UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

ITERATIVE DECODING FOR MAGNETIC RECORDING CHANNELS

A Dissertation

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

degree of

Doctor of Philosophy

By

YUNXIANG WU
Norman, Oklahoma
2000

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

ITERATIVE DECODING FOR MAGNETIC RECORDING CHANNELS

A Dissertation APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY

Chair

This dissertation is dedicated to my parents, for the memory of my mother

ACKNOWLEDGEMENTS

I would like to express sincere gratitude to my advisor, Professor J. R. Cruz, for his insightful guidance and interesting projects. Without his support, I would not have had any knowledge on magnetic recording channels. I am grateful to my advisory committee members, Professor Art M. Breipohl, Professor Kevin A. Grasse, Professor Joseph P. Havlicek, and Professor Fred N. Lee, for their advice and assistance.

I am appreciative of Kenneth E. Hild, a former student of Professor J. R. Cruz, for his precious help during the project on Performance Evaluation of Write Equalization and Run-Length Limited Codes Design. The work in Chapter 2 was conducted collaboratively with a research group at Seagate Technology, Inc. The individuals are Robert D. Cronch, Aravind Raghunathan, and Dr. Nikki M. Bruner. I would like to thank them for their efforts in the experimental evaluation of the performance of serially concatenated convolutional codes.

Thanks are due to my wife Yajuan for being supportive in the pursuit of my career. Thanks to her for doing much of the housework at home while being successful in her own career. I am also grateful to my brothers and sisters, who always make time from their busy life to take care of my parents and let me study in the United States.

I am thankful to all the students and staff in the Communications Signal Processing Laboratory. I especially enjoy the discussions and collaborations

with Richard M. Todd, Jun Xing, Hongxin Song, Shuqing Liu, and Cheng Zhong.

It is a pleasant memory the three-years spent with People's Publishing House of Telecommunications and Posts, Beijing, China. I appreciate my former colleagues for their encouragement during the pursuit of my degree.

Norman, September 2000

Yunxiang Wu

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

The success of turbo codes indicates that performance close to the Shannon limit may be achieved by iterative decoding. This has in turn stimulated interest in the performance of iterative detection for partial-response channels, which has been an active research area since 1999. In this dissertation, the performance of serially concatenated recording systems is investigated by computer simulations as well as experimentally. The experimental results show that the iterative detection algorithm is not sensitive to channel nonlinearities and the turbo coded partial-response channel is substantially better than partial-response maximum-likelihood channels. The classical iterative decoding algorithm was originally designed for additive white Gaussian noise channels. This dissertation shows that the performance of iterative detection can be significantly improved by considering the noise correlation of the magnetic recording channel. The idea is to iteratively estimate the correlated noise sequence at each iteration. To take advantage of the noise estimate, two prediction techniques were proposed, and the corresponding systems were named noise predictive turbo systems. These noise predictive turbo systems can be generalized to other detector architectures for magnetic recording channels straightforwardly.

Chapter 1

Introduction

1.1 Digital Magnetic Recording

Since the epoch-making invention of the hard disk drive in 1951, it has been the major form for information storage [1]. However, most of the useful information in the world has not been stored. With the blooming of the world-wide-web, accessing the information stored everywhere in the world will become routine for billions of people.

The storage capacity of today's hard disk drive is astonishingly large. In the disk drive industry, the storage capacity is usually measured by areal density. The areal density of the first hard disk was 2 Kb/in^2 [1]. In the 1970s and 1980s, the areal density growth rate was approximately 30% per year. In the 1990s, the success of magnetoresistive heads and partial-response maximum-likelihood (PRML) technology boosted the growth rate to 60% per year. In this new millennium, the hard disk industry is facing a number of fundamental challenges. Since we are in the information age, one of the major challenges is the explosive increase in information. To store the information, higher recording capacity will be continuously needed. The other challenge is the steadily expansion of the market for new magnetic recording products, e.g., storage devices for various mobile internet access equipment, TV set top boxes, and digital cameras. These market forces put pressure on the disk drive industry to develop new products, and at the same time point to a brilliant future.

1.2 Overview of Hard Disk Drive Technology

The major magnetic information storage devices are tape, floppy disk, and hard disk drives. Among these devices, the hard disk drive is especially important because it provides both large storage capacity and random accessibility. A hard disk drive consists of four parts [1], which are, the read/write heads and magnetic disks, data detection electronics and write circuit, mechanical servo and control system, and interface.

The read/write heads are located on a slider. The encoded information is written on the magnetic media through the write head. When a write operation request is issued, the write head moves to a proper position over the disk, and the disk moves in a circular motion. At the same time, the pattern of the write current causes corresponding magnetization transitions on the recording media immediately under the write head, and the information is stored on the disk in the form of magnetization transitions. The concentric circles where information is stored are called tracks. The capacity of a hard disk drive is commonly measured by areal density. The areal density of recording is then the product of the number of tracks per inch (tpi) and the linear density of information along a track measured in bits per inch (bpi). When the information is to be read, a read head is positioned over the track. When the disk moves, the magnetization pattern in the disk causes a corresponding voltage pattern to the read head. The detection circuits then retrieve the recorded information through this voltage pattern.

The detection circuits consist of many very large scale integrated circuit chips. Among these chips, this dissertation is particularly interested in the algorithms for the detector chip. The first generation detector was the peak detector [1]. Peak detection is simple and reliable. It was so successful that it had been the dominant detection method until the end of the 1980s. Since then, the partial response channel quickly became the main detector since an efficient decoding algorithm, i.e., the Viterbi algorithm, can reliably deal with channel interference at high densities. With the discovery of turbo codes, iterative detection has been shown to provide excellent performance [2], [3]. An iterative detector usually consists of several constituent decoders. Each constituent decoder improves its decisions using the information available from other constituent decoder/decoders. The improved decision is then used by other constituent decoder/decoders so that the final decisions are reached in an iterative fashion. It is widely believed that the next generation of recording channel detection algorithms will be some type of iterative soft detection algorithm. Although its performance is excellent, the implementation of iterative detection algorithms is quite difficult. Both the computation and memory requirements for an iterative detector are substantially higher than that for a Viterbi detector.

The spindle speed of today's commercial hard disk drives is as high as 12,000 rotations per minute (RPM), and the flying height is as low as 25 nm [1]. This extremely small flying height is necessary for sensing the recorded magnetization patterns, and has been compared with a Boeing 747 jet plane constantly flying only several meters above the earth [4]. When a data read/write request is issued, the arm where the heads are located must move to the position that is immediately above the correct track. For a commercial hard disk drive, this time is typically only 2 to 30 milliseconds [1]. The functional part for accomplishing this astonishingly fast and accurate movement is the mechanical servo and control system.

Hard disk drives serve as the secondary level memory for computer systems. The interface is a connector through which the disk drive exchanges information with the outside world.

1.3 Channel Model

1.3.1 Continuous time channel model

The magnetic recording channel is inherently nonlinear because of the hysteresis effect of the magnetic media. The actual noise in the read process is, in general, neither additive nor stationary [5]. The noise can be categorized into media noise and electronic noise. The primary noise sources are the magnetic recording media, the read/write heads, and the preamplifier. It is possible to describe the readback signals using an accurate channel model, but this accurate channel usually is too complicated for theoretical analysis and for the performance evaluation of new detection algorithms. To obtain a simpler channel model, it is necessary to use some approximations with the trade off being accuracy.

The channel can be linearized if the current into the head is constrained to take only two possible values, and the amplitude of the current is sufficiently large so that magnetic storage media can be completely magnetized in one of two directions [5]. The read head senses the transitions of the magnetization pattern, and generates the corresponding voltage pattern. For an isolated transition it produces a pulse $g(t)$ or its inverse $-g(t)$, depending on the direction of the transition. One usually refers to $g(t)$ as the isolated transition response or just transition response.

The widely used model for the transition response is the Lorentzian pulse shape given by (1.1), where PW_{50} denotes the width at half amplitude which defines the resolution of the recording process,

$$g(t) = \frac{1}{1+\left(\frac{2t}{PW_{50}}\right)^2}. \tag{1.1}$$

For the Lorentzian channel, recording density is defined as the normalized PW_{50}. If we denote user recording density as D_u, channel recording density as D_c, user bit duration as T_u, channel bit duration as T_c, then user recording density is defined as

$$D_u \equiv \frac{PW_{50}}{T_u}, \tag{1.2}$$

and channel recording density is defined as

$$D_c \equiv \frac{PW_{50}}{T_c}. \tag{1.3}$$

The advantage of using this model is that it describes the shape of the isolated transition response reasonably well, and requires only the single parameter PW_{50}. Fig. 1.1 shows Lorentzian pulses for three different values of normalized PW_{50}. Fig. 1.2 shows a real transition response taken from a spinstand. The waveform $h(t) = g(t) - g(t - T)$ represents two transitions at minimum spacing, and is designated dibit response.

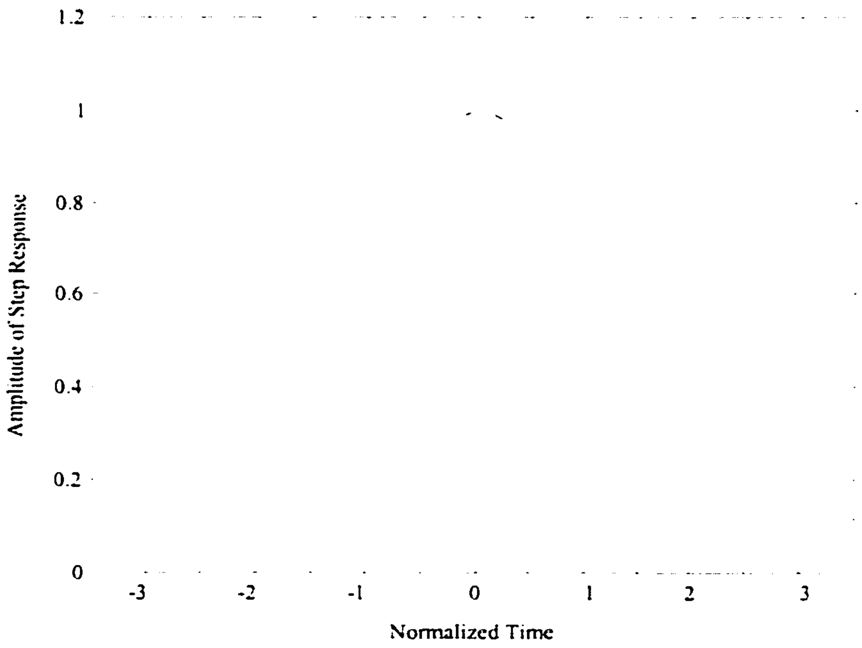


Fig. 1.1. Lorentzian pulses for recording densities 2.0, 2.5, and 3.0.

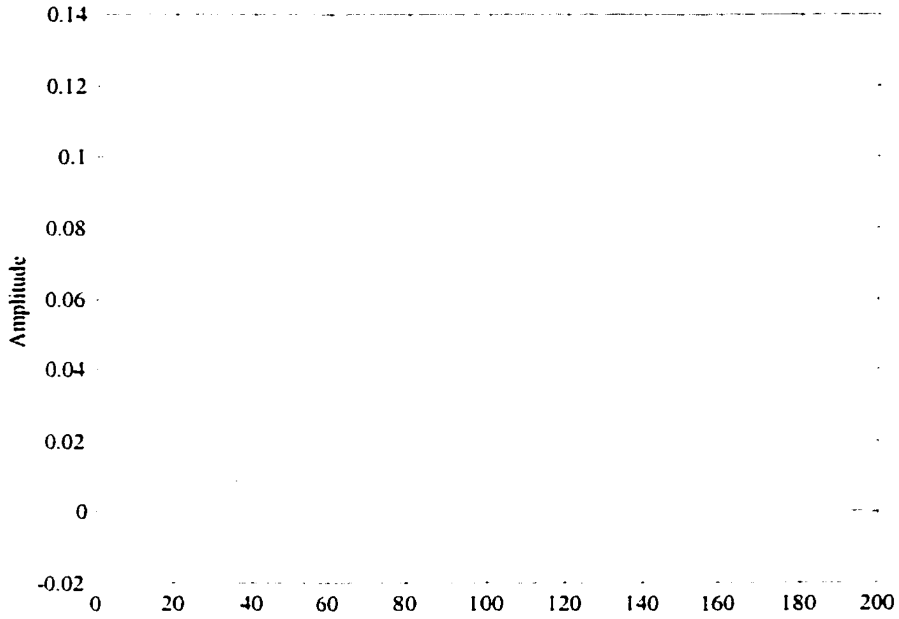


Fig. 1.2. A channel step response captured from a spinstand. Channel density is approximately 2.

In mathematical terms we may describe the data component $r(t)$ of the read head output as

$$r(t) = \sum_{k=-\infty}^{\infty} z_k h(t - kT), \tag{1.4}$$

where z_k is the recorded bit sequence with symbol constellation {-1, +1}. Upon realizing that $h(t) = g(t) - g(t - T)$ we may equivalently express $r(t)$ as [4]

$$r(t) = \sum_{k=-\infty}^{\infty} b_k g(t - kT), \tag{1.5}$$

where $b_k = z_k - z_{k-1}$, $b_k \in \{-2,0,2\}$ is the transition sequence associated with z_k. Therefore, $r(t)$ can equivalently be thought of as a linear filtered version of the data sequence z_k or of the transition sequence b_k.

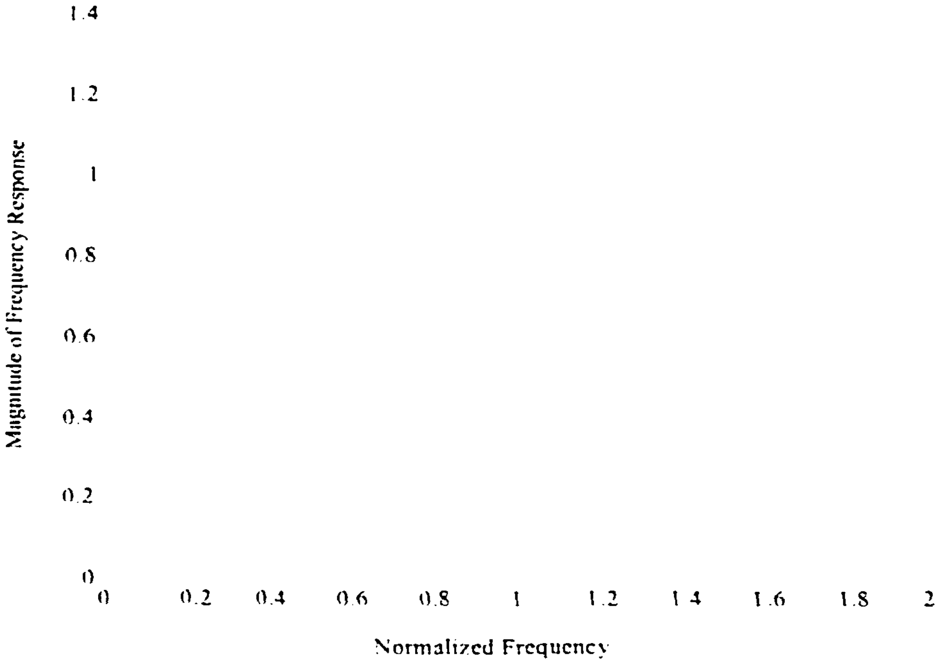


Fig. 1.3. Frequency response of the dibit pulse. Recording density is 2.

The frequency response of the dibit response is

$$H(f) = -j\pi^2 PW_{50} f e^{2\pi(fPW_{50})^2} . \tag{1.6}$$

where $j = \sqrt{-1}$. The frequency response of the dibit is shown in Fig. 1.3 for a density of 2.

If the noise magnitude is not too large, it may be viewed as additive [5]. Furthermore, if the transition response $g(t)$ has negligible excess bandwidth, then a stationary model can be used [4]. The simplest model for channel noise $n(t)$ assumes that the noise is additive white Gaussian (AWGN). With the white noise model, the continuous channel model is shown in Fig. 1.4, and the readback signal takes the form [6]

$$y(t) = r(t) + n(t). \tag{1.7}$$

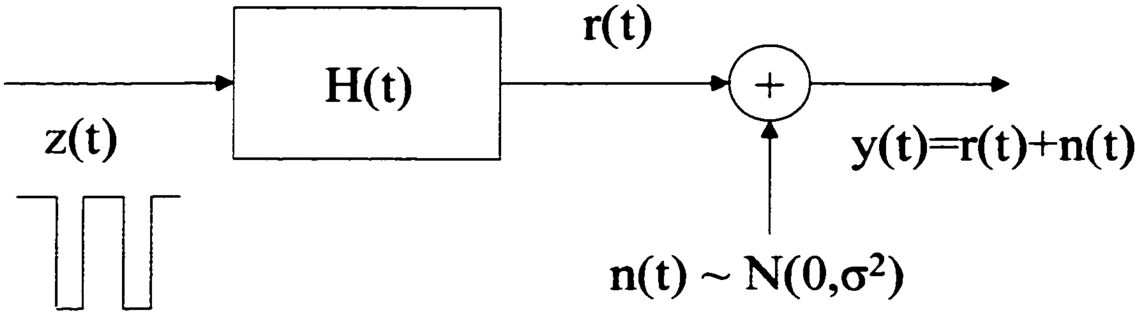


Fig. 1.4. Continuous time channel model.

