

**FILTERING RANDOM NOISE
FROM ONE AND TWO DIMENSIONAL SIGNALS
VIA DATA COMPRESSION**

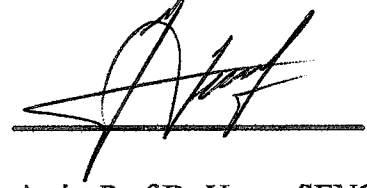
**A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of
Dokuz Eylül University
in Partial Fulfilment of Requirements for
the Degree of Master of Science in Electronics Engineer, Electronic Program**

**by
Lale ÇILDIR**

**September, 1997
İZMİR**

M.Sc THESIS EXAMINATION RESULT FORM

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.



Assist.Prof.Dr. Yavuz ŞENOL
(Advisor)



Dr. Haldun Sarnel
(Committee Member)



Dr. Reyat...Tifmaz...
(Committee Member)

Approved by the
Graduate School of Natural and Applied Sciences



Prof.Dr. Cahit Helvacı
Director

DOĞUMANTASTAYON MERKEZİ

ACKNOWLEDGMENTS

I am grateful to my advisor, Yard. Doç.Dr.Yavuz Şenol, who supported and encouraged me to build up this M.Sc. Thesis. I would also like to thank, Yard. Doç.Dr.Gökalp Kahraman and Prof.Dr.Yusuf Öztürk who have helped me for clarifying several of the theoretical problems contained in this thesis.

Many thanks to the individuals ; Balas K. Natarajan and Konstantinos Konstantinides who provided helpful comments and recent papers.

Also thanks to my friends who have helped me to collect documents about the subject and have provided suggestions.

ABSTRACT

Digital signal processing is a rapidly evolving field with growth in science and engineering. As a result of these developments, we are using more and more digital data in our work and daily life.

However, the large amount of data usually prevents us from using digital processing equipment effectively. Moreover, with the increase in the number of such equipment around, noise reduction is becoming more important.

In this thesis these two main fields of digital signal processing, signal compression and filtering, are studied. In chapter two, lossy compression schemes are explained. In chapter three and four, filtering of one and two dimensional signals are given respectively. Chapter four covers a new type of compression method that can also be used as a filter and that solves the problem of filtering broadband deterministic signals. A comparison between this last method called the "Occam Filter" and the others is made. The results show that Occam filter outperforms the other methods especially when medical data is in question.

ÖZET

Sayısal İşaret İşleme fen ve mühendislik bilimlerindeki ilerlemete paralel olarak gelişmektedir. Bu gelişmelerin sonucu olarak da hayatımızın birçok alanında daha çok sayısal data işlemleri girmiştir.

Fakat, sayısal data miktarının büyüklüğü, sayısal cihazların etkin kullanımını engellemektedir. Dahası, etraftaki elektronik cihazların çokluğu elektronik gürültüyü arttırmıştır.

Bu tezde, sinyal işlemenin iki ana konusu olan ve yukarıda belirtilen problemleri çözmeye çalışan sinyal sıkıştırma ve filtreleme incelenmiştir. İkinci konuda kayıplı sıkıştırma teknikleri, üçüncü ve dördüncü konuda sırasıyla bir ve iki boyutlu filtreleme teknikleri, besinci konuda sıkıştırarak filtreleme yapan yeni bir teknik olan Occam filtre anlatılmıştır. Altıncı konuda ise klasik sıkıştırma ve filtreleme yöntemleri ile Occam filtreleme deneysel olarak karşılaştırılmıştır. Sonuçta, özellikle tek boyutta, Occam filtrenin diğer yöntemlerden daha iyi olduğu anlaşılmıştır.

CONTENTS

	page
Contents	V
List of Figures	IX
List of Tables	X

Chapter One

INTRODUCTION

Chapter two

SIGNAL COMPRESSION

2.1 Introduction	3
2.2 Fundamentals	3
2.2.1 Measuring Information	3
2.2.2 Data Redundancy	5
2.2.2.1 Coding Redundancy	6
2.2.2.2 Interpixel Redundancy	7
2.2.2.3 Psychovisual Redundancy	8
2.2.3 Fidelity Criteria	9
2.3 Signal Compression Frame Work	10
2.4 Run Length Coding	12
2.5 Transform Coding	12
2.5.1 General Transform Description	13
2.5.2 KLT	15

2.5.3 DCT	15
2.5.4 DFT	16
2.5.5 Choosing the Right Transform	16
2.5.6 Block Size Selection	18
2.5.7 Bit Allocation	18
2.5.7.1 Zonal Coding	18
2.5.7.1 Threshold Coding	19
2.6 Vector Quantization	19
2.6.1 Training	20
2.6.2 Quantization	21
2.6.2.1 First Method	21
2.6.2.2 Second Method	21

Chapter Three

ONE DIMENTIONAL SIGNAL FILTERING

3.1 introduction	23
3.2 FIR Filter Specifications	23
3.2.1 FIR Design	25
3.2.2 Windows	27
3.2.3 Filter Transformations	29
3.3 Design of Recursive Filters	30
3.3.1 Analog Filter Design	30
3.3.1.1 Butterworth Filter	30
3.3.1.2 Chebyshev Filter	34
3.3.1.3 Bessel Filter	40
3.3.1.4 Frequency transformations	45
3.3.1.5 s-z Domain Transformations	47
3.3.1.5.1 The Impulse Invariant Method	47
3.3.1.5.2 Modified Impulse Invariant Method	49
3.3.1.5.3 Matched z Transform Method	50

Chapter Four

TWO DIMENTIONAL SIGNAL FILTERING

4.1 Introduction	51
4.2 Background	51
4.2.1 Spatial Domain Methods	51
4.2.2 Frequency Domain Methods	53
4.3 Spatial Filtering	55
4.3.1 Smoothing Filters	57
4.3.1.1 Lowpass Filtering	57
4.3.1.2 Median Filtering	58
4.3.2 Sharpening Filters	58
4.3.2.1 Basic Highpass Spatial Filtering	58
4.3.2.2 Highboost Filtering	59
4.3.2.3 Derivative Filters	60
4.4 Filtering in the Frequency Domain	62
4.4.1 Lowpass Filtering	62
4.4.2 Highpass Filtering	64

Chapter Five

OCCAM FILTERS

5.1 Introduction	66
5.2 Noise Strength	67
5.3 Description of the Algorithm	71

Chapter Six

APPLICATIONS

6.1 Introduction	72
------------------	----

6.2 Experiments for Filters	73
6.3 Experiments for Compression	73
6.4 Results of the Filter Experiments	73
6.5 Comparison of the Filter Types	75
6.6 Results of the Compression Experiments	88
6.7 Comparison of the Compression Types	98
6.8 Image Filtering with Occam Filters	98
6.9 Image Experiments and Results	100

Chapter Seven

CONCLUSION

References	103
------------	-----

LIST OF TABLE

page

Table 2.1 Variable-Length Coding Example

7



LIST OF FIGURES

	page
Figure 2.1 General compression Scheme	10
Figure 2.2 A Transform Coding System:	13
(a) encoder	
(b) decoder	
Figure 2.3 The Periodicity implicit in the 1-D:	17
(a) DFT	
(b) DCT	
Figure 3.1 Digital Filter Specifications a)LP b)HP c)BP d)BS	24
Figure 3.2 Ideal Lowpass Filter Response	25
Figure 3.3 The Impulse Response of an Ideal (Zero Phase Shift) LP Filter	26
Figure 3.4 The Impulse Response of an Approximating Filter	27
Figure 3.5 Ideal and Approximate Frequency Responses	27
Figure 3.6 Loss of LP normalized Butterworth Approximations	32
Figure 3.7 Roots of Third Order Butterworth	34
Figure 3.8 The Equiripple Passband Characteristic	35
Figure 3.9 Plots of	39
a) Third Order Chebyshev Function	
b) Fourth Order Chebyshev Function	
c) Third Order Chebyshev Approximation	
d) Fourth Order Chebyshev Approximation	
Figure 3.10 Locus of Roots of Chebyshev Approximation	39
Figure 3.11 Characteristics of a Fourth Order Chebyshev ($A_{\max}=0.5$ dB):	40
a) Loss	
b) Delay	
c) Step Input	
d) Step Response	

Figure 3.12	Characteristics of a Fourth Order Butterworth ($A_{\max}=3$ dB); a) Loss b) Delay c) Step Input d) Step Response	41
Figure 3.13	Loss of LP Bessel Approximations	43
Figure 3.14	Delay of LP Bessel Approximations	44
Figure 3.15	Characteristics of a Fourth Order Bessel Approximation ($A_{\max}=3$ dB); a) Loss b) Delay c) Step Input d) Step Response	44
Figure 3.16	Block Diagram of the Frequency Transformation Procedure	45
Figure 3.17	A typical Band-pass Function	46
Figure 3.18	A typical Band-reject Function	47
Figure 4.1	A3 x 3 Neighborhood about a Point (x, y) in an Image	52
Figure 4.2	Gray-level Transformation Functions for Contrast Enhancement	53
Figure 4.3	Top:Cross-Sections of Basic Shapes for Circularly Symmetric Frequency Domain Filters.Bottom: Cross-Section of Corresponding Spatial Domain Filters.	56
Figure 4.4	A 3x3 Mask with Arbitrary Coefficients (Weights)	57
Figure 4.5	A Basic Highpass Spatial Filter	58
Figure 4.6	Mask Used for High-Boost Spatial Filtering	60
Figure 4.7	A 3x3 Region of an Image and Various Masks Used to Compute the Derivative at Point Labeled Z_5	61
Figure 4.8	a)Perspective Plot of an Ideal Lowpass Filter Transfer Function b) Filter Cross-Section	63
Figure 4.9	a) A Butterworth Lowpass Filter b) Radial Cross-Section for $n=1$	64
Figure 4.10	Perspective Plot and Radial Cross-Section of Ideal Highpass Filter Perspective	64
Figure 4.11	Plot and Radial Cross-Section of Butterworth Highpass Filter for $n=1$	65

Figure 5.1	Illustrative Plot of Compressed Size Versus Allowed Loss for the Noise Sequence	68
Figure 5.2	Illustrative Plot of Compressed Size Versus Allowed Loss for the Noise-Free Sequence	69
Figure 5.3	Illustrative Plot of Compressed Size Versus Allowed Loss for the Noisy Sequence with Second Derivative Plot	70
Figure 6.1	a) Frequency Response of Noise b) Frequency Response of Noisy Sequence	76
Figure 6.2	a) Plot of Noise-Free Sequence b) Noisy Sequence c) Lowpass Filtered Sequence ($n=300, W_n=0.2$)	77
Figure 6.3	a) Lowpass Filtered Sequence ($n=300, W_n=0.7$) b) Lowpass Filtered Sequence ($n=2, W_n=0.2$) c) Lowpass Filtered Sequence ($n=2, W_n=0.7$)	78
Figure 6.4	a) Butterworth Filter Magnitude Response b) Phase Response	79
Figure 6.5	Results of Butterworth Filters a) $R_p=0.1\text{dB}$ b) $R_p=1\text{dB}$ c) $R_p=10\text{dB}$	80
Figure 6.6	a) Chebyshev Filter Magnitude Response b) Phase Response	81
Figure 6.7	Results of Chebyshev Filters a) $A=0.1\text{dB}, R_p=0.5\text{dB}$ b) $A=1\text{dB}, R_p=0.5\text{dB}$ c) $A=10\text{dB}, R_p=0.5\text{dB}$	82
Figure 6.8	Results of Chebyshev Filters a) $A=0.1\text{dB}, R_p=0.1\text{dB}$ b) $A=1\text{dB}, R_p=0.1\text{dB}$ c) $A=10\text{dB}, R_p=0.1\text{dB}$	83
Figure 6.9	a) Bessel Filter Magnitude Response b) Phase Response	84

Figure 6.10	Results of Bessel Filters	85
	a) $n=2$, $W_n=0.05\text{dB}$	
	b) $n=2$, $W_n=0.1\text{dB}$	
	c) $n=2$, $W_n=0.2\text{dB}$	
Figure 6.11	a) Plot of Compressed Size Versus Allowed Loss	86
	b) Second Derivative	
Figure 6.12	a) Original Noise-Free Signal	87
	b) Wiener Filtered Sequence	
	c) Occam Filtered Sequence	
Figure 6.13	ECG Signal	89
Figure 6.14	a) Decompressed Sequence After it is Compressed by Vector Quantization	90
	b) Decompressed Sequence After it is Compressed by DCT	
	c) Decompressed Sequence After it is Compressed by Run Length Coding	
Figure 6.15	a) Plot of the Compressed Size versus Allowed Loss ε	91
	b) Second Derivative	
Figure 6.16	Results of Occam Compressed Sequences	92
	a) $\varepsilon=0.15$	
	b) $\varepsilon=0.34$	
	c) $\varepsilon=0.8$	
Figure 6.17	EMG Signal	94
Figure 6.18	a) Decompressed Sequence After it is Compressed by Vector Quantization	95
	b) Decompressed Sequence After it is Compressed by DCT	
	c) Decompressed Sequence After it is Compressed by Run Length Coding	
Figure 6.19	a) Plot of the Compressed Size versus Allowed Loss ε	96
	b) Second Derivative	
Figure 6.20	Results of Occam Compressed Sequences	97
	a) $\varepsilon=500$	
	b) $\varepsilon=1112$	

c) $\epsilon = 1500$

Figure 6.21

a) Original Image

101

b) Noisy Image

c) Lowpassed Filtered Image

d) Occam Filtered Image

e) Median Filtered Image

f) f domain Filtered Image



CHAPTER ONE

INTRODUCTION

The developments in electronic technology eased data processing. However, it seems the more the technology develops, the more data there is to process.

For example, the growth in medical electronics and space technology brought with it a large number of data collected by instruments. The number of data is far more than that could be stored and moreover, during the transmission and collection of data, noise causes problem.

Compression and filtering are two branches of signal processing that deal with these problems.

To understand compression, clear distinction must be made between data and information. Data is the means by which information is carried. The aim of compression is to find the minimum amount of data that can carry the same amount of information. The redundancy of data that the compression tries to ignore is grouped as:

1. coding redundancy,
2. intersample redundancy,
3. psychovisual redundancy.

To ignore the redundant data, lossy and lossless compression schemes are used. In lossless compression no information is lost at the end of the compression and in lossy compression some information is lost. Consequently, lossy schemes compress more than the lossless ones.

Digital filters are also grouped into two. The first group is the FIR (Finite Impulse Response) filters. FIR filters are designed by first finding the appropriate frequency

response and then by inverse Fourier transforming this to impulse response coefficients. The second group is the digital filters found by applying s- to z-transforms to existing analog filters. Both of these groups have highpass, lowpass, bandpass and band reject types defined according to the frequency ranges they modify.

Two dimensional filtering is a branch of picture enhancement. The approaches to enhancement fall into two broad categories as the spatial domain and frequency domain methods. In spatial domain methods a direct manipulation is done to the pixel values themselves, whereas in the frequency domain methods, the Fourier Transform of a picture is modified.

Recently a new type of filter-compression “Occam Filter” is developed. The essence of its algorithm is to compress the signal with a loss tolerance that is equal to the noise strength, because a loss value that is equal to the noise strength cancels the noise. Hence, the output compressed signal is also a filtered signal. Occam filters can be applied to higher dimensions as well.

In this thesis, the subjects mentioned above are studied in the second, third, fourth and fifth chapters. In the sixth chapter the results of the traditional filtering and compression techniques are compared with the new Occam filter and it is shown that, especially in one dimension, Occam filter outperforms the others.

CHAPTER TWO

SIGNAL COMPRESSION

2.1 Introduction

The demand for handling signals in digital form has increased dramatically in recent years with the exponential growth in computing power over the past decade. Especially in communications and in some branches of medicine storage of signals is very important. However, the large number of bits required to represent the signals prevents the efficient use of today's computer technology.

Hence signal compression tries to solve this problem of reducing the amount of data required to represent signals. Fortunately digital signals, in their canonical representation, generally contain a significant amount of redundancy. Signal compression, which is the art/science of efficient coding of data, aims to take advantage of this redundancy to reduce the number of bits required to represent a signal. From the mathematical point of view, this amounts to transforming a sample array into statistically uncorrelated data set.

2.2 Fundamentals

As we have mentioned above, the term data compression refers to the process of reducing the amount of data required to represent a given quantity of information. To do this, a clear distinction must be made between data and information. They are not synonymous. Infact data are the means by which information is conveyed.

2.2.1 Measuring Information

The fundamental premise of information theory is that the generation of information can be modelled as a probabilistic process that can be measured in a manner that agrees with

intuition. In accordance with this supposition, a random event E that occurs with probability $P(E)$ is said to contain

$$I(E) = \log_r \frac{1}{P(E)} = -\log_r P(E) \quad \text{r-array units of information.} \quad (2.1)$$

where $I(E)$ is usually called the self information of E . Note that communicating that E has not occurred conveys more information, because this outcome is less likely on the other hand if $P(E)=1$, then $I(E)=0$ and no information is attributed to it. That is, because no uncertainty is associated with the event, no information would be transferred by communicating that the event has occurred.

Let's say a source has J samples a_j ($j=1, \dots, J$) with probabilities $P(a_j)$. If k source samples are generated (the law of large numbers stipulates that, for sufficiently large value of k , sample a_j will -on average- be output $kP(a_j)$ times), the average self information obtained from k output is

$$\begin{aligned} & -kP(a_1)\log P(a_1) - kP(a_2)\log P(a_2) - \dots - kP(a_J)\log P(a_J) \\ & = -k \sum_{j=1}^J P(a_j) \log P(a_j) \end{aligned} \quad (2.2)$$

The average information per source output is $H(z)$

$$H(z) = - \sum_{j=1}^J P(a_j) \log P(a_j) \quad (2.3)$$

and is called the uncertainty or the entropy of the source. It defines the average amount of information (in r-array units/sample) obtained by observing a single source output.

If the source samples are equally probable, the entropy or uncertainty of Equation 2.3 is maximised, then the source provides the greatest possible average information per source sample.

2.2.2 Data Redundancy

If two individuals use a different number of words to tell the same basic story, two different versions of the story are created, and at least one includes nonessential data. That is, it contains data (or words) that either provide no relevant information or simply restate that which is already known. It is thus said to contain *data redundancy*.

If n_1 and n_2 are taken as the number of information carrying units in the two data sets represents the same information, *relative data redundancy* R_D of the first data set is

$$R_D = 1 - \frac{1}{C_R} \quad (2.4)$$

where C_R is the compression ratio

$$C_R = \frac{n_1}{n_2} \quad (2.5)$$

Here, if

i) $n_2 = n_1$ then $C_R = 1$, $R_D = 0$

which means,(relative to the second data set) the first representation of the information contains no redundant data.

ii) $n_2 \ll n_1$, then $C_R \rightarrow \infty$, $R_D \rightarrow 1$

which indicates that there is highly redundant data in the first representation relative to the second data.

iii) $n_2 \gg n_1$, then $C_R \rightarrow 0$, $R_D \rightarrow -\infty$

which implies that second data set contains much more data than the original representation.

For example (10:1) compression ratio means that the first data set contains 10 information carrying units (say bits) for every 1 unit in the second or compressed data set. In digital signal sequence compression, there are three basic data redundancies that can be identified and explained. Namely,

1. coding redundancy,
2. intersample redundancy,
3. psychovisual redundancy.

2.2.2.1 Coding Redundancy

If r_k is a value level in a signal sequence, then the probability that each r_k occurs is

$$P_r(r_k) = \frac{n_k}{n} \quad k = 0, 1, \dots, L-1 \quad (2.6)$$

where

L : number of sample types.

n_k : the number of times the k^{th} value level occurs

n : signal sequence length (number of samples)

Taking $L(r_k)$ as the number of bits used to represent each value r_k , we can find the average number of bits required to represent each sample as

$$L_{\text{avg}} = \sum_{k=0}^{L-1} \ell(r_k) \cdot p_r(r_k) \quad (2.7)$$

If the value levels of a signal sequence are coded in a way that uses more code samples than absolutely necessary to represent each value level, that is if the code fails to minimise Equation 2.7 the resulting signal sequence is said to contain *coding redundancy*.

In most signal sequences, certain value levels are more probable than the others. A natural binary coding of these levels assigns the same number of bits to both the most and least probable values, thus failing to minimise Equation 2.7. However, if fewer bits to the more probable value levels are assigned than the less probable areas, then this type (Equation 2.4 type) of data compression is called *variable code length*.

As an example consider the 8-level signal sequence distribution shown in Table 2.1.

Table 2.1 Variable-Length Coding Example

r_k	$p_r(r_k)$	Code 1	$l_1(r_k)$	Code 2	$l_2(r_k)$
$r_0 = 0$	0.19	000	3	11	2
$r_1 = 1/7$	0.25	001	3	01	2
$r_2 = 2/7$	0.21	010	3	10	2
$r_3 = 3/7$	0.16	011	3	001	3
$r_4 = 4/7$	0.08	100	3	0001	4
$r_5 = 5/7$	0.06	101	3	00001	5
$r_6 = 6/7$	0.03	110	3	000001	6
$r_7 = 1$	0.02	111	3	000000	6

L_{avg} for the first type of coding is 3 while for the second type of coding it is 2.7 bits.

$$\begin{aligned}
 L_{avg} &= \sum_{k=0}^7 \ell_2(r_k) \cdot p_r(r_k) \\
 &= 2(0.19) + 2(0.25) + 2(0.21) + 3(0.16) + 4(0.08) + 5(0.06) + 6(0.03) + 6(0.02) \\
 &= 2.7 \text{ bits}
 \end{aligned}$$

Thus, we can say that the first coded signal sequence contains coding redundancy and the exact level of redundancy can be calculated from Equation 2.4

$$R_D = 1 - \frac{1}{1.11} = 0.099$$

2.2.2.2 Intersample Redundancy

This is another form of data redundancy -one directly related to the intersample correlation within a signal sequence. Because the value of given sample can be reasonably predicted from the value of its neighbours, the information carried by individual samples is relatively small and much of the visual contribution of a single sample to a signal sequence is redundant, because it could have been guessed on the basis of its neighbour's values.

In order to reduce the intersample redundancies in a signal sequence, sample array must be transformed into more efficient (but not visual) format. For example, the differences between adjacent samples can be used to represent a signal sequence. Transformations of this type are referred to as *Mappings*.

Run Length coding is an example of a mapping where the samples are mapped to sequence of pairs (g_i, r_i) , (g_2, r_2) , ... in which g_i denotes the i^{th} value level encountered along the line and r_i the Run Length of i^{th} run.

Given below a sequence. Four value levels 0 and 1 are used.

[00, 00, 01, 01, 01, 11, 10, 10, 10, 11, 11]

the signal sequence is mapped to pairs (00, 2), (01, 3), (11, 1), 10, 3), (11, 2). These pairs of course can be further compressed.

2.2.2.3 Psychovisual Redundancy

Sometimes the eye does not respond with equal sensitivity to all visual information. Certain information simply has less relative importance than other information in normal processing. This information is said to be *psychovisually redundant*. It can be eliminated without significantly impairing the quality of signal sequence perception.

Psychovisual redundancy is fundamentally different from the redundancies discussed earlier. Unlike coding and intersample redundancy, psychovisual redundancy is associated with real or quantifiable visual information. Its elimination is possible only because of the information itself is not essential for normal visual processing. Since the elimination of psychovisual redundant data results in a loss of quantitative information, it is commonly referred to as *quantization*. This terminology is consistent with normal usage of the word, which generally means the mapping of a broad range of input values to a limited number of output values. As it is irreversible operation (visual information is lost), quantization results in lossy data compression.

2.2.3 Fidelity Criteria

The criteria used in signal processing for defining the better signal is called the *Fidelity Criteria* which has two general classes as:

1. Objective fidelity criteria
2. Subjective fidelity criteria

When the level of information loss can be expressed as a function of the original or input signal sequence and the compressed and subsequently decompressed output signal sequence, it is said to be based on *objective fidelity criterion* a good example is the root-mean-square.

$$e(x) = \hat{f}(x) - f(x) \quad (2.8)$$

total error:

$$\sum_{x=0}^{N-1} [\hat{f}(x) - f(x)] \quad (2.9)$$

$$e_{\text{rms}} = \frac{1}{N} \sum_{x=0}^{N-1} [\hat{f}(x) - f(x)]^{1/2} \quad (2.10)$$

Another, closely related objective fidelity criterion is *the mean square signal to noise ratio of the compressed-decompressed signal sequence*

$$\text{SNR}_{\text{rms}} = \sqrt{\frac{\sum_{x=0}^{N-1} \hat{f}(x)^2}{\sum_{x=0}^{N-1} [\hat{f}(x) - f(x)]^2}} \quad (2.11)$$

Although objective fidelity criteria offer a simple and convenient mechanism for evaluating information loss, if decompressed signal sequences ultimately are viewed by

Although objective fidelity criteria offer a simple and convenient mechanism for evaluating information loss, if decompressed signal sequences ultimately are viewed by humanbeings, measuring signal quality by the subjective evaluations of a human observer often is more appropriate. This can be accomplished by showing a “typical” decompressed signal sequence to an appropriate cross section of viewers and averaging their evaluations. The evaluations may be made using an absolute rating scale or by means of side-by-side comparisons of $f(x)$ and $\hat{f}(x)$. In either case, the evaluations are said to be based on *subjective fidelity criteria*

2.3 Signal Compression Frame Work

Until now some techniques that are used to compress or reduce the amount of data required to represent a signal sequence are explained. Generally speaking compression techniques are classified into two as lossy or lossless compression. In lossless compression, the reconstructed signal sequence is numerically identical to the original signal sequence on a sample by sample basis. In the lossy compression, some of the information of the original signal sequence is lost during compression, consequently lossy compression has a higher compression ratio as compared to lossy schemes. A general compression scheme is illustrated in Figure 2.1.

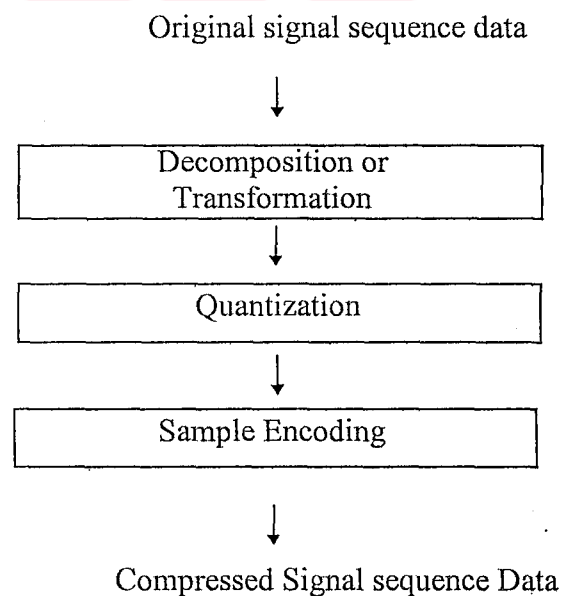


Figure 2.1 General Compression Scheme

The signal sequence decomposition or transformation is performed to reduce the dynamic range of signal, to eliminate redundant information, or in general to provide a representation that can be coded more efficiently. This stage is common for lossless and lossy techniques. A reversible operation can be performed on the original signal sequence.

The primary difference between lossy and lossless schemes is inclusion of the next stage, namely quantization in lossy techniques. By quantising the data, the number of possible output samples is reduced. The type and degree of quantization has a large impact on the bit rate and quality of lossy scheme. It is also desirable to perform the quantization in such a way that the resulting output sequence can be subsequently encoded efficiently. The sample encoding process might include techniques such as Huffman coding or arithmetic coding as a means of achieving rates close to entropy of the quantised sample source.

Since our main purpose is to use compression techniques for filtering random noise, we will concentrate on lossy compression techniques.

In general, any of the components of a lossy scheme may be implemented in an adaptive or nonadaptive mode. A compression scheme is adaptive, if the structure of a component or its parameters changes locally within a signal sequence to take advantage of variations in local statistics. Adaptivity offers the potential for improved performance in exchange for an increase in complexity. The adaptivity can be achieved either in a casual or a noncasual fashion.

In systems with casual adaptivity, the coder parameters are based only on the previously reconstructed sample values and any process leading to a decision at the encoder is duplicated at the decoder. These systems have the advantage of requiring no overhead information. Unfortunately there are two disadvantages associated with such system. First, the encoder may fail to adopt to abrupt changes in input statistics that can not be inferred from previously reconstructed values. Second, casual adaptivity usually increases the complexity of the encoder and the decoder by the same amount since the decoder must duplicate the decision making process of the encoder.

not available at the decoder, the encoder must send additional bits to the decoder to inform it of any adaptations. Although this results in a higher bit rate, it also increases the overall system performance. Moreover, the increase in the decoder complexity due to adaptation is minimal since the decoder does not need to repeat the adaptation selection process at the encoder.

In the following sections different approaches to compression are explained.

2.4 Run Length Coding

An effective coding type to represent each row of a signal sequence or bit plane by a sequence of lengths that describe successive runs of sample values is Run Length coding. Since a coding is done compression is achieved.

