (Q-LDPC) codes

The first work on Q-LDPC codes appeared in [64],[65]. Similar to B-LDPC codes, a Q-LDPC can be described by a low-density parity-check matrix $\mathbf{H}_{M\times N}$. Each element $\mathbf{H}_{i,j}$ of $\mathbf{H}_{M\times N}$, is now an element from $\mathrm{GF}(q=2^p)$. The null space of the row space of $\mathbf{H}$ is the Q-LDPC code. A row vector $\mathbf{x}$ of length N is a codeword if

$$\sum_n H_{m,n} x_n = 0, \quad m = 1, \cdots, M. \tag{5.1}$$

Similar to B-LDPC codes, a Q-LDPC code can be regarded as a collection of M sub-codes, which are simply parity check codes. Also, a Q-LDPC code can be represented by a bipartite graph, but the edges may carry q-1 different values. For regular Q-LDPC codes, column weight W_c and row weight W_r can be defined similarly as B-LDPC codes by counting the number of non-zero GF(q) elements.

At rate 1/4 to 1/2, it was shown in [64]-[66] that Q-LDPC codes outperform B-LDPC codes on the AWGN channel. On channels with noise bursts, the consecutive bits in the burst window are grouped into fewer symbols, therefore it is easier for the Q-LDPC code to recover.

5.2 Q-LDPC decoding and its complexity

Any decoding method for B-LDPC codes can be extended to Q-LDPC codes by modifying the field operation. However, the efficient implementation of the BP algorithm for B-LDPC codes using LLR cannot be done for Q-LDPC codes. This fact increases the decoding complexity of Q-LDPC codes.

5.2.1 BP decoding for Q-LDPC codes

Given the probability mass function $pmf(x_n)$, $n = 1, \cdots, N$, where x_n can be any $f_i \in \mathrm{GF}(q), i = 0, \cdots, q-1$. BP decoding for Q-LDPC codes is done in exactly the same two steps as for B-LDPC codes: row step and column step,

$$r_{mn}^{f_i} = P(\text{sub-code } m \text{ is satisfied} \,|\, x_n = f_i, pmf(x_{n'}) = q_{mn'} \text{ for } n' \in \Phi(m) \setminus \{n\}) \tag{5.2}$$

$$q_{mn}^{f_i} = P(x_n = f_i \,|\, \Psi(n) \setminus \{m\}) . \tag{5.3}$$

In the row step, the sub-codes are decoded, or equivalently the check nodes are updated. The BCJR algorithm can be used for MAP decoding. However, the trellis of each sub-code is one with q states and radix-q, as shown in Figure 5.1.

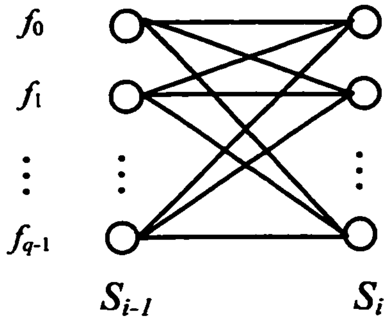


Figure 5.1. Trellis diagram section of Q-LDPC sub-codes.

Simplify the notation of sub-code constraint to $\sum_{i=1}^{N} h_i x_i = 0$. As described in Section 4.1.3, the forward-backward algorithm involves three steps: forward recursion, backward recursion and combination step. The forward recursion is

$$P(S_i = f_j) = \sum_{k=0}^{q-1} P(S_{i-1} = f_k) \cdot P\left(x_i = h_i^{-1}(f_j - f_k)\right), \tag{5.4}$$

where h_i^{-1} is the inverse of h_i in GF(q). The operation needed for one step of forward recursion is q^2 multiplications, $q^2 - q$ additions, plus $2q^2$ field operations. Same operations are needed for the backward recursion and combination step.

There are M sub-codes to be decoded. For a row weight W_r Q-LDPC code, each sub-code has length W_r, therefore in total $3q^2MW_r$ multiplications, $3(q^2-q)MW_r$ additions and $6q^2MW_r$ field operations are needed for a horizontal step updating.

In the column step, message nodes are updated with the independence assumption

$$q_{mn}^{f_i} = \alpha_{mn} f_n^{f_i} \prod_{m' \in \Psi(n) \backslash \{m\}} r_{m'n}^{f_i}, \qquad (5.5)$$

and the posterior probabilities are computed as

$$q_n^{f_i} = \alpha_n f_n^{f_i} \prod_{m' \in \Psi(n)} r_{m'n}^{f_i}. \qquad (5.6)$$

The hard decision is chosen as $\hat{x}_n = \arg\left\{\max\left(q_n^{f_i}\right)\right\}$. For large W_c, it is advantageous to use the forward-backward recursion to compute (5.6). The direct multiplication is, however, simpler for small W_c (for instance $W_c = 3$), as is the case of interest here. Updating each message node takes $q(W_c^2 + W_c)$ multiplications and $(q-1)W_c$ additions. For the total N message nodes, N times this computation is needed.

In summary, the operations needed per bit per iteration are $\frac{1}{Np}\left(3q^2MW_r + qN\left(W_c^2 + W_c\right)\right)$ multiplications, $\frac{1}{Np}\left(3(q^2-q)MW_r + (q-1)NW_c\right)$ additions and $\frac{1}{Np}6q^2MW_r$ field operations. Using the relation $MW_r = NW_c$, these numbers can be simplified to $\frac{1}{p}\left(3q^2W_c + q\left(W_c + 1\right)\right)$, $\frac{1}{p}\left(3(q^2 - q)W_c + (q-1)W_c\right)$ and $\frac{1}{p}6q^2W_c$.

5.2.2 Fast implementation of the BP algorithm using fast Fourier transforms

The computation complexity described above is $O(q^2 / p)$, but it can be simplified. The idea of using a fast Fourier transform (FFT) in the BP decoding was proposed in [66] and [56].

Notice that in the row step, decoding of the sub-codes is to find $pmf(\sum_i x_i)$ with known $pmf(x_i)$, and $pmf(\sum_i x_i)$ is the same as the convolution of all $pmf(x_i)$, which can be efficiently computed using the FFT,

$$pmf(\sum_i x_i) = \mathrm{IFFT}\left(\prod_i \mathrm{FFT}\left(pmf(x_i)\right)\right) \tag{5.7}$$

where IFFT is the inverse FFT. Notice that for B-LDPC codes, (5.7) is actually the same as the difference BP in Section 4.1.3

Since the function $pmf(x_i)$ is defined on GF(q), $\mathrm{FFT}\left(pmf(x_i)\right)$ is not a q-point FFT but a p-dimension 2-point FFT. An example for $q = 8$ is illustrated in Figure 5.1. The field elements are represented in polynomial form. In the first layer, the FFT computes the sum and difference of the probabilities of two field elements differing from each other by only one bit location.

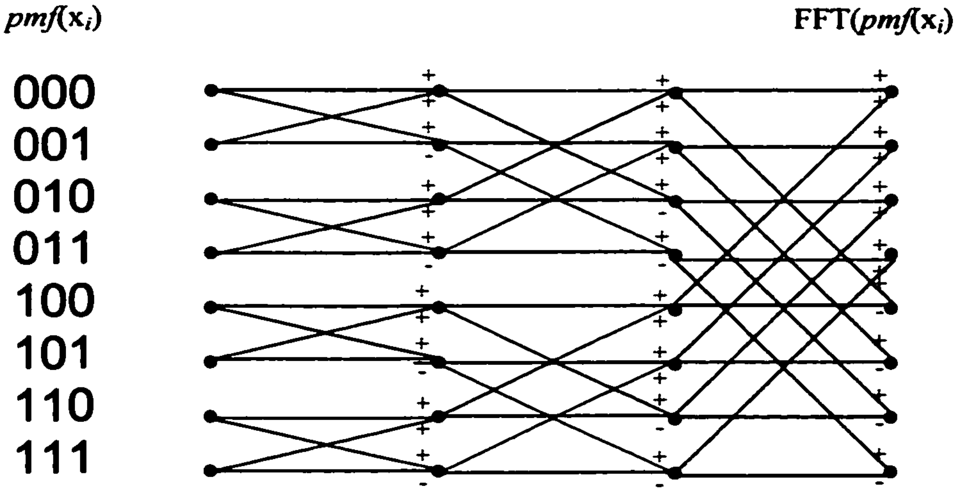


Figure 5.2. FFT of $pmf(x_i)$ for $q = 8$.

Using the FFT, the forward recursion (5.4) becomes

$$\mathrm{FFT}\left(pmf(\mathrm{S}_i)\right) = \mathrm{FFT}\left(pmf(\mathrm{S}_{i-1})\right) \cdot \mathrm{FFT}\left(pmf(h_i x_i)\right), \tag{5.8}$$

which needs only $3qW_r$ multiplications and the overhead of the FFT and IFFT. Each FFT, as well as IFFT, needs pq additions. Therefore the computation needed for the horizontal step is $3qMW_r$ multiplications and $pqMW_r$ additions. The column step remains the same as in (5.5).

In summary, the computation needed per bit per iteration is $\frac{q}{p}\left(W_c^2 + 4W_c\right)$ multiplications and $2qW_c + \frac{q}{p}$ adds, which is $O(q)$. This algorithm is called the FFT-BP algorithm.

5.2.3 Logarithm domain implementation of the FFT-BP

In a practical implementation of the decoder, it is highly desirable to eliminate the need for real-valued multiplications. In the following, a technique is described to meet this requirement.

In the FFT-BP algorithm, real-valued multiplication occurs in both the row step and the column step. In the column step, the multiplicands are $pmf(x_i)$. Intuitively, one should define new variables as the logarithm of these multiplicands. Let v be a probability (>0), and define

$$u = \log(v)\,. \tag{5.9}$$

Then in the column step, only additions are needed.