1.3.2 Equivalent discrete time channel model

When $y(t)$ is received by a matched-filter, the sampled output sequence has the form [7]

$$m(D) = z(D)R_{ff}(D) + n'(D). \tag{1.8}$$

Here $n'(D)$ is zero-mean colored Gaussian noise with autocorrelation function $\sigma^2 R_{ff}(D)$, $R_{ff}(D)$ is the autocorrelation of the Lorentzian pulse, and σ^2 is the spectral density of the noise $n(t)$ in (1.7). Furthermore, Forney [7] showed that $R_{ff}(D)$ can be factorized as

$$R_{ff}(D) = f(D)f(D^{-1}). \tag{1.9}$$

The correlated noise sequence $n'(D)$ can be represented as

$$n'(D) = n(D)f(D^{-1}). \tag{1.10}$$

If we denote $c(D) = m(D)/f(D^{-1})$, we obtain a sequence

$$c(D) = z(D)f(D) + n(D), \tag{1.11}$$

where $n(D)$ is white noise. The cascade of a matched filter $h(-t)$ with a transversal filter characterized by $1/f(D^{-1})$ is called the whitened matched filter, and is shown in Fig. 1.5.

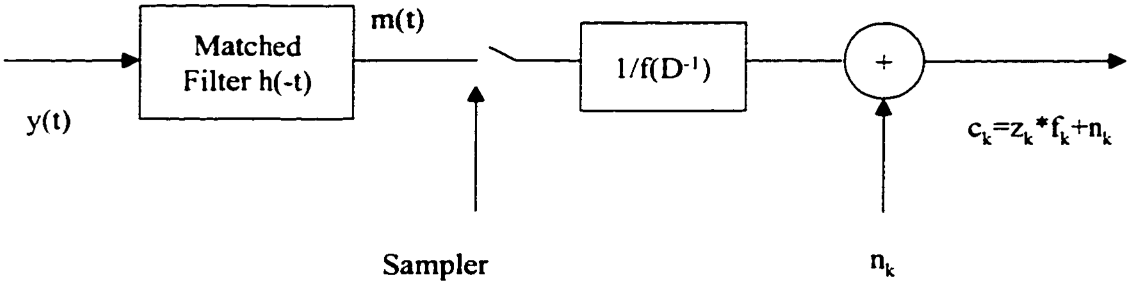


Fig. 1.5. Whitned-matched filter model.

It has also been shown in [7] that the sampled output sequence in (1.8) forms a set of sufficient statistics for the estimation of the input sequence $z(D)$. For infinite and reasonable finite delays, this channel model is optimal for maximum-likelihood estimation of the entire transmitted sequence.

1.4 Outline of the Dissertation

We have briefly introduced the magnetic recording technology and magnetic recording channel models. While PRML magnetic recording systems are the state of the art technology for hard disk drives, the research on more advanced detection schemes never stops. The use of turbo codes in magnetic recording systems has attracted a great deal of interest since 1998. Compared with PRML systems, iterative decoding promises a significant performance improvement [2], [3]. In Chapter 2, serially concatenated convolutional turbo recording systems are discussed, and the performance of the serially concatenated convolutional turbo codes is studied for various partial response channels by computer simulation. Furthermore, the performance of serially concatenated convolutional turbo codes is experimentally evaluated for an extended partial-response class-4 (EPR4) channel in a real hard disk drive. The results show that the performance improvement obtained in the experiment is consistent with simulations. This was the first experimental work published in the literature.

Partial response (PR) channels have been used in many applications, and in general, they require a filter to shape the received signal to a given PR target. At high density, significant equalization is required for a partial response class-4 (PR4), EPR4 and even modified E^2PR4 (ME^2PR4) channels, causing strong noise correlation at the detector input, which degrades performance. To improve the performance of the Viterbi detector, a noise prediction scheme was proposed in [8]. The prediction is based on the tentative decisions obtained from the surviving path corresponding to each state. This scheme substantially improves the performance of the Viterbi detector. In Chapter 3, noise predictive turbo systems are studied. Computer simulations show that significant signal-to-noise ratio gain can be obtained through iterative prediction. The proposed prediction scheme using conditional expected decisions is simple and very suitable to be combined with iterative detectors. As an example, simulation results for a serially concatenated convolution turbo coded PR4 system are provided.

Pearl's belief propagation algorithm is an efficient algorithm for solving inference problems. It has been widely used in medical diagnosis systems, machine learning, and various decision systems. Recently, it was found that the iterative soft decoding algorithm for parallel and serially concatenated turbo systems is an instance of the belief propagation algorithm applied on the corresponding loopy Bayesian networks [9], [10]. This result provides an alternative way to view iterative decoding algorithms, and has motivated much interest in the study of the connections between the belief propagation algorithm and iterative decoding.

In magnetic recording, error-correcting codes are required to ensure a low bit error rate. The concatenation of the error-correcting code and the partial-response equalized channel can be viewed as two serially concatenated convolutional codes. In Chapter 4, iterative detection for magnetic recording systems is explained from the viewpoint of belief propagation. The suitable Bayesian network for the recording system considered is given for both white Gaussian noise channels and correlated/signal dependent noise channels. The detection for correlated noise recording channels is much more complicated than for white Gaussian noise channels. Through probability propagation on the corresponding Bayesian networks, the difference of the two decoding algorithms is made explicit. This gives us a better understanding of iterative detection for magnetic recording channels. This chapter also shows that the noise predictive turbo decoding algorithm is a probability propagation process.

Chapter 5 is a summary of the results in this dissertation, as well as future research suggestions.

Chapter 2

Performance of Serially Concatenated Convolutional Turbo Codes for Magnetic Recording

2.1 Introduction

The use of turbo codes in magnetic recording systems has attracted a great deal of interest since 1998. Using parallel concatenated convolutional codes in a PR4 magnetic recording channel led to substantially better performance than the uncoded PRML system [2]. A relatively simpler serially concatenated scheme was proposed in [11]. In this scheme, the combination of the precoder and channel is viewed as one of the constituent convolutional codes. Therefore, the detector is only composed of two maximum *a posteriori* probability decoders matched to the precoder/channel and the outer convolutional code, respectively. Although much simpler, its performance is comparable to the parallel concatenated system.

Extensive research done by others and us has confirmed the results presented in [2] and [11]. However, most of the research activities have focused on Lorentzian channel simulations, which assumed ideal linear superposition channel models, perfect timing recovery and gain control. Since iterative decoding operates at a lower signal-to-noise ratio (SNR), the performance of the timing recovery and gain control might be seriously degraded due to inaccurate feedback. In order to determine the impact of channel nonlinearities and other impairments, we have compared simulation and real drive experimental results for serially concatenated convolutional turbo codes.

2.2 Lorentzian Channel Simulation

The diagram of the Lorentzian channel simulation is shown in Fig. 2.1. The details of the various components are described in this section.

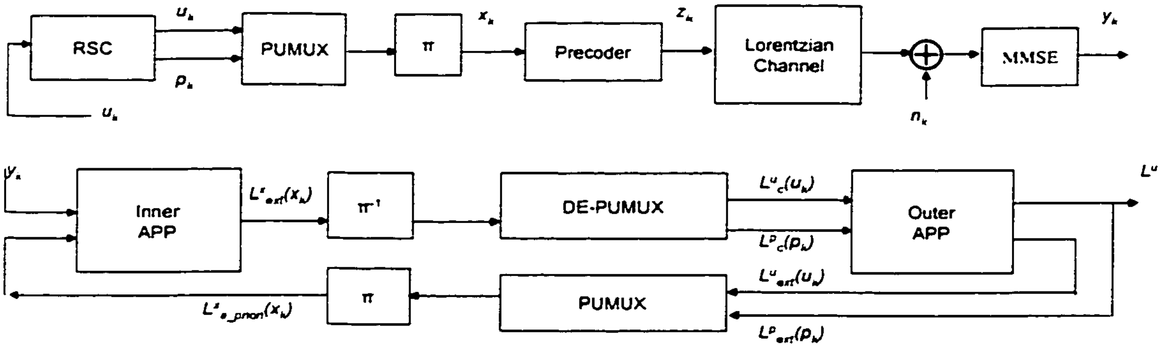


Fig. 2.1. Diagram of a serially concatenated system for magnetic recording.

2.2.1 Encoder, puncturer and precoder

User data u_k is encoded by a rate ½, $(31,33)_{oct}$ recursive systematic convolutional (RSC) encoder. To achieve a code rate of $k/(k+1)$, one parity bit is transmitted following every k information bits. The puncturer and multiplexer (PUMUX) block in Fig. 2.1 performs this operation. Following the random interleaver, the interleaved data is precoded, and then fed into the channel. Both the convolutional code and the precoder were terminated to the zero state at the end of every block. The length of the interleaver is 4096 bits. The partial response targets considered are PR4, EPR4, $\mathrm{E^2PR4}$, and a $\mathrm{ME^2PR4}$ target $(1-D^2)(2+2D+D^2)$ [12]. The precoders are of the form $1/(1+D^2)$, $1/(1+D+D^3)$ and $1/(1+D+D^2+D^3+D^4)$. We used these precoders because they are of the same order as the target partial response channels, and they do not increase the complexity of the decoder.

2.2.2 Channel model and signal-to-noise ratio

The equalized channel model comprises a Lorentzian channel, an additive white Gaussian noise source $n(t)$ and a minimum-mean-squared-error (MMSE) equalizer. The readback signal at the input of the MMSE equalizer can be written as

$$s(t)=\sum_k z_k[g(t-kT)-g(t-(k+1)T)]+n(t), \tag{2.1}$$

where $z_k \in \{\pm 1\}$ is the precoded data that was written on the magnetic media; T is the sampling interval; $g(t)$ is the step response of the Lorentzian channel, which was defined by (1.1). The MMSE equalizer was designed so that the Lorentzian dipulse $g(t-kT)-g(t-(k+1)T)$ can be equalized to a desired partial response target when sampled at the sampling rate of $1/T$.

We defined SNR as

$$SNR = 10\log\left(\frac{1}{\sigma^2}\right), \tag{2.2}$$

where σ is the standard deviation of the additive white Gaussian noise. Because the maximum amplitude of (1.1) has been normalized to 1, the SNR is controlled only by the amount of added noise.

2.2.3 Iterative decoding

The iterative decoder includes two *a posteriori* probability (APP) decoders. The inner APP decoder is matched to the combination of precoder/channel, whereas the outer APP is matched to the convolutional encoder. The inner APP processes not only the channel values but also *a priori* information about the symbols to be detected. The outer APP, on the other hand, processes only the de-interleaved and de-multiplexed extrinsic information of the inner APP decoder. The log-likelihood ratio (LLR) of *a posteriori* probabilities of the precoder input x_k can be calculated according to

$$L(x_k \mid \mathbf{y}) = \ln\frac{P(x_k = +1 \mid \mathbf{y})}{P(x_k = -1 \mid \mathbf{y})} = \ln\left(\frac{\sum_{(s',s),x_k=+1}\alpha_{k-1}(s')\cdot\gamma_k(s',s)\cdot\beta_k(s)}{\sum_{(s',s),x_k=-1}\alpha_{k-1}(s')\cdot\gamma_k(s',s)\cdot\beta_k(s)}\right), \tag{2.3}$$

where s' and s denote the states of the combined trellis of the precoder and channel at time instant *k-1* and *k*, respectively. Here α_k and β_k can be calculated recursively as

$$\alpha_k(s) = \sum_{s'} \gamma_k(s',s) \cdot \alpha_{k-1}(s') \tag{2.4}$$

$$\beta_{k-1}(s') = \sum_{s} \gamma_k(s',s) \cdot \beta_k(s) \,. \tag{2.5}$$

Since we terminated each block to state zero for the precoder, α and β are initialized as

$$\begin{aligned} &\alpha_0(0) = 0 \text{ and } \alpha_0(s) = -\infty, \text{ where } s \neq 0 \\ &\beta_N(0) = 0 \text{ and } \beta_N(s) = -\infty, \text{ where } s \neq 0, \end{aligned} \tag{2.6}$$

where N is the block length of the inner APP decoder, and $\gamma_k(s',s)$ is the branch transition probability. In the logarithm domain

$$\gamma_k(s',s) = -\frac{1}{2\sigma^2}\left|y_k - \bar{y}_k(s',s)\right|^2 + \frac{1}{2}x_k \cdot L_{a\_priori}(x_k), \tag{2.7}$$

where $\bar{y}_k(s',s)$ is the ideal channel output for a transition from s' to s, and σ denotes the noise standard deviation at the input of the inner APP decoder. $L_{a\_priori}(x_k)$ is the *a priori* information about x_k obtained from the extrinsic information of the outer APP. The extrinsic information passed to the outer APP decoder is

$$L_{ext}^{x}(x_k) = L(x_k \mid \mathbf{y}) - L_{a\_priori}(x_k). \tag{2.8}$$

The decoding process for the outer APP decoder is done similarly. Since it matches the binary systematic convolutional code, the algorithm in [13] can be employed. The initialization is the same as the inner decoder.

Iterative decoding starts with the inner decoder. During the first iteration, the *a priori* information for the inner decoder is set to zero. After the extrinsic information of the inner decoder is calculated using (2.8), it is de-interleaved and de-multiplexed and then input to the outer decoder as the soft estimate of the channel value. Using this soft estimate, the outer decoder calculates its own extrinsic information for both the user bits and their parity bits. The extrinsic information of the outer decoder is multiplexed and interleaved, and fed back to the inner APP decoder. After several iterations, the process is stopped when the hard decisions for the user bits cannot be improved.

2.3 Drive Data Experiment

In the Lorentzian channel simulation, we assumed that the channel is a linear superposition of isolated Lorentzian pulses. Furthermore, we also assumed that the readback signal is ideally down sampled. This is an approximation to the real disk drive channel. In our experiment, we encoded, permuted and precoded user data as in Fig. 2.1. The precoded data was written to a disk drive and then oversampled 3.526 times. To cope with channel variations and the need to downsample the readback signal, a more complicated scheme was used to process the readback signal before decoding. The front end of the read channel is shown in Fig. 2.2.

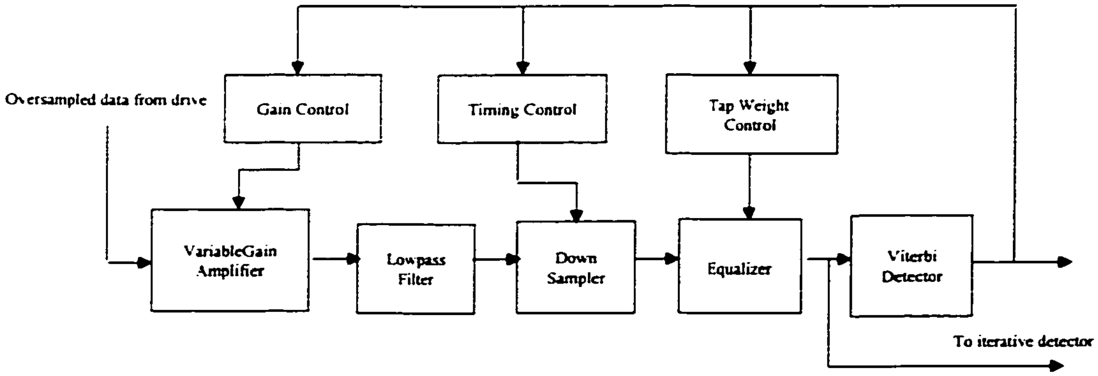


Fig. 2.2. Front end of the read channel for the drive data experiment.

2.3.1 Gain control and timing recovery

Generally, the readback data is not zero mean. This mean is removed and the zero mean signal is then passed to the variable gain amplifier which compensates for the amplitude variations caused by the channel and filters. For fast initial adjustment of the gain and timing phase, a 2T preamble is written prior to the user data. A zero phase restart (ZPR) technique was employed for quick timing acquisition. In the tracking mode, the timing control block adjusts both the sampling rate and the sampling phase of the down sampler.

2.3.2 Adaptive equalization

Adaptive equalization using a least mean square algorithm was used in the real data experiment. We trained the equalizer by setting all the initial taps to zero, and used enough data to guarantee that all coefficients converged. Automatic gain control, timing recovery and adaptive equalization all worked in a decision-directed mode. The error signals that were used for these adaptive loops were obtained from the difference of the equalized samples and the signal extracted from the Viterbi detector. This Viterbi detector was used specifically to obtain better error signals. While we cannot guarantee that we found the optimum loop gains for the three feedback loops, the same equalized signal was simultaneously used for iterative detection and Viterbi detection, which provided a fair basis for comparison.

2.3.3 Lowpass filter

A third order lowpass filter was used in the drive data experiment. The cutoff frequency was optimized based on the minimum-mean-squared-error criterion.

2.3.4 Signal-to-noise ratio definition

The definition of SNR in (2.2) is suitable for the linear superposition channel model, but not for a real disk drive. When channel nonlinearities are considered, calculating the actual amount of channel noise becomes very difficult. To circumvent this problem, we defined SNR after equalization as

$$SNR = 10\log\frac{1}{2R\sigma'^2} \qquad (2.9)$$

where σ' is the standard deviation of the noise sequence which was obtained by subtracting the ideal channel output from the equalized channel output and R is the code rate.

2.4 Results

2.4.1 Lorentzian channel results

We have simulated rate 16/17 and 32/33 systems on precoded PR4, EPR4, E^{2}PR4 and ME^{2}PR4 equalized channels, as shown in Fig. 2.1. The user density was 2.7, and the maximum number of iterations was set to ten. As a baseline, the uncoded PR4 system was compared with the serially concatenated systems and the results are shown in Figs. 2.3 and 2.4. The bit-error-rate performance of the ME^{2}PR4 channel is 4.7 dB better than the uncoded PR4 system at 10^{-5}.

In order to reduce data processing time in the drive data experiment, we investigated two stopping criteria for the Lorentzian channel and used one of them in our experiments. The method of observed hard decisions consists of observing the hard decisions made by the channel APP in each iteration. If the hard decisions do not change in two consecutive iterations, the iteration process is stopped.

