



NEAR EAST UNIVERSITY

**GRADUATE SCHOOL OF APPLIED
AND SOCIAL SCIENCES**

**ADAPTIVE NOISE CANCELLATION
USING LMS ALGORITHM**

Jamal Fathi Abu Hasnah


Master Thesis

**Electrical And Electronic Engineering
Department**

Nicosia - 2000

**Jamal Fathi Abu Hasnah: Adaptive Noise Cancellation Using
LMS Algorithm**

**Approval of Director of the Graduate School of Applied
and Social Sciences**

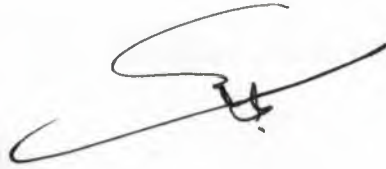


Prof. Dr. Ergin Iğrek

**We certify that this thesis is satisfactory for the award of the
degree of Master of Science in Electrical Engineering**

Examining Committee in Charge:

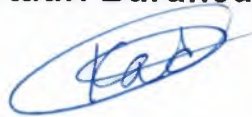
**Asst. Prof. Dr. Adnan Khashman, Chairman of Committee,
Chairman of the Computer
Engineering Department, NEU**



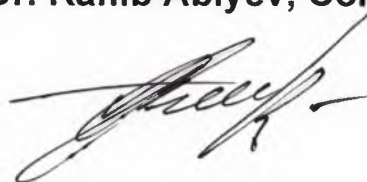
**Prof. Dr. Fakhraddin Mamedov, Head of the Electrical and
Electronic Engineering Department,
Supervisor, NEU**



**Assist. Prof. Dr. Kadri Bürüncük, Electrical and Electronic
Engineering Department,
Member, NEU**



**Assist. Prof. Dr. Rahib Abiyev, Computer Engineering
Department,
Member, NEU**



ACKNOWLEDGEMENTS

First, I thank my mother and father for their love, support, and encouragement. I could not have achieved this without their help and guidance. I am grateful to them for their love and support, and for their help and guidance.

Second, I thank my friends, especially Paul, Dr. Adnan Khatib, and Dr. Paul. I am grateful to them for their love and support, and for their help and guidance.

Third, I thank my teachers, especially Dr. Adnan Khatib, for their help and guidance.

Fourth, I thank my friends, especially Paul, Dr. Adnan Khatib, and Dr. Paul. I am grateful to them for their love and support, and for their help and guidance.

I thank my friends, especially Paul, Dr. Adnan Khatib, and Dr. Paul. I am grateful to them for their love and support, and for their help and guidance.

Dedicated to my Mother and my Father

I thank my friends, especially Paul, Dr. Adnan Khatib, and Dr. Paul. I am grateful to them for their love and support, and for their help and guidance.

I thank my friends, especially Paul, Dr. Adnan Khatib, and Dr. Paul. I am grateful to them for their love and support, and for their help and guidance.

I thank my friends, especially Paul, Dr. Adnan Khatib, and Dr. Paul. I am grateful to them for their love and support, and for their help and guidance.

I thank my friends, especially Paul, Dr. Adnan Khatib, and Dr. Paul. I am grateful to them for their love and support, and for their help and guidance.

I thank my friends, especially Paul, Dr. Adnan Khatib, and Dr. Paul. I am grateful to them for their love and support, and for their help and guidance.

I thank my friends, especially Paul, Dr. Adnan Khatib, and Dr. Paul. I am grateful to them for their love and support, and for their help and guidance.

ACKNOWLEDGEMENTS

I wish to thank my adviser, Prof. Dr. Fakhraddin Mamedov, for intellectual support, encouragement, and enthusiasm which made this thesis possible, and for his patience in correcting both my stylistic and scientific errors.

Second, I would like to thank Assist. Prof. Dr. Adnan Khashman, for his help when needed, his patience in correcting both my stylistic and scientific errors.

Third, I would like to thank Prof. Dr. Khalil Ismailov, for his help and his knowledge.

Fourth, also I would like to thank Assist. Prof. Dr. Kadri Bürüncük, Assist. Prof. Dr. Rahib Abiyev, for their help and making this thesis possible.

I thank my brothers and sister, my sincere thanks are beyond words for thier continuous encouragement.

I thank my love Asma, for her constant encouragement and support during my study.

I thank all my friends espicially HALIL ARSLAN, ADNAN IDO, Mr. GUNER.....etc. Who helped me alot.

To all of them, all my love and respect.

ABSTRACT

The contamination of a desired signal by unwanted and unpredictable noise is a problem often encountered in digital communication and control systems.

When the signal and noise occupied fixed and separate frequency band conventional linear filters (analog or digital) can be based. When spectral overlap between signal and noise occurs or band occupied by the noise is unknown or varies with time, it is necessary to design a filter adapted to changes of the signal characteristics.

One of the important problems in long distance communication system over a telephone channel is removing residual and far end echo signals.

Aim of this thesis is the analysis and interpretation of the adaptive noise canceller based on LMS algorithm for removing hybrid and acoustic echo noises within the digital environment for improving a voice quality of the long distance telecommunication systems.

CONTENTS

Dedication	
Acknowledgements	i
Abstract	ii
Contents	iii
INTRODUCTION	1
1. OVERVIEW OF FILTERS	3
1.1 Classic Analog Filters	5
1.2 Polynomial Approximation of the Frequency	8
1.2.1 Butterworth Approximation	8
1.2.2 Chebyshev Approximation	11
1.2.3 Type 1 Chebyshev Approximation	11
1.2.4 Elliptic Approximation	13
1.2.5 Linear-Phase Approximation	15
1.3 A Comparison of the Filter Types	16
2. ADAPTIVE FILTERS	19
2.1 Overview	19
2.2 General Properties	23
2.3 Open-And Closed-Loop Adaptation	26
2.4 Applications	28
2.5 Applications of the Adaptive Filter	31
2.5.1 Loud Speaking Telephones	34
2.5.2 Radar Signal Processing	36

2.5.3 Separation of Speech Signals From Background Noise	36
3. ADAPTIVE ECHO CANCELLATION	37
3.1 Overview, Definitions	37
3.2 History of Echo Cancellation	38
3.3 Acoustic Echo	39
3.4 Hybrid Echo	40
3.5 The Combined Problem on Digital Cellular Networks	42
3.6 Process of Echo Cancellation	44
3.7 Controlling Acoustic Echo	45
3.8 Controlling Complex Echo in a Wireless Digital Network	46
3.9 Echo Cancellation System For Radio Telephony	49
4. THE FUNDAMENTAL PROBLEMS AND SOLUTIONS TO ECHO CANCELLATION	51
4.1 Overview	51
4.2 Introduction	52
4.3 Long-Distance International Calls Between Ordinary Fixed Telephone	53
4.3.1 Echo Suppressor	55
4.3.2 Adaptive Echo Canceller	56
5. THE LMS ALGORITHM	62
5.1 Overview, Derivation	62

5.2	Convergence of the Weight Vector	64
5.3	Noise In The Weight-Vector Solution	67
5.4	The Basic LMS Adaptive Algorithm	68
5.5	Implementation of The Basic LMS Algorithm	70
5.6	Practical Limitations Of The Basic LMS Algorithm	72
5.6.1	Effect of Non-Stationarity	72
5.6.2	Effects Of Signals Component On The Interference Input Channel	74
5.6.3	Computer Wordlength Requirements	75
5.6.4	Coefficient Drift	76
5.7	Fast LMS Algorithm	77
5.8	Recursive Least Squares Algorithm	77
5.9	Limitations of The Recursive Least Squares Algorithm	80
6.	FINITE-PRECISION EFFECTS	82
6.1	Overview	82
6.2	Total Output Mean-Squared Error	85
6.3	Leaky LMS Algorithm	87
6.4	Stalling Phenomenon	88
6.5	Parameter Drift	90
6.6	Recursive Least-Squares Algorithm	94
6.7	Error Propagation Model	96
7.	PRACTICAL CONSIDERATIONS AND DESIGN FILTERS	103
7.1	Design Procedure	103
7.2	MATLAB Implementations	103

7.3 Lowpass Filter Design	107
7.4 Comparison of Three Filters	113
7.5 Adaptive LMS	120
8. CONCLUSION	127
9. REFERENCES	129

INTRODUCTION

The design of a Wiener filter requires *a priori* information about the statistics of the data to be processed. When this information is not known completely, however, it may not be possible to design the Wiener filter or else the design may no longer be optimum. A straightforward approach that used in such situations is the "estimate and plug" procedure. For *real-time* operation, this procedure has the disadvantage of requiring excessively elaborate and costly hardware.

A more efficient method is to use an *adaptive filter*. By such a device we mean one that is *self-designing* in that the adaptive filter relies for its operation on a *recursive algorithm*, which makes it possible for the filter to perform satisfactorily in an environment where complete knowledge of the input signal characteristics is not available.

Introducing the least-mean-square LMS algorithm. Is important because of its simplicity, ease of computation, and because it does not require off-line gradient estimations or repetitions of data.

In this thesis, design adaptive filters based on Least Mean Square Algorithm are discussed.

Fundamental problems related with hardware and software implementation of the echo cancellator are considered.

This has led to intensive research into the area of echo cancellation, with the aim of providing solutions that reduce background noise and remove hybrid and acoustic echo before any transcoder processing.

The first chapter represents classification of filters, approximation of the frequency response characteristics using Butterworth, Chebyshev, and Elliptic Filters. Chapter provides comparison of analog and digital filters and different frequency response Characteristic.

Chapter two is devoted to the adaptive filter that provides real time operation in unknown input signal characteristic. General properties, Open loop, closed-loop adaptation are examined.

End sections of the chapter consider application of adaptive filter in identification, noise cancellation.

Chapter three explores adaptive echo cancellation, acoustic echo, hybrid echo, and combined problems of the digital cellular telephone.

Chapter four examines the fundamental problems and solutions of echo cancellation and echo suppression in the telephone network.

Chapter five presents analysis of LMS algorithm and their software and hardware implementation. Basic limitations related with the effect of nonstationarity of input signals, computer wordlength requirement, drift of coefficients are considered.

Chapter six treats Finite Precision Effect, stalling phenomenon, and parameter drifts of the precision of filtering using LMS.

Chapter seven is devoted to the practical implementation and design of classical filter and adaptive filter using MATLAB. As input signal, is used signal combining of 15 and 30 harmonic component and white noise.

Structure of adaptive filter and timing diagram of signal are plotted by MATLAB argument.

In the conclusion are given important results obtained by the author of the thesis on analysis interpretation and practical realization of the adaptive filter.

1. OVERVIEW OF FILTERS

Filtering is a process by which the frequency spectrum of signal can be modified, reshaped, or manipulated according to some desired specification. It may entail amplifying or attenuating a range of frequency components, rejecting or isolating one specific or attenuating a range of frequency component, etc. The uses of filtering are manifold, e.g., to eliminate signal contamination such as noise to remove signal distortion brought about by an imperfect transmission channel or by inaccuracies in measurement, to separate two or more distinct signals which were purposely mixed in order to maximize channel utilization, to demodulate signals, to convert discrete-time signals into continuous-time signals.

The digital filter is a digital system that can be used to filter discrete-time signals. It can be implemented by mean of software (computer programs) or by means of dedicated hardware, and in either case it can be used to filter real-time signals or non-real-time (recorded) signals.

Software digital filters made their appearance along with the first digital computer in the late forties, although the name digital filter did not emerge until the midsixties. Early in the history of the digital computer many of the classical numerical analysis formulas of NEWTON, STARLING, etc., and others were used to carry out interpolation, differentiation, and integration of function (signals) represented by mean of sequences of numbers (discrete-time signals). Since interpolation, differentiation, or integration of a signal represents a manipulation of the frequency spectrum of the signal, the subroutines or programs constructed to carry out these operations were essentially digital filters. In subsequent years, many complex and highly sophisticated algorithms and programs were

developed to perform a variety of filtering tasks in numerous application, e.g., data smoothing and prediction, pattern recognition, electrocardiogram processing, and spectrum analysis. In fact, as time goes on, interest in the software digital filter is becoming progressively more intense while its applications are increasing at an exponential rate.

Band-limited continuous-time signals can be transformed into discrete-time signals by means of sampling. Conversely, the discrete-time signals so generated can be used to regenerate the original continuous-time signals by means of interpolation, by virtue of Shannon's sampling theorem. As a consequence, hardware Digital's filters can be used to perform real-time filtering tasks, which in the not too distant past were performed almost exclusively by analog filters. The advantages to be gained are the traditional advantages associated with digital systems in general:

1. Component tolerances are uncritical.
2. Component drift and spurious environmental signals have no influence on the system performance.
3. Accuracy is high.
4. Physical size is small.
5. Reliability is high.

A very important additional advantage of digital filters is the ease with which filter parameters can be changed in order to change the filters characteristics. This feature allows one to design programmable filters which can be used to perform a multiplicity of filtering tasks. Also one can design new types of filters such as adaptive filters. The main disadvantage of hardware digital filters at present is their relatively high cost. However, with the tremendous advancements in the domain of large-scale integration, the cost of

hardware digital filters is likely to drop drastically in the not too distant future.

1.1 Classic Analog Filters

While the importance of analog filters is continuously being reduced by their digital counterparts, they remain an important study, if for no other reason than they provide a gateway to the study of digital filters. The design of a contemporary analog filter, in many cases, remains today as it was during the early days of radio. The design objective of the radio engineers was to shape the frequent-spectrum of a received or transmitted signal using modulators, demodulators, and frequency-selective filters. The frequency-selective filters were defined in terms of a mathematical ideal. The ideal models represent low-pass, high-pass, band-pass, band-stop, and all-pass filters. These are graphically interpreted in Figure 1.1. Their shape represents the steady-state magnitude-frequency response of a filter with a transfer function of $H(\Omega) = H(s) |_{s=j\Omega}$ where Ω denotes an analog frequency measured in radians per second. The mathematical specification of each ideal filter is summarized as,

$$\text{Ideal Low-pass } |H(\Omega)| = \begin{cases} 1 & \text{if } \Omega \in [-B, B] \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Ideal High-pass } |H(\Omega)| = \begin{cases} 0 & \text{if } \Omega \in [-B, B] \\ 1 & \text{otherwise} \end{cases}$$

$$\text{Ideal Band-pass } |H(\Omega)| = \begin{cases} 1 & \text{if } \Omega \in [-B_2, -B_1] \text{ or } \Omega \in [B_1, B_2] \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Ideal Band-stop } |H(\Omega)| = \begin{cases} 0 & \text{if } \Omega \in [-B_2, -B_1] \text{ or } \Omega \in [B_1, B_2] \\ 1 & \text{otherwise} \end{cases}$$

All-pass $|H(\Omega)| = 1$ for all $\Omega \in [-\infty, \infty]$

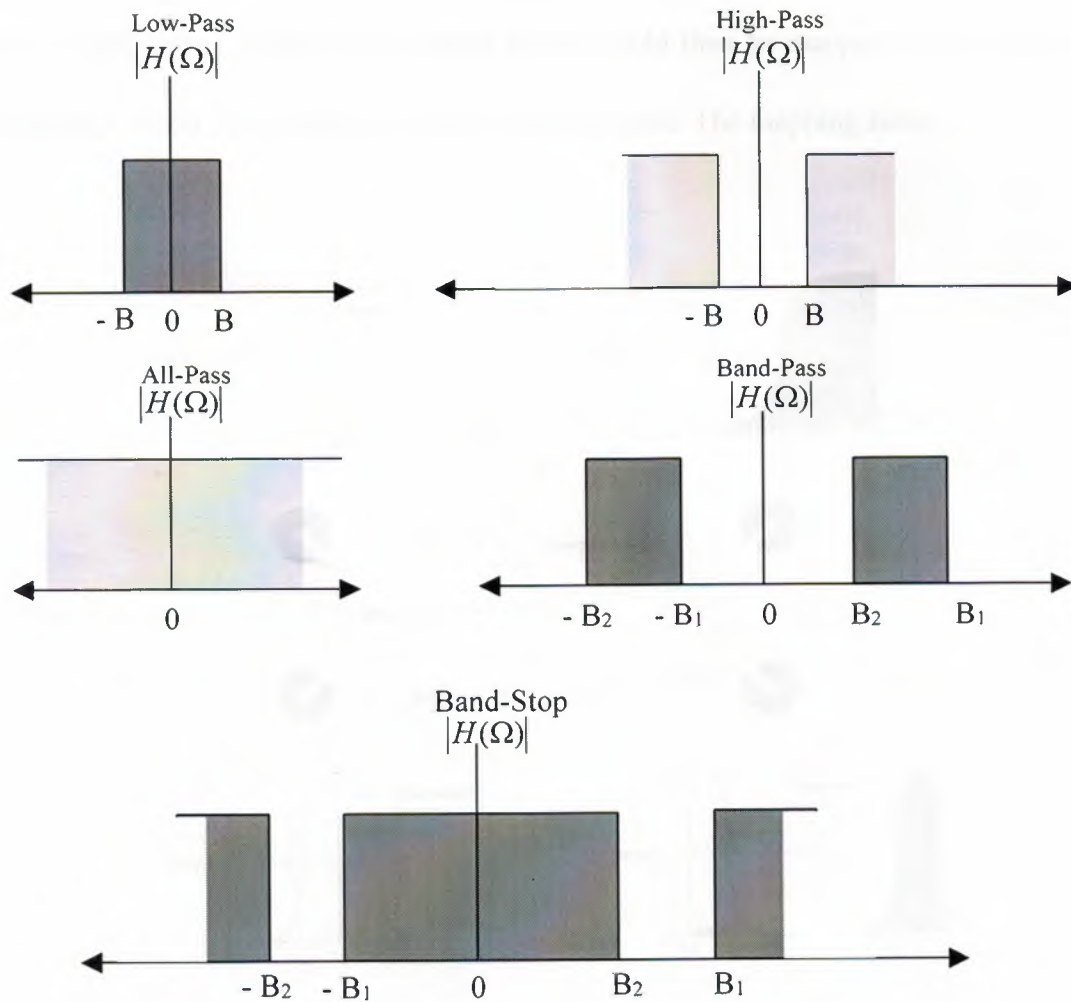


Figure 1.1 Basic Ideal Filter Types

Analog filter design is often based on the use of several well-known models called Butterworth, Chebyshev, and elliptic (Cauer) filters. To standardize the design procedure, a set of normalized analog prototype filter models was agreed upon and reduced to tables, charts, and graphs. These models, called prototypes, were all developed as low-pass

systems having a known gain (typically -1 dB or -3 dB pass-band attenuation) at a known critical cut-off frequency (typically 1 radian/second). The transfer function of an analog prototype filter, denoted $H_p(s)$, would be encapsulated in a standard table as a function of filter type and order. The prototype filter $H_p(s)$ would then be mapped into a final filter $H(s)$ having critical frequencies specified by the designer. The mapping rules,

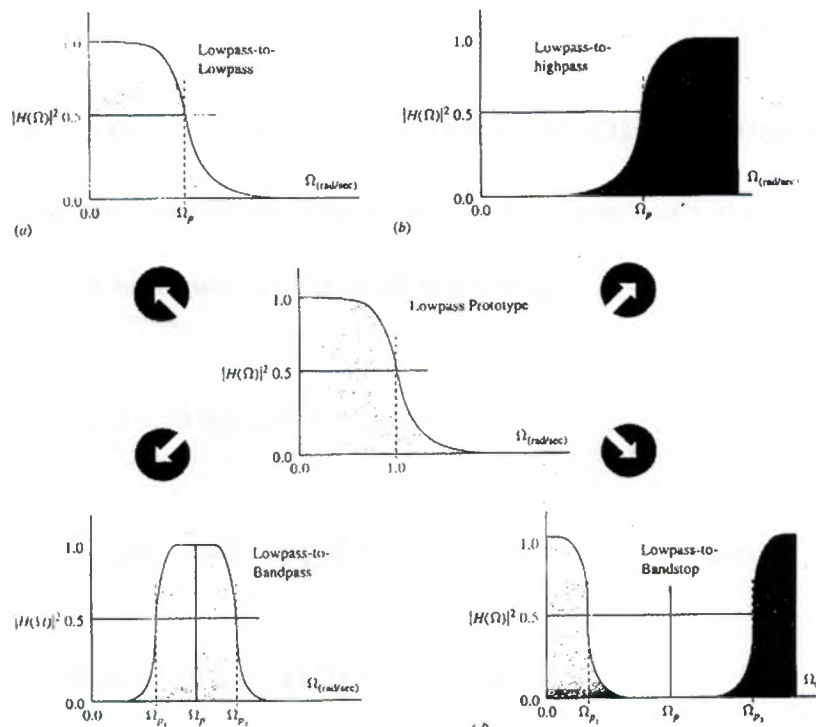


Figure 1.2 Frequency Transform

called frequency - frequency transforms, as shown in figure 1.2.

1.2 Polynomial Approximations of the Frequency

1.2.1 Butterworth Approximation

The magnitude-squared response of an analog low-pass Butterworth filter $H_a(s)$ of N th order is given by [13],

$$|H_a(j\Omega)|^2 = \frac{1}{1 + (\Omega / \Omega_c)^{2N}} \quad (1.1)$$

It can be easily shown that the first $2N-1$ derivatives of $|H_a(j\Omega)|^2$ at $\Omega = 0$ are equal to zero, and as a result, the Butterworth filter is said to have a maximally-flat magnitude at $\Omega = 0$. The gain of the Butterworth filter in dB is given by,

$$g(\Omega) = 10 \log_{10} |H_a(j\Omega)|^2 \text{ dB.}$$

At dc i.e., at $\Omega = 0$, the gain in dB is equal to zero, and at $\Omega = \Omega_c$, the gain is,

$$g(\Omega_c) = 10 \log_{10} (1/2) = -3.0103 \cong -3 \text{ dB}$$

and therefore, Ω_c is often called the 3-dB cutoff frequency. Since the derivative of the squared-magnitude response, or equivalently, of the magnitude response is always negative for positive values of Ω , the magnitude response, is monotonically decreasing with increasing Ω . For $\Omega \gg \Omega_c$, the squared-magnitude function can be approximated by,

$$|H_a(j\Omega)|^2 \approx \frac{1}{(\Omega / \Omega_c)^{2N}}$$

The gain $g(\Omega_2)$ in dB at $\Omega_2 = 2\Omega_1$ with $\Omega_1 \gg \Omega_c$ is given by,

$$g(\Omega_c) = -20 \log_{10} \left(\frac{\Omega_2}{\Omega_c} \right)^{2N} = g(\Omega_1) - 6N \text{ dB},$$

where $g(\Omega_1)$ is the gain in dB at Ω_1 . As a result, the gain roll-off per octave in the stop-band decreases by 6 dB, or equivalently, by 20 dB per decade for an increase of the filter order by one. In other words, the pass-band and the stop-band behaviors of the magnitude response improve with a corresponding decrease in the transition band as the filter order N increases. A plot of the magnitude response of the normalized Butterworth low-pass filter with $\Omega_c = 1$ for some typical values of N is shown in figure.