In the row step, as in (5.8), the multiplicands are $\mathrm{FFT}(pmf(x_i))$. Since $\mathrm{FFT}(pmf(x_i))$ may have negative values, the definition of the logarithm domain variables is complicated. Define $LG: \mathbf{R} \rightarrow \{1,-1\} \times \mathbf{R}$ by

$$u = (u', u^*) = \left(\mathrm{sgn}(v), \log|v|\right) \tag{5.10}$$

where **R** is the field of reals. The inverse $LG^{-1}: \{1,-1\} \times \mathbf{R} \rightarrow \mathbf{R}$ is

$$v = u' \exp(u^*)\,. \tag{5.11}$$

Then for $u_1 = LG(v_1) \triangleq \left(u_1', u_1^*\right)$, and $u_2 = LG(v_2) \triangleq \left(u_2', u_2^*\right)$, where $v_1, v_2 \in R$, define the operations +, -, ×, and ÷ such that

$$u_1 \odot u_2 = LG(v_1 \odot v_2) \tag{5.12}$$

where $\odot$ stands for any of the four operations. It is straightforward to show that

1. $$u_1 \times u_2 \triangleq LG(v_1 \times v_2) \tag{5.13}$$

$$= LG\left(u_1^{\cdot} \exp(u_1^{\bullet}) u_2^{\cdot} \exp(u_2^{\bullet})\right)$$

$$= \left(u_1^{\cdot} u_2^{\cdot}, u_1^{\bullet} + u_2^{\bullet}\right)$$

2. $$u_1 \div u_2 \triangleq LG(v_1 \div v_2) \tag{5.14}$$

$$= LG\left(u_1^{\cdot} \exp(u_1^{\bullet}) / \left(u_2^{\cdot} \exp(u_2^{\bullet})\right)\right)$$

$$= \left(u_1^{\cdot} u_2^{\cdot}, u_1^{\bullet} - u_2^{\bullet}\right)$$

3. $$u_1 + u_2 \triangleq LG(v_1 + v_2) \tag{5.15}$$

$$= LG\left(u_1^{\cdot} \exp(u_1^{\bullet}) + u_2^{\cdot} \exp(u_2^{\bullet})\right)$$

$$\triangleq \left(u^{\cdot}, u^{\bullet}\right)$$

where $u^{\cdot}$ is determined as

$$u^{\cdot} = \begin{cases} 1 \quad \text{if } u_1^{\cdot} = u_2^{\cdot} = 1 \\ \quad \text{or } (u_1^{\cdot} = 1) \cap (u_2^{\cdot} = -1) \cap (u_1^{\bullet} > u_2^{\bullet}) \\ \quad \text{or } (u_1^{\cdot} = -1) \cap (u_2^{\cdot} = 1) \cap (u_1^{\bullet} < u_2^{\bullet}) \\ -1 \quad \text{if } u_1^{\cdot} = u_2^{\cdot} = 1 \\ \quad \text{or } (u_1^{\cdot} = 1) \cap (u_2^{\cdot} = -1) \cap (u_1^{\bullet} > u_2^{\bullet}) \\ \quad \text{or } (u_1^{\cdot} = -1) \cap (u_2^{\cdot} = 1) \cap (u_1^{\bullet} < u_2^{\bullet}) \end{cases} . \tag{5.16}$$

and $u^{\bullet}$ is calculated in two cases:

a. $u_1^{\cdot} = u_2^{\cdot}$

$$\begin{aligned} u^{\bullet} &= \log\left(\exp\left(u_1^{\bullet}\right) + \exp\left(u_2^{\bullet}\right)\right) \\ &= \max(u_1^{\bullet}, u_2^{\bullet}) + \log\left(1 + \exp\left(-\left|u_1^{\bullet} - u_2^{\bullet}\right|\right)\right) \end{aligned} \tag{5.17}$$

b. $u_1^{\cdot} \neq u_2^{\cdot}$

$$\begin{aligned} u^{\bullet} &= \log\left|\exp\left(u_1^{\bullet}\right)-\exp\left(u_2^{\bullet}\right)\right| \\ &= \max(u_1^{\bullet},u_2^{\bullet})+\log\left(1-\exp\left(-\left|u_1^{\bullet}-u_2^{\bullet}\right|\right)\right) \end{aligned} \tag{5.18}$$

4.
$$\begin{aligned} u_1-u_2 &= LG(v_1-v_2) \\ &= LG\left(u_1^{\cdot}\exp(u_1^{\bullet})-u_2^{\cdot}\exp(u_2^{\bullet})\right) \\ &\triangleq \left(u^{\cdot},u^{\bullet}\right) \end{aligned} \tag{5.19}$$

where $u^{\cdot}$ and $u^{\bullet}$ can be determined similarly to (5.16)-(5.18).

In (5.17), $\log\left(1+\exp\left(-\left|u_1^{\bullet}-u_2^{\bullet}\right|\right)\right)$ can be obtained by table look-up. Similarly, in (5.18), $\log\left(1-\exp\left(-\left|u_1^{\bullet}-u_2^{\bullet}\right|\right)\right)$ can also be obtained by table look-up. Therefore, neglecting binary operations, the computations needed for (5.15) are one comparison, one addition and one table look-up. The above algorithm is called the Log-FFT-BP.

To summarize, (5.15) and (5.19) are used in the FFT; (5.13) and (5.14) are used in the forward-backward recursion. Also, calculating $\sum_{i\neq j} u_i$ for all j can be efficiently implemented by first calculating $\sum_i u_i$ then subtracting each u_j (similar idea cannot be applied to $\prod_{i\neq j} v_i$ for all j because of the 'divided by zero' problem).

With the technique above, for each iteration, the total required computations for the column step and the LG domain row step are $2NW_cq$ and $2MW_rq$ LG additions or subtractions, respectively; the FFT and IFFT overhead requires $2NW_cpq$ LG multiplications or divisions. Interestingly, for large p (and q), the FFT and IFFT overhead stands for most of the complexity. For instance, for $p=4$ ($q=2^p=16$), the FFT and IFFT overhead takes 2/3 of the total complexity.

In Table 5-I, the complexities of B-LDPC and Q-LDPC codes are compared. The complexity of B-LDPC codes is from [37]. For $p=4$, the Log-FFT-BP Q-LDPC decoding is 12 times more complex than the Log-BP B-LDPC decoding algorithm.

TABLE 5-I

Complexity comparison between B-LDPC and Q-LDPC codes

Per bit per iteration	Multiplication	Addition	Table look-up
B-LDPC (Max-Log-BP)	0	$4W_c$-1	$4W_c$
Q- LDPC (FFT-BP)	$\frac{q}{p}\left(W_c^2+4W_c\right)$	$2qW_c+\frac{q}{p}$	0
Q- LDPC (Log-FFT-BP)	0	$\left(2q+\frac{4q}{p}\right)W_c$	$2qW_c$

5.3 Code design for noise bursts

In this section, the term noise burst is used to include both burst erasures and thermal asperities. At most one burst per LDPC codeword is considered.

In principle, Q-LDPC codes can be generated by modifying B-LDPC codes. By substituting the 1's in the parity check matrix $\mathbf{H}^b$ for a B-LDPC code with elements from GF(q), a Q-LDPC parity check matrix $\mathbf{H}$ is obtained [64],[65]. It is shown in

[66] that the GF(q) elements replacing the 1's in $\mathbf{H}^b$ cannot be all the same, otherwise the resultant Q-LDPC code is simply comprised of p disjointed (also interleaved) B-LDPC codes.

Conceptually, any B-LDPC code (random or algebraic, regular or irregular) parity check matrix $\mathbf{H}^b$ can be used to generate a Q-LDPC code parity check matrix $\mathbf{H}$. However, since it is shown in Section 4.3 that geometric LDPC codes have large low weight multiplicities and irregular LDPC codes have larger decoding complexity than regular LDPC codes, only random regular Q-LDPC codes are considered.

5.3.1 Minimum space distance

For a low-density matrix, the *minimum space distance* (MSD) is defined as the minimum length of runs of zeros in all rows, and denoted as s. For example, $s = 1$ for the following matrix,

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

It is shown in [60] that a B-LDPC code with MSD s is guaranteed to recover a burst erasure of length s+1 bits. A p-bit-symbol Q-LDPC code with MSD s is guaranteed to recover a burst erasure of s+1 symbols, or of length ps+1 bits. Shorter burst erasures can be recovered in one LDPC iteration. Since one of the reasons for considering Q-LDPC codes is to improve the error correction capability under long bursts, the MSD should be maximized. For a matrix with N columns and row weight W_r, clearly $s < \frac{N}{W_r}$.

To obtain a parity check matrix $\mathbf{H}^b$ with large MSD, the following method is used. First, a reasonable value $s < \frac{N}{W_r}$ is chosen. Then starting from the first column, W_c locations are randomly chosen and filled with ones. For each latter column, both cycle-4 and MSD constraint are checked, and priority is given to the row locations with smallest current row weight (row weight of all previous columns). So, the generated matrix will have uniform W_c but not necessarily uniform row weights, but typically the row weights do not vary much.

Considering the complexity (see Section 5.2), GF(16) is probably the largest field that one can handle Q-LDPC codes and only codes with $W_c \le 3$ are considered. For sector size (4096 bits) codes, the Q-LDPC codes designed on GF(16) are summarized in Table 5-II.

TABLE 5-II

Q-LDPC codes on GF(16), $W_c = 3$

Code	N	M	R	s
1	1182	94	0.9205	20
2	1152	128	0.8889	30
3	1234	137	0.8890	30

Notice that Code 1 has rate 0.9205, while the maximum code rate for a $W_c = 3$, $M = 94$ LDPC code is 0.9267, showing that the MSD rule does not hinder the design of high-rate codes.

5.3.2 Performance of Q-LDPC codes on AWGN with burst erasures

The erasure is modeled as the received channel value being zero. Q-LDPC codes have excellent erasure recovery capability. Code 2 is chosen to illustrate the superiority of Q-LDPC codes over B-LDPC codes for erasure recovery. As shown in Table 5-II, Code 2 has $s = 30$. On a binary erasure channel, where only a single burst erasure can occur per codeword, this code is guaranteed to recover erasures of 31 symbols of 4-bit each. In the worst case, a length 118 bits erasure can result in a 31-symbol erasure, with the first and the last bit in the sequence the only erased bit in the corresponding erased symbols. Therefore, Code 2 is guaranteed to recover single burst erasures of length 118 bits.

Simulations, however, show that Code 2 is able to recover all single burst erasures of length up to 344 bits, and some single burst erasures of length 352 bits cannot be recovered. By comparison, a B-LDPC code with the same length (in bits) and code rate was designed, referred to as Code B, which has $s = 166$, and only has erasure recovery capability of 240 bits.

TABLE 5-III

MSD comparison between Q-LDPC and B-LDPC codes

	Code 2 (Q-LDPC)	Code B (B-LDPC)
MSD	30 symbols	166 bits
Actual burst erasure recovery capability	344 bits	240 bits

The performance of Code 2 and Code B on AWGN is shown in Figure 5.3, together with the uncoded performance. The channel is assumed to be binary-input with AWGN. It can be seen that the Q-LDPC code has steeper waterfall curve than the B-LDPC code.