Although Run Length coding is in itself an effective method of compressing a signal sequence, additional compression can be realised by variable-length coding the Run Lengths themselves. Run Length coding is done as follows:

1. The first sample value is taken as the reference value (V_{ref}) and a counter C is initialised.
2. A threshold value t is predefined.
3. While $V_{ref} - t < \text{next sample value} < V_{ref} + t$, the counter C is increased by one.
4. When a sample value does not satisfy the inequality of step three, V_{ref} and the C are kept as the first pair and that new sample value is defined as the new V_{ref} .
5. C is initialised.
6. Steps 3, 4 and 5 are repeated until all the samples are done.

2.5 Transform Coding

Another type of compression technique is the transform coding which is based on modifying the transform of a signal sequence. In transform coding, by using a reversible, linear transform (such as DCT, FT) the signal sequence is mapped onto a set of transform coefficients, which are then quantised and coded. A significant number of coefficients in

coefficients, which are then quantised and coded. A significant number of coefficients in most natural signal sequences have small magnitudes therefore they can be coarsely quantized or even discarded entirely with little signal sequence distortion.

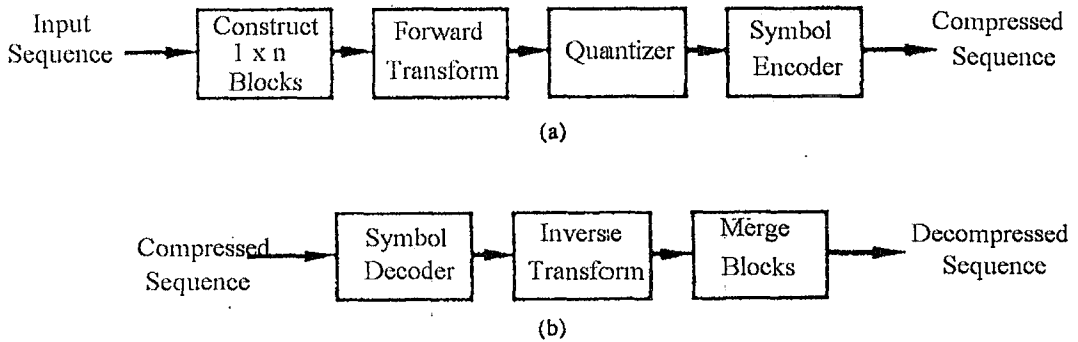


Figure 2.2 A Transform Coding System: (a) encoder; (b) decoder

In Figure 2.2 a typical transform coding system is shown. An input signal sequence is first divided into $1 \times n$ blocks. Next, the transform of each block is taken to form the block transform arrays. Then the coefficients which have small magnitudes are eliminated or coarsely quantized by the quantizer. As the last step, the output of the quantizer is coded adaptively as explained in section 2.3. To get the decompressed signal sequence from the compressed one, the inverse transform of the block is taken after it is decoded and they are merged to form the whole signal sequence.

One thing to notice here is that the compression is achieved during the quantization not transformation!

2.5.1 General Transform Description

An $n \times n$ signal sequence $f(x)$ can be expressed as a function of its 1D transform $T(u)$

$$f(x) = \sum_{u=0}^{n-1} T(u) h(x, u) \quad x = 0, 1, \dots, n-1 \quad (2.12)$$

$$H_u = [h(0, u), \dots, h(n-1, u)]$$

F which is an $1 \times n$ matrix containing the samples of $f(x)$ can be defined as a linear combination of H_u matrices which are the basis signal sequences of the transform used to compute the series expansion weighting coefficients $T(u)$

$$F = \sum_{u=0}^{n-1} T(u) H_u \quad (2.13)$$

A masking function is formed to eliminate the coefficients which satisfy a truncating function, (which make the smallest contribution to the signal sequence)

$$\hat{F} = \sum_{u=0}^{n-1} T(u) m(u) H_u \quad (2.14)$$

$$m(u) = \begin{cases} 0 & \text{if } T(u) \text{ satisfies a specified truncating criterion.} \\ 1 & \text{otherwise} \end{cases} \quad (2.15)$$

The mean square error between block F and its approximation \hat{F} is,

$$= E \left\{ \left\| \sum_{u=0}^{n-1} T(u) H_u - \sum_{u=0}^{n-1} T(u) m(u) H_u \right\|^2 \right\} \quad (2.16)$$

$$= E \left\{ \left\| \sum_{u=0}^{n-1} T(u) H_u [1 - m(u)] \right\|^2 \right\} \quad (2.17)$$

$$= \sum_{u=0}^{n-1} \sigma_{T(u)}^2 [1 - m(u)] \quad (2.18)$$

where $\|F - \hat{F}\|$ is the matrix norm of $(F - \hat{F})$ and $\sigma_{T(u)}^2$ is the variance of the coefficient at transform location (u). The final simplification is based on the orthonormal nature of the

where $\|F - \hat{F}\|$ is the matrix norm of $(F - \hat{F})$ and $\sigma_{T(u)}^2$ is the variance of the coefficient at transform location (u). The final simplification is based on the orthonormal nature of the basis signal sequences and the assumption that the samples of F are generated by a random process with zero mean and known covariance. The total mean square approximation error thus is the sum of the variances of the discarded transform coefficients (that is, the coefficients for which $m(u)=0$, so that $[1-m(u)]$ in Equation 2.18 is 1) transformations that redistribute or pack the most information into the fewest coefficients provide the best block approximations and consequently the smallest reconstruction errors. Finally, under the assumptions that led to Equation 2.18, the mean square error of the blocks of an $N \times N$ signal sequence are identical. Thus the mean square error (being measure of average error) of the signal sequence equals the mean square error of a single block.

2.5.2 Karhunen - Loève (KLT) Transform

The KLT Transform of a signal sequence is found by the following steps,

1. A $1 \times N$ signal sequence matrix is divided into $1 \times n$ blocks of sequences.
2. The autocorrelation matrix of the whole signal sequence C is calculated
3. The eigen values of the C are found by

$$|C - I\lambda| = 0$$

4. The eigenvectors are computed

$$C \cdot \delta_i = \lambda_i \cdot \delta_i$$

5. λ and δ are put in descending order
6. Transform vector is performed by

$$\Phi = [\delta_{\min} \cdots \delta_{\max}]$$

7. Transform is done by

$$v_i = \Phi x_i$$

1. By keeping only the first j elements of v_i (masking the rest) a compression of J/n^2 is achieved

$$V_i = [V_1 \quad V_2 \quad \cdots \quad V_j \quad 0 \quad 0 \quad \cdots \quad 0 \quad 0 \quad 0]$$

2.5.3 Discrete Cosine Transform (DCT)

The one dimensional DCT for an $1 \times N$ signal sequence $f(x)$ is

$$C(u) = \sum_{x=0}^{N-1} 2 f(x) \cos \left[\frac{(2x+1)u\pi}{2N} \right] \quad (2.19)$$

$$u = 0, 1, 2, \dots, N-1$$

and the inverse discrete cosine transform is

$$f(x) = \frac{1}{N} \sum_{u=0}^{N-1} \alpha(u) C(u) \cos \left[\frac{(2x+1)u\pi}{2N} \right] \quad (2.20)$$

$$x = 0, 1, 2, \dots, N-1$$

where

$$\alpha(u) = \begin{cases} \frac{1}{2} & \text{for } u = 0 \\ 1 & \text{for } u = 1, 2, \dots, N-1 \end{cases} \quad (2.21)$$

2.5.4 Discrete Fourier Transform (DFT)

The one dimensional DFT pair is:

$$F(u) = \sum_{x=0}^{N-1} f(x) \cdot \exp \left[-j2\pi \left(\frac{ux}{N} \right) \right] \quad u = 0, 1, \dots, N-1 \quad (2.22)$$

$$f(x) = \frac{1}{N} \sum_{u=0}^{N-1} F(u) \cdot \exp \left[j2\pi \left(\frac{ux}{N} \right) \right] \quad x = 0, 1, \dots, N-1 \quad (2.23)$$

2.5. Choosing the Right Transform

Transform coding systems based on KLT, DCT, DFT or Walsh-Hadamard (WHT) and various other transforms have been studied extensively. To choose which one of the

Although for most natural signal sequences DCT is better than WHT or KLT in information packing ability, the mean square error described by the Equation 2.10 is minimized by KLT for any input signal sequence or retained coefficient. However because KLT is data dependent, calculating the signal sequence basis for each signal sequence is a nontrivial task, therefore sinusoidal transforms such as DFT or DCT which have fixed signal sequence base sets are preferred. In fact nonsinusoidal transforms such as WHT or Haar transform are easier to apply, but sinusoidal transforms show a closer performance to the information packing ability of optimal KLT.

When DFT of a signal sequence is taken, a block like appearance forms on the signal sequence which is called the “blocking artefact” which results when the boundaries between the block sequences become visible. This is because of the Gibbs phenomenon. This phenomenon occurs because the FT fails to converge uniformly at discontinuities, Fourier expansions take on the mean values. That is, at the boundaries of the block sequences discontinuities are formed and at these boundaries, the blocks take on the mean value of the boundaries.

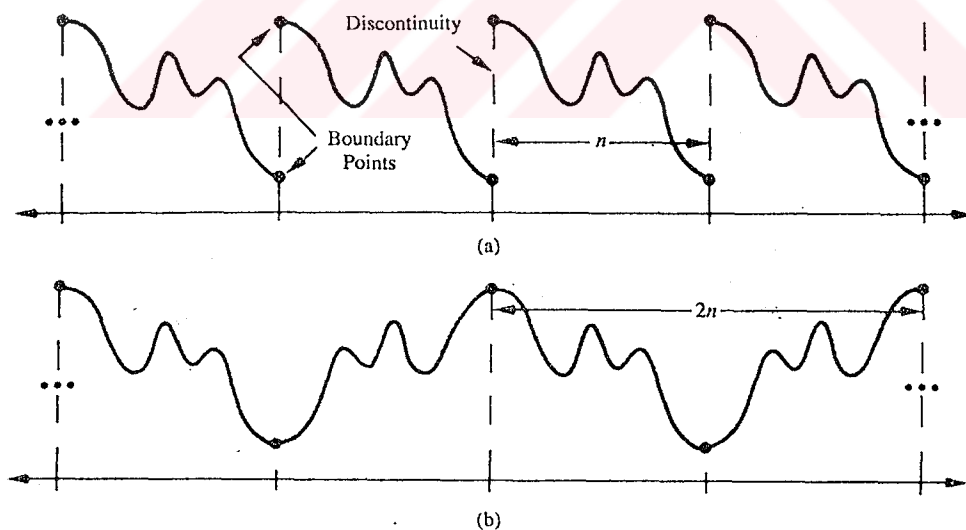


Figure 2.3 The Periodicity Implicit in the 1-D (a) DFT and (b) DCT

As shown in Figure 2.3 DCT minimizes this blocking artefact. Moreover, for most natural signal sequences DCT packs the most information into the fewest coefficients. The KLT basis signal sequences of a first order Markov signal sequence source closely resemble the DCT's basis signal sequences (Ahmet [1974]) as the correlation between

KLT basis signal sequences of a first order Markov signal sequence source closely resemble the DCT's basis signal sequences (Ahmet [1974]) as the correlation between adjacent samples approaches one, the input independent KLT basis signal sequences become identical to the input independent DCT basis signal sequences, (Clarke [1985]). In addition to these properties, DCT has the advantage of being implemented in a single integrated circuit.

As a result, DCT provides a good compromise between information packing ability and computational complexity and reduces the blocking artefact. Therefore it is proved be a such practical value that it has become the international standard for transform coding systems.

2.5.6 Block Size Selection

Another significant factor affecting transform coding error and computational complexity is block size. In most applications, signal sequences are subdivided so that the correlation (redundancy) between adjacent block is reduced to some acceptable level and so that n is an integer power of 2 where, as before, n is the block dimension. The latter condition simplifies the computation of the block transforms. In general, both the level of compression and computational complexity increase as the block size increases. The most popular block size are 1×8 and 1×16 .

2.5.7 Bit Allocation

The reconstruction error associated with the truncated series expansion of Equation 2.14 is a function of the number and relative importance of the transform coefficients that are discarded, as well as the precision that is used to represent the retained coefficients. In most transform coding systems, the retained coefficients are selected on the basis of maximum variance, called *zonal coding*, or on the basis of maximum magnitude, called *threshold coding*. The overall process of truncating, quantizing, and coding the coefficients of a transformed block is commonly called *bit allocation*.

2.5.7.1 Zonal Coding

Zonal coding is based on the information theory concept of viewing information as uncertainty. Therefore the transform coefficients of maximum variance carry the most picture information and should be retained in the coding process. The variances themselves can be calculated directly from the ensemble of transformed block arrays or based on an assumed signal sequence model (say, a Markov autocorrelation function). In either case the zonal sampling process can be viewed, in accordance with Equation 2.14, as multiplying each $T(u)$ by the corresponding element in a zonal mask, which is constructed by placing a 1 in the locations of maximum variance and a 0 in all other locations. Coefficients of maximum variance usually are located around the origin of a signal sequence transform, resulting in the typical zonal mask .

2.5.7.2 Threshold Coding

Zonal coding usually is implemented by using a single fixed mask for all blocks. Threshold coding, however, is inherently, adaptive in the sense that the location of the transform coefficients retained for each block vary from one signal sequence to another. In fact, threshold coding is the adoptive transform coding approach most often used in practice because of its computational simplicity. The underlying concept is that, for any block, the transform coefficients of largest magnitude make the most significant contribution to reconstructed block quality. Because the locations of the maximum coefficients vary from one block to another, the elements of $T(u)m(u)$ normally are reordered (in a predefined manner) to form a 1-D, run-length coded sequence.

2.6 Vector Quantization

Vector quantization is a compression that can be used alone or as a submethod of another compression method.

In this method, each of m sample which we have a value between $[k, l]$ of a signal sequence is mapped onto the closest one of f different values. Where $n \ll m$, and $k \ll n \ll l$. There are two main mapping methods as training and quantization.

2.6.1 Training

An example to training is *Pairwise Nearest Neighbour* method which is explained in following steps.

- 1- The signal sequence is divided into $1 \times m$ blocks. (The signal sequence in the examples is divided into 1×2 blocks for a better explanation)
- 2- Each block accepted as a point in space, the value of these points are unimportant, but the coordinates are the sample values of the block.
- 3- The no of quantization levels (mentioned as n above) is decided.
- 4- Each point is paired with its closest neighbour. The distance is calculated as

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \text{ (each point must be used once.)}$$

- 5- The center of each pair is calculated using

$$x_c = \frac{x_1 + x_2}{2} \quad y_c = \frac{y_1 + y_2}{2}$$

- 6- The pairs are deleted and only their center points are left.
- 7- The number of points in the space is decreased to its one half.
- 8- steps 4,5 and 6 are repeated till the number of points in space equals the preferred quantization level.

The average of the coordinates of each point left is calculated and the averages form a codebook.

$$\text{Point}_{\text{average}} = \frac{x_i + y_i}{2}$$

- 10- The value of the each sample of the original signal sequence is compared with the values of the codebook and the closest codebook value is taken to represent the sample.

2.6.2 Quantization

There are several types of quantization. Two most popular ones are given below.

2.6.2.1 First Method

- 1- Each sample of the original signal sequence is represented in space as in the pairwise nearest neighbour method.
- 2- The center of gravity of the points C_0 is calculated by taking the average of points.
- 3- Two neighbouring points C_{01} and C_{02} are framed by

$$C_{01} = C_0 + \Delta$$

$$C_{02} = C_0 - \Delta$$
 according to a Δ value chosen previously.
- 4- C_0 is deleted from space.
- 5- The points which are closer to C_{01} is grouped under C_{01} and the rest is grouped under C_{02} .
- 6- Steps 2, 3 and 4 are applied to these new groups separately.
- 7- The C_{01} and C_{02} calculated in step 6 is compared with previously calculated C_{01} and C_{02} .
- 8- Steps 2, 3, and 4 are repeated till the two C_{01} and C_{02} that follow one another is the same. There are the real averages.
- 9- All the original signal sequence points are grouped under these two real averages (according to their distance under whichever of the two averages as closest.)
- 10- A codebook of two values is formed by these real averages.
- 11- If more codebook values are wanted, steps 1-10 are repeated to reach the two groups separately.

2.6.2.2 Second Method

- 1- Points of signal sequence are formed in space as in the preceding methods.
- 2- Center of gravity C_0 is calculated
- 3- The furthers point to C_0 is found (C_1)
- 4- The furthers point to C_1 is found (C_2)

- 5- The points that are closest to C_1 is grouped under C_1 and the rest is grouped under C_2 .
- 6- The center of gravity of each group is found. (C_{01} and C_{02})
- 7- The average of the coordinates of C_{01} and C_{02} is calculated.

$$C_{01 \text{ arg}} = \frac{x_{01} + y_{01}}{2}$$

$$C_{02 \text{ arg}} = \frac{x_{02} + y_{02}}{2}$$

- 8- Instead of the original signal sequence sample, $C_{01 \text{ arg}}$ or $C_{02 \text{ arg}}$ is send -whichever is closest in value-
- 9- If a codebook of more values are expected then steps 1-6 are repeated on each group separately as if they were on original signal sequence.
- 10- If steps 1-6 are repeated k times, then 2^k valued codebook is found.



CHAPTER THREE

ONE DIMENTIONAL SIGNAL FILTERING

3.1 Introduction

The problem of designing filters can be approached from two ways:

- i) We can specify the frequency response of the filter and by inverse Fourier transform we can determine from this the impulse response coefficients. This leads to Nonrecursive (or Finite Impulse Response -FIR-) filters.
- ii) We can apply transformations from s plane to z plane to existing analog filters.

3.2 FIR Filter Specifications

Digital filters are usually specified in the frequency domain in terms of the frequency ranges which they are to leave unaffected and those which are to be removed from any input signal. The amplitude response of the four main types are shown in Figure 3.1. It should be remembered that the frequency response of a discrete system is periodic with period ω_s and so we need only consider the response on the interval $[-\omega_s/2, \omega_s/2]$.

As for the phase response, it is impossible to achieve zero phase shift over the entire frequency range, but it is possible to achieve a linear phase response in nonrecursive filter which has a gain (transfer) function

$$G(z) = \sum_{i=0}^m a_i z^{-i} \quad (3.1)$$

By definition of the z-transform of the impulse response $\{g(n)\}$ of the filter

$$G(z) = \sum_{i=0}^{\infty} g(i)z^{-i} \quad (3.2)$$

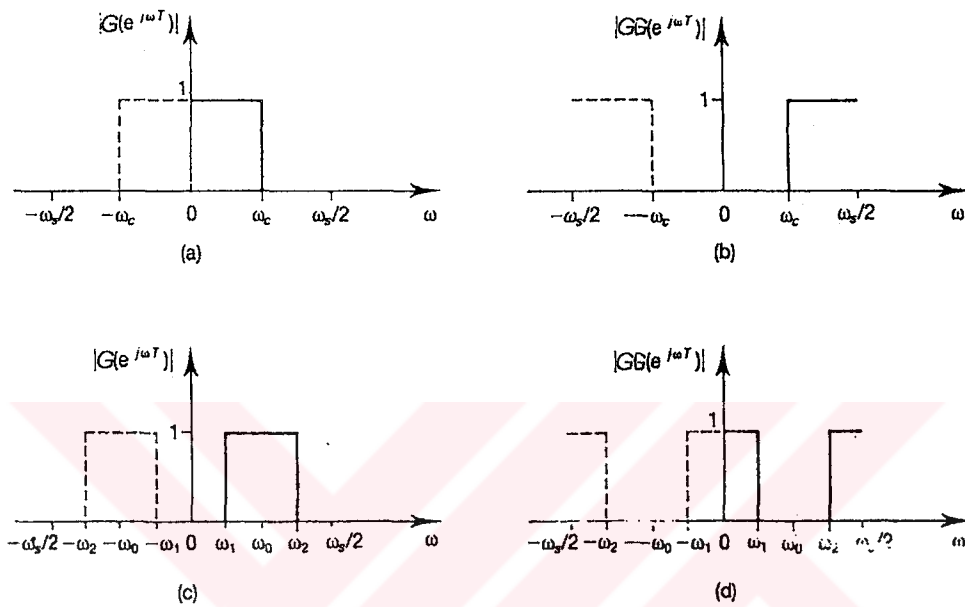


Figure 3.1 Digital Filter Specifications a)LP b)HP c)BP d)BS

Hence from (3.1) and (3.2) we have

$$\left. \begin{aligned} g(i) &= a_i \quad 0 \leq i \leq m \\ g(i) &= 0 \quad i > m \end{aligned} \right\} \quad (3.3)$$

and so the coefficients a_i in the difference equation defining the filter are just the impulse response coefficients.

Let m be even, say $m=2N$, and write

$$G(e^{j\omega T}) = \sum_{k=0}^{2N} g(k)e^{-jk\omega T}$$

$$= e^{-jN\omega T} \left\{ g(0)e^{jN\omega T} + g(1)e^{j(N-1)\omega T} + \dots + g(N) + \dots + g(2N)e^{-jN\omega T} \right\} \quad (3.3)$$

Then if we impose the conditions

$$\left. \begin{aligned} g(0) &= g(2N) \\ g(1) &= g(2N-1) \\ &\vdots \\ g(N+1) &= g(N-1) \end{aligned} \right\} \quad (3.4)$$

then we obtain

$$G(e^{j\omega T}) = e^{-j\omega T} \left\{ g(N) + \sum_{i=0}^{N-1} g(i) \cos[(N-i)\omega T] \right\} \quad (3.5)$$

Since the expression in brackets is real the phase response of the filter is $-N\omega T$, which is a linear function of ω .

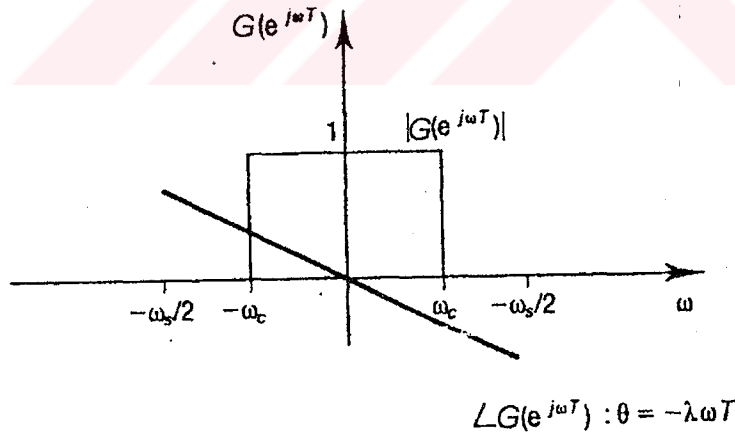


Figure 3.2 Ideal Lowpass Filter Response.

3.2.1 FIR Design

The complete design specification of an ideal lowpass filter is shown in Figure 3.2. The amplitude and phase responses are included on the same graph.

Since

$$G(e^{j\omega T}) = \begin{cases} e^{j\lambda\omega T} & |\omega| \leq \omega_c \\ 0 & \text{otherwise} \end{cases} \quad (3.6)$$

$g(n)$ will be

$$g(n) = \frac{\omega_c T}{\pi} \frac{\sin[(n-\lambda)\omega_c T]}{(n-\lambda)\omega_c T} \quad n = 0, \pm 1, \pm 2, \dots \quad (3.7)$$

If we regard the zero phase shift in the interval $[-\omega_s/2, \omega_s/2]$, then $\lambda=0$ and if for example $\omega_c = \omega_s/8$, then from 2.3.1 we have

$$g(n) = \frac{1}{4} \frac{\sin n\pi/4}{n\pi/4} = \frac{1}{4} \text{sinc}(n\pi/4) \quad (3.8)$$

where

$$\text{sinc}(x) = \frac{\sin x}{x} \quad (3.9)$$

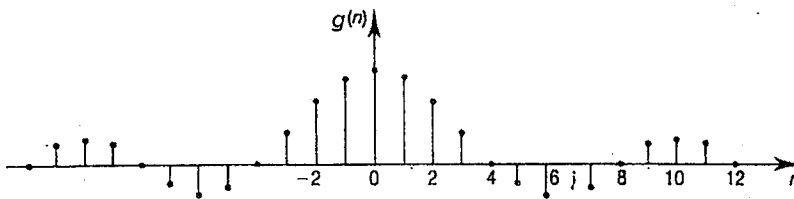


Figure 3.3 The Impulse Response of an Ideal (Zero Phase Shift) LP Filter

From the impulse response sequence of Equation 3.8 shown in Figure 3.3 we can see that an ideal LP filter with zero phase shift can never be achieved because of two reasons. First, the filter is noncausal since $g(n) \neq 0$ for some $n < 0$, and so

cannot be implemented in real time. Second, it is not a finite impulse response filter since $g(n) \neq 0$ for infinitely many n .

If we truncate the impulse response so that $g(n)=0$ for $|n|>10$, and introduce a delay of $\lambda=10$ in Equation 3.7, then we obtain the approximate impulse response $g_a(n)$ shown in Figure 3.4. By Equation 3.5 we see that the frequency response of $\{g_a(n)\}$ is given by

$$G_a(e^{j\omega T}) = e^{-j10\omega T} \left\{ g_a(10) + 2 \sum_{i=0}^9 g_a(i) \cos[(10-i)\omega T] \right\} \quad (3.10)$$

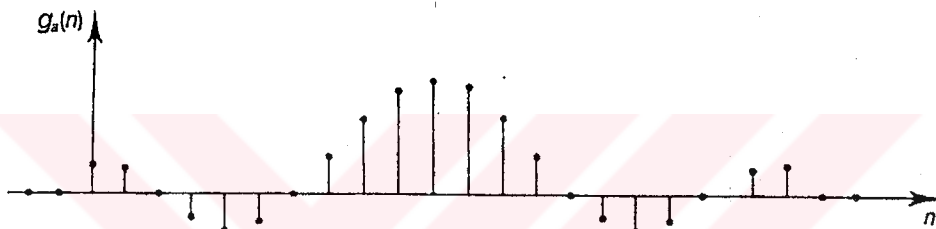


Figure 3.4 The Impulse Response of an Approximating Filter

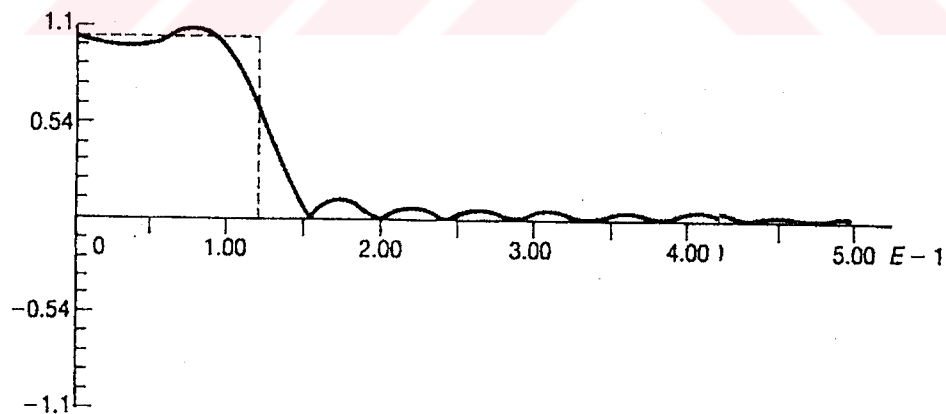


Figure 3.5 Ideal and Approximate Frequency Responses

3.2.2 Windows

In the pervious section, a realizable FIR filter is obtained by truncating the exact impulse response and introducing a delay. Truncating an impulse response

$g=\{g(n)\}$ outside the interval $|n|\leq N$ is equivalent to multiplying g by the sequence w_R defined by

$$w_R(n) = \begin{cases} 1 & \text{if } |n| \leq N \\ 0 & \text{if } |n| > N \end{cases} \quad (3.11)$$

Thus $g_a(n)$ is given by

$$g_a(n) = g(n)w_R(n) \quad (3.12)$$

$w_R(n)$ is called the rectangular window. More generally, if $\{w(n)\}$ is any sequence,

$$g_w(n) = g(n)w(n) \quad (3.13)$$

It can be shown that the frequency spectrum of the rectangular window has large side lobes of alternating sign [2]. In order to overcome this problem, a window function which has a large central lobe and small side lobes can be defined as

$$w_H(n) = \begin{cases} \alpha + (1-\alpha) \cos(\pi n / N) & \text{if } |n| \leq N \\ 0 & \text{if } |n| > N \end{cases} \quad (3.14)$$

$w_H(n)$ is called the von Hann window for $\alpha=0.5$ and for $\alpha=0.54$ is called the Hamming window.

The Blackman window, defined by

$$w_B(nT) = \begin{cases} 0.42 + 0.5 \cos(\pi n / N) + 0.08 \cos(2\pi n / N) & \text{if } |n| \leq N \\ 0 & \text{if } |n| > N \end{cases} \quad (3.15)$$

reduces the ripple compared with the last two windows, but a wider lobe width.

The Kaiser window, defined by

$$w_K(nT) = \begin{cases} I_0(\beta)/I_0(\alpha) & \text{if } |n| \leq N \\ 0 & \text{if } |n| > N \end{cases} \quad (3.16)$$

where α is a parameter and

$$\beta = \alpha \left\{ 1 - (n/N)^2 \right\}^{1/2} \quad (3.17)$$

a good trade-off between ripple ratio and primary lobe width.