In [14], the cross-entropy stopping method was simulated for a mobile communications system. This criterion calculates the change of cross-entropy for every iteration using the log-likelihood ratios generated in the decoding process.

We have applied this method to the Lorentzian channel and stopped the iteration if

$$\sum_{k}\left(\frac{\left|L_{ext}^{x}(i)-L_{ext}^{x}(i-1)\right|^{2}}{\exp(L_{ext}^{x}(i-1)+L_{a\_priori}^{x}(i))}\right)< \textit{threshold} \qquad (2.10)$$

where i represents current iteration, and $L_{a\_priori}^{x}(i)$ is the multiplexed and permuted extrinsic information of the outer APP decoder. A suitable threshold can be obtained by observing the variation of the left hand of (2.10). In Fig. 2.5, we have plotted the average required number of iterations versus SNR for an E^2PR4 system with a code rate of 32/33. It can be seen that the cross-entropy method is slightly better than the observed hard decision method. Both stopping criteria give a much smaller average number of iterations at high SNR. No extra decoding errors were introduced by applying these two methods in our simulations.

2.4.2 Drive data experiment results

Drive data were collected at user density 2.25 and code rate 16/17. We simulated ideal uncoded EPR4 channels and compared them with drive EPR4 channels. In Fig. 2.6, we use the term ideal to refer to a perfectly equalized channel with additive white Gaussian noise and the term drive for the real hard disk drive. We note that the performance of the drive channels (both uncoded and turbo coded EPR4) are around 0.6 dB worse than that of ideal channels. At bit-error-rate of

10^{-5}, the turbo-coded drive EPR4 system has a 4-dB SNR improvement over the uncoded drive EPR4 system. In the drive data experiments, the maximum number of iterations was initially set at ten and later reduced to six without any discernible impact on error, while the observed hard decision stopping criterion was applied.

The distribution of bit errors per sector was also studied on the entire data set and the results are shown in Fig. 2.7. It is shown that if a block cannot be correctly detected, the turbo coded EPR4 system tends to have more decoding errors in a sector than the uncoded system. It should be noted that these results were obtained at an SNR of 3.9 dB, which is much lower than the normal operating SNR.

At the same SNR, the dominant error events were also studied and the results are shown in Fig. 2.8. An error event is defined using a reset length of eight bits. The vertical axis is the average error events per sector, and the horizontal axis is the length of the error event. It can be seen that for uncoded EPR4, error events shorter than five bits are most frequent. The dominant error event for turbo coded EPR4 is the single bit error event. This is due to the presence of interleavers.

2.5 Conclusions

We have evaluated the performance of serially concatenated convolutional codes for high order partial response systems using a Lorentzian channel model and a real hard disk drive. At channel density 2.25 and bit error rate 10^{-5}, a coding gain of 4 dB was obtained for the disk drive experiment, which is consistent with that of the simulation results for a Lorentzian channel. We also found that serially concatenated convolutional turbo systems tend to exhibit larger number of errors per sector and that the dominant error event is a single-bit error.

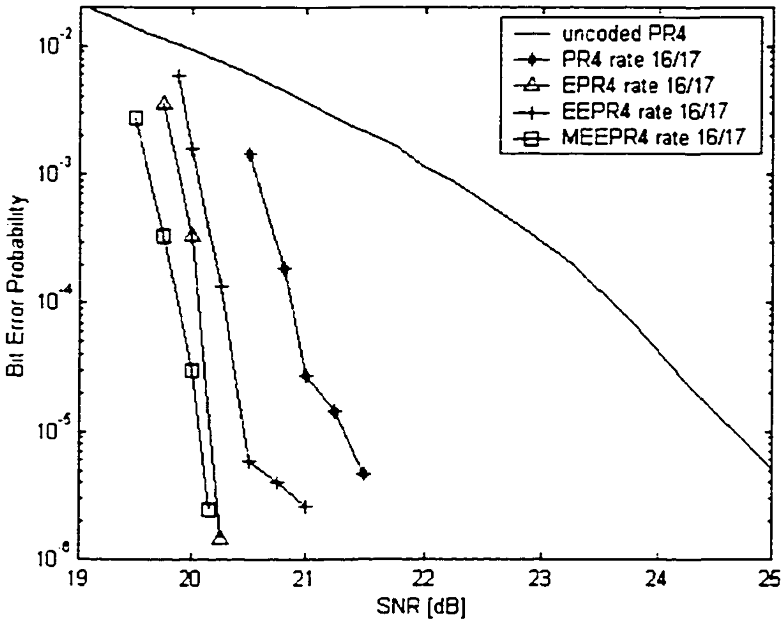


Fig. 2.3. Performance comparison of partial response channels, code rate 16/17.

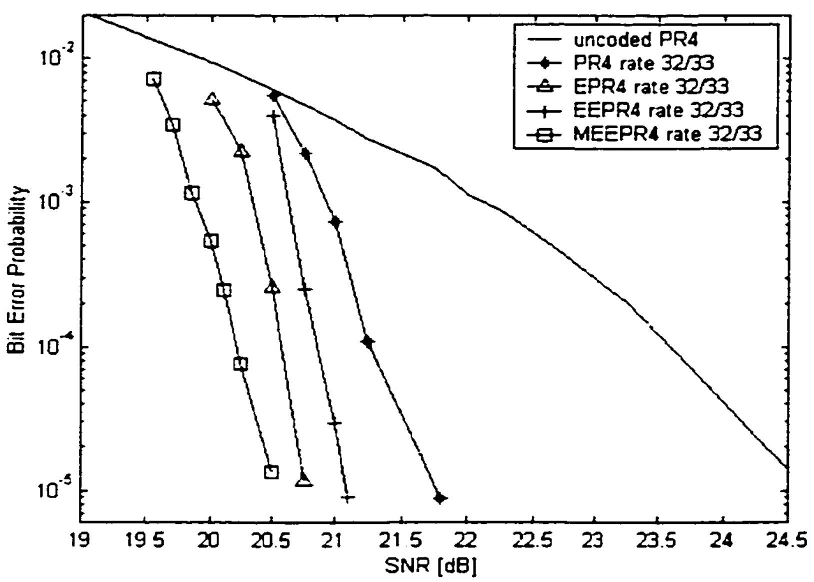


Fig. 2.4. Performance comparison of partial response channels, code rate 32/33.

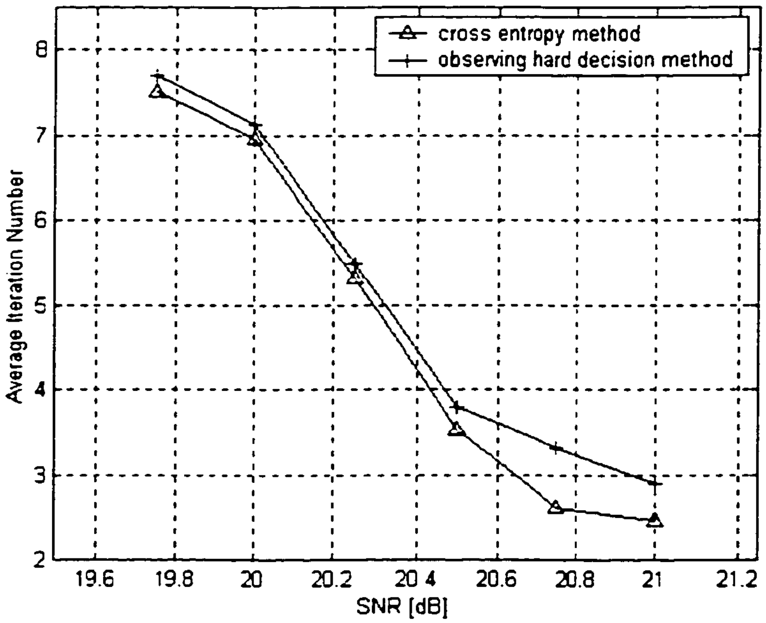


Fig. 2.5. Average number of iterations versus SNR for EEPR4, rate 32/33 system.

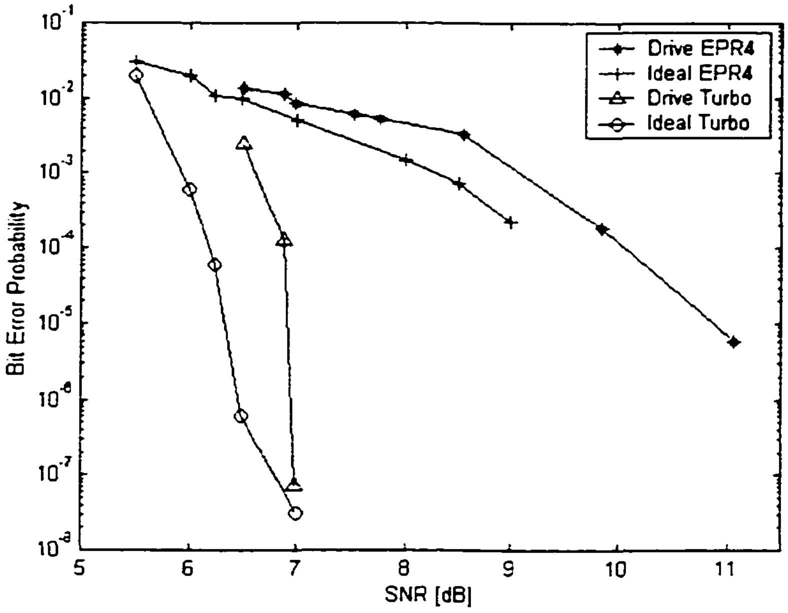
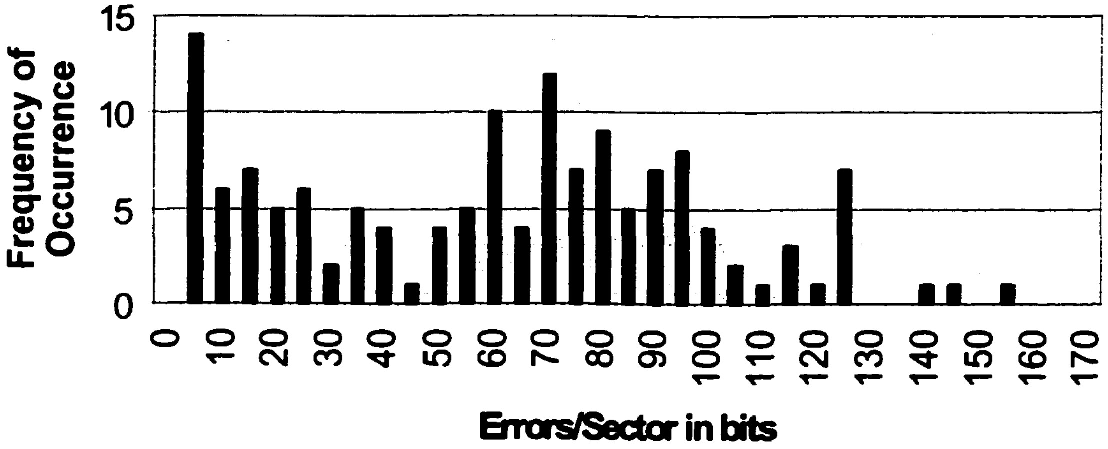
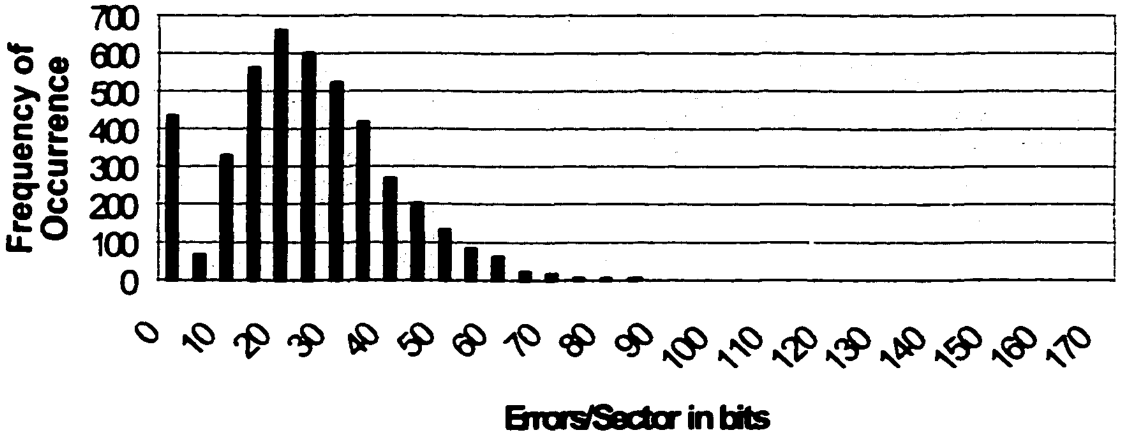


Fig. 2.6. Hard disk drive experiment results for EPR4 channels, code rate 16/17.



(a)



(b)

Fig. 2.7. Distribution of bit errors per sector at $\sigma'=0.6$: (a) turbo coded EPR4, and (b) uncoded EPR4 system.

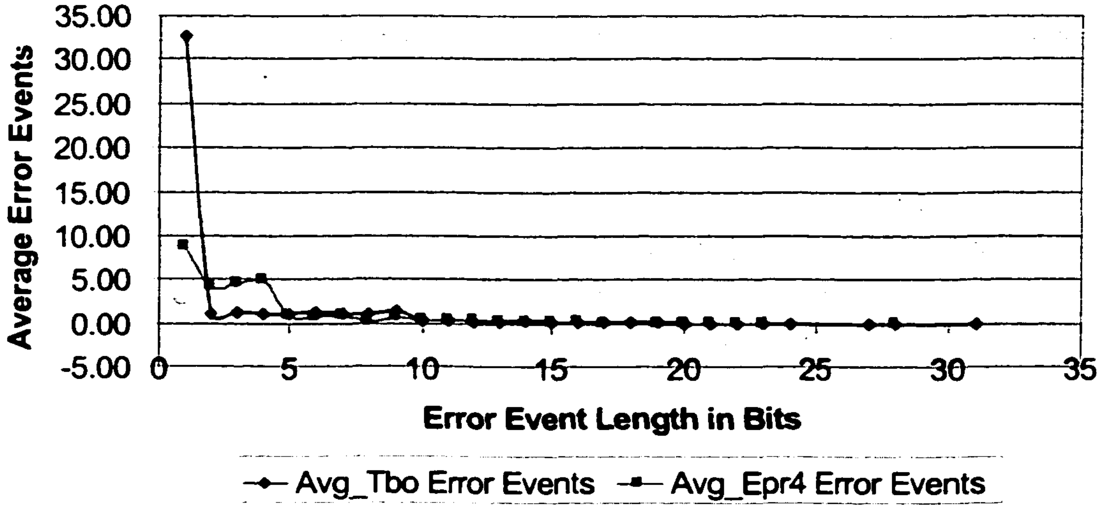


Fig. 2.8. Error events distribution for uncoded EPR4, and turbo coded EPR4, $\sigma'=0.6$.

Chapter 3

Noise Predictive Turbo Systems

3.1 Introduction

Partial response channels have been used in many applications, and in general, a filter to shape the received signal to a given PR target is required. The noise component at the detector input can be modeled as a white noise process filtered by the partial response equalizer. In this chapter, we discuss the application of prediction techniques to serially concatenated convolutional turbo systems of partial response class-4 magnetic recording channels. The idea can be extended to higher order PR channels in a straightforward manner.

Partial-response maximum-likelihood channels have been the dominant detector technology for magnetic recording since 1990. At high density, significant equalization is required for a PR4 channel, causing strong noise correlation at the detector input, which degrades performance. To improve the performance of the Viterbi detector, a noise prediction scheme was proposed in [8]. The prediction is based on the tentative decisions obtained from the surviving path corresponding to each state. This scheme substantially improves the performance of the Viterbi detector.

With the advent of turbo codes, iterative soft detection is expected to be used in high performance receivers for various applications. Turbo equalization for partial response channels involves at least two decoders. One decoder matched to the channel and the others matched to the codes concatenated with the channel. For convenience, we refer to the channel decoder as the turbo equalizer and to the other decoders as the outer decoders. The BCJR algorithm [13] is usually used for the turbo equalizer because it provides a soft output to the various probabilistic decoding algorithms, which can be applied to the codes concatenated with the channel. This algorithm requires that a whole channel output block be received before starting the decoding process. The channel output is stored and used for every iteration. For a given iteration, the *a posteriori* probabilities of the channel symbols generated in the previous iteration or obtained from the outer decoders are useful in estimating the noise sequence at the channel output. Signal processing techniques, such as linear prediction, filtering and smoothing, can be used to reduce the noise.

The decoding techniques in this chapter can be viewed as an extension of Chapter 2. To compare the noise predictive turbo systems with the systems described in Chapter 2, we used the same system and the same SNR definition.

3.2 Noise Predictive Turbo Systems

3.2.1 Linear prediction

Linear prediction has been used to de-correlate the noise in partial-response maximum-likelihood magnetic recording channels [8]. Because the Lorentzian channel is assumed time-invariant, the predictor coefficients can be calculated offline and used for all the input signals from the channel.

