## AN ABSTRACT OF THE THESIS OF

Dan Onu for the degree of Master of Science in

Electrical and Computer Engineering presented on March 7, 1997.

Title: Equalizer Design for MDFE Channels Using Nonlinear Optimization.

Abstract approved: Redacted for Privacy

John G. Kenney

Decision feedback equalization (DFE) is a sampled–data technique used for data recovery in digital communications channels. Multi–level decision feedback equalization (MDFE) has been developed for channels using the 2/3(1,7) RLL code.

The optimum detector for a digital communication channel affected by ISI and noise consists of a matched filter, followed by a symbol rate sampler and a maximum likelihood sequence estimator. The optimal detector is unrealizable for saturation recording channels. A compromise structure uses fixed filter types with adjustable parameters. The objective is to maximize the signal–to–noise ratio in order to minimize the error rate.

The read–channel waveform is corrupted at sampling instants by noise generated by various sources. We use a continuous–time low–pass filter cascaded with an all–pass filter at the receiver front–end. The low–pass filter band–limits high–frequency noise before sampling, and the all–pass filter equalizes the signal.

This thesis examines different structures of the receiver and their optimal parameter placing. A design methodology developed specifically for choosing the poles and zeros location of the linear front–end part of the receiver is presented. It makes use of nonlinear optimization, and a software package written in $MATLAB$ for equalizer computer aided design (CAD) is included in the appendix.

The optimization criterion usually mentioned in the literature for digital channel optimal design is the sum of the intersymbol interference and noise. A new objective function is proposed in the thesis, and the error rate probability is shown to decrease by 30%.

Issues pertaining to digital simulation of continuous-time systems are discussed. Design results are presented for different receiver structures, and bit error rate simulations are used for design validation.

# Equalizer Design for MDFE Channels Using Nonlinear Optimization

by

Dan Onu

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Completed March 7, 1997
Commencement June 1997

Master of Science thesis of Dan Onu presented on March 7, 1997

APPROVED:

Redacted for Privacy

Major Professor, representing Electrical and Computer Engineering

Redacted for Privacy

Chair of the Department of Electrical and Computer Engineering

Redacted for Privacy

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Redacted for Privacy

Dan Onu, Author

## TABLE OF CONTENTS

# LIST OF FIGURES

## LIST OF FIGURES (Continued)

# LIST OF TABLES

# ABBREVIATIONS

| | |
|---|---|
| APF | All–pass Filter |
| AWGN | Additive White Gaussian Noise |
| CAD | Computer Aided Design |
| DFE | Decision Feedback Equalization |
| FDTS/DF | Fixed-Delay Tree Search with Decision Feedback |
| FE | Forward Filter |
| FIR | Finite Impulse Response |
| ISI | Intersymbol Interference |
| LTI | Linear Time–Invariant |
| LPF | Low–pass Filter |
| MDFE | Multi-level Decision Feedback Equalization |
| ML | Maximum Likelihood |
| MLSD | Maximum Likelihood Sequence Detection |
| MSE | Mean–Square Error |
| PD | Peak Detection |
| PMF | Probability Mass Function |
| PW50 | Half Height Width of the transition response |
| RLL | Run Length Limited code |
| SNR | Signal to Noise Ratio |
| TF | Transfer Function |
| VA | Viterbi Algorithm |
| WMF | Whitened Matched Filter |
| ZFE | Zero Forcing Equalization |

# EQUALIZER DESIGN FOR MDFE CHANNELS USING NONLINEAR OPTIMIZATION

## 1. INTRODUCTION

The advent of the information age created an enormous demand for the storage of data in digital form. Over the last decade the bit capacity of mass–market hard–disks has increased almost 100–fold, due mainly to improvements in the read/write heads and magnetic materials [1], [2]. As advances in the non-electronics part of the data storage systems tend to become more and more costly and difficult to obtain, attention is being focused on read–channel ICs as a way to increase hard–disks capacity and throughput rates. Mixed–signal semiconductor technology, combined with sophisticated digital signal processing, promise improved detection capability in the face of diminished signal–to–noise ratio from the playback signal.

The disk reading and writing process is inherently a digital communication channel with a peculiar type of transmission medium. Various equalization and detection methods used for data transmission and detection were analyzed for mass storage applications [2], [15]. However, saturation recording has some differences relative to other types of digital data transmission channels, one of the most important being the two–sided impulse response associated with the magnetic media.

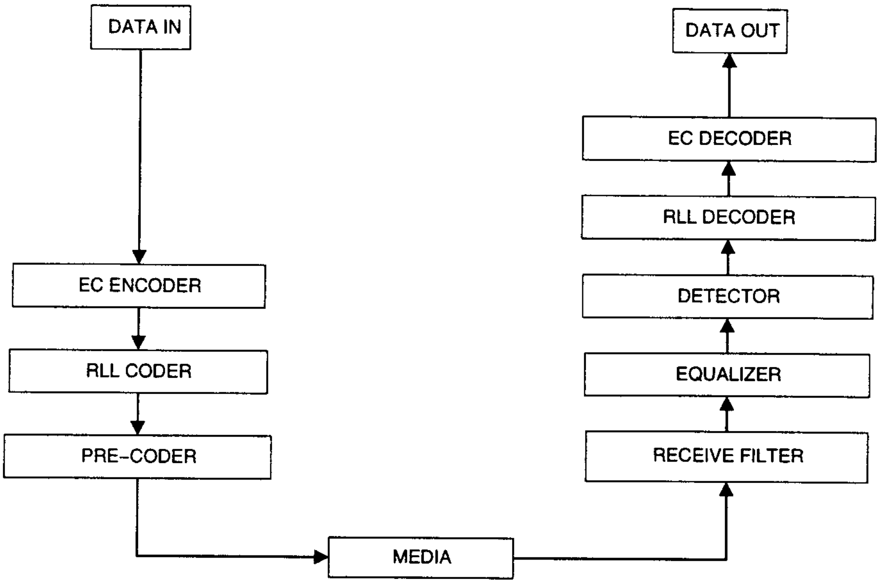

FIGURE 1.1. Block diagram of magnetic recording

Figure 1.1 depicts a block-level representation of the read and write process for magnetic recording. The Error Correcting (EC) encoder uses Reed–Solomon coding with interleaving to eliminate byte errors from the detector. The Run-length limited (RLL) coder limits the minimum and maximum number clock cycles between bit transitions, and improves timing information recovery. The pre–coder prevents catastrophic error propagation at decoding.

At the receiver front–end, a filter is employed to band–limit the high–frequency noise before sampling. Next, the channel is equalized for obtaining the best signal–to–noise ratio (SNR) at the decision time instance. At the detector level, decisions are made regarding recorded bits on the magnetic media. For MDFE the detector has a feedback system structure, with a simple comparator in the forward path. The RLL and EC decoders are employed for retrieving the data bits sent across the channel.

This thesis studies the usage of linear filters, of different types and orders, in the playback channel. The joint specification of a low–pass filter and an equalization filter found in the receiver front–end is typically based on maximizing SNR at the detector. Since the SNR at the detector is influenced by the front–end design, a very good understanding of the magnetic channel model is crucial. The characteristic waveform of the channel impulse response, from the pre—coder output to the input of the detector, is going to be designed using nonlinear optimization.

The read/write process on magnetic medium is illustrated in figure 1.2. The data is recorded using a write current at the write–head coil. The polarity of the input current is changed according to the data bits. At the write–head level, the symbols '1' are recorded as transitions by changing the polarity of the write current – this results in a change of direction of the magnetic domains along the track [11]. The read–head senses the flux transitions on the storage medium, and outputs a voltage signal.

The read/write process is nonlinear, but it is usually modeled using superposition, with linearity assumed. This makes possible to equate the digital magnetic recording channel with binary transmission of data over a linear communication channel [3]. The read–head output for two isolated transitions spaced a sample period apart is called the dibit response.

**Peak detection**: Peak detection has been widely used, because it could be implemented with high–speed analog circuits. The recorded bits can be retrieved by determining the peaks of the readback voltage. This is equivalent to finding the zero-crossings of the derivative of the playback signal, as depicted in figure 1.3.

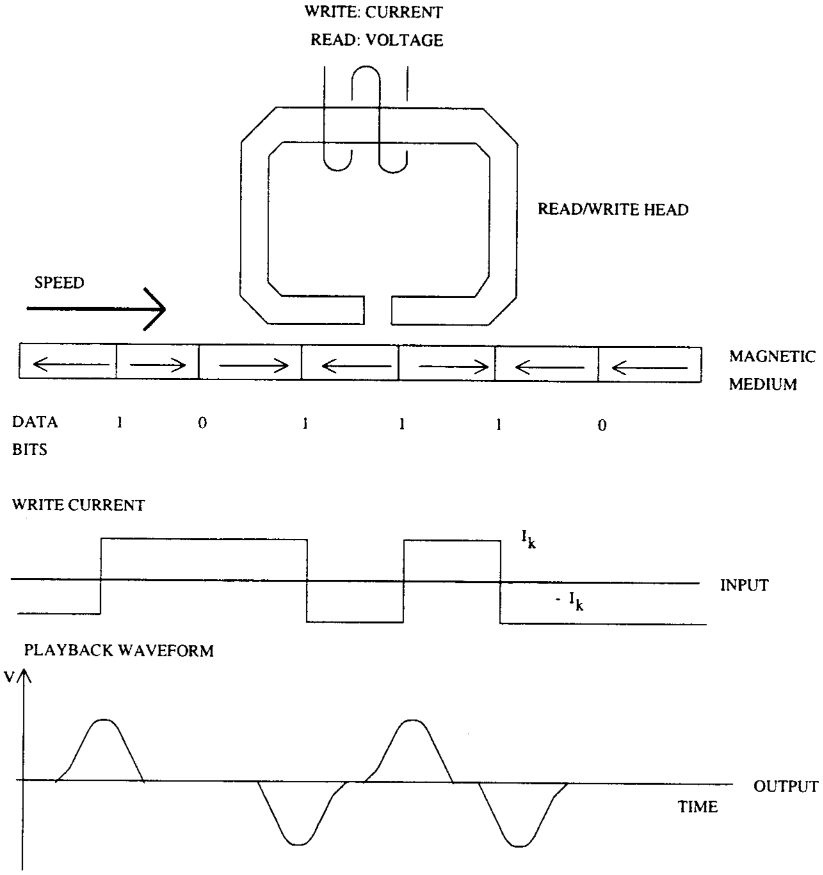

FIGURE 1.2. Read and write process in magnetic recording

Peak detection employs a full–wave rectifier and a threshold detector, as means to eliminate the noise amplification due to the differentiation operation [2]. As in many digital communications channels, the clock has to be recovered from the received signal. Constraints must be imposed on the maximum number of bits between transitions, so that the phase-locked loop can operate properly in random data mode. The minimum number of bits during the transitions is equally important since the data transmission over the magnetic medium creates a significant amount of intersymbol interference (ISI). For systems using RLL coding with a minimum–run-length constraint, the peak detection method renders a good compromise between ISI and noise [4]. At high bit densities, RLL coding cannot prevent

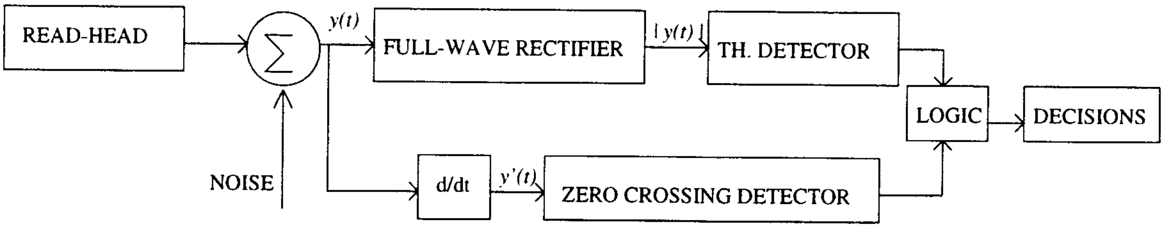

FIGURE 1.3. Peak detection

the onset of peak shifts or missing peak errors because of severe ISI. More elaborate detection techniques were developed to improve upon the performance of the peak detection.

**Maximum-Likelihood Sequence Detection (MLSD)**: MLSD can be implemented by using the Viterbi Algorithm (VA), yielding an optimal detector in the presence of ISI [4]. VA is essentially a dynamic programming method, and it was originally proposed by Viterbi in 1967 for convolutional codes [5]. Forney proposed in [14] an implementation of MLSD as a Viterbi decoder for channels with ISI and noise.

Even with the great reduction in complexity brought about by VA, MLSD with a whitened matched filter (WMF) is a demanding architecture such that it is not practical in many cases. The objective in MLSD is to minimize the Euclidean norm between a noisy sequence of observation and one for a sampled ideal output waveform. The length of the sequence upon which decisions are made is variable, and the output bits are produced in bursts.

As Moon and Carley summarized in [4], the noise–whitening discrete–time filter placed after the sampler is the optimum forward filter for both VA and DFE, as depicted in figure 1.4. At the output of the whitened matched filter, the noise, which is assumed to be additive white and Gaussian (AWGN), has again a white spectrum. It is worth mentioning that the matched filter has the impulse response

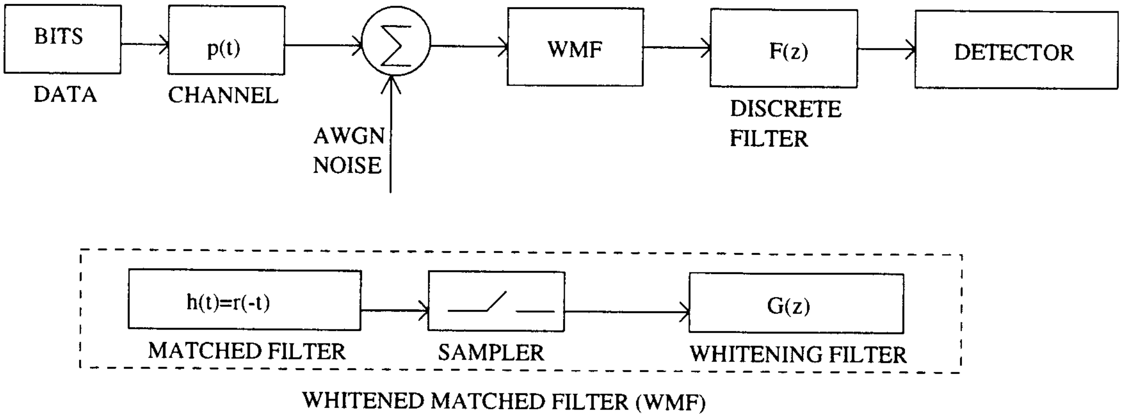

FIGURE 1.4. Communication channel and whitened mathed filter

as being the reverse in time of the impulse response of the channel; details regarding this subject are abundant in the literature, the book by Lathi being sufficiently detailed [6]. At the output of the WMF, a transversal filter is used. The whitening filter and the previously mentioned transversal filter are jointly referred as the linear equalizer (LE). The exact form of the transfer function of the LE depends on the method of detection used.

**Partial–Response Maximum–Likelihood (PRML)**: PRML employs MLSD on a channel previously equalized to a known impulse response. The partial response refers to the fact that the dibit response has more than one non–zero sample. In PR4+ML, see [3], the discrete transfer function of the equalized channel up to the ML detector is $1 - D^2$, where $D$ is the delay operator. The presence of ISI is obvious for PRML because of the $D^2$ term. As a benefit, the power of the signal at the VA is increased, and a higher density of the stored bits can be achieved when compared to peak detection. PRML is easy to implement in practice due to the simplicity of the equalized channel impulse response.

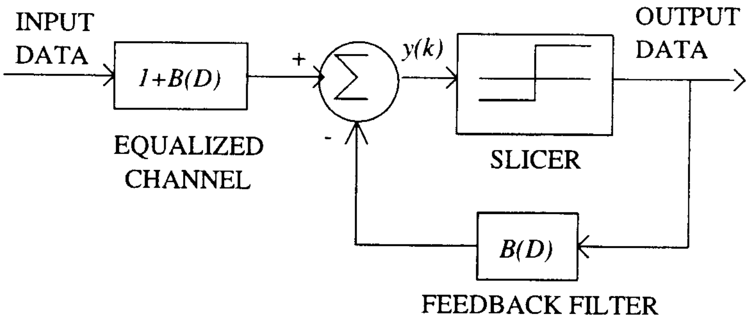

FIGURE 1.5. Block diagram of decision–feedback equalization

**Decision–Feedback Equalization (DFE)**: DFE, and its improvement MDFE for RLL channels, will be explained in more detail in Chapter 2. As a concept, DFE is an advanced method for coping with ISI. Past decisions are used to cancel the causal ISI. It allows more freedom in the specification of the receiver's forward filter, since a feedback filter is used for noiseless cancellation of causal ISI. A block diagram of the decision–feedback equalization method is given in figure 1.5. The combined transfer function of the entire channel is the Kronecker Delta function, thus allowing the decisions to be all correct in the absence of channel noise. In practice, causal ISI due to the finite number of taps in the feedback filter, and non-causal ISI affects the system. The DFE method does not try to achieve a zero–forcing equalization (ZFE), that is, it is not targeting a Dirac impulse as the impulse response of the equalized channel for a real implementation. One of the main advantages is that noise enhancement is less of a problem than for other detection methods.