Here $I_0(x)$ is the zeroth-order Bessel function of the first kind, defined by the series

$$I_0(x) = 1 + \sum_{k=1}^{\infty} \left\{ \frac{1}{k!} (x/2)^k \right\}^2 \quad (3.18)$$

3.2.3 Filter Transformations

We can obtain other types of filters such as HP, BP, BR from LP filters by applying some transformations given below

$$1) g_{HP}(n) = (-1)^n g_{LP}(n) \quad (3.19)$$

$$2) g_{BP}(n) = (2 \cos n\omega_0 T) g_{LP}(n) \quad (3.20)$$

ω_0 = center frequency

$$\omega_1 = \omega_0 - (\omega_c)_{LP}$$

$$\omega_2 = \omega_0 + (\omega_c)_{LP}$$

$$3) g(0)_{BS} = 1 - g(0)_{BP}, g(n)_{BS} = -g(n)_{BP}, n = +1, +2, \dots \quad (3.21)$$

3.3 Design of Recursive Filters

As explained in section 3.1, another method of filtering digital information is to use analog filters which are transformed to z-plane. In the following section some analog filters are given.

3.3.1 Analog Filter Design

In the following sections, first Lowpass Analog Filters which have the general form

$$|H(j\omega)|^2 = 1 + |K(j\omega)|^2 = 1 + \left| \frac{N(j\omega)}{D(j\omega)} \right|^2 \quad (3.22)$$

where $H(s)$ is the desired loss function and $K(s) = N(s)/D(s)$ is a rational function in s , will be explained. Then adaptation of LP filters to other types will be given.

General requirements for a LP filter are, passband from dc to ω_p , the stopband from ω_s to infinity, the maximum passband loss A_{\max} and the minimum stopband loss A_{\min} .

3.3.1.1 Butterworth Filter

The simplest lowpass Butterworth approximation is derived by assuming that $K(s)$ is a polynomial of the form:

$$K(s) = \epsilon \left(\frac{s}{\omega_p} \right)^n \quad (3.23)$$

where ϵ is a constant, n is the order of the polynomial, and ω_p is the desired passband edge frequency. The corresponding loss function is

$$|H(j\omega)| = \left| \frac{V_{IN}(j\omega)}{V_o(j\omega)} \right| = \sqrt{1 + \epsilon^2 \left(\frac{\omega}{\omega_p} \right)^{2n}} \quad (3.24)$$

At dc, from Equation 3.24 the loss is seen to be unity. The slope of the function at dc is obtained by expanding Equation 3.25 as a binomial series. Near $\omega=0$,

$$\epsilon^2 \left(\frac{s}{\omega_p} \right)^{2n} \ll 1$$

so

$$\left[1 + \epsilon^2 \left(\frac{s}{\omega_p} \right)^{2n} \right]^{1/2} = 1 + \frac{1}{2} \epsilon^2 \left(\frac{s}{\omega_p} \right)^{2n} - \frac{1}{8} \epsilon^4 \left(\frac{s}{\omega_p} \right)^{4n} + \frac{1}{16} \epsilon^6 \left(\frac{s}{\omega_p} \right)^{6n} + \dots \quad (3.25)$$

This expression shows that the first $2n-1$ derivatives are zero at $\omega=0$. Such $K(s)$ was chosen to be an n -th order polynomial, this is the maximum number of derivatives that can be made to zero. Thus the slope is as flat as possible at dc. For this reason the Butterworth approximation is known as the maximally flat approximation. From Equation 3.24 the loss is given by

$$A(\omega) = 10 \log_{10} \left[1 + \epsilon^2 \left(\frac{s}{\omega_p} \right)^{2n} \right] \text{ dB} \quad (3.26)$$

In particular the loss at ω_p is

$$A(\omega_p) = 10 \log_{10}(1 + \epsilon^2)$$

and at high frequencies ($\omega \gg \omega_p$) the loss asymptotically approaches

$$20 \log_{10} \epsilon \left(\frac{\omega}{\omega_p} \right)^n \quad (3.27)$$

In terms of normalized frequency

$$\Omega = \epsilon^{1/n} \left(\frac{\omega}{\omega_p} \right) \quad (3.28)$$

the loss is given by

$$A(\Omega) = 10 \log_{10}(1 + \Omega^{2n}) \quad (3.29)$$

In Figure 3.6 the loss of normalized Butterworth approximations are given for some n .

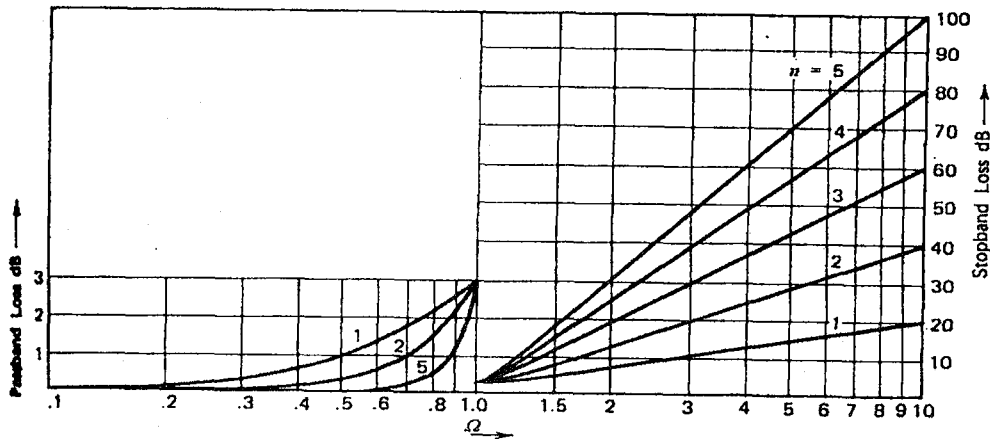


Figure 3.6 Loss of LP normalized Butterworth Approximations

Thus far the magnitude of the loss function, namely $|H(j\omega)|$, is explained. Derivation of $H(s)$ from the expression of $|H(j\omega)|$ is as follows:

$$|H(s)|^2 = H(s)H(-s) \quad (3.30)$$

where s is the normalized frequency variable $\Sigma + j\Omega$. Now the roots of $H(s)$ are the roots of $H(-s)$, reflected about the origin. Since the desired must have all its poles in the left half s plane, we must associate the left half plane roots of $|H(j\omega)|^2$ with $H(s)$ and right half plane roots of $|H(j\omega)|^2$ with $H(-s)$.

$$|H(j\Omega)|^2 = 1 + \Omega^{2n} = 1 + [-(j\Omega)^2]^n \quad (3.31)$$

Extending this to the s domain

$$|H(s)|^2 = 1 + (s^2)^n \quad (3.32)$$

The roots of $|H(j\omega)|^2$ are obtained by solving the equation

$$1 + (s^2)^n = 0 \quad (3.33)$$

The solution of this equation is:

$$s_k = \exp\left[\frac{j\pi\left(\frac{2k+n-1}{n}\right)}{2}\right] \text{ where } k=1,2,\dots,2n \quad (3.34)$$

These $2n$ roots are located on the unit circle and are equally spaced at π/n radian intervals. The s domain loss function is therefore given by

$$H(s) = \prod_j (s - s_j) \quad (3.35)$$

where s_j are the left half plane roots of Equation 3.33.

After the expression of $H(s)$ is found it must be denormalized by replacing s by $s \left(\frac{\varepsilon^{1/n}}{\omega_p} \right)$.

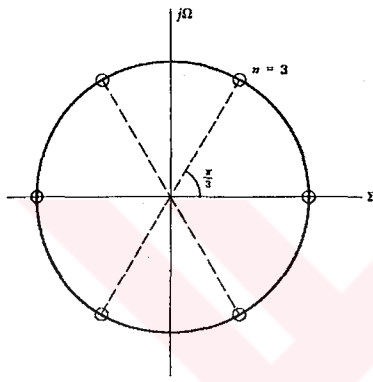


Figure 3.7 Roots of Third Order Butterworth

3.3.1.2 Chebyshev Filter

The main feature of the Butterworth Filters is, the loss is maximally flat at the origin. Thus the approximation to a flat passband is very good at the origin but it gets progressively poorer as ω approaches ω_p . Moreover, the attenuation provided in the stopband is less than the attainable using some other polynomial types, such as the Chebyshev. The increased stopband attenuation is achieved by changing the approximation conditions in the passband. The criterion used is to minimize the maximum deviation from the ideal flat characteristic shown in Figure 3.8. The Chebyshev polynomials are ideal for this purpose.

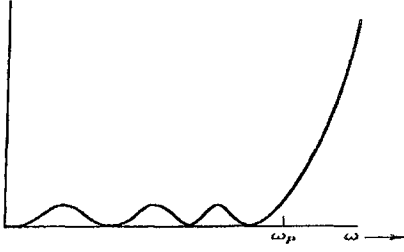


Figure 3.8 The Equiripple Passband Characteristic

The n -th order Chebyshev function $C_n(\Omega)$ is defined as

$$C_n(\Omega) = \cos(n \cos^{-1} \Omega) \quad |\Omega| \leq 1 \quad (3.36a)$$

$$= \cosh(n \cosh^{-1} \Omega) \quad |\Omega| > 1 \quad (3.36b)$$

where, Ω is the normalized frequency,

$$\Omega = \frac{\omega}{\omega_p} \quad (3.37)$$

The Chebyshev function can also be expressed as a polynomial in Ω , as shown in the following. From Equation 3.36a

$$C_{n+1}(\Omega) + C_{n-1}(\Omega) = \cos[(n+1) \cos^{-1} \Omega] + \cos[(n-1) \cos^{-1} \Omega] \quad (3.38)$$

$$2 \cos(\cos^{-1} \Omega) \cos(n \cos^{-1} \Omega) = 2\Omega C_n(\Omega) \quad (3.39)$$

which yields to the recursive relationship:

$$C_{n+1}(\Omega) = 2\Omega C_n(\Omega) - C_{n-1}(\Omega) \quad (3.40)$$

From Equation 3.36a we have

$$C_0(\Omega) = 1 \quad (3.41)$$

$$C_1(\Omega) = \Omega \quad (3.42)$$

The higher order polynomials are obtained from the recursive relationship of Equation 3.40.

A plot of the Chebyshev functions using the above polynomial form shows that they do indeed have an equiripple characteristic in the band $-1 \leq \Omega \leq 1$. Figure 3.9a-b

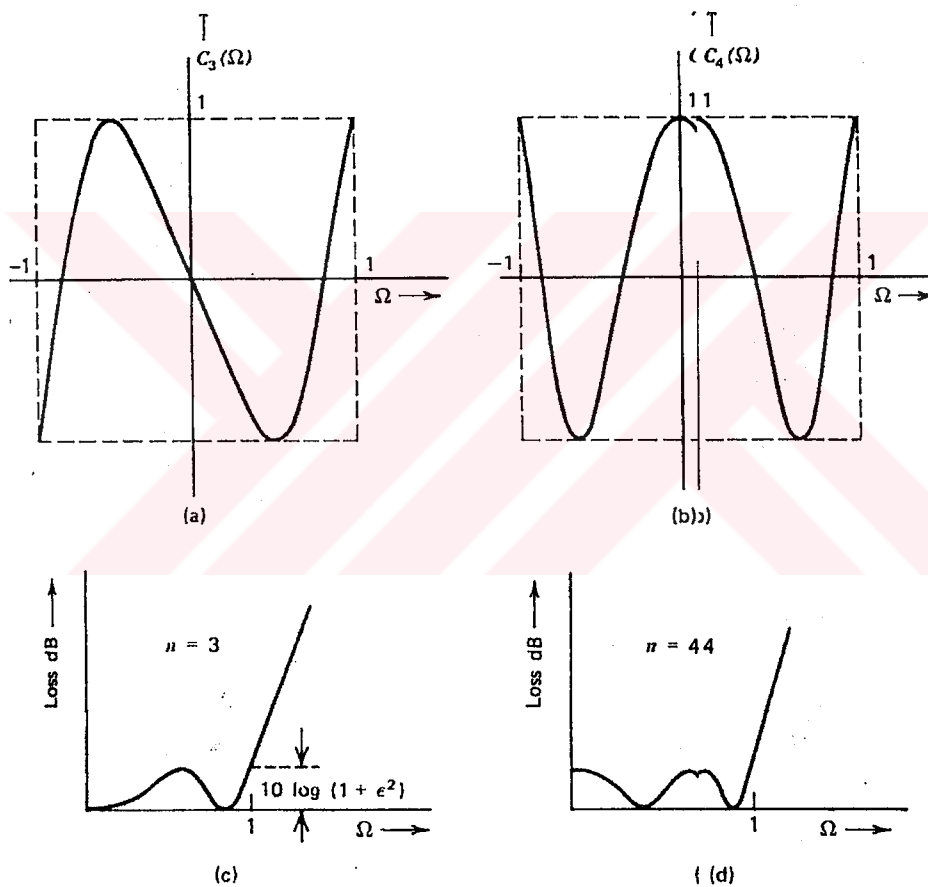


Figure 3.9 Plots of a) Third Order Chebyshev Function b) Fourth Order Chebyshev Function c) Third Order Chebyshev Approximation d) Fourth Order Chebyshev Approximation

The Chebyshev low-pass approximation function is obtained from the Chebyshev polynomials and is given by:

$$|H(j\omega)| = \left| \frac{V_{IN}(j\omega)}{V_O(j\omega)} \right| = \sqrt{1 + \epsilon^2 C_n^2(\Omega)} \quad (3.43)$$

The loss functions for $n=3$ and $n=4$ are sketched in Figure 3.9c and d. The functions ripple between a minimum of one and a maximum of $\sqrt{1+\epsilon^2}$ for $|\Omega| \leq 1$ and that the number of minima of $|H(j\Omega)|$ in the band $-1 \leq \Omega \leq 1$ equal to the order n . It can readily be shown that these properties apply to Chebyshev approximations of all orders [3].

The loss at the passband frequency $\omega=\omega_p$ is (Ω will equal to 1 and $C_n(1)=1$)

$$A_{\max} = 10 \log_{10}(1 + \epsilon^2) \quad (3.44)$$

One of the objectives in considering an equiripple passband was to improve on the stopband attenuation provided by the Butterworth filtering. The Butterworth attenuation for $\omega \gg \omega_p$ is approximately

$$20 \log_{10} \epsilon \left(\frac{\omega}{\omega_p} \right)^n \quad (3.45)$$

The Chebyshev attenuation is obtained from Equation 3.43, where $\omega \gg \omega_p$ (i.e., $\Omega \gg 1$) the term $\epsilon C_n(\Omega) \gg 1$. Thus

$$A(\Omega) \Big|_{\Omega \gg 1} \cong 20 \log_{10} \epsilon C_n(\Omega) \quad (3.46)$$

From the recursive formula, for $\Omega \gg 1$

$$C_n(\Omega) \cong 2^{n-1} \Omega^n \quad (3.47)$$

Using this expression, (3.46) reduces to

$$A(\Omega)|_{\Omega \gg 1} = A\left(\frac{\omega}{\omega_p}\right)\bigg|_{\frac{\omega}{\omega_p} \gg 1} = 20 \log_{10} \left[\varepsilon \left(\frac{\omega}{\omega_p}\right)^n 2^{n-1} \right] \quad (3.48)$$

Comparing Equations 3.45 and 3.48 it is seen that the Chebyshev approximation provides

$$20 \log(2)^{n-1} = 6(n-1) \text{ dB} \quad (3.49)$$

more attenuation than a Butterworth of the same order. Therefore for the same loss requirements the Chebyshev approximation will usually require a lower order than the Butterworth.

To find the roots of $H(s)$ same technique is used as in the Butterworth case. The roots are found by first evaluating the roots of $|H(s)|^2$, where

$$|H(s)|^2 = 1 + \varepsilon^2 C_n^2(\Omega) \bigg|_{\Omega=s/j} \quad (3.50)$$

The roots of the function can be shown to be [3]

$$s_k = \sigma_k \pm \omega_k \quad k=0,1,2,\dots,2n-1 \quad (3.51)$$

where

$$\sigma_k = \pm \sin \frac{\pi}{2} \left(\frac{1+2k}{n} \right) \sinh \left(\frac{1}{n} \sinh^{-1} \frac{1}{\varepsilon} \right) \quad (3.52a)$$

$$\omega_k = \cos \frac{\pi}{2} \left(\frac{1+2k}{n} \right) \cosh \left(\frac{1}{n} \sinh^{-1} \frac{1}{\varepsilon} \right) \quad (3.52b)$$

As in the Butterworth approximation the n left half plane roots corresponding to negative σ , are associated with $H(s)$. Furthermore, from Equations 3.52a and b, it can easily be seen that

$$\left[\frac{\sigma_k}{\sinh\left(\frac{1}{n} \sinh^{-1} \frac{1}{\epsilon}\right)} \right]^2 + \left[\frac{\omega_k}{\cosh\left(\frac{1}{n} \sinh^{-1} \frac{1}{\epsilon}\right)} \right]^2 = 1 \quad (3.53)$$

which is the equation of an ellipse. Thus the roots of the Chebyshev approximation lie on an ellipse in the s plane, whose real and imaginary intercepts are indicated in Figure 3.10

The Chebyshev approximation can now be expressed in factored form, as

$$H(s) = \prod_j (s - s_j) \quad (3.54)$$

where s_j are the left half plane roots of Equation 3.50.

For the general LP filter $H(s)$ can be denormalized by replacing s by $\left(\frac{s}{\omega_p}\right)$

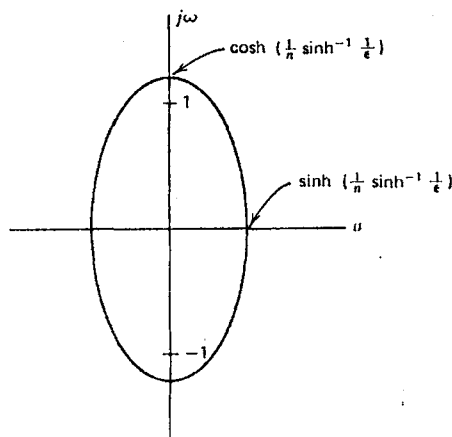


Figure 3.10 Locus of Roots of Chebyshev Approximation

3.3.1.3 Bessel Filter

Thus far, gain (loss) characteristics odd filter functions are discussed, no attention is paid to their phase and delay characteristics.

The magnitude delay of a fourth-order Chebyshev filter function ($A_{\max}=0.5$ dB), are sketched in Figure 3.11a and b. The delay characteristic in the passband is far from flat, the high frequencies being delayed much more than the low frequencies. Considering the response to the rectangular step input shown in Figure 3.11c, the high frequencies are expected to appear at the output of the filter later than the low frequencies. Since the high frequencies control the sharp rising edge of the step, the rise time of the pulse will be increased as indicated in Figure 3.11d. When the high frequencies arrive at the output they show up as a high frequency ringing in the step response. Thus, it can be seen that this Chebyshev filter function would greatly deteriorate the time response of digital signals.

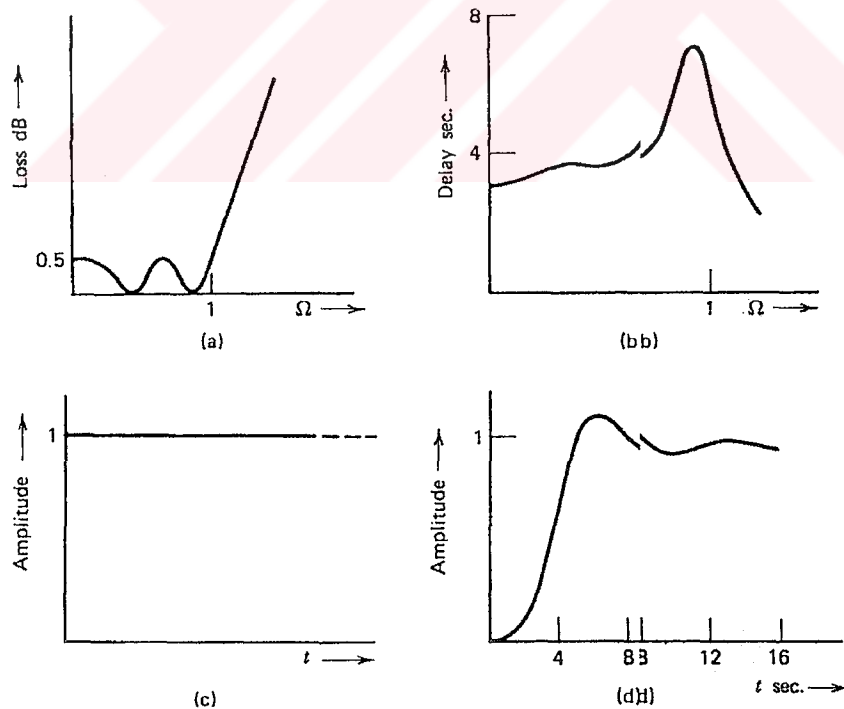


Figure 3.11 Characteristics of a Fourth Order Chebyshev ($A_{\max}=0.5$ dB); a) Loss b) Delay c) Step Input d) Step Response

In the Butterworth Filter case, the magnitude characteristic is monotonic in the passband and the delay is relatively flat. Figure 3.12a and b show the magnitude and delay characteristics of a fourth order Butterworth ($A_{\max}=3$ dB). The step response, shown in Figure 3.12c, has less ringing and the rise time is smaller than in the Chebyshev case. It can be observed that the smoother the magnitude characteristic the flatter is the delay characteristic. However, the smoother magnitude characteristic of the Butterworth approximation provides much less stopband attenuation than the equiripple Chebyshev approximation.

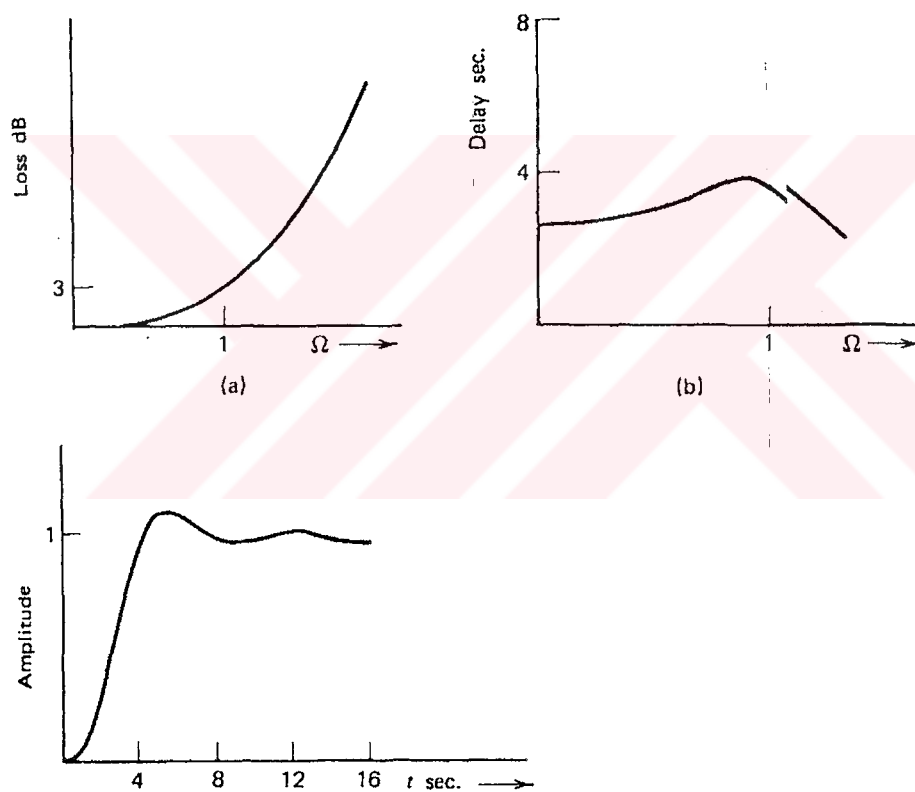


Figure 3.12 Characteristics of a Fourth Order Butterworth ($A_{\max}=3$ dB); a) Loss b) Delay c) Step Response

In the Bessel approximation the goal is to obtain as flat a delay characteristic as possible in the passband. The loss function for the ideal delay characteristic is [2]

$$H(s) = e^{sT_0} \quad (3.55)$$

The Bessel approximation is a polynomial that approximates this ideal characteristic. In this approximation the delay at the origin is maximally flat, that is, as many derivatives as possible are zero at the origin. It is convenient to consider the approximation of the normalized function, with the dc delay $T_0 = 1$ second, that is,

$$H(s) = e^s \quad (3.56)$$

It can be shown that [3] the Bessel approximation to this normalized function is

$$H(s) = \frac{B_n(s)}{B_n(0)} \quad (3.57)$$

where $B_n(s)$ is the n -th order Bessel polynomial which is defined by the following recursive equation

$$B_0(s) = 1 \quad (3.58)$$

$$B_1(s) = s + 1 \quad (3.59)$$

and

$$B_n(s) = (2n-1)B_{n-1}(s) + s^2B_{n-2}(s) \quad (3.60)$$

Using this recursion formula the higher-order approximations of e^s can be found.

The loss and delay of the Bessel approximations ($n=1$ to 5) are sketched in Figure 3.13 and 3.14, respectively. In these figures the normalized frequency Ω is related to ω by

$$\Omega = \omega T_0 \quad (3.61)$$

Figure 3.14 shows that the higher the order n , the wider is the band of frequencies over which the delay is flat. The delay characteristics of the Bessel approximation are far superior to those of the Butterworth and the Chebyshev. As a result, the step response (Figure 3.15c) is also superior, having no overshoot. However, the flat delay is achieved at the expense of the stopband attenuation which, for the Bessel approximation, is even lower than for the Butterworth.