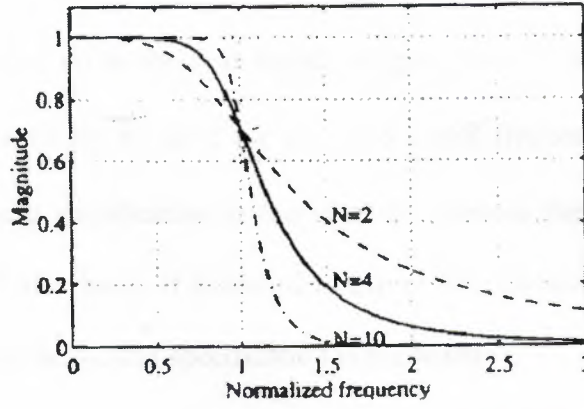


Figure 1.3 Typical Butterworth low-pass filter response.

The two parameters completely characterizing a Butterworth filter are therefore the 3-dB cutoff frequency Ω_c and the order N . These are determined from the specified pass-band edge Ω_p , the minimum pass-band magnitude $1/\sqrt{1+\varepsilon^2}$, the stop-band edge Ω_s , and the maximum stop-band ripple $1/A$. From Eq. (1.1) we get,

$$|H_a(j\Omega_p)|^2 = \frac{1}{1 + (\Omega_p / \Omega_c)^{2N}} = \frac{1}{1 + \varepsilon^2} \quad (1.2a)$$

$$|H_a(j\Omega_s)|^2 = \frac{1}{1 + (\Omega_s / \Omega_c)^{2N}} = \frac{1}{A^2} \quad (1.2b)$$

By solving the above we get the expression for the order N as,

$$N = \frac{1}{2} \frac{\log_{10}[(A^2 - 1)/\epsilon^2]}{\log_{10}(\Omega_s / \Omega_p)} = \frac{\log_{10}(1/k_1)}{\log_{10}(1/k)} \quad (1.3)$$

Since the order N of the filter must be an integer, the value of N computed using the above expression is rounded up to the next higher integer. This value of N can be used next in either Eq. (1.2a) or (1.2b) to solve for the 3-dB cutoff frequency Ω_c . If it is used in Eq. (1.2a), the pass-band specification is met exactly, whereas the stop-band specification is exceeded. On the Other hand, if it is used in Eq. (1.2b), the stop-band specification is met exactly, whereas the pass-band specification is exceeded.

The expression for the transfer function of the Butterworth low-pass filter is given by,

$$H_a(s) = \frac{C}{D_N(s)} = \frac{\Omega_c^N}{s^N + \sum_{\ell=0}^{N-1} d_\ell s^\ell} = \frac{\Omega_c^N}{\prod_{\ell=1}^N (s - p_\ell)} \quad (1.4)$$

Where,

$$p_\ell = \Omega_c e^{j[\pi(N+2\ell-1)/2N]}, \quad \ell = 1, 2, \dots, N \quad (1.5)$$

The denominator $D_N(s)$ of Eq. (1.4) is known as the Butterworth polynomial of order N and is easy to compute.

1.2.2 Chebyshev Approximation

In this case, the approximation error, defined as the difference between the ideal brick wall characteristic and the actual response, is minimized over a prescribed band of frequencies. In fact, the magnitude error is equiripple in the band. There are two types of Chebyshev transfer functions [2]. In the Type 1 approximation, the magnitude characteristic is equiripple in the pass-band and monotonic in the stop-band, whereas in the Type 2 approximation, the magnitude response is monotonic in the pass-band and equiripple in the stop-band.

1.2.3 Type 1 Chebyshev Approximation

The type 1 Chebyshev transfer function $H_a(s)$ has a magnitude response given by,

$$|H_a(j\Omega)|^2 = \frac{1}{1 + \varepsilon^2 T_N^2(\Omega/\Omega_p)}, \quad (1.7)$$

Where $T_N(\Omega)$ is the Chebyshev polynomial of order N :

$$T_N(\Omega) = \begin{cases} \cos(N \cos^{-1} \Omega), & |\Omega| \leq 1, \\ \cosh(N \cosh^{-1} \Omega), & |\Omega| > 1, \end{cases} \quad (1.8)$$

The above polynomial can also be derived by recurrence relation given by,

$$T_r(\Omega) = 2\Omega T_{r-1}(\Omega) - T_{r-2}(\Omega), \quad r \geq 2, \quad (1.9)$$

With $T_0(\Omega) = 1$ and $T_1(\Omega) = \Omega$

The zeros are on the $j\Omega$ -axis and are given by,

$$z_\ell = j \frac{\Omega_s}{\cos\left[\frac{(2\ell-1)\pi}{2N}\right]}, \quad \ell = 1, 2, \dots, N. \quad (1.10)$$

If N is odd, then for $\ell = (N+1)/2$, the zero is at $s = \infty$. The poles are located at,

$$p_\ell = \sigma_\ell + j\Omega_\ell, \quad \ell = 1, 2, \dots, N, \quad (1.11)$$

Where,

$$\begin{aligned} \sigma_\ell &= \frac{\Omega_s \alpha_\ell}{\alpha_\ell^2 + \beta_\ell^2}, & \Omega_\ell &= \frac{\Omega_s \beta_\ell}{\alpha_\ell^2 + \beta_\ell^2}, \\ \alpha_\ell &= -\Omega_p \varsigma \sin\left[\frac{(2\ell-1)\pi}{2N}\right], & \beta_\ell &= \Omega_p \xi \cos\left[\frac{(2\ell-1)\pi}{2N}\right], \\ \varsigma &= \frac{\gamma^2 - 1}{2\gamma}, & \xi &= \frac{\gamma^2 + 1}{2\gamma}, & \gamma &= \left(A + \sqrt{A^2 - 1}\right)^{1/N}. \end{aligned} \quad (1.12)$$

The order N of the Type 2 Chebyshev low-pass filter is determined from given ϵ , Ω_s , and A using Eq. (1.11).

1.2.4 Elliptic Approximation:

An elliptic filter [8], also known as a Cauer filter, has an equiripple pass-band and an equiripple stop-band magnitude response, as indicated in Figure 1.4 for typical elliptic low-pass filters. The transfer function of an elliptic filter meets a given set of filter specifications, pass-band edge frequency Ω_p , stop-band edge frequency Ω_s , pass-band ripple ϵ , and minimum stop-band attenuation A , with the lowest filter order N . The theory of elliptic filter approximation is mathematically quite involved. The square-magnitude response of an elliptic low-pass filter is given by,

$$|H_a(j\Omega)|^2 = \frac{1}{1 + \epsilon^2 R_N^2(\Omega/\Omega_p)} \quad (1.13)$$

where $R_N(\Omega)$ is a rational function of order N satisfying the property $R_N(1/\Omega) = 1/R_N(\Omega)$, with the roots of its numerator lying within the interval $0 < \Omega < 1$ and the roots of its denominator lying in the interval $1 < \Omega < \infty$. For most applications, the filter order meeting a given set of specifications of pass-band edge frequency Ω_p , pass-band ripple ϵ , stop-band edge frequency Ω_s , and the minimum stop-band ripple A can be, estimated by using the approximate formula,

$$N \cong \frac{2 \log_{10}(4/k_1)}{\log_{10}(1/p)} \quad (1.14)$$

where k_1 is the discrimination parameter and p is computed as follows :

$$\begin{aligned}
 k' &= \sqrt{1 - k^2} \\
 \rho_0 &= \frac{1 - \sqrt{k'}}{2(1 + \sqrt{k'})} \\
 \rho &= \rho_0 + 2(\rho_0)^5 + 15(\rho_0)^9 + 150(\rho_0)^{13}.
 \end{aligned} \tag{1.15}$$

in Eq.(1.15a), k is the selective parameter.

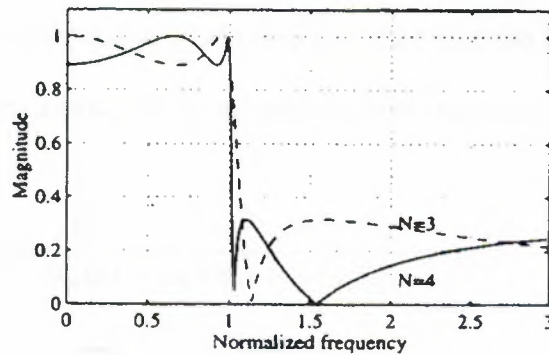


Figure 1.4 Typical elliptic low-pass filter responses with 1 dB pass-band ripple and 10 dB minimum stop-band attenuation.

1.2.5 Linear-Phase Approximation

The previous three approximation techniques are for developing analog low-pass transfer functions meeting specified magnitude or gain response specifications without any concern for their phase responses. In a number of applications it is desirable that the analog low-pass, filter being designed have a linear-phase characteristic in the pass-band, in addition to approximating the magnitude specifications. One way to achieve this goal is to cascade an

analog all-pass filter with the filter designed to meet the magnitude specifications, so that the phase response of the overall cascade realization approximates linear-phase response in the pass-band. This approach increases the overall hardware complexity of the analog filter and may not be desirable for designing an analog anti-aliasing filter in some A/D conversion or designing an analog reconstruction filter in D/A conversion applications. It is possible to design a low-pass filter that approximates a linear-phase characteristic in the pass-band but with a poorer magnitude response than that can be achieved by the previous three techniques. Such a filter has an all-pole transfer function of the form,

$$H(s) = \frac{d_0}{B_N(s)} = \frac{d_0}{d_0 + d_1s + \dots + d_{N-1}s + s^N} \quad (1.16)$$

and provides a maximally flat approximation to the linear-phase characteristic at $\Omega = 0$, i.e., has a maximally flat constant group delay at dc ($\Omega = 0$). For a normalized group delay of unity at dc, the denominator polynomial $B_N(s)$ of the transfer function, called the Bessel polynomial[2,8], can be derived via the recursion relation,

$$B_N(s) = (2N-1)B_{N-1}(s) + s^2B_{N-2}(s) \quad (1.17)$$

starting with $B_1(s) = s + 1$ and $B_2(s) = s^2 + 3s + 3$. Alternatively, the coefficients of the Bessel polynomial $B_N(s)$ can be found from,

$$d_\ell = \frac{(2N-\ell)!}{2^{N-\ell} \ell! (N-\ell)!}, \quad \ell = 0, 1, \dots, N-1 \quad (1.26)$$

These filters are often referred to as Bessel filters.

1.3 A Comparison of the Filter Types

In the previous sections we have discussed four types of analog low-pass filter approximations, three of which have been developed primarily to meet the magnitude response specifications while the fourth has been developed primarily to provide a linear-phase approximation. In order to determine which filter type to choose to meet a given magnitude response specification, we need to compare the performances of the four types of approximations. To this end, we compare here the frequency responses of the normalized Butterworth, Chebyshev, and elliptic analog low-pass filters of same order. The pass-band ripple of the Type 1 Chebyshev and the equiripple filters are assumed to be the same, while the minimum stop-band attenuation of the Type 2 Chebyshev and the equiripple filters are assumed to be the same. The filter specifications used for comparison are as follows: filter order of 6, pass-band edge at $\Omega = 1$, maximum pass-band deviation of 1 dB, and minimum stop-band attenuation of 40 dB. The frequency responses computed using MATLAB are plotted in Figure 1.5.

As can be seen from Figure 1.5, the Butterworth filter has the widest transition band, with a monotonically decreasing gain response. Both types of Chebyshev filters have

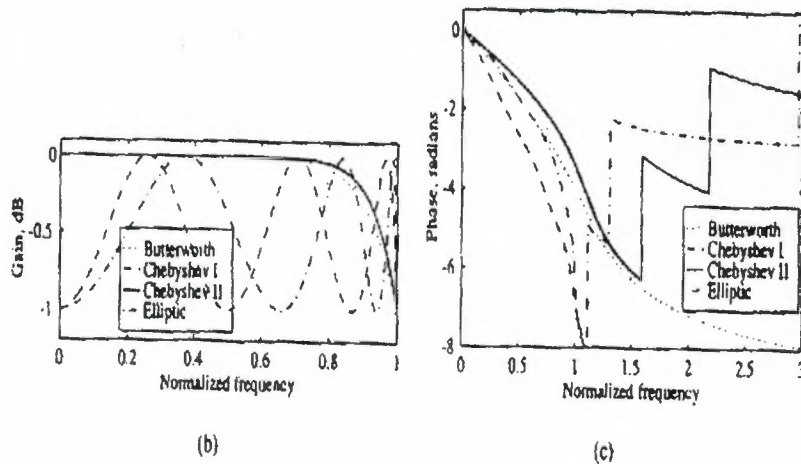


Figure 1.5 A comparison of the frequency response of the four types of analog low-pass.

A transition band of equal width that is smaller than that of the Butterworth filter but greater than that of the elliptic filter. The Type 1 Chebyshev filter provides a slightly faster roll-off in the transition band than the Type 2 Chebyshev filter. The magnitude response of

the Type 2 Chebyshev filter in the pass-band is nearly identical to that of the Butterworth filter. The elliptic filter has the narrowest transition band, with an equiripple pass-band and an equiripple stop band response.

The Butterworth and Chebyshev filters have a nearly linear-phase response over about three-fourths of the pass-band, whereas the elliptic filter has a nearly linear-phase response over about one-half of the pass-band. On the other hand, the Bessel filter may be more attractive if the linearity of the phase response over a larger portion of the pass-band is

desired at the expense of a poorer gain response. However, the Bessel filter provides a minimum of 40 dB attenuation at approximately $\Omega = 9.4$ and as a result, has the largest transition band compared to the other three types.

2. ADAPTIVE FILTERS

2.1 Overview

The design of a Wiener filter requires *a priori* information about the statistics of the data to be processed. The filter is optimum only when the statistical characteristics of the input data match the *priori* information on which the design of the filter is based. When this information is not known completely, however, it may not be possible to design the Wiener filter or else the design may no longer be optimum.

A straightforward approach that we may use in such situations is the "estimate and plug" procedure. This is a two-stage process whereby the filter first "estimates" the statistical parameters of the relevant signals and then "plugs" the results so obtained into a *non-recursive* formula for computing the filter parameters. For *real-time* operation, this procedure has the disadvantage of requiring excessively elaborate and costly hardware. A more efficient method is to use an *adaptive filter*. By such a device we mean one that is *self-designing* in that the adaptive filter relies for its operation on a *recursive algorithm*, which makes it possible for the filter to perform satisfactorily in an environment where complete knowledge of the relevant signal characteristics is not available. The algorithm starts from some predetermined set of *initial conditions*, representing whatever we know about the environment. Yet, in a stationary environment, we find that after successive iterations of the algorithm it *converges* to the optimum Wiener solution in some statistical sense.

In a non-stationary environment, the algorithm offers a *tracking* capability, in that it can track time variations in the statistics of the input data, provided that the variations are sufficiently slow.

As a direct consequence of the application of a recursive algorithm whereby the parameters of an adaptive filter are updated from one iteration to the next, the parameters become *data dependent*. This, therefore, means that an adaptive filter is in reality a *nonlinear device, in the sense that it does not obey the principle of superposition*. The adaptive filters are commonly classified as linear or nonlinear. An adaptive filter is said to be *linear* if the estimate of a quantity of interest is computed adaptively (at the output of the filter) as a *linear combination of the available set of observations applied to the filter input*. Otherwise, the adaptive filter is said to be *nonlinear*.

A wide variety of recursive algorithms have been developed in the literature for the operation of linear adaptive filters [13]. In the final analysis, the choice of one algorithm over another is determined by one or more of the following factors:

- *Rate of convergence*. This is defined as the number of iterations required for the algorithm, in response to stationary inputs, to converge "close enough" to the optimum Wiener solution in the mean-square sense. A fast rate of convergence allows the algorithm to adapt rapidly to a stationary environment of unknown statistics.
- *Misadjustment*. For an algorithm of interest, this parameter provides a quantitative measure of the amount by which the final value of the mean-squared error, averaged over an ensemble of adaptive filters, deviates from the minimum mean-squared error that is produced by the Wiener filter.
- *Tracking*. When an adaptive filtering algorithm operates in a non-stationary environment, the algorithm is required to *track* statistical variations in the environment. The tracking performance of the algorithm, however, is influenced by two contradictory features: (a) rate of convergence, and (b) steady-state fluctuation due to algorithm noise.

- *Robustness.* For an adaptive filter to be *robust*, small disturbances (i.e., disturbances with small energy) can only result in small estimation errors. The disturbances may arise from a variety of factors, internal or external to the filter.
- *Computational requirements.* Here the issues of concern include (a) the number of operations (i.e., multiplications, divisions, and additions/subtractions) required to make one complete iteration of the algorithm, (b) the size of memory locations required to store the data and the program, and (c) the investment required to program the algorithm on a computer.
- *Structure.* This refers to the structure of information flow in the algorithm, determining the manner in which it is implemented in hardware form. For example, an algorithm whose structure exhibits high modularity, parallelism, or concurrency is well suited for implementation using very large-scale integration (VLSI).
- *Numerical properties.* When an algorithm is implemented numerically, inaccuracies are produced due to *quantization errors*. The quantization errors are due to analog-to-digital conversion of the input data and digital representation of internal calculations. Ordinarily, it is the latter source of quantization errors that poses a serious design problem. In particular, there are two basic issues of concern: numerical stability and numerical accuracy. Numerical stability is an inherent characteristic of an adaptive filtering algorithm. Numerical accuracy, on the other hand, is determined by the number of *bits* (i.e., binary digits) used in the numerical representation of data samples and filter coefficients. An adaptive filtering algorithm is said to be numerically robust when it is insensitive to variations in the word-length used in its digital implementation.

These factors, in their own ways, also enter into the design of nonlinear adaptive filters, except for the fact that we now no longer have a well-defined frame of reference in the

form of a Wiener filter. Rather, we speak of a nonlinear filtering algorithm that may converge to a local minimum or, hopefully, a global minimum on the error-performance surface.

In recent years, growing field of research in "adaptive systems" has resulted in a variety of adaptive automatos whose characterestics in limited ways resemble certain characterestics of living systems and biological adaptive processes.

Some meanings of "adaptation" can be applied in industrial, biological, social and etc.

An adaptive automation is asystem whose structure is alterable or adjustable in such a way that its behavior or performance (according to some desired criterion) improves through contact with its environment. A simple example of an automation or automatic adaptive system is the automatic gain control (AGC) used in radio and television receivers. The function of this circuit is to adjust the sensitivity of the receiver inversely as the average incoming signal strength. The receiver is thus able to adapt a wide range of input levels and to produce a much narrower range of output signals.

The purpose of this work is to present certain basic principles of adaptation; to explain the design, operating characteristics, and applications of the simpler forms of adaptive systems; and to describe means for their physical realization. The types of systems discussed include those designed primarily for the purposes of adaptive control and adaptive signal processing. Such systems usually have some or all of the following characteristics:

1. They can automatically adapt (self-optimize) in the face of changing (nonstationary) environments and changing system requirements.
2. They can be trained to perform specific filtering and decision-making tasks.

Synthesis of systems having these capabilities can be accomplished

automatically through training. In a sense, adaptive systems can be “programmed” by a train process.

3. Because of the above, adaptive systems do not require the elaborate synthesis procedures usually needed for nonadaptive systems. Instead, they tend to be “self-designing.”
4. They can extrapolate a model of behavior to deal with new situations after having been trained on a finite and often small number of training signals or patterns.
5. To a limited extent, they can repair themselves; that is, they can adapt around certain kinds of internal defects.
6. They can usually be described as nonlinear systems with time-varying parameters.
7. Usually, they are more complex and difficult to analyze than nonadaptive systems, but they offer the possibility of substantially increased system performance when input signal characteristics are unknown or time varying.

2.2 General Properties

The essential and principal property of the adaptive system is its time-varying, self-adjusting performance. The need for such performance may readily be seen by realizing that if a designer develops a system of fixed design which he or she considers optimal, the implications are that the designer has foreseen all possible input conditions, at least statistically, and knows what he or she would like the system to do under each of these conditions. The designer has then chosen a specific criterion whereby performance is to be judged, such as

the amount of error between the output of the actual system and that of some selected model or "ideal" system.

Finally, the designer has chosen the system that appears best according to the performance criterion selected, generally choosing this system from an a priorirestricted class of designs (such as linear systems).

In many instances, however, the complete range of input conditions may not be known exactly, or even statistically; or the conditions may change from time to time. In such circumstances, an adaptive system that continually seeks the optimum within an allowed class of possibilities, using an ordinary search process, would give superior performance compared with a system of fixed design.

By their very nature, adaptive systems must be time varying and nonlinear. Their characteristics depend, among other things, on their input signals. If an input signal x_1 is applied, an adaptive system will adapt to it and produce an output y_1 . If another input signal, x_2 , is applied, the system will adapt to this second signal and will again produce an output y_2 .

Generally, the form or the structure or the adjustments of the adaptive system will be different for the two different inputs. If the sum of the two inputs is applied to the adaptive system, the latter will adapt to this new input-but it will produce an output that will generally not be the same as y_1+y_2 , the sum of the outputs that would have corresponded to inputs x_1 and x_2 . In such a case, as illustrated in Figure 2.1, the principle of superposition does not work as it does with linear systems. If a signal is applied to the input of an adaptive system to test its response characteristics, the systems adapts to this specific input and

thereby changes its own form. Thus the adaptive system is inherently difficult to characterize in conventional terms.

Whithin the realm of nonlinear systems, adaptive systems cannot be distinguished as belonging to an absolutely clear subset. However, they have two features that generally distinguish them from other forms of nonlinear systems.