**Fixed Delay Tree Search (FDTS)**: FDTS with decision feedback (FDTS/DF) uses DFE to cancel part of the ISI, and MLSD as a mean to perform sequence detection [4]. FDTS/DF utilizes more energy in making decisions than DFE, and is capable of working at higher symbol densities. A simpler imple-

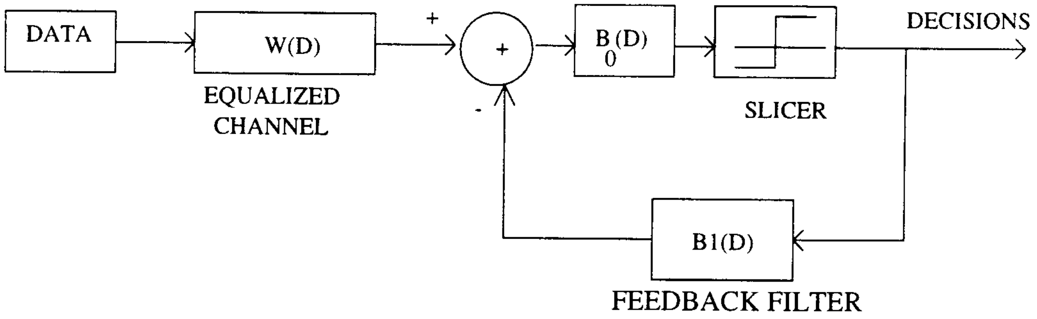

FIGURE 1.6. Block diagram of FDTS/DF

mentation of FDTS/DF can be found in [7], and it is shown in figure 1.6. The input of slicer contains a combination of transmitted data bits, and noise terms. If the depth of the tree–search is two, the MLSD part of the detector is still a challenging design for high speed implementation. It is also shown in [7] that placing the filter $B_0(D)$ in the forward path and feedback path leads to a MDFE structure of the detector. MDFE achieves the same performance as the FDTS/DF, with a simpler structure of the detector [7], [4], [9].

This thesis considers different structures for the forward filter of the receiver for MDFE, and compares the designs obtained by nonlinear optimization. In the next chapter the structure and signals characteristic to MDFE channel are described. Chapter 3 analyzes the noise and ISI characteristic to MDFE, states the criteria used in nonlinear optimization, and presents the design procedure. In Chapter 4, different designs obtained using *MATLAB* software are presented. The simulation results are interpreted and relevant comparisons are made. Chapter 5 contains conclusions. Commented *MATLAB* code written for nonlinear optimization is enclosed in the Appendix.

# 2. MULTI–LEVEL DECISION FEEDBACK EQUALIZATION – OVERVIEW

This chapter summarizes multi–level decision feedback equalization (MDFE) as a detection method used for data reading from magnetic channels. The underlying principle of this approach will be reviewed, and developments regarding efficient implementations will be mentioned. Kenney and Melas [10] showed that a MDFE channel can be equalized so that the operations in the critical path are spread over two symbols, compared to one for regular DFE. This recent improvement of the MDFE architecture spurred considerable interest in IC design for fast channels based on the feedback equalization concept.

## 2.1. Pre–receiver path

### 2.1.1. Signal shapes for magnetic media

The coded symbols $\{a_k\}$ form a stream of +1's and -1's to be recorded on the magnetic medium, where $a_k$ modulates the write current. The case considered in this thesis is for a 100 Mbits/s bit rate, which is equivalent to a 10ns time spacing between symbols at the write–head level. The read–process is characterized by the step response $s(t)$, see figure 2.1. It represents the response to a positive transitions, from -1 to +1, recorded on the disc. Physically, this is the voltage waveform seen at the read–head output as it passes over a point separating differently oriented magnets on the magnetic medium.

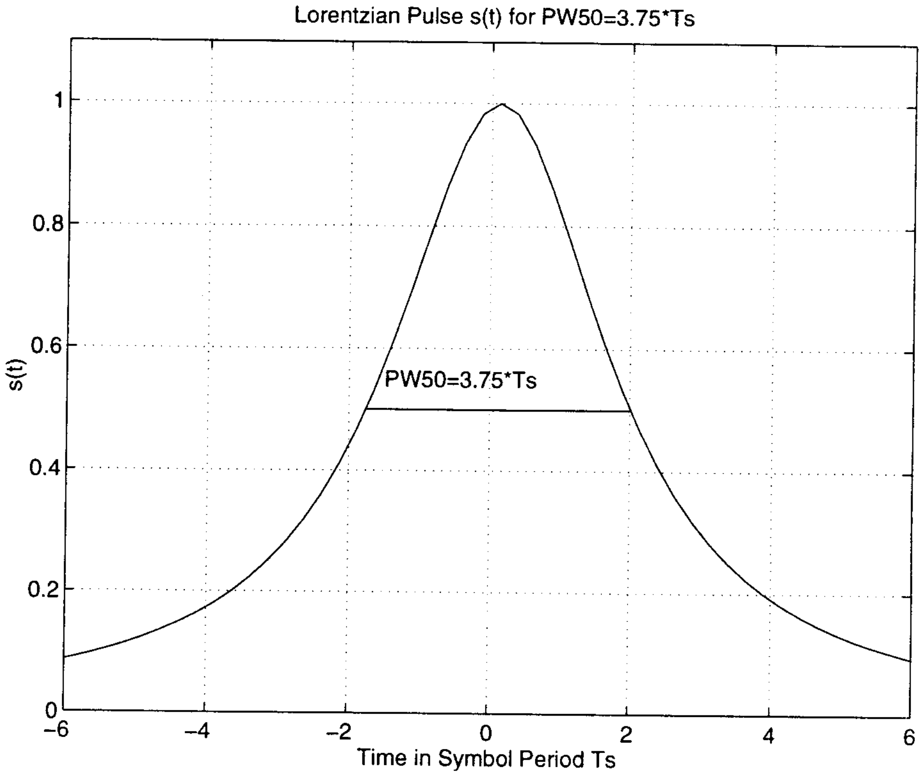

FIGURE 2.1. Step response of the magnetic medium

Usually, a magnetic recording channel is modeled as a linear system [7]. $s(t)$ is approximately symmetric in time, and it is a common practice to model it using a Lorentzian pulse

$$s(t) = \frac{A}{1 + (\frac{2t}{PW50})^{2\alpha}} \qquad 1 \leq \alpha \leq 1.5 \tag{2.1}$$

$A$ is the gain factor, and is it assumed to be unity for the rest of the thesis. $PW50$ is the half–height width of the transition pulse. $PW50$ represents a convenient way to specify the density of data on the hard–disk. $\alpha$ is a model parameter, which is frequently set to 1; a more accurate model of the transition response can be obtained by setting $\alpha > 1$ [2].

Although the step response is a starting point in system analysis, it is not the easiest to use in simulation and analysis. To evaluate the magnetic medium

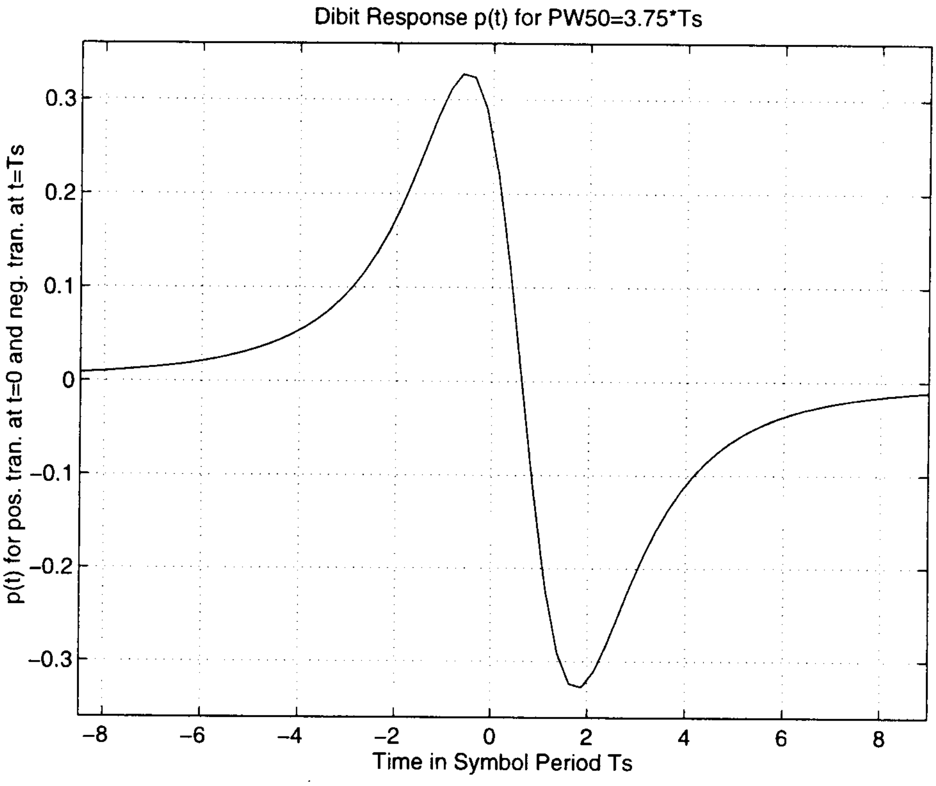

FIGURE 2.2. Dibit response of the magnetic medium

response to a data symbol pulse, the dibit response $p(t)$ is used in practice. $p(t)$ is obtained from $s(t)$ by a convolution with $1-D$, where $D$ is the unit delay. Denoting by $T_s$ the symbol period of the digital system, the dibit response is expressed as

$$p(t) = s(t) - s(t - T_s) \tag{2.2}$$

For $\alpha = 1$, the Fourier transform of the $s(t)$ given by formula 2.1 is

$$s(t) = \frac{A}{1 + (\frac{2t}{PW50})^2} \quad \longleftrightarrow \quad S(w) = \frac{A\pi PW50}{2} e^{-\frac{|w|PW50}{2}} \tag{2.3}$$

Using the shifting property of the Fourier transform, the value for $P(w)$ becomes

$$P(w) = S(w)(1 - e^{-jT_s w}) \tag{2.4}$$

The frequency content of a system exhibiting a dibit impulse response is presented in figure 2.3. The spectrum of the readback signal is bandpass characterized. As

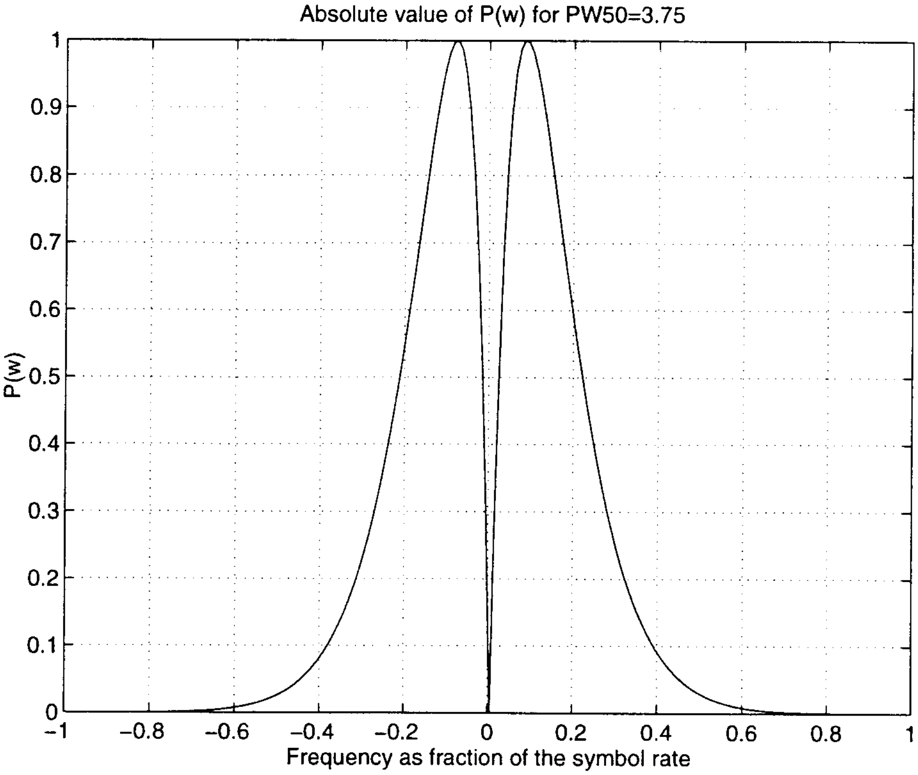

FIGURE 2.3. The frequency content of the dibit response

symbol density is increased, the spectral energy of the playback signal becomes concentrated at lower frequencies. This observation will be very useful in interpreting the design of the receive filter. Since the noise is considered AWGN and band–unlimited, the low–pass filter of the receiver should strike a compromise between the noise power and the intersymbol interference (ISI). Figure 2.3 suggests that cut–off frequencies of less than half of the symbol frequencies are conceivable for the case of a symbol density $PW50 = 3.75T_s$.

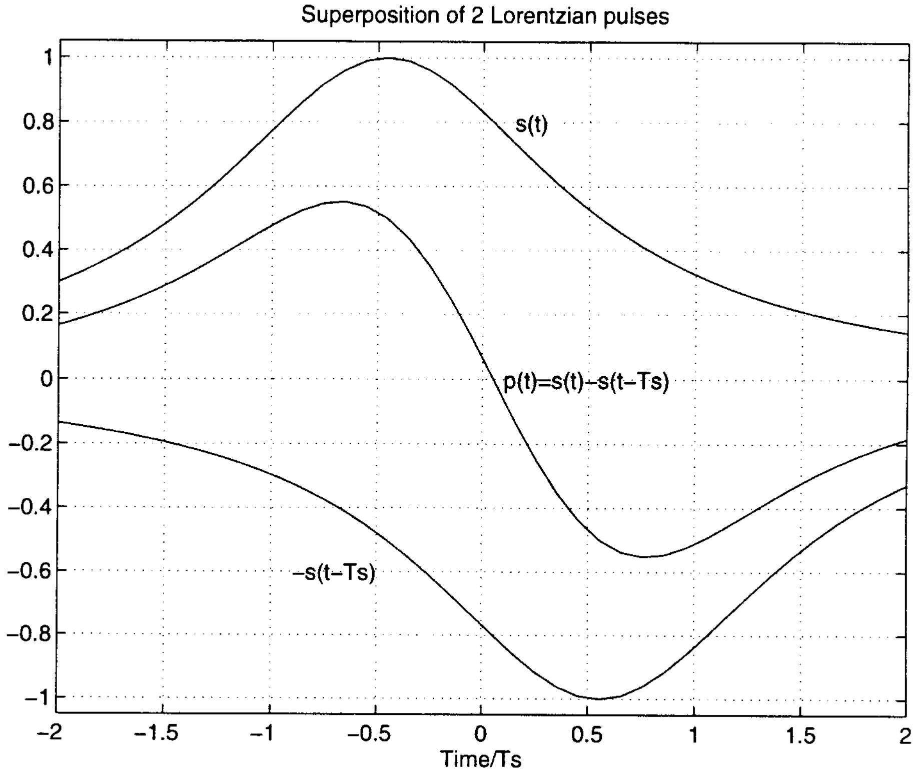

FIGURE 2.4. Pulse superposition in magnetic recording

### 2.1.2. RLL coding

One of the major problems in magnetic recording is the intersymbol interference (ISI). Figure 2.4 illustrates this effect by superposing two transitions. As depicted for a sequence of two consecutive isolated transitions, the neighboring transition responses (step responses) of the magnetic medium influence each other. This leads to reduced amplitude of the readback signal, and also causes peak shifts [11]. The obvious effect is reduced signal power, which directly translates into a higher bit error rate.