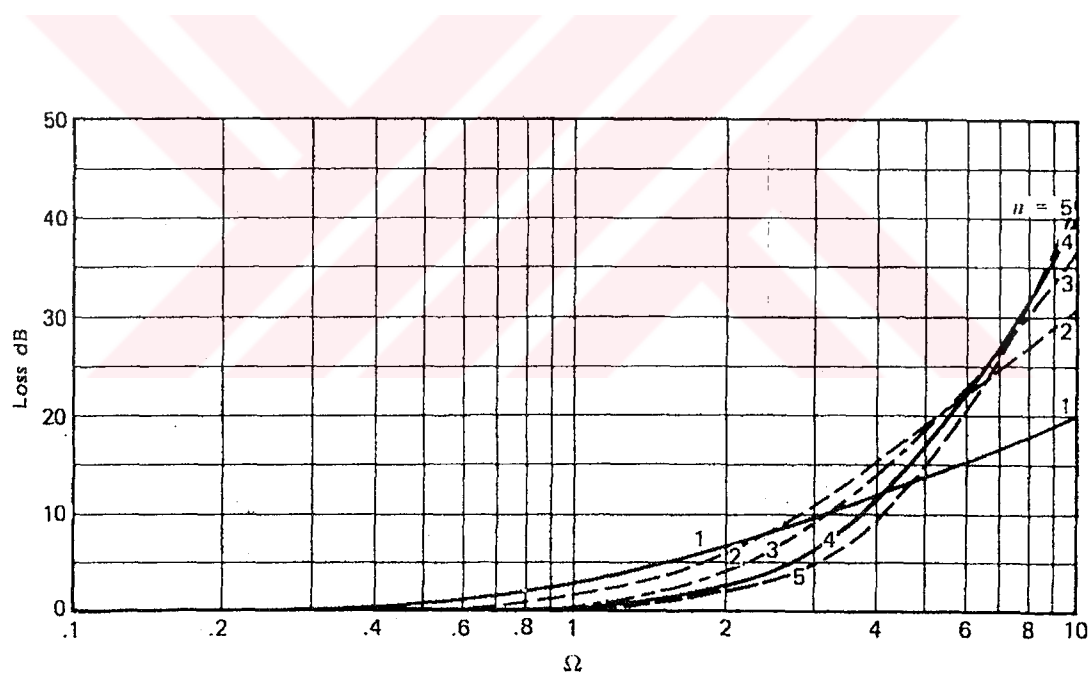


Figure 3.13 Loss of LP Bessel Approximations

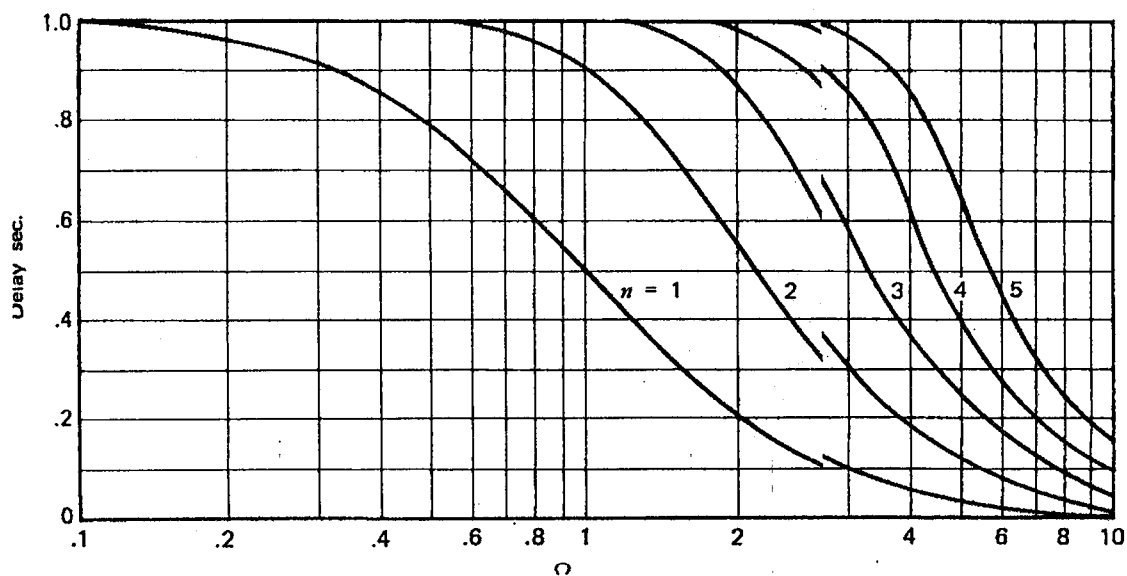


Figure 3.14 Delay of LP Bessel Approximations

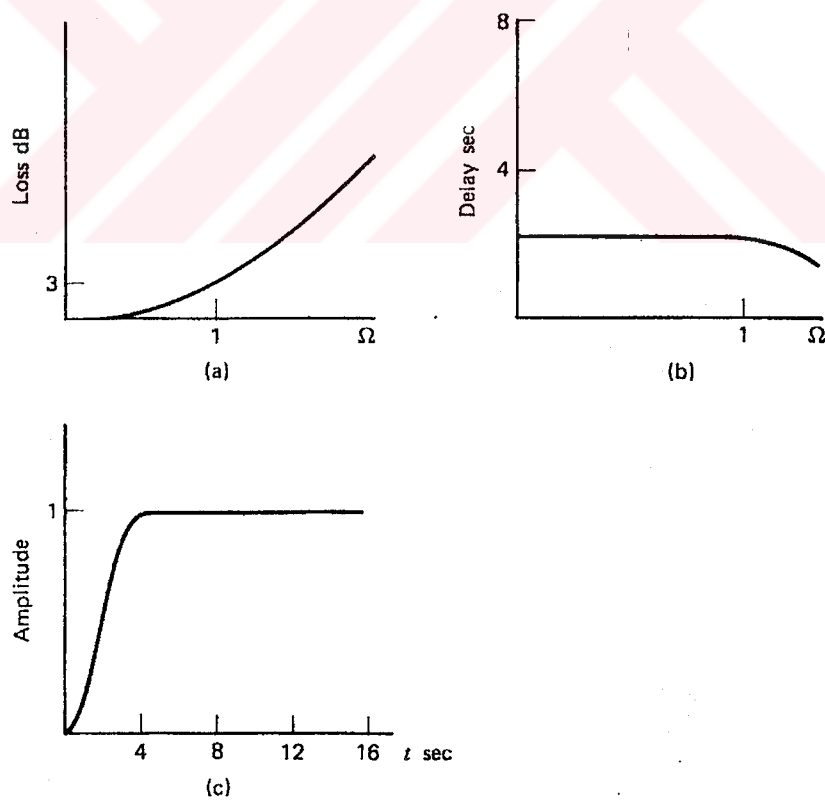


Figure 3.15 Characteristics of a Fourth Order Bessel Approximation ($A_{\max} = 3$ dB); a) Loss b) Delay c) Step Response

3.3.1.4 Frequency Transformations

In the last few sections, LP filter approximations are explained. These can be adapted to high-pass, symmetrical band-pass and symmetrical band reject filters. A block diagram of the steps in the approximation of these filters is shown in Figure 3.16.

The first step is to translate the given HP, BP or BR requirement to a related low-pass requirement by using a frequency transformation function. The resulting low-pass requirement is then approximated using the methods described in the pervious sections. Finally the low-pass approximation function is transformed to the desired HP, BP, or BR approximation function.

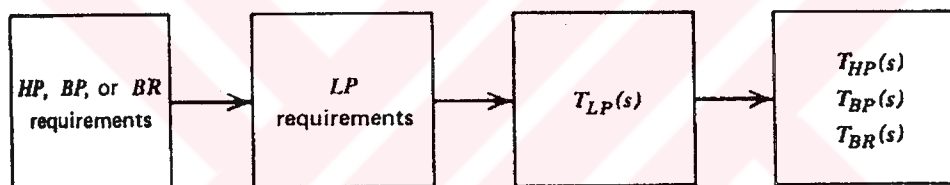


Figure 3.16 Block Diagram of the Frequency Transformation Procedure

The transformation functions are as follows:

High-pass Filters:

$$T_{HP}(s) = T_{LP}(S) \Big|_{S=\omega_p/s} \quad (3.62)$$

Band-pass Filters:

First the normalized LP requirements are characterized by

$$A_{\max}, A_{\min}, \Omega_p=1, \Omega_s=(\omega_4-\omega_3)/(\omega_2-\omega_1)$$

where the $\omega_4, \omega_3, \omega_2, \omega_1$ are as described in Figure 3.17

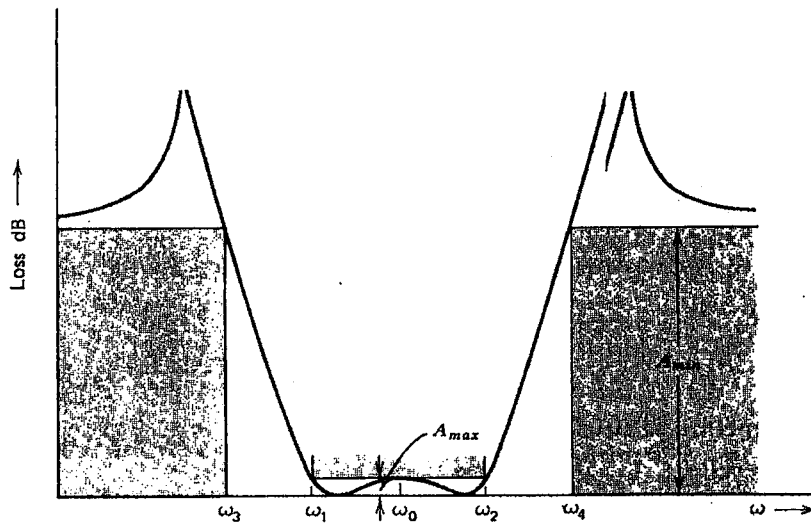


Figure 3.17 A typical Band-pass Function

Then, the required band-pass function is obtained by

$$T_{BP} = T_{LP}(S) \Big|_{S=(s^2 + \omega_0^2)/Bs} \quad (3.63)$$

where

$B = \omega_2 - \omega_1$ is the passband width

$\omega_0 = \sqrt{\omega_1 \omega_2}$ is the center (geometric mean) of the passband.

Band-Reject Filters:

To realize the BR requirements shown in Figure 3.18, we first approximate the LP requirements characterized by

$$A_{\max}, A_{\min}, \Omega_p = 1, \Omega_s = (\omega_2 - \omega_1) / (\omega_4 - \omega_3)$$

After this LP requirements are approximated, the desired BR filter is found by

$$T_{BR} = T_{LP}(S) \Big|_{S=Bs/(s^2 + \omega_0^2)} \quad (3.64)$$

where

$B = \omega_2 - \omega_1$ is the passband width

$\omega_0 = \sqrt{\omega_1 \omega_2}$ is the center (geometric mean) of the passband.

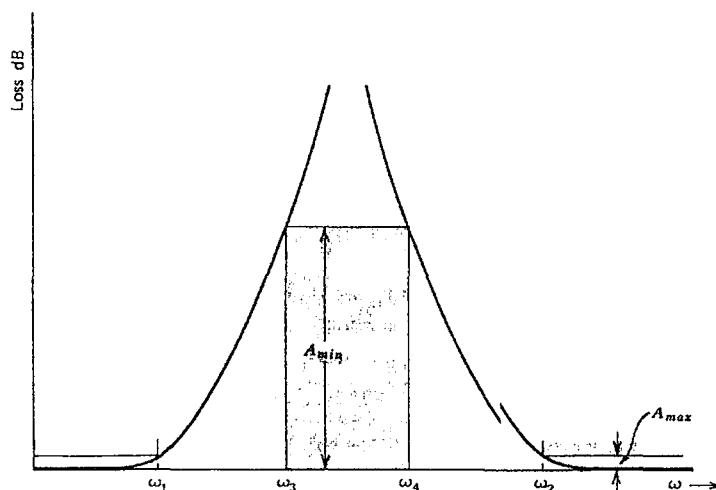


Figure 3.18 A typical Band-reject Function

3.3.1.5 s- to z-Domain Transformations

As mentioned earlier, the second way of obtaining digital filters is finding an appropriate transformation from s to z domain and substituting this for s in the analog filter transfer functions. One major drawback with this method, compared with the nonrecursive design, is that because a linear phase characteristic can not be obtained there is bound to be some distortion of the input signal.

In the following sections, some transformations from s- to z-domain will be given.

3.3.1.5.1 The Impulse-Invariant Method

In this method the impulse response coefficients of the digital filter corresponds to the impulse response coefficients of the chosen analog filter at the sampling times.

Since this introduces aliasing into the filter, this method is mostly applied to all pole filters, such as Butterworth, Bessel or Chebyshev. For any such filter, its transfer function can be written in the form (assuming simple poles)

$$G_A(s) = \sum_{i=1}^n \frac{A_i}{s - p_i} \quad (3.65)$$

Hence,

$$g_A(t) = L^{-1}\{G_A(s)\} = \sum_{i=1}^n A_i e^{p_i t} \quad (3.66)$$

Sampling g_A ,

$$g_A(kT) = \sum_{i=1}^n A_i e^{p_i kT} \quad (3.67)$$

and so the corresponding digital filter $G_D(z)$ is given by

$$G_D(z) = Z\{g_A(kT)\} = \sum_{i=1}^n \frac{A_i z}{z - e^{Tp_i}} \quad (3.68)$$

From Equation 3.68 it can be seen that a pole p_i of the analog filter gives rise to a pole e^{Tp_i} of the digital filter and so a realizable analog filter leads to a realizable digital filter. Since G_A is band limited (to a reasonable approximation) then $G_D(e^{j\omega T})$ approximates $G_A(j\omega)$ in the interval $[-\pi/T, \pi/T]$ and so the amplitude and phase responses of the analog filter are preserved under this transformation.

3.3.1.5.2 Modified Impulse Invariant Method

If an analog filter is given in the form

$$G_A(s) = \frac{KN(s)}{D(s)} = K \frac{\prod_{i=1}^m (s - z_i)}{\prod_{j=1}^n (s - p_j)} \quad (3.69)$$

where we have $m=n$, then $G_A(s)$ can be written as

$$G_A(s) = K G_{A1}(s) / G_{A2}(s) \quad (3.70)$$

where

$$G_{A1}(s) = \frac{1}{D(s)}, \quad G_{A2}(s) = \frac{1}{N(s)}$$

$$G_{A1}(s) = \sum_{j=1}^n \frac{A_j}{s - p_j}, \quad G_{A2}(s) = \sum_{i=1}^m \frac{B_i}{s - z_i} \quad (3.80)$$

Applying the transformation Equation 3.68

$$G_{D1}(z) = \sum_{j=1}^n \frac{A_j z}{z - e^{T p_j}}, \quad G_{D2}(z) = \sum_{i=1}^m \frac{B_i z}{z - e^{T z_i}} \quad (3.81)$$

The overall transfer function of the approximating digital filter is then

$$G_D(z) = \frac{G_{D1}(z)}{G_{D2}(z)} \quad (3.82)$$

A problem remains, however; some of the poles of 3.82 may be unstable because of the zeros of G_{D2} . This can be remedied easily, since if p_i is a real pole, then

CHAPTER FOUR

TWO DIMENSIONAL SIGNAL FILTERING

4.1 Introduction

The principle objective of enhancement techniques is to process an image so that the result is more suitable than the original image for a specific application. The word “specific” is important, because it establishes at the outset that the techniques discussed are very much problem oriented.

The approaches discussed in this chapter fall into two broad categories. Spatial domain methods and frequency domain methods. The spatial domain refers to the image plain itself and approaches this category based on direct manipulation of samples in an image. Frequency domain processing techniques are based on modifying the Fourier transform of an image. Enhancement techniques based on various combinations of methods from these two categories are not unusual.

4.2 Background

In this section, the fundamental ideas underlying and relating these two approaches will be explained.

4.2.1 Spatial Domain Methods

The term spatial domain refers to the aggregate of samples composing an image, and spatial domain methods are procedures that operate directly on these samples. Image processing functions in the spatial domain may be expressed as

$$g(x,y) = T[f(x,y)] \quad (4.1)$$

where $f(x, y)$ is the input image, $g(x, y)$ is the processed image and T is the function operating on f .

T may operate only on the point (x, y) of image f or it may operate on the neighbouring samples taking point (x, y) as its center and forming a rectangle (sometimes an approximation to a circle) called a subimage around it by the neighbouring samples as shown in Fig 4.1

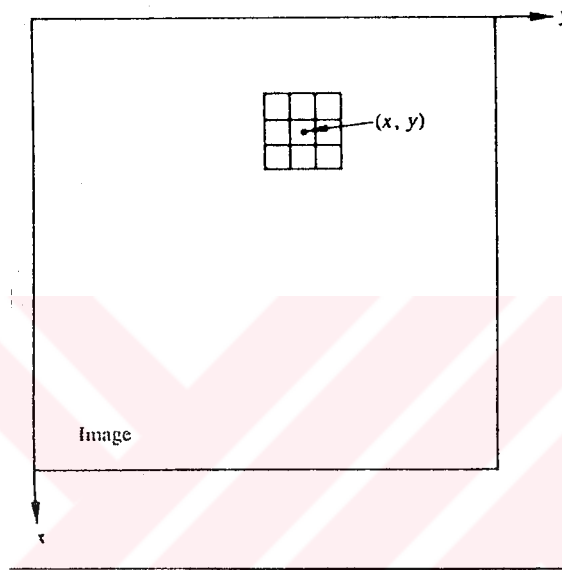


Figure 4.1 A 3x3 Neighbourhood about a Point (x, y) in an Image

Then subimage center is moved from sample to sample applying the operator T on each sample. The simplest form of T is when the neighbourhood is 1×1 . In this case the resultant image g depends only on the value of f at (x, y) , and T becomes a grey-level transformation (also called mapping) function of the form

$$s = T(r) \quad (4.2)$$

where t and s are value levels of $f(x, y)$ and $g(x, y)$ respectively.

If for example, if $T(r)$ has the form shown in Figure 4.2 then T has the effect of increasing the contrast on the image by darkening the value levels that have a value less than m ($r < m$) and lightening the value levels that are higher than m ($r > m$)

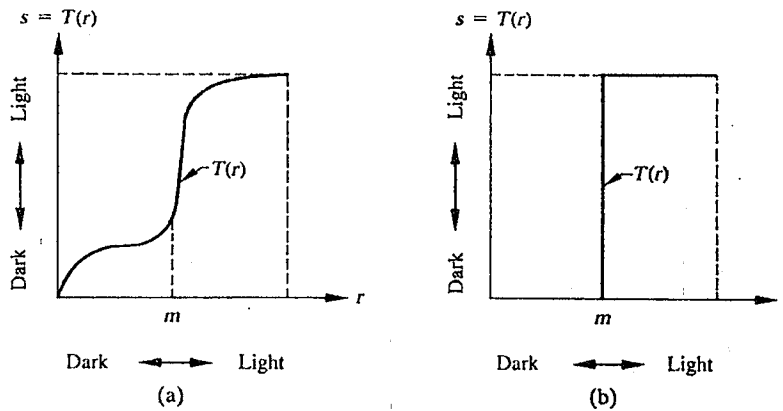


Figure 4.2 Grey-level Transformation Functions for Contrast Enhancement

In the limiting case shown in Figure 4.2 $T(r)$ produces two level (binary) image. Some fairly simple, yet powerful, processing approaches can be formulated with value level transformations. Because enhancement at any point in an image depends only on the value level at that point, techniques in this category often are referred to as *point processing*.

When larger neighbourhoods are used to allow a variety of processing functions that go beyond just image enhancement, operations used are called *masks* (also referred to as templates, windows or filters). Basically, a mask is small (say, 3x3) 2-D array, such as the one shown in Fig 4.1 in which the values of the coefficients determine the nature of the process, such as image sharpening. Enhancement techniques based on this type of approach often are referred to as *mask processing* or *filtering*.

4.2.2 Frequency domain methods

The foundation of frequency domain techniques is the convolution theorem.

$$g(x, y) = h(x, y) * f(x, y) \quad (4.3)$$

$$G(u, v) = H(u, v)F(u, v) \quad (4.4)$$

where $G(u, v)$, $H(u, v)$ and $F(u, v)$ are the Fourier transforms of $g(x, y)$, $h(x, y)$ and $f(x, y)$ respectively.

$H(u, v)$ is called the transfer function of the proceed and in optics it is called the optical transfer function. It is the system response to a unit impulse (that is a point of light) and according to linear system theory, a linear, position invariant system (linear time invariant system -if the input varies according to time-) is completely specified by its response to an impulse. $h(x, y)$ is the inverse transform of $H(u, v)$ and called the impulse response in the terminology of linear system theory. In optics $h(x, y)$ as called the point spread function. This name is based on the optical phenomenon that the impulse correspond to a point of light and that an optical system responds by blurring (spreading) the point with the degree of blurring being determined by the quality of the optical components. Thus the optical transfer function and the point spread function of a linear system are Fourier transforms of each other. Equation 4.3 describes a spatial process that is analogous to the use of the masks discussed in the previous selection. Infact the discrete convolution expression given in Equation 4.5 basically as a mathematical representation of mechanics involved in implementing the mask shifting process explained in Figure 4.1. For this reason $h(x, y)$ is often referred to as *spatial convolution mask*. Strictly speaking this term is not correct in general because convolution involves flipping one of the images about the origin. Using this name in connection with the masks in the previous section is correct only when the mask is symmetric about its origin.

$$f(x, y) * g(x, y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) g(x - m, y - n) \quad \begin{matrix} x = 0, 1, 2, \dots, M-1 \\ y = 0, 1, 2, \dots, N-1 \end{matrix} \quad (4.5)$$

Although it may already be obvious, we note that there is no general theory of image enhancement. When an image is processed for visual interpretation, the viewer is the ultimate judge of how well a particular method works. Visual evaluation of image quality is a highly subjective process, thus making the definition of a “good image” an elusive standard by which to compare algorithm performance. When the problem is one of processing images for machine perception, the evaluation task is somewhat easier. For example in dealing with a character recognition application the best image processing method would be the one yielding the best machine recognition results. However, even in

situations when a clear-cut criterion of performance can be imposed on the problem, the analyst usually is still faced with a certain amount of trial and error before being able to settle on a particular image processing approach.

4.3 Spatial Filtering

The use of spatial masks for image processing usually is called *spatial filtering* and the masks themselves are called *spatial filters*. In this section we consider linear and nonlinear spatial filters for image processing.

Linear filters are filters that satisfy

$$a\alpha_1 + b\alpha_2 \rightarrow \text{LinearFilter} \rightarrow a\beta_1 + b\beta_2 \quad (4.6)$$

where a and b are constants, α_1 and α_2 are two inputs and β_1 and β_2 are the two corresponding outputs. Consequently, linear systems have their transfer function and impulse or point spread function as their inverse Fourier transforms of each other. These filters can be classified into three as lowpass, bandpass and highpass filtering. A lowpass linear filter, eliminates the high frequency components in the frequency domain while leaving the low frequencies untouched. The highpass linear filter does the opposite. It eliminates the low frequency components and leaves the high frequency components untouched. Since the high frequency components of an image characterize edges and other sharp details, the result of eliminating them -that is the lowpass filtering- is image blurring. On the other hand the result of highpass filtering is the apparent sharpening of edges and other sharp details because it eliminates the low frequency components which characterize slowly varying characteristics of an image such as overall contrast and average intensity. Bandpass filtering removes selected frequency regions between low and high frequencies. These filters are used for image restoration and are seldom of interest in image enhancement.

Figure 4.3 shows cross sections of circularly symmetric lowpass, highpass and bandpass filters in the frequency domain and their corresponding spatial filters. The horizontal axes for the figures in the top row correspond to frequency and their counterparts in the bottom

row are spatial coordinates. The shapes in the bottom row are used as guidelines for specifying linear spatial filters.

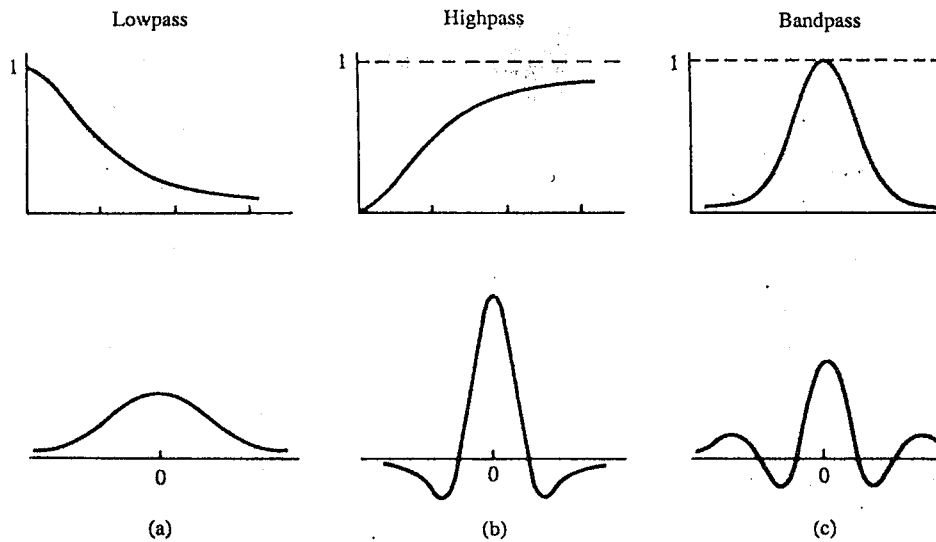


Figure 4.3 Top: cross Sections of Basic Shapes for Circularly Symmetric Frequency Domain Filters. Bottom: Cross Sections of Corresponding Spatial Domain Filters

Regardless of the type of linear filter used, the basic approaches to sum the products of predefined coefficients and the sample value level intensities under a mask at a specific location in an image. For example, for an 3x3 mask

$$R = w_1 z_1 + w_2 z_2 + \dots + w_9 z_9 \quad (4.7)$$

is calculated with the center of the mask placed at (x, y) and the value level of the value of sample (x, y) is replaced by R. The mask is then moved to the next sample location. This process is repeated until all the sample levels are replaced by their corresponding R value.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Figure 4.4 A 3x3 Mask with Arbitrary Coefficients (Weights).

Usually a new image with R values is created, so that the original sample values are used -not the altered ones- during each R computation.

Nonlinear spatial filters also operate on neighbourhoods. In general, however, their operation is based directly on the values of the samples in the neighbourhood under consideration and they do not explicitly use coefficients in the manner described in Equation 4.7. Noise reduction can be effectively achieved by nonlinear filters.

4.3.1 Smoothing Filters

Smoothing filters are used for blurring and noise reduction. There are both linear and nonlinear smoothing filters.

4.3.1.1 Lowpass Filtering

The shape of the impulse response needed to implement a lowpass (smoothing) spatial filter indicates that the filter has to have all positive coefficients. (See Figure 4.3a). Although the spatial filter shape shown in Figure 4.3a can be modelled by, say, a sampled Gaussian function, the key requirement is that all the coefficient be positive. The simplest arrangement for a 3x3 spatial filter would be a mask in which all coefficients have a value 1. In this case R will be the sum of values of all the neighbouring samples, but this sum may be larger than the expected maximum value level value, so the sum is divided by 9 which gives the average of all samples in the area of the mask. For this reason, the masks which have 1 as their coefficients are called *neighbourhood averaging*. In this filtering, as the dimensions of the mask increases, blurring increases.

4.3.1.1 Median Filtering

As mentioned above, the result of lowpass spatial filtering is blurring of edges and other sharp details. If the objective is to achieve noise reduction rather than blurring, median filters are used. That is the value level of each sample is replaced by the median of its neighbourhood.

The median m of a set of values is such that half of the values in the set are less than m and half are greater than m . For example, for a 3×3 mask, if the neighbourhood has values (10, 15, 20, 20, 30, 15, 10, 30, 35), it is first put into ascending (or descending) order as (10, 10, 15, 15, 20, 20, 30, 30, 35). Then the median is found as the fifth neighbour as 20. (there are four values less than and greater than 20). Thus the principle function of median filtering is to force points with distinct intensities to be more like their neighbours, actually eliminating intensity spikes that appear isolated in the area of the filter mask.

4.3.2 Sharpening Filters

The principle objective of sharpening filters is to highlight fine detail in an image or enhance detail that has been blurred.

4.3.2.1 Basic Highpass Spatial Filtering

The shape of the impulse response needed to implement a highpass (sharpening) spatial filter indicates that the filter should have positive coefficients near its center, and negative coefficients in the outer periphery (See Figure 4.3b). For a 3×3 mask, choosing a positive value in the center location with negative coefficients in the rest of the mask meets this condition.

$$\frac{1}{9} \times \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Figure 4.5 A Basic Highpass Spatial Filter

Figure 4.5 shows the classic implementation of a 3x3 sharpening filter. Note that the sum of the coefficients is zero. Thus, when the mask is over an area of constant or slowly varying value level, the output of the mask is zero or very small. This filter also eliminates the zero frequency term which reduces the average value level in the image to zero, reducing significantly the global contrast of image.

Reducing the average value of an image to zero implies that the image must have some negative value levels. As we deal only with positive levels, the result of highpass filtering involve some form of scaling and/or clipping so that the value levels of the final result span the range $[0, L-1]$.

4.3.2.2 Highboost Filtering

A highpass filter may be computed as

$$\text{Highpass} = \text{Original} - \text{Lowpass}$$

and a highboost filter is defined as

$$\text{Highboost} = (A)\text{Original} - \text{Lowpass} \quad (4.8)$$

$$= (A-1)\text{Original} + \text{Original} - \text{Lowpass}$$

$$= (A-1)\text{Original} + \text{Highpass} \quad (4.9)$$

where A is called the amplification factor. When $A=1$, the result is a standard highpass filter. When $A>1$, the part of the original signal is added back to the highpass result, which restores partially the low frequency components lost in the highpass filtering operation. The result highboost looks more like the original image with relative degree of edge enhancement that depends on the value of A .

In terms of implementation, the preceding results can be combined by letting the center weight of the mask shown in Figure 4.6 be

$$w = 9A - 1 \quad A \geq 1 \quad (4.10)$$

$$\begin{array}{|c|c|c|} \hline 1^- & 1^- & 1^- \\ \hline 1^- & a & 1^- \\ \hline 1^- & 1^- & 1^- \\ \hline \end{array} \times \frac{6}{1}$$

Figure 4.6 Mask Used for High-Boost Spatial Filtering.

4.3.2.3 Derivative Filters

Averaging of samples over a region tends to blur detail in an image. As averaging is analogous to integration, differentiation can be expected to have the opposite effect and thus sharpen an image.

The most common method of differentiation in image processing applications is the gradient of f at coordinates (x, y) is defined as the vector

$$\nabla \vec{f} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \quad (4.11)$$

The magnitude of this vector,

$$\nabla f = \text{mag}(\nabla \vec{f}) = \left[\left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2 \right]^{1/2} \quad (4.12)$$

is the basis for various approaches to image differentiation. For an image as shown in Figure 4.7a where z 's are the level values, Equation 4.11 can be approximated at point z_5 in a number of ways. The simplest is to use the difference $(z_5 - z_8)$ in the x direction and $(z_5 - z_6)$ in the y direction, combined as

$$\nabla f \approx \left[(z_5 - z_8)^2 + (z_5 - z_6)^2 \right]^{1/2} \quad (4.13)$$

instead of square roots, absolute values can be used:

$$\nabla f \approx |z_5 - z_8| + |z_5 - z_6| \quad (4.14)$$

Another way to approximate Equation 4.12 is to use cross differences

$$\nabla f \approx \left[(z_5 - z_9)^2 + (z_6 - z_8)^2 \right]^{1/2} \quad (4.15)$$

$$\nabla f \approx |z_5 - z_9| + |z_6 - z_8| \quad (4.16)$$

Figure 4.7 b, c, d show masks to implement derivative filters

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

(a)

1	0
0	-1

0	1
-1	0

(b) Roberts

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

(d) Sobel

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

(c) Prewitt

Figure 4.7 A 3x3 Region of an Image and Various Masks Used to Compute the Derivative at Point Labelled z_5 .