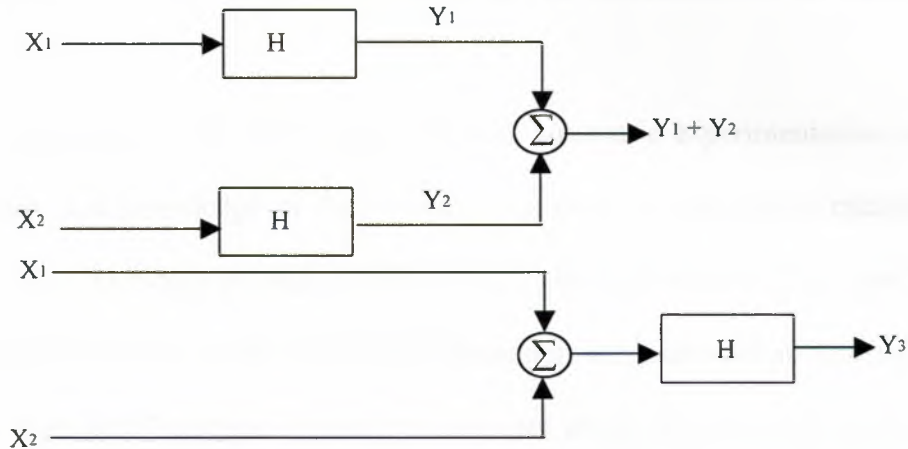


Figure 2.1. The lower output Y_3 if H is a linear system, if H is adaptive Y_3 is generally different from $Y_1 + Y_2$

First, adaptive systems are adjustable, and their adjustments usually depend on finite-term average signal characteristics rather than on instantaneous values of signals or instantaneous values of the internal system state. Second, the adjustments of the adaptive systems are changed purposefully in order to optimize specified performance measures.

Certain forms of adaptive systems become linear systems when their adjustments are held constant after adaptation. These may be called "linear adaptive systems." They are very useful; they tend to be mathematically tractable; and they are generally easier to design than other forms of adaptive systems.

2.3 Open-And Closed-Loop Adaptation

Several ways to classify adaptive schemes have been proposed in the literature [4]. It is most convenient here to begin by thinking in terms of open-loop and closed-loop adaptation. The open-loop adaptive process involves making measurements of input or environmental characteristics, applying this information to a formula or to a computational algorithm, and using the results to set the adjustments of the adaptive system.

Closed-loop adaptation, on the other hand, involves automatic experimentation with these adjustments and knowledge of their outcome in order to optimize a measured system performance. The latter process called adaptation by “performance feedback.”

The principles of open- and closed-loop adaptation are illustrated in figures 2.2 and 2.3. The “other data” in these figures may be data about the environment of the adaptive system, or in the closed-loop case, it may be a desired version of the output signal.

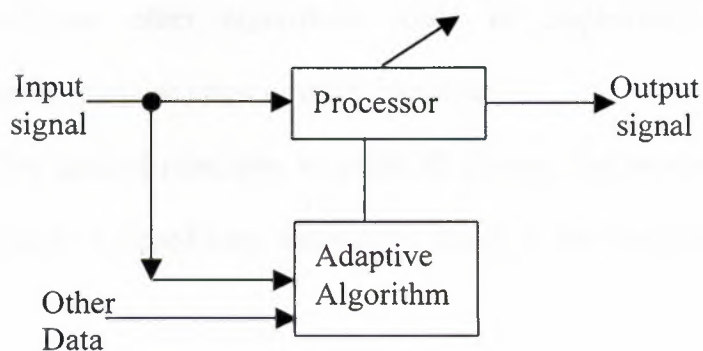


Figure 2.2 Open-loop adaptations

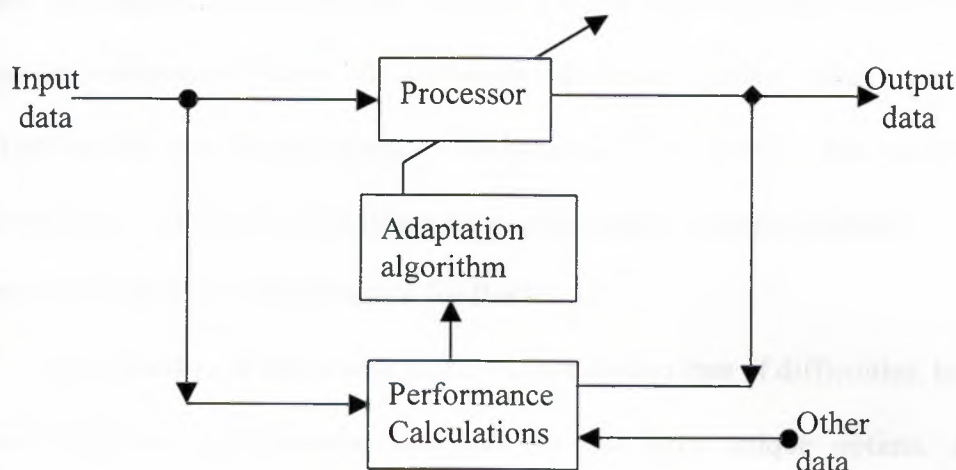


Figure 2.3 Closed loop adaptation

When designing an adaptive process, many factors determine the choice of closed-loop versus open-loop adaptation. The availability of input signals and performance-indicating signals is a major consideration. Also, the amount of computing capacity and the type of computer required to implement the open-loop and closed-loop adaptation algorithms will generally differ. Certain algorithms require the use of a general-purpose digital computer, whereas other algorithms could be implemented far more economically with special-purpose chips or other apparatus.

It is difficult to develop general principles to guide all choices, but several advantages and a few disadvantages of closed-loop adaptation, which is the main subject can be pointed out here.

Closed-loop adaptation has the advantages of being workable in many applications where no analytic synthesis procedure either exists or is known, for example, where error criteria other than mean-square are used, where systems are nonlinear or time variable, where signals are nonsatationary, and so on.

Closed-loop can also be used effectively in situations where physical system component values are variable or inaccurately known. Closed-loop adaptation will find the best choice of component values. In the event of partial system failure, an adaptation mechanism that continually monitors performance will optimize this performance by adjusting and reoptimizing the intact parts. As a result, system reliability can often be improved by the use of performance feedback.

The closed-loop adaptation process is not always free of difficulties, however. In certain situations, performance functions do not have unique optima. Automatic optimization is an uncertain process in such situations. In other situations, the closed-loop adaptation process, like a closed-loop control system, could be unstable. The adaptation process could diverge rather than converge. In spite of these possibilities, performance feedback is a powerful, widely applicable technique for implementing adaptation.

2.4 Applications

The ability of an adaptive filter to operate satisfactorily in an unknown environment and track time variations of input statistics make the adaptive filter a powerful device for signal-processing and control applications. Indeed, [4] adaptive filters have been successfully applied in such diverse fields as communications, radar, sonar, seismology, and biomedical engineering. Although these applications are indeed quite different in nature, nevertheless, they have one basic common feature: an input vector and a desired response are used to compute an estimation error, which is in turn used to control the values of a set of adjustable filter coefficients. The adjustable coefficients may take the form of tap weights, reflection coefficients, rotation parameters, or synaptic weights, depending on the filter structure employed. However, the essential difference between

the various applications of adaptive filtering arises in the manner in which the desired response is extracted. In this context, we may distinguish four basic classes of adaptive filtering applications, as depicted in Fig. 2.4. For convenience of presentation, the following notations are used in this figure:

u = input applied to the adaptive filter

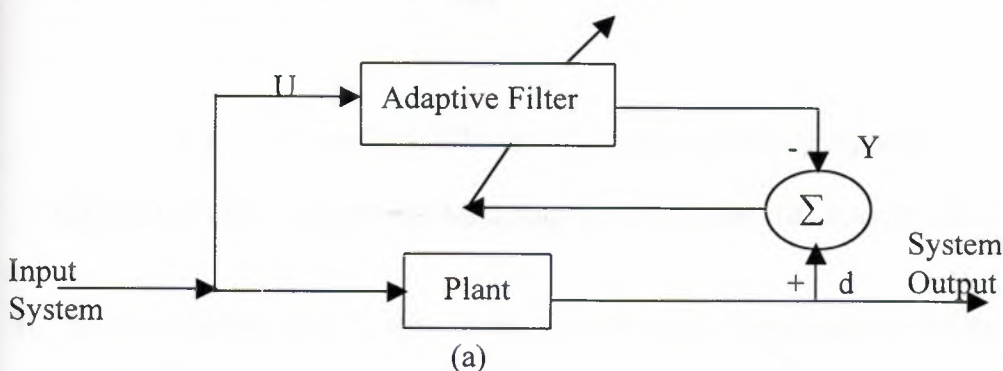
Y = output of the adaptive filter

d = desired response

$e = d - y$ = estimation error .

The functions of the four basic classes of adaptive filtering applications depicted here in are as follows:

I. Identification Fig. 2.4(a). The notion of a *mathematical model* is fundamental to sciences and engineering. In the class of applications dealing with identification, an adaptive filter is used to provide a linear model that represents the best fit (in some sense) to an *unknown plant*. The plant and the adaptive filter are driven by the same input. The plant output supplies the desired response of the adaptive filter. If the plant is dynamic in nature, the model will be time varying.



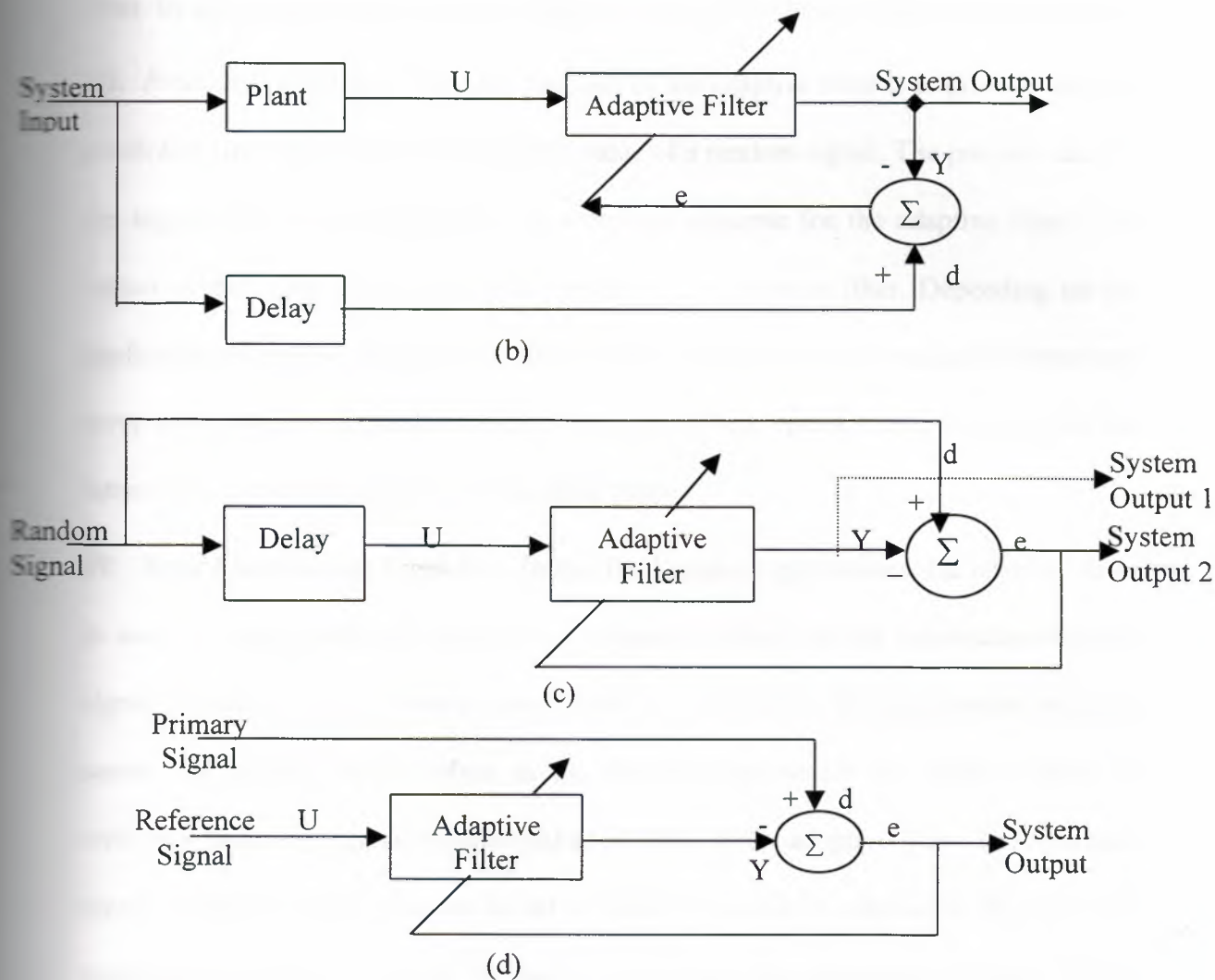


Fig. 2.4 Four Basic Classes of Adaptive Filtering Applications

(a) Identification (b) Inverse Modeling (c) Prediction (d) Interference Canceling

II. Inverse modeling Fig. 2.4(b). In this second class of applications, the function of the adaptive filter is to provide an *inverse model* that represents the best fit (in some sense) to an *unknown noisy plant*. Ideally, in the case of a linear system, the inverse model has

a transfer function equal to the *reciprocal (inverse)* of the plant's transfer function, such that the combination of the two constitutes an ideal transmission medium. A delayed version of the plant (system) input constitutes the desired response for the adaptive filter. In some applications, the plant input is used without delay as the desired response.

III. Prediction Fig.2.4(c). Here the function of the adaptive filter is to provide the best *prediction* (in some sense) of the present value of a random signal. The present value of the signal thus serves the purpose of a desired response for the adaptive filter. Past values of the signal supply the input applied to the adaptive filter. Depending on the application of interest, the adaptive filter output or the estimation (prediction) error may serve as the system output. In the first case, the system operates as a *predictor*; in the latter case, it operates as a *prediction-error filter*.

IV. Noise Cancellation Fig.2.4(d). In this final class of applications, the adaptive filter is used to cancel *unknown interference* contained (alongside an information-bearing signal component) in a *primary signal*, with the cancellation being optimized in some sense. The primary signal serves as the desired response for the adaptive filter. A *reference (auxiliary) signal* is employed as the input to the adaptive filter. The reference signal is derived from a sensor or set of sensors located in relation to the sensor(s) supplying the primary signal in such a way that the information-bearing signal component is weak or essentially undetectable.

2.5 Application of Adaptive Filters

The contamination of a signal of interest by other unwanted, often larger, signals or noise is a problem often encountered in many applications. Where the signal and noise occupy fixed and separate frequency bands, conventional linear filters with fixed coefficients are normally used to extract the signal. [4]. However, there are many

instances when it is necessary for the filter characteristics to be variable, adapted to changing signal characteristics, or to be altered intelligently. In such cases, the coefficients of the filter must vary and cannot be specified in advance. Such is the case where there is a spectral overlap between the signal and noise, see Figure 2.5. or if the band occupied by the noise is unknown or varies with time.

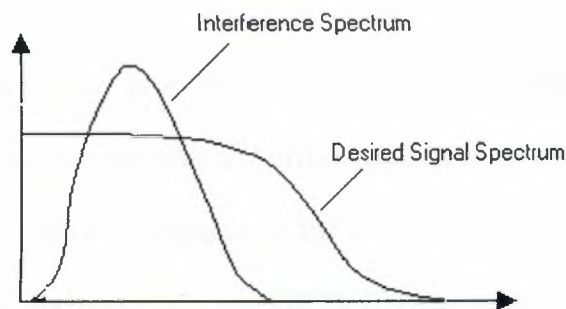


Figure 2.5. An illustration of spectral overlap between a signal and a strong interference

Typical applications where fixed coefficient filters are inappropriate are the following.

- Electroencephalography (EEG), where artefacts or signal contamination produced by eye movements or blinks is much larger than the genuine electrical activity of the brain and shares the same frequency band with signals of clinical interest. It is not possible to use conventional linear filters to remove the artefacts while preserving the signals of clinical interest.
- Digital communication using a spread spectrum, where a large jamming signal, possibly intended to disrupt communication, could interfere with the desired signal. The interference often occupies a narrow but unknown band within the wideband spectrum, and can only be effectively dealt with adaptively.

- In digital data communication over the telephone channel at a high rate. Signal distortions caused by the poor amplitude and phase response characteristics of the channel lead to pulses representing different digital codes to interfere with each other (intersymbol interference), making it difficult to detect the codes reliably at the receiving end. To compensate for the channel distortions which may be varying with time or of unknown characteristics at the receiving end, adaptive equalization is used.

An adaptive filter has the property that its frequency response is adjustable or modifiable automatically to improve its performance in accordance with some criterion, allowing the filter to adapt to changes in the input signal characteristics. Because of their self-adjusting performance and in-built flexibility, adaptive filters have found use in many diverse applications such as telephone echo canceling, radar signal processing, navigational systems, equalization of communication channels, and biomedical signal enhancement.

In summary we use adaptive filters

- When it is necessary for the filter characteristics to be variable, adapted to changing conditions,
- When there is spectral overlap between the signal and noise, or
- If the band occupied by the noise is unknown or varies with time.

In most adaptive systems, the digital filter in figure 2.6. Is realized using a transversal or finite impulse response (FIR) structure. Other forms are sometimes used, for example the infinite impulse response (IIR) or the lattice structures, but the FIR structure is the

most widely used because of its simplicity and guaranteed stability. For the N-point filter depicted in figure 2.6, the output is given by

$$\hat{n}_k = \sum_{i=0}^{N-1} w_k(i) x_{k-i} \quad (2.1)$$

where $w_k(i)$, $i=0,1,\dots$, are the adjustable filter coefficients (or weights) and $x_k(i)$ and \hat{x}_k are the input and output of the filter. Figure 2.6. Illustrates the single-input, single-output system. In a multiple-input single-output system, the x_k may be simultaneous inputs from N different signal sources.

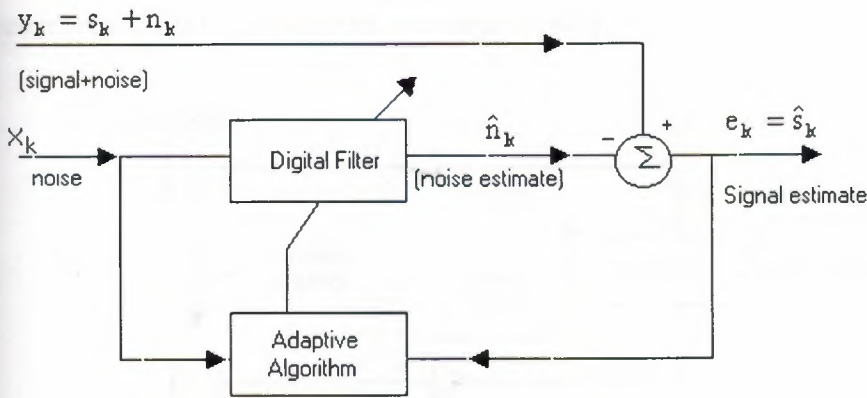


Figure 2.6. Block Diagram of an Adaptive Filter as a Noise Canceller

2.5.1 Loud speaking telephones

- The hybrid network is used to separate the transmit and receive paths (that is, the loudspeaker from the microphone), but there is a significant acoustic coupling between the loudspeaker and the microphone because of their proximity as well as a leakage across the imperfectly matched hybrid network (South et al., 1979).
- The difficulty then is how to provide adequate gain for the receive and transmit directions without causing instability.

- The conventional solution to the problems is to use a voice-activated switch to select the transmit and receive paths, but this is not satisfactory because it does not allow full duplex communication.
- A better solution is to use adaptive filtering techniques to estimate and control the acoustic and hybrid echoes Figure 2.7(b). The number of filter coefficients here can be quite large, for example 512, making the use of a fast algorithm attractive.
- In teleconferencing networks (or public address systems) acoustic feedback leads to problems similar to those described above. Adaptive filters used for these may require large numbers of coefficients (250 to 1000), especially in rooms with long reverberation times, and must converge rapidly.

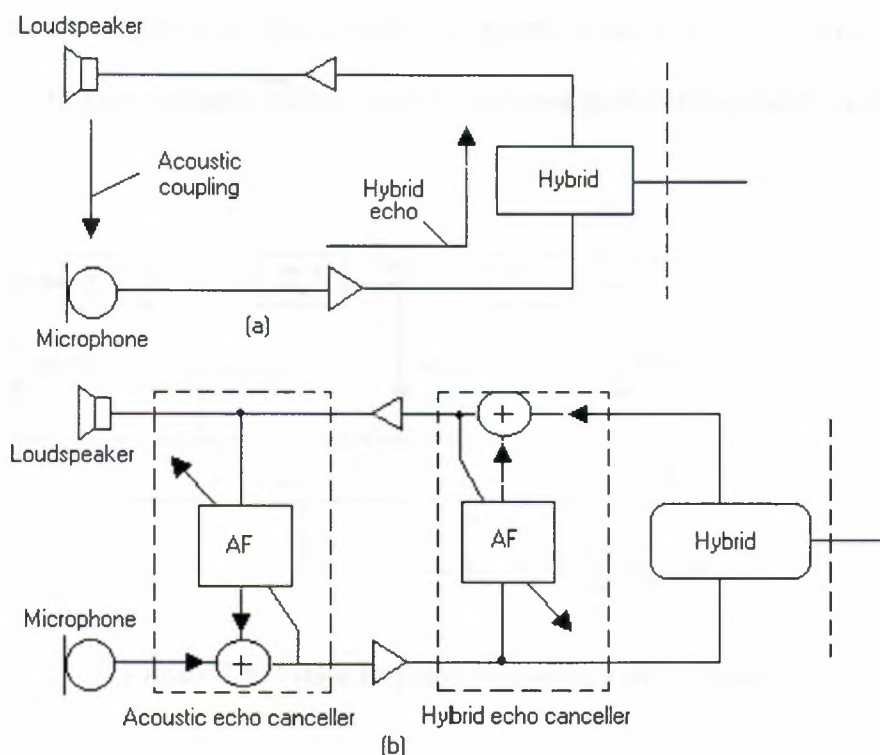


Figure 2.7. (a) Loud Speaking Telephone (b) Acoustic and Hybrid Echo Cancellation in Loud speaking Telephone

2.5.2 Radar Signal Processing

Adaptive signal processing techniques are widely used to solve a number of problems associated with radar. For example, adaptive filters are used in monostatic radar systems to remove or cancel clutter components from the desired target signals. In HF ground wave radar, adaptive filters are used to reduce co-channel interference which is a major problem in the HF band.

2.5.3 Separation of Speech signals from background noise

Acoustic background noise is a serious problem in speech processing. An adaptive filter may be used to enhance the performance of speech systems in noisy environments (for example in fighter aircrafts, tanks, cars) to improve both intelligibility and recognition of speech.