Coding methods were designed to generate more peak separation. An error correcting code (ECC) includes redundant bits in the transmitted signal so that decision errors can be detected and/or eliminated. The second approach is to use

| Data | Basic Code | Data in RLL Violation | Code | Violation Substitution Code |
|---|---|---|---|---|
| 0 0 | 1 0 1 | 0 0 - 0 0 | 1 0 1 - 1 0 1 | 1 0 1 - 0 0 0 |
| 0 1 | 1 0 0 | 0 0 - 0 1 | 1 0 1 - 1 0 0 | 1 0 0 - 0 0 0 |
| 1 0 | 0 0 1 | 1 0 - 0 0 | 0 0 1 - 1 0 1 | 0 0 1 - 0 0 0 |
| 1 1 | 0 1 0 | 1 0 - 0 1 | 0 0 1 - 1 0 0 | 0 1 0 - 0 0 0 |

TABLE 2.1. Look ahead technique for RLL(1,7)

recording codes (or modulation codes), which try to alleviate the effects of ISI by imposing constraints on the number of encoded bits between two adjacent transitions; these are run–length–limited codes (RLL). MDFE is designed for RLL channels, and a review of the RLL(1,7) code is given in the rest of this section.

The specification for a RLL code is RLL($d$,$k$), where $d$ is the minimum number of encoded bits between transitions, and $k$ is the maximum number of encoded bits between transitions. The direct effect of $d$ is to reduce ISI, while the parameter $k$ insures that there is enough timing information in the signal – if no transitions are written on the disk the signal will be 0.

The trade–off in using a RLL(1,7) code in MDFE is that the ratio of the number of user bits to the number of stored data bits is of 2:3. This is referred to as the code rate, and another way for specifying a RLL code includes it, e.g. 2/3(1,7)RLL code. The code rate gives information about disk space used in excess because of a recording code.

At an implementation level, RLL(1,7) uses look–ahead techniques; table 2.1 shows the finite state machine implementation of the encoder proposed by Cohn,

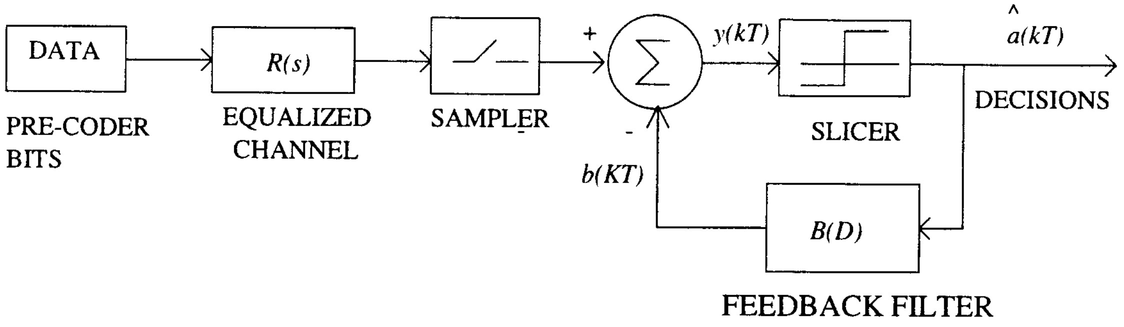

FIGURE 2.5. Block diagram of the MDFE receiver

Jacoby, and Bates [17]. The first two columns describe the basic encoding table. It is obvious that this simple encoding mechanism would not prevent violations of the RLL(1,7) code, the data sequence 00.00 being one of them [11]. The method to cope with it still uses a 2/3 code rate, and it is based on an increased horizon for the look–ahead technique. This is depicted in the third and fifth columns of table 2.1, and represents the violation substitution table. The last mentioned table is used when it is detected that the next two incoming bits form one of the four combinations unacceptable for the basic code table when combined with the present bits .

RLL(1,7) introduces inherent correlation between the encoded bits. This is useful in detection schemes, as they provide more information about the patterns of the signals at different points in the equalized channel. RLL(1,7) can be used for peak detection. For MDFE the constraint *d=1* is a requirement, and a performance enhancement.

## 2.2. The structure of the receiver

The block diagram of an MDFE receiver is the one from figure 2.5. The structure is identical to DFE with the difference being in the transfer function from

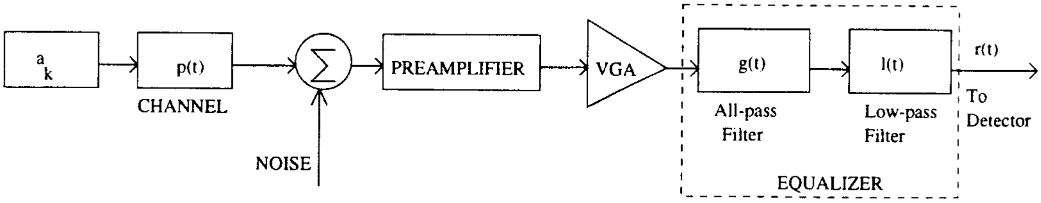

FIGURE 2.6. Forward path of MDFE

$a_k$ to the sampler, as depicted in figure 1.5. From an implementation point–of–view, an advantage of MDFE is that it can be implemented in the same structure as DFE. On a conceptual level, it is not suggested that the channel be equalized to ZF–DFE. The form of the signal at the slicer input will be presented in the next section. The forward filter, $R(s)$, is continuous–time for reasons of speed and complexity. The finite impulse response (FIR) filter 'B(D)' is usually digital. In more advanced implementations, the feedback filter has adaptive coefficients.

In this thesis, we consider the forward path of the MDFE receiver as shown in Figure 2.6. The read–back signal from the read–head is affected by gain and offsets. The purpose of the forward path of the MDFE is to amplify the input signal, filter it to remove high–frequency noise, and shape it to a form suitable for the detector. The variable gain amplifier (VGA) can be implemented in front of the detector, and the preamplifier is not relevant from the channel dynamics point of view. A system–level approach to the MDFE channel design is now considered. For these reasons, the forward path is equivalent to the equalizer (see figure 2.6).

The all–pass continuous–time filter $G(s)$ equalizes the incoming signal from the read–back circuitry. The transition response of the magnetic medium is symmetric in time and unravels a system exhibiting a two–sided response, i.e., the signal contains both causal and non–causal information. The dibit response also exhibits symmetry in time. Half of its energy is responsible for non–causal ISI, which cannot be canceled in MDFE. It was shown by McEwen and Kenney in [13] that using an all–pass equalizer makes it possible to obtain a minimum-phase response characteristic for equalized magnetic channels.

The low–pass filter $L(s)$ has its corner frequency as a parameter. The noise, assumed AWGN, has to be band–limited before it is sampled at the detector. In reality, little noise energy exists at high–frequencies, but it has to be removed in order to avoid aliasing it back into the signal bandwidth. For a maximally flat design a Butterworth filter can be used for implementing $L(s)$. However, this kind of filter has a nonlinear phase characteristic, and the amount of phase dispersion introduced can significantly hamper the benefits of equalization. Bessel filters can be employed instead, but the trade–off between fast roll–off of the magnitude response, phase distortion and filter order becomes readily apparent [1].

The signal at the output of the forward path, $r(t)$, is applied to the last stage of the MDFE structure. This block is a feedback system, and past decisions are weighted and fed back by the decision loop to compensate for the effect of the causal ISI. An advanced architecture for MDFE is presented in [10] by Kenney and Melas. Two feedback detectors are used operating in ping pong to double the overall symbol rate. The block diagram of this architecture is shown in figure 2.7. $B$ symbols represent analog buffers blocks, and the blocks labeled with $C$ are comparators. The signal $FeqOut$ is equivalent to $r(t)$.

The feedback filter should have of the order of 10 taps, although the figure 2.7 shows only 4. The intricate relationship between the specification of the forward filter and the optimization criterion also involves the number of taps of the FIR feedback filter. The number of taps for the feedback filter $B(D)$ has to be taken into consideration at the same design stage as the forward filter, since one of its main purposes is to compensate for forward channel dispersion.

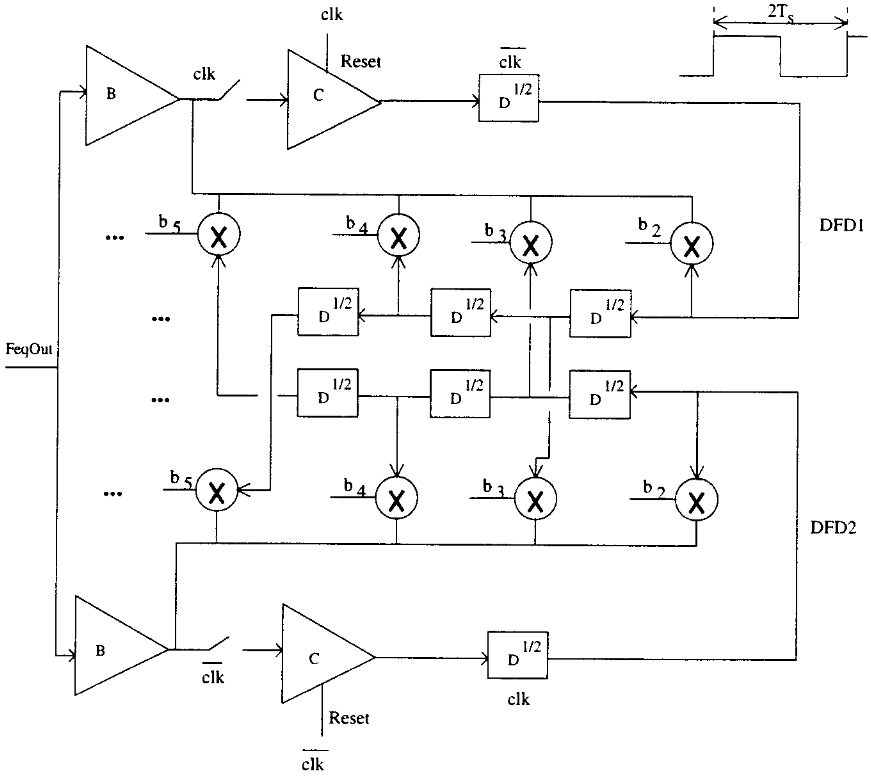

FIGURE 2.7. MDFE structure with two decision feedback detectors

## 2.3. Equalized channel characterization

In this section the MDFE block diagram from figure 2.5 is considered as a reference. It is assumed that the feedback filter $B(D)$ is a FIR filter, and thus the post–cursor (causal) ISI in only partially canceled. About 8 to 10 taps are usually used for $B(D)$ [10], and increasing the length of the feedback filter beyond this limit yields diminished returns in reducing ISI at the detector. A typical waveform of $r(t)$, the impulse response of the equalized channel, is given in figure 2.8.

In DFE, the first tap $b_1$ of $B(D)$ would be equal to $r_1$, and the rest of the taps would follow the relationship

$$b_i = r_i \qquad 1 \leq i \leq L \tag{2.5}$$

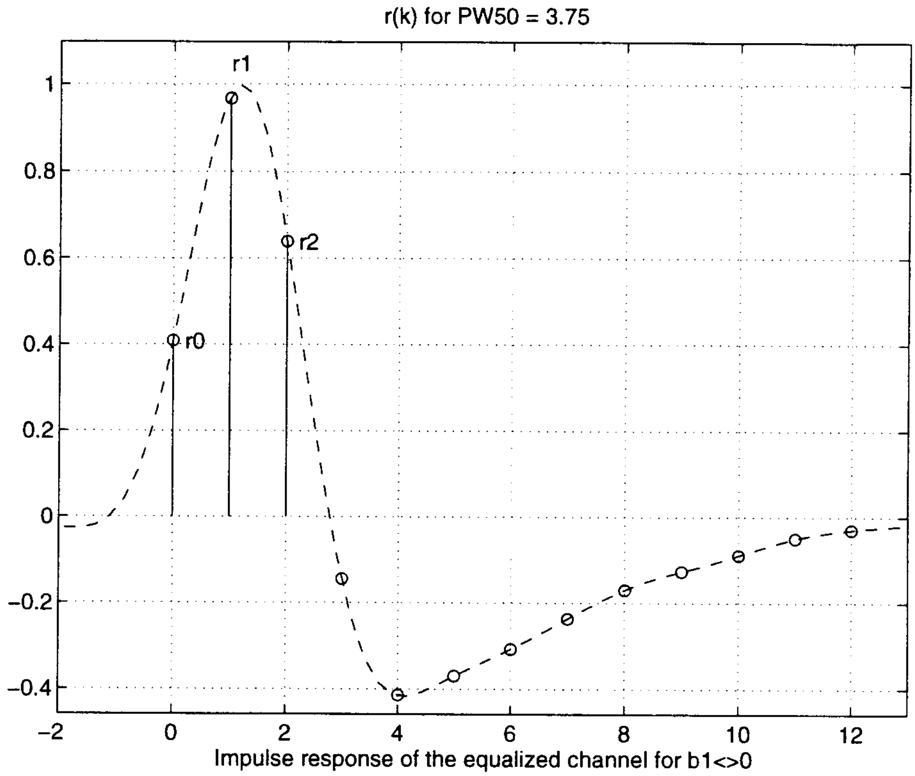

FIGURE 2.8. Equalized impulse response for MDFE for $b1 \neq 0$

where $L$ is the length of the feedback filter. The decisions in DFE are taken upon the $r_0$ as a main sample. In MDFE the SNR is increased at decision instants by allowing a controlled amount of ISI after feedback equalization [10]. The taps of the feedback filter are specified as $b_1 = r_2 - r_0$ and

$$b_i = r_{i+1} \qquad 1 \leq i \leq L \tag{2.6}$$

Assuming that the estimates $\hat{a}(kT_s)$ are correct, the signal $y_k = y(kT_s)$ takes the form

$$y(kT_s) = r_0(a((k+1)T_s) + a((k-1)Ts) + r_1 a(kT_s) + ISI_{nc} + n_c(kT_s) \tag{2.7}$$

Equation 2.7 assumes a simpler form for the case where both the uncanceled $ISI_{nc}$ and the colored noise $n_c(k)$ are 0.

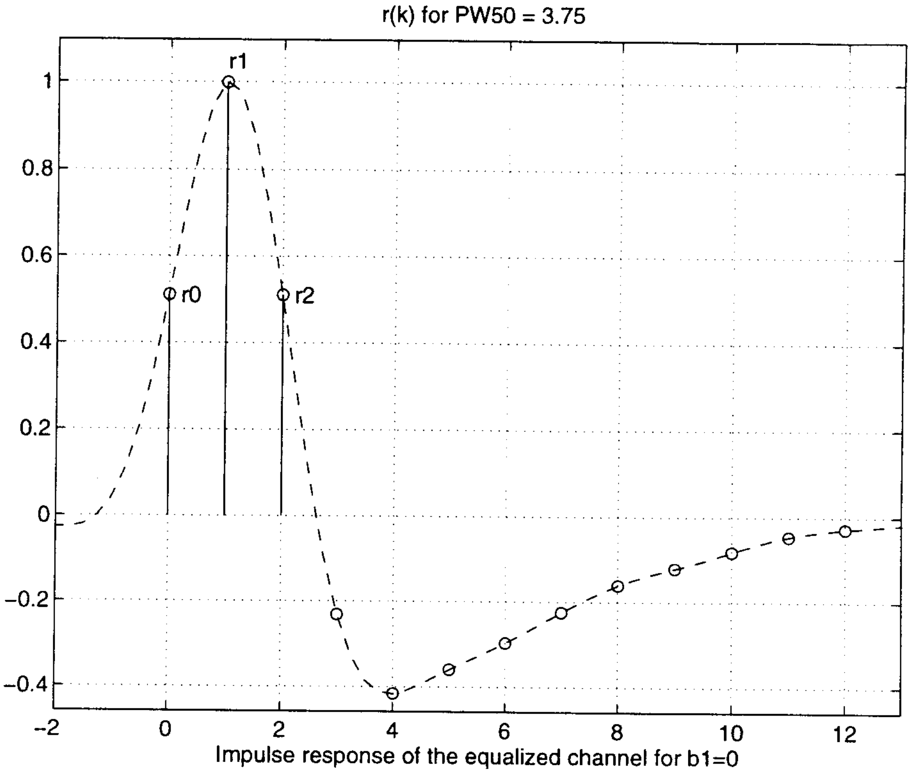

FIGURE 2.9. Equalized impulse response for MDFE for $b_1 = 0$

$$y_k = r_0(a_{k+1} + a_{k-1}) + r_1 a_k \tag{2.8}$$

The possible values of $y_k$ are restricted by the RLL(1,7) code. Sequences such as {-1,+1,-1} and {+1,-1,+1} are prohibited by the code, and they can not be encountered for the encoded sequence $\{a_{k-1}, a_k, a_{k+1}\}$.

In [10] Kenney and Melas showed that the setting $r_2 = r_0$ renders a feedback filter $B(D)$ with a null first tap coefficient. This allows an implementation of MDFE based on parallel detector, and the overall symbol rate is doubled for the same critical timing path. Figure 2.9 shows the impulse response of the equalized channel for the case when $b_1 = 0$.