4.4 Filtering in the Frequency Domain

Enhancement in the frequency domain in principle is straight forward. We simply compute the Fourier Transform of the image to be enhanced, multiply the result by a filter transfer function and take the inverse transform to produce the enhanced image. Lowpass, highpass and bandpass filtering which are explained before under spatial domain filtering are valid for frequency as well. In practice, small spatial masks are used considerably more than the Fourier transform because of their simplicity of implementation and speed of operation. However, for some problems, spatial techniques are not good.

The 2-D Fourier transform pair is

$$F(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \exp[-j2\pi(ux + vy)] dx dy \quad (4.17)$$

and

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u, v) \exp[j2\pi(ux + vy)] dx dy \quad (4.18)$$

4.4.1 Lowpass Filtering

As mentioned earlier, in lowpass filtering, the high frequency components which indicate sharp transitions (such as noise) and edges, are eliminated.

A 2-D ideal lowpass filter (ILPF) transfer function is

$$H(u, v) = \begin{cases} 1 & \text{if } D(u, v) \leq D_0 \\ 0 & \text{if } D(u, v) > D_0 \end{cases} \quad (4.19)$$

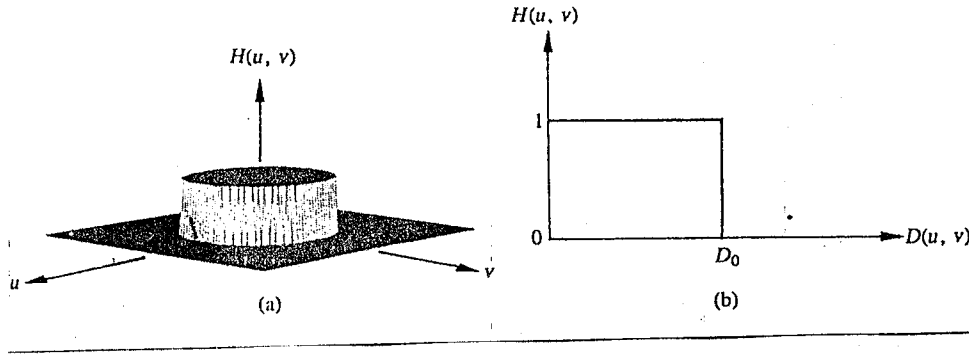


Figure 4.8 (a) Perspective Plot of an Ideal Lowpass Filter Transfer Function; (b) Filter Cross section

where, D_0 is a specified nonnegative quantity, $D(u, v)$ is the distance from point (u, v) to the origin of the frequency plane, that is

$$D(u, v) = (u^2 + v^2)^{1/2} \quad (4.20)$$

And the transformed image is

$$G(u, v) = H(u, v) F(u, v)$$

which is $g(x, y) = h(x, y) * f(x, y)$ in the spatial domain. By taking the inverse transform of $G(u, v)$, $g(x, y)$, the filtered image is found.

Butterworth filter

The transfer function of the Butterworth lowpass filter (BLPF) of order n and with cutoff frequency locus at a distance D_0 from the origin is defined by the relation

$$H(u, v) = \frac{1}{1 + [D(u, v) / D_0]^{2n}} \quad (4.21)$$

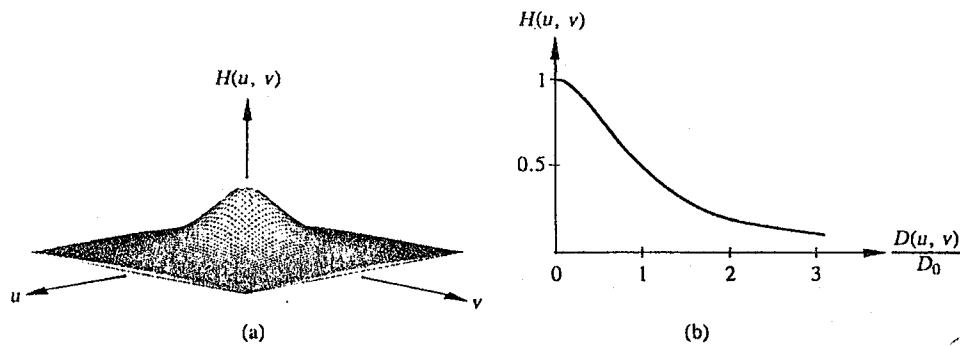


Figure 4.9 (a) A Butterworth Lowpass Filter; (b) Radial Cross Section for $n=1$.

4.4.2 Highpass Filtering

As explained earlier, edges and other abrupt changes in value levels are associated with high frequency components and image sharpening can be achieved by attenuating the low frequency components leaving the high frequency components untouched.

A 2-D ideal highpass filter (IHPF) has the transfer function $H(u, v)$ as

$$H(u, v) = \begin{cases} 0 & \text{if } D(u, v) \leq D_0 \\ 1 & \text{if } D(u, v) > D_0 \end{cases} \quad (4.22)$$

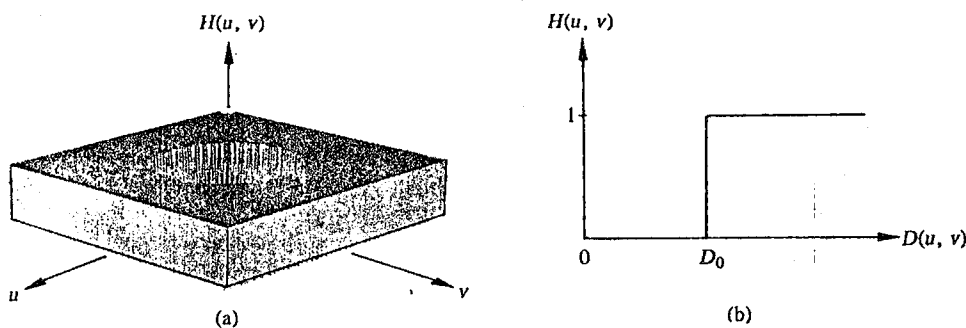


Figure 4.10 Perspective Plot and Radial Cross Section of Ideal Highpass Filter

Butterworth Filter

The transfer function of the Butterworth highpass filter (BHPF) of order n and with cutoff frequency locus at a distance D_0 from the origin is defined by the relation

$$H(u, v) = \frac{1}{1 + [D_0 / D(u, v)]^{2n}} \quad (4.23)$$

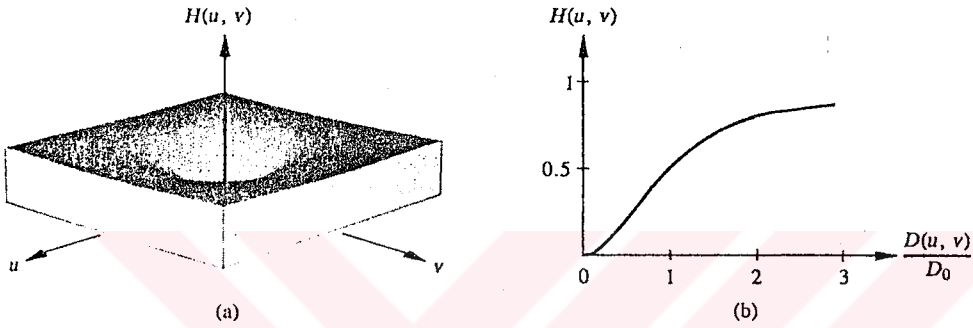


Figure 4.11 Perspective Plot and Radial Cross Section of Butterworth Highpass Filter for $n=1$.

In typical image enhancement, $f(x, y)$ is given and the goal after computation of $F(u, v)$ is to select $H(u, v)$ so that the desired image

$$g(x, y) = F^{-1}[H(u, v) \cdot F(u, v)] \quad (4.25)$$

exhibits some extended feature of $f(x, y)$. For instance if the edges are to be accentuated then the high frequency components of $F(u, v)$ must be emphasised since edges are the regions where intensive altering -changing of sample values occur.

CHAPTER FIVE

OCCAM FILTERS

5.1 Introduction

Recently Imai and Iri, Natarajan and Konstantinides presented an efficient algorithm that can both be used for compression and filtering of signals.

The essence of this algorithm is compressing the signal with a loss tolerance that is equal to the strength of the noise. They observed that the loss tends to cancel out the noise, with the cancellation depending on the compression achieved and how often the signal is sampled, and the decompressed output is closer to the noise-free signal than the noisy signal.

They explain why they have named the class of filters realizable by their technique as “Occam Filters” as “The essence of our technique is the principle of Occam’s razor (after English philosopher William of Occam): ‘The simplest explanation of the observed phenomenon is more likely to be correct.’” (Natarajan, 1995).

Actually the algorithm answers the question: Given a piecewise linear function $F:[0,1]$, $|F(x)-G(x)| \leq \epsilon$ and G consists of the fewest number of segments over all such functions. This result is also extended to univariable functions in higher dimensions and the curves in the plane. Two dimensional form of the algorithm will be given in later sections.

5.2 Noise Strength

As mentioned above, the strength of the noise $\|v\|$ is needed for this algorithm and how to find $\|v\|$ can be best explained by an experiment.

If we could get the noise v itself and run the algorithm on this noise sequence for various values of allowed loss ϵ and plot the compressed size against the logarithm of ϵ , which is the rate distortion plot of v , we would get the plot in Figure 5.1. When $\epsilon > \|v\|$, the noise sequence can be approximated by a constant because at such high loss tolerances the size of the compressed signal will decrease.

If we could run the algorithm on the noise-free sequence f , then we would get the rate distortion plot of f as shown in Figure 5.2 because, the less the allowed tolerance is the more the number of data samples is.

It must be noted that on the rate distortion plot of noise-free signal there is no edge as there is one on the rate distortion plot of the noise sequence, because the loss ϵ is never allowed to be more than the noise-free signal strength (the values of noise sequence values much smaller than noise-free sequence).

When the algorithm is applied to the noisy sequence $f+v$, the rate distortion plot in Figure 5.3 is the result. Because, when $\epsilon < \|v\|$, the noise dominates the signal f , but when $\epsilon > \|v\|$, the noise is regarded. At $\epsilon = \|v\|$ the plot makes a sharp knee point which is the point at which the second derivative is maximum.

As a result, the algorithm works as follows: First, run the algorithm with various values of ϵ on the noisy sequence $f+v$ and plot the compressed size against $\log(\epsilon)$. Next, find the knee point ϵ^* of the plot and then, run the algorithm on $f+v$ once more, but for this time use ϵ^* . The result is the compressed and filtered signal.

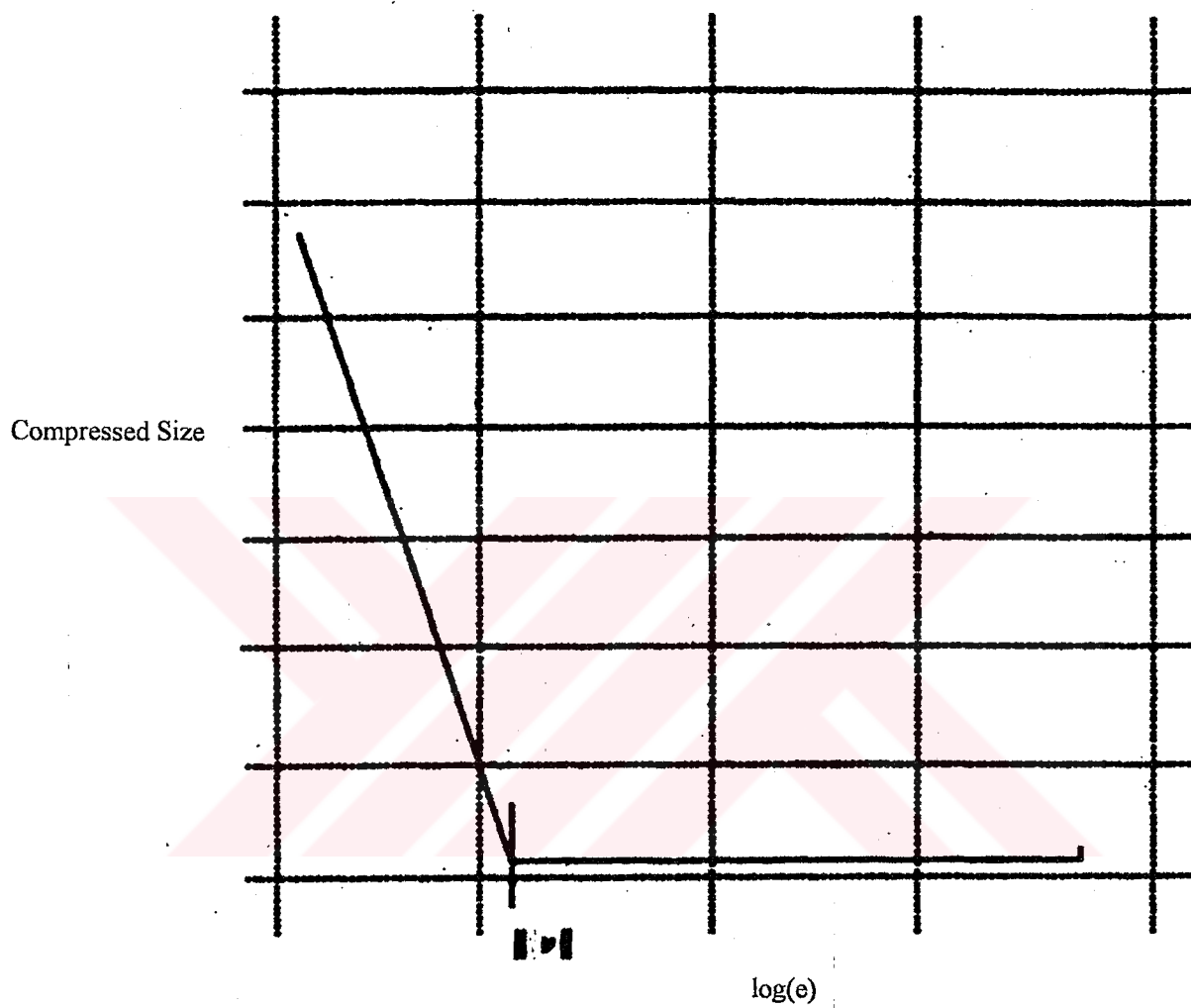


Figure 5.1 Illustrative Plot of Compressed Size Versus Allowed Loss for the Noise Sequence

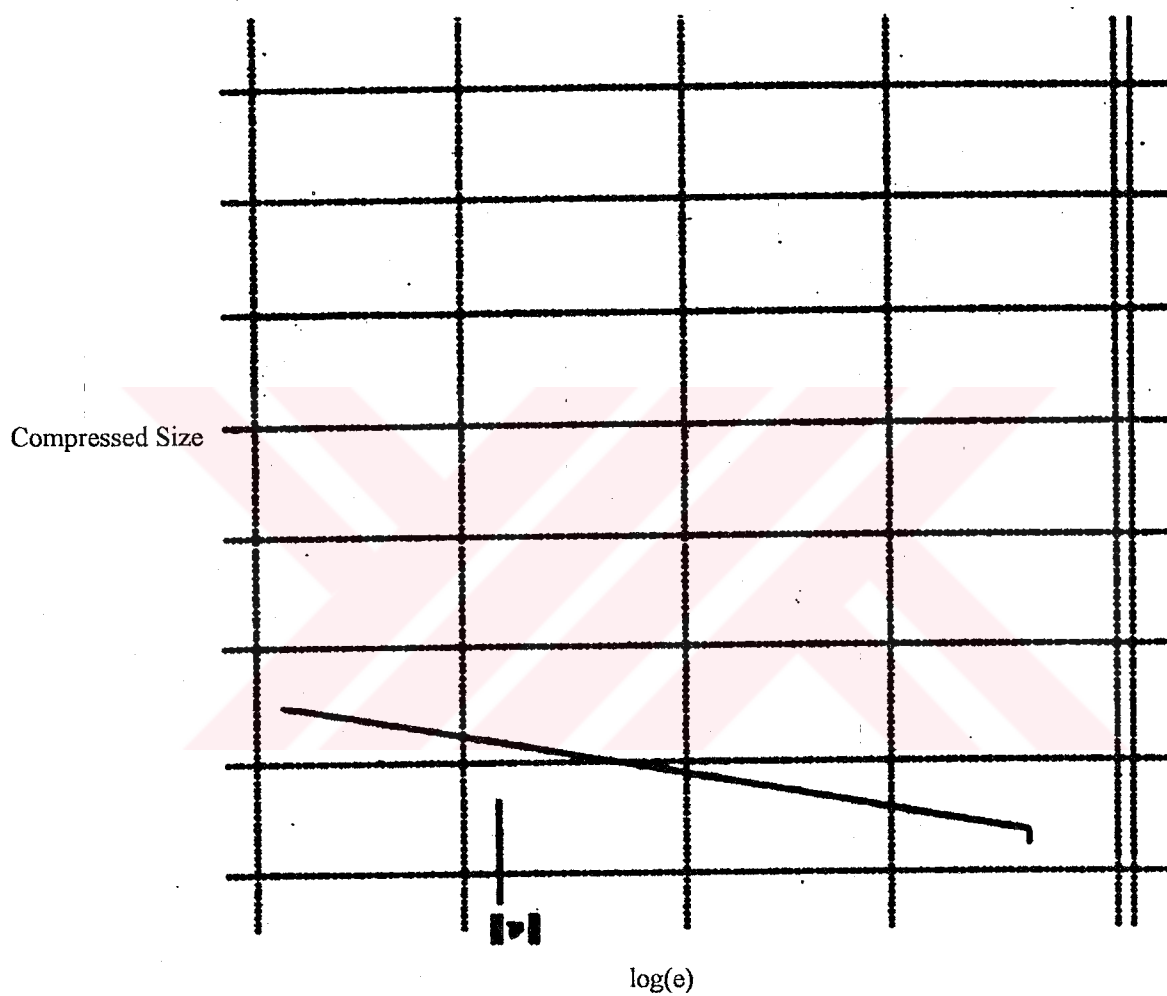


Figure 5.2 Illustrative Plot of Compressed Size Versus Allowed Loss for the Noise-Free Sequence

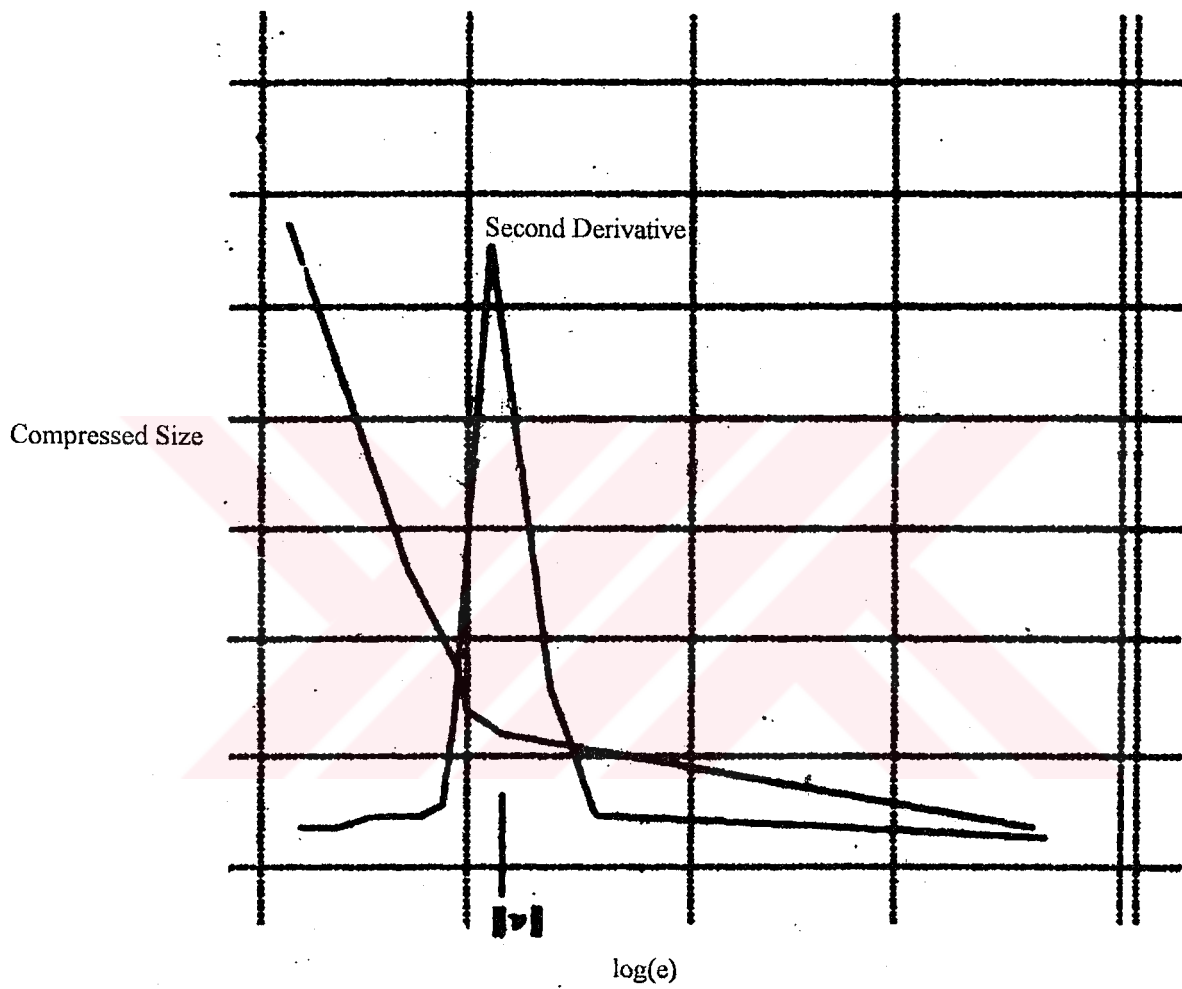


Figure 5.3 Illustrative Plot of Compressed Size Versus Allowed Loss for the Noisy Sequence with Second Derivative Plot

5.3 Description of the Algorithm

The algorithm calculates the upper and the lower envelope points (f^+ and f^-) by adding and subtracting ϵ from the sequence values and upper and lower tangents are drawn starting from (x_s, f_s) to the envelope points (x_{s+1}, f_{s+1}^+) and (x_{s+1}, f_{s+1}^-) , then it tests the next envelope points and tries to draw next tangent lines to these points. This is carried on iteratively until no tangent line exists. Then, the algorithm cuts the line in two from this point. Starts over from the first point of the second portion regarding the first part. From this portion only the first and the last points are recorded. the points in between are found during decomposition by linear interpolation.

The algorithm is:

input: list of points $\{(x_1, F(x_1)), (x_2, F(x_2)), \dots, (x_N, F(x_N))\}$, and $\epsilon > 0$.

begin

Let $x_s = x_1$; $f_s = F(x_1)$;

$f_1^+ = F(x_1) + \epsilon$; $f_1^- = F(x_1) - \epsilon$;

Construct the upper envelope $U = \{(x_s, f_s^+), (x_2, f_2^+), \dots, (x_N, f_N^+)\}$

and the lower envelope $L = \{(x_s, f_s^-), (x_2, f_2^-), \dots, (x_N, f_N^-)\}$

$p = 1$;

while U is not empty **do**

$k = s + 1$;

$a_h = (f_k^+ - f_s) / (x_k - x_s)$;

$a_l = (f_k^- - f_s) / (x_k - x_s)$;

$b_h = f_s - a_h x_s$;

$b_l = f_s - a_l x_s$;

$k = k + 1$;

while $f_k^+ \geq a_l x_k + b_l$ and $f_k^- \leq a_h x_k + b_h$ **do**

if $f_k^+ < a_h x_k + b_h$ **then**

$a_h = (f_k^+ - f_s) / (x_k - x_s)$;

$b_h = f_s - a_h x_s$;

if $f_k^- > a_l x_k + b_l$ **then**

$a_l = (f_k^- - f_s) / (x_k - x_s)$;

$b_l = f_s - a_l x_s$;

$k = k + 1$;

end

let $g_p(x) = a_l x + b_l$;

output the end point $(x_{k-1}, g_p(x_{k-1}))$;

$x_s = x_{k-1}$;

$f_s = g_p(x_{k-1})$;

$U = \{(x_s, f_s^+), (x_k, f_k^+), \dots, (x_N, f_N^+)\}$;

$L = \{(x_s, f_s^-), (x_k, f_k^-), \dots, (x_N, f_N^-)\}$;

$p = p + 1$;

end

end

CHAPTER SIX

APPLICATIONS

6.1 Introduction

In this chapter our first aim is to compare the performances of the filter types explained in Chapter 3 that of the Occam filter. Since noise is accepted to add high frequency components to the signal, lowpass filters are used for the first three filter types of:

- i) Butterworth Filter
- ii) Chebyshev Filter
- iii) Bessel Filter
- iv) Wiener Filter which has a transfer function as $H(\omega) = S(\omega) / [S(\omega) + N(\omega)]$ and is commonly used in broad band signals.

Our second aim is to find out if Occam filter compresses better than the other compression algorithms. Of the two compression experiments done, which have been described in Chapter 2, the most efficient one from each group is selected to compare with Occam Filter-Compression. The selected ones are:

- i) Run Length Cod.
- ii) Vector Quantizaion 2nd method (8-Level)
- iii) DCT Compression

The programs were created in MATLAB environment to handle large number of data easier and to use some of the matrix functions provided by MATLAB.

6.2 Experiments for Filters

The first experiment done consists of a broad band signal corrupted with uniformly distributed random noise. Here, the aim was to filter out the noise. The results are shown in Figures 6.1-6.12.

6.3 Experiments for Compression

The second experiment was done on an ECG signal of a healthy person taken at the Electronics and Electircs Department of 9 Eylul University. The results are shown in Figures 6.13-6.16.

The third experiment was done on a EMG signal of a patient at Ege University Hospital. The results are shown in Figures 6.17-6.20.

6.4 Results of the Filter Experiment

First, the noise-free signal function $y(t)$ as defined below is selected.

$$y(t) = \begin{cases} 0 & t \leq 0.02 \\ \sin\left(\frac{2\pi}{t}\right) & \text{otherwise} \end{cases} \quad (6.1)$$

Since this is a wide-band signal, it is difficult to filter it with classical filtering methods. 1000 uniformly spaced samples of $y(t)$ is shown in Figure 6.2a. Next a uniformly distributed random noise shown in Figure 6.1, which has values $[-0.1, +0.1]$ is added to form the noisy signal $-ny(t)-$ as shown in Figure 6.2b. Since noise adds high frequency components to signals, lowpass filter types were applied.

FIR Lowpass Filter:

- n is the filter order, $\max(n)$ is 330, W_n is the cutoff frequency. (scaled by taking the Nyquist frequency -which is the half of the sampling frequency 1000Hz- as 1.) Results are shown in Figure 6.2c and 6.3.

- $lp1(t)$: $n=300$, $W_n=0.2$
- $lp1(t)$: $n=300$, $W_n=0.7$
- $lp1(t)$: $n=2$, $W_n=0.2$
- $lp1(t)$: $n=2$, $W_n=0.7$

Butterworth Filter:

- The program is arranged so that it finds the best filter order for the given specifications.
- stopband attenuation is always taken as $20 \log_{10} N$, where N is the maximum Fourier coefficient of noise, as shown in Figure 6.4. The results are given in Figure 6.5.
- A_p is the passband attenuation.
- $f(t)$: $A_p=0.1\text{dB}$
- $f2(t)$: $A_p=1\text{dB}$
- $f3(t)$: $A_p=10\text{dB}$

Chebyshev Filter:

- The program is arranged so that it finds the best filter order for the given specifications. The filter is given in Figure 6.6 and the results are shown in Figure 6.7.
- stopband attenuation is always taken as $20 \log_{10} N$ where N is the maximum Fourier coefficient of noise.
- A_p and R_p are the passband attenuation and passband ripple respectively.
- $c1(t)$: $A_p=0.1$, $R_p=0.5$
- $c2(t)$: $A_p=1$, $R_p=0.5$
- $c1(t)$: $A_p=10$, $R_p=0.5$
- $c1(t)$: $A_p=0.1$, $R_p=0.1$
- $c1(t)$: $A_p=1$, $R_p=0.1$
- $c1(t)$: $A_p=10$, $R_p=0.1$

Bessel Filter:

- n is the filter order, $\max(n)$ is 330, W_n is the cutoff frequency. (scaled by taking the Nyquist frequency -which is the half of the sampling frequency 1000Hz- as 1. The filter is shown in Figure 6.9 and the results in Figure 6.10.
- $b1(t)$: $n=2$, $W_n=0.05$

- $b_1(t)$: $n=2$, $W_n=0.1$
- $b_1(t)$: $n=2$, $W_n=0.2$

Wiener Filter:

- the algorithm explained in Chapter 3 is followed. Result is given in Figure 6.12a

Occam Filter:

- Algorithm explained above is applied.
- PSD of noisy signal is as shown in Figure 6.11.
- ε is found to be 0.1002. Result is in Figure 6.12b.

6.5 Comparison of the Filter Types

Because the noise-free signal is a wideband signal and the lowpass filters of Butterworth, Chebyshev, Bessel and FIR filters are used, at high frequencies the filtered signals do not follow the noise-free signal at all. If high pass filters were used, opposite would have happened.

Another handicap with these type of filters and the Wiener filter, is that some information such as the cutoff frequencies, or the noise strength must be known in advance.