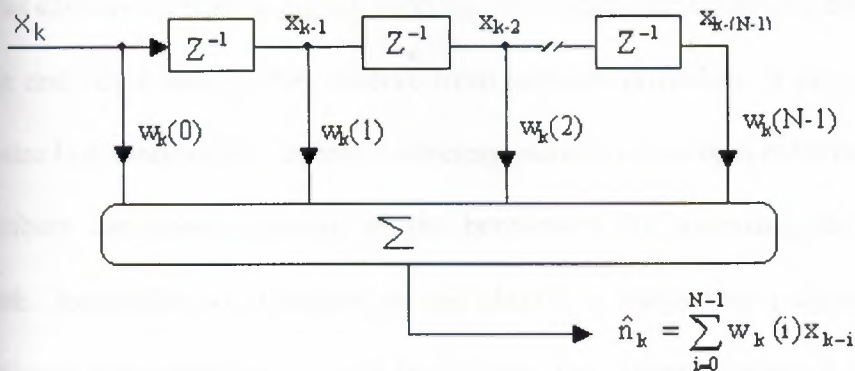


Figure 2.8. Finite Impulse Response Filter Structure

3. Adaptive Echo Cancellation

3.1 Overview, Definitions

The performance limits of adaptive echo cancellation techniques are investigated. In particular we analyze the effects of signal characteristics such as auto and cross correlation on the achievable echo suppression. Techniques to enhance signal characteristics such as to improve both the learning ability and the steady state echo suppression quality are identified. A nice feature of our work is that it links in a natural way the complexity of the learning task (via the dimension of the adapted parameter vector), the available information (via the signal characteristics) to the achievable echo suppression quality. [7].

Wireless phones are increasingly being regarded as essential communications tools, dramatically impacting how people approach day-to-day personal and business communications. As new network infrastructures are implemented and competition between wireless carriers increases, digital wireless subscribers are becoming ever more critical of the service and voice quality they receive from network providers. A key technology to provide near-wire line voice quality across a wireless carrier's network is echo cancellation. Subscribers use speech quality as the benchmark for assessing the overall quality of a network. Regardless of whether or not this is a subjective judgment; it is the key to maintaining subscriber loyalty. For this reason, the effective removal of hybrid and acoustic echo inherent within the digital cellular infrastructure is the key to maintaining and improving perceived voice quality on a call. This has led to intensive research into the area of echo cancellation, with the aim of providing solutions that can reduce background noise and remove hybrid and acoustic echo before any transcoder processing. By employing this technology, the overall efficiency of the coding can be enhanced, significantly improving the

quality of speech. This tutorial discusses the nature of echo and how echo cancellation is helpful in making mobile calls meet acceptable quality standards.

3.2 History of Echo Cancellation

The late 1950s marked the birth of echo control in the telecommunications industry with the development of the first echo-suppression devices. These systems, first employed to manage echo generated primarily in satellite circuits, were essentially voice-activated switches that transmitted a voice path and then turned off to block any echo signal. Although echo suppressers reduced echo caused by transmission problems in the network, they also resulted in choppy first syllables and artificial volume adjustment. In addition, they eliminated double-talk capabilities, greatly reducing the ability to achieve natural conversations.

Echo-cancellation theory was developed in the early 1960s by AT&T Bell Labs, followed by the introduction of the first echo-cancellation system in the late 1960s by COMSAT TeleSystems (previously a division of COMSAT Laboratories). COMSAT designed the first analog echo canceller systems to demonstrate the feasibility and performance of satellite communications networks. Based on analog processes, these early echo-cancellation systems were implemented across satellite communications networks to demonstrate the network's performance for long-distance, cross-continental telephony. These systems were not commercially viable, however, because of their size and manufacturing costs.

In the late 1970s, COMSAT TeleSystems developed and sold the first commercial analog echo cancellers, which were mainly digital devices with an analog interface to the network. The semiconductor revolution of the early 1980s marked the switch from analog to digital telecommunications networks. More sophisticated digital interface, multichannel echo-canceller systems were also developed to address new echo problems associated with long-distance digital telephony systems. Based on application-specific integrated circuit (ASIC) technology, these new echo cancellers' utilized high-speed digital signal-processing

techniques to model and subtract the echo from the echo return path. The result was a new digital echo-cancellation technique that outperformed existing suppression-based techniques, creating improved network performance.

The 1990s have witnessed explosive growth in the wireless telecommunications industry, resulting from deregulation that has brought to market new analog and digital wireless handsets, numerous network carriers, and new digital network infrastructures such as TDMA, CDMA, and GSM. According to the Cellular Telecommunications Industry Association (CTIA), new subscribers are driving the growth of the wireless market at an annual rate of 40 percent. With wireless telephony being widely implemented and competition increasing as new wireless carriers enter the market, superior voice transmission quality and customer service have now become key determining factors for subscribers evaluating a carrier's network. Understanding and overcoming the inherent echo problems associated with digital cellular networks will enable network operators and telecommunications to offer subscribers the network performance and voice quality they are demanding today.

3.3 Acoustic Echo

Acoustic echo is generated with analog and digital handsets, with the degree of echo related to the type and quality of equipment used. This form of echo is produced by poor voice coupling between the earpiece and microphone in handsets and hands-free devices. Further voice degradation is caused as voice-compressing encoding/decoding devices (vocoders) process the voice paths within the handsets and in wireless networks. This results in returned echo signals with highly variable properties. When compounded with inherent digital transmission delays, call quality is greatly diminished for the wire line caller.

Acoustic echo was first encountered with the early video/audioconferencing studios and now also occurs in typical mobile situations, such as when people are driving their cars. In this situation, sound from a loudspeaker is heard by a listener, as intended. However, this same

sound also is picked up by the microphone, both directly and indirectly, after bouncing off the roof, windows, and seats of the car. The result of this reflection is the creation of multipath echo and multiple harmonics of echo, which, unless eliminated, are transmitted back to the distant end and are heard by the talker as echo. Predominant use of hands-free telephones in the office has exacerbated the acoustic echo problem.

Acoustic echo cancellation is required in order to provide full duplex, fully interruptible speech. The acoustic echo canceller functions by modeling the speech being passed to the loudspeaker and removing any echoes picked up by the microphone. This type of operation necessitates a much more complex unit than is used in telephony in order to remove the many acoustic (multi-path) echoes generated with each syllable of speech. The tail circuit requirement, or the amount of time the canceller has to hold the power.

3.4 Hybrid Echo

Hybrid echo is the primary source of echo generated from the public-switched telephone network (PSTN). This electrically generated echo is created as voice signals are transmitted across the network via the hybrid connection at the two-wire/four-wire PSTN conversion points, reflecting electrical energy back to the speaker from the four-wire circuit. Hybrid echo has been around almost since the advent of the telephone itself. The signal path between two telephones, involving a call other than a local one, requires amplification using a four-wire circuit. Although not a factor in itself on digital cellular networks, hybrid echo becomes a problem in PSTN-originated calls. The cost and cabling required rules out the idea of running a four-wire circuit out to the subscriber's premise from the local exchange. For this reason, an alternative solution had to be found. Hence, the four-wire trunk circuits were converted to two-wire local cabling,

using a device called a "hybrid" (see Figure 3.1).

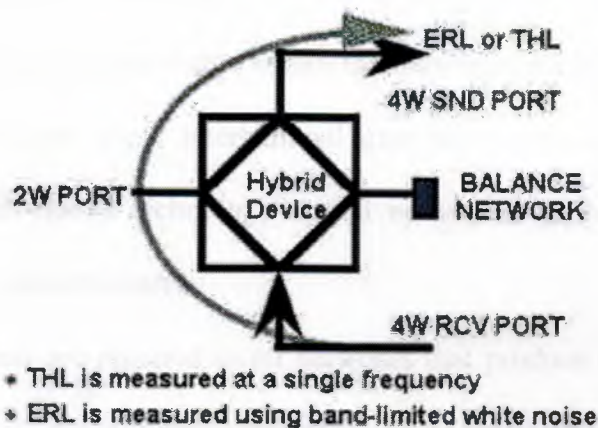


Figure 3.1 Hybrid Echo

Unfortunately, the hybrid is by nature a leaky device. As voice signals pass from the four-wire to the two-wire portion of the network, the energy in the four-wire section is reflected back on itself, creating the echoed speech. Provided that the total round-trip delay occurs within just a few milliseconds (i.e., within 28 ms), it generates a sense that the call is live by adding side tone, which makes a positive contribution to the quality of the call.

In cases where the total network delay exceeds 36 ms, however, the positive benefits disappear, and intrusive echo results. The actual amount of signal that is reflected back depends on how well the balance circuit of the hybrid matches the two-wire line. In the vast majority of cases, the match is poor, resulting in a considerable level of signal reflecting back. This is measured as echo return loss (ERL). The higher the ERL, the lower the reflected signal back to the talker, and vice versa. [7].

Acoustic echo apart, background noise is generated through the network when analog and digital phones are operated in hands-free mode. As the microphone directly and indirectly picks up additional sounds, multi-path audio is created and transmitted back to the talker. The surrounding noise, whether in an automobile or in a crowded, public environment, passes through the digital cellular vocoder, causing distorted speech for the wire line caller.

Digital processing delays and speech-compression techniques further contribute to echo generation and degraded voice quality in wireless networks. Delays are encountered as signals are processed through various routes within the networks, including copper wire, fiber optic lines, microwave connections, international gateways, and satellite transmission. This is especially true with mixed technology digital networks, where calls are processed across numerous network infrastructures.

Echo-control systems are required in all networks that produce one-way time delays greater than 16 ms. In today's digital wireless networks, voice paths are processed at two points in the network within the mobile handset and at the radio frequency (RF) interface of the network. As calls are processed through vocoders in the network, speech processing delays ranging from 80 ms to 100 ms are introduced, resulting in an unacceptable total end-to-end delay of 160 ms to 200 ms. As a result, echo cancellation devices are required within the wireless network to eliminate the hybrid and acoustic echoes in a digital wireless call.

3.5 The Combined Problem on Digital Cellular Networks

To deal with hybrid echo created by vocoder processing delays, it is mandatory for digital cellular mobile calls to have a group echo canceller installed—even for local calls. As a result, all calls on to the PSTN must pass through an echo canceller to remove what would otherwise be a noticeable and annoying echo, as shown in Fig 3.2

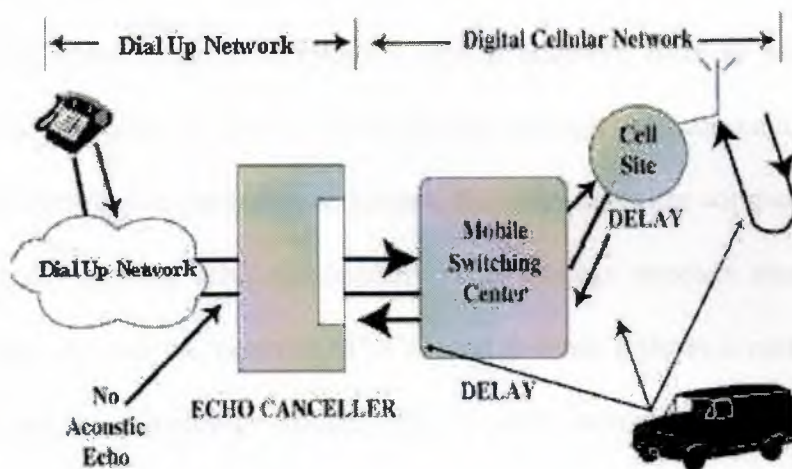


Figure 3.2 Digital Cellular Networks

For example, consider a digital cellular mobile user who makes a call to the PSTN without an echo canceller in place. The user would hear his or her own speech being echoed back 180 ms or more later, even if the called person is in the same locality. The mobile user will either be using a hands-free system installed in his or her vehicle or a hand portable. In either case, these units will involve the occurrence of direct and indirect coupling between the microphone and the speaker, creating acoustic echo. In this situation, however, it is the PSTN user who suffers by experiencing poor speech quality. Hence, the echo canceller installed in the digital cellular network must be capable of handling both sources of echoes.

3.6 Process of Echo Cancellation

In modern telephone networks, echo cancellers are typically positioned in the digital circuit, as shown in Figure 3.3 the process of canceling echo involves two steps. First, as the call is set up, the echo canceller employs a digital adaptive filter to set up a model or characterization of the voice signal and echo passing through the echo canceller. As a voice path passes back through the cancellation system, the echo canceller compares the signal and the model to cancel existing echo dynamically. This process removes more than 80 to 90 percent of the echo across the network. The second process utilizes a non-linear processor (NLP) to eliminate the remaining residual echo by attenuating the signal below the noise floor.

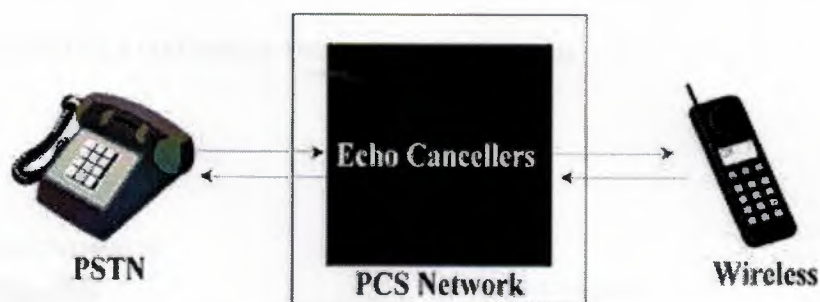


Figure 3.3 Typical Locations of Echo Cancellers

Today's digital cellular network technologies, namely TDMA, CDMA, and GSM, require significantly more processing power to transmit signal paths through the channels. As these technologies become even more sophisticated, echo control will be more complex. Echo cancellers designed with standard digital signal processors (DSPs), which share processing time in a circuit within a channel or across channels, provide a maximum of only 128 ms of cancellation and are unable to cancel acoustic echo. With network delays occurring in excess of 160 ms in today's mixed-signal network infrastructures, a more powerful, application-

Specific echo-cancellation technology is required to control echo across wireless networks effectively

7 Controlling Acoustic Echo

In echo cancellation, complex algorithmic procedures are used to compute speech models. This involves generating the sum from reflected echoes of the original speech, and then subtracting this from any signal the microphone picks up. The result is the purified speech of the person talking. The echo canceller in a process known as adaptation must learn the format of this echo prediction. It might be said that the parameters learned from the adaptation process generate the prediction of the echo signal, which then forms an audio picture of the room in which the microphone is located. Figure 3.4 shows the basic operation of an echo canceller in a conference room type of situation.

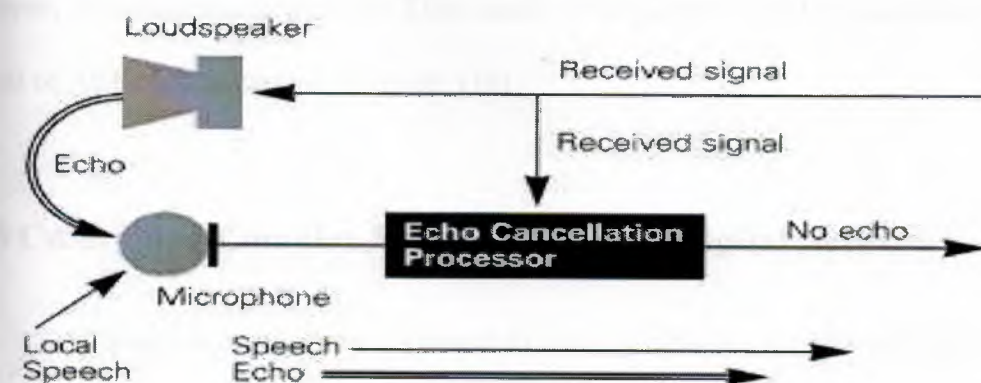


Figure 3.4 Operation of an Acoustic Echo Cancellation

During the conversation period, this audio picture constantly alters, and, in turn, the canceller must adapt continually. The time required for the echo canceller to fully learn the acoustic

picture of the room is called the convergence time. The best convergence time recorded is 50 ms, and any increase in this number results in syllables of echo being detected.

Other important performance criteria involve the acoustic echo canceller's ability to handle acoustic tail circuit delay. This is the time span of the acoustic picture and roughly represents the delay in time for the last significant echo to arrive at the microphone. The optimum requirement for this is currently set at 270 ms—any time below this could result in echoes being received by the microphone outside the ability of the echo canceller to remove them, and hence in participants hearing the echoes.

Another important factor is acoustic echo return loss enhancement (AERLE). This is the amount of attenuation, which is applied to the echo signal in the process of echo cancellation—i.e., if no attenuation is applied, full echo will be heard. A value of 65 dB is the minimum requirement with the non-linear processor enabled, based on an input level of -10 dBm white noise electrical and 6 dB of echo return loss (ERL).

The canceller's performance also relies heavily on the efficiency of a device called the center clipper, or non-linear processor. This needs to be adaptive and has a direct bearing on the level of AERLE that can be achieved. [14].

2.8 Controlling Complex Echo in a Wireless Digital Network

Although acoustic echo is present in every hands-free mobile call, the amount of echo depends on the particular handset design and model that the mobile user has. On the market are a few excellent handsets that limit the echo present, but, due to strong price pressures, most handsets do not control the echo very well at all—in fact, some phones on the market have been determined to have a terminal compiling loss of 24 dB. Echo becomes a problem when the processing inherent to the digital wireless network adds an additional delay

(typically in excess of 180 ms round-trip). This combination makes for totally unacceptable call quality for the fixed network customer, as shown in Fig. 3.5

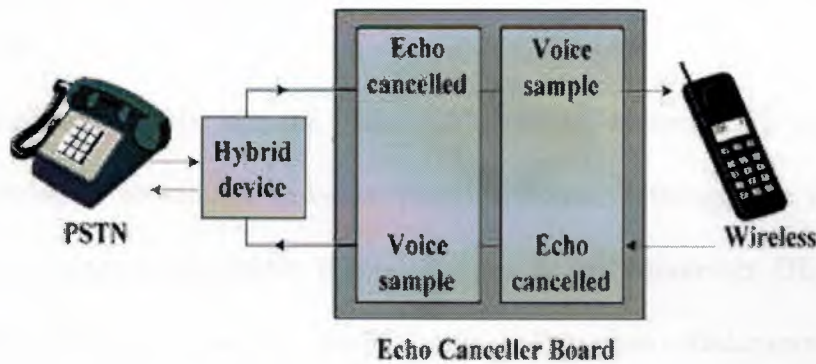


Figure 3.5 Bi-directional Echo Cancellation

This back-to-back configuration ensures a high audio quality for both PSTN and mobile customers. In addition, the echo canceller's software configuration is designed to provide a detailed analysis of background noises, including acoustic echo from the mobile user's end. Some echo cancellers incorporate a user-settable network delay, which enables network operators to fine-tune the echo control to suit their parameters via a menu option on the canceller's hand-held terminal or on the network management system (NMS).

Applying effective echo control via the echo-cancellation platform is one way of improving the overall call clarity on digital cellular networks. Another derives from improvements that must be made within the handset or terminal itself. There also is considerable room to enhance the network itself, focusing principally on vocoder development.

Recent headlines have charted the ongoing commercial battles regarding which digital technologies will eventually emerge as the winners, as equipment manufacturers fight it out. However, this public battle will soon be overshadowed by another battle concerning handsets.

At present, there are four major players in the digital cordless market. Europe has cordless telephony (CT2) and digital European cordless telephony (DECT), while Japan has the personal handy phone system (PHS) and the United States has personal communications services (PCS).

Connecting directly into the plain old telephone system, CT2 was one of the first digital technologies to provide low-cost mobile phones. Although the technology worked well, it had a fundamental problem: it could not handle cell handovers. DECT and GSM have overcome this problem and will eventually dominate European cellular services.

During the development of early cordless telephony, attention was paid to basic and enhanced functions and interworking with different network architectures. While the early generation of handsets looked very elegant and aesthetically pleasing, very little attention was paid to designing the handset with echo suppression/cancellation in mind. The result was that they looked good but were extremely poor at reducing acoustic echo.

In the setting of standards for GSM and PCS, handset design and the impact of different design approaches on call quality were researched. As a result, recommendations stated a range of parameters; including side-tone tolerance and echo return loss performance. With the resultant advent of new recommendations with much tighter requirements for handsets, there is a call for greatly improved designs to be implemented. This, complemented by ongoing improvements in network technology and echo cancellation techniques, will bring digital wireless telephony much closer to matching wire line quality.

3.9 Echo Cancellation System For Radio Telephony

The customer has developed technology for major manufacturers in the telecommunications industry. This has included a host of popular electronic devices, from cellular and cordless telephones, to computers and digital television products.

Microsystems Engineering has worked extensively with the customer over a number of years developing echo cancellation systems for radiotelephony projects. The project described here is an example of a recent echo cancellation system designed and implemented by Microsystems Engineering. Many radiotelephony devices require additional echo control to be incorporated into the system. The main reason for this is the group delay usually imposed by the radio link protocol. Sources of echo that would not normally be noticeable to the user become annoying due to the 10 - 20ms round trip delay that often exists between the portable part and the fixed part of the system.

The diagram below illustrates the elements of a typical radiotelephony system that relate to echo and its control.

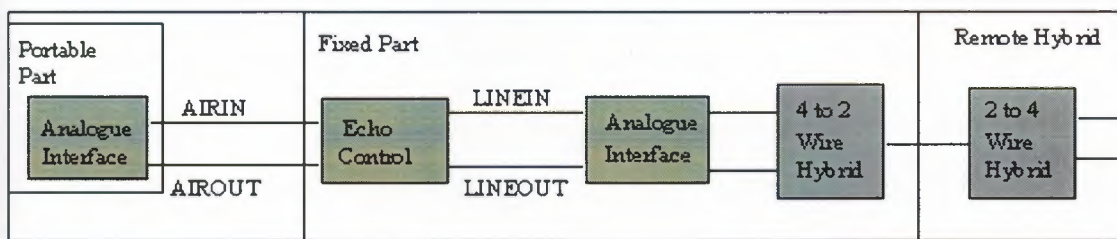


Figure 3.6 typical radio telephony system

The echo control part of the system would generally reside at the base station (in the Fixed Part of the above diagram) and would be responsible for three kinds of echo, [8]:

- Coupling at the portable part resulting in echo of the AIROUT signals back into the AIRIN signal. This is generally of the order of 20-21ms. For the PSTN equipment to suppress this an artificial echo signal needs to be added by the system.
- Reflection at the fixed part 4-wire to 2-wire hybrid resulting in short delay echo of LINEOUT signals in the LINEIN signal (0- 4ms). This is cancelled using an adaptive FIR algorithm.
- Reflection at the exchange 2-wire to 4-wire hybrid resulting in long delay echo of LINEOUT signals in the LINEIN signal. This can be between 0 and 70ms. This is reduced to acceptable levels using a soft suppressor algorithm.