The condition $r_2 = r_0$ is not restrictive in any sense because it can be satisfied at the level of phase–shift specification. During training and acquisition,

| $a_{k-1}$ | $a_k$ | $a_{k+1}$ | $y_k$ | Decision: $\hat{a}_k$ |
|---|---|---|---|---|
| +1 | +1 | +1 | $2r_0 + r_1$ | +1 |
| +1 | +1 | -1 | $r_1$ | +1 |
| -1 | +1 | +1 | $r_1$ | +1 |
| +1 | -1 | -1 | $-r_1$ | -1 |
| -1 | -1 | +1 | $-r_1$ | -1 |
| -1 | -1 | -1 | $-2r_0 - r_1$ | -1 |

TABLE 2.2. Allowable data patterns for MDFE

the recovered clock period is adjusted. The phase–locked loop (PLL) adapts the sampling phase so that the mean–square error (MSE) at the comparator input is minimized. Starting the system in a configuration where the first tap of $B(D)$ is set to 0 will ideally lead to the PLL triggering sampling at the phase–shift for which $r_2 = r_0$. In practice, a trade–off is made between pre–cursor and post–cursor ISI and a new error term $(r_2 - r_0)^2$ appears in the $ISI_{nc}$ expression.

The decisions $\hat{a}_k$ are made by a simple comparator. The four values that the input of the comparator can take are $\{-2r_0 - r_1, -r_1, r_1, r_1 + 2r_0\}$. These correspond to the sequence $\{-1, -1, +1, +1\}$ of the $\{a_k\}$ bits to be detected. A comparator threshold set at zero produces the correct $\hat{a}_k$.

The effect of equalization can be observed by comparing the waveforms from figure 2.9 and 2.2. Equalization has the effect of sharpening the leading edge of the channel impulse response, and a concentration of the signal power towards the origin. The signal is thus suitable at this point for feedback equalization using a small number of feedback taps. Usually 8 to 10 input taps are used in the transversal FIR feedback filter, including a DC offset cancellation tap.

# 3. EQUALIZATION FOR DIGITAL COMMUNICATION CHANNELS

An ideal communication channel has flat gain and linear phase over the Nyquist bandwidth. When the requirements are not satisfied, ISI is introduced. An equalizer can then be used to undo the effects of the gain and phase distortion.

ISI can be completely eliminated by a zero–forcing linear equalizer (ZF–LE). This implies a frequency characteristic of the equalizer as the inverse of the one of the medium, making the zero–ISI equalizer a simple inverse filter. The approach ignores the effect of noise altogether, and implies boosting high-frequency noise.

In digital communication channels the objective is to make correct decisions based upon the equalized signal. A ZF–LE is not the most robust equalizer for a transmission channel, and equalizers which minimize the sum of the ISI and noise were derived [15]. A simple threshold detector is employed in many cases, and consequently pulse dispersion can be tolerated up to some extent.

Noise energy reduction at the detector requires high–frequency attenuation. This creates significant ISI, the form in which dispersion affects digital data transmission. The design of an optimum equalizer is therefore a challenge, because it has to strike a fine compromise between the ISI and the noise interfering with the decisions.

The Nyquist 1-st criterion can not be satisfied for saturation recording [16]. This thesis is concerned with the equalizer design for MDFE channels. The equalizer is a compromise structure using fixed filter types with adjustable parameters. The overall equalized channel is nonlinear, since a threshold detector is used to provide input to a feedback filter implemented as an finite impulse response (FIR) filter. The advantages and disadvantages of nonlinear equalization over linear equalization structures are not so clear, but DFE can compensate for amplitude distortion with less noise enhancement than a linear equalizer [15].

### 3.1. Noise and intersymbol interference

The writing and reading of magnetic disks are usually modeled as linear processes. However, recording is based on fully-saturating the magnetic medium. A whole range of effects and component imperfections end up appearing as noise in the readback signal.

The noise in magnetic recording channels consists of media noise, crosstalk between tracks at high data densities, and electronic noise. Accurately modeling these noise sources and nonlinearities is not expedient at the design stage, and assumptions have to be made to prevent the loss of analytical tractability.

In this thesis we assume that the noise is additive, white and Gaussian (AWGN) with variance $N/2$. This implies that the noise power–spectral density is flat and has a value

$$S_n(w) = N/2 = \sigma_n^2 = constant \tag{3.1}$$

For simulation and analysis purposes, noise is injected at the output of the read–back channel, as in figure 3.1. The impulse response of the forward equalizer is given by the convolution of the impulse responses of the linear filters in the forward path

$$f(t) = l(t) * g(t) \tag{3.2}$$

The AWGN noise $n(t)$ is assumed to be wide–sense stationary, and the equalizer is a linear time–invariant (LTI) system. The colored noise at the output of the equalizer has the power-spectral density

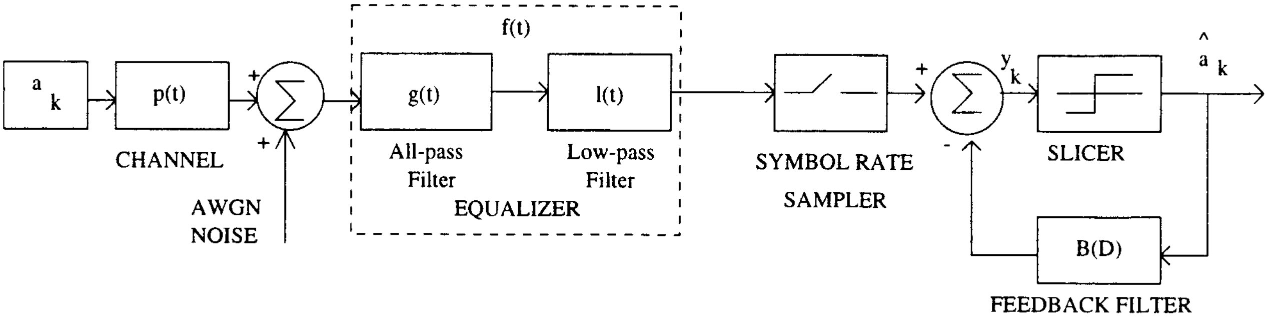

FIGURE 3.1. MDFE system structure used in design

$$\sigma_{nc}^2 = \frac{1}{2\pi} \int_{-\infty}^{\infty} \sigma_n^2 |G(jw)L(jw)|^2 \, dw$$
$$= \sigma_n^2 \int_{-\infty}^{\infty} f^2(t)\, dt \tag{3.3}$$

The channel simulation is done in discrete time. Noise is injected at the output of the read channel, and the SNR (dB) is computed according to

$$SNR_{MF} = 10log_{10} \left( \frac{\int_{-\infty}^{\infty} f^2(t)\, dt}{\sigma_n^2} \right) \tag{3.4}$$

Equation 3.4 defines the SNR at the output of the matched filter, as mentioned by Tyner and Proakis in [16].

The series $r_k$ consists of samples of the equalized channel impulse response. In equation 2.6 the length of the feedback filter $B(D)$ was specified to be $L$, implying the presence of post–cursor ISI starting with the forward–path channel impulse response sample $r_{L+2}$. Equation 3.5 defines $S_{ISI}$ as the set of indices corresponding to causal and non-causal ISI

$$S_{ISI} = \{k| -\infty < k \leq -1, L+2 \leq k < \infty\} \tag{3.5}$$

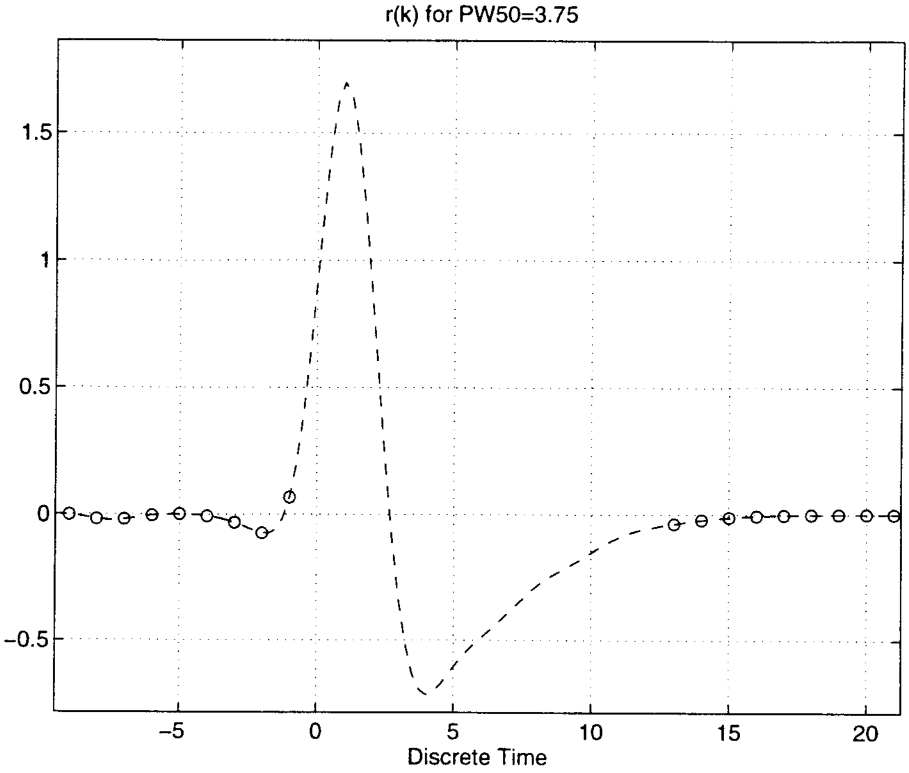

FIGURE 3.2. The set of causal and non–causal ISI terms

Figure 3.2 depicts the elements of the set $S_{ISI}$ for a finite time window. The absolute time value is of no importance in the derivation of the optimization criteria. It is assumed in the following that MDFE requires that $r_2 = r_0$ sets the discrete–time origin. For design purposes, the time index $k = 0$ corresponds to the absolute time moment $T_0$ specified by equation 3.6.

$$\begin{aligned} r_0 &= r(T_0) \\ T_0 &= \{t | r(T_0) = r(T_0 + 2T_s)\} \end{aligned} \tag{3.6}$$

Information about the value of $T_0$ can be obtained at the design stage, where a search algorithm is employed for finding the time–shift for which the two adjacent samples to the main–lobe sample are equal. The significance of the indices for the discrete–time samples of the channel impulse response are specified by equation 3.7.

$$r_k = r(T_0 + kT_s) \tag{3.7}$$

A search algorithm is employed to determine $T_0$ for a set of design parameters. The impulse response $r(t)$ is computed as the series $\{r_k\}$ by evaluation of $r(t)$ on a discrete–time grid. An oversampling ratio of $R$ is used, so that linear interpolation can be employed for finding $T_0$ with sufficient accuracy. The search algorithm is briefly described as follows.

1) Locate the index $k_R$ of the oversampled signal so that $r(k_R T_s/R)$ is maximum.

2) Recursively adapt $k_R$ according to equation 3.8

$$k_R = k_R + \beta(r(k_R + R) - r(k_R - R)) \tag{3.8}$$

until

$$\mid r_2 - r_0 \mid = \mid r(k_R + R) - r(k_R - R)) \mid \leq tol \tag{3.9}$$

( where $\beta$ is a small positive constant chosen so that the search algorithm can render a specified tolerance *tol* ).

The ISI at the input of the slicer has the form

$$y_{ISI} = \sum_{k \in S_{ISI}} r_{k+1}\, a_{-k} \tag{3.10}$$

The mean–square ISI due to uncanceled pre–cursor and post–cursor ISI is given by

$$ISI = \sum_{k \in S_{ISI}} {r_k}^2 \tag{3.11}$$

Let $E$ be the sum of the ISI and colored noise $n_k^c$

$$E = y_{ISI} + n_k^c \tag{3.12}$$

Prediction of error rates in digital communication channels is based upon assumptions regarding the statistics of noise affecting the input of the detector. Decisions are error free in the case where no ISI and noise are present. The optimization criterion usually used for optimal digital channel equalizer design requires the minimization of the sum between ISI and noise [16], [15]. A normalizing factor $\frac{1}{r_1}$ is employed because it makes unity the sample of the main lobe of the equalized channel impulse response.

The noise and ISI are independent random variables, due to the fact that the input data stream and the AWGN noise are independent. Taking into consideration the scaling, the variance of the error term affecting the detector input is

$$\sigma_E^2 = \sigma_{ISI}^2 + \sigma_N^2 \tag{3.13}$$

where

$$\sigma_{ISI}^2 = \frac{ISI}{{r_1}^2} \tag{3.14}$$

and

$$\sigma_N^2 = \frac{\sigma_{nc}^2}{{r_1}^2} \tag{3.15}$$

For nonlinear optimization via software computation, the formula applied in the calculation of the noise power is given by equation 3.16.

$$\sigma_N^2 = \frac{\sigma_n^2}{{r_1}^2} \sum_{k=-\infty}^{\infty} f_k^2 \tag{3.16}$$

Based on previous notation and assumptions, the design objective is to maximize the ratio

$$SNR = \frac{1}{\sigma_N^2 + \sigma_{ISI}^2} \tag{3.17}$$

Equation 3.17 offers an alternative way of specifying the SNR at the detector input.

## 3.2. MDFE nonlinear equalization

### 3.2.1. Optimization Criteria

The goal of the equalizer design is to reduce the probability of error at the decision time instance. The decision process in MDFE is affected by errors because of the ISI, noise and feedback error propagation. In equalizer design, the error propagation phenomenon is difficult to model and take into consideration. The usual objective is the minimization of the sum of ISI and noise.

The decision process would be error free for the case

$$| E |< r_1 \tag{3.18}$$

since the scaling of $r_1$ to unity was employed. Limiting the perturbing signal $E$ to absolute values less than one guarantees error free decisions.

The noise and ISI are independent random variables. Since the noise is AWGN, a fast way of estimating the probability of error $P_e$ is to assume $y_{ISI}$ to be Gaussian distributed. $E$, being Gaussian distributed, is the sum of 2 independent Gaussian random variables, and has variance $\sigma_E^2$.

For this case the estimate of the bit error probability is

$$\begin{aligned} P_e &= P[E > 1] \\ &= \frac{1}{\sqrt{2\pi}\sigma_E} \int_{x=1}^{+\infty} e^{-\frac{1}{2}\left(\frac{x}{\sigma_E}\right)^2} dx \end{aligned} \tag{3.19}$$

The previous result can be expressed in terms of the $Q$-function

$$Q(y) = \frac{1}{\sqrt{2\pi}} \int_{y}^{+\infty} e^{-\frac{1}{2}x^2} dx \tag{3.20}$$

to obtain

$$P_e = Q_e\left(\frac{1}{\sigma_E}\right) \tag{3.21}$$

Equations 3.13 and 3.21 relates the bit–error probability to the minimization of the sum of the ISI and noise. The expression of the objective function for this criterion is given by equation 3.13.

Assuming that the ISI has a Gaussian distribution provides for a simple objective criterion and quick error estimates. This is very conveniently mathematically, but would not yield an optimal design or an accurate error rate estimate.

$y_{ISI}$ is a discrete random variable, and statistical methods can be employed for computing its probability mass function (PMF). Let $S_Y$ be the set, finite or countable, of values taken by the discrete random variable $y_{ISI}$. $S_Y$ can be generated starting from equation 3.10 by exhausting the allowed combinations of input symbols $a_k$.

$$S_Y = \{s_i \mid P(y_{ISI} = s_i) \neq 0\} \tag{3.22}$$

It is assumed that the autocorrelation of the input data stream $\{a_k\}$ is the Kronecker Delta function $\delta_k$.

$$R_{a_k a_k} = \delta_k \tag{3.23}$$

$R_{a_k a_k}$ expression is not consistent with the RLL constraint, but provides for analytical tractability.

For numerical estimation of the PMF of $y_{ISI}$, it is assumed that $S_{ISI}$ is finite. Let

$$S_{ISI} = \{j_0, j_1, ..., j_{M-1}\} \tag{3.24}$$

where $j_k$'s are indices corresponding to causal and non–causal ISI. $y_{ISI}$ is, according to equation 3.10, a sum of random variables $\{a_k\}$, weighted by the set of coefficients

$$S_R = \{r_{j0}, r_{j1}, ..., r_{jM-1}\} \tag{3.25}$$

The symbols $\{+1\}$ and $\{-1\}$ are equally probable

$$P(a_k = +1) = P(a_k = -1) = \frac{1}{2} \tag{3.26}$$

With a probability of one, no 2 elements of the set $S_R$ have the same absolute value. The set $S_Y$

$$S_Y = \{s_i | 1 \leq i \leq 2^M\} \tag{3.27}$$

has than $2^M$ elements. The set $S_Y$ can be constructed according to the equation 3.28,

$$s_i = \sum_{k=0}^{M-1} r_{j_k}(2b_k - 1) \qquad 1 \leq i \leq 2^M \tag{3.28}$$

where

$$i_2 = b_{M-1}b_{M-2}...b_0 \tag{3.29}$$

is the binary representation of $i$ ($b_0$ is the least significant bit). The set $S_Y$ has to be refined, based on the RLL constraints. Not all the combination of coded bits are allowed. $S_Y$ is reduced to a length of $2^{M_{RLL}}$ elements, $M_{RLL} < M$, in a stage which is software implemented.