Let us denote the noise component in a received signal block for the turbo equalizer as $\mathbf{n} = [n_1, n_2, n_3, \cdots, n_N]$ and the P-tap noise predictor as $\mathbf{a} = [a_1, a_2, a_3, \cdots, a_P]$; the mean squared prediction error function can be written as [15]

$$E(\bar{e}_k^2) = E\left((n_k - \sum_{m=1}^{P} a_m n_{k-m})^2 \right). \tag{3.1}$$

By minimizing this error function we find the forward predictor coefficients

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_P \end{pmatrix} = \begin{pmatrix} r_{nn}(0) & r_{nn}(1) & r_{nn}(2) & \dots & r_{nn}(P-1) \\ r_{nn}(1) & r_{nn}(0) & r_{nn}(1) & \cdots & r_{nn}(P-2) \\ r_{nn}(2) & r_{nn}(1) & r_{nn}(0) & \cdots & r_{nn}(P-3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{nn}(P-1) & r_{nn}(P-2) & r_{nn}(P-3) & \cdots & r_{nn}(0) \end{pmatrix}^{-1} \begin{pmatrix} r_{nn}(1) \\ r_{nn}(2) \\ r_{nn}(3) \\ \vdots \\ r_{nn}(P) \end{pmatrix}, \tag{3.2}$$

where $r_{nn}(k)$ are the elements of the noise correlation vector. Similarly, we find that the backward predictor is the time inverse of the forward predictor. If we denote the backward predictor as $\mathbf{a}'$, we have

$$\begin{aligned} \mathbf{a}' &= [a'_1, a'_2, \cdots, a'_P] \\ &= [a_P, a_{P-1}, \cdots, a_1] \ . \end{aligned} \tag{3.3}$$

3.2.2. Noise predictive turbo systems with soft feedback (NPTS/SF)

For a white noise channel, the performance of the concatenated decoder is near optimal, but its performance can be seriously degraded by the noise correlation of a partial response equalized channel. The reason is that the turbo equalizer does not take into account the noise correlation. A straightforward way to improve the performance is to de-correlate the noise. To predict the noise in the channel output y_k, knowledge of the ideal channel outputs $\bar{y}_{k-m}$ or $\bar{y}_{k+m}$, $m=1, 2, \ldots, P$, is needed. Perfect knowledge of these values is impossible, however, in [8], they are obtained from the surviving path for each state and called tentative decisions. Since the Viterbi detector only keeps one path for each state, correct tentative decisions cannot be guaranteed. Wrong tentative decisions lead to a wrong prediction that will enhance the noise in y_k instead of reducing it. The accuracy of the prediction is determined by the accuracy of the estimates of the tentative decisions.

In turbo equalization, the log-likelihood ratio of the *a posteriori* probabilities for the channel symbols, available from the outer decoder/decoders or from the turbo equalizer, enables us to find expected values of the tentative decisions that are not available to the Viterbi detector. In magnetic recording applications, the ideal channel output $\bar{y}_k$ is a discrete random process. In the sense of minimum-mean-squared-error, finding the best estimate of $\bar{y}_k$, denoted as $\tilde{y}_k$, is equivalent to minimizing

$$Var(e_k^{\,2}) = E\left((\bar{y}_k - \tilde{y}_k)^2\right). \tag{3.4}$$

It is well known that (3.4) is minimized when $\tilde{y}_k$ is the conditional expected value of $\bar{y}_k$. Since $\tilde{y}_k$ is a better estimate than the hard tentative decisions, a more accurate prediction can be expected.

Instead of finding a tentative hard decision as in [8], using $\tilde{y}_k$ is equivalent to use all possible values of $\bar{y}_k$ and scale each value by its *a posteriori* probability. If $\bar{y}_k \in A = \{A_0, A_1, \ldots, A_{M-1}\}$, $\tilde{y}_k$ can be represented as

$$\begin{aligned} \tilde{y}_k &= E(\bar{y}_k \mid \mathbf{y}) \\ &= \sum_{m=0}^{M-1} A_m \Pr(\bar{y}_k = A_m \mid \mathbf{y}). \end{aligned} \tag{3.5}$$

Since the *a posteriori* probabilities for the channel symbols are available for an entire block we can find both the past and future tentative decisions so that the noise prediction can be performed from either side. It is even possible to improve the performance by using forward-backward linear prediction. To make the *a posteriori* probabilities of the turbo equalizer available for the next iteration, they need to be saved. To overcome this problem, we can let the outer decoder pass not only extrinsic information but also its *a posteriori* information to the turbo equalizer. After multiplexing and interleaving, it is used to calculate $\tilde{y}_k$. Therefore, the memory required for the implementation of NPTS/SF is small. We will describe two approaches for calculating $\tilde{y}_k$: one approach uses the APP of the turbo equalizer and the other one uses the APP of the outer decoder.

3.2.3 Noise predictive turbo systems with hard feedback (NPTS/HF)

Although "surviving paths" do not exist in a maximum *a posteriori* probability (MAP) decoder, the feedback in this scheme can be obtained similarly to the NPML technique [8]. The similarity to the NPML method stems from the fact that the feedback is state related. In NPTS/HF, the noise prediction for the branches emitting from state s_i at time instant k, uses the tentative decision at k-1

obtained by taking the ideal channel output associated with the more probable branch that comes into this state. By repeating this process, we can obtain the tentative decisions at time instants *k-P*, *k-P*+1,..., *k*-1. For all the other states at time *k*, tentative decisions are calculated in the same fashion. To explain this process, an example for a PR4 trellis is given.

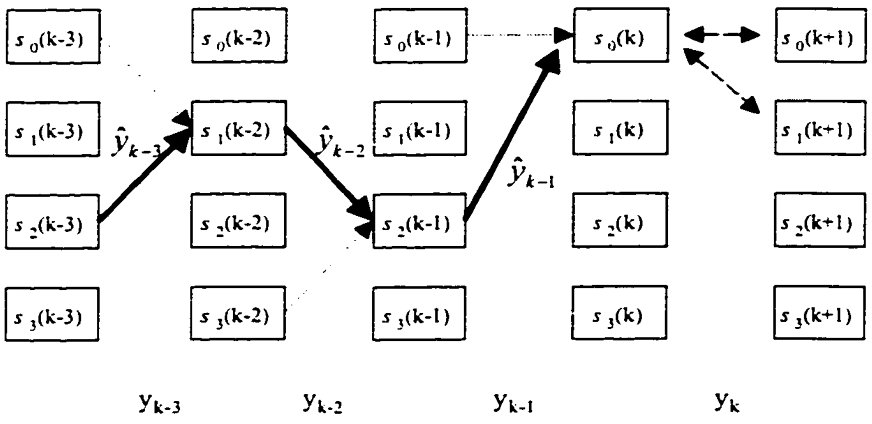


Fig. 3.1. Illustration of finding tentative decisions.

In this example, the bold arrows represent the selected branches for state $s_0(k)$. For time *k-1*, we compare the two joint probabilities, $p(s_0(k-1), s_0(k), \mathbf{y})$ and $p(s_2(k-1), s_0(k), \mathbf{y})$ [13], of the previous iteration and make the decision using

$$B_{k-1} = \frac{p(s_0(k-1), s_0(k), \mathbf{y})}{p(s_2(k-1), s_0(k), \mathbf{y})}. \tag{3.6}$$

If $B_{k-1} \geq 1$, the branch $B_{k-1}(s_0(k-1), s_0(k))$ is selected, otherwise the branch $B_{k-1}(s_2(k-1), s_0(k))$ is selected.

Assume the latter was selected as shown ir Fig. 3.1. At time *k-2*, we compare the two joint probabilities for the branches going into state $s_2(k-1)$. Similarly, the decision can be made using

$$B_{k-2} = \frac{p(s_1(k-2), s_2(k-1), \mathbf{y})}{p(s_3(k-2), s_2(k-1), \mathbf{y})}. \tag{3.7}$$

Obviously, *P* comparisons and selections are made for a *P*-tap predictor for each state.

Assume the selected path for a 3-tap predictor labeled by the output $\hat{y}_{k-3}, \hat{y}_{k-2}$, and $\hat{y}_{k-1}$. The noise for state $s_0(k)$ is estimated as

$$\hat{n}_k = \sum_{i=1}^{P} a_i (y_{k-i} - \hat{y}_{k-i}), \qquad P = 3. \tag{3.8}$$

The noise estimate $\hat{n}_k$ will be used to calculate the transition probabilities $\gamma(s_0(k), s_0(k+1))$ and $\gamma(s_0(k), s_1(k+1))$ in the forward and backward recursions respectively in (2.7)

$$\gamma_k(s', s) = -\frac{1}{2\sigma^2}\left|(y_k - \hat{n}_k) - \bar{y}_k(s', s)\right|^2 + \frac{1}{2} x_k \cdot L_{a\_priori}(x_k). \tag{3.9}$$

As we have seen this method is very similar to NPML. The difference is that we have to predict the noise in the previous iteration and save it for the consecutive iteration. From an implementation point of view, it is not necessary to go back *P* steps to search for the tentative decisions at each time instant since *(P-1)* computations have already been performed. For a *P*-step prediction and an M-state partial response target, an M × (*P*-1) matrix might be used to save the selected paths for time instant *k*. At time *k+1* and for state $s_i(k+1)$, we compare the two branches that come into this state and select one of them for time instant *k*. The remained path can be directly obtained from the M × (*P*-1) matrix. After the prediction is made, the matrix should be updated using the newly selected branch. Therefore, only one step needs to be traced back at any given time instant for each state.

3.3 Noise Predictive Serially Concatenated Convolutional Turbo System with Soft Feedback

3.3.1 Calculation of the soft tentative decision $\tilde{y}_k$

In the log-MAP algorithm, the log-likelihood ratio of the precoder output z_k is obtained by performing the sum in the numerator and denominator of (2.3) as

$$\begin{aligned} L(z_k \mid \mathbf{y}) &= \ln \frac{P(z_k = +1 \mid \mathbf{y})}{P(z_k = -1 \mid \mathbf{y})} \\ &= \ln \left(\frac{\sum\limits_{(s',s),z_k=+1} \alpha_{k-1}(s') \cdot \gamma_k(s',s) \cdot \beta_k(s)}{\sum\limits_{(s',s),z_k=-1} \alpha_{k-1}(s') \cdot \gamma_k(s',s) \cdot \beta_k(s)} \right) \quad . \end{aligned} \tag{3.10}$$

Note that only one set of α's, β's and γ's needs to be calculated for $L(x_k \mid \mathbf{y})$ and $L(z_k \mid \mathbf{y})$. The conditional expected value of z_k can be calculated from its log-likelihood ratio [16]

$$\begin{aligned} \tilde{z}_k &= E\{z_k \mid \mathbf{y}\} = (+1)P(z_k = +1 \mid \mathbf{y}) + (-1)P(z_k = -1 \mid \mathbf{y}) \\ &= (+1) \cdot \frac{e^{L(z_k \mid \mathbf{y})}}{1 + e^{L(z_k \mid \mathbf{y})}} + (-1) \cdot \frac{1}{1 + e^{L(z_k \mid \mathbf{y})}} \\ &= \tanh(L(z_k \mid \mathbf{y})/2) \qquad\qquad (-1 < \tilde{z}_k < 1). \end{aligned} \tag{3.11}$$

Then, the estimation of the noiseless channel output for PR4, given in (3.5), takes the form

$$\begin{aligned} \tilde{y}_k &= E(z_k \mid \mathbf{y}) - (z_{k-2} \mid \mathbf{y}) \\ &= \tanh(L(z_k \mid \mathbf{y})/2) - \tanh(L(z_{k-2} \mid \mathbf{y})/2), \qquad (-2 < \tilde{y}_k < 2). \end{aligned} \tag{3.12}$$

The above approach for calculating $\tilde{y}_k$ was used in all the simulations in this work. Another approach is to calculate $\tilde{y}_k$ recursively from $L(x_k \mid \mathbf{y})$ using log-likelihood algebra. From the polynomial of the precoder, we have

$$\begin{aligned} L(z_k) &= L\left(\frac{1}{x_k \oplus z_{k-2}}\right) \\ &= L(x_k \oplus z_{k-2}) \\ &= \ln\left[\frac{e^{L(x_k \mid \mathbf{y})} + e^{L(z_{k-2})}}{1 + e^{L(x_k \mid \mathbf{y})} e^{L(z_{k-2})}}\right]. \end{aligned} \tag{3.13}$$

Since the trellis is terminated for each block, the initial values for both $L(z_{-1})$ and $L(z_{-2})$ are $-\infty$. Once we have $L(z_k)$, the estimate of the noiseless channel output follows from (3.12)

$$\begin{aligned} \tilde{y}_k &= \tanh(L(z_k)/2) - \tanh(L(z_{k-2})/2) \\ &= \frac{e^{L(x_k \mid \mathbf{y})} + e^{L(z_{k-2})} - (1 + e^{L(x_k \mid \mathbf{y})} e^{L(z_{k-2})})}{e^{L(x_k \mid \mathbf{y})} + e^{L(z_{k-2})} + (1 + e^{L(x_k \mid \mathbf{y})} e^{L(z_{k-2})})} - \frac{e^{L(x_{k-2} \mid \mathbf{y})} + e^{L(z_{k-4})} - (1 + e^{L(x_{k-2} \mid \mathbf{y})} e^{L(z_{k-4})})}{e^{L(x_{k-2} \mid \mathbf{y})} + e^{L(z_{k-4})} + (1 + e^{L(x_{k-2} \mid \mathbf{y})} e^{L(z_{k-4})})}. \end{aligned} \tag{3.14}$$

The forward predicted noise sample for the input y_k is

$$\tilde{n}_k = \mathbf{a}^{\mathrm{T}}(\mathbf{y}_{k-1}^{k-P} - \tilde{\mathbf{y}}_{k-1}^{k-P}), \tag{3.15}$$

and the backward predicted noise sample is

$$\tilde{n}_k = \mathbf{a}'^{\mathrm{T}}(\mathbf{y}_{k+1}^{k+P} - \tilde{\mathbf{y}}_{k+1}^{k+P}), \tag{3.16}$$

where the superscript **T** stands for vector transpose. The required modification of the channel *a posteriori* decoder involves only the branch transition probability. The modified branch transition probability is

$$\gamma_k(s',s) = \left(-\frac{1}{2\tilde{\sigma}^2}\left|y_k - \bar{y}_k(s',s) - \tilde{n}_k\right|^2\right) + \left(\frac{1}{2}x_k \cdot L(x_k)\right), \tag{3.17}$$

where $\tilde{\sigma}^2$ is the noise variance. Since we predict the noise and subtract it in the calculation of the branch transition probability, the actual noise in each iteration is different. It can be approximately estimated from the previous iteration as

$$\tilde{\sigma}^2 = Var(\mathbf{y} - \hat{\mathbf{y}} - \tilde{\mathbf{n}}), \tag{3.18}$$

where $\hat{\mathbf{y}}$ and $\tilde{\mathbf{n}}$ are the hard decision and predicted noise obtained from the previous iteration. Our simulations show that the performance of NPTS/SF is not very sensitive to changes in $\tilde{\sigma}^2$ for the system considered. Therefore, the original σ^2 can be used for all iterations in (3.17), avoiding the computation in (3.18).

3.3.2 Estimation error for NPTS/SF

For NPTS/HF, if the selected paths contain the correct path, the prediction is optimal. Otherwise, the prediction is wrong and error propagation occurs. Unlike NPTS/HF, NPTS/SF never makes an exact prediction. In this section, we will study the characteristics of the estimation error in (3.4). This is helpful in designing prediction schemes. Let us assume that the hard decisions of $L(z_k \mid \mathbf{y})$ and $L(z_{k-2} \mid \mathbf{y})$ are correct. Fig. 3.2 shows a three-dimension plot of $\tilde{y}_k$ versus $L(z_k \mid \mathbf{y})$ and $L(z_{k-2} \mid \mathbf{y})$ for a PR4 channel. From (3.12) we find that when the amplitudes of both $L(z_k \mid \mathbf{y})$ and $L(z_{k-2} \mid \mathbf{y})$ are infinite, $\tilde{y}_k$ converges to one of the three correct decision planes $\bar{y}_k = \{+2, 0, -2\}$. Since $\tilde{y}_k$ exponentially converges to $\bar{y}_k$, the estimation error is negligible for most values of $L(z_k \mid \mathbf{y})$ and $L(z_{k-2} \mid \mathbf{y})$. We call the regions of the $L(z_k \mid \mathbf{y})$ and $L(z_{k-2} \mid \mathbf{y})$ plane which correspond to a non-negligible estimation error the transition areas. The transition areas are the regions

$$\left|L(z_k \mid \mathbf{y})\right| < a \tag{3.19}$$

and

$$\left|L(z_{k-2} \mid \mathbf{y})\right| < a\,, \tag{3.20}$$

where a is the boundary that satisfies a required estimation error. For example, setting a=6, if $L(z_k \mid \mathbf{y})$=-6 and $L(z_{k-2} \mid \mathbf{y})$=6, the noiseless channel output is

$$\begin{aligned} \bar{y}_k &= sign\big(L(z_k \mid \mathbf{y})\big) - sign\big(L(\bar{z}_{k-2} \mid \mathbf{y})\big) \\ &= -2, \end{aligned} \tag{3.21}$$

and its estimate is

$$\begin{aligned} \tilde{y}_k &= \tanh\big(L(z_k \mid \mathbf{y})/2\big) - \tanh\big(L(z_{k-2} \mid \mathbf{y})/2\big) \\ &= -1.99\,. \end{aligned} \tag{3.22}$$

The absolute error introduced by using $\tilde{y}_k$ is 0.01, which is negligible for making a prediction. In these regions, the tentative hard decision is correct, but the absolute value of e_k is

$$0 < |y_k - \tilde{y}_k| \le 1. \tag{3.23}$$

The joint region

$$|L(z_k \mid \mathbf{y})| < a \cap |L(z_{k-2} \mid \mathbf{y})| < a \tag{3.24}$$

should be specifically taken care of because the transition from $\tilde{y}_k \approx -2$ to $\tilde{y}_k \approx +2$ occurs in this area. The maximum absolute error introduced by $\tilde{y}_k$ occurs when

$$L(z_k \mid \mathbf{y}) \to 0^- \cap L(z_{k-2} \mid \mathbf{y}) \to 0^+ \tag{3.25}$$

or

$$L(z_k \mid \mathbf{y}) \to 0^+ \cap L(z_{k-2} \mid \mathbf{y}) \to 0^- \tag{3.26}$$

where "-" means approaching from the negative direction, and "+" means approaching from the positive direction. It is obvious that the maximum absolute error introduced by $\tilde{y}_k$ in this area is

$$0 < |\bar{y}_k - \tilde{y}_k| \le 2. \tag{3.27}$$

In a similar way and considering the fact that whenever $L(z_k \mid \mathbf{y})$ has an erroneous sign it tends to have low amplitude, we find that $\tilde{y}_k$ introduces less estimation error when the tentative hard decisions are not correct.