4. THE FUNDAMENTAL PROBLEMS AND SOLUTIONS TO ECHO CANCELLATION

4.1 Overview

Communication applications are discussed. The applications that yield line echo are the long-distance calls between ordinary fixed telephones and the digital data transmission on subscribers' loop. The application of calls between a cellular to a fixed telephone can produce either line echo or both line and acoustic echoes depending on whether the hands-free operation on the cellular is used. Tele-conferencing/ videoconferencing application causes acoustic feedback coupling between the loudspeaker and microphone, and thus creates the acoustic echo. To remove the line and acoustic echoes successfully requires the use of adaptive echo cancellers. These devices have better performance than the non-adaptive echo suppressors. There are several problems associated with the design of effective echo cancellers, i.e. divergence due to double-talking or silent far-end signal and residual echoes. Most existing echo cancellers are designed with adaptive transversal finite impulse response (FIR) digital filters, and based on variations of the least mean square (LMS) and least square (LS) algorithms. Therefore, the concept of conventional LMS and LS algorithms for the use in echo cancellers are discussed. Other methods are also recommended that can overcome the inherent problems of slow convergence in the LMS algorithm and high computation in the LS algorithm.

4.2 Introduction

Echo is a phenomenon in which a delayed and distorted version of an original sound or electrical signal is reflected back to the source. There are two types of echo, namely line and acoustic echoes. Telephone line echo the author is with the Signal Processing Group, Dept. of Applied Electronics, Chalmers University of Technology, and Gothenburg, Sweden. Results from impedance mismatch at the telephone ex-change hybrids where the subscriber's two-wire line is connected to a four-wire line. If the communication is just between two handsets, then only line echo will occur. However, if the telephone connection is between one or more hands-free telephones, a feedback path is set up between the loudspeaker and microphone at each end. This acoustic coupling is due to the reflection of loudspeaker's sound from walls, floor, ceiling, windows and other objects back to the microphone 1. Adaptive cancellation of this acoustic echo has become very important in hands-free telephony or teleconference communication system. The effects of an echo depend on the time delay between the incident and the reflected waves, the strength of the reflected waves and the number of paths through which the waves are reflected. If the time delay is not long, the acoustic echo can be perceived as soft reverberation, which adds artistic quality for example in a concert hall. However, echo arriving a few tens of milliseconds or more after the direct sound will be highly undesirable because long delayed echo is irritating. Likewise in line echo, the short delayed echo cannot be distinguished from the normal side tone of the telephone, which is intentionally inserted to make the telephone communication channel sound "alive", and a round trip delay of more than 40msec will cause significant disturbances to the talker. Such a long delay is caused by the propagation time over long distances and/or the digital encoding of the transmitted signals. In digital cellular systems, the one-way transmission delay is about 100ms when blocks of speech samples are transmitted in wireless, and this

delay is caused by the speech and channel coding methods used in radio communication. It is worse in geostationary satellite links, which have a round trip delay of about 520msec. The *International Telecommunication, Union-Telecommunication, Standardization Section* (ITU-T) G.131 recommends the use of echo cancellers for calls with round-trip delay that is above 50msec. In digital cellular communications, these devices are normally located at the *mobile switching center* (MSC), while in long distance telephony path; they are usually located in an *international switching center* (ISC). This report describes the echo phenomena and the general methods of removing the echo in a long-distance international call between ordinary fixed telephones, in a full-duplex data transmission between voice-band modems, in a national call between a fixed telephone and a cellular telephone and in a teleconference/videoconference communication system.

4.3 Long-Distance International Calls Between Ordinary Fixed Telephones

A simplified long-distance telephone connection is shown in Fig.4.1. This connection contains two-wire sections on the ends (the subscriber loops and possibly some portion of the local network), and a four-wire section in the center (carrier systems for medium to long-haul transmission).

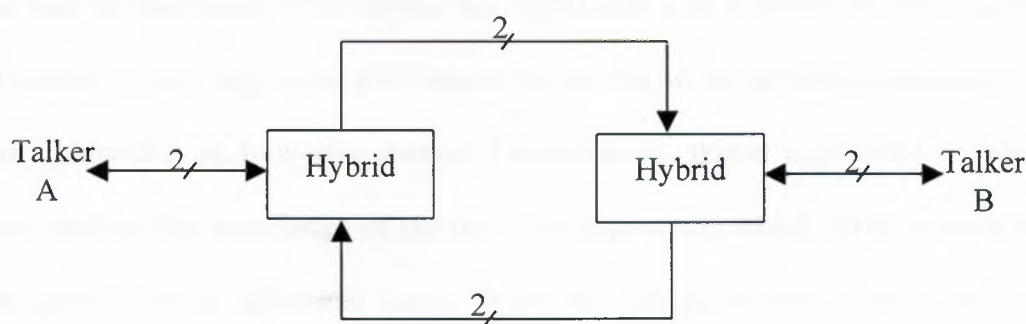


Figure 4.1 A long-distance connection.

Every telephone in a given geographical area is connected to the local exchange by a two-wire line, called the subscriber's loop, which carries connection for both directions of transmission. A local call is established simply by connecting the two subscribers' loops at the local exchange. However, repeaters are used to amplify the speech signal when the distance between the two telephones exceeds 50km. Thus, a four-wire line is required which segregates the two directions of transmission on two different transmission paths. A hybrid is used to convert from the two-wire to four-wire line and vice versa as shown in Fig.4.2 and it is basically a bridge network [3].

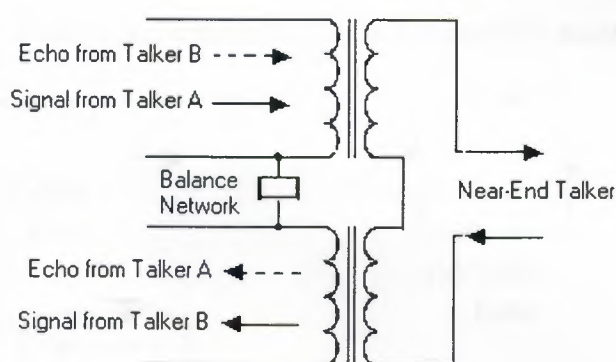


Figure 4.2. Two-wire to four-wire hybrid.

An echo can be decreased if the hybrid has significant loss between its two four-wire ports. To achieve this large loss will require the hybrid to be perfectly balanced by an impedance located at its four-wire portion. Unfortunately, this is impossible in practice because it requires the knowledge of the two-wire impedance, which varies considerably over the population of subscriber loops. When the bridge is not perfectly balanced, impedance mismatch occurs and this causes some of the talker's signal energy to be reflected back as echo. The crucial talker path, as shown in Fig. 4.3a, requires that the hybrid does not have a lot of attenuation between its two-wire and either four-wire port. There are two types of echo as shown in Fig. 4.3b and 4.3c. Talker echo results in the

talker hearing a delayed version of his or her own speech, while in listener echo it is the listener who hears a delayed version of the talker's speech. When there is insignificant transmission delay, this phenomena presents no problem, and, in any case, the talker or listener already heard the "side tone" of his or her own speech via the telephone instrument. The effects of echo can be controlled by adding an insertion loss to the four-wire portions of the connection, since the echo signals experience this loss two or three times (for talker and listener echo respectively) while the talker speech suffers this loss only once. However, on long connections, this loss can become very significant and as a result it is not an optimum solution and other echo control techniques must be used.

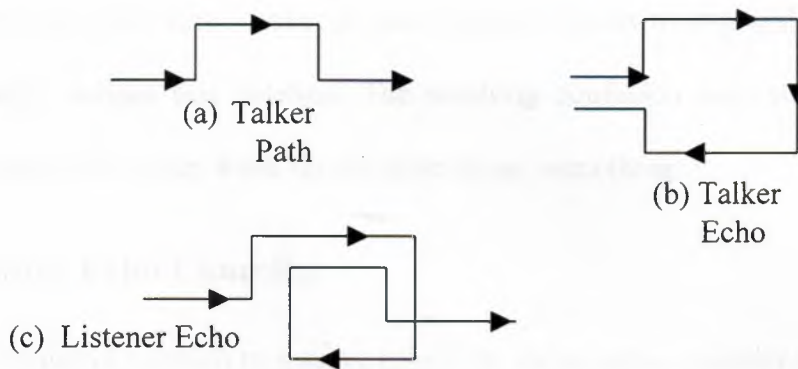


Figure 4.3. Sources of echo in the telephone network

3.1 Echo Suppressor

Echo suppressors have been used since the introduction of long-distance communication.

This device takes advantage of the fact that people seldom talk simultaneously. It is also helped by the fact that during such double-talking, poor transmission quality is less noticeable. The echo suppressor dynamically controls the connection based on who is talking, which is decided by the speech and double talking detector. Double-talking is detected if the level of signal in path L1 is significantly lower than that in path L2. When

the far-end talker A is speaking, the path used to transmit the near-end speech is opened so that the echo is prevented. Then, when talker B speaks or in double talking case, the same switch is closed and a symmetric one at the far-end speech path is opened. However, echo suppressors can clip speech sounds and introduce impairing interruptions. For example, if the near-end talker is initially listening to the far-end but suddenly wants to talk, it is quite likely that the switch preventing his or her speech from being transmitted will not close quickly enough, and the far-end talker may not receive the entire message. This distortion is more pronounced in long transmission with a round-trip delay of more than 200ms. Due to the long delay, a quick response by talker B may cause suppression of something said by talker A at a later time. Talker B, encouraging him/her to stop and wait for talker A to get through, notices this deletion. The resulting confusion may stop the conversation entirely while each party waits for the other to say something.

4.3.2 Adaptive Echo Canceller

An alternative solution to remove echo is to use an echo canceller as shown in Fig.4.4. The echo canceller mimics the transfer function of the echo path to synthesize a replica of the echo, and then subtracts that replica from passes talker A's signal and blocks his/her echo with the open switch. The combined echo and near-end speech signal to obtain the near-end speech signal alone. However, the transfer function is unknown in practice and so it must be identified. The solution to this problem is to use an adaptive filter that gradually matches its impulse response to the impulse response of the actual echo path, as shown in Fig.4.4. The echo path is highly variable, depending on the distance to the hybrid, the characteristics of the two-wire circuit, etc. These variations are taken care of by the adaptive control loop built into the canceller. The canceller in Fig.(4.5,4.6) is for one direction of transmission only (from talker A to talker B).

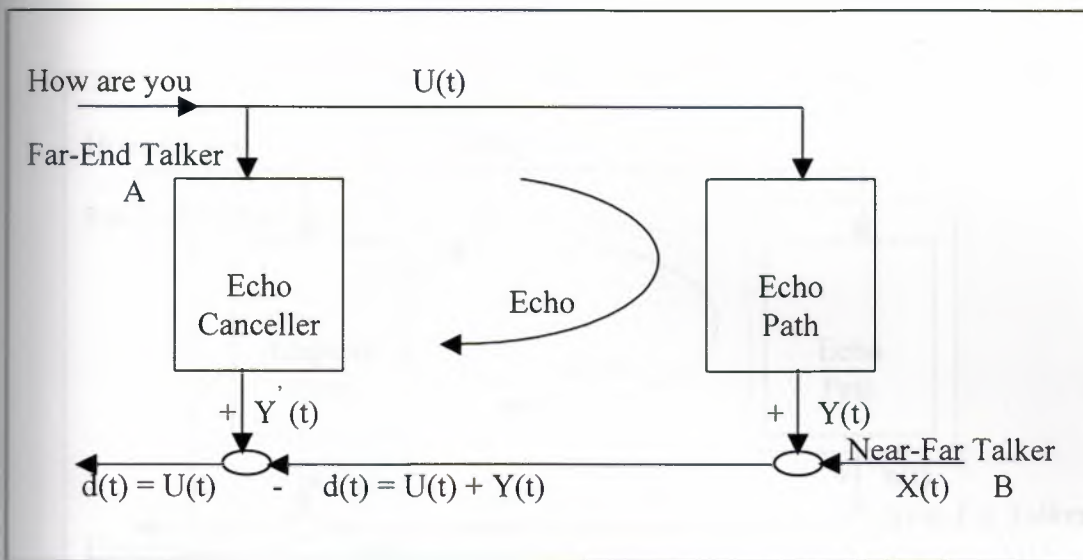


Figure 4.4. Principle of a non-adaptive echo canceller

The adaptive canceller in Fig. 4.6 is placed at the four-wire path near the origin of the echo. The synthetic echo, $r(n)$ is generated by passing the reference input signal, $y(n)$ from the far-end talker through the adaptive filter that will ideally match the transfer function of the echo path. The echo signal, $r(n)$ is produced when $y(n)$ passes through the hybrid. The signal, $r(n)$ plus the near-end talker signal, $x(n)$ constitute the “desired” response for the adaptive canceller. The two signals, $y(n)$ and $r(n)$ are correlated because the later is obtained by passing $y(n)$ through the echo channel. The synthetic echo $r(n)$ is subtracted from the desired response $r(n) + x(n)$ to yield the canceller error signal,

$$e(n) = r(n) - r(n) + x(n) \quad (4.1)$$

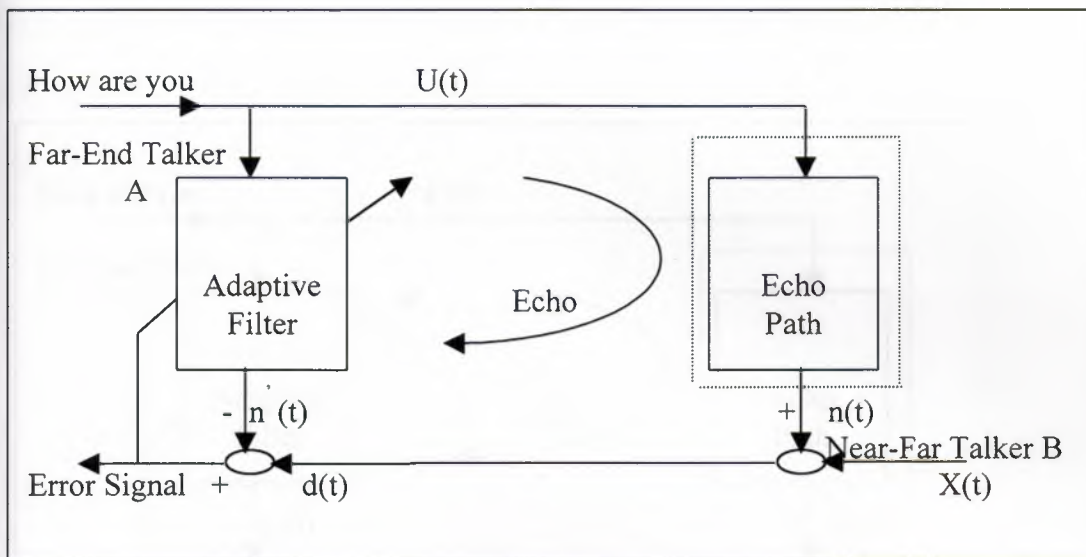


Figure 4.5. General configuration of an adaptive echo canceller

Similar to the echo suppressors, adaptive echo cancellers also face the problem of double-talking. The situation that must be avoided is interpreting $x(n)$ as part of the true error signal which results in making large corrections to the estimated echo path in a doomed-to-failure attempt to cancel it. In order to avoid this possibility, the tap weights must not be up-dated as soon as double-talking is detected as shown in Fig. 4.6. The design of a good double-talking detector is difficult. Even with the assumption of a fast-acting detector, there is still a possibility of changes occurring in the echo channel during the time that the canceller is frozen, which leads to increased unconcealed echo. But, fortunately the duration of double-talking is usually short. In the system, as shown in Fig. 4.6, the effect of the speech/echo misclassification is that the echo is sub optimally cancelled. This is more acceptable than in the case of echo suppressor, which removes part of the speech signal during misclassification.

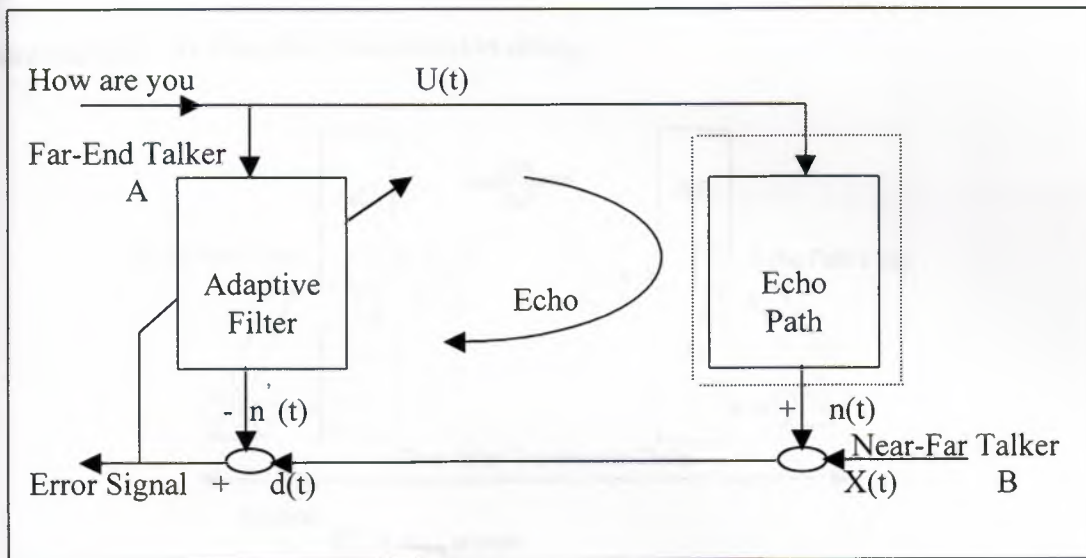


Figure 4.6. The double-talking detector stops adjustment when the near-end talker is active

To add to the problems of effective echo canceller design, it sometimes occur that no far-end signal is present and even an echo canceller which is working well will leave some residual unconcealed echo. In the former case, the adaptation is generally halted once the signal is estimated to be insignificant and in the latter case, a non-linear processor is used to remove the residual echo [8]. The presences of residual echo or the limitations on the achievable cancellation ratio are imposed by the presence of additive noise, nonlinear distortion, echo dispersion beyond the length of the transversal filter and digital resolution constraints. The working mechanism of the non-linear processor is to block this small-unwanted signal if the signal magnitude is lower than a certain (small) threshold value during single talking. However, the non-linear processor will only distort and not block the near-end signal during double-talking. The distortion is generally unnoticeable and so

the processor does not have to be removed during double-talking. In practice, it is desirable to cancel the echoes in both directions of the trunk. Therefore, two adaptive echo cancellers are used as shown in Fig.4.7. One of the cancellers removes the echo from each end of the connection. The near-end talker for one canceller is the far-end talker of the other. The requirements of an echo canceller are influenced by the following transmission characteristics: in *One-Way transmission delay*:

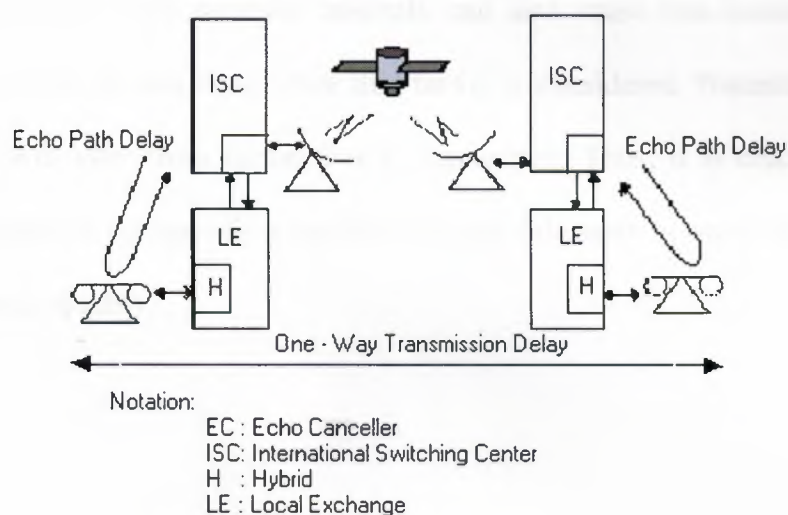


Figure 4.7. Location of echo canceller

The required total echoes return loss (TERL) for a connection is determined by the one-way transmission delay (depicted in Fig.4.7). This loss is defined as the total level loss between the talker's mouth and his ear. ITU-T Recommendation G.131 sets the minimum value of TRL at the range of 46-54dB. *Echo path delay*: The number of coefficients needed in the adaptive filter depends on the echo path delay and the length of the impulse response of the hybrid, which both are relatively short. The echo delay is defined as two times the delay from the canceller to the hybrid as shown in Fig.4.7. For the adaptive echo canceller to operate properly, the impulse response of the adaptive filter should have a length greater than the combined effect of the hybrid's impulse response length and echo

path delay, [2]. Let T_s be the sampling period of the digitized speech signal, M be the number of adjustable coefficients in an adaptive finite impulse response filter, and τ be the combined effect to be accommodated. Therefore, $M T_s > \tau$. The value of T_s is 125 μ s in the telephone network, and if $T_s = 30$ ms, then we must choose $M > 240$ taps for a satisfactory performance. *Type of transmission and end-user equipment*: Non-linearities in the echo path will affect the performance of the echo canceller. Devices such as bit-rate coders and end-user equipment with acoustic crosstalk can also cause non-linearities. The TERL requirement must be met even when this factor is considered. Naturally, the degree of impairment will vary from connection to connection. Thus, it is crucial that the echo canceller adapts to the specific situation on a per call basis in order to achieve the best possible speech quality.

5. THE LMS ALGORITHM

Overview, Derivation

In this chapter we introduce the Least-Mean-Square (LMS) Algorithm. The LMS algorithm is important because of its simplicity and ease of computation, and because it does not require off-line gradient estimations or repetitions of data. If the adaptive system is an adaptive linear combiner, and if the input vector X_k and the desired response d_k are available at each iteration, the LMS algorithm is generally the best choice for many different applications of adaptive signal processing.

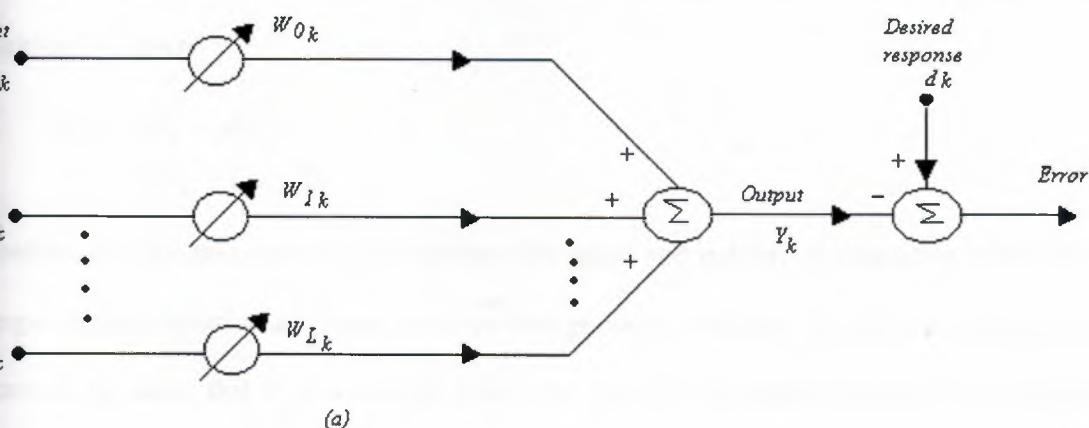
We recall that the adaptive linear combiner was applied in two basic ways, depending on whether the input is available in parallel (multiple inputs) or series (single input) form. These two ways are shown in figure 5.1.