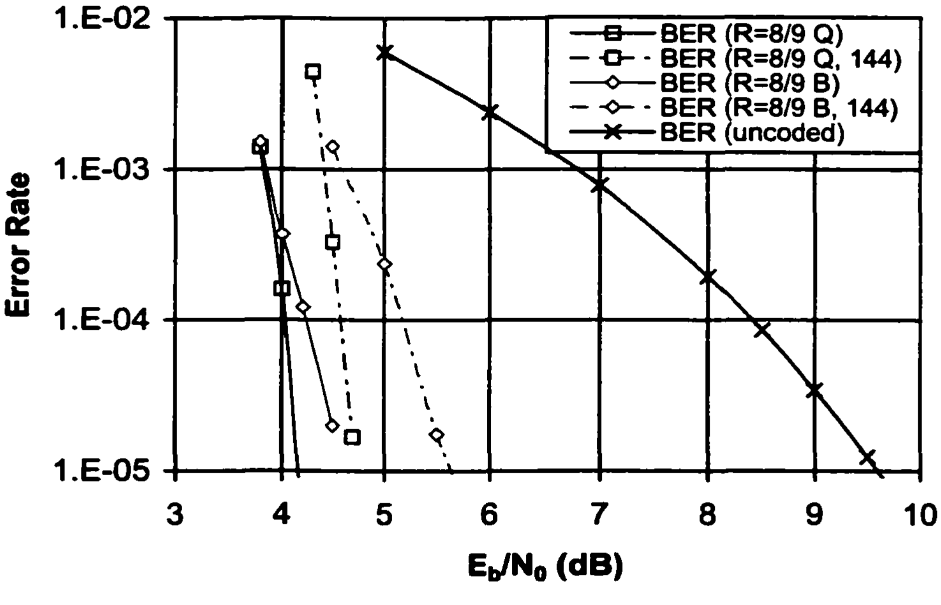


Figure 5.3. Performance of Code 2 and Code B on ISI-free AWGN channels.

Also shown in Figure 5.3 is the performance of Code 2 and Code B on AWGN with 144-bit erasures. For the same length of erasure, the performance degradation for the B-LDPC code is larger than for the Q-LDPC code.

5.3.3 Undetected burst and noise overestimation

In Section 5.3.2, the noise burst is assumed to be detected, and erasures are declared by setting the channel values to zero for the burst. If the burst is not detected, in the case of 100% fading, the channel values in the burst simply contain the noise.

When a channel has a noise burst in addition to AWGN, strictly speaking, the noise is non-stationary and the distribution of the noise is unknown. However, the Gaussian distribution is still assumed. The variance $\tilde{\sigma}^2$ of the overall noise is larger than the variance σ^2 of the AWGN. The use of a noise variance for the decoder larger than σ^2 is called *noise over-estimation.*

Table 5-IV shows the effect of noise over-estimation on the performance of Code 2 on a binary-input AWGN channel with undetected burst erasures. The erasure window is uniformly distributed throughout the codeword. Without noise over-estimation, something odd can be found from the table. With 48-bit undetected 100% erasures, the code performs worse at $E_b / N_0 = 10\,\text{dB}$ than at 7 dB. This can be explained as follows. At 7 dB, σ^2 is larger than at 10 dB. With noise over-estimation, the decoder assigns less confident probabilities to the erased symbols, which are prone to be in error. This helps the BP decoder to find the correct codewords. The effectiveness of noise over-estimation is shown by the results with 96-bit full erasures. At SNR 7 dB, the word error rate is 0.0024 without noise over-estimation, but no error occurs if the noise variance is over-estimated by 2 dB.

TABLE 5-IV

Performance of Code 2 with noise over-estimation

E_b/N_0 (dB)		48-bit 100% erasures	96-bit 100% erasures
AWGN	Estimated		
7	7	0	0.0024
	5		0
10	10	0.04	

5.4 Performance of Q-LDPC codes on MRCs

Q-LPDC codes perform well on AWGN channels, and they are resistant to long erasures. These two properties are exactly what MR systems need. In this section, Q-LDPC coded MR systems are investigated.

5.4.1 Q-LDPC coded EPR4-equalized channel

Shown in Figure 5.4 is the diagram of a proposed system. The Lorentzian channel is assumed. The channel is MMSE equalized to the EPR4 target. The rate 16/17 RLL code is not implemented in the simulation, rather it is included in the diagram to compensate the coding penalty present in the actual system.

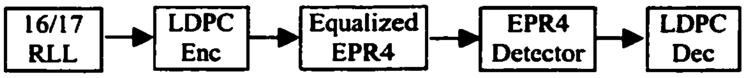


Figure 5.4. Q-LDPC coded EPR4-equalized MR system.

The system is simulated with Code 1 and Code 2, respectively, and is compared with the uncoded system, at user density $S_u = 2.505$. The BP decoder is set to perform at most 50 iterations. Turbo equalization is not implemented. Plotted in Figure 5.5 are the BER and symbol error rate performance. These two codes perform very similarly, and both provide over 3.5 dB gain over the uncoded system at BER 10^{-5}. At BER 10^{-5}, less than three iterations are executed on average.

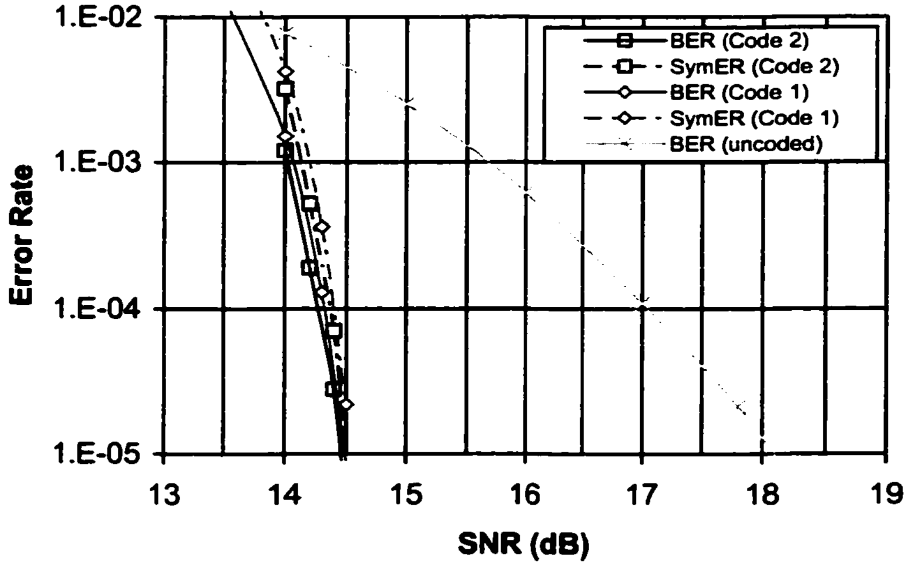


Figure 5.5. Performance of Q-LDPC coded EPR4-equalized MR channel.

The system in Figure 5.4 with Code 2 is also evaluated on AWGN in the presence of burst erasures at channel density $S_c = 2.975$. Each sector is assumed to have 48-bit full burst erasures. The test SNR is 19.5 dB, or at raw BER 6×10^{-6} excluding the erasures. Define $\Delta\,(\text{dB}) = \text{SNR (dB)-SNR}_{\text{est}}\,(\text{dB})$, Table 5-V shows the simulation results with different noise over-estimation. As in the case of AWGN, appropriate noise over-estimation is necessary for good performance. Without noise over-estimation ($\Delta = 0$ dB), all sectors simulated are in error. As Δ becomes larger, the performance improves. But when $\Delta > 12$ dB, the performance deteriorates. The best Δ is around 6 dB.

TABLE 5-V

Code 2 coded system at 19.5 dB with 48-bit full erasures and noise over-estimation

Estimated SNR (dB)	Sector Failure Rate
19.5, Δ=0 dB	≈1
16.5, Δ=3 dB	0.13
15.5, Δ=4 dB	0.003
14.7, Δ=4.8 dB	0
13.5, Δ=6 dB	0
10.5, Δ=9 dB	0
7.5, Δ=12 dB	≈1

This system is also simulated at SNR=19.5 dB with full erasures of different length, and the performance is shown in Table 5-VI. In the simulation, the noise over-estimation is Δ=9 dB. Roughly, this system is able to correct full erasures of length 160 bits.

Intuitively, the system should be able to correct longer partial erasures than full erasures. Performance of the system at SNR=19.5 dB with 50% erasures is shown in

Table 5-VII. Also Δ=9 dB of noise over-estimation is used. It can be seen that 280-bit half erasures can be corrected, almost doubled the length for full erasures.

TABLE 5-VI

Performance of Code 2 coded system with different full erasures

Length of Defects (bit)	Sector Error Rate
48	0/3000
80	0/5000
120	0/5000
160	0/5000
200	1/5000
240	41/5000
280	80/5000

TABLE 5-VII

Code 2 system performance at 19.5 dB with 50% erasures

Length of Defect (bit)	Sector Error Rate
48	0/3000
80	0/5000
120	0/5000
160	0/5000
200	0/5000
240	0/5000
280	0/5000
320	2/5000
400	61/5000

When a thermal asperity (TA) occurs, the analog-to-digital converter is saturated for a period of time. For simplicity, the TA is modeled as a rectangular window in which the readback signals equal the maximum signal level possible for the PR target. Table 5-VIII shows the simulation result of the system with TAs. The length of correctable TA is 80 bits, which is not as good as for erasures. The reason, intuitively, is the fact that the TA is equivalent to a noise with variance $16-4=12$, while full erasures are equivalent to a noise with variance 4.

TABLE 5-VIII

System performance with different TA lengths

Length of TA (bit)	Sector Error Rate
48	0/5000
80	0/5000
120	13/5000

In practice, TAs might be detectable. In that case, the channel values in the TA window can simply by zeroed out. The noise condition is therefore improved and the system must perform better than with full erasures of the same length as the TA. Furthermore, one can perform channel detection excluding the TA window, and set the LLR to zero in the TA window, as done in [60].