3.3.3 Selecting prediction direction

Ideally, noise prediction needs to be performed only for those y_k where the confidence for x_k is low. Since we actually do not know how large the confidence of x_k should be, we predict the noise sample for all the elements of $\mathbf{y}$ in our simulations. From (3.12), the confidence of $\tilde{y}_k$ is determined by the amplitude of $L(z_k \mid \mathbf{y})$ and $L(z_{k-2} \mid \mathbf{y})$. To determine which side to use in the prediction, we use the following criteria:

i) Consider both $\tilde{y}_k$ and the amplitude of the predictor coefficients. For example, in our simulation, we used a 3-tap linear predictor. For a PR4 channel, the amplitudes of tap-1 and tap-2 are much larger than that of tap-3. Therefore, the accuracy of the prediction is not heavily dependent on the values of $\tilde{y}_{k-3}$ or $\tilde{y}_{k+3}$.

ii) The $\tilde{y}_k$'s that correspond to the two transition areas are not recommended to be used for prediction, especially for the $\tilde{y}_k$'s that are calculated from the joint region (3.24).

We found that the following two empirical criteria work well, at least for a third order predictor. Let

$$
\begin{aligned}
c = &\sum_{m=1}^{P} \left|a_m\right| \left(\left|\tanh(L(z_{k-m} \mid \mathbf{y})/2)\right| + \left|\tanh(L(z_{k-2-m} \mid \mathbf{y})/2)\right| \right) \\
&- \sum_{m=1}^{P} \left|a_m\right| \left(\left|\tanh(L(z_{k+m} \mid \mathbf{y})/2)\right| + \left|\tanh(L(z_{k-2+m} \mid \mathbf{y})/2)\right| \right)
\end{aligned}
\tag{3.28}
$$

or simply

$$
\begin{aligned}
c = &\left(\left|\tanh(L(x_{k-1} \mid \mathbf{y})/2)\right| + \left|\tanh(L(x_{k-2} \mid \mathbf{y})/2)\right| \right) \\
&- \left(\left|\tanh(L(x_{k+1} \mid \mathbf{y})/2)\right| + \left|\tanh(L(x_{k+2} \mid \mathbf{y})/2)\right| \right).
\end{aligned}
\tag{3.29}
$$

If $c \geq 0$, we use forward prediction, otherwise backward prediction is used. For the first and the last several elements of **y**, only one side prediction can be performed.

Although forward and backward prediction can also be performed for NPTS/HF, we only investigated the performance of forward prediction.

3.3.4 Predictor bank

In trying to avoid the use of an unreliable $\tilde{y}_k$, we propose the use of a predictor bank with NPTS/SF. The predictor bank consists of several linear predictors, which we call component predictors. Each component predictor has the same length and corresponds to a different set of past or future neighbors of $\tilde{y}_k$. At each time, only one component predictor is used. This gives us more flexibility to avoid using the expected values calculated in the joint region. For example, we can chose three out of the five past estimates of y_k for forward prediction.

3.4 Performance Evaluation and Discussion

3.4.1 Performance evaluation of NPTS/SF

We consider a serially concatenated convolutional PR4 system, shown in Fig. 2.1. The channel density is 2.86. In the signal-to-noise ratio region of interest, we observed that most values of $L(z_k \mid \mathbf{y})$ are large enough so that $\tilde{z}_k$ is approximately the ideal value of the precoder output z_k. This is equivalent to $L(z_k \mid \mathbf{y})$ being outside the transition regions. The histograms of $L(x_k \mid \mathbf{y})$ and $L(z_k \mid \mathbf{y})$ at SNR=19.6 dB for iteration one are shown in Figs. 3.3 and 3.4, respectively. It should be noted that at this SNR, the bit-error-rate is approximately 10^{-2}, which means the confidences of $L(x_k \mid \mathbf{y})$ and $L(z_k \mid \mathbf{y})$ are very low.

For all simulations, a one-step, three-tap linear predictor is used. We predicted the noise sample for each y_k. The prediction direction was selected using (3.29), which means that the tentative soft decisions that were calculated from the transition areas and their joint region were also used in the prediction.

To evaluate the performance of NPTS/SF, we first simulated the system without noise prediction. Fig. 3.5 shows the results of this serially concatenated system for iterations five to ten. It can be seen that between iteration eight and ten only a small performance gain can be achieved, which means that increasing the maximum number of iterations would not give any significant performance improvement. Fig. 3.6 compares the performance of the NPTS/SF from iteration six to twenty with a standard system after ten iterations. The performance gain at a bit-error rate of 5×10^{-6} is approximately 1.7 dB.

At SNR=19.8 dB, Fig. 3.7 shows the absolute value of the noise autocorrelation $r_{nn}(0)$ to $r_{nn}(8)$ for twenty iterations of the noise predictive PR4 system. It can be seen that $r_{nn}(0)$ to $r_{nn}(3)$ have been reduced with each iteration. It can also be seen that the effect of noise whitening is concentrated on the first several iterations. Since we used a three tap predictor, $r_{nn}(4)$ stays unchanged.

Although we could not simulate the error floor performance of the noise predictive PR4 system, it is reasonable to expect that it might have been significantly improved. This can be verified by the variation of $r_{nn}(0)$ in Fig. 3.7. We can see that almost one third of the noise energy has been removed by using prediction. With less noise energy and whiter noise, the error floor of the noise predictive PR4 should be lower.

NPTS/SF improves the performance of the system by increasing the amplitude of $L_{ext}^{x}(x_k)$, which is the information passed to the outer error-correction decoder. The outer decoder takes this information as its channel value, which is now more reliable. For SNR=19.6 dB, block length 4354 bits, Figs. 3.8 and 3.9 show $L_{ext}^{x}(x_k)$ for twenty iterations of the standard and predictive systems, respectively. In these two figures, bits 0~4353 on the horizontal axis correspond to the first iteration, bits 4354~8707 correspond to the second iteration, and so on. For the standard system, we observed approximately an average of 180 bits in error after each iteration for this block. The reason is that at this low SNR the iterative decoding system cannot improve its decisions from one iteration to the next. This can be explained by Fig. 3.8, where the amplitude of the turbo equalizer output $L_{ext}^{x}(x_k)$ was not improved by iterating. For the noise predictive system, the number of erroneous bits decreases with every iteration. The decoding is successful at the seventh iteration. This is consistent with Fig. 3.9, where the amplitude of $L_{ext}^{x}(x_k)$ is continually improved. Fig. 3.6 shows that at this SNR, the bit-error rate of the predictive systems is approximately 10^{-5}, whereas it is much higher for the conventional system.

We should point out that we did not predict the noise samples for the first and the last two y_k's. These bits benefit from the initialization in (2.6), and therefore they usually have a large LLR of *a posteriori* probabilities. We can see in Fig. 3.9 that these bits are the isolated points whose amplitude did not improve with successive iterations.

We also simulated the same system with a predictor bank, which selects three out of four past or future tentative decisions. In predicting the noise in y_k, we simply looked at the amplitudes of $[L(x_{k-4} \mid \mathbf{y}), \cdots, L(x_{k-1} \mid \mathbf{y})]$ and $[L(x_{k+1} \mid \mathbf{y}), \cdots, L(x_{k+4} \mid \mathbf{y})]$, and used these amplitudes as the corresponding confidences for selecting $\tilde{y}_{k-m}$ or $\tilde{y}_{k+m}$. The simulated performance was worse than that of the single predictor. Intuitively, we think the performance of the predictor bank should be better as long as we can properly select the tentative decisions. From (3.14) we find that $\tilde{y}_k$ is not determined only by $L(x_k \mid \mathbf{y})$. Therefore, our selection criterion for the predictor bank is not adequate. Generally, selecting prediction direction and tentative decisions is more complex for a predictor bank because we should take the tradeoff between using unreliable tentative decisions and using multi-step prediction into consideration. Multi-step prediction is inherently less reliable than single-step prediction.

3.4.2 Performance evaluation of NPTS/HF

We consider the same system and density as in the previous section. To compare NPTS/HF with NPTS/SF, we also used a three-tap linear predictor in this section. At channel density 2.86, the bit-error-rate performance of NPTS/HF is shown in Fig. 3.10 for iterations six to twenty. As a comparison, the performances of a conventional PR4 turbo system at iteration ten and an uncoded PR4 system are also shown. It can be seen that at bit-error-rate of 5×10^{-6}, NPTS/HF has 1.9 dB gain over the conventional turbo system.

The performance of a ME^{2}PR4 channel, which has the same parameters except for the precoder which is $1/(1+D+D^2+D^3+D^4)$, was studied in [17]. In Fig. 3.11, the bit-error rate performance of NPTS/HF, NPTS/SF, conventional PR4 and ME^{2}PR4 turbo systems is compared. It can be seen that both NPTS/HF and NPTS/SF outperform the ME^{2}PR4 turbo system.

3.4.3 Performance evaluation at very high density

Performance of NPTS is also simulated at a channel density of 3.19. Fig. 3.12 compares NPTS/SF with the conventional PR4 turbo system. Fig. 3.13 shows the performance comparison of NPTS/HF and the conventional PR4 turbo system. At bit-error-rate 5×10^{-5}, NPTS/SF has approximately 1.5 dB gain over the conventional PR4 turbo system, whereas NPTS/HF has a 2-dB SNR improvement.

We found that the bit-error-rate of NPTS/SF is improved almost equally at each value of SNR, which shows it is reasonable to deduce that additional iterations may still improve the bit-error-rate. On the other hand, the performance of NPTS/HF was not improved for iterations sixteen to twenty at most points in Figs. 3.10 and 3.13. Although the performance of NPTS/HF is better than NPTS/SF within twenty iterations, which one is better with unlimited iterations is not known.

An important phenomenon is shown in Fig. 3.13 where NPTS/HF does not converge steadily. We would like to point out that the input data and noise sequence at the receiver are identical for NPTS/SF and NPTS/HF. Since it is hard to analyze the convergence properties of conventional turbo systems and noise predictive turbo systems, we conjecture that NPTS/HF causes more severe error propagation, which in turn leads to convergence problems.

3.5 Conclusions

We have introduced iterative noise prediction techniques, which were shown to be powerful methods to improve iterative decoding of partial response channels. For NPTS/SF, there are two main reasons for its excellent performance. First, the availability of soft tentative decisions $\tilde{y}_k$, which are optimal estimates of the ideal channel output in the minimum-mean-squared-error sense; and secondly, turbo equalization makes it possible to perform iterative noise prediction with improved accuracy.

In the simulation, NPTS/HF shows better performance than NPTS/SF within twenty iterations, but the latter has the potential to improve further with additional iterations and possibly suffer less error propagation.

Real hard disk drive experiments have shown that the most frequent error event for a turbo coded EPR4 system is the single-bit error event [17]. This is well suited to the use of a short predictor because the single-bit error is likely to be corrected by noise prediction, and it also might be avoided in predicting the noise sample for other bits. We think that this technique can also improve the performance of higher order partial response channels as long as its noise is correlated.

It has been shown that the correlation and energy of the noise were significantly reduced, and therefore performance of the detector improved. These techniques are expected to be particularly attractive for very high-density magnetic recording where correlated noise will be particularly significant. Since NPTS/SF can be implemented with a simple finite-impulse response filter, even longer predictors do not increase the detector complexity significantly.

As a conclusion, we would like to highlight other potential applications of iterative estimation, which is the basis of the noise predictive turbo systems. A block Viterbi decoder was proposed in [18] to take advantage of correlated channel noise. Using the same idea, a block MAP decoder can be derived directly. Straightforward implementation of a block MAP decoder is impractical since the number of branches of the trellis increases exponentially with the block length. Using the principle of iterative estimation, we can ignore noise correlation in the first iteration, and then estimate the noise component at the channel output for the consecutive iterations. With the vector form probability density function [6], better decoding results might be obtained by considering noise correlation. This iterative approach keeps the number of trellis branches the same as in the white Gaussian noise case.

Low-density-parity-check (LDPC) codes were recently rediscovered and have been used as powerful error-correction codes [19]. If the combination of the outer encoder and the interleaver in Fig. 2.1 are replaced with an LDPC encoder, we can use the pseudo-posteriori probability to estimate the noise component in the received channel output. In our simulations, we used the *a posteriori* probability of the turbo equalizer to do it. Obviously, the pseudo-posteriori probability is much more reliable than the *a posteriori* probability of the turbo equalizer after the first iteration. Therefore, using pseudo-posteriori probability provides a better starting point. The potential benefits are twofold: less iteration is required and probably larger performance gain can be obtained; computation is simple since the pseudo-posteriori probability can be viewed as the "soft bits" of the precoder input.

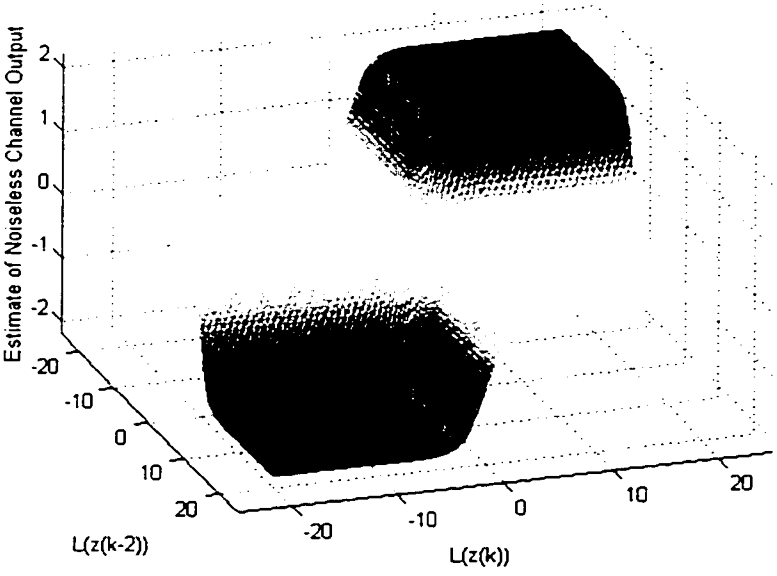


Fig. 3.2. $\tilde{y}_k$ versus $L(z_k \mid \mathbf{y})$ and $L(z_{k-2} \mid \mathbf{y})$.

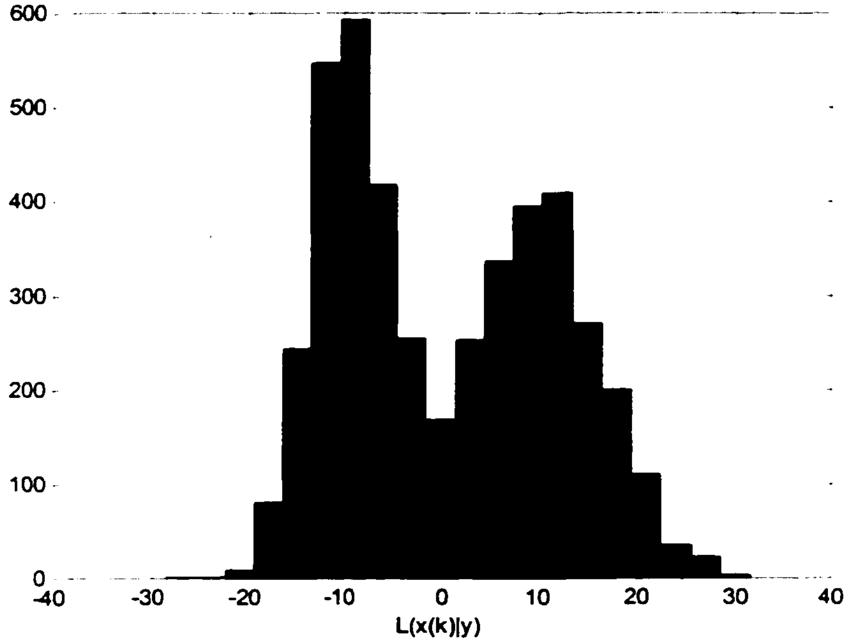


Fig. 3.3. Histogram of $L(x_k \mid \mathbf{y})$ after the first iteration, SNR = 19.6 dB.

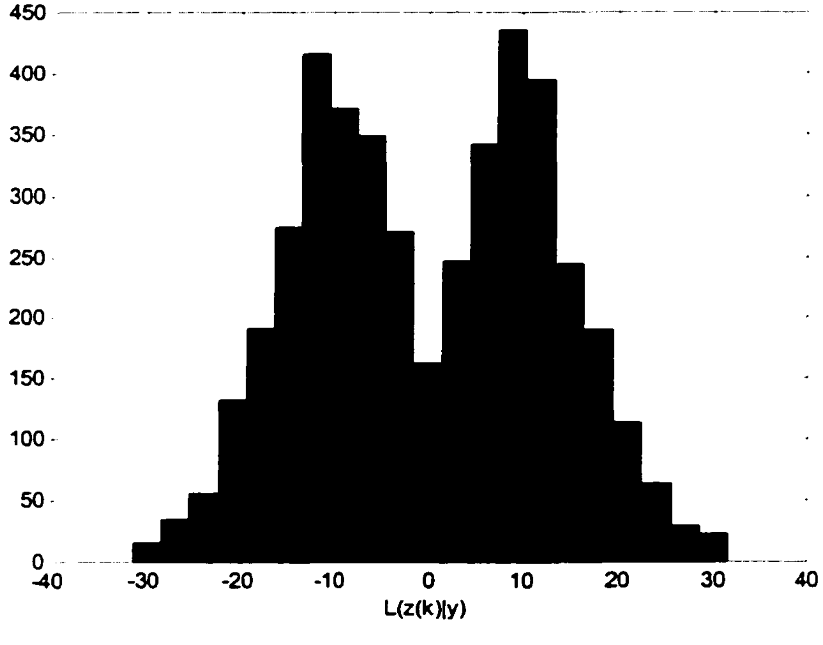


Fig. 3.4. Histogram of $L(z_k \mid \mathbf{y})$ after the first iteration, SNR = 19.6 dB.