The PMF of the random variable $y_{ISI}$ is

$$P(y_{ISI} = s_i) = \frac{1}{2^{M_{RLL}}} \tag{3.30}$$

$E$ is a mixed random variable, and the probability of error is computed as

$$\begin{aligned} P_e &= P[E > 1] \\ &= \sum_{y_{ISI} \in S_Y} P(y_{ISI} = s_i) P(n_k^c \geq 1 - s_i) \end{aligned} \tag{3.31}$$

A comparison will be made in this thesis between the usage of the two mentioned criteria. Minimization of $\sigma_E^2$ is easier to implement, but the computational power available fully justifies optimizing directly for the probability of error given by equation 3.31. Numerical simulation in Chapter 4 show that the bit error rate can be reduced by using the objective function specified in equation 3.31.

### 3.2.2. Design Procedure

In [14] it is shown that the optimal demodulator for a digital communication channel affected by ISI and noise consists of a matched filter, followed by a symbol rate sampler and a maximum likelihood sequence estimator (MLSE). The computational complexity of the VA–based MLSE increases exponentially with the length of the channel response $p(t)$ [16]. Obviously, the infinite length of $\{p_k\}$ makes this approach unrealizable for the case of magnetic recording.

A solution to designing the receiving filter is a sub–optimal approach, due to assuming a particular filter form and adjusting its parameters to maximize the SNR at detector.

The structure of the forward equalizer for which the design optimization is done in this thesis consists of an all–pass filter (APF), cascaded with a low–pass filter (LPF). The design parameters are the pole/zero of the APF, and the type and corner frequency of the LPF.

A symbolic approach to the equalizer design for the MDFE read channel was found not to be feasible. The transfer function of the channel up to the summing node has an order higher than 3. The analytical expression of the impulse response $r(t)$, as a convolution product, is a sum of weighted exponentials. The objective function given by equation 3.13 does not have a close–form solution for its minimum in terms of the design parameters.

*MATLAB* routines were written to facilitate the equalizer design for MDFE channels. The software was organized so that the design process is highly automated. A valuable computer aided design (CAD) tool was thus created, providing a unified framework for design and testing. The software can be easily extended to account for other structures of the forward equalizer.

*MATLAB* is a high–performance, interactive software package for scientific and engineering computations. This complete integrated system was chosen as a programming environment because it provides proven tools in numerical analysis, matrix computation, signal processing and graphics. The developed CAD programs tap the generous facilities offered by *MATLAB* in system analysis and simulation.

The programs developed for nonlinear optimization make usage of the *MATLAB* function *'constr'*, which finds the constrained minimum of a function of several variables. This allows a greater flexibility in extending the number of design parameters, and it speeds up the search for the best design. For a starting point, an extensive search over a grid as dense as possible was used.

The *'constr'* function approximates the *Lagrangian* with a quadratic function with linear constraints. It solves a quadratic program (QP) to obtain new estimates of the Lagrange multipliers. The Hessian matrix is computed by successive application of Quasi–Newton approximations. The convergence is super–linear (quadratic) in a small vicinity of the solution, and the method is known in the literature as SQP [18].

The results of the constrained optimization for the MDFE equalizer does not depend on the method of optimization employed. The objective is to find the optimum design parameters, which can be done using other numerical minimization techniques for a starting point in the vicinity of the optimum. The *MATLAB* function 'constr' was employed because its availability in the *MATLAB* optimization toolbox. Since the design is done off line, it is not sensitive to a nonlinear minimization algorithm which is not optimized for fast convergence.

Two criteria are used for optimization, with the objective functions $Criterion_1$ and $Criterion_2$.

$$Criterion_1 = \sigma_E^2 = \sigma_{ISI}^2 + \sigma_N^2 \tag{3.32}$$

$Criterion_1$ was first mentioned in equation 3.13, and it is assumed that the dependence of the probability of error to $\sigma_E^2$ value is monotonic, and given by the $Q$ function according to equation 3.21.

$$Criterion_2 = P_e = \sum_{y_{ISI} \in S_Y} P(y_{ISI} = s_i) P(n_k^c \geq 1 - s_i) \tag{3.33}$$

$Criterion_2$ was first mentioned in equation 3.31, and it estimates directly the probability of error via PMF calculation.

Figures 3.3 and 3.4 depict the inverse of objective function around the optimum, for an equalizer consisting of a first–order all–pass filter and a fourth–order Butterworth low–pass filter. Details about these designs make the object of Chapter 4. It is important to remark here that both criteria exhibit local minima, and the off–line optimization algorithm can be speed–wise improved by using custom developed optimization methods.

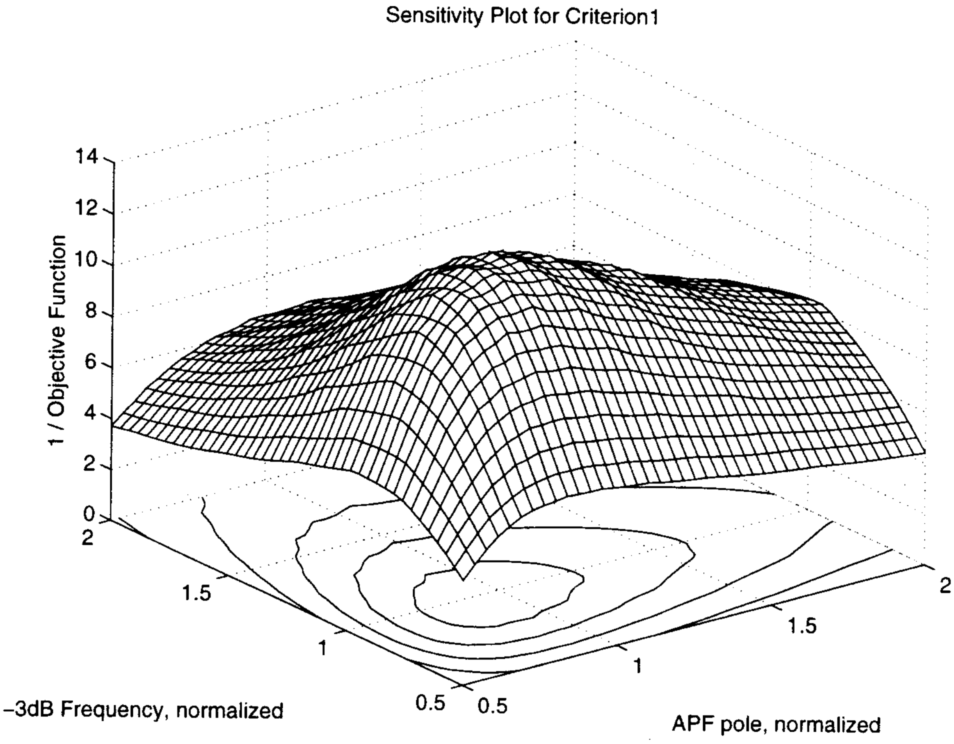

FIGURE 3.3. Sensitivity plot for *Criterion*$_1$

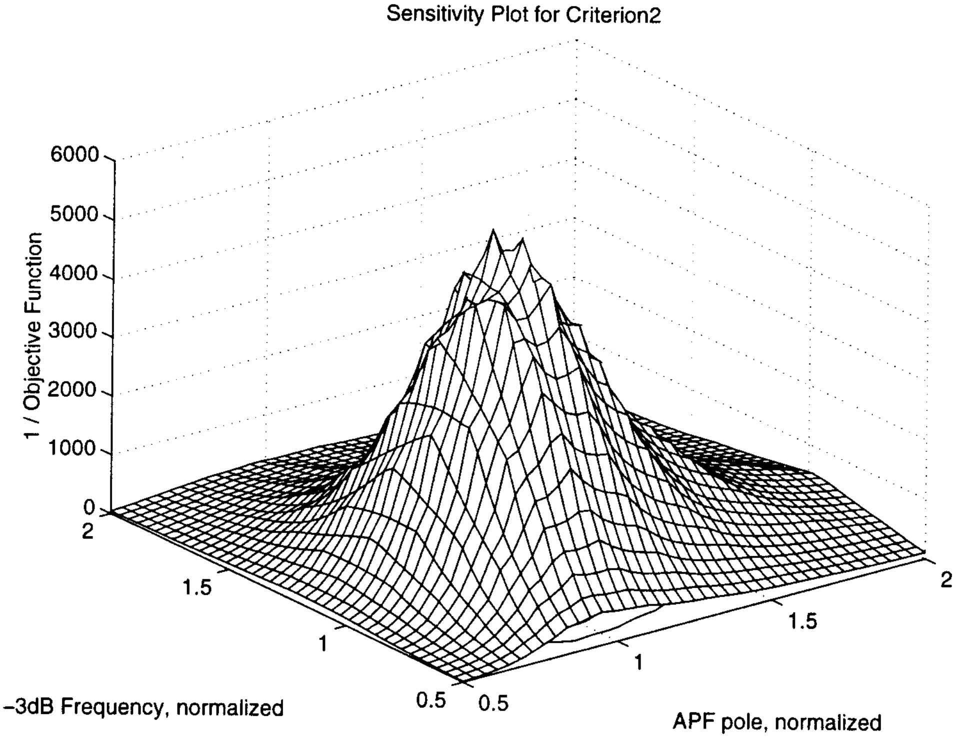

FIGURE 3.4. Sensitivity plot for *Criterion*$_2$

The higher level *MATLAB* program used in optimization is called *'srch.m'*. It provides options for the specification of different orders of the APF, and filter types and orders of the LPF. *'srch.m'* allows the easy specification of filter design parameters, as shown in the following

```
                    DESIGN PARAMETER SPECIFICATION
MAIN()
BEGIN
    INPUT_OSR;          % Oversampling ratio used in design
    INPUT_PW50;         % Half-height of the Lorentzian pulse
    INPUT_WINDW;        % Time-window width (in symbol periods) used in
                        %  truncating the infinite-long Lorentzian pulse
    INPUT_SNR;          % SNR at the read-channel output
    INPUT_LPFtype;      % Butterworth or Bessel LPF
    INPUT_LPFOrd;       % Order of the LPF
    INPUT_APFOrd;       % Order of the APF
    ...
END
```

Estimating numerically the objective functions corresponding to the 2 optimization criteria poses problems regarding the numerical accuracy of the result. The forward equalizer (FE) is a continuous–time filter (CTF), implemented as a cascade of different linear stages. The samples of its impulse $r(t)$ can be numerically evaluated using 2 main approaches, denoted by $M1$ and $M2$:

$M1$) The design parameters search is done in the discrete–time domain. The FE overall transfer function (TF) is computed by multiplication of the sub–

block TFs, and the *MATLAB* function *'dimpulse'* is used to compute the series $\{r_k = r(kT_s) \mid k \in Z\}$.

$M2$) The design parameters search is done in the continuous–time domain. The FE overall TF is obtained by multiplication of the continuous–time transfer functions of the FE sub–blocks and the *MATLAB* function *'impulse'* is used to compute the samples of $r(t)$ for the time series $\{kT_s \mid k \in Z\}$.

The $M1$ approach is the faster one because the search is performed entirely in discrete–time. However, extracting the specification in the continuous–time usually renders a TF which has a different order of the numerator when compared to the discrete–time equivalent. It is difficult to impose at design inception the structure of the TF of the forward equalizer, and this is obviously a drawback.

$M2$ has the advantage that the conversion to discrete–time is postponed as much as possible, and thus the numerical accuracy of the result is the highest obtainable using *MATLAB*. This computational approach was used for the nonlinear optimization software developed.

Describing the way in which the nonlinear optimization is performed can be done by tracing the nested function calls for the optimization program *'srch.m'*; the pseudo–code is included in the followings.

A problem in design is in estimating the relevant time–span over which the impulse response $r(t)$ has to be computed. The usage of the design method $M2$ previously mentioned offers the chance of highly automating this task based on *MATLAB* function *'impulse'*. When invoked with no time arguments, an estimate of the sampling period and of the relevant time–window is computed, so that the impulse response is accurately represented. This information is obtained based on estimates of eigenvalues of the analyzed system, and possible simplifications of poles/zeros. For each set of design parameters, the finite length series $\{r_k\}$

## FUNCTION CALLS FOR DESIGN OPTIMIZATION CAD

```
MAIN()
BEGIN
    INPUT_design.parameters;     % Design parameters are specified.
    INPUT_design.bounds;         % Sets the design parameters' bounds.
    SET_numerical.options;       % Sets various numerical options for
                                 % the numerical minimization alg.
    CALL('constr');              % Call to MATLAB function
     BEGIN
       WHILE(crit_stop=FALSE)    % Iterative criterion estimation
         EVAL(impulse resp.);    % Evaluate the equalized channel
                                 %  impulse response r(kTs)
         EVAL(criterion);        % Evaluates the objective function
                                 %  based on r(kTs) and SNR
       END                       % End WHILE
     END                         % Constrained minimization ended
    DISPLAY_results;            % displays optimization results
    SAVE_design;                % stores a design in a file
END
```

is computed as described by the pseudo–code for estimation of the channel impulse response.

Some observation have to be made with respect to the approach for estimating the probability of error in the MDFE channel.

```
                 ESTIMATION OF THE CHANNEL IMPULSE RESPONSE
ROUTINE CT_Filter()
BEGIN
    INPUT_design.parameters ;  % Design parameters specification
    EVAL_TF;                   % System TF eval. by multiplication
    T_max=impulse(TF);         % Obtain the pulse time-window width
    t_vect=0:Ts/R:T_max;       % Compute the sampling time-grid
    r_vect=impulse(TF,t_vect); % Estim. the channel impulse response
END
```

First, it should be pointed out that regardless of the method used, approximating $P_e$ via the $Q$–function or via the PMF estimates implicitly assumes that no error–propagation occurs. This process, characteristic to a feedback decision system, is difficult to model or estimate at the design stage. The cost due to this effect is in the order of fractions of dB, and numerical examples will be provided in the next chapter.

Secondly, the design is conservative since it does not consider the case of the sequences {-1 -1 -1} and {+1 +1 +1}. These cases, arising with a probability of 1/3, provide for a signal decision absolute level of $2r_0 + r_1$, rendering a noise margin far better than for most encountered signal decision levels (inner levels) of $r_1$.

The relevant parts of the the written CAD *MATLAB* programs are listed as source code in the Appendix.

## 3.3. Filter types for read–channel ICs

The aim of this section is to provide a better understanding of the reasons behind the choice of the equalizer filter type. An analog all–pass system $H_0(s)$ is defined as a causal, stable system with unit amplitude [12]:

$$\mid H_0(jw) \mid = 1 \tag{3.34}$$

Denoting by $u(t)$ the input, and by $y(t)$ the output of an all–pass system, see figure 3.5, the following energy relation holds:

$$\int_{-\infty}^{\infty} \mid u(t) \mid^2 dt = \int_{-\infty}^{\infty} \mid y(t) \mid^2 dt \tag{3.35}$$

In addition, for every $t_0$

$$\int_{-\infty}^{t_0} \mid u(t) \mid^2 dt \geq \int_{-\infty}^{t_0} \mid y(t) \mid^2 dt \tag{3.36}$$

It is obvious that noise energy is not changed by an all–pass filter stage. The way the probability of error is influenced in this case is via the ISI. For MDFE, it would be ideal to concentrate the energy of the equalized channel impulse response into a small number of samples; this would be advantageous from the number of

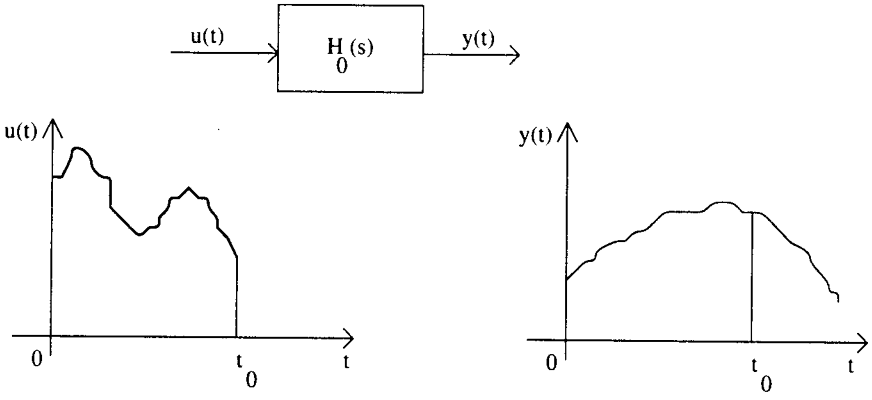

FIGURE 3.5. Input/output waveforms for an all–pass system

feedback taps and ISI point of view. For a causal, stable and minimum–phase system an all–pass filter would rather have an opposite effect. Suppose that $u(t)$ is nonzero only in the $[0, t_0]$ time interval. According to the equations 3.35 and 3.36, $H_0(s)$ has the effect of spreading the incoming signal energy outside $[0, t_0]$, see Figure 3.5, and thus most probably reduce the impulse response peak sample value and noise margin.