It is verified through the above simulations that Q-LDPC codes perform well on MR channels with noise bursts. Since the SNR is quite high in these simulations, the results reflect the error correction capability on erasure-dominant systems. For a practical system, it is necessary to know the performance of the system at lower SNR. An extensive simulation was carried out for the system and shown in Figure 5.4 at SNR=17 dB with 80-bit full erasures. Out of 10^7 sectors simulated, only 3 sectors were in error, which corresponds to a sector error rate at 3×10^{-7} roughly. The bit error distribution of the three sectors in error is plotted in Figure 5.6. It can be seen that the bit errors have been spread, and in fact scattered, therefore, an RS code concatenated to the Q-LDPC code would not be effective.

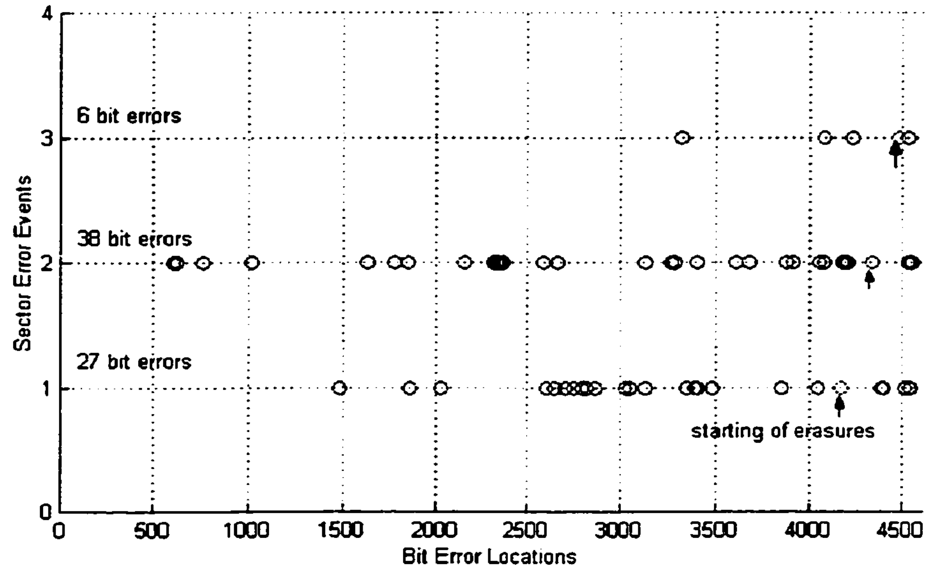


Figure 5.6. Bit error locations of the sectors in error.

5.4.2 Q-LDPC vs. RS systems on equalized ME^{2}PR4

In the previous section, the Q-LDPC codes are examined on an EPR4-equalized Lorentzian channel with AWGN. The performance of the Q-LDPC coded system is compared with the uncoded system. In this section, media noise is included, and the performance of a Q-LDPC coded system is compared with current RS coded systems, on ME^{2}PR4-equalized Lorentzian-Gaussian channels.

Shown in Figure 5.7 is the system diagram of a HDD system. For simulation purposes, a variation of Figure 5.7 is shown in Figure 5.8, with the compensation for 16/17 code rate included. The random data at the input of the MRC are assumed to be RS codewords, and pseudo-RS decoding is performed. The overall code rate is 0.8425.

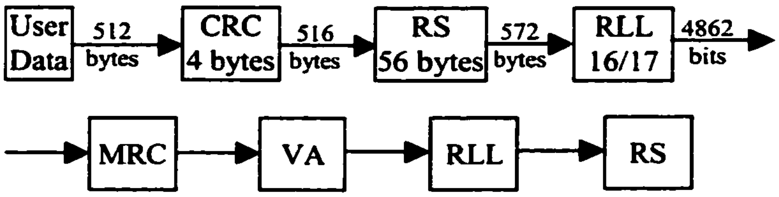


Figure 5.7. An RS coded system.

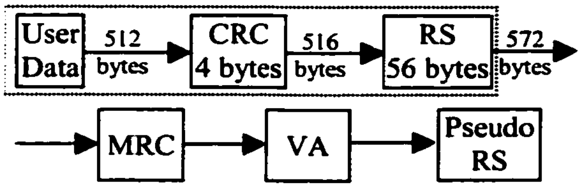


Figure 5.8. Model for an RS coded system.

The proposed Q-LDPC system is shown in Figure 5.9. These two systems have similar code rates. The Q-LDPC code is Code 3 in Table 5-II with rate 0.8890 and MSD $s = 30$, whose performance on ISI-free AWGN channel is within 0.1 dB from that of Code 2, as shown in Figure 5.3. The overall code rate is 0.8298, close to the RS system.

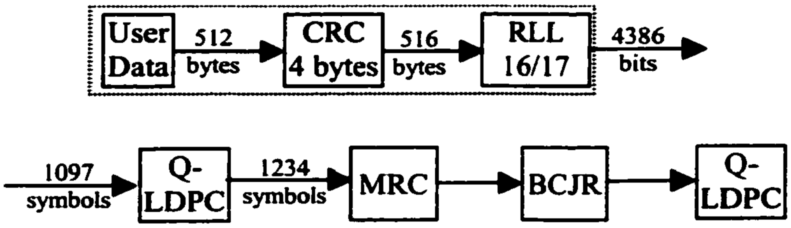


Figure 5.9. Proposed Q-LDPC coded system.

The two systems are simulated at $S_u = 2.5$ on channels with purely AWGN, and also on channels with 90% jitter noise power. Turbo equalization is not performed and at most 50 LDPC iterations are allowed.

Shown in Figure 5.10 are the results of the Q-LDPC and RS coded systems under purely AWGN, with both sector error rate and byte (8-bit) error rate shown. At sector error rate 10^{-4}, the proposed Q-LDPC coded system outperforms current RS system by 2.2 dB. Also shown in Figure 5.10 is the sector error rate performance of the irregular B-LDPC code in Section 4.4, which is thought to be the best B-LDPC code for the ME^{2}PR4 channel. It can be observed that not only the Q-LDPC code performs about 0.5 dB better than the irregular B-LDPC at $\text{SER} = 10^{-4}$, the Q-LDPC curve is much steeper, indicating even larger gains at lower error rates.

Shown in Figure 5.11 are the results of the Q-LDPC and RS coded systems under 90% jitter noise power, also with both sector error rate and byte (8-bit) error rate shown. At a sector error rate of 10^{-4}, the proposed Q-LDPC coded system outperforms current RS system by 1.4 dB.

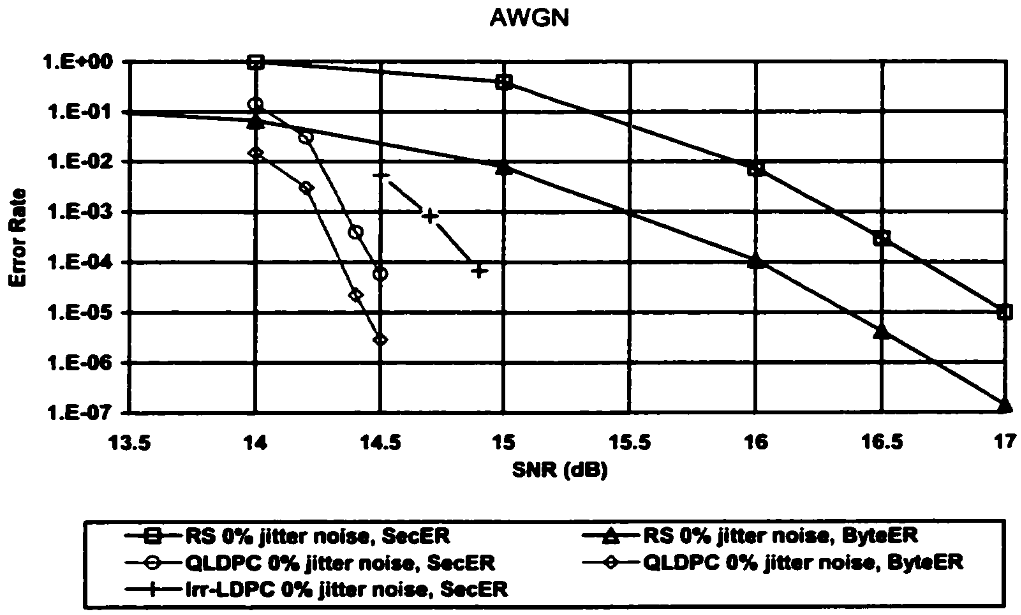


Figure 5.10. Performance on channels with purely AWGN.

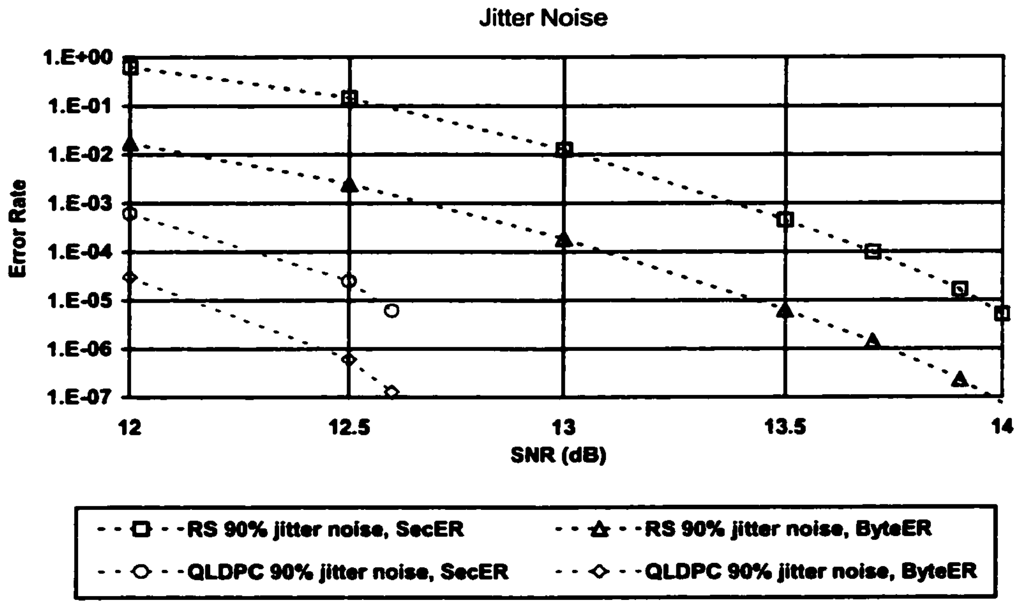


Figure 5.11. Performance on channels with 90% jitter noise power.