As a result, Occam filter outperforms the other types of signals when the signal is a broad band one and nothing is known about the signal beforehand.

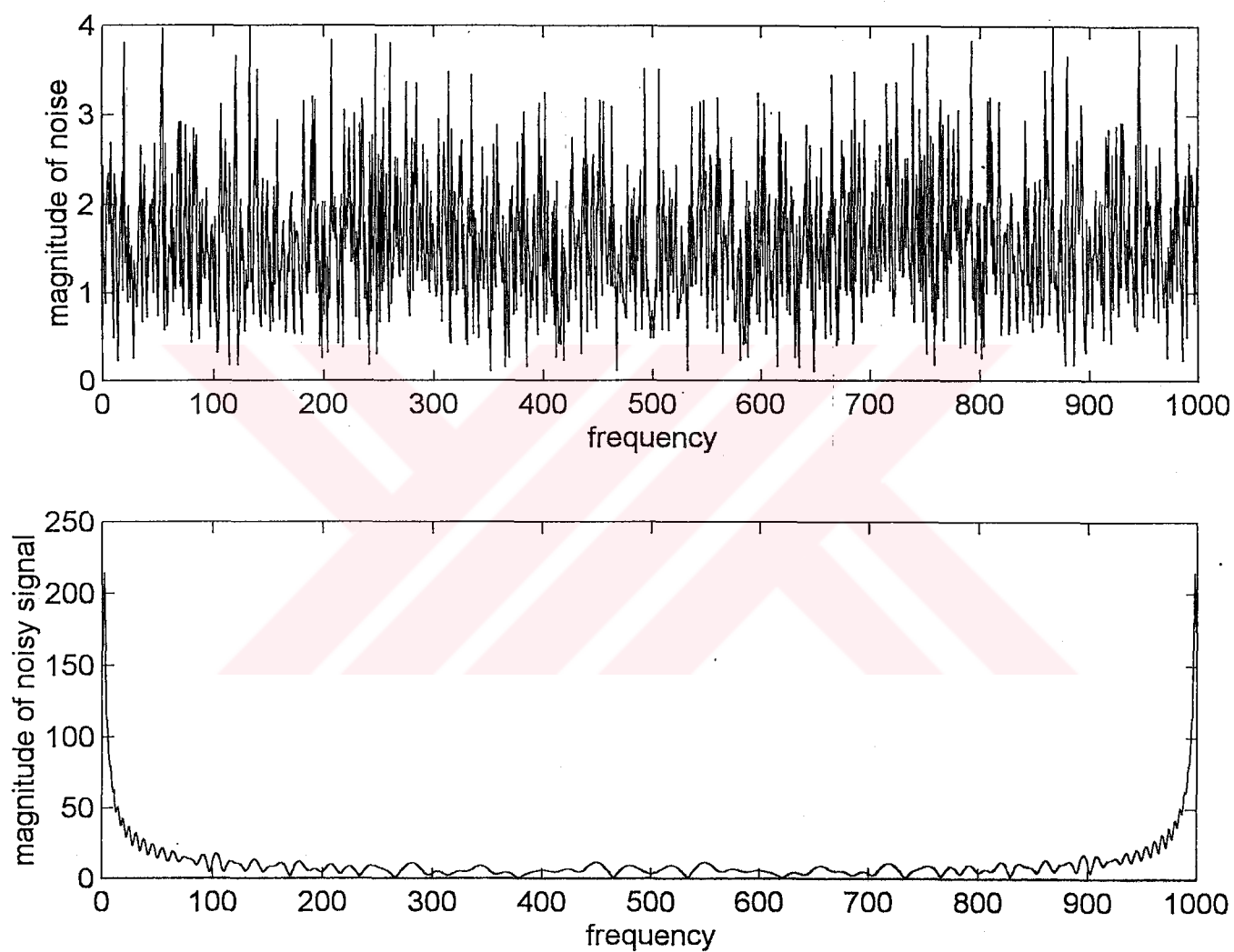
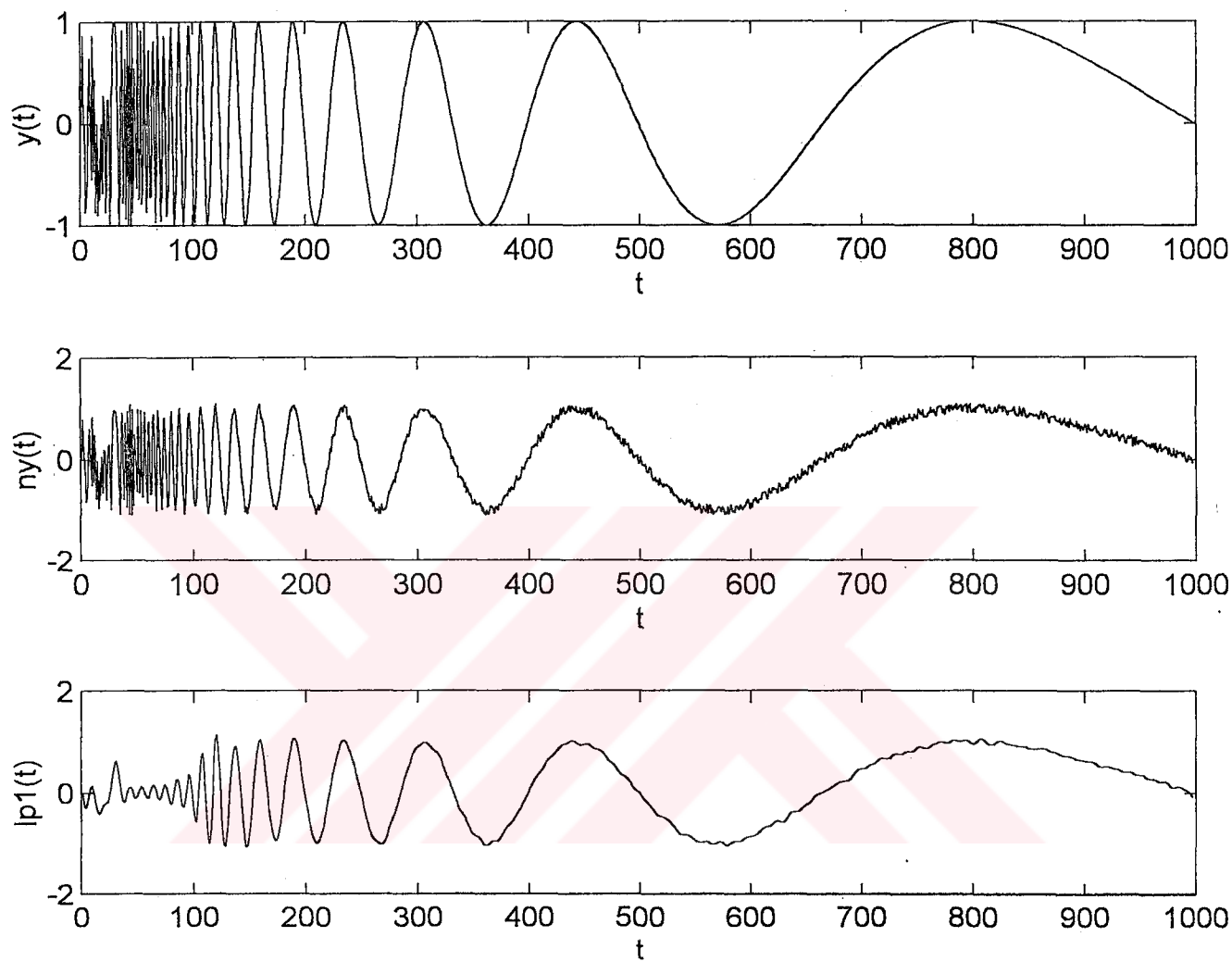


Figure 6.1 a) Frequency Response Spectrum of Noise b) Frequency Response Spectrum of Noisy Sequence



**Figure 6.2 a) Plot of Noise-Free Sequence b) Noisy Sequence
c) Lowpass Filtered Sequence ($n=300, W_n=0.2$)**

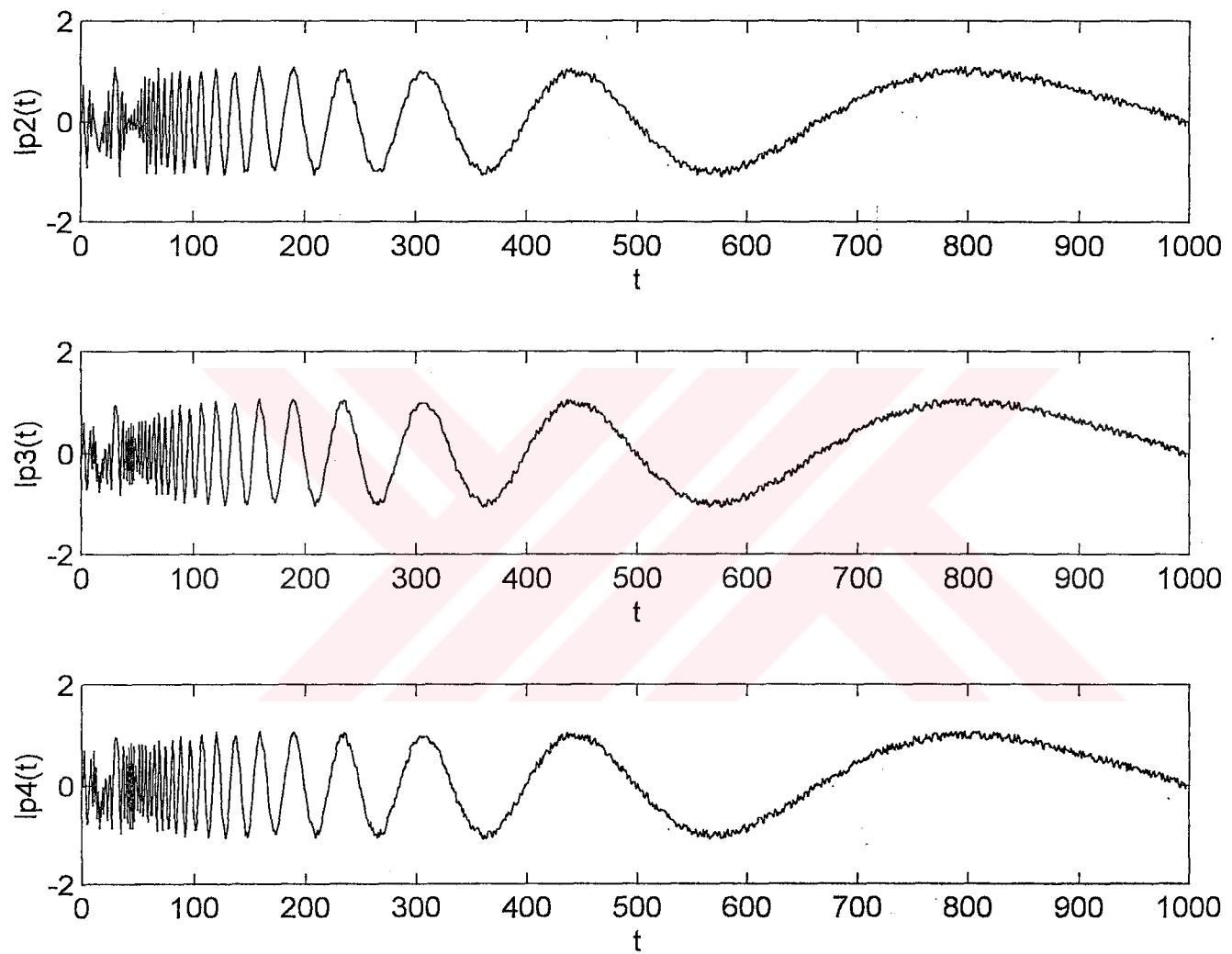


Figure 6.3 a) Lowpass Filtered Sequence ($n=300, Wn=0.7$) b) Lowpass Filtered Sequence ($n=2, Wn=0.2$) c) Lowpass Filtered Sequence ($n=2, Wn=0.7$)

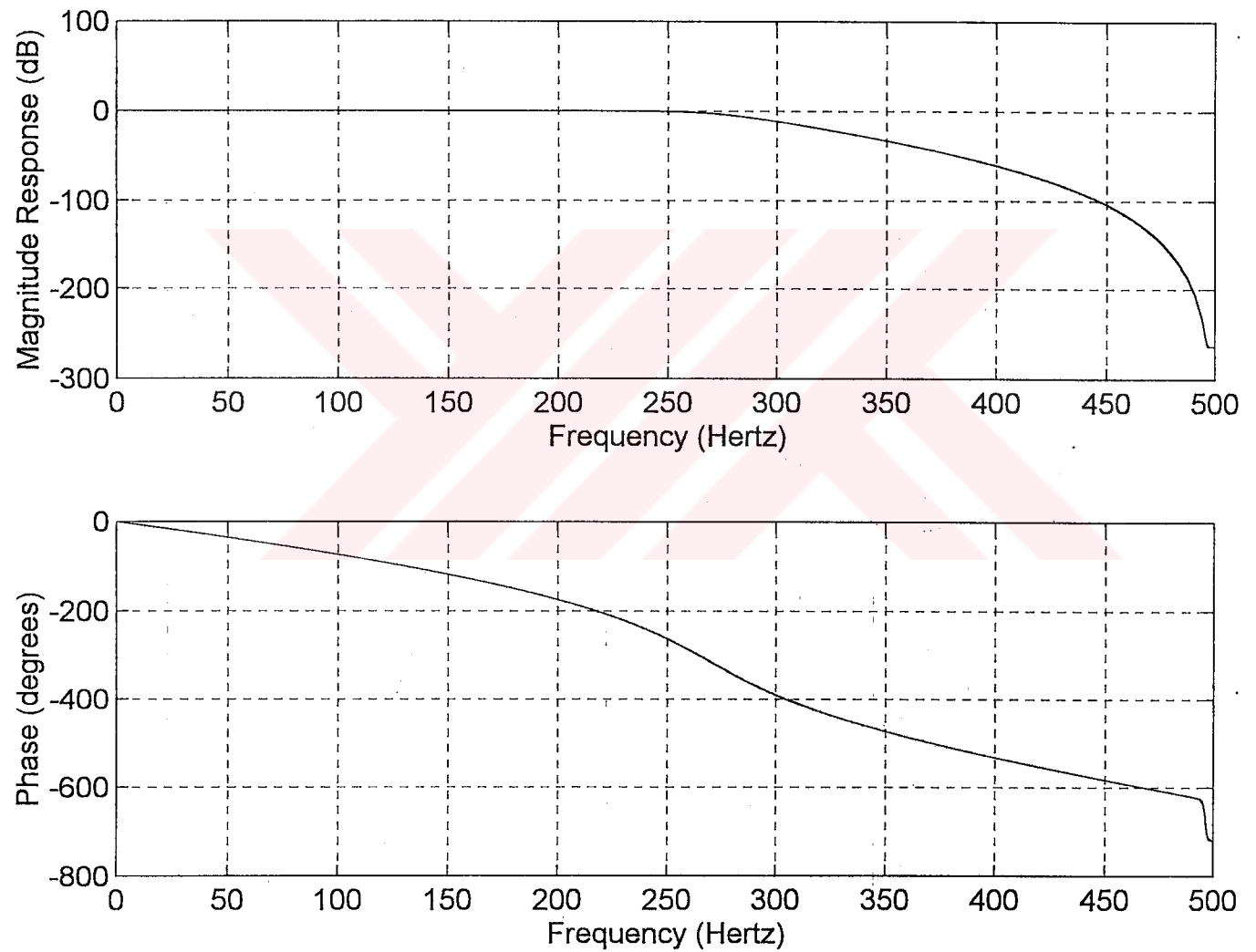


Figure 6.4 a) Butterworth Filter Magnitude Response b) Phase Response

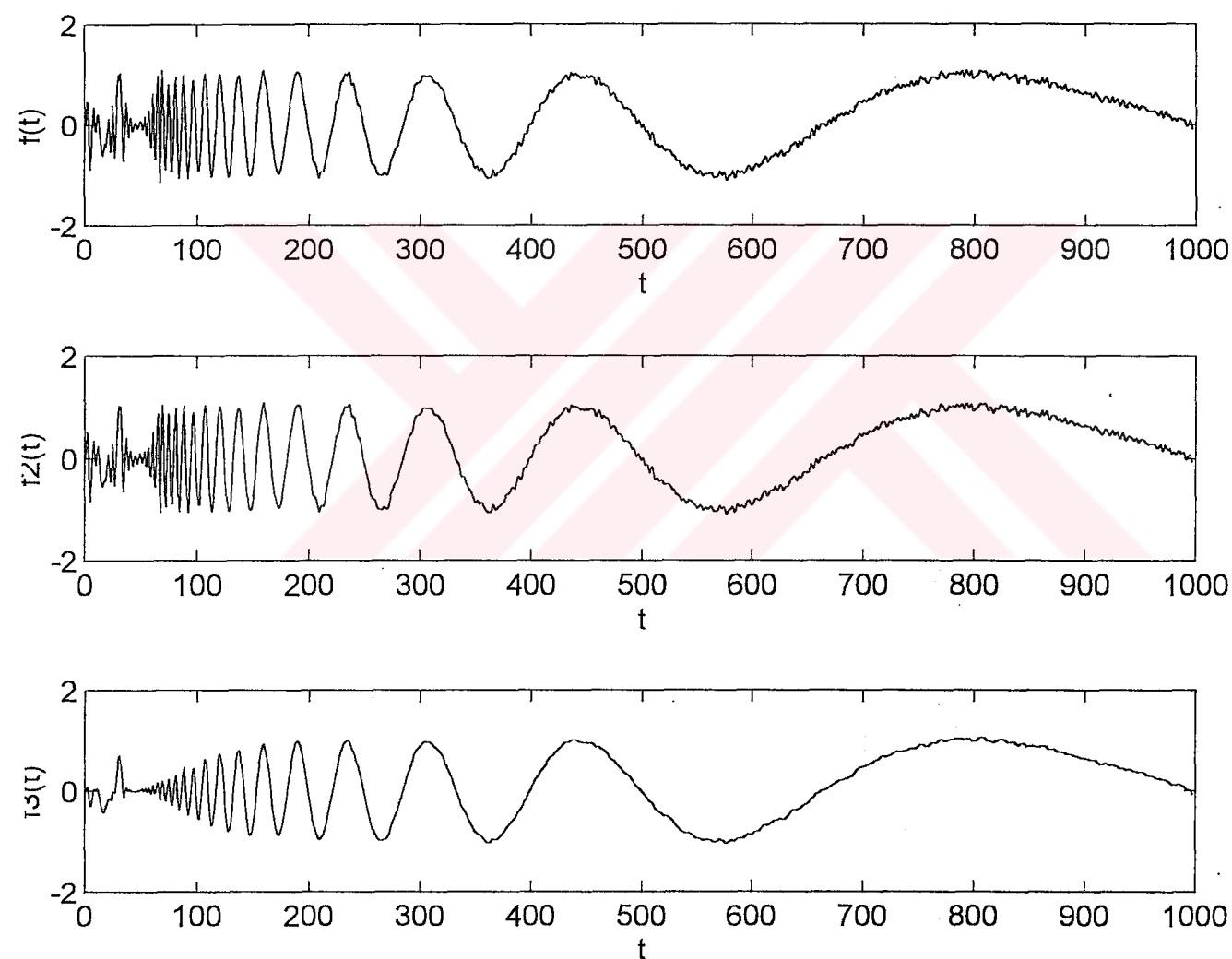


Figure 6.5 Results of Butterworth Filters a) $R_p=0.1\text{dB}$ b) $R_p=1\text{dB}$ c) $R_p=10\text{dB}$

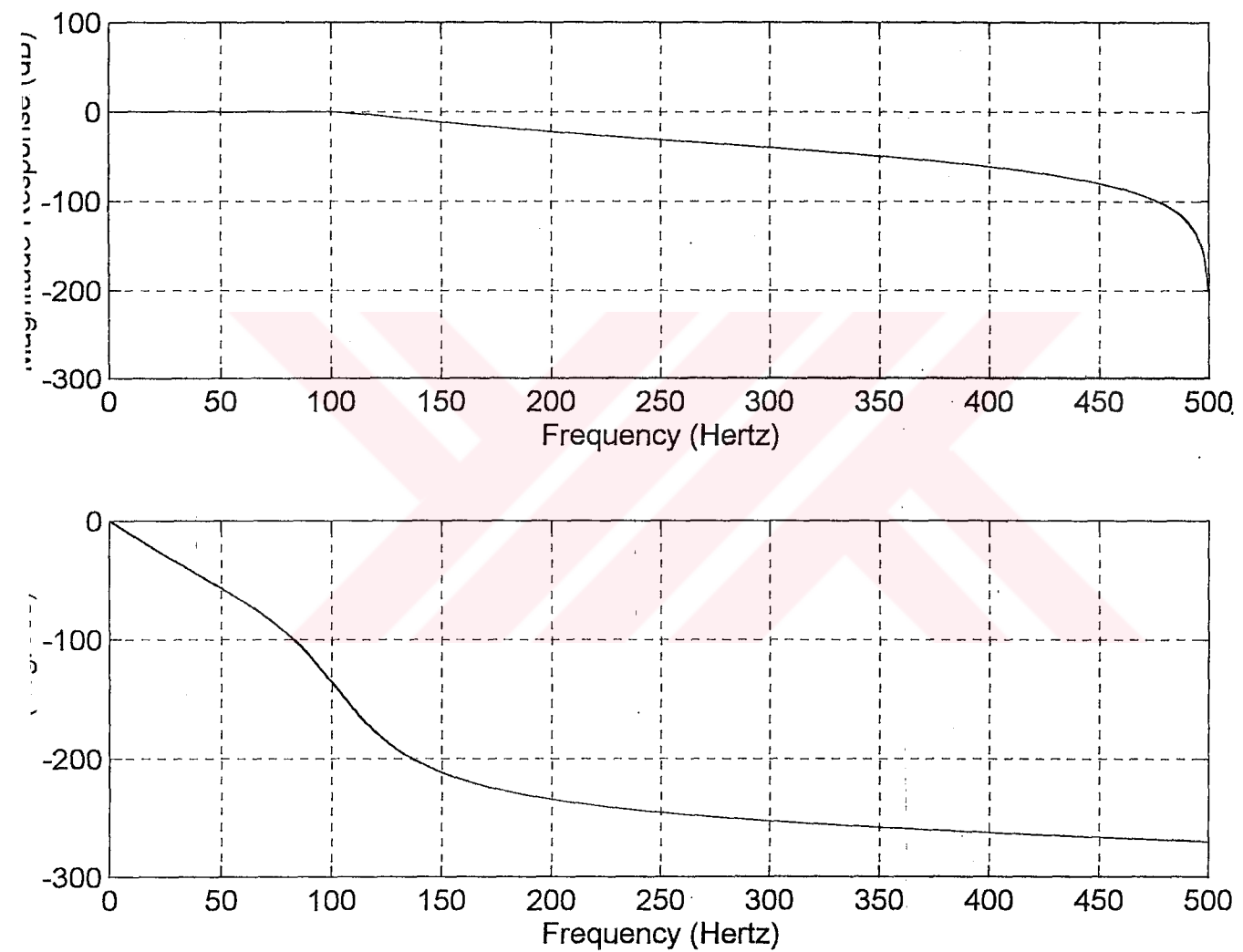


Figure 6.6 a) Chebyshev Filter Magnitude Response b) Phase Response

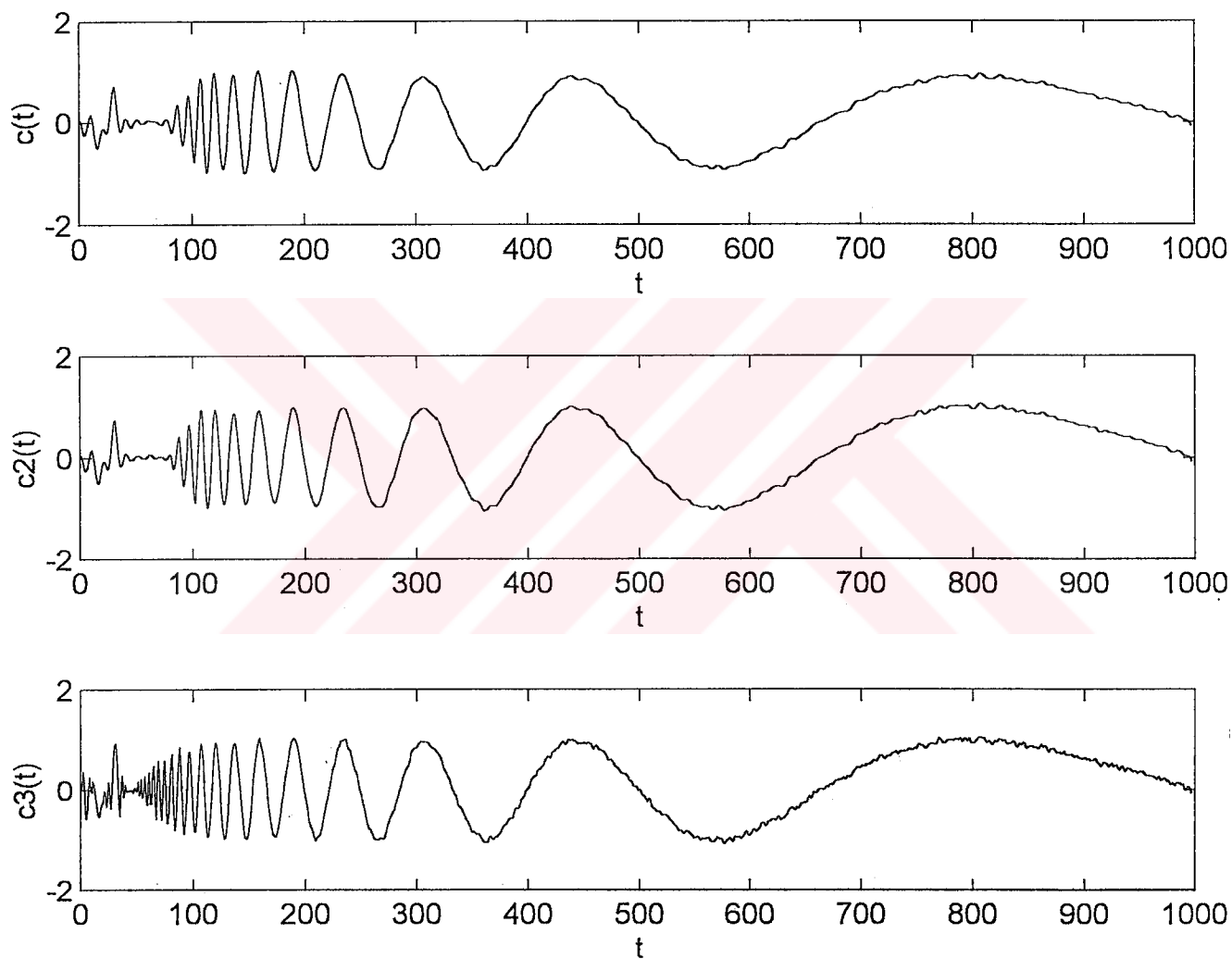


Figure 6.7 Results of Chebyshev Filters a) $A=0.1\text{dB}$, $R_p=0.5\text{dB}$ b) $A=1\text{dB}$, $R_p=0.5\text{dB}$ c) $A=10\text{dB}$, $R_p=0.5\text{dB}$