In both cases we have the combiner output, y_k , as a linear combination of the input samples.

We have

$$\varepsilon_k = d_k - X_k^T W_k \quad (5.1)$$

where X_k is the vector of the input samples in either of the two configurations in figure 5.1.



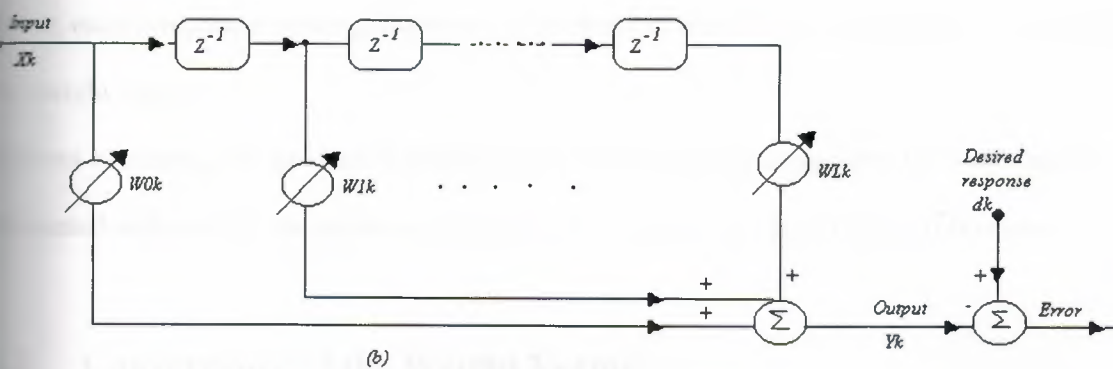


Figure 5.1 the adaptive Linear Combiner: (a) in general form: (b) as a transversal.

To develop an adaptive algorithm using the previous methods, we would estimate the gradient of $\xi = E[\varepsilon_k^2]$ by taking differences between short-term averages of ε_k^2 . Instead, to develop the LMS algorithm, we take ε_k^2 itself as an estimate of ξ_k .

Then, at each iteration in the adaptive process, we have a gradient estimate of the form

$$\hat{\nabla}_k = \begin{bmatrix} \frac{\partial \varepsilon_k^2}{\partial W_0} \\ \vdots \\ \frac{\partial \varepsilon_k^2}{\partial W_L} \end{bmatrix} = 2\varepsilon_k \begin{bmatrix} \frac{\partial \varepsilon_k}{\partial W_0} \\ \vdots \\ \frac{\partial \varepsilon_k}{\partial W_L} \end{bmatrix} = -2\varepsilon_k X_k \quad (5.2)$$

The derivatives of ξ_k with respect to the weights follow directly from (5.1).

With simple estimate of the gradient, we can now specify a steepest-descent type of adaptive algorithm. We have

$$\begin{aligned} W_{k+1} &= W_k - \mu \hat{\nabla}_k \\ &= W_k + 2\mu \varepsilon_k X_k \end{aligned} \quad (5.3)$$

As before, μ is the gain constant that regulates the speed and stability of adaptation. Since the weight changes at each iteration are based on imperfect gradient estimates, we would expect the adaptive process to be noisy, that is, it would not follow the true line of steepest descent on the performance surface, [6].

From its form in (5.3), we can see that the LMS algorithm can be implemented in a practical system without squaring, averaging, or differentiation and is elegant in its simplicity and efficiency. As noted

above, each component of the gradient vector is obtained from a single data sample without perturbing the weight vector.

Without averaging, the gradient components do contain a large component of noise, but the noise is attenuated with time by the adaptive process, which acts as a low-pass filter in this respect.

5.2 Convergence of the Weight Vector

As with all adaptive algorithms, a primary concern with the LMS algorithm is its convergence to the optimum weight vector solution, where $E[\varepsilon_k^2]$ is minimized. To examine LMS convergence, we first note that the gradient estimate in (5.2) can readily be shown to be unbiased when the weight vector is held constant. The expected value of (5.2) with W_k held equal to W is

$$\begin{aligned} E[\hat{\nabla}_k] &= -2E[\varepsilon_k X_k] \\ &= -2E[d_k X_k - X_k X_k^T W] \\ &= 2(RW - P) = \nabla \end{aligned} \quad (5.4)$$

The second line of (5.4) follows from (5.1) plus the fact that ε_k is a scalar and can thus be commuted. Since the mean value of $\hat{\nabla}_k$ is equal to the true gradient ∇ , $\hat{\nabla}_k$ must be an unbiased estimate.

Seeing that the gradient estimate is unbiased, we could make the LMS algorithm into a true steepest-descent algorithm, at least in limiting case, by estimating ∇ at each step as in (5.2) but not adapting the weights until many steps have occurred, in this way $\hat{\nabla}_k$ could be made to approach ∇_k . With the weight vector changing at each iteration, we need to examine the weight vector convergence in a different manner, as follows.

From (5.3) we can see that the weight vector W_k is a function only of the past input vectors $X_{k-1}, X_{k-2}, \dots, X_0$. If we assume that successive input vectors are independent over time W_k is independent of X_k . For stationary input processes meeting this condition, the

expected value of the weight vector $E[W_k]$ after a sufficient number of iterations can be shown as follow to converge to the Wiener optimal solution, that is, to $W^* = R^{-1}P$

Taking the expected value of both sides of (5.3) yields the difference equation

$$\begin{aligned} E[W_{k+1}] &= E[W_k] + 2\mu E[\varepsilon_k X_k] \\ &= E[W_k] + 2\mu(E[d_k X_k] - E[X_k X_k^T W_k]) \end{aligned} \quad (5.5)$$

Using the foregoing assumption that X_k and W_k are independent, we have the expected products as in (5.5). Also, we have the optimum weight vector given as $W^* = R^{-1}P$. Thus (5.5) becomes

$$\begin{aligned} E[W_{k+1}] &= E[W_k] + 2\mu(P - RE[W_k]) \\ &= (I - 2\mu R)E[W_k] + 2\mu RW^* \end{aligned} \quad (5.6)$$

Using expected values, the solution is

$$E[V'_k] = (I - 2\mu R\Lambda)^k V'_0 \quad (5.7)$$

Where V' is the weight vector, W , in the principal-axis system Λ is the diagonal eigenvalue matrix of R , and V'_0 the initial weight vector in the principal-axis system.

Thus, as k increases without bound, we see that expected weight vector in (5.7) reaches the optimum solution (i.e., zero in the principal-axis system) only if the right side of the optimum convergence to zero.

$$\frac{1}{\lambda_{\max}} > \mu > 0 \quad (5.8)$$

Where λ_{\max} is the largest eigenvalue, that is, the largest diagonal element in Λ . So in (5.8), we have bounds on μ for convergence of the weight vector mean to the optimum weight vector. Within these bounds, the speed of adaptation and also the noise in the weight vector solution are determined by the size of μ . We also note that λ_{\max} cannot be greater than the trace of R , which is the sum of the diagonal elements of R , that is,

$$\lambda_{\max} \leq \text{tr}[\Lambda] = \sum (\text{diagonal.elements.of } \Lambda)$$

$$= \sum (\text{diagonal.elements.of } R) = \text{tr}[R] \quad (5.9)$$

Furthermore, with a transversal adaptive filter gives $\text{tr}[R]$ as just $(L+1)E[x_k^2]$ or $L+1$ times the input signal power. Thus convergence of the weight vector mean is assured by:

In general: $0 < \mu < 1/\text{tr}[R]$

Transversal filter $0 < \mu < 1/(l+1)$ (5.10)

But is much easier to apply, because the elements R and the signal power can generally be estimated more easily eigenvalues of R .

The assumption of deceleration and stationary of input vector used to drive the result in this section are not necessary condition for convergence of the LMS algorithm but have been adopted in this chapter for analytic convergence.

Convergence with certain correlated and non-stationary inputs is demonstrated in the literature on the LMS algorithm [4].

Under these conditions the analysis becomes much more complex. We know of no unconditional proof of convergence of the LMS algorithm.

5.3 Noise in The Weight-Vector Solution

With the LMS algorithm, the gradient estimate as given by (5.2) is not based on weight perturbation, so we must reexamine its variance.

Let us define N_k as a vector of noise in the gradient estimate at the k th iteration. Thus

$$\hat{\nabla}_k = \nabla_k + N_k \quad (5.11)$$

If we assume that the LMS process, using a small value of the adaptive gain constant μ , has converged to a steady-state weight vector solution near W^* , then ∇_k in (5.11) will be close to zero. Then, in accordance with (5.2), the gradient noise is close to

$$N_k = \hat{\nabla}_k = -2\varepsilon_k X_k \quad (5.12)$$

The covariance of the noise is thus given by

$$\text{cov}[N_k] = E[N_k N_k^T] = 4E[\varepsilon_k^2 X_k X_k^T] \quad (5.13)$$

If we assume that the weight vector, W_k , remains near its optimum, W^* , we conclude that ε_k^2 is approximately uncorrelated with the signal vector, so that (5.13) becomes

$$\begin{aligned} \text{cov}[N_k] &\approx 4E[\varepsilon_k^2]E[X_k X_k^T] \\ &\approx 4\xi_{\min} R \end{aligned} \quad (5.14)$$

We need to transform (5.14) into the principal-axis coordinate system, as follows:

$$\begin{aligned} \text{cov}[N'_k] &= \text{cov}[Q^{-1}N_k] \\ &= E[Q^{-1}N_k(Q^{-1}N_k)^T] \\ &= Q^{-1}E[N_k N_k^T]Q \\ &= Q^{-1}\text{cov}[N_k]Q \approx 4\xi_{\min}\Lambda \end{aligned} \quad (5.15)$$

The weight vector covariance in the principal-axis coordinate system. The result is

$$\begin{aligned} \text{cov}[V'_k] &= \frac{\mu}{4}(\Lambda - \mu\Lambda^2)^{-1}\text{cov}[N'_k] \\ &\approx \mu\xi_{\min}(\Lambda - \mu\Lambda^2)^{-1}\Lambda \end{aligned} \quad (5.16)$$

In practical situations the elements of $\mu\Lambda$ tend to be considerably less than 1, so we simplify the expression in (5.16) by neglecting the term $\mu\Lambda^2$ to obtain

$$\begin{aligned}\text{cov}[V'_k] &\approx \mu\xi_{\min}\Lambda^{-1}\Lambda \\ &\approx \mu\xi_{\min}I\end{aligned}\tag{5.17}$$

Thus, transforming back to unprimed coordinates, we have the steady-state noise in the weight vector solution given approximately by

$$\begin{aligned}\text{cov}[V_k] &= Q\text{cov}[V'_k]Q^{-1} \\ &\approx \mu\xi_{\min}QIQ^{-1} \\ &\approx \mu\xi_{\min}I\end{aligned}\tag{5.18}$$

5.4 The basic LMS adaptive algorithm

One of the most successful adaptive algorithms is the LMS algorithm developed by Widrow and his coworkers (Widrow et al., 1975a). Instead of computing W_{OPT} in one go as suggested by equation 5.18, in the LMS the coefficients are adjusted from sample to sample in such a way as to minimize the MSE.

The LMS is based on the steepest descent algorithm where the weight vector is updated from sample to sample as follows:

$$W_{k+1} = W_k - \mu\nabla_k\tag{5.19}$$

Where W_k and ∇_k are the weight and the true gradient vectors, respectively, at the k th sampling instant. μ Controls the stability and rate of convergence.

The steepest descent algorithm in Equation 5.18 still requires knowledge of R and P , since ∇_k is obtained by evaluating Equation 5.16. The LMS algorithm is a practical method of obtaining estimates of the filter weights W_k in real time without the matrix inversion in Equation 5.17 or the direct computation of the auto correlation and cross-correlation. The Widrow-Hopf LMS algorithm for updating the weights from sample to sample is given by

$$W_{k+1} = W_k + 2\mu e_k X_k \quad (5.20a)$$

Where:

$$e_k = y_k - W_k^T X_k \quad (5.20b)$$

Clearly, the LMS algorithm above does not require prior knowledge of the signal statistics (that is the correlation's R and P), but instead uses their instantaneous estimates. The weights obtained by the LMS algorithm are only estimates, but these estimates improve gradually with time as the weights are adjusted and the filter learns the characteristics of the signals. Eventually, the weights converge. The condition for convergence is

$$0 < \mu < \frac{1}{\lambda_{\max}} \quad (5.21)$$

Where λ_{\max} is the maximum eigenvalue of the input data covariance matrix. In practice, W_k never reaches the theoretical optimum (the Wiener solution), but fluctuates about it see figure (5.2).

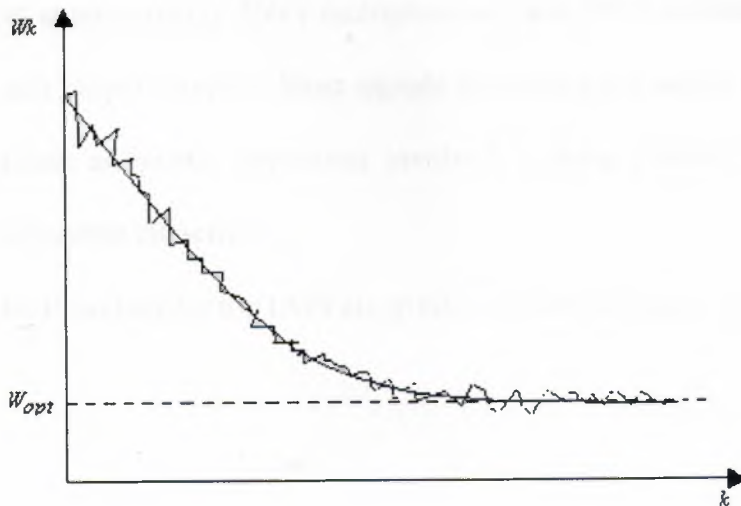


Figure 5.2 An illustration of the variations in the filter weights.

5 Implementation of the basic LMS algorithm

The computational procedure for the LMS algorithm is summarized below.

- (1) Initially, set each weight $w_k(i)$, $i = 0, 1, \dots, N-1$, to an arbitrary fixed value, such as 0.

For each subsequent sampling instants, $k=1, 2, 3, \dots$, carry out steps (2) to (4) below:

- (2) Compute filter output.

$$\hat{n}_k = \sum_{i=0}^{N-1} w_k(i) x_{k-i}$$

- (3) compute the error estimate

$$e_k = y_k - \hat{n}_k$$

- (4) update the next filter weights

$$w_{k+1}(i) = w_k(i) + 2\mu e_k x_{k-i}$$

The simplicity of the LMS algorithm and ease of implementation, evident from above, make it the algorithm of first choice in many real-time systems. The LMS algorithm requires approximately $2N+1$ multiplications and $2N+1$ additions for each new set of input and output samples. Most signals processors are suited to the mainly multiply-accumulate arithmetic operations involved, making a direct implementation of the LMS algorithm attractive.

The flowchart for the LMS algorithm is given in Figure 5.3 figure 5.4

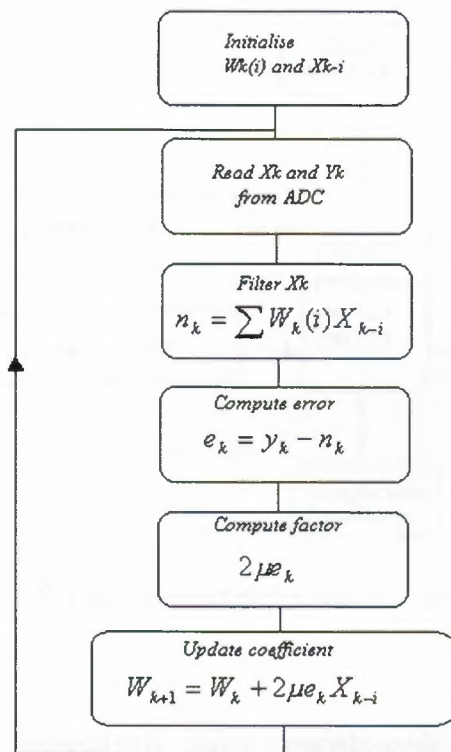


Figure 5.3 Flowchart for the LMS adaptive filter.

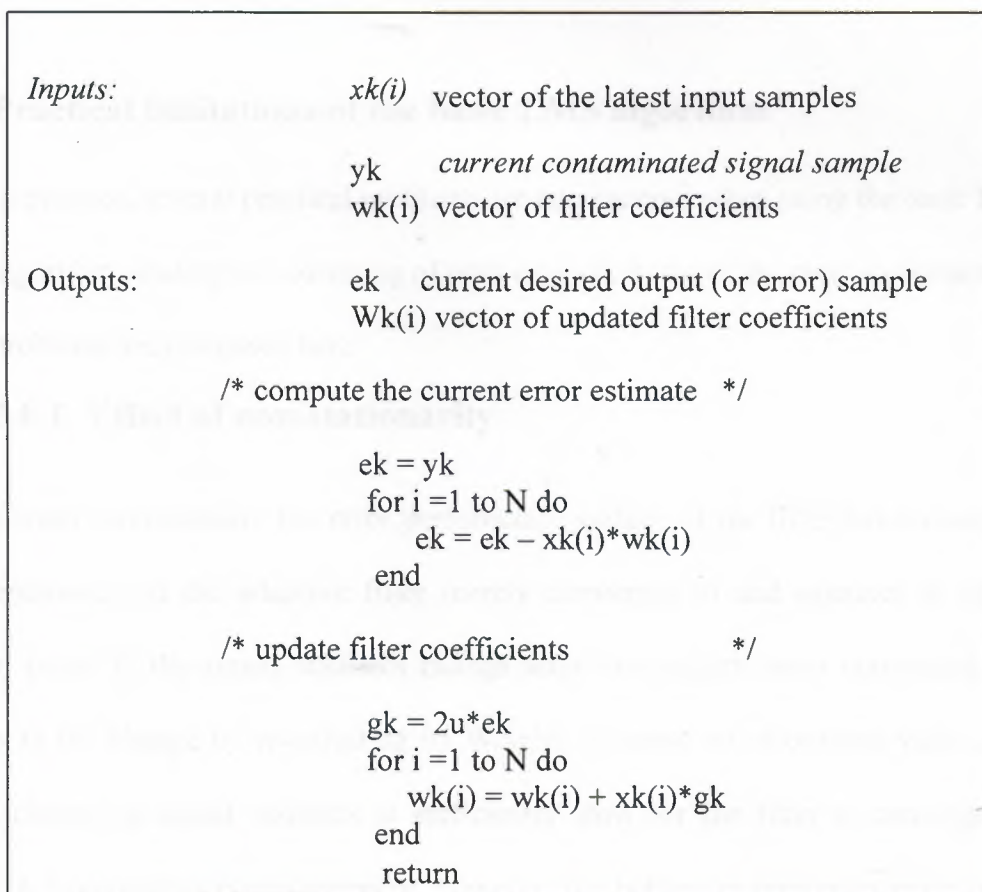


Figure 5.4 Coding of the LMS adaptive filter.

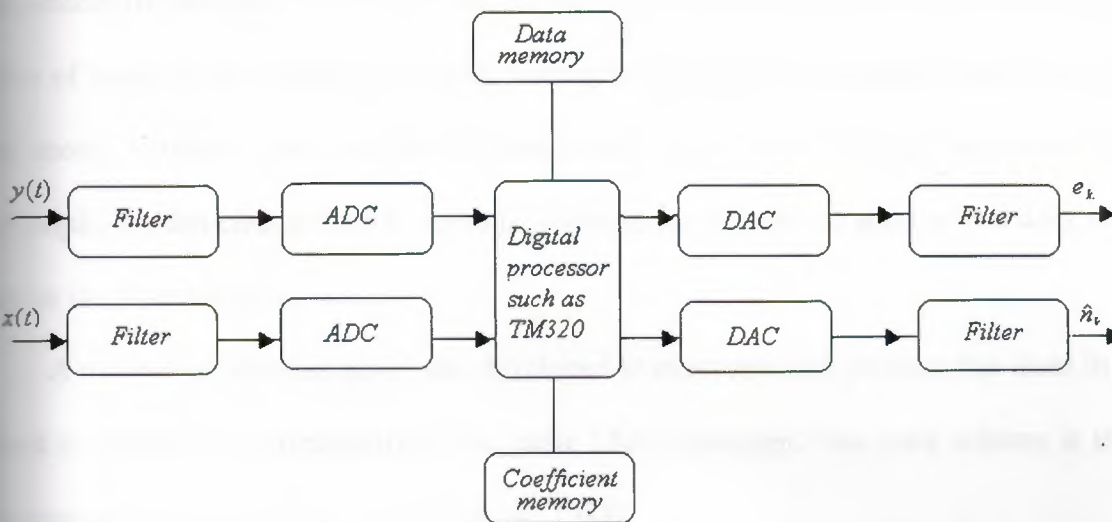


Figure 5.5 Hardware implementation for real-time LMS adaptive filtering.

And 5.5, respectively, show a pseudo-code for the software and hardware implementations.

5.6 Practical limitations of the basic LMS algorithm

In practice, several practical problems are encountered when using the basic LMS algorithm, leading to a lowering of performance. Some of the more important problems are discussed here.

5.6.1 Effect of non-stationarity

In a stationary environment, the error performance surface of the filter has a constant shape and orientation, and the adaptive filter merely converges to and operates at or near the optimum point. If the signal statistics change after the weights have converged, the filter responds to the change by re-adjusting its Weights to a new set of optimal values, provided that the change in signal statistics is sufficiently slow for the filter to converge between change. In a nonstationary environment, however, the bottom or minimum point continually moves, and its orientation and curvature may also be changing (see Figure 5.6) thus the

algorithm in this case has the task not only of seeking the minimum point of the surface but also of tracking the changing position, leading to significant lowering of performance. (Such as mean, variance, autocorrelation) change with time. Such change can result from, for example, sudden changes due to sporadic interference of short duration or bad data, and often upset the filter weights).

A number of schemes have been developed to overcome this problem but these in general tend to increase the complexity of the basic LMS algorithm. One such scheme is the time-sequenced adaptive (Ferrari and Windrow, 1981).