The erasure performance of the systems is summarized in Table 5-IX. With purely AWGN, the raw BER at $\mathrm{SNR}=18\,\mathrm{dB}$ is approximately 3e-4; and with 90% jitter noise power, the raw BER at $\mathrm{SNR}=15\,\mathrm{dB}$ is also around 3e-4. In both cases, the Q-LDPC system cannot correct 80-bit burst erasures.

TABLE 5-IX

Erasure performance of Q-LDPC equalized ME^{2}PR4 system

Noise	SNR	Full Erasure	Sectors Simulated	Failed Sectors
AWGN	17 dB	80-bit	10,000	29
	18 dB	80-bit	5,000	3
Jitter Noise	14 dB	80-bit	10,000	153
	15 dB	64-bit	5,000	6

Compared with the Q-LDPC coded EPR4 system, shown in Table 5-VI, both systems have raw channel BER 10^{-3}-10^{-4}, same erasure length, and same code rate, similar length, but the ME^{2}PR4 system does not perform as well as the EPR4 system. Since the only significant difference is the PR target, the channel BCJR output was examined and compared for the two systems.

Shown in Figure 5.12 are the channel APP output LLRs of a sector in error, on the equalized ME^{2}PR4 channel at the channel density in Table 5-IX and SNR=18 dB with 80-bit full erasures. It can be seen that the channel APP output LLRs in the erasure window have the same sign and almost the same close-to-zero magnitude. This can be easily explained as follows. Since the channel values in the erasure window are close to zero, a sequence of non-transitions will be the most likely data. This explains the similarity of the sign. Furthermore, the sequence of 0's and the sequence 1's have very close probabilities, which explains the small magnitude. However, the non-zero values stand for the "leakage" from outside the erasure window due to the partial response.

Shown in Figure 5.13 are the channel APP output LLRs of a sector in error on the equalized EPR4 channel at the same channel density and SNR (17 dB) as in Figure 5.6 with 80-bit full erasures.

For sectors in error, the average LLR magnitude inside the erasure window is obtained through simulation. So is the average LLR magnitude outside the erasure window. The ratio of the former to the latter is found to be 0.41 for the ME^{2}PR4-equalized channel and 0.23 for the EPR4-equalized channel. The large magnitude in the erasure window represents large noise. Therefore, the erasure performance of the ME^{2}PR4 system is not as good as the EPR4 system.

If the full erasure or TA is detected, then by zeroing the channel APP output LLRs in the erasure window, in all cases in Table 5-IX, the 80-bit noise bursts are correctable.

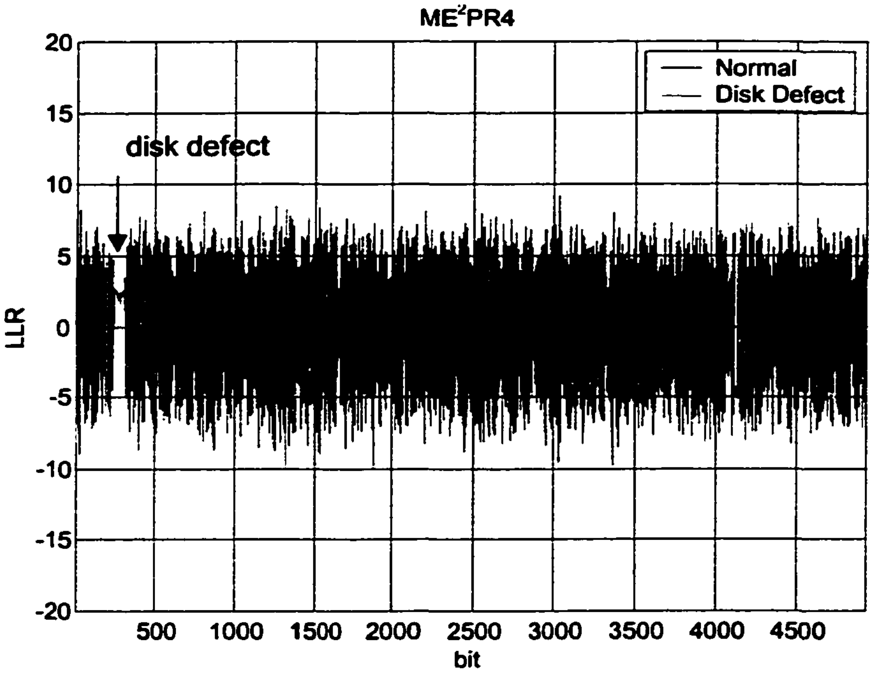


Figure 5.12. Equalized ME^{2}PR4 channel BCJR output with 80-bit full erasures.

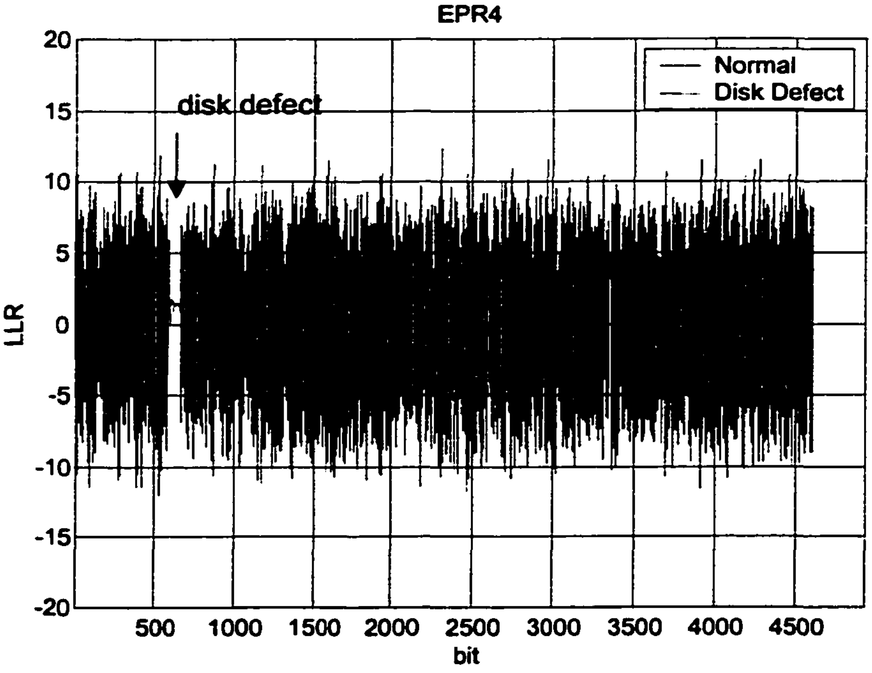


Figure 5.13. Equalized EPR4 channel BCJR output with 80-bit full erasures.

5.5 Array codes

Array codes refer to a class of codes defined on a two-dimension array and are very good for burst error detection and correction [73]. They can be constructed with the symbols lying in rings [76]. Algebraic decoding for array codes is similar to that for RS codes. Recently, array codes were found to have binary low-density parity-check matrices, and therefore are LDPC codes [74],[75].

For a prime number p, the codewords of an array code can be defined as a square matrix $\mathbf{A}_{p\times p}=\left[\mathbf{A}_0,\mathbf{A}_1,\ldots,\mathbf{A}_{p-1}\right]$, where each $\mathbf{A}_i$, $i=0,1,\ldots,p-1$ is a column vector and also a symbol. Parameter r defines the number of parity check symbols and $\mathbf{A}$ is a codeword if for all $k=0,1,\ldots,r-1$ and $i=0,1,\ldots,p-1$,

$$\sum_{j=0}^{p-1} A_{\langle i+jk\rangle_p,j}=0. \tag{5.20}$$

where $\langle\bullet\rangle_p$ is the modulo-p residue.

The constraints in (5.20) can be rewritten as

$$\begin{bmatrix} I & I & \cdots & I \\ I & \sigma & \cdots & \sigma^{p-1} \\ \vdots & \vdots & & \vdots \\ I & \sigma^{r-1} & \cdots & \sigma^{(p-1)(r-1)} \end{bmatrix}\begin{bmatrix} \mathbf{A}_0 \\ \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_{p-1} \end{bmatrix} \triangleq \mathbf{H}\begin{bmatrix} \mathbf{A}_0 \\ \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_{p-1} \end{bmatrix}=\mathbf{0} \tag{5.21}$$

where σ is $p\times p$ single-cyclic-shift matrix for a column vector of length p. $\sigma=\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$ for $p=3$, for example. The $rp\times p^2$ parity check matrix $\mathbf{H}$ in (5.21) is a sparse matrix with column weight $W_c = r$ and free of cycles of length four. Therefore the array code is in fact a B-LDPC code.

Since each submatrix of $\mathbf{H}$ containing p rows adds up to an all-one row vector, the rank of $\mathbf{H}$ is $r(p-1)+1$. Thus the code rate is $1-\frac{r(p-1)+1}{p^2}$.

Array codes can be shortened by using fewer columns of $\mathbf{H}$, for instance

$$\begin{bmatrix} I & I & \cdots & I \\ I & \sigma & \cdots & \sigma^{k-1} \\ \vdots & \vdots & & \vdots \\ I & \sigma^{r-1} & \cdots & \sigma^{(k-1)(r-1)} \end{bmatrix}$$

defines the parity check matrix of a length-k array code with p-bit symbol size. Also, short cycles can be eliminated to improve the performance with BP decoding [74].

With algebraic decoding, an array code with r check symbols is able to correct one symbol error and r-2 symbol erasures, or r symbol erasures. The burst erasure recovery capability is better than B-LDPC and Q-LDPC codes. For example, the array code with $p=71$, $k=65$, and $r=7$, thus code rate $R=0.8936$ is able to correct seven symbol erasures. If the erasure is a single consecutive bit sequence, this is at least $(7-1)\times 71+1=427$ bits, better than the B-LDPC and Q-LDPC codes discussed in Section 5.3. Also, multiple erasures can also be recovered with guarantee, which is not addressed for B-LDPC and Q-LDPC codes. It is interesting that with MSD code design, small W_c results in large MSD thus large (guaranteed) burst erasure recovery capability, however, we see here large r (i.e. W_c) results large burst erasure recovery capability. One thing should be noted, though, that the error recovery is done in one LDPC iteration in the former case but in more than one iteration in the latter case, if BP decoding is used.

With BP decoding, the burst erasure recovery capability should not be inferior to that with algebraic decoding. Also, it is shown in [74],[75] that the performance of sector size high-rate array codes is slightly inferior to similar random B-LDPC codes.