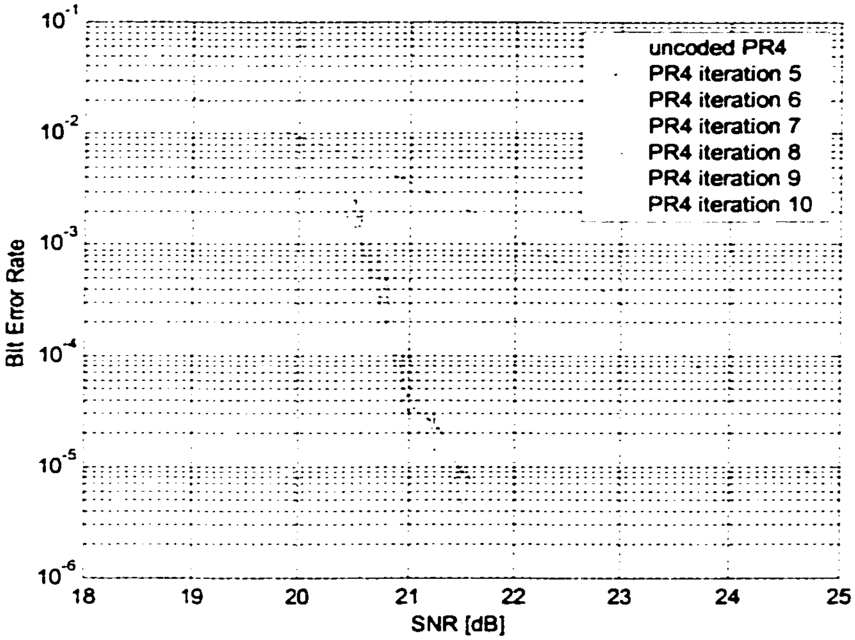


Fig. 3.5. Bit-error-rate performance of the PR4 turbo system.

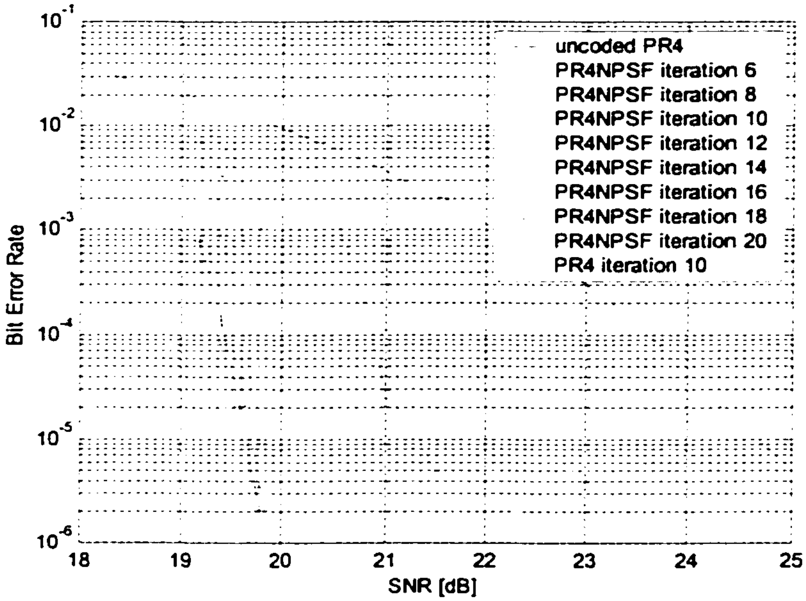


Fig. 3.6. Performance comparison of NPTS/SF with the PR4 turbo system.

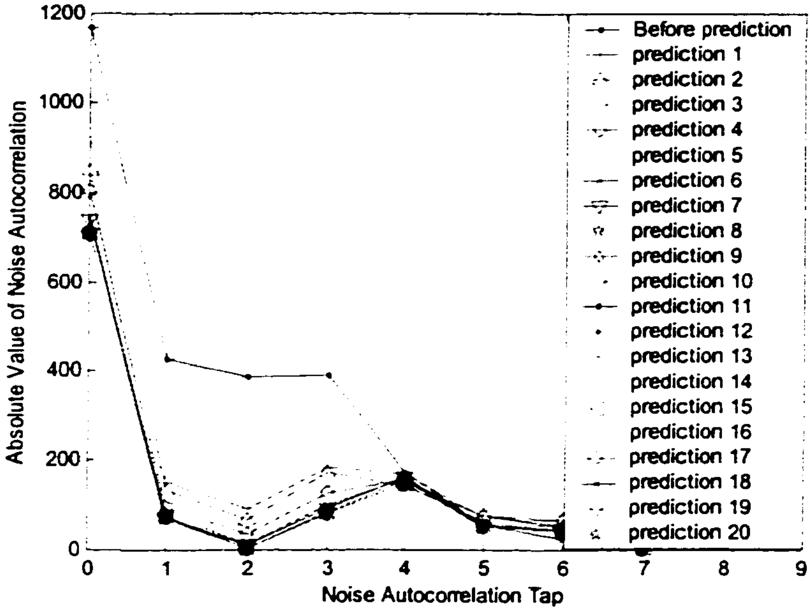


Fig. 3.7. Absolute values of the noise autocorrelation with number of iterations as a parameter.

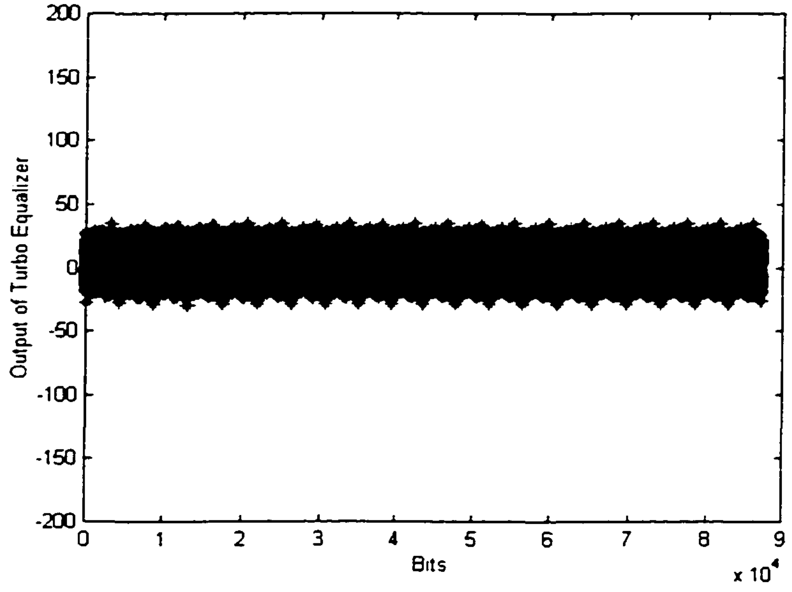


Fig. 3.8. $L_{ext}^{x}(x_k)$ for twenty iterations of the standard PR4 system.

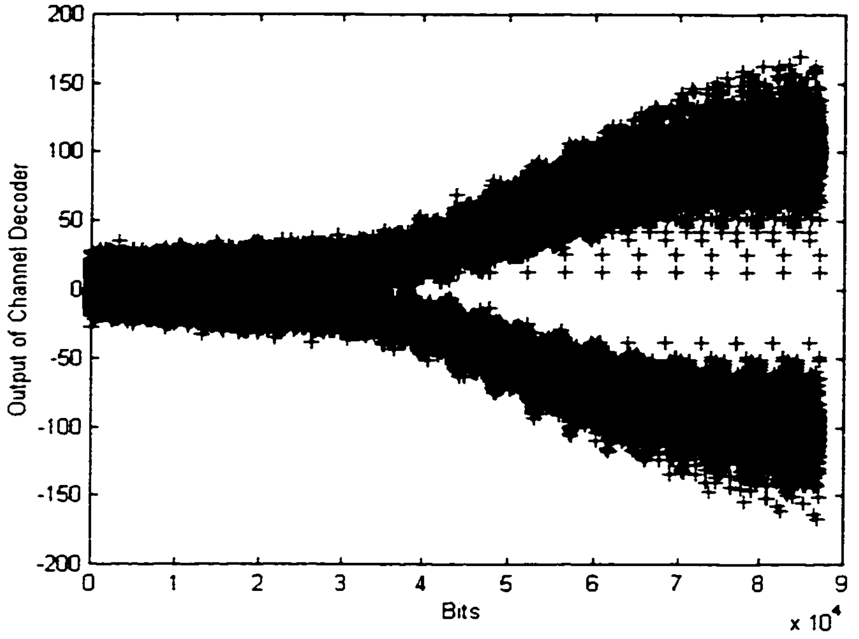


Fig. 3.9. $L_{ext}^{x}(x_k)$ for twenty iterations of the noise predictive PR4 system.

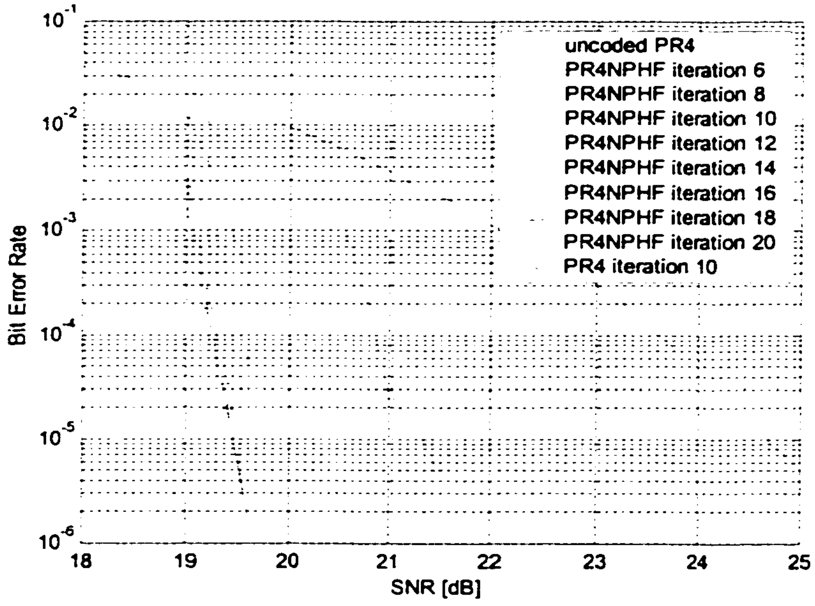


Fig. 3.10. Performance comparison of NPTS/HF with the PR4 turbo system.

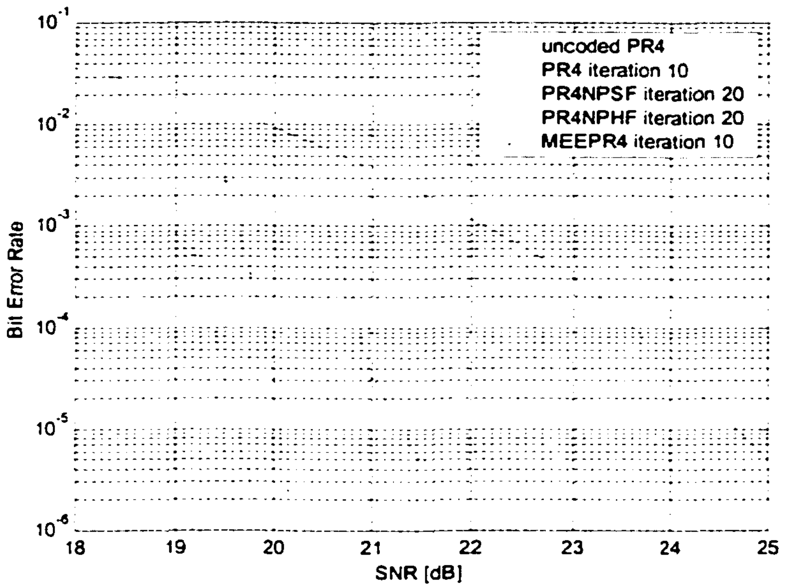


Fig. 3.11. Perforamnce comparison of NPTS/HF, NPTS/SF, MEEPR4 and the PR4 turbo systems.

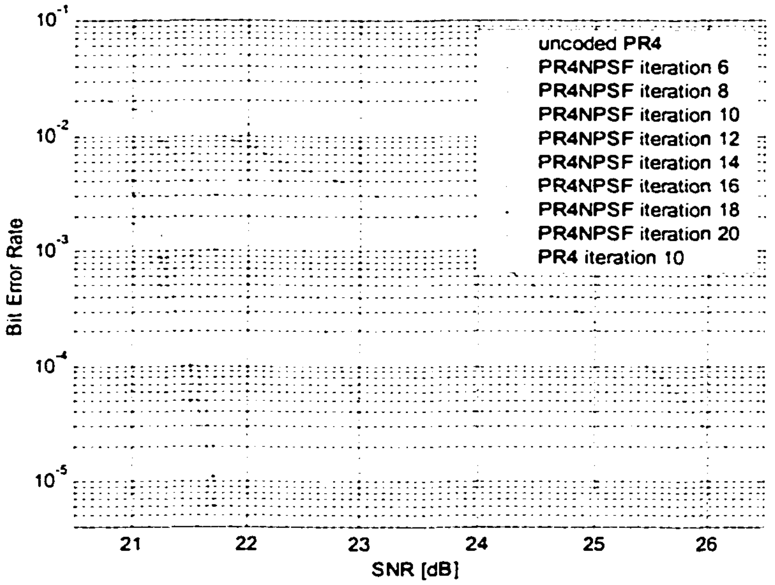


Fig. 3.12. Performance of NPTS/SF at very high recording density.

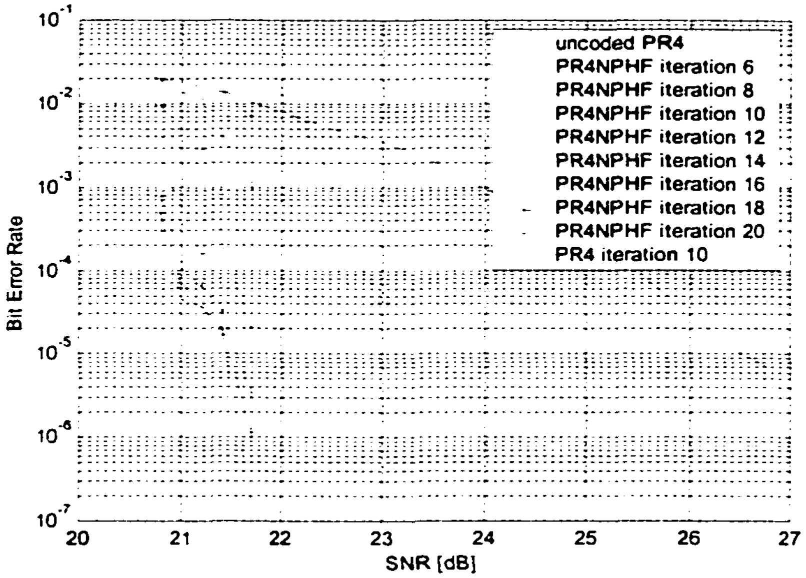


Fig. 3.13. Performance of NPTS/HF at very high recording density.

Chapter 4

Iterative Detection for Partial Response Magnetic Recording Channels – A Graphical View

4.1 Introduction

The study of parallel concatenated convolutional codes (turbo codes) led to the invention of iterative decoding algorithms for serially concatenated convolutional codes and the rediscovery of low-density parity-check codes. Recently, it was recognized that iterative decoding algorithms can be categorized as probability propagation algorithms. This insight provides us with a new way to view iterative decoding algorithms [9].

The investigation of the performance of iterative detection on magnetic recording channels started in the late 1990s. All the reported results verified that iterative soft detection was significantly better than conventional hard detection schemes. In this chapter, we seek to understand various decoding techniques for partial response equalized magnetic recording channels using graphical models. After the introduction of Bayesian networks and the belief propagation algorithm, we derive the proper Bayesian network and belief passing schemes for correlated noise channels. Secondly, noise predictive turbo systems were recently proposed in [20], and shown to have excellent performance. We will show that this technique is also a belief propagation algorithm applied to the parameterized Bayesian network of a serially concatenated system. Finally, the concept of iterative noise estimation will be introduced, and its possible application to signal dependent noise channels will be discussed.

4.2 System Model

We consider a partial response magnetic recording system. The major components in this system are the encoder of a single recursive systematic convolutional code (RSC), the inner encoder and the detector. The inner encoder represents the combination of the precoder and equalized channel. Assume that the recording channel is equalized to a partial response target of the form

$$(1 - D)(1 + D)^n, n = 1, 2, 3. \tag{4.1}$$

It is well known that the inner encoder can be viewed as a non-systematic and non-binary convolutional code, and the recording system is essentially a serially concatenated convolutional code. The codewords of the inner encoder are the noiseless channel outputs. After transmission, noise is added to the codeword. In this chapter, the added noise takes into account the equalized channel noise and misequalization. In addition, we assume the information data u_k is an independent and identically distributed, equally-likely binary sequence taken from {0, +1}.