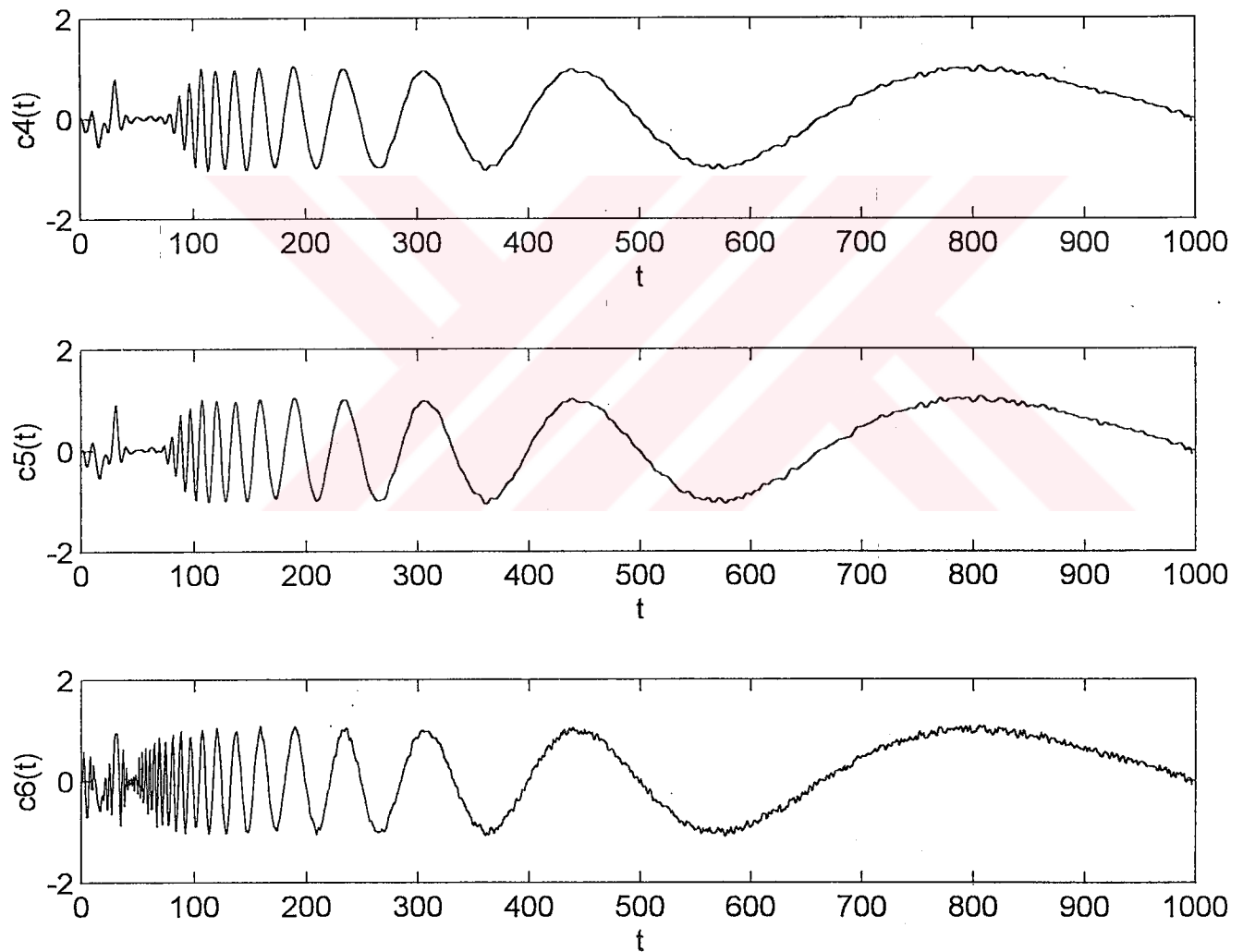


Figure 6.8 Results of Chebyshev Filters a) $A = 0.1\text{dB}$, $R_p = 0.1\text{dB}$ b) $A = 1\text{dB}$, $R_p = 0.1\text{dB}$ c) $A = 10\text{dB}$, $R_p = 0.1\text{dB}$

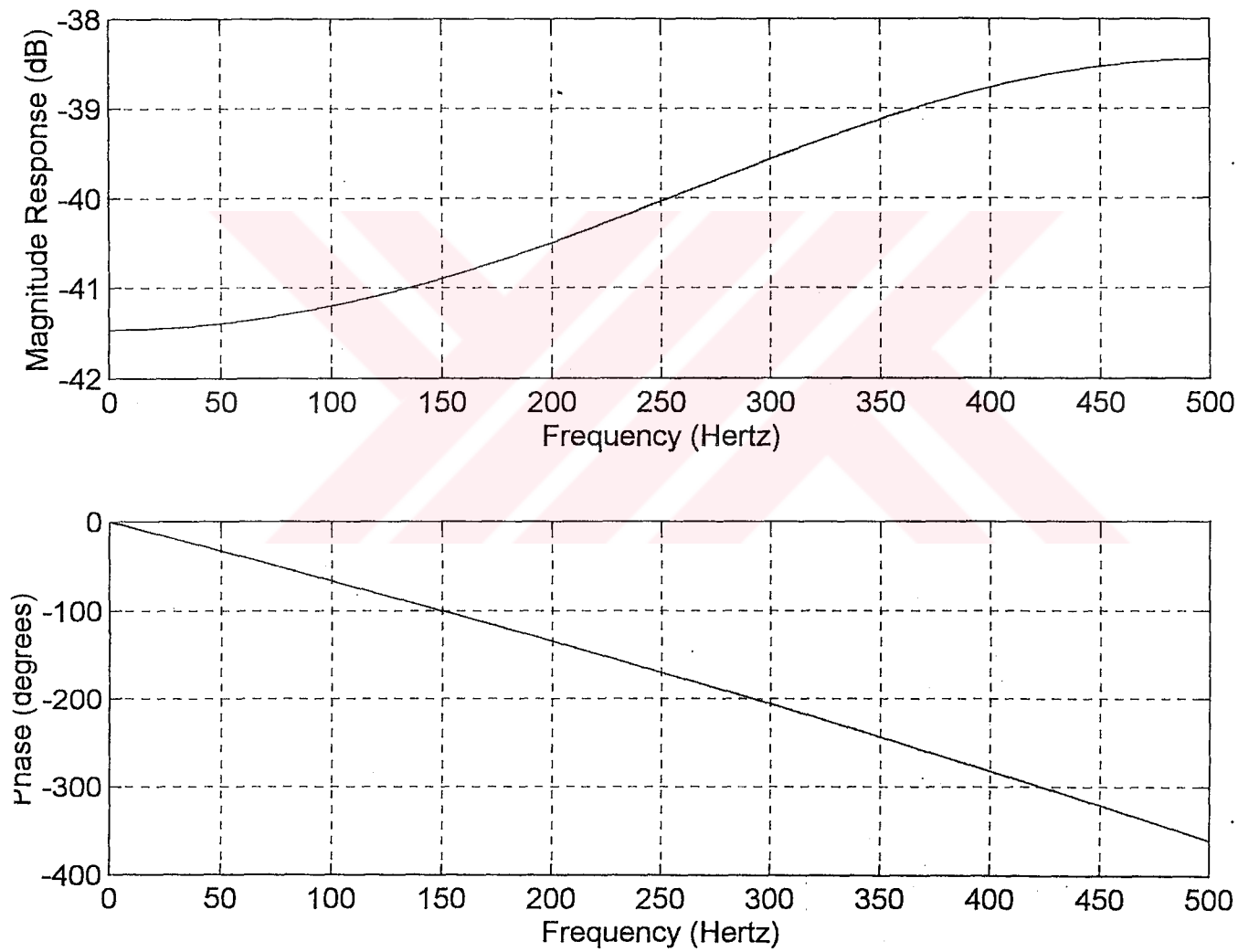


Figure 6.9 a) Bessel Filter Magnitude Response b) Phase Response

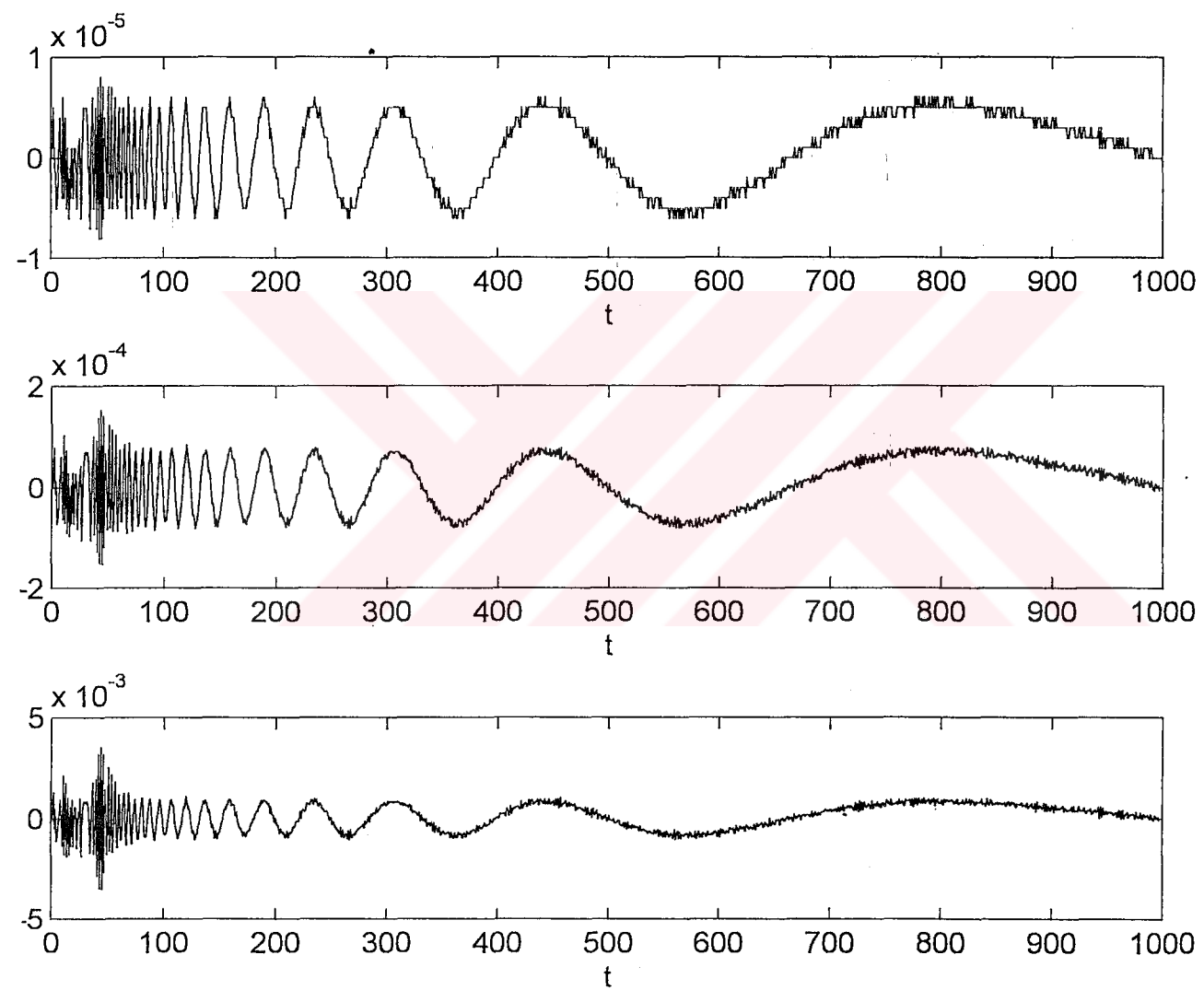


Figure 6.10 Results of Bessel Filters a) $n=2, W_n=0.05\text{dB}$ b) $n=2, W_n=0.1\text{dB}$ c) $n=2, W_n=0.2\text{dB}$

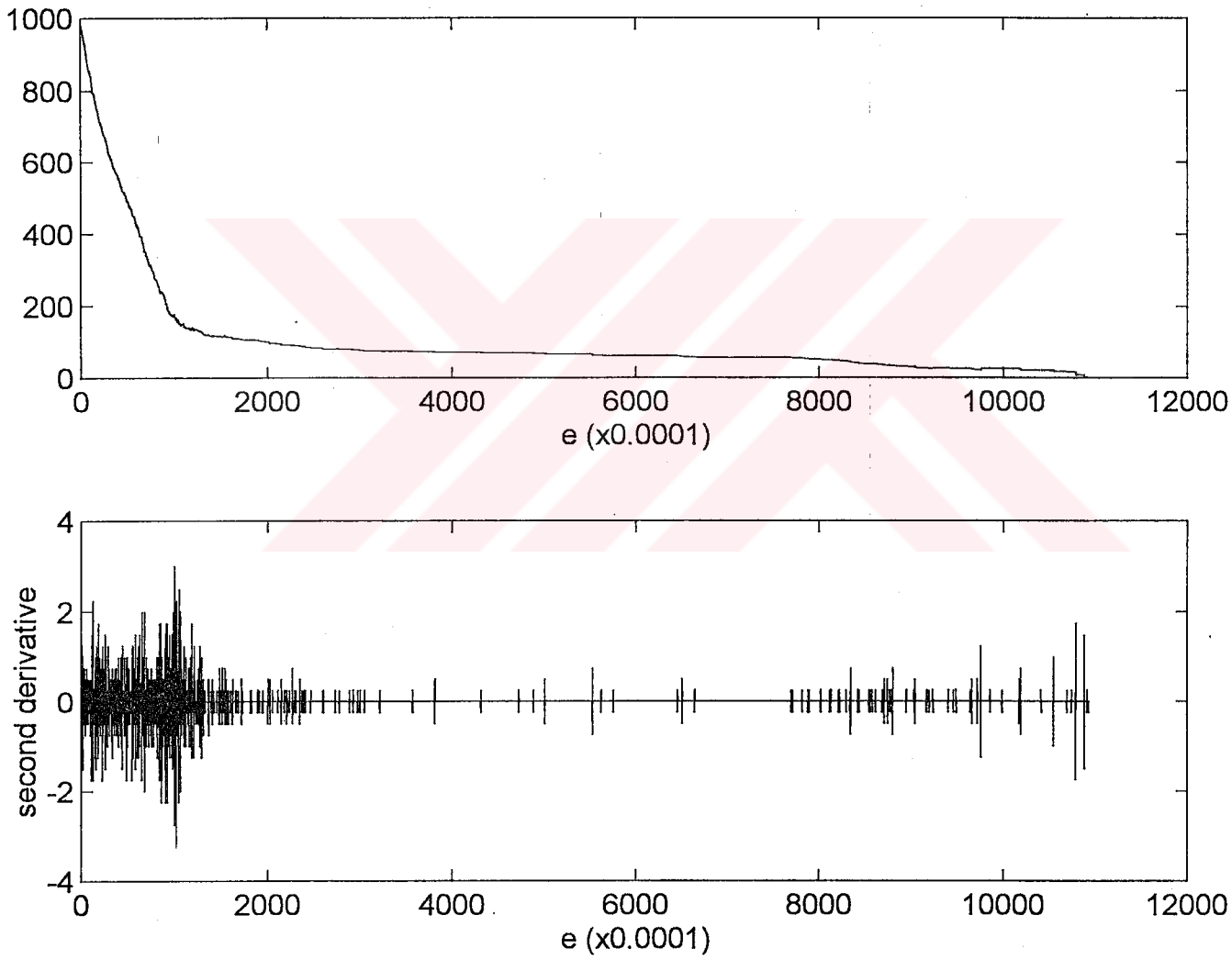


Figure 6.11 a) Plot of Compressed Size Versus Allowed Loss b) Second Derivative

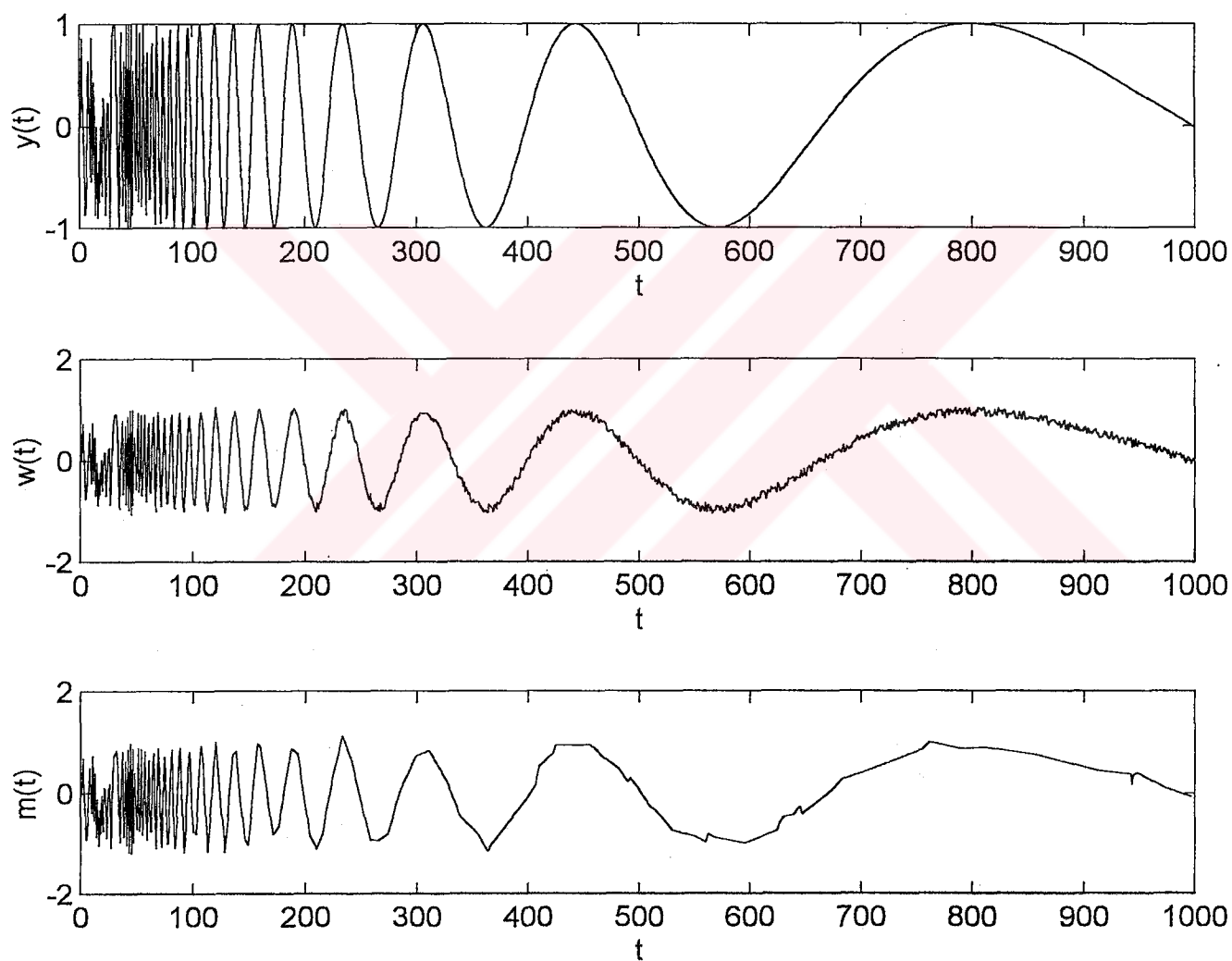


Figure 6.12 a) Original Noise-Free Signal b) Wiener Filtered Sequence c) Occam Filtered Sequence

6.6 Results of the Compression Experiment

The number of samples in the original sequence, which is shown in Figure 6.13, of the first compression experiment is 1000.

Vector Quantization:

- 8 codebook values, each represented by 32 bits
- 8 level can be represented by 3 bits.
- $8 \times 32 + 3 \times 1000 = 3256$ bits used after compression.
- $1000 \times 16 = 16000$ bits needed for the original signal
- 79.65% of compression is achieved. Result is given in Figure 6.14a

DCT:

- The energy ratio (%97.99882) that is kept is arranged so that the compression ratios of Occam filter and DCT compression is the same. Output is given in Figure 6.14b.

Run Length Coding:

- The resulting compressed signal had 673 pairs as (A, B) where A is the sample value and B is the number of times it repeats itself. (Largest B was 20)
- $673 \times 16 + 1000 \times 4 = 14768$ bits used after compression
- $1000 \times 16 = 16000$ bits needed for the original signal
- 7.7% of compression is achieved. The result is given in Figure 6.14c.

Occam Filter: (PSD is in Figure 6.15)

- At $\epsilon=0.15$, only 720 points are left after compression. 28.00 % of compression is achieved (Figure 6.16a).
- At $\epsilon=0.34$, only 517 points are left after compression. 48.40% of compression is achieved. (Figure 6.16b).
- At $\epsilon=0.8$, only 250 points are left after compression. 75.00% of compression is achieved. (Figure 6.16c).

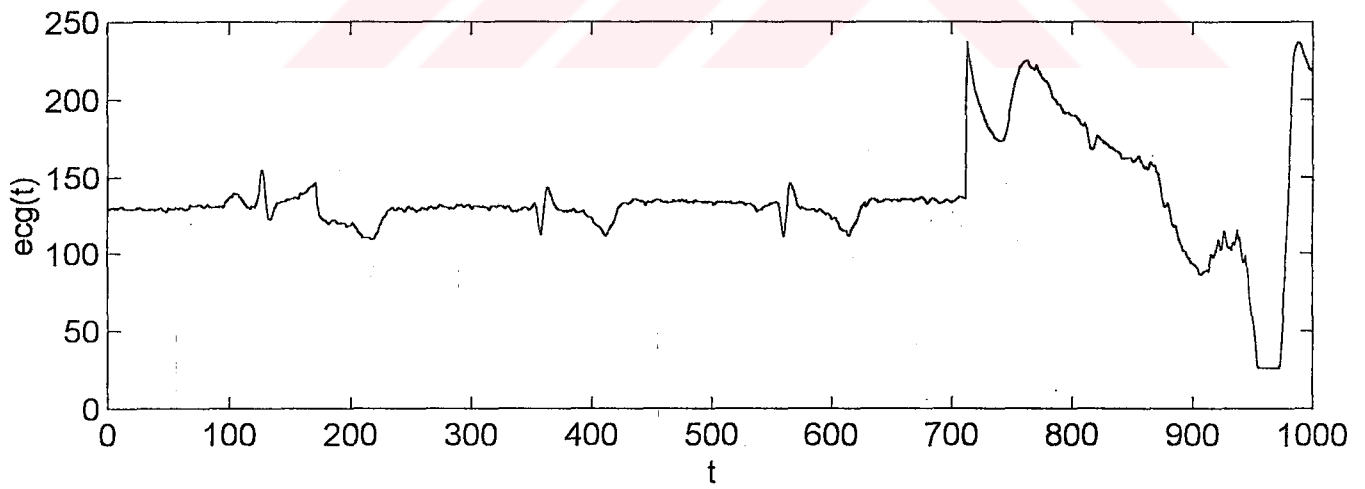


Figure 6.13 ECG Signal

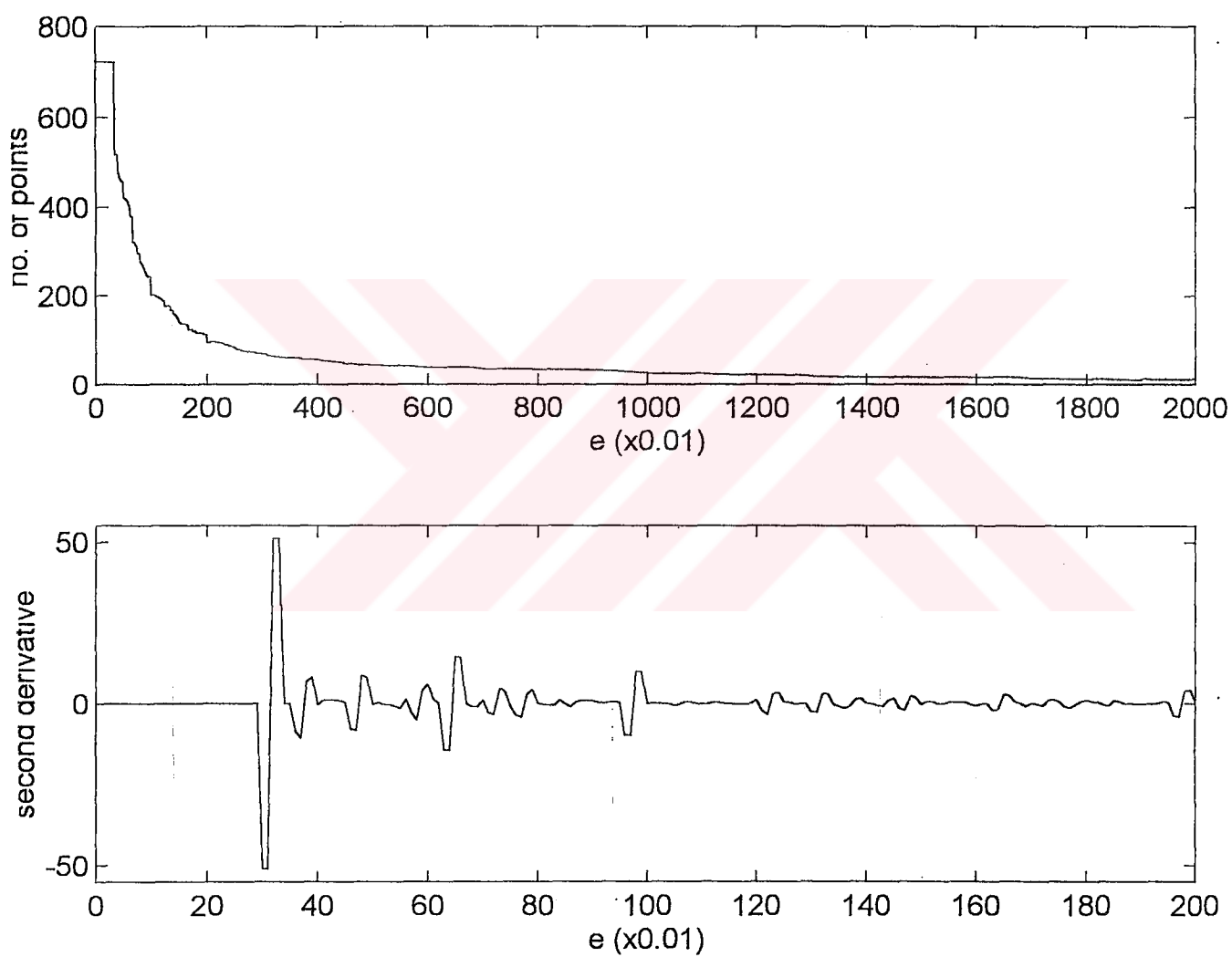


Figure 6.15 a) Plot of the Compressed Size versus Allowed Loss ϵ
b) Second Derivative

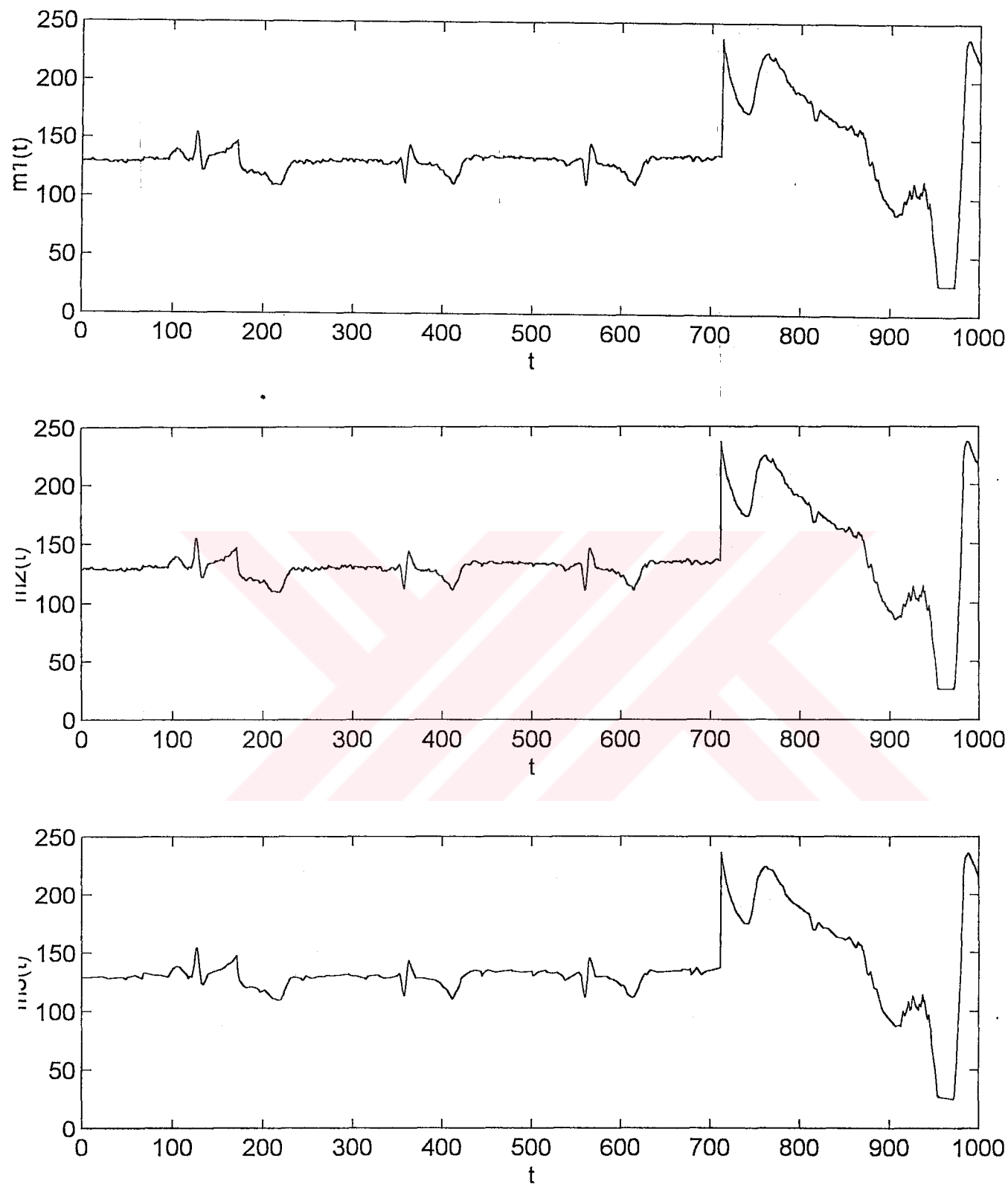


Figure 6.16 Results of Occam Compressed Sequences a) $\epsilon=0.15$ b) $\epsilon=0.34$ c) $\epsilon=0.8$

The number of samples in the original sequence, which is shown in Figure 6.17, of the second compression experiment is 121.