Array codes are potentially useful on MRCs in which noise bursts are detectable and occur more often than random errors. When random errors are dominant, a random B-LDPC code is a better option because of its small W_c and Q-LDPC is an even better option because of its good performance.

5.6 Conclusions

Q-LDPC coded MR systems were described in this chapter, and code design for burst erasure was developed. On an equalized EPR4 channel, it is shown that a Q-LDPC code performs very well with random noise and noise burst. On an equalized ME^{2}PR4 channel, it is shown that under random noise, a Q-LDPC system outperforms an RS system with similar code rate.

BP decoding was extended to Q-LDPC codes and an FFT technique was adopted for efficient decoding. In addition, a logarithm domain efficient decoding algorithm is developed. However, it is shown that Q-LDPC codes are about ten times more complex than B-LDPC codes.

Array codes are good codes in terms of burst erasure recovery. However, the performance of array codes with both random noise and noise burst needs further research.

CONCLUSIONS AND FURTHER WORK

Three coding schemes, for which soft iterative decoding is performed, were investigated for usage in the next generation of MR systems. Block turbo codes achieve moderate coding gains, but suffer either low code rate or potentially high error floor. B-LDPC codes provide substantial gains under random noise and show no error floor up to BER=10^{-7}. At moderately low BER (~10^{-6}), typically very few iterations are actually performed. However, there are many (several tens) bit errors given a sector error. Q-LDPC codes also provided substantial gains under random noise, outperforming RS codes by several dB. Q-LDPC codes are also robust to noise bursts. Efficient and simplified Q-LDPC decoding was investigated, but the decoding complexity of Q-LDPC codes is still about an order of magnitude larger that B-LDPC codes.

LDPC coding seems to be promising for MR systems. In systems where both random noise and noise bursts substantially contribute to the errors, a near-term solution might be the concatenation of a very-high-rate B-LDPC code and a RS code. In such a system, the role of the B-LDPC code is to correct some or all random bit errors and reduce the number of symbol errors to within the ECC capability of the RS codes. A single high-rate Q-LDPC code might be the ultimate solution should the semiconductor technology allow.

The performance of array codes on MRCs needs to be investigated further, especially on noise burst dominant environments.

As might be noted, all results presented are at some BER larger than 10^{-8}. This is probably the limit for computer simulation, but still far from the actual operating point. Experiments must be carried out before any final conclusion can be made. This is the more important but also more difficult, or even impossible, future work.

This dissertation includes the following original contributions.

- Product codes were introduced for MR systems, and the system performance was evaluated.
- B-LDPC coded MR systems were introduced, and substantial gains were achieved. The characteristics of the iterative decoding were investigated. It was shown that RS-LDPC concatenation is not beneficial in systems where random errors are dominant.
- Q-LDPC coded systems were proposed, which improve the system performance under long erasures. An efficient decoding algorithm was developed for Q-LDPC codes. Q-LDPC systems were shown to outperform RS systems.

REFERENCES

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

[2] N. Bertram, *Theory of Magnetic Recording*. Cambridge, England: Cambridge University Press, 1994.

[3] S. Wang and A. Taratorin, *Magnetic Information Storage Technology*. San Diego, CA: Academic Press, 1999.

[4] W. G. Bliss, S. She and L. C. Sundell, "The performance of generalized maximum transition run trellis codes," *IEEE Trans. Magn.*, vol. 34, pp. 85-90, Jan. 1998.

[5] J. Moon, "Discrete-time modeling of transition-noise-dominant channels and study of detection performance," *IEEE Trans. Magn.*, vol. 27, pp. 4573-4578, Nov. 1991.

[6] J. Proakis, *Digital Communications*, 3rd Edition. New York: McGraw-Hill, 1995.

[7] G. D. Forney, JR., "Maximum-likelihood sequence estimation of digital sequence in the presence of intersymbol interference," *IEEE Trans. Inform. Theory*, vol. 18, pp. 363-378, May 1972.

[8] R. D. Cideciyan, F. Dolivo, R. Hermann, W. Hirt, and W. Schott, "A PRML system for digital magnetic recording," *IEEE J. Select. Areas Commun.*, vol. 10, pp. 38-56, Jan. 1992.

[9] 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. 2868-2873.

[10] J. Moon and L. Carley, *Sequence Detection for High-Density Storage Channels.* Boston, MA: Kluwer Academic Publishers, 1992.

[11] K. Immink, P. Siegel and J. Wolf, "Codes for digital recorders," *IEEE Trans. Inform. Theory*, vol. 44, pp. 2260-2299, Oct. 1998.

[12] J. Moon and B. Brickner, "Maximum transition run codes for data storage systems," *IEEE Trans. Magn.* vol. 32, pp. 3992-3994, Sept. 1996.

[13] K. K. Fitzpatrick and C. S. Modlin, "Time-varying MTR codes for high density magnetic recording," in *Proc. IEEE Global Telecommun. Conf.*, 1997, pp. 1250-1253.

[14] T. Souvignier, A. Friedman, M. Oberg, P. H. Siegel, R. E. Swanson, and J. K. Wolf, "Turbo decoding for PR4: Parallel versus serial concatenation," in *Proc. IEEE Int. Conf. Commun.*, 1999, pp. 1638-1642.

[15] J. Li, E. Kurtas, K.R. Narayanan and C.N. Georghiades, "On the performance of turbo product codes and LDPC codes over partial-response channels," in *Proc. IEEE Int. Conf. on Commun*, 2001, pp. 2176 –2183.

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

[17] M. Oberg and P.H. Siegel, "Parity check codes for partial response channels," in *Proc.IEEE Global Telecommun. Conf.*, 1999, pp. 717 –722.

[18] J. Moon, "Signal-to-noise ratio definition for magnetic recording channels with transition noise," *IEEE Trans. Magn.*, vol. 36, pp. 3881-3883, Sept. 2000.

[19] W. E. Ryan, "Optimal code rates for concatenated codes on a PR4-equalized magnetic recording channel," *IEEE Trans. Magn.* vol. 36, pp. 4044-4049, Nov. 2000.

[20] J. Bergmans, "Discrete-time models for digital magnetic recording," *Philips J. Res.* vol. 41, pp. 531-558, 1986.

[21] W. E. Ryan, "Performance of high rate turbo codes on a PR4-equalized magnetic recording channel," in *Proc. IEEE Int. Conf. Commun.*, 1998, pp. 947-951.

[22] W. E. Ryan, L. L. McPheters, and S. W. McLaughlin, "Combined turbo coding and turbo equalization for PR4-equalized Lorentzian channels," in *Proc. Conf. Inform. Sciences and Systems*, 1998, pp. 489-494.

[23] R. Koetter and A. Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes," to appear *IEEE Trans. Inform. Theory.*

[24] R.M. Roth and G. Ruchenstein, "Efficient decoding of Reed-Solomon codes beyond half the minimum distance," *IEEE Trans. Inform. Theory*, vol. 46, pp. 246-257, Jan. 2000.

[25] H. Xia, H. Song and J. R. Cruz, "Retry mode soft Reed-Solomon decoding," in *Digest IEEE Inter. Magnetics Conf., (INTERMAG)*, 2002.

[26] T. Conway, "A new target response with parity check coding for high density magnetic recording channels," *IEEE Trans. Magn.* vol. 34, pp. 2382-2386, July 1998.

[27] 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.

[28] R. M. Pyndiah, "Near optimum decoding of products codes: Block turbo codes," *IEEE Trans. Commun.*, vol. 46, pp. 1003-1010, Aug. 1998.

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

[30] J. Wolf and G. Ungerboeck, "Trellis coding for partial response channels," *IEEE Trans. Commun.*, vol. 34, pp. 765-773, Aug. 1986.

[31] T.M. Duman and E. Kurtas, "Performance bounds for high rate linear codes over partial-response channels," *IEEE Trans. Inform. Theory*, vol. 47, pp. 1201-1205, Mar. 2001.

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

[33] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inform. Theory*, vol. 18, pp. 170-182, Jan. 1972.

[34] 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.

[35] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," in *Proc. IEEE Global Telecommun. Conf.*, 1989, pp. 1680-1686.

[36] G.D. Forney Jr., *Concatenated Codes*, Cambridge, MA: MIT Press, 1966.

[37] W. G. Bliss, "Circuitry for performing error correction calculations on baseband encoded data to eliminate error propagation," *IBM Tech. Disclosure Bulletin*, vol. 10, pp. 38-56, Jan. 1992.

[38] J. L. Fan, *Constrained Coding and Soft Iterative Decoding for Storage*, Ph.D. Dissertation, Stanford University, 1999.

[39] K.A.S. Immink, "A practical method for approaching the channel capacity for constrained channels," *IEEE Trans. Inform. Theory*, vol. 43, pp. 1389-1399, Sept. 1997.

[40] K.D. Anim-Appiah and S.W. McLaughlin, "Constrained-input turbo codes for (0,k) RLL channels," in *Proc. 1999 Conf. on Inform. Sciences and Systems*.

[41] S. Benedetto and G. Montorsi, "Unveiling turbo codes: some results on parallel concatenated coding schemes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 409-428, Mar. 1996.

[42] H. Song and J. R. Cruz, "Block turbo codes for magnetic recording channels," in *Proc. IEEE Int. Conf. on Commun.*, 2000, pp. 85-88.

[43] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. 8, pp. 21-28, Jan. 1968.

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

[45] M.G. Luby, M. Mitzenmacher, M. Shokrollahi and D.A. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. Inform. Theory*, vol. 47, pp. 585-698, Feb. 2001.

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

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

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

[49] 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.

[50] I. S. Reed, "A class of multiple-error-correcting codes and decoding schemes," *IRE Trans. Inform. Theory*, vol. 4, pp. 38-49, Sept. 1954.

[51] R. J. McEliece, D. J. C. MacKay and J.-F. Cheng, "Turbo decoding as an instance of Pearl's 'belief propagation' algorithm," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 140-152, Feb. 1998.

[52] 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.

[53] T. Mittelholzer, A. Dholakia, and E. Eleftheriou, "Reduced-complexity decoding of LDPC codes for generalized partial response channels," *IEEE Trans. Magn.*, vol. 37, pp. 721-728, Mar. 2001

[54] 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.