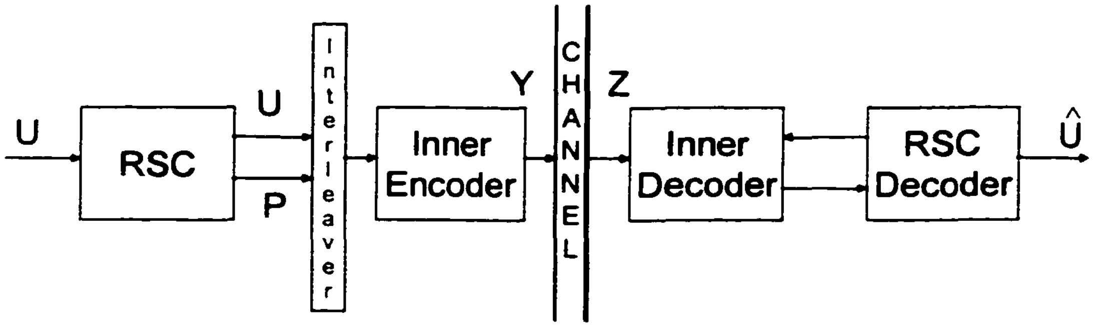


Fig. 4.1. Diagram of a concatenated magnetic recording system.

Particularly, the rate ½ recursive systematic convolutional encoder output is punctured to give a code rate of 16/17. Therefore, the encoding process can be represented as

$$\mathbf{U} \rightarrow (\mathbf{U}, \mathbf{P}) \rightarrow \mathbf{Y}, \tag{4.2}$$

where **U** is the systematic part and **P** is the parity part of the codeword.

4.3 Bayesian networks and the belief propagation algorithm

4.3.1 Bayesian networks

Bayesian networks are directed acyclic graphs, where nodes represent the random variables and edges represent probabilistic dependencies among the nodes. Fig. 4.2 shows a simple (actually a very simple) Bayesian network for uncoded partial response equalized magnetic recording channels. Here [U_1, U_2] is the binary data written into the channel. $[S_1, S_2]$ are the state variables of the channel, and $[X_1, X_2]$ and $[Y_1, Y_2]$ are sampled noiseless channel outputs and noise corrupted channel outputs, respectively. For simplicity, we assume the equalized channel noise and misequalization are a white noise process in this example.

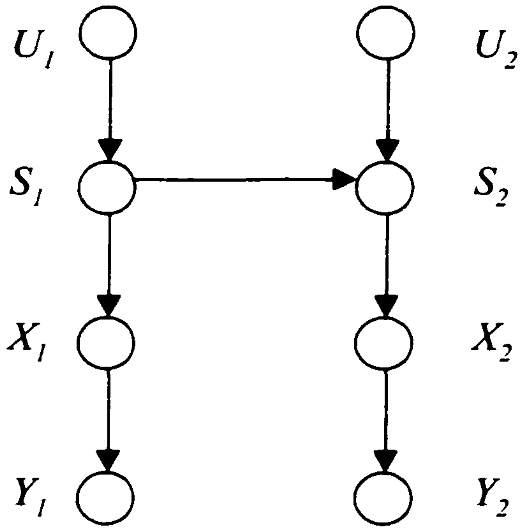


Fig. 4.2. A simple Bayesian network for uncoded megnatic recording systems.

In a Bayesian network, if there is a directed edge from node A to node B, A is called a parent of B and B is called a child of A. If we denote the set of the parents of node A as $pa\{A\}$, these relationships hold for nodes S_1 and S_2 in Fig. 4.2,

$$pa\{S_1\} = \{U_1\}, \qquad pa\{S_2\} = \{S_1, U_2\}. \tag{4.3}$$

Similarly, if we denote the set of children of node A as $c\{A\}$, these relationships hold for nodes S_1 and S_2 in Fig. 4.2.

$$c\{S_1\} = \{X_1, S_2\}, \qquad c\{S_2\} = \{X_2\}. \tag{4.4}$$

In addition to represent probabilistic dependencies among variables, a Bayesian network also clearly shows conditional probabilistic independence. That is the probability of a node is independent to the non-parent nodes given the values of its parent nodes. The probabilistic dependencies naturally enable us to model a problem as a Bayesian network, and the probabilistic independencies help us to simplify the inference problems.

Assume that our task is to determine the probability that U_1 is u_1 based on the observation $[Y_1 = y_1,\ Y_2 = y_2]$. This problem can be solved by the joint distribution, e. g., $P(U_1, U_2, S_1, S_2, X_1, X_2, Y_1, Y_2)$, represented by Fig. 4.2,

$$\begin{aligned} &P(U_1 = u_1 \mid Y_1 = y_1, Y_2 = y_2) \\ &\quad = \alpha \sum_{(S_1,S_2,U_2,X_1,X_2)} P(u_1, U_2, S_1, S_2, X_1, X_2, y_1, y_2), \end{aligned} \tag{4.5}$$

where α is a constant independent of U_1. According to the chain rule [10]

$$\begin{aligned} &P(u_1, U_2, S_1, S_2, X_1, X_2, y_1, y_2) \\ &\quad = P(u_1)P(U_2 \mid u_1)P(S_1 \mid u_1, U_2) \ldots P(y_2 \mid u_1, U_2, S_1, S_2, X_1, X_2, y_1). \end{aligned} \tag{4.6}$$

A straightforward calculation of this probability is not efficient. Using the conditional probabilistic independence represented by the Bayesian network we can simplify the computation. For example, we can find

$$P(Y_2 = y_2 \mid u_1, U_2, S_1, S_2, X_1, X_2, Y_1 = y_1) = P(Y_2 = y_2 \mid X_2). \tag{4.7}$$

If we also note that the probabilities $P(S_1 \mid pa\{S_1\})$, $P(S_2 \mid pa\{S_2\})$, $P(X_1 \mid pa\{X_1\})$, and $P(X_2 \mid pa\{X_2\})$ are either zeros or ones, this detection problem can be easily solved. Through this example, we find that the general magnetic recording channel can be represented by the framework of Bayesian networks, and the Bayesian networks provide us with a convenient way to take advantage of the conditional probabilistic independence inherent in the recording process.

4.3.2 Probability propagation algorithm

Detection in an uncoded magnetic recording system is the process of determining the probability of each user bit using the observed noisy channel output. Once we obtained the Bayesian network for the system, this task can be efficiently solved by applying Pearl's probability propagation algorithm on the network. In this algorithm, probability information is passed across each node (except the observation nodes) of the Bayesian network so that the *posteriori* probabilities for each node can be calculated conditioned on the observation [10]. The messages can be categorized into two types according to the parent-child and child-parent relationships among the nodes. The parent-child message is a vector of probabilities. Each value of an element of the vector is a probability of the parent node conditioned on all the information available for a child node from the sub-network that is separated by the edge between the parent node and the child node. For example, the parent-child messages that state S_1 passed to state S_2 in Fig. 4.2 is

$$\{P(S_1 = s_k \mid State_{S_1 \to S_2},\quad S_1 \in [s_1,......s_M]\}, \tag{4.8}$$

where $State_{S_1 \to S_2}$ denotes the mentioned information about the sub-network consisting of nodes U_1, S_1, X_1, and Y_1 , [10]. To the local environment of node S_2, this information can be viewed as the *a priori* information of S_1.

To update the belief (*a posteriori* probability) of node S_2, the message passed from its children is also required. Generally, this child-parent type message is also a vector of probabilities. Each probability is the likelihood of the current state of the sub-network that is separated by the edge between S_2 and X_2 conditioned on the value of S_2

$$\{P(State_{X_2 \to S_2} \mid S_2 = s_k),\quad S_2 \in [s_1,.....s_M]\}, \tag{4.9}$$

where the sub-network consists of the nodes X_2 and Y_2.

After receiving messages from all its parent nodes and children nodes, S_2 can calculate its *a posteriori* probability using these messages. To ensure the message passing can be continuously performed on its parent and children nodes, S_2 also needs to calculate the child-parent type message for all of its parent nodes and the parent-child type message for all of its child nodes. From the example of message passing for S_2 we can see that both of the two type messages must be ready for a node before updating the probabilities of the node. In this chapter, only a functional description of Pearl's belief propagation algorithm is given. The initialization of Bayesian networks and the mathematical part of the algorithm have been thoroughly discussed in [9] and [10].

4.4 Bayesian Network for Serially Concatenated Magnetic Recording Systems

4.4.1 Probability propagation on the Bayesian network of the recording system with white Gaussian noise

When the outer encoder of Fig. 4.1 is a rate ½ recursive systematic convolutional encoder (RSC), and we assume the channel noise to be additive white Gaussian noise (AWGN), the recording system turns out to be a serially concatenated convolutional coded system. The turbo decoding algorithm [21] has been shown to be an instance of the probability propagation algorithm [9], which also mentioned that the classical decoding algorithm for serially concatenated turbo codes [22] can be derived from the probability propagation algorithm. Two suitable Bayesian networks for serially concatenated convolutional codes were given in [10] and [9]. The Bayesian network for this magnetic recording system can be obtained by slightly modifying the Bayesian network given in [10]. Fig. 4.3(a) shows a section of the Bayesian network for a rate 16/17 serially concatenated recording system. This high rate code was obtained by systematically puncturing the first to fifteenth parity bits in each sixteen parity bits. As an extension, the message passing schedule will be given for this network which indicates how the decoding algorithm can be obtained by applying the probability propagation on Fig. 4.3(a). It is worth to point out that a different message passing schedule usually leads to a different decoding algorithm, and in many cases there are different message passing schedules for a given Bayesian network. The general description of the iterative decoding algorithm for a white

Gaussian noise channel through graphical models can be found in [23], [10], and [9]. The main focus of this chapter is to describe (without proof) probability propagation on Bayesian networks for magnetic recording systems with both white Gaussian noise and correlated/signal dependent Gauss-Markov noise. Therefore, although we can find some content of this section in [10], [9], the Bayesian networks for serially concatenated convolutional codes transmitted through a white noise channel will be described because a slightly different Bayesian network and message-passing scheme form the basis of iterative decoding for correlated noise channels.

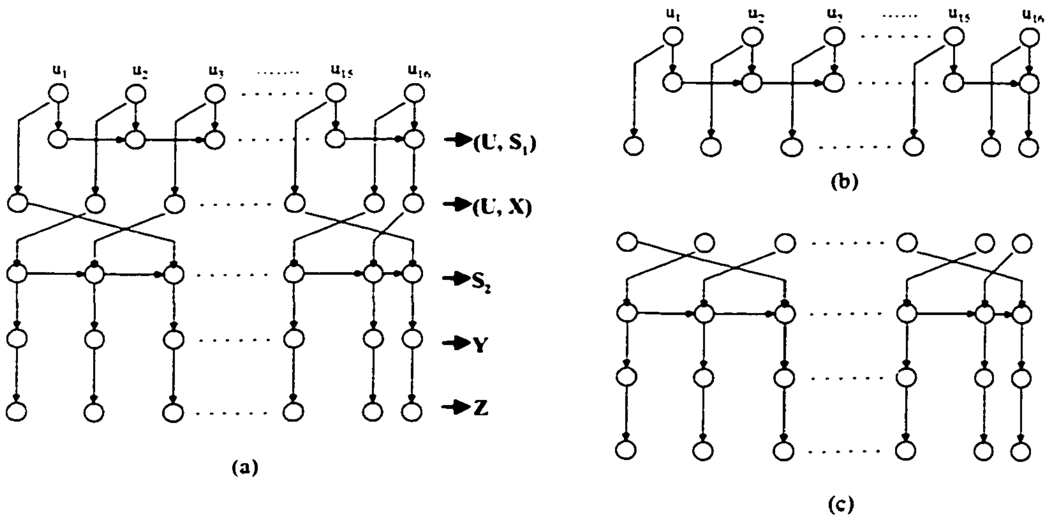


Fig. 4.3. Bayesian network for a serially concatenated magnetic recording system, white Gaussian noise, code rate 16/17.

In Fig. 4.3(a), [u_1, u_2, ..., u_{16}] represents a segment of user bits. The nodes in the second line, denoted (U, S_1), represent the combination of the hidden state nodes and information nodes of the recursive systematic convolutional code

[10]. Due to the effect of puncturing, the length of the convolutional codeword, (**U**, **X**), is seventeen bits. Under these assumptions, the channel is viewed as a non-systematic and non-binary convolutional encoder. The state of the channel is denoted as $\mathbf{S}_2$, and its codeword is **Y**. **Z** is the observed channel output, which is the noisy version of **Y**.

Fig. 4.3(b) shows the Bayesian network for the convolutional code and puncturer, and Fig. 4.3(c) shows the Bayesian network for the random interleaver and the equalized channel. Obviously, both of the two individual networks are singly-connected models. Applying the forward-backward message passing schedule on the two Markov-type models [10], the optimal decoding algorithm which is the same as the BCJR algorithm can be obtained. Since the proof is straightforward, the mathematical details will be ignored.

Although the Bayesian network for the serially concatenated recording system is loopy, probability propagation can still be applied by ignoring the loops. However, this leads to a non-optimal decoding algorithm, and the decoding result is not guaranteed to be useful.

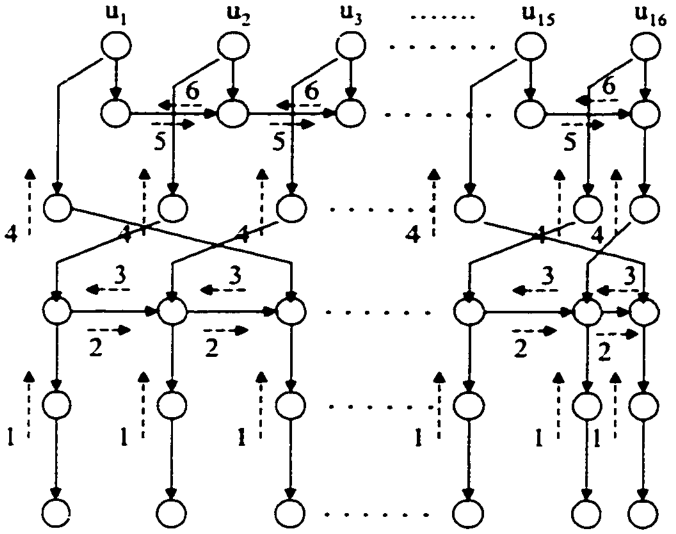


Fig. 4.4. Probability propagation schedule for serially concatenated magnetic recording systems.

Fig. 4.4 shows the message passing process for the serially concatenated recording system. One decoding iteration consists of six message-passing steps, and the six message passing steps can be divided into two parts. Each part corresponds to the message passing in a constituent code. The iteration starts from the Bayesian network for the channel. When the channel output **Z** is observed, messages propagate up to the state vertices by step one. According to Section 4.3.2, this child-parent type message is a vector of likelihood, and each element of the vector is the probability of $[Y_k = y_k, Z_k = z_k]$ conditioned on $S_{2,k} = s_{2,k}$. The state vertices temporarily save the up-going message for the computation of steps two and three. Here, updating the *a posteriori* probability of node $S_{2,k}$ is not necessary because our task is to determine the *a posteriori* probability of each node of **U**. In step two, messages are passed between state nodes from left to right. This is equivalent to the computation of α's in the BCJR algorithm. Step three is a child-parent type message passing between the state nodes from right to left, and the computation involved in this step is exactly the same as the calculation of β in the BCJR algorithm [10].

In step four, the channel decoder calculates the *a posteriori* probability (belief) of each symbol of the codeword (**U**, **X**) and passes it according to the permutation to the state nodes (**U**, $\mathbf{S}_I$) of the convolutional code in the same fashion with step one. After the forward-backward probability propagation in the outer code, steps five and six, the a posteriori probability of each state given the observed channel output is calculated. If the decoding is not performed in an iterative fashion, each state node of the convolutional code passes its *a posteriori* probability to the corresponding information node of **U** to calculate the *a posteriori* probability of each information symbol. In iterative decoding, the process restarts from step one after the sixth step with the consideration of the newly calculated probabilities (extrinsic information). Since the Bayesian network of Fig. 4.4 is a loopy structure, the probability propagation procedure never self-terminates. Some termination criterion must be applied.

4.4.2 Probability propagation on the Bayesian network of the recording system with correlated Gauss-Markov noise

We have derived the Bayesian network for serially concatenated convolutional codes for white Gaussian noise channels, and described the probability propagation schedule for that network. Now, we will discuss the Bayesian network and its probability propagation for correlated noise channels with Gauss-Markov statistics.

As we can see in Fig. 4.1, the convolutional decoder is not connected with channel directly. Therefore, its probability propagation should not be changed for a particular channel. On the other hand, the channel decoder needs to be tuned to the signal and noise statistics because of the noise memory. The decoding algorithm for this channel decoder is essentially a merge of the maximum likelihood sequence detection (MLSD) [24] algorithm for correlated/signal dependent noise channels and the BCJR algorithm. The probability $r_k(s',s)$ in the BCJR algorithm and its modified version are given in [25], respectively

$$\gamma_k(s',s) = \Pr\{S_k = s; z_k \mid S_{k-1} = s'\}, \tag{4.10}$$

$$\gamma'_k(s',s) = \Pr(S_k = s, z_k \mid S_{k-1} = s', \mathbf{z}_{k-1}^{k-L}), \tag{4.11}$$

where L is the length of the memory of the Gaussian-Markov noise. Obviously, the probability $r'_k(s',s)$ for the additive correlated noise channel is also dependent on the noise component of the previous L channel outputs. According to (4.11), the suitable Bayesian network can be obtained. An example for L equal one is given in Fig. 4.5. Comparing Figs. 4.5 and 4.4, we find that the Bayesian network for the system with correlated channel noise is not much different from that for white Gaussian channel noise. Although the Bayesian network for the inner encoder with correlated channel noise has cycles (the lower part of Fig. 4.5), the probability propagation algorithm can solve the decoding problem exactly for this channel. The reason is that the edges between the observation nodes consist of a part of each cycle. Since we do not update the probabilities of the observation nodes, there is no message passing along these edges. Therefore, the effect of each observation node will not be multiply considered. The message passing scheme for Fig. 4.4 also applies to Fig. 4.5 except for step one.