The case of $u(t)$ being the dibit response $p(t)$ is different. $p(t)$ has a slow rising ramp, and its energy is not concentrated around the main positive peak. The symmetry of the dibit response was used by McEwen in [13] to model $p(t)$ with an all–pole transfer function having a minimum–phase/causal component and a maximum–phase/anti–causal component. The justification for an all–pass equalizer follows from the fact that a non–causal zero can be used to eliminate the non–minimum phase component of the dibit.

The way the all–pass filter improves the ISI and concentrates the energy is by shifting energy of the signal from the period preceding the peak into the future. This renders an equalized impulse channel response $r(t)$ with better energy compaction, and with a smaller causal undershoot when compared to the original dibit response $p(t)$. The group delay for an all–pass filter is frequency dependent, and high–frequency signals have shorter group delay than low–frequency ones, see figure 3.6. Thus, the fast rising edge accounts for the reduction of the non–causal ISI terms.

To confirm the performance of an all–pass filter relative to a pole–zero filter, simulations were done to test the reduction in ISI achievable in MDFE. The case considered is of a symbol density $PW50 = 3.75T_s$, and a fourth order low–pass Butterworth filter. For a first order all–pass and the low–pass filter mentioned, minimizing $\sigma_E^2$ specifies the optimal design: the pole of the all–pass is located at

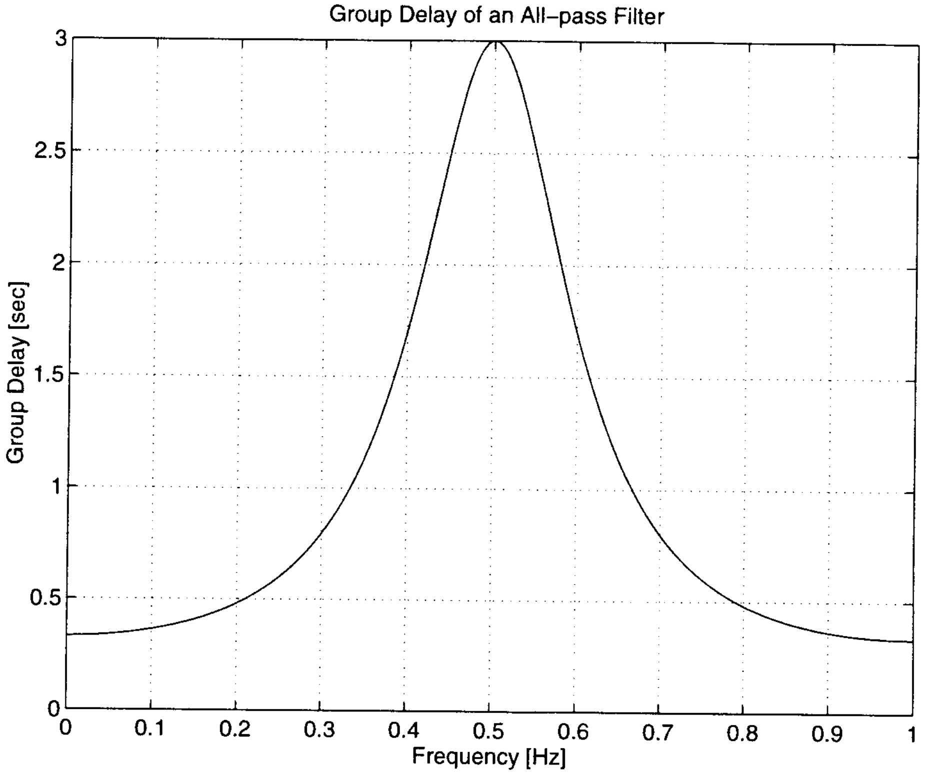

FIGURE 3.6. Group delay all–pass filter

8.82MHz, and corner frequency (-3dB frequency) of the Butterworth filter is at 33.73MHz.

The variation of $\sigma_{ISI}^2$ for a parameter range including the optimal design is depicted in figure 3.7. The pole/zero location was varied and a plot was generated for the ISI term of $\sigma_E^2$. It can be observed that tolerances of about 25% are possible for practical implementation without degrading the values of the objective function significantly.

Figure 3.8 depicts the variation of the ISI with respect to the zero location when a pole/zero filter is employed in place of the all–pass filter. To make the comparison possible, the pole location was fixed to a value previously obtained by optimum design for the all–pass pole. The effects on the ISI as the zero location is

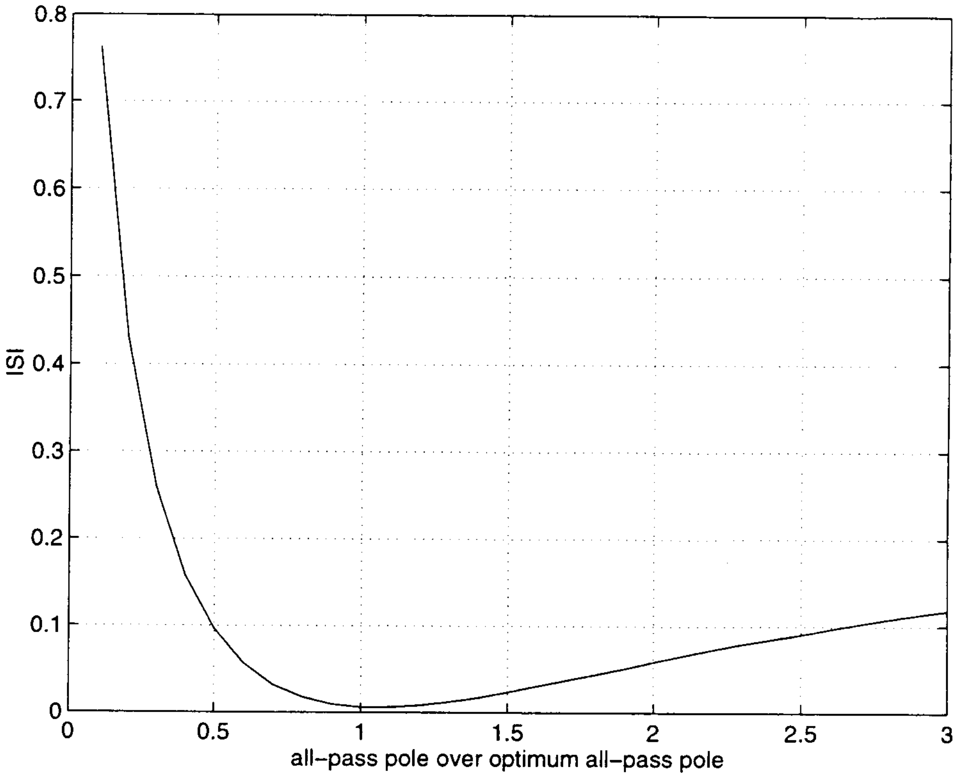

FIGURE 3.7. ISI as a function of the all–pass pole location

varied are shown in figure 3.8. The plot indicates that ISI is minimum for a zero location very close to the optimal zero location for an all–pass filter.

It is thus confirmed by simulation that an all–pass filter gives results very close to optimum for a pole/zero filter of the same order. The matching between the pole and the zero is critical as far as the objective function is concerned. Simulations show that the most important parameter is the ratio between the pole and zero location.

These considerations support the choice of an all–pass filter as part of the equalizer in a read–channel IC. The all–pass filter can be tightly controlled with feedback loops, and the matching between pole and zero for a practical implementation is better than for a pole/zero design.

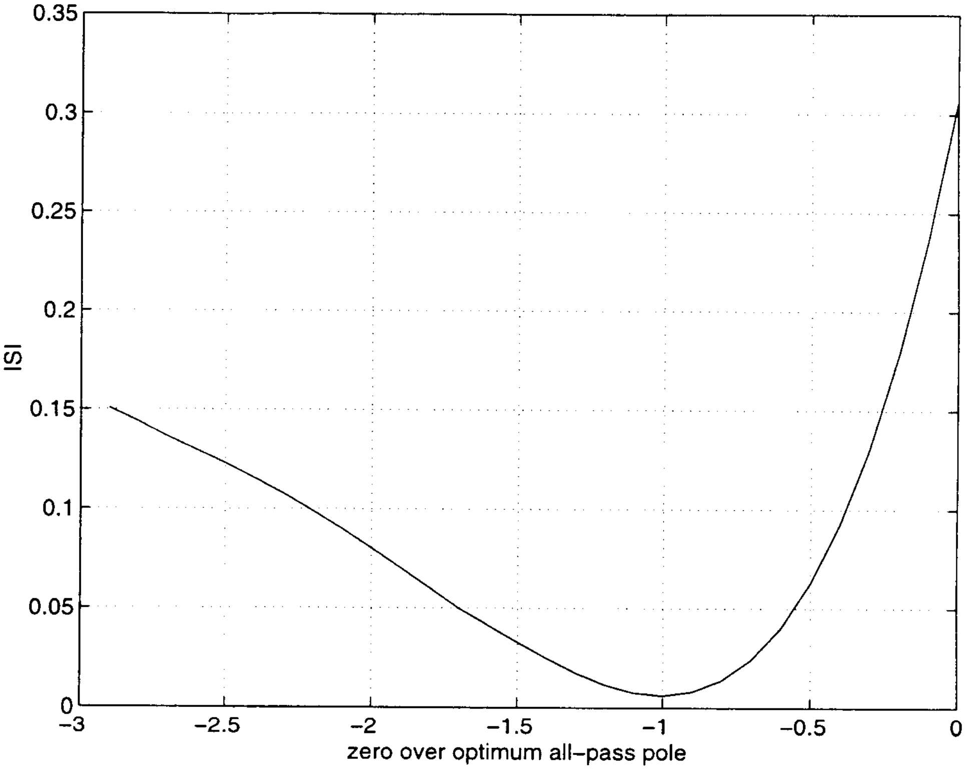

FIGURE 3.8. ISI as a function of the zero/pole ratio

# 4. NONLINEAR OPTIMIZATION APPROACH TO EQUALIZER DESIGN

This chapter is dedicated to presenting simulation results for equalizer designs based on the developed nonlinear optimization software.

Two types of low–pass filters were considered, Butterworth and Bessel. They can be easily specified only by their corner frequency and order, which makes the comparisons meaningful. The Butterworth filter has a flat magnitude response, but its phase characteristic is nonlinear and is responsible for significant pulse dispersion. The Bessel filters have a nearly linear frequency response over a specified frequency range, but the high frequency roll–off characteristic is not as good as for a Butterworth filter of the same order.

Figure 4.1 shows the magnitude and phase characteristic for 4–th order Butterworth and Bessel filter. The -3dB frequency was fixed at the same value, 1 rad/sec, for both filters to allow for fair comparisons. The plots confirm the trade–off between limiting high–frequency noise and pulse dispersion, when choosing between these two filter types.

For the designs and comparisons presented in the following, the number of taps for the FIR feedback filter is fixed at 9. A value of $PW50 = 3.75T_s$ is assumed for the half–height of the Lorentzian pulse, unless otherwise specified. AWGN noise is added at the output of the read–back channel so that $SNR = 13dB$.

Tables 4.1 and 4.2 present optimal design results obtained for different orders of the low–pass filter. For the same filter order, the performances in terms of the $\sigma_E^2$ criterion are very close for both the filter types considered.

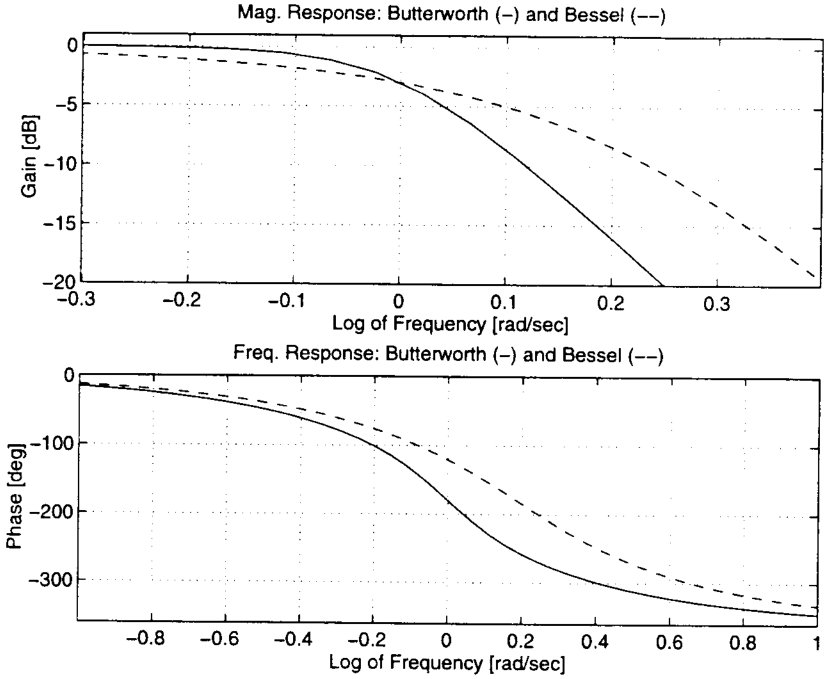

FIGURE 4.1. Magnitude and frequency response of 2 low–pass filters

The -3dB frequencies, obtained by design, for the Bessel filters are lower than that for the Butterworth case. This was expected because of the better high–frequency roll–off characteristic of the Butterworth filters, and the fact that a Bessel filter passes more high–frequency noise energy for the same -3dB frequency than a Butterworth filter.

For a first–order all–pass, the best order for the Butterworth case is 4, and 5 for the Bessel case. This indicates that increasing the order of the low–pass filter does not necessarily improve the objective function. In the same time, the order of the LPF is lower for the same attainable performances for the case of a Butterworth filter than for the case of a Bessel filter.

In the case of $Criterion_1$, among LPF choices of orders up to six and different filter types, the best design for a first–order APF is obtained for the case of a

| $\sigma_E^2$ | AP Order | AP Poles<br>$MHz$ | LP Type | LP Order | LP -3dB Frequency<br>$MHz$ |
|---|---|---|---|---|---|
| 7.79e-02 | 1 | 7.83 | Butterworth | 2 | 24.74 |
| 7.55e-02 | 1 | 7.65 | Butterworth | 3 | 24.11 |
| 7.52e-02 | 1 | 7.39 | Butterworth | 4 | 27.35 |
| 8.21e-02 | 1 | 7.94 | Butterworth | 5 | 25.50 |
| 7.74e-02 | 1 | 7.30 | Butterworth | 6 | 30.78 |

TABLE 4.1. Optimal design for different Butterworth filter orders

| $\sigma_E^2$ | AP Order | AP Poles<br>$MHz$ | LP Type | LP Order | LP -3dB Frequency<br>$MHz$ |
|---|---|---|---|---|---|
| 8.07e-02 | 1 | 7.74 | Bessel | 2 | 20.07 |
| 7.67e-02 | 1 | 7.71 | Bessel | 3 | 22.09 |
| 7.61e-02 | 1 | 7.71 | Bessel | 4 | 22.88 |
| 7.59e-02 | 1 | 7.62 | Bessel | 5 | 22.37 |
| 7.82e-02 | 1 | 8.12 | Bessel | 6 | 28.41 |

TABLE 4.2. Optimal design for different Bessel filter orders

fourth–order low–pass Butterworth filter. In the following comparisons the low–pass filter order is therefore fixed to 4, as a reference.