[55] E. Yeo, P. Pakzad, B. Nikolic and V. Anantharam, "High throughput low-density parity-check decoder structures," in *Proc. IEEE Global Telecommun. Conf.*, 2001, pp. 3019-3024.

[56] 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.

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

[58] S. Lin and D. J. Costello, Jr., *Error Control Coding: Fundamental and Applications*, Englewood Cliffs, NJ: Prentice Hall, 1983.

[59] W. Peterson and E. Weldon, *Error-Correcting Codes*, Cambridge, MA: MIT Press, 1972.

[60] 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.

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

[62] 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, no. 5, pp. 918-923, May 2001.

[63] R. M. Todd, Private communication.

[64] 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.

[65] 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.

[66] M. C. Davey, *Error-correction using Low-Density Parity-Check codes*, Ph.D Dissertation, Univ. of Cambridge, Dec. 1999.

[67] H. Song R. M. Todd and J. R. Cruz, "Performance of low-density parity-check codes on magnetic recording channels," *Proc. 2nd Int. Symp. on Turbo Codes and Related Topics*, France, Sept. 2000, pp. 395-398.

[68] H. Song, R. M. Todd and J. R. Cruz, "Low density parity check codes for magnetic recording channels," *IEEE Trans. Magn.* vol. 36, no. 5, pp. 2183-2186, Sept. 2000.

[69] D. J. C. MacKay, S. T. Wilson and M. C. Davey, "Comparison of constructions of irregular Gallager codes", *IEEE Trans. Commun.*, vol. 47, pp. 1449-1454, Oct. 1999.

[70] 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.

[71] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 429-445, Mar. 1996.

[72] T. Souvignier, C. Argon, S. W. McLaughlin and K. Thamvichai, "Turbo product codes for partial response channels," in *Proc. IEEE Int. Conf. Commun.*, 2001, pp. 2184-2188.

[73] S. B. Wicker, *Error control system for digital communications and storage*, New York: Prentice Hall, 1995.

[74] J. L. Fan, "Array codes as low-density parity-check codes," *Proc. 2nd Int. Symp. on Turbo Codes and Related Topics*, France, pp. 543-546, Sept. 2000.

[75] E. Eleftheriou and S. Olcer, "G.gen: LDPC codes for G.dmt.bis and G.lite.bis," *ITU Telecommunication Standardization Sector*, Study Group 15, Clearwater, Florida, 8-12 Jan. 2001, Temporary Document CF-060.

[76] M. Blaum and R. Roth, "New array codes for multiple phase burst correction," *IEEE Trans. Inform. Theory*, vol. 39, pp. 66-77, Jan. 1993.

APPENDIX A. SOFT OUTPUT CHANNEL DETECTION AND DECODING

The output of a PR channel detector or an error-correcting decoder may be hard decisions (0 or 1) or probabilities associated with the bit being a one or a zero, called soft decision. In a system with multi-stage decoding and/or detection, soft decision from the previous stage may improve the performance of the next stage.

Suppose the data vector $\mathbf{x}$ is transmitted, and the received vector is $\mathbf{r}$, the required soft information is $P(x_i = 1 | \mathbf{r})$ or equivalently $P(x_i = 0 | \mathbf{r})$, where x_i stands for the i-th bit in the transmitted vector. A device that performs this function (MAP or approximate) is called an *a posterior probability* (APP) module. Bahl-Cocke-Jelinek-Raviv (BCJR) and soft output Viterbi algorithm (SOVA) are two techniques use to implement the APP module.

A.1 BCJR algorithm for PR channels

The BCJR algorithm is an efficient realization of the MAP detector for AWGN PR channels (in general, any Markov process) with finite number of states. Using the trellis of the given PR polynomial, the algorithm updates the conditional probabilities in a recursive fashion. The following description is based on [34].

Let $\mathbf{x} = \{x_1, x_2, \cdots, x_N\}$. The prior information for x_i is known as,

$$\Lambda^{prior}(x_i)=\frac{P(\mathrm{x}_i=+1)}{P(\mathrm{x}_i=-1)}. \tag{A.1}$$

The noiseless channel vector is corrupted by the noise vector $\mathbf{n}=\{n_1,n_2,\cdots,n_N\}$, resulting in the received noisy channel vector $\mathbf{r}=\{r_1,r_2,\cdots,r_N\}$. This Markov process can be depicted as a linear trellis diagram with N+1 time stages, from 0 to N, driven by $\mathbf{x}$, and M states in each stage. Let the M states in the state space be labeled as 0, 1, ..., M-1. The state of the process at time stage i is labeled as S_i, where $i=0,1,\cdots,N$. The initial state S_0 and final state are known.

In the following, the notation $\mathbf{r}_i^j$ is defined as the vector $r_i,r_{i+1},\cdots,r_j$.

The state transition probabilities are known as

$$P(m\mid m')=P(S_i=m\mid S_{i-1}=m') \tag{A.2}$$

and

$$P(x\mid m',m)=P(x_i=x\mid S_{i-1}=m',S_i=m) \tag{A.3}$$

Define

$$\alpha_i(s)=P(S_i=s,\mathbf{r}_1^i) \tag{A.4}$$

$$\beta_i(s)=P(\mathbf{r}_{i+1}^N\mid S_i=s) \tag{A.5}$$

and

$$\begin{aligned}\gamma_i(s',s)&=P(S_i=s,r_i\mid S_{i-1}=s')\\&=P(S_i=s\mid S_{i-1}=s')P(r_i\mid S_i=s,S_{i-1}=s').\end{aligned} \tag{A.6}$$

In (A.6), $P(S_i=s\mid S_{i-1}=s')$ is the prior information, and $P(r_i\mid S_i=s,S_{i-1}=s')$ depends on the noise vector $\mathbf{n}$, or simply on n_i if the noise is independent. $\alpha_0(s)$

and $\beta_N(s)$ are initialized according to the initial and ending states. Then the recursion is done as

$$\alpha_i(s) = \sum_{s' \in S} \alpha_{i-1}(s')\gamma_i(s',s) \tag{A.7}$$

and

$$\beta_i(s) = \sum_{s' \in S} \gamma_{i+1}(s,s')\beta_{i+1}(s'). \tag{A.8}$$

The soft information, which is the *likelihood ratio*, for x_i is calculated as

$$\Lambda(x_i) = \frac{P(x_i = +1 \mid \mathbf{r}_1^N)}{P(x_i = -1 \mid \mathbf{r}_1^N)} = \frac{\sum_{S^+} \alpha_{i-1}(s')\gamma_i(s',s)\beta_i(s)}{\sum_{S^-} \alpha_{i-1}(s')\gamma_i(s',s)\beta_i(s)} \tag{A.9}$$

where S^+ and S^- are the set of transition from state s' to s that is driven by input $+1$ and -1, respectively.

$\Lambda(x_i)$ can be rewritten as,

$$\Lambda(x_i) = \Lambda^{priori}(x_i) \cdot \Lambda^{ext}(x_i) \tag{A.10}$$

in which $\Lambda^{ext}(\mathbf{x}_i)$ is called the *extrinsic information.*

The above algorithm is a MAP algorithm. A similar recursive algorithm, Log-MAP, can be realized in the logarithm domain [71]. Also, the MAX-Log-MAP simplifies Log-MAP by using table look-up instead of logarithmic computations [71].

A.2 SOVA for PR channels

The BCJR algorithm yields optimal bit-wise (symbol-wise) estimation for ISI channels with AWGN. However, the complexity is high since all possible paths are traced. SOVA modifies the VA to produce soft information, and is a low complexity sub-optimal detection algorithm [35]. SOVA is used in Chapter 1 in block turbo coded MR channels.

Shown in Figure A.1 is trellis section for a PR channel. In addition to the VA, soft information is stored for each bit on every survivor path. The soft information is defined as the probability of a hard decision being in error.

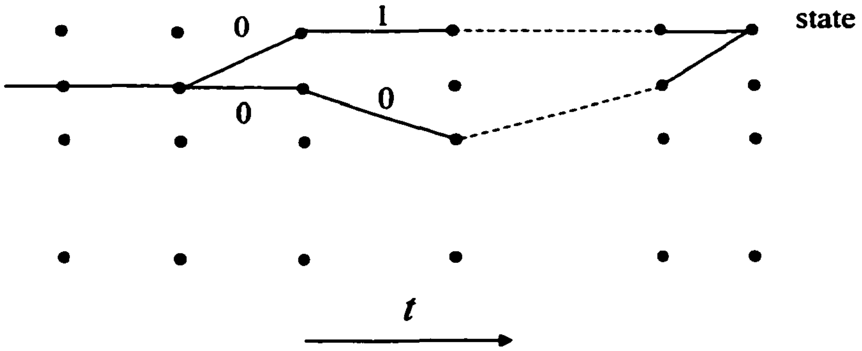


Figure A.1. Trellis diagram for a PR channel.

In the update at time i at state s, two candidate paths are compared and the one with smaller Euclidean path metric is selected. Denote the selected path with path 1 and the other path 2, and their path metric M_1 and M_2, respectively. Assuming AWGN, the probability of selecting the wrong path is

$$p_{s,i} = \frac{1}{1+e^{M_2-M_1}}.$$

The probability of the j-th bit being wrong is denoted as p_j on each survivor path. At places where the hard decisions differ on path 1 and path 2, the soft information is updated as,

$$p_j \leftarrow p_j(1-p_{s,k})+(1-p_j)p_{s,j} \tag{A.11}$$

or in the log-likelihood ratio domain in which $L_j = \log\frac{1-p_j}{p_j}$,

$$L_j \leftarrow \log\frac{1+e^{\Delta}e^{L_j}}{e^{\Delta}+e^{L_j}} \tag{A.12}$$

where $\Delta = M_2 - M_1$.

At the beginning of the SOVA, p_j is initialized to the $\Lambda^{prior}(x_i)$ in (A.1). Similarly to (A.10), the output likelihood ratio can be written as

$$\Lambda(x_i) = \Lambda^{prior}(x_i)\cdot\Lambda^{ext}(x_i)\,.$$

APPENDIX B. PRECODED PR CHANNELS

Two questions are to be answered in this appendix.

1. What is the relationship between the soft output and the hard decision, or can the soft output be used to predict performance?
2. What is the effect of the precoder on performance?

The effect of the precoder on PR channels performance has been widely noticed [14],[16],[17]. In order to simplify the description of the quality of the soft information, mean reliability, is defined as [16]

$$\gamma_0 = E\left\{x_i\left[p(x_i = 1) - p(x_i = -1)\right]\right\}, \tag{B.1}$$

where x_i is the i-th bit of the recorded data vector. It is shown in [16] that γ_0 is consistent with the hard decision performance.