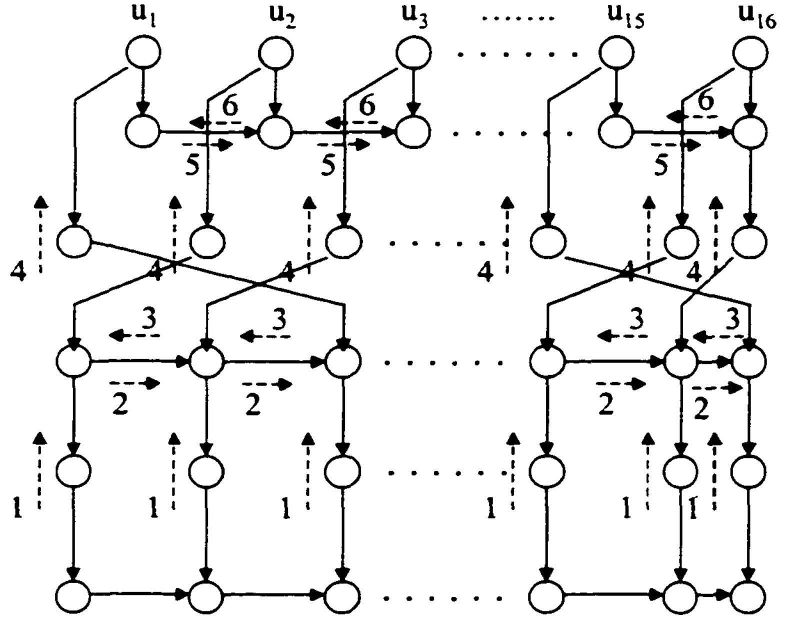


Fig. 4.5. Bayesian network for serially concatenated convolutional system, L=1.

The computation complexity of the iterative decoding algorithm for the correlated (more complicated if noise is signal dependent) noise channels compared with that for white Gaussian noise channels is made explicit by comparing the Bayesian networks in Figs. 4.4 and 4.5. Since the probability propagation steps two to six involve the same nodes, computations are exactly the same in these steps for the two Bayesian networks. We note that $\gamma_k(s',s)$ is a function of the noise sample added in y_k and $\gamma'(s',s)$ is a function of the noise sequence added to the codeword segment $\mathbf{y}_{k-L}^{k}$. By clustering the noise segment added to $\mathbf{y}_{k-L}^{k}$ into one node, the Bayesian network for the inner encoder with correlated channel noise can be converted to a singly connected structure, which is identical to the Bayesian network for white channel noise, Fig. 4.6. According to Section 4.3.2, the message passed from node Y_k to state $S_{2,k}$ is a vector of likelihood, and each element of the vector corresponds to a particular state of the sub-network. To calculate the likelihood, each configuration of the noise component in $\mathbf{z}_{k-L}^{k-1}$ has to be considered. Assume $Y_k \in \{y_1, y_2, \ldots\ldots, y_K\}$, the number of messages passing from Y_k in Fig. 4.6 is K^L times more than that for Fig. 4.3(c). Since each operation to calculate this message for the correlated noise channel involves the computation of vectors, the complexity of the decoding algorithm for a correlated noise channel is not practical, especially when K and L are large.

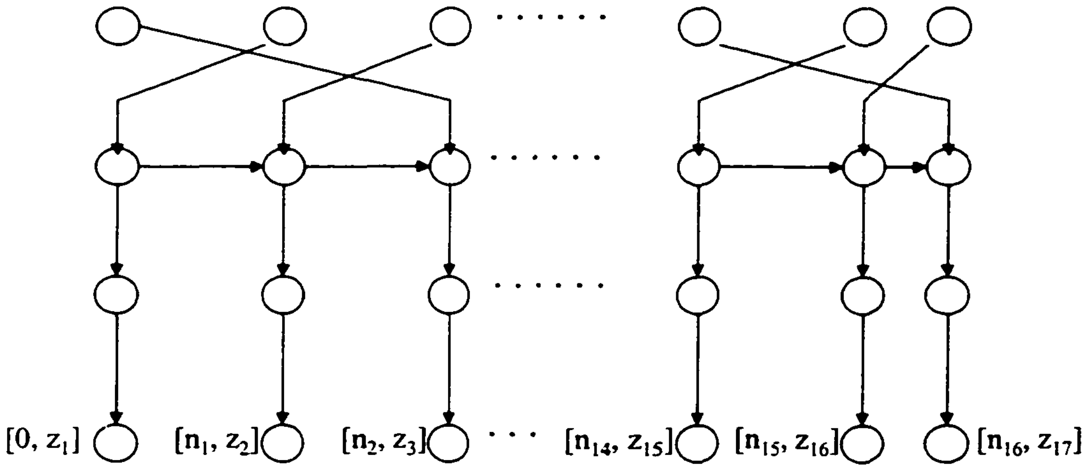


Fig. 4.6. Transformed Bayesian network for the inner encoder with correlated channel noise, Markov memory length $L = 1$.

4.5 Iterative decoding combined with iterative noise estimation

We have discussed that decoding for correlated noise channels is too complicated when channel noise memory (L) and noiseless channel output levels (K) are large. Using classical decoding algorithms, e. g., the BCJR algorithm or the soft Viterbi algorithm, is equivalent to ignore the states of the sub-network involving the previous L noisy channel outputs. This naturally motivated us to design sub-optimal iterative decoding algorithms considering the trade off between complexity and optimality. In [25], the optimal maximum likelihood sequence detector (MLSD) for correlated/signal dependent noise channels was derived and several suboptimal sequence detectors with lower complexity were discussed. Sub-optimal iterative decoding schemes can be derived by merging the branch matrix calculation techniques of the sub-optimal MLSD detectors with the BCJR algorithm. In this section, we will propose an iterative way, namely iterative noise estimation, to simplify the computation of the detector for correlated noise channels.

4.5.1 Iterative noise estimation concept

It was pointed out that the framework of realizations based on Tanner graphs seems to be an ideal tool to model the more complicated channels, in particular, channels with memory [23]. The idea of iterative noise estimation is similar to the idea proposed in [23].

A common way to simplify the detector is using decision feedback. The simplicity of current iterative detectors (compared to performance) is also a result of taking this approach, where each constituent decoder takes the extrinsic information of other decoders as the feedback. For a constituent decoder, the *a posteriori* probabilities of other constituent decoders can be obtained along the decoding iterations without much effort, and it also can be used to estimate the channel noise components. In this method, the conventional decoding algorithm is used in the first iteration, which ignores the correlation information. The decoding result of the first iteration is then used to estimate the desired information, such as channel state, channel symbols or noise component in the channel output. Through the estimate of signals and noise, the correlated/signal dependent Markov noise can be considered in the consecutive iterations. As the iteration proceeds, more and more accurate estimates can be obtained which in turn leads to more and more accurate decoding results. This technique takes advantage of the *a priori* knowledge of channel statistics in a similar fashion to the decoding algorithm of serially concatenated convolutional codes, where *a priori* information is not considered in the first iteration.

4.5.2 Noise predictive turbo systems (NPTS)

An example of the application of iterative noise estimation is the noise predictive turbo system [20]. As discussed earlier, y_k passes the likelihood of all the states of the sub-network given a particular value of node $S_{2,k}$. The truth is that the noise segment $\mathbf{n}_{k-L}^{k-1}$ has only one configuration in the observed channel output, but we have to consider all its possible configurations because this noise segment is not accessible before decoding. This leads to inefficient computations. To overcome this problem, NPTS simplifies the computation by estimating the true configuration of $\mathbf{n}_{k-L}^{k-1}$. According to Pearl's belief propagation algorithm, the *a posteriori* probability of each node can be updated during massage passing. It is necessary to update the *a posteriori* probability of each node of **Y** for noise predictive turbo systems because this probability will be used to calculate the optimal estimate of the true configuration of $\mathbf{n}_{k-L}^{k-1}$ conditioned on the observation.

After the estimation of $\mathbf{n}_{k-L}^{k-1}$, the calculation of the desired likelihood usually involves matrix operations. NPTS simplifies the computation further by using linear prediction techniques, which avoid matrix operations. The suitable Bayesian network for NPTS is identical to that for white noise channels except that the observation node is a combination of Z_k and the introduced parameter Θ_k, where Θ_k accounts for the influence of $\mathbf{n}_{k-L}^{k-1}$ on n_k.

Although NPTS is efficient, and its performance for the magnetic recording channel has been verified by computer simulations [20], it cannot be directly used for signal dependent noise channels. For signal dependent noise channels, the parameter Θ_k also depends on the symbols of permuted codeword (**U**, **P**), see [26] for details of the signal-dependent autoregressive channel model. In this case, denoting the data memory length as I [26], and denoting $\Psi(\mathbf{U},\mathbf{P})_{k-I}^{k}$ as the permuted version of $(\mathbf{U},\mathbf{P})_{k-I}^{k}$, the noiseless channel output can be estimated using the conditional expected value of the look-up table corresponding to $\Psi(\mathbf{U},\mathbf{P})_{k-l}^{k}$. To find the conditional expected value, the probability of the vector $\Psi(\mathbf{U},\mathbf{P})_{k-l}^{k}$ is needed, and it can be calculated from

$$P\left(\Psi(\mathbf{U},\mathbf{P})_{k-l}^{k} \mid \mathbf{Z}\right)= P\left((U,P)_{k-l} \mid \mathbf{Z}\right)P\left(\Psi(U,P)_{k-l+1} \mid \mathbf{Z}\right).....P\left(\Psi(U,P)_{k} \mid \mathbf{Z}\right), \quad (4.12)$$

where $P\left((U,X)_{k} \mid \mathbf{Z}\right)$ is the *a posteriori* probability of the symbols written on the channel. Once we have the estimated noiseless channel output, the signal-dependent noise can be estimated by simply subtracting the noiseless channel output from the noisy channel output.

4.6 Conclusions

Iterative detection techniques for various magnetic recording channels have been reviewed from the viewpoint of probability propagation, and the suitable graphical models for these channels were given. This review provides us an alternative approach to understanding the detection process for magnetic recording channels. In Section 4.5.1, the technique used in noise predictive turbo systems was generalized as the concept of iterative channel noise estimation. The possible application of this concept to signal dependent noise channels was discussed.

Chapter 5

Conclusions

5.1 Summary of the Dissertation

High performance and high complexity are the two features of iterative detection algorithms. It maybe difficult to implement these algorithms using today's technology, but advances in microelectronics will enable us to build soft iterative detectors in the near future. Compared to their excellent performance, the complexity of iterative detectors is low. Therefore, it is a promising approach to meet the demand for high performance detectors for very high recording densities.

The turbo coded ME^{2}PR4 serially concatenated system exhibits near 5 dB performance improvement over the uncoded PR4 system. In the simulation results, turbo EPR4 systems outperform turbo E^{2}PR4 systems. The possible reason is that different precoders were used for the two systems. Precoding significantly affects the performance of serially concatenated iterative systems.

Since iterative detectors are expected to work at very high densities, the white noise channel models are not realistic. The closed transitions of the magnetization patterns and equalization will lead to intolerable signal dependent and correlated noise. This indicates that the correlation and signal dependency of the noise should be considered in order to significantly improve the performance. Noise predictive turbo systems are proposed in this dissertation, which consider the correlation of the noise. The performance improvement of noise predictive turbo systems at channel density 3.18 is not significantly better than at channel density 2.86. The reason is that at very high densities, the system failed to accurately estimate the noise sequence. Recent results show that substituting the single convolutional code by a low-density parity-check code (LDPC) can achieve almost the same performance without turbo equalization [27]-[30]. This sheds light on the advantages of using iterative noise prediction combined with LDPC codes. With turbo equalization, the LDPC decoder provides a better estimate of the noise sequence. This better noise estimate can lead to improved performance.

It is said that Pearl's belief propagation algorithm is an alternative way to understand iterative detection. Chapter 4 gives the suitable Bayesian networks for a magnetic recording system for both additive white and correlated Gaussian noise channels. Through the viewpoint of probability propagation, the two decoding difficulties, e.g., large number of the configurations of the sub-networks (Section 4.4.2) and the matrix computation, for the correlated noise channels are made clear. Noise predictive turbo systems consist of two steps, namely the iterative noise estimation and prediction. These two steps are aimed at solving the corresponding two difficulties.

5.2 Future Research Directions

The idea of noise predictive turbo systems is completely described in this dissertation, but the understanding of it is far from conclusive because theoretical analysis of the noise predictive turbo systems is difficult. To overcome this problem, more research through computer simulations for the noise predictive LDPC systems is in order.

The error propagation issue is the fatal drawback of noise predictive maximum likelihood (NPML) systems [8]. Since error propagation is hard to control, it precludes this technique from many applications. Fortunately, the soft feedback prediction scheme (NPTS/SF) seems to have less error propagation problems than the conventional hard feedback prediction. This shows that the soft feedback prediction might be a practical technique. It is important to investigate the extent of the error propagation problem for soft feedback noise predictive turbo systems.

At very high recording densities, signal dependent noise is dominant. It is necessary to take into account the signal dependent noise for any good detection algorithm. The channel model in [26] is an accurate and simple model for designing practical detectors. In [26], signal dependent noise is modeled as a filtered value, which is a function of a channel input symbol segment. It is obvious that the *a posteriori* probability of each symbol and each symbol segment can be routinely obtained during turbo equalization. Therefore, the signal dependent noise can be similarly estimated by the conditional expected value. It is important to investigate the accuracy of this estimate. If it is accurate, the signal-dependent noise can be reduced by subtracting the estimate from the received channel output in each iteration.

References

[1] S. X. Wang, and A. M. Taratorin, *Magnetic Information Storage Technology*, San Diego, California: Academic Press, 1998, ch. 2, pp. 5-10.

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

[3] W. Ryan, L. McPheters, and S. McLaughlin, "Combined turbo coding and turbo equalization for PR4-equalized Lorentzian channels," in *Proc. Conf. Inform. Sci. and Sys.*, 1998, pp.489-493.

[4] J. W. M. Bergmans, *Digital Baseband Transmission and Recording*, Dordrecht, Netherlands: Kluwer Academic Publishers, 1996, ch. 3, pp. 105-150.

[5] P. H. Siegel, and J. K. Wolf, "Modulation and coding for information storage," *IEEE Comm. Mag.*, vol. 29, pp. 68-85, Dec. 1991.

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

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

[8] E. Eleftheriou and W. Hirt, " Improving performance of PRML/EPRML through noise prediction," *IEEE Trans. Magn.*, vol. 32, pp. 3968-3969, Sep. 1996.

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

[10] F. R. Kschischang, and B. J. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 1-11, Jan. 1998.

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

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

[13] 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. 1973.

[14] G. Bauch, H. Khorram, and J. Hagenauer, "Iterative equalization and decoding in mobile communications systems," in *Proc. 2nd EPMCC'97 and 3rd ITG-Fachtagung "Mobile Kommunikation"*, Bonn, Germany, Oct. 1997.

[15] S. V. Vaseghi, *Advanced Signal Processing and Digital Noise Reduction*. New York: Wiley, 1996, ch. 7, pp. 185-212.

[16] 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.

[17] Y. Wu, A. Raghunathan, R. D. Cronch, N. M. Bruner, and J. R. Cruz, "Performance of serially concatenated convolutional turbo codes for magnetic recording," *IEEE Trans. Magn.*, vol. 36, Sep. 2000.

[18] S. A. Altekar, "Detection and coding techniques for magnetic recording channels," Ph. D. Dissertation, University of California, San Diego, 1997.

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

[20] Y. Wu, and J. R. Cruz, "Noise predictive turbo systems," submitted to *IEEE Trans. Magn.*, 2000.

[21] G. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding: Turbo codes," in *Proc. Int. Conf. Commun.,* Geneva, Switzerland, May 1993, pp. 1064-1070.

[22] S. Benedetto, G. Montorsi, D. Divsalar, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design, and iterative decoding," *IEEE Trans. Inform. Theory,* pp.909-926. vol. 44, May 1998..

[23] N. Wiberg, "Codes and decoding on general graphs," Ph. D. dissertation, Linkoping University, Sweden, 1996.

[24] A. Kavcic and J. M. F. Moura, "The Viterbi algorithm and Markov noise memory," *IEEE Trans. Inform. Theory,* Jan. 2000.

[25] A. Kavcic, "Soft-output detector for channels with intersymbol interference and Markov noise memory," in *Proc. IEEE Global Telecommunications Conf.,* 1999, pp. 728-732 .

[26] A. Kavcic, and A. Patapoutian, " A signal-dependent autoregressive channel model," *IEEE Trans. Magn.*, vol. 35, pp. 2136-2138, Mar. 1999.

[27] 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.

[28] H. Song, J. R. Cruz, and R. M. Todd, "Low density parity check codes for magnetic recording channels," *IEEE Trans. Magn.*, vol. 36, 2000.

[29] J. L. Fan, A. Friedmann, E. Kurtas, and S. W. McLaughlin, "Low density parity check codes for magnetic recording," Submitted to *IEEE J. Select Areas Commun.*

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