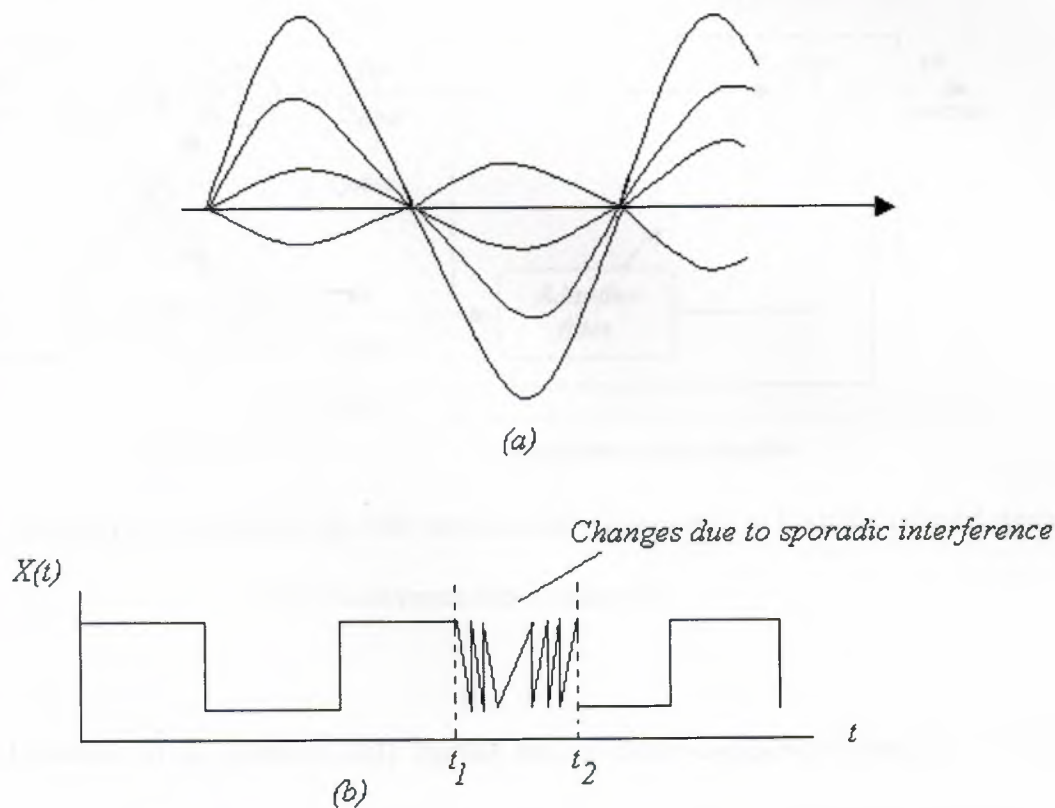


Figure 5.6 An illustration of nonstationary processes

(a) modulated waveform; (b) sporadic interference.

5.6.2 Effects of signals component on the interference input channel

The performance of the algorithm relies on the measured interference signal, $X_k(i)$ being highly correlated with the actual interference, but weakly correlated (theoretically zero) with the desired signal. In most cases, this condition is not met. In some applications, the contaminating input may contain both the undesired interference as well as low-level signal components. Such a situation is illustrated in Figure (5.7). It is shown in Windrow et al. (1975a) that the adaptive noise canceling process still leads to a significant improvement in the desired signal-to-noise ratio in these cases but only at the expense of a small signal

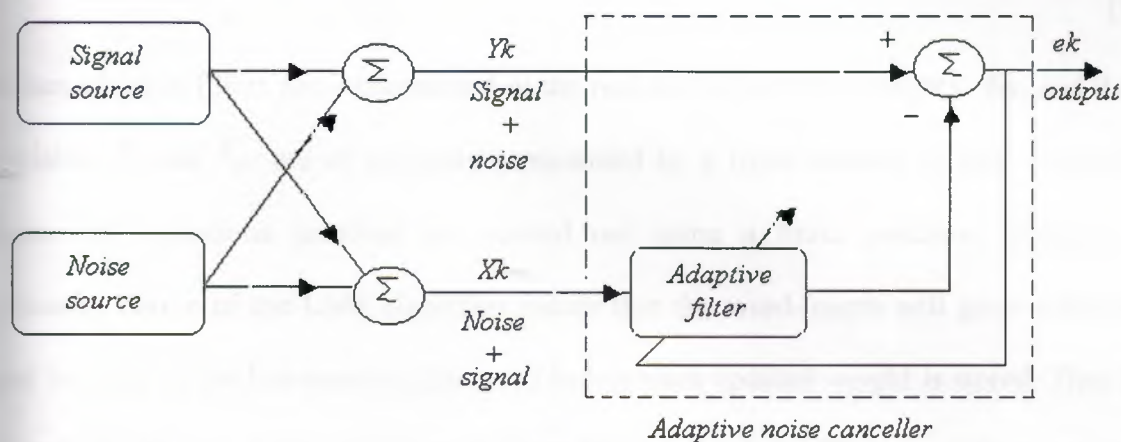


Figure 5.7 Adaptive noise canceling with some signal components in both the desired signal and interference input channels.

Distortion. However, if x_k contains only signals and no noise component what so ever, the desired signal in Y_k may be completely obliterated. Our work in biomedical signal processing confirms their results (Ifeachor et al. 1986).

5.6.3 Computer wordlength requirements

The LMS-based FIR adaptive filter is characterized by the following equations:

For the digital filter,

$$n = \sum_{i=0}^{N-1} w_k(i) x_{k-1} \quad (5.22a)$$

For the adaptive algorithm,

$$W_{k+1} = W_k + 2\mu e_k X_k \quad (5.22b)$$

When adaptive filters are implemented in the real world, the filter weights, W_k , and the input variable, X_k and Y_k , are of necessity represented by a finite number of bits. Similarly, the numerical operations involved are carried out using a finite precision arithmetic. The excessive nature of the LMS algorithm means that the word-length will grow without limit and so some of the bits must be discarded before each updated weight is stored. Thus the Y_k and $W_k(i)$ may differ significantly from their true values. The use of filter weights and results of arithmetic operations with limited accuracy may include (i) possible non-convergence of the adaptive filter whose effects may include (i) possible non-convergence of the adaptive filter to the optimal solution, leading to an inferior performance. For example, if the filter is used as an interference canceller some residual interference may remain, (ii) the filter outputs may contain noise, which will cause it to fluctuate randomly, and (iii) aperture termination of the algorithm may occur. Thus sufficient number of bits should be used to keep these errors at tolerable levels. Most adaptive system described in the open literature represent the digital signals, x_{k-1} and y_k , as fixed point numbers of between 8 and 16 bits, with the coefficients quantized to between 16 and 24 bits. The multipliers used range from 8×8 to 4×16 bits, and accumulators of between 16 and 40 bits are used. It appears that for low order

filters (up to about 100 coefficient) it is sufficient to store the coefficient to no more than 16-bit accuracy and to use a 16*16 bit multiplier with an accumulator of length 32 bits.

5.6.4 Coefficient drift

In the presence of certain types of inputs (for example narrowband signals), the filter coefficient may drift from the optimum values and grow slowly, eventually exceeding the permissible wordlength. This is an inherent problem in the LMS algorithm and leads to a long-term degradation in performance. In practice, introducing a leakage factor, which gently nudges the coefficients towards zero, counteracts coefficient drift. Two such schemes are given in Equations 5.23:

$$w_{k+1}(i) = \delta w_k + 2\mu e_k x_{k-i} \quad (5.23a)$$

$$w_{k+1}(i) = w_k + 2\mu e_k x_{k-i} \pm \delta \quad (5.23b)$$

small δ , the leakage factor, ensures that drift is contained, but introduce bias in the error term, e_k .

The usefulness of the basic LMS algorithm has been extended by more sophisticated LMS-based algorithm as mentioned before. These include

- (1) The complex LMS algorithm which allows the handling of complex data,
- (2) The block LMS algorithm which offers substantial computational advantages in some cases faster convergence, and
- (3) Time-sequenced LMS algorithm to deal with particular types of non-stationary.

5.7 Fast LMS algorithm

A number of blocks LMS algorithms have been proposed which offer substantial computational saving especially when the number of filter coefficients is large. The computational saving result from processing the data in blocks instead of one sample at a time. Frequency domain implementations of the block LMS exploit the computational advantage of the fast Fourier transforms (FFT) in performing convolutions (Mansour and Gray, 1982).

5.8 Recursive least squares algorithm

The RLS algorithm is based on the well known least square method (Figure 5.8). An output signal, Y_k , is measured at the discrete time, k , in response to a set of input signals, $X_k(i)$, $i = 1, 2, 3, \dots, n$. The input and output signals are related by the simple regression model

$$y_k = \sum_{i=0}^{n-1} w(i)x_k(i) + e_k \quad (5.24)$$

Where e_k represents measurements errors or other effects that cannot be accounted for, and $w(i)$ represents the proportion of the i th input that is contained in the primary signal, Y_k . The problem in the LS method is, given the $X_k(i)$ and Y_k above, to obtain estimates of $w(0)$ to $w(n-1)$.

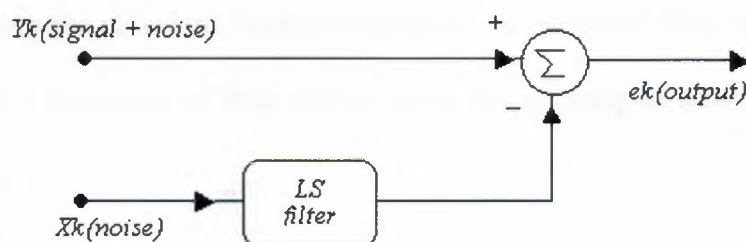


Figure 5.8 An illustration of the basic idea of the least-squares method.

Optimum estimates (in the least square sense) of the filter weights, $w(i)$, are given by

$$W_k = [X_m^T X_m]^{-1} X_m^T Y_m \quad (5.25)$$

Where Y_m , W_m and X_m are given by

$$Y_m = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{m-1} \end{bmatrix} \quad X_m = \begin{bmatrix} x^T(0) \\ x^T(1) \\ x^T(2) \\ \vdots \\ x^T(m-1) \end{bmatrix} \quad W_m = \begin{bmatrix} w(0) \\ w(1) \\ w(2) \\ \vdots \\ w(n-1) \end{bmatrix}$$

$$x^T(k) = [x_k(0) x_k(1) \dots x_k(n-1)], \quad k=0,1,2,3,\dots,m-1$$

The suffix m indicates that each matrix above is obtained using all m data points and T indicates transposition. Equation 5.30 gives the OLS estimates of W_m which can be obtained using any suitable matrix inversion technique. The filter output is then obtained as

$$n_k = \sum_{i=0}^{n-1} w(i) k_{k-1}, \quad k=1,2,3,\dots,m \quad (5.26)$$

The computation of W_m in Equation 5.25 requires the time-consuming computation of the inverse matrix. Clearly, the LS method above is not suitable for real-time or on-line filtering. In practice, when continuous data is being acquired and we wish to improve our estimate of W_m using the new data, recursive methods are preferred. With the recursive least squares algorithm the estimates of W_m and to allow the tracking of slowly varying signal characteristics. Thus

$$W_k = W_{k-1} + G_k e_k \quad (5.27a)$$

$$P_k = \frac{1}{\gamma} [P_{k-1} - G_k X^T(k) P_{k-1}] \quad (5.27b)$$

where

$$G_k = \frac{P_{k-1}x(k)}{\alpha_k}$$

$$e_k = y_k - X^T(k)W_{k-1}$$

$$\alpha_k = \gamma + X^T(k)P_{k-1}X(k)$$

P_k is essentially a recursive way of computing the inverse matrix $[X_k^T X_k]^{-1}$.

The argument k emphasizes the fact that the quantities are obtained at each sample point. γ is referred to as the forgetting factor. This weighting scheme reduces to that of the LS when $\gamma = 1$. Typically, γ is between 0.98 and 1. Smaller values assign too much weight to the more recent data, which leads to wildly fluctuating estimates. The number of previous samples that significantly contribute to the value of W_k at each sample point is called the asymptotic sample length (ASL) given by

$$\sum_{k=1}^{\infty} \gamma^k = \frac{1}{1-\gamma} \quad (5.28)$$

This effectively defines the memory of the RLS filter. When $\gamma=1$, that is when it corresponds to the LS, the filter has an infinite memory.

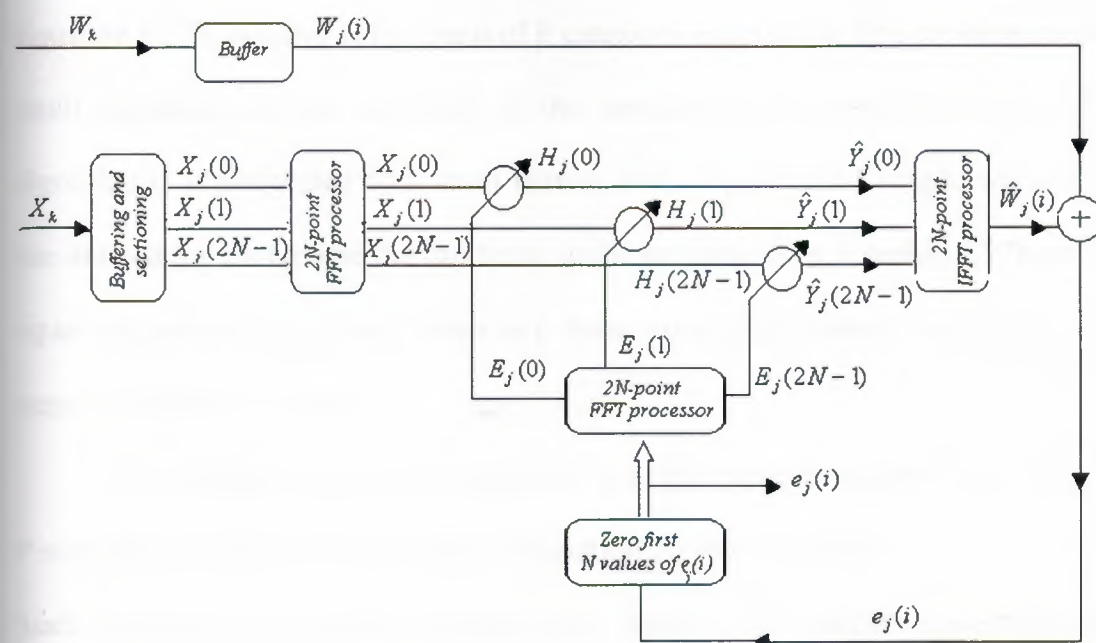


Figure 5.9 Simplified block diagram of a frequency domain LMS filter

5.9 Limitations of the recursive least squares algorithm

The RLS method is very efficient and involves exactly the same number of arithmetic operations between samples as W_k and P_k in Equations 5.27 a and b; have fixed dimensions. This is an important requirement for efficient real-time filtering. There are, however, two main problems that may be encountered when the RLS algorithm is implemented directly. The first, referred to as 'blow-up', results if the signal $X_k(i)$ is zero for a long time, when the matrix P_k will grow exponentially as a result of division by γ (which is less than unity) at each sample point:

$$\lim_{k \rightarrow \infty} P_k = \lim_{k \rightarrow \infty} \left(\frac{P_{k-1}}{\gamma_{k-1}} \right) \quad (5.29)$$

The second problem with the RLS is its sensitivity to computer round off errors, which results in a negative definite P matrix and eventually to instability. For successful estimation of W, it is necessary that the P be positive semi definite which is equivalent to requiring in the LS method that the matrix $X^T X$ be inevitable, but, because of differencing of terms in Equation 5.27b, positive definiteness of P cannot be guaranteed. This problem can be worse in multi parameter models, especially if the variables are linearly dependent and when the algorithm is implemented on a small system with a finite word length, when the algorithm has iterated for a long time the two terms in the parentheses in Equation 5.27b are very nearly equal and subtraction of such terms in a finite word length system may lead to errors and a negative definite P_k matrix.

The problem of numerical instability may be solved by suitably factorizing the matrix P such that the differencing of terms in Equation 5.27b is avoided.

Such factorization algorithms are numerically better conditioned and have accuracies that are comparable with the RLS algorithm that uses double precision. Two such algorithms are the

square root and the UD factorization algorithms. In terms of storage and computation the UD algorithm is more efficient, and is thus preferred. In fact, the UD algorithm is a square-root-free formulation of the square root algorithm and thus shares the same properties as the latter.

6.1 Overview

As in the previous chapters, we will assume that the input signal $x(n)$ is a zero-mean, white Gaussian process with variance σ_x^2 . We will assume that the noise $v(n)$ is also a zero-mean, white Gaussian process with variance σ_v^2 . The filter coefficients are initialized to zero. This



Figure 6.1 Block diagram representation of the LMS algorithm.

Assuming that the input signal $x(n)$ is a zero-mean, white Gaussian process with variance σ_x^2 , and the noise $v(n)$ is also a zero-mean, white Gaussian process with variance σ_v^2 , the LMS algorithm can be represented by the block diagram shown in Figure 6.1.

A block diagram of the LMS algorithm is shown in Figure 6.1. The input signal $x(n)$ is a zero-mean, white Gaussian process with variance σ_x^2 . The noise $v(n)$ is also a zero-mean, white Gaussian process with variance σ_v^2 . The filter coefficients are initialized to zero. This

1. The input signal $x(n)$ is a zero-mean, white Gaussian process with variance σ_x^2 .

2. The noise $v(n)$ is also a zero-mean, white Gaussian process with variance σ_v^2 .

3. The filter coefficients are initialized to zero.

6. FINITE-PRECISION EFFECTS

6.1 Overview

In order to simplify the discussion of finite-precision effects on the performance of the LMS algorithm. We will depart from the practice followed in previous chapters, and assume that the input data and therefore the filter coefficients are all *real valued*. This

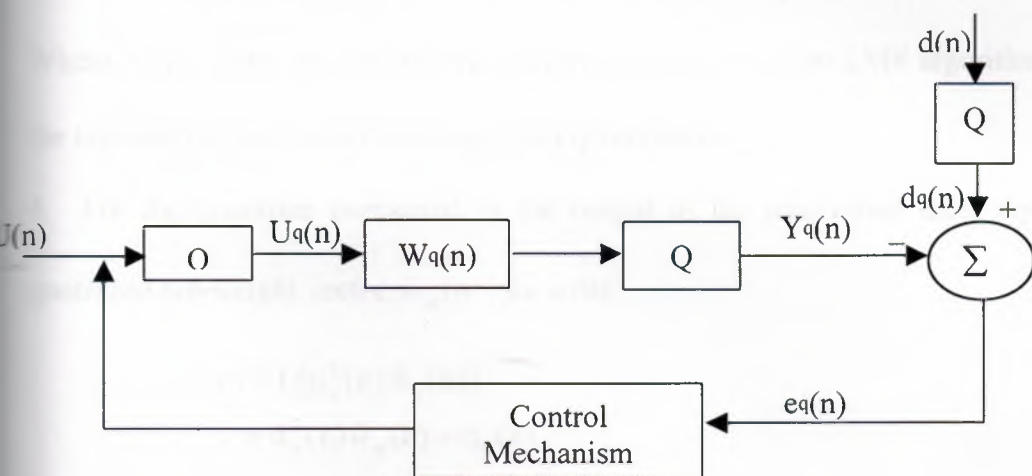


Figure 6.1 Block diagram representation of the finite-precision form of LMS algorithm

Assumption, made merely for convenience of presentation, will in no way affect the validity of the findings presented in this section.

A block diagram of the *finite-precision least-mean-square (LMS) algorithm* depicted in Fig 6.1. Each of the blocks (operators) labeled Q represents a quantizer. Each one introduces a quantization, or round-off error of its own. Specifically, we mean the input-output relations of the quantizers operating in Fig. 6.1 as follows:

1. For the input quantizer connected to $u(n)$ we have

$$\begin{aligned} U_q(n) &= Q[U(n)] \\ &= U(n) + \eta_u(n) \end{aligned} \quad (6.1)$$

Where $\eta_u(n)$ is the input quantization error vector.

2. For the quantizer connected to the desired response $d(n)$, we have

$$\begin{aligned} d_q(n) &= Q[d(n)] \\ &= d(n) + \eta_d(n) \end{aligned} \quad (6.2)$$

Where $\eta_d(n)$ is the desired response quantization error.

3. For the quantized tap-weight vector $\hat{w}_q(n)$, we write

$$\begin{aligned} \hat{w}_q(n) &= Q[\hat{w}(n)] \\ &= \hat{w}(n) + \Delta\hat{w}(n) \end{aligned} \quad (6.3)$$

Where $\hat{w}(n)$ is the tap-weight vector in the infinite-precision LMS algorithm, and $\Delta\hat{w}(n)$ is the tap-weight error vector resulting from quantization.

4. For the quantizer connected to the output of the transversal filter represented by the quantized tap-weight vector $\hat{w}_q(n)$, we write

$$\begin{aligned} y_q(n) &= Q[u_q^T(n)\hat{w}_q(n)] \\ &= u_q^T(n)\hat{w}_q(n) + \eta_y(n) \end{aligned} \quad (6.4)$$

Where $\eta_y(n)$ is the *filtered output quantization error*.

The following pair of relations describes the finite-precision LMS algorithm:

$$e_q(n) = d_q(n) - y_q(n) \quad (6.5)$$

$$\hat{w}_q(n+1) = \hat{w}_q(n) + Q[\mu e_q(n)u_q(n)] \quad (6.6)$$

Where $y_q(n)$ is itself defined in Equation (6.4). The quantizing operation indicated on the right-hand side of Equation (6.6) is not shown explicitly in Fig. 6.1; nevertheless. It is basic to the operation of the finite-precision LMS algorithm. The use of Equation (6.6) has the following practical implication. The product $\mu e_q(n)u_q(n)$, representing a scaled version of the gradient vector estimate, is quantized *before* addition to the contents of the *tap-weight accumulator*. Because of hardware constraints, this form of digital implementation is preferred to the alternative method of operating the tap-weight accumulator in double

precision and then quantizing the tap weight to single precision at the accumulator output.