B.1 Ideal PR channels

In this test, the channel is assumed to be an ideal PR channel. The output of the channel is corrupted by AWGN. The SNR for ideal PR channels is defined as the ratio of the mean signal power over the noise variance. For example, the energy in the EPR4 polynomial $1 + D + D^2 + D^3$ is 4.

Shown in Figure B.1 is the mean reliability of PR4, EPR4 and ME^{2}PR4 with different precoding, namely no precoding, $\frac{1}{1 \oplus D}$ and $\frac{1}{1 \oplus D^2}$ precoding. The results are obtained by Monte-Carlo simulation of 100 length 4096-bit blocks.

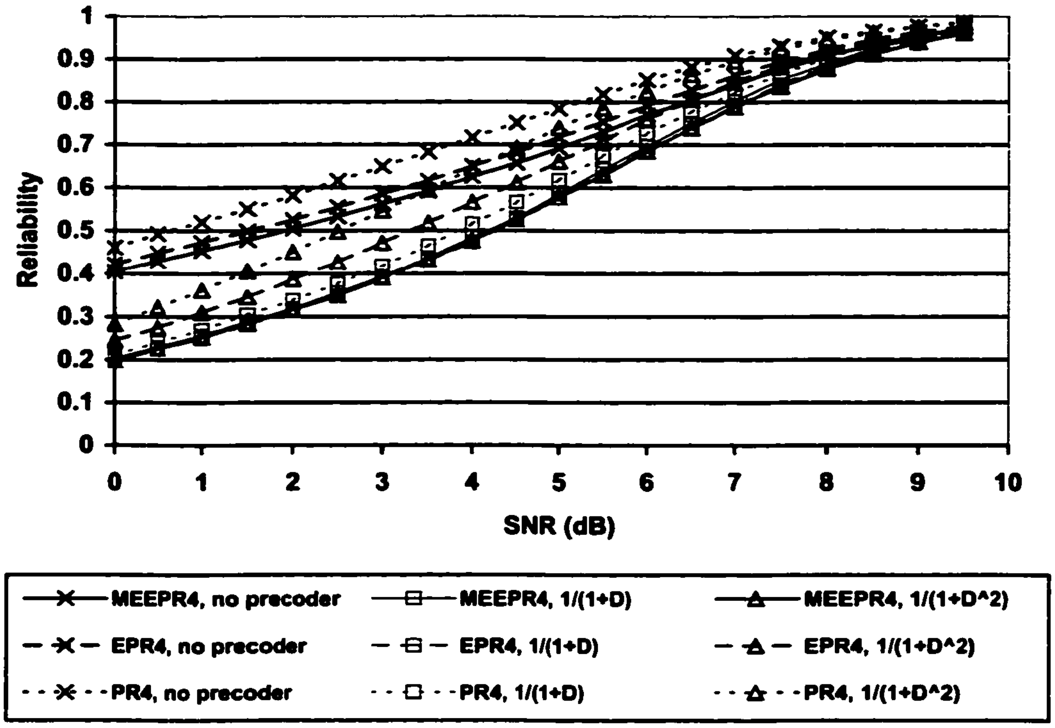


Figure B.1. Mean reliability of ideal PR channels with different precoders.

At the SNR range shown, for a fixed precoding, PR4 yields better reliability than EPR4, and better than ME^{2}PR4. This is consistent with [16] in that long ISI results in more uncertainty. However, the effect of precoding on the mean reliability is not so simple. For PR4 and EPR4, no precoding yields best reliability, $\frac{1}{1 \oplus D^2}$ next and $\frac{1}{1 \oplus D}$ worst. For ME^{2}PR4, however, no precoding yields the best reliability, $\frac{1}{1 \oplus D}$

next and $\frac{1}{1 \oplus D^2}$ worst. It can thus be concluded that the effect of precoder depends on the PR channel.

B.2 MR channels

Now we consider MR channels. The step response is assumed to be Lorentzian-Gaussian as in Section 1.1, and the channel is subject to both AWGN and jitter noise. The channels are precoded by different precoders as in the previous section, and are equalized to EPR4 or ME^{2}PR4. At the output of the channel detector, the noise is correlated. The SNR is defined as in (1.10).

Shown in Figures Figure B.2 and Figure B.3 are the mean reliability and BER for equalized EPR4 and ME^{2}PR4 channels with pure AWGN, with different precoding. The following observations can be made:

1. No precoding yields worst mean reliability throughout all SNRs shown. Recall that on ideal PR channels, no precoding yields best performance, as shown in Figure B.3
2. The mean reliability curves have a crossover. At low SNR, precoder $\frac{1}{1 \oplus D}$ is not as good as $\frac{1}{1 \oplus D^2}$, but the opposite is true at high SNR.
3. BER is consistent with the mean reliability. Large mean reliability corresponds to low BER. BER curves have a crossover at the same SNR as mean reliability curves.

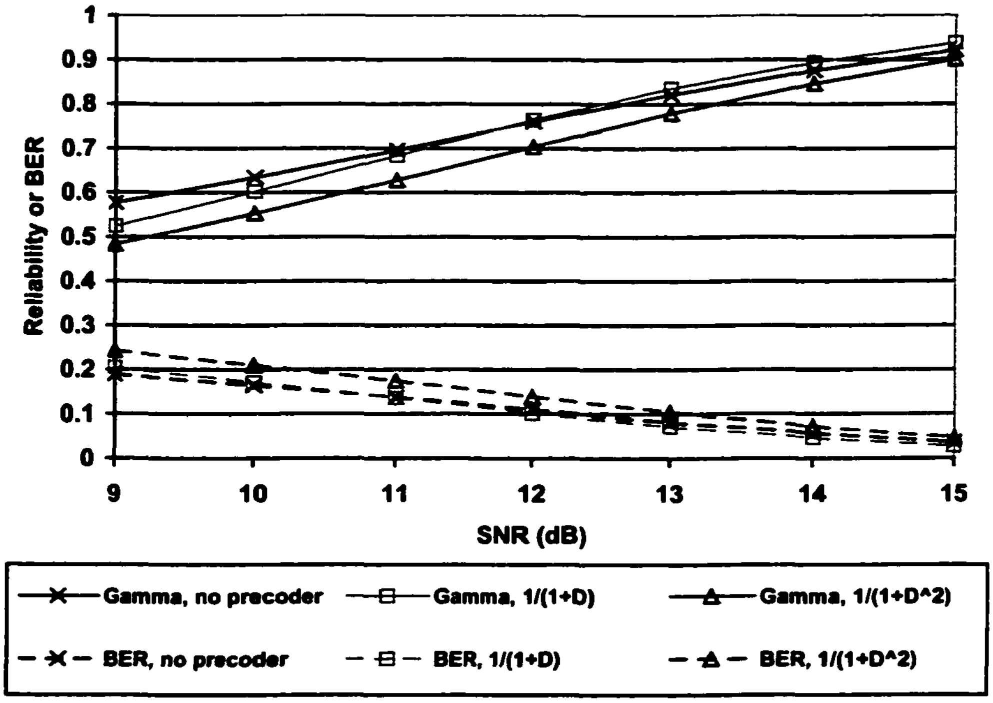


Figure B.2. Mean reliability and BER of equalized EPR4 Lorentzian-Gaussian channel with different precoders, $S_c = 3.0127$, AWGN.

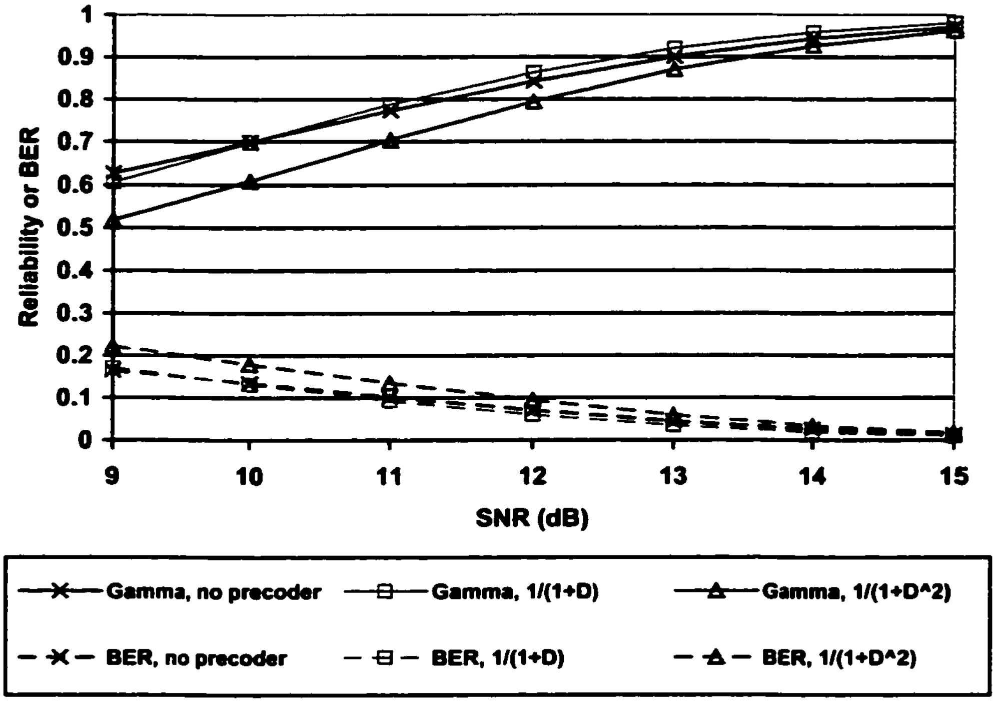


Figure B.3. Mean reliability and BER of equalized ME^{2}PR4 Lorentzian-Gaussian channel with different precoders, $S_C = 3.0127$, AWGN.

For equalized EPR4 and ME^{2}PR4 channels with 90% jitter noise, similar conclusions can be made except that at low SNR $\frac{1}{1 \oplus D}$ yields the worst performance.

In coded MR channels, as the systems shown in Chapters 3, 4 and 5, low raw BER at the channel detector corresponds to good overall performance and the overall system performance is very sensitive to that. The raw BER at the channel detector is even more important in systems in which turbo equalization is not performed. Therefore, appropriate precoding needs to be carefully chosen.