Table 4.3 gives $\sigma_E^2$ criterion performance results for the case of a fourth–order low–pass filter, and a second order all–pass filter. From the objective function point of view, the designs with a higher order all–pass are better for both the Butterworth

| $\sigma_E^2$ | AP Order | AP Poles | LP Type | LP Order | LP -3dB Frequency |
|---|---|---|---|---|---|
| | | $MHz$ | | | $MHz$ |
| 7.39e-02 | 2 | 13.00 12.26 | Butterworth | 4 | 27.08 |
| 7.48e-02 | 2 | 9.92 13.40 | Bessel | 4 | 20.47 |

TABLE 4.3. Optimal design for second order all-pass

| $P_e$ | AP Order | AP Poles | LP Type | LP Order | LP -3dB Frequency |
|---|---|---|---|---|---|
| | | $MHz$ | | | $MHz$ |
| 1.30e-04 | 1 | 7.71 | Butterworth | 4 | 31.07 |
| 1.69e-04 | 1 | 6.78 | Bessel | 4 | 24.17 |
| 1.23e-04 | 2 | 11.56 13.04 | Butterworth | 4 | 24.47 |
| 1.52e-04 | 2 | 11.81 13.95 | Bessel | 4 | 18.92 |

TABLE 4.4. Optimal design for probability of error minimization

and the Bessel filters. However, the improvement is relative small, and in a practical implementation a first–order all–pass might be preferred due to simplicity reasons.

Table 4.4 considers the second optimization criterion, which is the probability of error $P_e$ (see equation 3.33). For the same order of the low–pass filter, the second–order all–pass filter designs are again predicting better performance, for both filter types. At the same filter order, a Butterworth filter is better than a Bessel filter for a second order all–pass filter stage.

The designs are verified by simulations of bit–error rates (BER). Lowering the BER is the design objective, and represents the method of validating different sets of design parameters. For BER simulations, noise is injected in the channel according to a specified SNR. To approximate a realistic bandwidth of the noise, an oversampling ratio of 4 times the symbol rate is used.

In Chapter 3 the second optimization criterion, $Criterion_2$, was proposed as an improvement of $Criterion_1 = ISI + noise$ which is usually used in digital channel optimization. Since the new criterion optimizes directly the estimate of the error probability, an improvement in BER is expected when using $Criterion_2$ as compared to using $Criterion_1$.

Figure 4.2 depicts BER curves for the case of a forward equalizer consisting of a first–order APF, and a fourth–order low–pass Butterworth. The dotted line was obtained for a design based on minimizing the first criterion (see Table 4.1), and the solid line corresponds to a design based on the second criterion, (see Table 4.4).

A 0.22 dB improvement in BER was obtained for a $SNR = 14dB$ by using $Criterion_2$ instead of $Criterion_1$. This translates into more than 0.15 reduction in the logarithm of the BER, representing approximately 30% reduction in the probability of error.

Figure 4.3 confirms the accuracy with which the error probability was computed for the $Criterion_2$ based design. The solid line depicts the BER obtained at simulation, and the dashed line depicts the estimated BER based on $Criterion_2$ objective function. The differences are very small for a low SNR. As the probability of error decreases, the numerical accuracy affects the values of the $Q$-function and the estimated BER is bigger than for BER simulation.

The degradation of the BER due to the feedback of incorrect decision is depicted in Figure 4.4. The solid line corresponds to the same $Criterion_2$ based design as in Figure 4.2. The dashed line depicts the error rate in the case of no error–propagation, and is obtained by inputing correct decision to the feedback

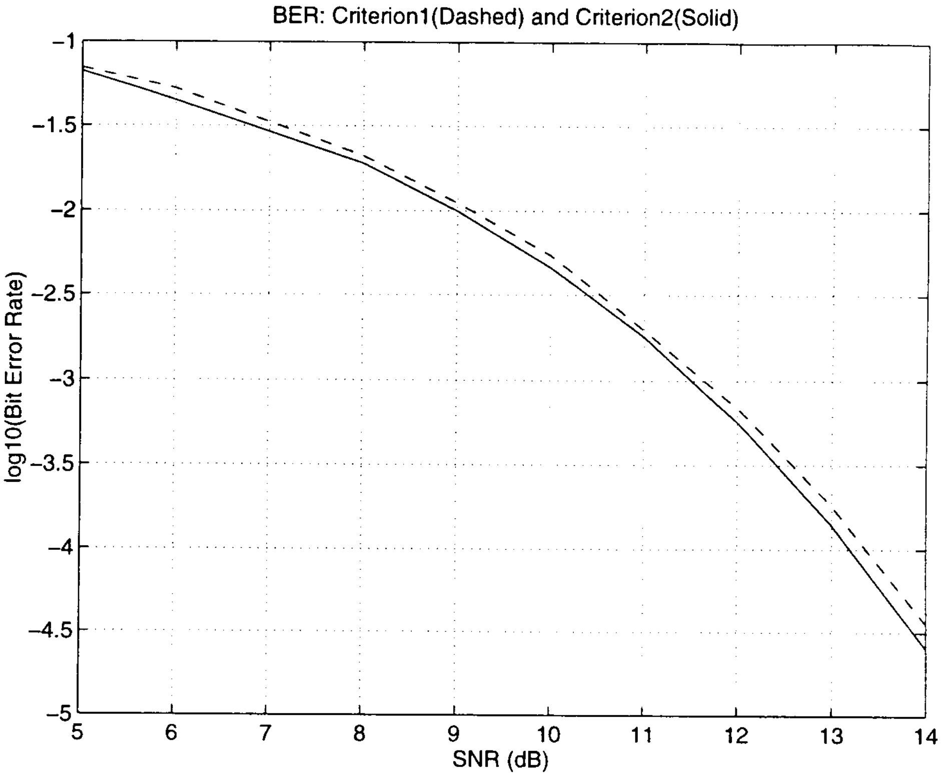

FIGURE 4.2. Error Rate Simulations for MDFE

equalizer $B(D)$ (see Figure 3.1). The degradation of BER due to error propagation is of 0.23dB for 13dB SNR, which corresponds to 33% degradation of BER. As the SNR increases the error-propagation affects the BER less; the probability that an error produces a burst–error length greater than 1 decreases as the SNR increases, being 0 for a noiseless channel.

Figure 4.5 shows the BER curves for $Criterion_1$ and $Criterion_2$ in the case of no error–propagation. No significant difference exist between the 2 plots. It is worth noticing that these curves represent a bound for the BER achievable by the MDFE, because the feedback filter is fed with correct decision.

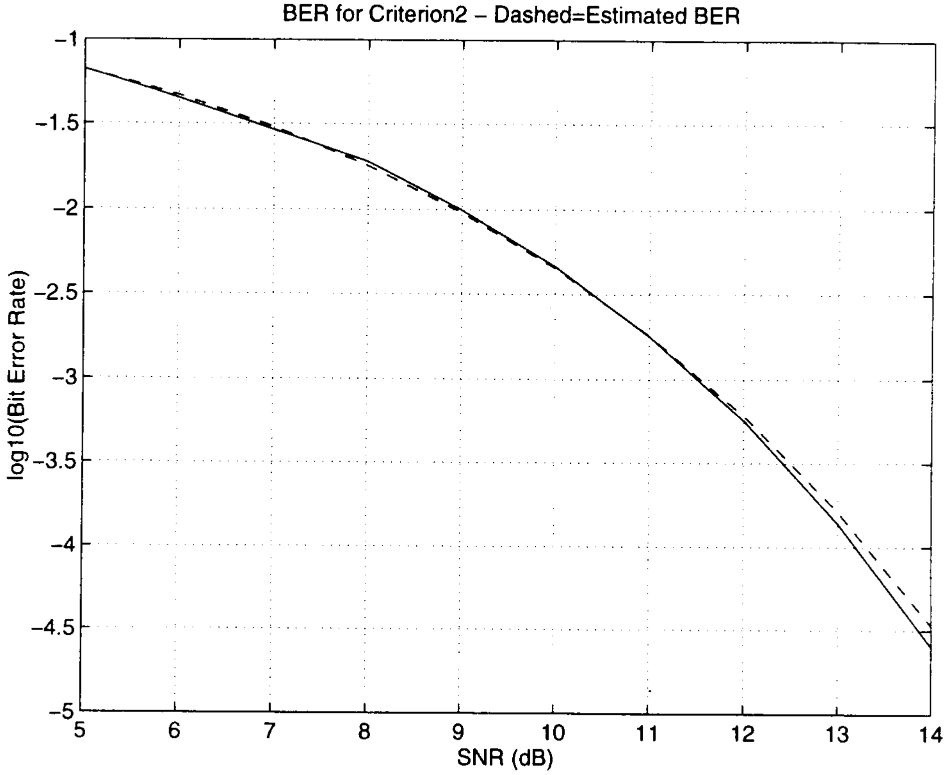

FIGURE 4.3. Error Rate Predictions for MDFE

Figures 4.2 and 4.5 show that the improvement in BER brought by the usage of the new proposed criterion does not come from an attenuation of the intensity of the feedback error-propagation mechanism.

Overall, the simulations show that for a receiver structure consisting of a APF followed by a LPF, a Butterworth LPF is better than a Bessel LPF of the same order. The usage of a APF of order greater than 1 is not recommended because of a lower yield in performance improvements over a first order APF.

A fourth-order Butterworth LPF is the best among designs of order up to 6 in terms of the optimum objective function. The magnitude response of this filter is maximally flat, and provides sufficient roll-off characteristic at high frequencies.

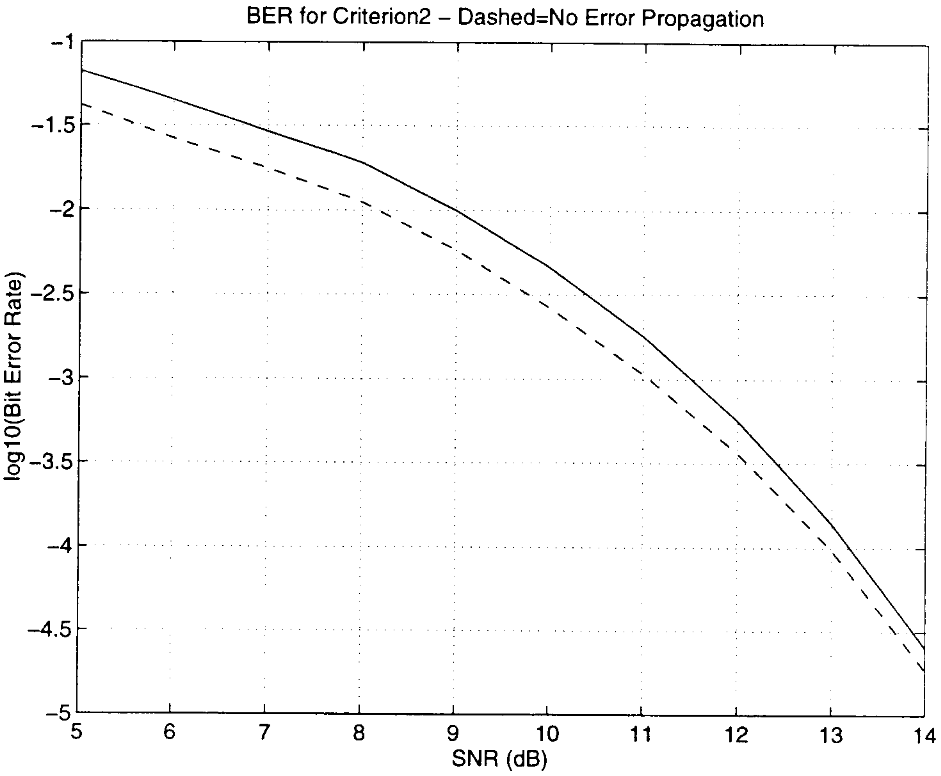

FIGURE 4.4. Error Propagation Effect for MDFE

High-frequency noise is thus band-limited, and its aliasing into the signal baseband is prevented.

The APF and the LPF employed at receiver are usually tunable. It is shown that a significant reduction in the bit error rate is possible, for the same receiver architecture, simply by performing a nonlinear design with a more refined objective function.

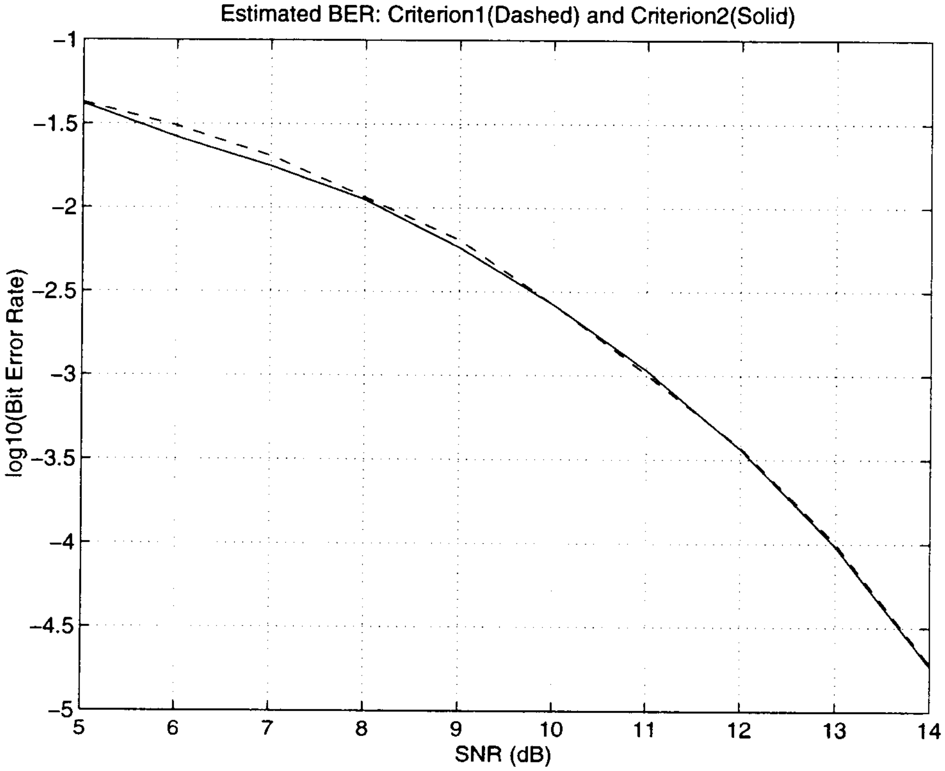

FIGURE 4.5. BER Bound for 2 Criteria for MDFE

# 5. CONCLUSIONS AND FUTURE WORK

## 5.1. Conclusions

The forward equalizer for an MDFE channel can be implemented by cascading an all–pass filter and a low–pass filter. The possibility of using an all–pass filter for equalizing a disk drive data recovery channel is argued, and the hypothesises are compared with simulation. It is estimated that an all–pass filter would guarantee better results in practice than a pole/zero filter due to implementation tolerance problems.

A numerical based approach to the optimal design of MDFE equalizers is proposed. In this context, it is shown that the estimated bit error probability can be used as an objective function. The widely used design based on the minimization of the sum of intersymbol interference and noise is also considered, and comparisons between the two approaches are made.

Programs were written for the *MATLAB* interactive package environment for optimizing the design of the MDFE forward equalizer. The software is organized so that it is easy to use and easy to expand. Different forward equalizer structures can be specified at the design stage, allowing for a unified framework for obtaining and testing different designs. This valuable tool for computer aided design (CAD) was used to compare MDFE designs for various all-pass orders, and low–pass filter orders and types.

Numerical simulations show that the usage of a first–order all–pass filter and a fourth–order low–pass Butterworth filter provides a good compromise between the complexity of the forward equalizer and the achievable bit error rates. A 30% reduction in the probability of decision errors was obtained by implementing a newly proposed objective function.

A good agreement was observed between the channel performance predicted at design, and the results obtained by simulation.

## 5.2. Future work

Future work should extend the capability of the CAD *MATLAB* software developed to specifying other forward equalizer structures. The integration of *MATLAB* code and *C* code for bit error rate simulation would create an environment in which designs can be validated in a minimum of time.

It would be useful to study possible refinements of the estimated bit error probability objective function. Accounting for the error propagation via the feedback filter might be possible by estimating its the probability mass function the same way it was performed for the ISI term.

# BIBLIOGRAPHY

[1] K. D. Fisher, W. L. Abbott, J. L. Sonntag and R. Nesin, "PRML Detection Boosts Hard-disk Drive Capacity", *IEEE SPECTRUM*, pp.70–76, November 1996.

[2] J. M. Cioffi, W. L. Abbott, H. K. Thapar, C. M. Melas and K. D. Fisher, "Adaptive Equalization in Magnetic-Disk Storage Channels", *IEEE Communications Magazine*, pp.15–29, February 1990.

[3] F. Dolivo, "Signal Processing for High-Density Digital Magnetic Recording", *Proc. VLSI and Microelectronic Applications in Intelligent Peripherals and their Interconnection Networks*, Hamburg, West Germany, pp.1/91–96, May 1989.