Vector Quantization:

- 8 codebook values, each represented by 16 bits
- 8 level can be represented by 3 bits.
- $8 \times 16 + 3 \times 121 = 491$ bits used after compression.
- $121 \times 16 = 1936$ bits needed for the original signal
- 74.64% of compression is achieved. Result is given in Figure 6.18a.

DCT:

- The energy ratio (%97.4) that is kept is arranged so that the compression ratios of Occam filter and DCT compression is the same. Result is given in Figure 6.18b.

Run Length Coding:

- The resulting compressed signal had 91 pairs as (A, B) where A is the sample value and B is the number of times it repeats itself. (Largest B was 20)
- $91 \times 16 + 91 \times 5 = 1911$ bits used after compression
- $121 \times 16 = 1936$ bits needed for the original signal
- 1.29% of compression is achieved. Result is given in Figure 6.18c.

Occam Filter: (PSD is in Figure 6.19)

- At $\epsilon=500$, only 113 points are left after compression. 6.61% of compression is achieved (Figure 6.20a).
- At $\epsilon=1112$, only 65 points are left after compression. 46.28% of compression is achieved (Figure 6.20b).
- At $\epsilon=1500$, only 46 points are left after compression. 31.98% of compression is achieved (Figure 6.20c).

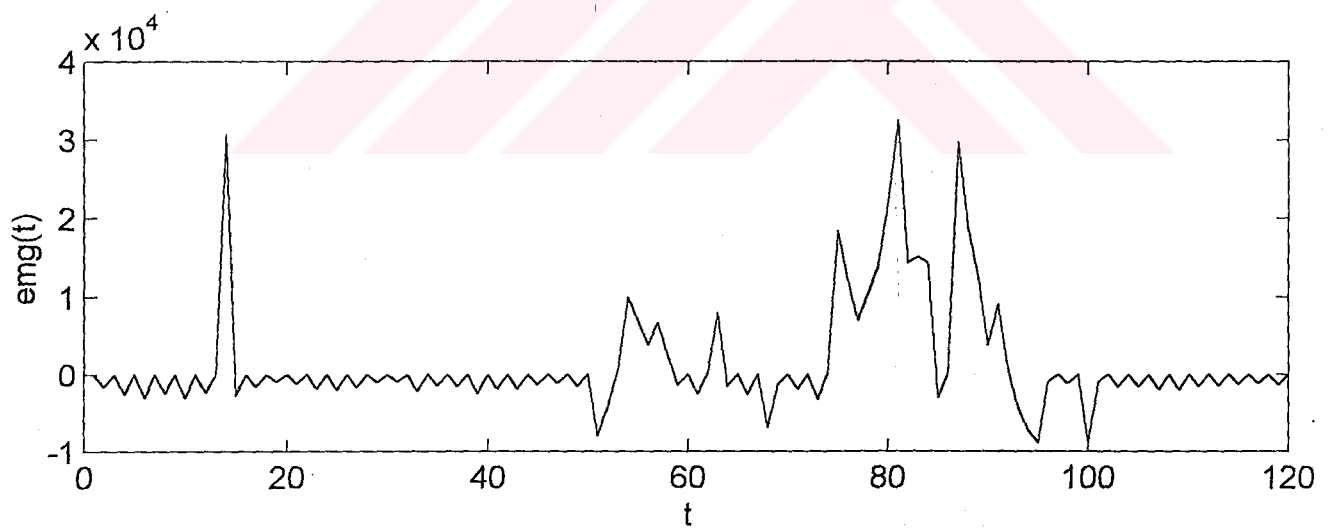


Figure 6.17 EMG Signal

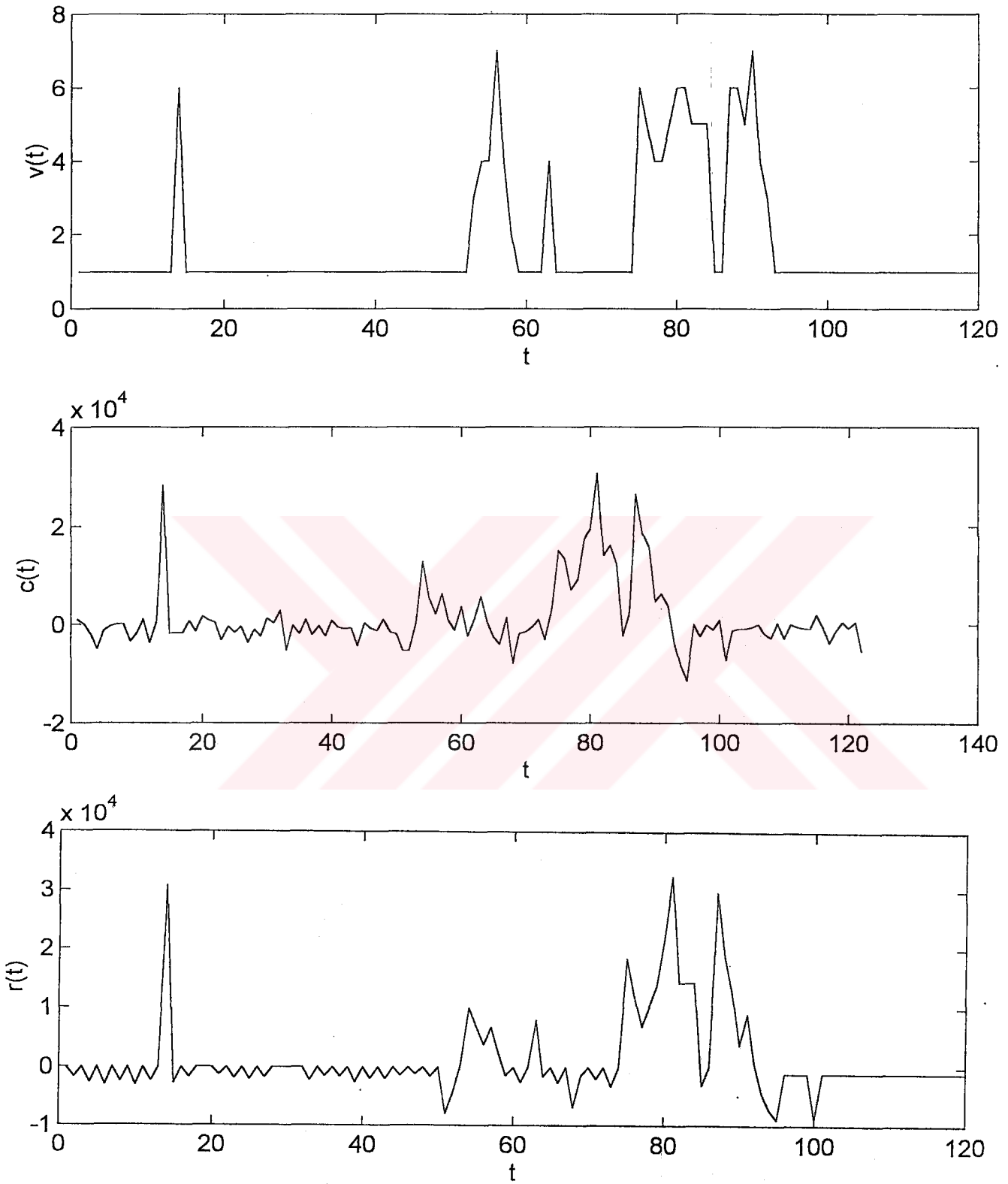


Figure 6.18 a) Decompressed Sequence After it is Compressed by Vector Quantization b) Decompressed Sequence After it is Compressed by DCT c) Decompressed Sequence After it is Compressed by Run Length Coding

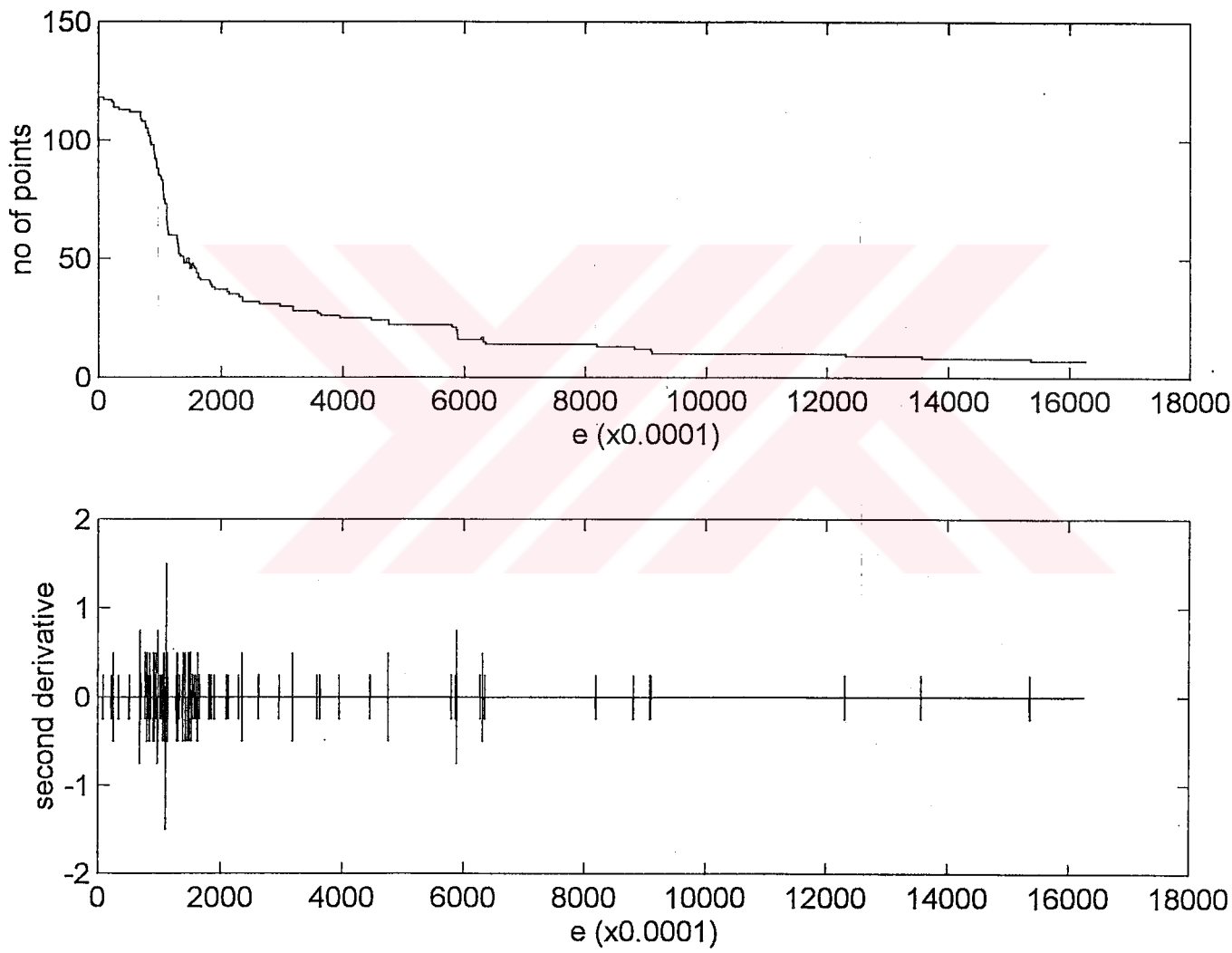


Figure 6.19 a) Plot of the Compressed Size versus Allowed Loss ϵ b) Second Derivative

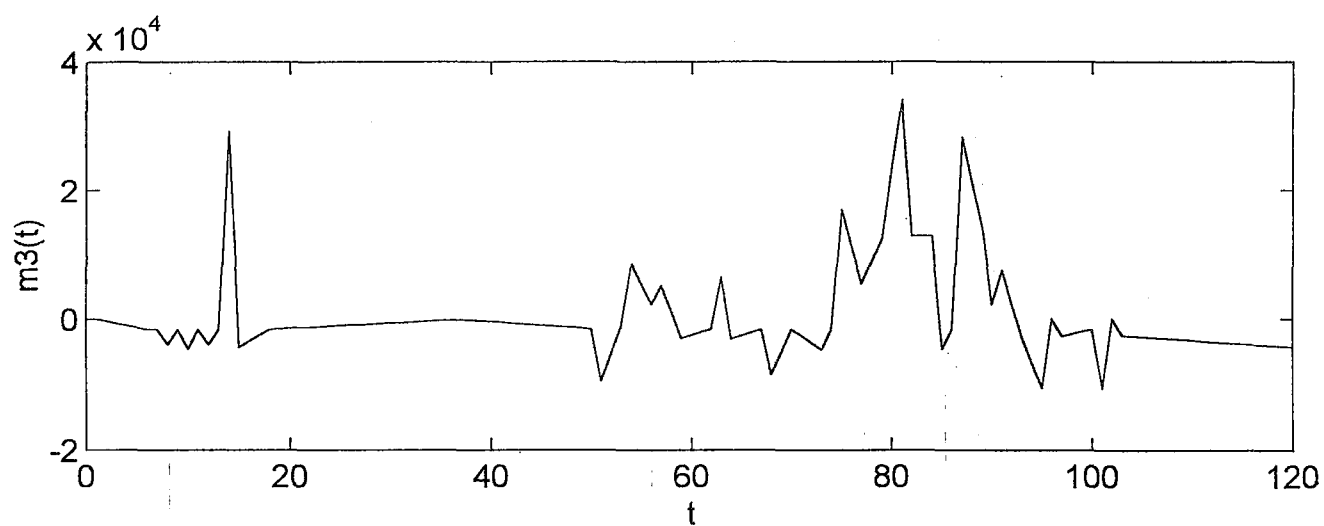
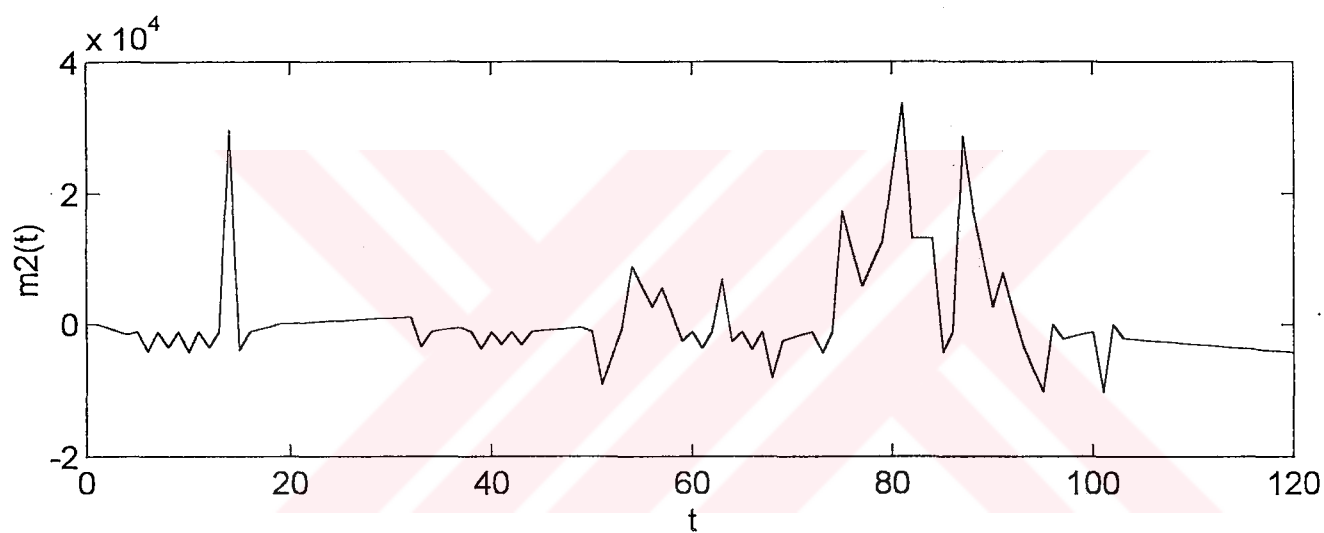
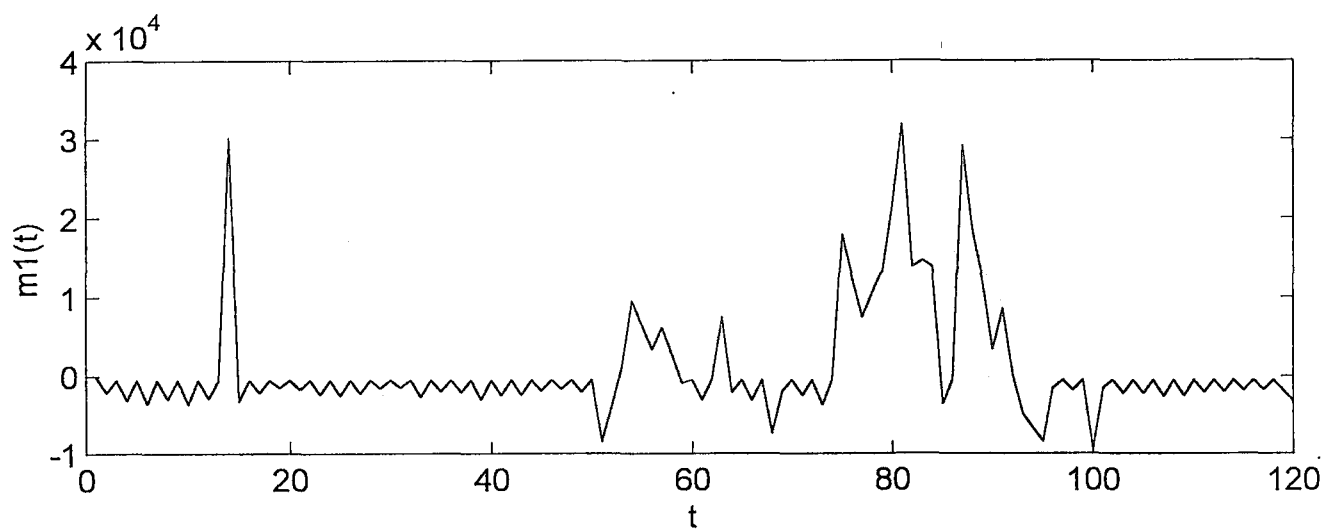


Figure 6.20 Results of Occam Compressed Sequences a) $\epsilon = 500$ b) $\epsilon = 1112$ c) $\epsilon = 1500$

6.7 Comparison of Compression Types

Although there is a big growth in computer technology, mainly because of their size, processing of medical data is still a problem. Lossless compression schemes do not compress the large amount of data enough, on the other hand lossy compression schemes cause to loose to much information for a right diagnosis.

From the reconstructed signals in Figure 5.18, 5.20, 5.22, 5.24, it can be seen that Vector Quantization Compression scheme resulted as the worst case. Although it achieved the best compression, some important information is totally lost. One may argue that it is because the codebook is only an 8-level one, however, because of the too much difference between data samples, further leveling is impossible.

The compression ratios of Occam Filter and the DCT Compression for this signal sequence is the same, but it is clear that the reconstructed signal of the DCT Compression is more complex than the original.

The second best reconstructed signal is achieved by Occam Filter and it is in the acceptable boundaries. According to the doctors, performance of the reconstructed signal of Run-Length coding is the best. However, the compression ratio of Run Length coding is the worst. Moreover, the problem with the Run Length Coding is to find the right threshold value to get the best reconstructed signal. In this example, the value of the noise strength found by the algorithm described in section 2.5 is used.

To conclude, balance between the compression ratio and the shape of the reconstructed signal is best achieved by Occam Filters.

6.8 Image Filtering with Occam Filters

In theory of SVD [7], any $M \times N$ image can be decomposed uniquely as

$$I = U \Sigma V^T = \sum_{i=1}^n \alpha_i \bar{u}_i \bar{v}_i^T \quad (6.2)$$

where U and V are orthogonal matrices with column vectors \bar{u}_i and \bar{v}_i^t and $\Sigma = \text{diag}(\alpha_1, \alpha_2, \dots, \alpha_n)$ is a diagonal matrix.

The α_i of Σ are called the singular values of I and when arranged in descending order, give r , the rank of the matrices I because

$$\alpha_1 \geq \alpha_2 \geq \dots \alpha_r > \alpha_{r+1} = \dots = \alpha_n = 0 \quad (6.3)$$

However, it is observed that when an image matrix I is corrupted with noise, the last $n-r$ α'_i of the noisy image $N+I=NI$ are small but not necessarily zero [5].

From this observation, we can conclude that, given a threshold ε and α'_i as the singular values of NI , if a window for α'_i

$$\beta_i = \begin{cases} \alpha'_i & \text{otherwise} \\ 0 & \alpha'_i \leq \varepsilon \end{cases} \quad (6.4)$$

is applied to NI , filtered image $B = U_B \Sigma'_B V_B^t$ where $\Sigma'_B = \text{diag}(\beta_1, \beta_2, \dots, \beta_n)$ can be found.

As a result, general algorithm for filtering an image can be summarised as:

1. divide $M \times N$ noisy image matrix NI into $n \times n$ subblocks B_s
2. perform SVD on each block
3. set to zero the singular values of each subblock that are less than a threshold ε
4. unite the subblocks to form the filtered image.

The problem with this algorithm is to find the right threshold value ε to control the amount of loss. The algorithm for finding ε in Occam filters applies to this situation as well.

6.9 Experiment and Results

Accepting that to reduce the noise lowpass filtering must be done, Median, Lowpass (window method) and Lowpass (frequency domain) are applied to a n image corrupted with random noise.

From the results of Figure 6.21, we can conclude that two dimensional Occam Filtered image does not perform well.



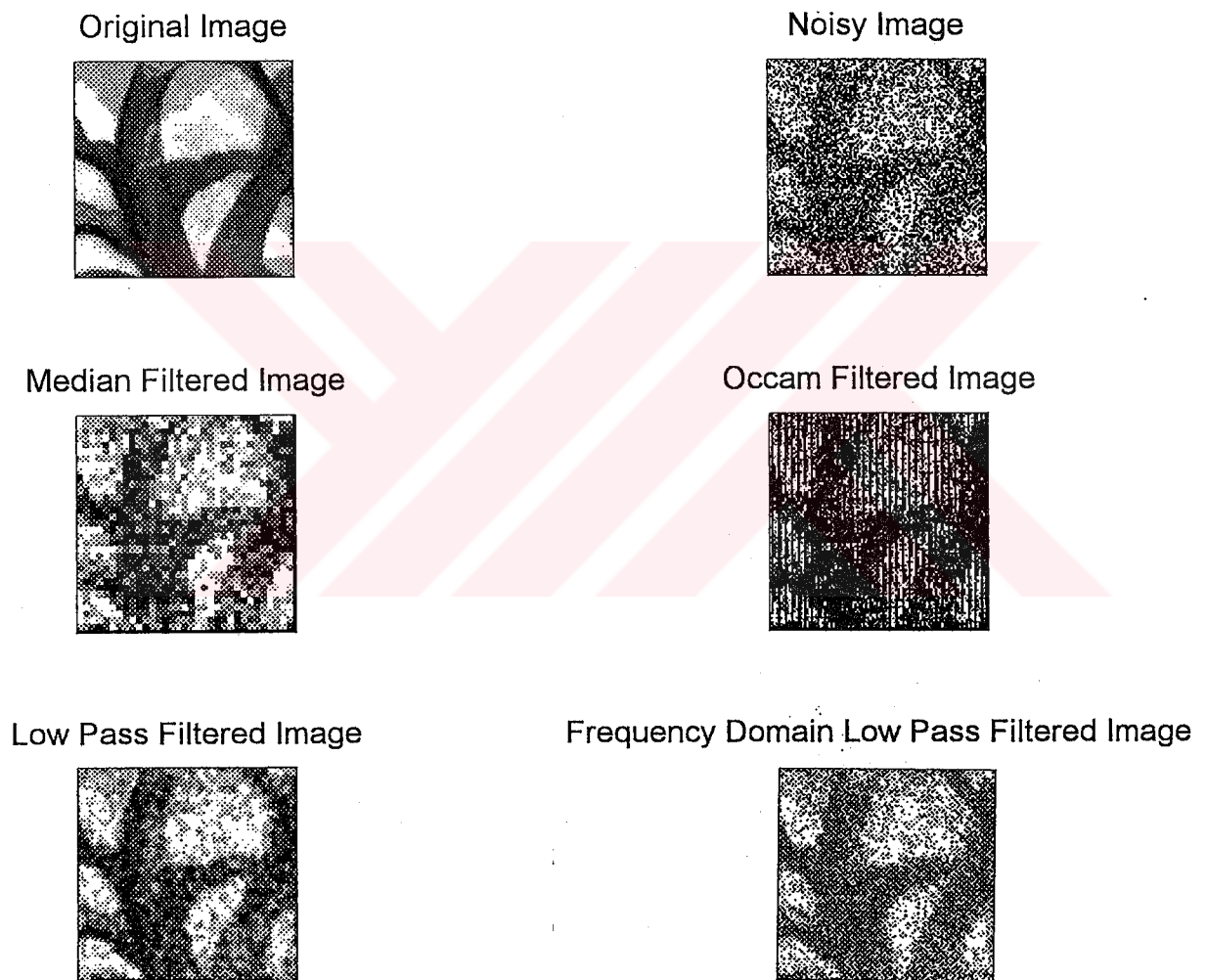


Figure 6.21 a) Original Image b) Noisy Image c) Lowpassed Filtered Image d) Occam Filtered Image e) Median Filtered Image f) f domain Filtered Image

CHAPTER SEVEN

CONCLUSION

The developments in computer technology helps us to ease signal processing. However, there are still some problems to be solved.

Especially, handling medical data is a problem. During the acquisition of the data, the current given to the patient causes pain. Moreover, because of the human body system, the retrieving must be repeated several times to get clear data.

After this, comes the second problem, the storage of large amount of data. To decrease the number of data samples that hold the same amount of information is possible by compression methods, but lossless compression does not compress enough and lossy compression techniques loose to much of the information for a right diagnosis.

Occam Filters, as proved by the experiments done, compress the medical data much better than the other methods. It also solves the problem of repeating the acquisition, which causes pain, several times by its noise removing property. This second property is also used in the random noise removal from the deterministic signals that are wide-band. As shown in text, lowpass or highpass filters remove important part of the signal while removing the noise because signal is a wideband one.

As a result, Occam Filters outperforms other classical methods in compressing and filtering of medical data and deterministic signals from random noise.

REFERENCES

- Lynn P. (1989). An Introduction to Analysis and Processing of Signals (3rd ed.). New York: Hemisphere Publishing Corporation
- Banks S. (1990). Signal Processing, Image Processing and Pattern Recognition. London, Prentice Hall.
- Israel B. & Greville E. (1974). Generalized Inverse Theory and Applications. New York: John Wiley & Sons, Inc.
- Ahmed N. & Natarajan T. (1983) Discrete Time Signals and Systems. Virginia: Prentice Hall.
- Daryanani G. (1976). Principles of Active Network Synthesis and Design. Singapore: John Wiley & Sons, Inc.
- Heulsman L. (1993). Active and Passive Analog Filter Design. New Jersey: Prentice Hall.
- Kayran H. (1990). Sayısal İşaret İşleme. İstanbul: İTÜ
- Chen C. (1989) System and Signal Analysis. New York: Harcourt Brace Jovanovich College Publishers.
- Natarajan B. (1995). Filtering Random Noise from Deterministic Signals via data Compression. IEEE Transactions on Signal Processing, 43, 2595-2605.
- Konstantinos K. & Natarajan B. (1994). An Architecture for Lossy Compression of Waveforms Using Piecewise-Linear Approximation. IEEE Transactions on Signal Processing, 42, 2449-2459.
- Iri M. & Imai H. (1986). An Optimal Algorithm for Approximating a Piecewise Linear Function. Journal of Information Processing, 9, 159-161.
- Hewlett-Packard Laboratories. (1991). On Piecewise Linear Approximations to Curves. California: Author.
- Hewlett-Packard Laboratories. (1996). Noise Estimation on Filtering Using Block Based Singular Value Decomposition. California: Author.
- Hewlett-Packard Laboratories. (1996). Noise Estimation on Filtering Using Block Based

Singular Value Decomposition. California: Author

Natarajan B., Konstantinides K. & Herley C. (1996). Occam Filters For Stochastic Sources with Application to Digital Images. Proc. IEEE ICIP-96.

Natarajan B., Konstantinides K. & Bahaskaran V. (1996). Effects of Noise Pre Filtering for Codebook Generation in Vector Quantization. Proc. IEEE ICIP-96.

Natarajan B., Konstantinides K. (1994). Comparing Occam and Wiener Filters on Broad-band Signals. Proc. 28th Asilomar Conference on Signals and Systems and Computers.,1,1254-1260

Rabbani M., & Paul W. Jones (1991). Digital Image Compression Techniques. Bellingham, Washington: Spie Optical Engineering Press.

Pearson D. (1991). Image Processing. London: McGraw-Hill Book Company

Wayne N., Digital Image Processing. London. Prentice-Hall (UK) Ltd.

Gonzales R. (1993). Digital Image Processing. USA: Addison-Wesley Publishing Company

Shalkof R. Digital Image Processing and Computer Vision. New York: John Wiley & Sons, Inc.

Jain, K. (1989). Fundamentals of Digital Image Processing. New Jersey: Prentice Hall.