[4] J. J. Moon and L. R. Carley, "Performance Comparison of Detection Methods in Magnetic Recording, *IEEE Transactions on Magnetics*", vol. 26, no. 6, pp.4567–4572, November 1990.

[5] A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm", *IEEE Transactions on Information Theory IT-13*, pp.260-269, April 1967.

[6] B. P. Viterbi, "Modern Analog and Digital Communications Channels", *The Dryden Press, Saunders College Publishing*, 1989.

[7] J. G. Kenney, L. R. Carley and R. Wood, "Multi-level Decision Feedback Equalization for Saturation Recording", *IEEE Transactions on Magnetics*, vol. 19, no.4, pp.2160–2171, July 1993.

[8] J. G. Kenney and R. Wood, "Multi-level Decision Feedback Equalization: An Efficient Realization of FDTS/DF", *IEEE Transactions on Magnetics*, vol. 31, no.2, pp.1115–1120, March 1995.

[9] J. G. Kenney, "Geometric Representation of the Three Search Detector and Its Hardware Implications", *Ph.D. dissertation* , Dept. of Electrical and Computer Engineering, Carnegie Melon University December 1991.

[10] J. G. Kenney and C. M. Melas, "Pipelining for Speed Doubling in MDFE", *Proceedings of the IEEE International Conference on Communications*, vol. 1, pp.561–565, June 1996.

[11] P. H. Siegel, "Recording Codes For Digital Magnetic Storage", *IEEE Transactions on Magnetics*, vol. MAG.21, no. 5, September 1985.

[12] A. Papoulis, "Signal Analysis", *McGraw–Hill Book Company*, 1977.

[13] P. A. McEwen and J. G. Kenney, "Allpass Forward Equalizer for Decision Feedback Equalization", *IEEE Transactions on Magnetics*, vol. 31, no. 6, pp.3045–3047, November 1995.

[14] G. D. Fourney, "Maximum–Likelihood Sequence Estimation of Digital Sequences in the Presence of Intersymbol Interference", *IEEE Transactions on Information Theory IT-18*, pp.363-378, May 1972.

[15] S. U. H. Qureshi, "Adaptive Equalization", *Proceedings of IEEE*, vol. 73, no. 9, pp. 1349-1386, September 1985.

[16] D. J. Tyner and J. G. Proakis "Partial Response Equalizer Performance in Digital Magnetic Recording Channels", *IEEE Transactions on Magnetics*, vol. 29, no. 6, pp. 4194–4208, November 1993.

[17] M. Cohn, G. V. Jacoby, and A. Bates III, "Data encoding method and system employing two–thirds code rate with full word look–ahead", *U.S. Patent 4,337,458*, 1982.

[18] A. Grace, "OPTIMIZATION TOOLBOX for use with $MATLAB^{TM}$ ", *The MathWorks, Inc.*, November 1990.

# APPENDIX

## *MATLAB* Nonlinear Optimization Code

```
****************************** srch.m *************************
% Design of MDFE receiver for minimum BER at detector.
% This function uses constrained optimization to find the
% optimal values for the 3dB attenuation frequency of a
% LPF, and the pole/zero location of an APF.
% The LPF can be Butterworth of Bessel, of different orders;
% the order of the APF is 1 or 2
clear; clear global;
format short e;
% Sets the tap number of the feedback FIR filter.
LenBFIR = input('LenBFIR=');
% Corrects FB filter length for MDFE: the first tap is r3.
LenBFIR=LenBFIR+3;
% Sets options used by 'constr'
options = foptions; % Gets the default options.
options(3) = .01;   % Sets the termination tolerance
options(14) = 5000; % Sets the maximum number of iterations.
% Reduce the function call parameter number by declaring global
% variables.
global OSR PW50 Windw SNRdB LPOrd LenBFIR tzero hk step_head opt;
OSR = input('OSR=');     % Input oversample ratio.
```

```
PW50 = input('PW50=');   % Input Lorentzian pulse parameter.
Windw = input('Windw='); % Input the width of the simulation
                         %  time-window (in symbol periods).
SNRdB = input('SNRdB='); % Input the design SNR value in dB.
LPOrd = input('LPOrd='); % Specifies the order of the LPF.
% Creates the Lorentzian and dibit response based on design
%  parameters.
[hk,step_head,tzero] = create_hk(Windw,OSR,PW50);
% Specifies the maximum allowed -3dB frequency "cw" of the LPF
% as a ratio of the symbol frequency.
cw_max = 5/10;
% Sets the minimum allowed -3dB frequency.
cw_min = cw_max/20;
% Sets the initial -3dB frequency.
cw_ini = cw_max/2;
% Specifies the forward equalizer structure
opt_text = str2mat('opt=0 -> LPF="Butterworth", 1-st order APF.',...
                   'opt=1 -> LPF="Bessel", 1-st order APF.', ...
                   'opt=2 -> LPF="Butterworth", 2-nd order APF.',...
                   'opt=3 -> LPF="Bessel", 2-nd order APF.');
disp(opt_text);
opt=input('Specify Design Option: opt=');
% Performs Constrained Minimization
if opt == 0 | opt == 1
   VLB = [0.01 cw_min]; % VLB=[min(APF_pole*Ts) min(cw)].
   VUB = [1 cw_max];    % VUB=[max(APF_pole*Ts) max(cw)].
   Arg0 = [0.1 0.35]; % Sets the initial design parameters values.
   % Constrained Optimization Function Call
   Arg = constr('fcstr',Arg0,foptions,VLB,VUB);
end;
if opt == 2 | opt == 3
   VLB = [0.01 0.01 Wc_min]; % VLB=[min(APF_poles*Ts)min(cw)].
   VUB = [0.4 0.4 Wc_max];   % VUB=[max(APF_poles*Ts) max(cw)].
   % Sets the initial design parameters values.
   Arg0=[0.13 0.14 Wc_ini];
   % Constrained Optimization Function Call
   Arg = constr('fcstr',Arg0,foptions,VLB,VUB);
end;
% Computes the impulse response of the forward equalizer for the
% optimal design.
fgk = ctfilt_th(Arg,opt,LPOrd,OSR);
% Computes relevant information corresponding to the best design.
[Pe,table,ANorm,sig_root,pk_sim,pk,p1,pk_ISI,i0,...
                              fract_shift,PhShift] = est_Pe(fgk);
% Saves the design in a file for future usage.
save filter_e_pdf.mat OSR PW50 Windw SNRdB ButOrd PhShift LenBFIR ...
     opt fgk i0 Arg hk tzero Pe table ANorm sig_root pk_sim pk p1 ...
     fract_shift;
```

```
**************************** fcstr.m ****************************

function [F,G] = fcstr(X)

% The function fcstr returns a scalar value of the function to be

% minimized, F, and a matrix of constraints;

% F is minimized such that G < zeros(G).

G = -1;

global OSR PW50 Windw SNRdB LPOrd LenBFIR tzero hk step_head opt;

% Computation of the impulse response "fgk" of the forward equalizer.

% cw_3dB is the frequency which should be used in MATLAB function

% calls to LPF design so that the -3dB frequency is exactly the

% one given in the design parameter vector X.

[fgk,cw_3dB] = ctfilt_th(X,opt,LPOrd,OSR);

% Computation of the objective function

[Pe,table,ANorm,sig_root,pk_sim,pk,p1,pk_ISI,i0, ...

                             fract_shift,PhShift] = est_Pe(fgk);

% Monitors the objective function, and the design parameters

% at each optimization iteration

F = Pe;

disp(F);

disp(X);
```

```
**************************** est_Pe.m ************************
function [Pe,table,ANorm,sig_root,pk_sim,pk,p1, ...
                 pk_ISI,i0,fract_shift,PhShift] = est_Pe(fgk);
% Estimates the objective function "Pe" based on:
%    "fgk"   - the impulse response of the forward filter
%    "hk"    - the step response (Lorentzian) of the media
%    "SNRdB" - the SNR (dB)
global OSR SNRdB LenBFIR tzero hk;
% Computes the impulse response of the equalized channel.
pk = conv(hk,fgk);
% Finds the index of the peak value of "pk".
index_max =  sort_max(pk);
% Finds the index i1+fract_shift for which
% pk(i0+fract_shift) = pk(i2+fract_shift);
% i0 = i1-OSR; i2 = i1+OSR; i1 is an integer.
[i0,i1,i2,fract_shift] = find_phsh(pk,OSR,index_max);
% "PhShift" is the discrete time, in the oversampled
% domain, corresponding to T0: pk(T0)=pk(T0+2*Ts).
PhShift = i0 - tzero + fract_shift;
% Computatinos necessary for preparing the interpolation.
if PhShift >= 0
t_start = 1+rem(PhShift,OSR);
else
t_start = 1+OSR+rem(PhShift,OSR);
end;
```

```
 if size(pk,1) > size(pk,2) pk = pk'; end;
pk_pre_i = interp1(1:length(pk),pk,i0+fract_shift-OSR:-OSR:1);
 if size(pk_pre_i,1) > size(pk_pre_i,2) pk_pre_i = pk_pre_i'; end;
pk_pre = interp1(1:length(pk),pk,i0+fract_shift-OSR:-OSR:1);
 if size(pk_pre,1) > size(pk_pre,2) pk_pre = pk_pre'; end;
pk_pre = fliplr(pk_pre);
pk_post = interp1(1:length(pk),pk,i0+fract_shift:OSR:length(pk));
 if size(pk_post,1) > size(pk_post,2) pk_post = pk_post'; end;
p1 = pk_post(2);
ANorm = 1/p1; % Normalizing factor for making the inner level 1.
% "pk_sim" is the sequence to be used in simulations.
pk_sim = [pk_pre pk_post];
% "pk_ISI" contains the terms of "pk" responsible for ISI.
pk_ISI = [pk_pre pk_post(LenBFIR+1:length(pk_post))];

% Retains M terms from post and pre--cursor for PMF calculations.
M = 4;
pk_pre = pk_pre_i(1:M);
pk_post = pk_post(LenBFIR+1:min(length(pk_post),LenBFIR+M));
table_pre = [];
table_post = [];

for i=1:2^M
% 'bin_rep' returns the binary representation of
% i, and a warning for RLL(1,7) code violation.
```

```
        [vect,viol] = bin_rep(i,M);
        if viol ~= 1
                table_pre = [table_pre [vect*pk_pre'; 0]];
                table_post = [table_post [vect*pk_post'; 0]];
        end;
end;

% Sets the second line of the tables describing the
% PMFs: the probability that ISI takes a given value
% is computed aposteriori.
table_pre(2,:) = table_pre(2,:)+1/size(table_pre,2);
table_post(2,:) = table_post(2,:)+1/size(table_post,2);

% Computes the PMF of the pre-cursor and post-cursor
% ISI random variable
table = conv_prob(table_pre,table_post);

% Reduction and sorting of the series describing the PMF.
table = red_tab(table);
table = sort_tab(table);

N_2 = (norm(hk,2)^2)*10^(-SNRdB/10);
sigma_2 = N_2 * norm(fgk,2)^2;
sig_root = sqrt(sigma_2);
Pe = PError(table,p1,sig_root);
```

```
****************************** PError.m ************************
function Pe = PError(table,p1,sig_root)


% Returns the probability for the ISI + Noise exceeding
% the threshold p1 in one sense (positive).
% Table contains the probability distribution function of the
% ISI, and sig_root is the standard deviation of the noise.


Pe = 0;
for i = 1:size(table,2)
        Pe = Pe + table(2,i)*(0.5-erf_book((p1-table(1,i))/sig_root));
end;



***************************** erf_book.m ***********************
function rez = erf_book(x)


% Uses MATLAB defined 'erf' function to obtain the function erf
% according to the definition from the page 61 of the book
% Probability, Random Processes, and Estimation Theory,
% by John W. Wood, Prentice-Hall, 1994.


rez = erf(x/sqrt(2))/2;
```

```
****************************** ctfilt_th.m *********************

function [fgk,cw_3dB] = ctfilt_th(X,opt,LPOrd,OSR)


% Creates the impulse response of the CT Filter.

% X is a vector parameter, interpreted according to the

% option "opt".


if opt == 0

        % Butterworth  + 1st AP

        pole = X(1);

        WBut = X(2);

        T_s = 1; % this is in seconds

        pole = pole*2*pi/T_s;

        % The pole results as a fraction of symbol frequency (100MHz)

        Ba = [1 -pole];

        Aa = [1 pole];

        cw = filt_3dB(opt,LPOrd,2*pi*WBut/T_s);

        % cw should be now in radians per second.

        % WBut corresponds to WBut/Ts in Hz

        % -> cw = 2*pi*WBut*(1/Ts)

        [Bc,Ac] = butter(LPOrd,cw,'s');

        Be = conv(Ba,Bc);

        Ae = conv(Aa,Ac);

        [Y,X,T_v] = impulse(Be,Ae);

        T_max = max(T_v);
        T_c = 0:T_s/OSR:T_max;
        fgk = -impulse(Be,Ae,T_c);
end;

if opt == 1
        % Butterworth  + 1st AP
        pole = X(1);
        WBut = X(2);
        T_s = 1; % this is in seconds
        pole = pole*2*pi/T_s;
        % The pole results as a fraction of symbol frequency (100MHz)
        Ba = [1 -pole];
        Aa = [1 pole];
        cw = filt_3dB(opt,LPOrd,2*pi*WBut/T_s);
        % cw should be now in radians per second.
        % WBut corresponds to WBut/Ts in Hz
        % -> cw = 2*pi*WBut*(1/Ts)
        [Bc,Ac] = besself(LPOrd,cw);
        Be = conv(Ba,Bc);
        Ae = conv(Aa,Ac);
        [Y,X,T_v] = impulse(Be,Ae);
        T_max = max(T_v);
        T_c = 0:T_s/4:T_max;
        fgk = -impulse(Be,Ae,T_c);
end;
```

```
if opt == 10
        % Butterworth  + zero + pole
        zero = X(1);
        pole = X(2);
        WBut = X(3);
        T_s = 1; % this is in seconds
        pole = pole*2*pi/T_s;
        zero = zero*2*pi/T_s;
        % The pole results as a fraction of symbol frequency (100MHz)
        Ba = [1 zero];
        Aa = [1 pole];
        cw = filt_3dB(opt,LPOrd,2*pi*WBut/T_s);
        % cw should be now in radians per second.
        % WBut corresponds to WBut/Ts in Hz
        % -> cw = 2*pi*WBut*(1/Ts)
        [Bc,Ac] = butter(LPOrd,cw,'s');
        Be = conv(Ba,Bc);
        Ae = conv(Aa,Ac);
        [Y,X,T_v] = impulse(Be,Ae);
        T_max = max(T_v);
        T_c = 0:T_s/OSR:T_max;
        fgk = -impulse(Be,Ae,T_c);
end;


if opt == 2
        % Butterworth  + 2nd AP
        pole1 = X(1);
        pole2 = X(2);
        WBut = X(3);
        T_s = 1; % this is in seconds
        pole1 = pole1*2*pi/T_s;
        pole2 = pole2*2*pi/T_s;
        % The pole results as a fraction of symbol frequency (100MHz)
        Ba = conv([1 -pole1],[1 -pole2]);
        Aa = conv([1 pole1],[1 pole2]);
        cw = filt_3dB(opt,LPOrd,2*pi*WBut/T_s);
        % cw should be now in radians per second.
        % WBut corresponds to WBut/Ts in Hz
        % -> cw = 2*pi*WBut*(1/Ts)
        [Bc,Ac] = butter(LPOrd,cw,'s');
        Be = conv(Ba,Bc);
        Ae = conv(Aa,Ac);
        [Y,X,T_v] = impulse(Be,Ae);
        T_max = max(T_v);
        T_c = 0:T_s/OSR:T_max;
        fgk = impulse(Be,Ae,T_c);
end;

if opt == 3
        % Bessel  + 2nd AP
        pole1 = X(1);
        pole2 = X(2);
```

```
    WBut = X(3);

    T_s = 1; % this is in seconds

    pole1 = pole1*2*pi/T_s;

    pole2 = pole2*2*pi/T_s;

    % The pole results as a fraction of symbol frequency (100MHz)

    Ba = conv([1 -pole1],[1 -pole2]);

    Aa = conv([1 pole1],[1 pole2]);

    cw = filt_3dB(opt,LPOrd,2*pi*WBut/T_s);

    % cw should be now in radians per second.

    % WBut corresponds to WBut/Ts in Hz

    % -> cw = 2*pi*WBut*(1/Ts)

    [Bc,Ac] = besself(LPOrd,cw);

    Be = conv(Ba,Bc);

    Ae = conv(Aa,Ac);

    [Y,X,T_v] = impulse(Be,Ae);

    T_max = max(T_v);

    T_c = 0:T_s/OSR:T_max;

    fgk = impulse(Be,Ae,T_c);

end;
```