In a statistical analysis of the finite-precision LMS algorithm, it is customary to make the following assumptions:

1. The input data are properly scaled so as to *prevent overflow* of the elements of the quantized tap-weight vector $\hat{w}_q(n)$ and the quantized output $y_q(n)$ during the filtering operation.
2. Each data sample is represented by B_D bits plus sign, and each tap weight is represented by B_w bits plus sign. Thus, the quantization error associated with a B_D -plus-sign bit number (i.e., data sample) has the variance

$$\sigma_D^2 = \frac{2^{-2B_D}}{12} \quad (6.7)$$

Similarly, the quantization error associated with a B_w plus-sign bit number (i.e., tap weight) has the variance

$$\sigma_w^2 = \frac{2^{-2B_w}}{12} \quad (6.8)$$

3. The elements of the input quantization error vector $\eta_u(n)$ and the desired response quantization error $\eta_d(n)$ are *white-noise* sequences, independent of the signals and from each other. Moreover, they have zero mean and variance σ_D^2 .
4. The output quantization error $\eta_y(n)$ is a white-noise sequence, independent of the input signals and other quantization errors. It has a mean of zero and a variance equal to $c\sigma_D^2$, where c is a constant that depends on the way in which the inner product $u_q^T(n)\hat{w}_q(n)$ is computed. If the individual scalar products in $u_q^T(n)\hat{w}_q(n)$ are all computed without quantization, then summed, and the final

result is quantized in B_D bits plus sign, the constant c is unity and the variance of $\eta_y(n)$ is σ_D^2 as defined in Eq. (6.7). If, on the other hand, the individual scalar products in $u^T(n)\hat{w}_q(n)$ are quantized and then summed, the constant c is M and the variance of $\eta_y(n)$ is $M\sigma_D^2$ where M is the number of taps in the transversal filter implementation of the LMS algorithm.

5. The independence theory dealing with the infinite-precision LMS algorithm, is invoked.

6.2 Total Output Mean-squared Error

The filtered output $y_q(n)$ produced by the finite-precision LMS algorithm, presents a quantized *estimate* of the desired response. The *total Output error* is therefore equal to the difference $d(n) - y_q(n)$. Using Equation (6.4), we may therefore express this error as

$$\begin{aligned} e_{\text{total}}(n) &= d(n) - y_q(n) \\ &= d(n) - u_q^T(n)\hat{w}_q(n) - \eta_y(n) \end{aligned} \quad (6.9)$$

Substituting Equation (6.1) and (6.3) in Equation (6.9), and ignoring all quantization error terms higher than first order, we get

$$e_{\text{total}}(n) = [d(n) - u^T(n)\hat{w}(n)] - [\Delta\hat{w}^T(n)u(n) + \eta_u^T(n)\hat{w}(n) + \eta_y(n)] \quad (6.10)$$

The term inside the first set of square brackets on the right-hand side of Equation (6.10) is the estimation error $e(n)$ in the infinite-precision LMS algorithm. The term inside the second set of square brackets is entirely due to quantization errors in the finite-precision algorithm. Because of assumptions 3 and 4 (i.e., the quantization errors η_u and η_y are independent of the input signals and of each other), the quantization error-related terms $\Delta\hat{w}^T(n)u(n)$, and η_y are uncorrelated with each other. Basically, for the same reason, the infinite-precision estimation

error $e(n)$ is uncorrelated with both $\eta_u^T(n)\hat{w}(n)$ and $\eta_y(n)$.

$$E[e(n)\Delta\hat{w}^T(n)u(n)] = E[\Delta\hat{w}^T(n)]E[e(n)u(n)]$$

Moreover, by invoking this same independence assumption. We may show that the expectation $E[\Delta\hat{w}(n)]$ is zero. Hence, $e(n)$ and $\Delta\hat{w}^T(n)u(n)$ are also uncorrelated.

In other words, the infinite-precision estimation error $e(n)$ is uncorrelated with all quantization-error-related terms $\Delta\hat{w}^T(n)u(n)$, $\eta_u^T(n)\hat{w}(n)$, and $\eta_y(n)$ in Equation (6.10).

Using these observations, and assuming that the step-size parameter μ is small, shown in Caraiscos and Liu (1984) that the total output *mean-squared error* produced by the finite-precision algorithm has the following *steady-state* structure:

$$E[e_{\text{total}}^2(n)] = J_{\min}(1 + M) + \zeta_1(\sigma_w^2, \mu) + \zeta_2(\sigma_D^2) \quad (6.11)$$

The first term $J_{\min}(1 + M)$ on the right-hand side of Equation (6.11) is the mean-squared error of the infinite-precision LMS algorithm. In particular, J_{\min} is the minimum mean squared error of the optimum Wiener filters, and M is the misadjustment of the infinite-precision LMS algorithm. The second term $\zeta_1(\sigma_w^2, \mu)$ arises because of the error $\Delta\hat{w}(n)$ in the quantized tap-weight vector $\hat{w}_q(n)$. This contribution to the total output mean-squared error is inversely proportional to the step-size parameter μ . The third term $\zeta_2(\sigma_D^2)$ arises because of two quantization errors: the error $\eta_u(n)$ in the quantized input vector $u_q(n)$ and the error $\eta_y(n)$ in the quantized filter output $y_q(n)$. However, unlike $\zeta_1(\sigma_w^2, \mu)$, this final contribution to the total output mean-squared error is, to a first order of approximation, independent of the step-size parameter μ .

We know that decreasing μ reduces the misadjustment M and thus leads to an improved performance of the algorithm. In contrast, the inverse dependence of the contribution $\zeta_1(\sigma_w^2, \mu)$ on μ in Equation (6.11) indicates that decreasing μ has the effect of increasing the deviation from infinite-precision performance. In practice, therefore. The step-size parameter

may only be decreased to a level at which the degrading effects of quantization errors in the tap weights of the finite-precision LMS algorithm become significant.

Since the misadjustment M decreases with μ and the contribution $\zeta_1(\sigma_w^2, \mu)$

Increases with reduced μ we may (in theory) find an optimum value of μ for which the total output mean-squared error in Equation (6.11) is minimized. However, it turns out that this minimization results in an optimum value μ_0 for the step-size parameter μ that is too small to be of practical value. In other words, it does not permit the LMS algorithm to converge completely. Indeed, Equation (6.11) for calculating the total output mean-squared error is valid only for a μ that is well in excess of μ_0 . Such a choice of μ is necessary so as to prevent the occurrence of a phenomenon known as stalling, described later in the section.

6.3 Leaky LMS Algorithm

To further stabilize the digital implementation of the LMS algorithm, we may use a technique known as leakage. Basically, leakage prevents the occurrence of overflow in a limited-precision environment by providing a compromise between minimizing the mean squared error and containing the energy in the impulse response of the adaptive filter. However, the prevention of overflow is attained at the expense of an increase in hardware cost and at the expense of degradation in performance compared to the infinite-precision form of the conventional LMS algorithm. In the leaky LMS algorithm, the cost function

$$J(n) = e^2(n) + \alpha \|\hat{w}(n)\|^2 \quad (6.12)$$

is minimized with respect to the tap-weight vector $w(n)$, where α is a positive control parameter. The first term on the right-hand side of Equation (6.12) is the squared estimation error, and the second term is the energy in the tap-weight vector $\hat{w}(n)$. [6] The minimization described herein (for real data) yields the following time update for the tap-weight vector

$$\hat{w}(n+1) = (1 - \mu\alpha)\hat{w}(n) + \mu e(n)u(n) \quad (6.13)$$

Where α is a constant that satisfies the condition

$$0 \leq \alpha < \frac{1}{\mu}$$

Except for the leakage factor $(1 - \mu\alpha)$ associated with the first term on the right-hand side of Equation (6.13), the algorithm is of the same mathematical form as the conventional LMS algorithm.

Note that the inclusion of the leakage factor $(1 - \mu\alpha)$ in Equation. (6.13) has the equivalent effect of adding a white-noise sequence of zero mean and variance α to the input process $u(n)$. This suggests another method for stabilizing a digital implementation of the LMS algorithm. Specifically. A relatively weak white-noise sequence (of variance α). Known as *dither*, is added to the input process $u(n)$. And samples of the combination are then used as tap inputs (Werner. 1983).

6.4 Stalling Phenomenon

There is another phenomenon. Known as the stalling or lock-up phenomenon, not evident from Equation (6.11), which may arise in a digital implementation of the LMS algorithm. This phenomenon occurs when the gradient estimate is not sufficiently noisy. To be specific. A digital implementation of the LMS algorithm stops adapting or stalls, whenever the correction term $\mu e_q(n)u_q(n-i)$ for the i th tap weight in the update equation is smaller in magnitude than the least significant bit (LSB) of the tap weight. As shown by (Gitlin et al.. 1973)

$$|\mu e_q(n_0)u_q(n_0 - i)| \leq \text{LSB} \quad (6.14)$$

ere, n_0 is the time at which the i th tap weights stops adapting. Suppose that the condition of equation (6.14) is first satisfied for the i th tap weight. To a first order of approximation. We may replace $u_q(n_0 - i)$ by its root-mean-square (RMS) value, A_{rms} . Accordingly, using this value in Equation (6.14), we get the following relation for the RMS value of the quantized estimation error when adaptation in the digitally implemented LMS algorithm stops:

$$|e_q(n)| \leq \frac{\text{LSB}}{\mu A_{rms}} = e_D(\mu) \quad (6.15)$$

The quantity $e_D(\mu)$, defined on the right-hand side of (6.15), is called the digital residual error.

To prevent the algorithm-stalling phenomenon due to digital effects, the digital residual error

$e_D(\mu)$ must be made as small as possible. According to the definition of

equation (6.15), this requirement may be satisfied in one of two ways:

1. The least significant bit (LSB) is reduced by picking a sufficiently large number of bits for the digital representation of each tap weight reduces
2. The step-size parameter μ is made as large as possible, while still guaranteeing convergence of the algorithm.

Another method of preventing the stalling phenomenon is to insert *dither* at the input of the

quantizer that feeds the tap-weight accumulator (Sherwood and Bershad, 1987). Dither is a

random sequence that essentially "linearizes" the quantizer. In other words, the addition of

dither guarantees that the quantizer input is *noisy* enough for the gradient quantization error

vector η_w , to be again modeled as white noise (i.e., the elements of η_w are uncorrelated in time

and with each other, and have a common variance σ_w^2). When dither is used in the manner

described here, it is desirable to minimize its effect on the overall operation of the LMS

algorithm. This is commonly achieved by shaping the power spectrum of the dither so that the

algorithm at its output effectively rejects it.

6.5 Parameter Drift

In addition to the numerical problems associated with the LMS algorithm. There is one other rather subtle problem that is encountered in practical applications of the algorithm. Specifically. Certain classes of input excitation can lead to parameter drift, that is parameter estimates or tap weights in the LMS algorithm attain arbitrarily large values despite bounded inputs. Bounded disturbances. And bounded estimation errors (Sethares et al., 1986). Although such an unbounded behavior may be unexpected, it is possible for the parameter estimates to drift to infinity while all the signals observable in the algorithm converge to zero. Parameter drift in the LMS algorithm may be viewed as a hidden form of instability, since the tap weights represent internal variables of the algorithm it may result in new numerical problems, increased sensitivity to un-modeled disturbances, and degraded long-term performance.

In order to appreciate the subtleties of the parameter drift problem. We need to introduce some new concepts relating to the parameter space. We therefore digress briefly from the issue at hand to do so.

A sequence of information-bearing tap-input vectors $u(n)$ for varying time n may be used to partition the real M -dimensional parameter space R^M into orthogonal subspaces where M is the number of tap weights (i.e., the available number of degrees of freedom). The aim of this partitioning is to convert the stability analysis of an adaptive filtering algorithm.

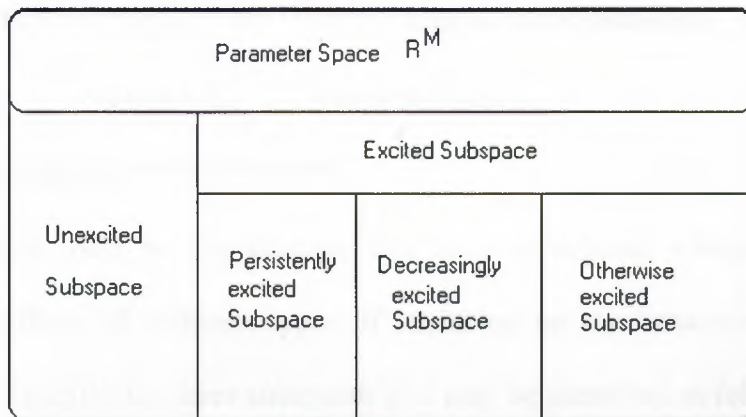


Figure 6.2 Decomposition of parameter space R^M , based on excitation.

(e.g., the LMS algorithm) into simpler subsystems and thereby provide a closer linkage between the transient behavior of the parameter estimates and the filter excitations. The partitioning we have in mind is depicted in Fig. 6.2. In particular, we may identify the following subspaces of R^M

1. The on-excited subspace. Let the M -by-1 vector z be any element of the parameter space R^M , which satisfies two conditions:

- The Euclidean norm of the vector z is 1; that is,

$$\|z\| = 1$$

- The vector z is orthogonal to the tap-input vector $u(n)$ for all but a finite number of n ; that is,

$$z^T u(n) \neq 0, \quad \text{only finitely often} \quad (6.16)$$

Let \mathcal{f}_u denote the subspace of R^M that is spanned by the set of all such vectors z . The subspace \mathcal{f}_u is called the unexcited subspace in the sense that it spans those directions in the parameter space R^M that are excited only *finitely often*.

2. The excited subspace. Let \mathcal{f}_e denote the orthogonal complement of the unexcited Subspace \mathcal{f}_u . Clearly, \mathcal{f}_e is also a subspace of the parameter space R^M . It contains those directions in the parameter space R^M that are excited *infinity often*. Thus, except for the null

vector, every element z *belonging to the subspace* \mathcal{f}_e satisfies the condition

$$z^T u(n) \neq 0, \quad \text{Infinitely often} \quad (6.17)$$

The subspace \mathcal{f}_e , is called the *excited subspace*.

The subspace \mathcal{f}_e may itself be decomposed into three orthogonal subspaces of its own, depending on the effects of different types of excitation on the behavior of the adaptive filtering algorithm. Specifically, three subspaces of \mathcal{f}_e may be identified as follows (Sethares et al. 1986):

- The persistently excited subspace. Let z be any vector of unit norm that lies in the excited subspace \mathcal{f}_e . For any positive integer m and any $\alpha > 0$, choose the vector z such that we have

$$z^T u(i) > \alpha \quad \text{for } n \leq i \leq n+m \quad \text{and for all but a finite number of } n \quad (6.18)$$

Given the integer m and the constant α , let $\mathcal{f}_p(m, \alpha)$ be the subspace spanned by all such vectors z that satisfy the condition of (6.18). There exist a finite m_0 and a positive α_0 for which the subspace $\mathcal{f}_p(m_0, \alpha_0)$ is maximal. In other words $\mathcal{f}_p(m_0, \alpha_0)$ contains $\mathcal{f}_p(m, \alpha)$ for all $m > 0$ and for all $\alpha > 0$. The subspace $\mathcal{f}_p \equiv \mathcal{f}_p(m_0, \alpha_0)$ is called the persistently excited subspace; and m_0 is called the interval of excitation. For every direction z that lies in the persistently excite subspace \mathcal{f}_p there is an excitation of level α_0 at least once in all but a finite number of intervals of length m_0 . In the persistently excited subspace, we are therefore able to find a tap-input vector $u(n)$ rich enough to excite all the internal modes that govern the transient behavior of the adaptive filtering algorithm being probed (Narendra and Annaswamy, 1989).

- The subspace of decreasing excitation. Consider a sequence $u(i)$ for which we have

$$\left(\sum_{i=1}^{\infty} |u(i)|^p \right)^{1/p} < \infty \quad (6.19)$$

Such a sequence is said to be an element of the normed linear space l^p for $1 < p < \infty$. The norm of this new space is defined by

$$\|u\|_p = \left(\sum_{i=1}^{\infty} |u(i)|^p \right)^{1/p} \quad (6.20)$$

Note that if the sequence $u(i)$ is an element of the normed linear space l^p for $1 < p < \infty$, then

$$\lim_{n \rightarrow \infty} (n) = 0 \quad (6.21)$$

Let z be any unit-norm vector z that lies in the excited subspace f_e such that for $1 < p < \infty$, the sequence $z^T u(n)$ lies in the normed linear space l^p . Let f_d be the subspace that is spanned by all such vectors z . The subspace f_d is called the subspace of decreasing excitation in the sense that each direction of f_d is decreasingly excited, for any vector $z \neq 0$, the two conditions

$$|z^T u(n)| = \alpha > 0. \quad \text{infinitely often}$$

and

$$\lim_{n \rightarrow \infty} z^T u(n) = 0$$

Cannot be satisfied simultaneously. In actual fact, we find that the subspace of decreasing excitation f_d is orthogonal to the subspace of persistent excitation f_p

The otherwise excited subspace. Let $f_p \cup f_d$ denote the union of the persistently excited subspace f_p and the subspace of decreasing excitation f_d . Let f_o denote the orthogonal complement of $f_p \cup f_d$ that lies in the excited subspace f_e . The subspace f_o is called the *otherwise excited subspace*. Any vector that lies in the subspace f_o is not unexciting, not persistently exciting, and not in the normal linear space L^p for any finite p . An example of such a signal is the sequence

$$z^T u(n) = \frac{1}{\ln(1+n)} \quad n=1,2,3,\dots \quad (6.22)$$

Returning to our discussion of the parameter drift problem in the LMS algorithm. We find that for bounded excitations and bounded disturbances. In the case of unexcited and persistently exciting subspaces the parameter estimates resulting from the application of the LMS algorithm are indeed bounded. It however, in the decreasing and otherwise excited cases, parameter drift may occur (Sethares et al., 1986). A common method of counteracting the parameter drift problem in the LMS algorithm is to introduce leakage into the tap-weight update equation of the algorithm. Here is another reason for using the leaky LMS algorithm that was described previously.

6.6 Recursive Least-Squares Algorithm

The recursive least-squares (RLS) algorithm offers an alternative to the LMS algorithm as a tool for the solution of adaptive filtering problems. We know that the RLS algorithm is characterized by a fast rate of convergence that is relatively insensitive to the eigenvalue spread of the underlying correlation matrix of the input data, and a negligible misadjustment (zero for a stationary environment without disturbances). Moreover, although it is computationally demanding (in the sense that its computational complexity is on the order of M^2 , where M is the dimension of the tap-weight vector), the mathematical formulation and therefore implementation of the RLS algorithm is relatively simple. However, there is a numerical instability problem to be considered when the RLS algorithm is implemented in finite-precision arithmetic.

Basically, numerical instability or explosive divergence of the RLS algorithm is of a similar nature to that experienced in Kalman filtering, of which the RLS algorithm is a special case. Indeed, the problem may be traced to the fact that the time-updated matrix $P(n)$ in the Riccati

equation is computed as the difference between two nonnegative definite matrices. Accordingly, explosive divergence of the algorithm occurs when the matrix $P(n)$ loses the property of positive definiteness or Hermitian symmetry.[9].

Table 6.1. Summary Of A Computationally Efficient Symmetry-Preserving Version
Of The RLS Algorithm

Initialize the algorithm by setting

$$P(0) = \delta^{-1}I. \quad \delta = \text{small positive constant}$$

$$\hat{w}(0) = 0$$

For each instant of time, $n = 1, 2, \dots$, compute

$$\pi(n) = P(n-1)u(n)$$

$$r(n) = \frac{1}{\lambda + u^H(n)\pi(n)}$$

$$K(n) = r(n)\pi(n)$$

$$\xi(n) = d(n) - \hat{w}^H(n-1)u(n)$$

$$\hat{w}(n) = \hat{w}(n-1) + K(n)\xi(n)$$

$$P(n) = \text{Tri}[\lambda^{-1}[P(n-1) - K(n)\pi^H(n)]]$$

This is precisely what happens in the usual formulation of the RLS algorithm

Described in Table 6.1 (Verhaegen. 1989).

How then can the RLS algorithm be formulated so that the Hermitian symmetry of the matrix $P(n)$ is preserved despite the presence of numerical errors? For obvious practical reasons, it would also be satisfying if the solution to this fundamental question can be attained in a computationally efficient manner. With these issues in mind, we present in Table 6.1 a particular version of the RLS algorithm from Yang (1994), which describes a computationally efficient procedure for preserving the Hermitian symmetry of $P(n)$ by design. The improved computational efficiency of this algorithm is achieved because it computes simply the upper / lower triangular part of the matrix $P(n)$, as signified by the operator $\text{Tri}\{ \}$, and then fills in

the rest of the matrix to preserve Hermitian symmetry. Moreover, Division by λ is replaced by multiplication with the precomputed value of λ^{-1} .

6.7 Error Propagation Model

According to the algorithm of Table 6.1. The recursions involved in the computation of

$P(n)$ proceed as follows:

$$\pi(n) = P(n-1)u(n) \quad (6.23)$$

$$r(n) = \frac{1}{\lambda + u^H(n)\pi(n)} \quad (6.24)$$

$$k(n) = r(n)\pi(n) \quad (6.25)$$

$$P(n) = \text{Tri}[\lambda^{-1}[P(n-1) - k(n)\pi^H(n)]] \quad (6.26)$$

Where λ is the exponential weighting factor. Consider the propagation of a *single* quantization error at time $n-1$ to subsequent recursions. Under the assumption that no other quantization errors are made. In particular, let

$$P_q(n-1) = P(n-1) + \eta_p(n-1) \quad (6.27)$$

Where the error matrix $\eta_p(n-1)$ arises from the quantization of $P(n-1)$. The corresponding quantized value of $\pi(n)$ is

$$\pi_q(n) = \pi(n) + \eta_p(n-1)u(n) \quad (6.28)$$

Let $r_q(n)$ denote the quantized value of $r(n)$. Using the defining equation (6.28), we may write

$$\begin{aligned}
r_q(n) &= \frac{1}{\lambda + u^H(n)\pi_q(n)} \\
&= \frac{1}{\lambda + u^H(n)\pi(n) + u^H(n)\eta_p(n-1)u(n)} \\
&= \frac{1}{\lambda + u^H(n)\pi(n)} \left(1 + \frac{u^H(n)\eta_p(n-1)u(n)}{\lambda + u^H(n)\pi(n)} \right)^{-1} \\
&= \frac{1}{\lambda + u^H(n)\pi(n)} - \frac{u^H(n)\eta_p(n-1)u(n)}{(\lambda + u^H(n)\pi(n))^2} + O(\eta_p^2) \\
&= r(n) - \frac{u^H(n)\eta_p(n-1)u(n)}{(\lambda + u^H(n)\pi(n))^2} + O(\eta_p^2)
\end{aligned} \tag{6.29}$$

Where $O(\eta_p^2)$ denotes the order of magnitude $\|\eta_p\|^2$.

In an ideal situation. The infinite-precision scalar quantity $r(n)$ is nonnegative, taking on values between zero and $1/\tilde{\lambda}$. On the other hand, if $u^H(n)\pi(n)$ is small compared to λ and λ itself is small enough compared to 1. Then according to Equation (6.29), in a finite-precision environment it is possible for the quantized quality $r_q(n)$ to take on a negative value large in magnitude than $1/\lambda$. When this happens. The RLS algorithm exhibits explosive divergence (Bottomley and Alexander. 1989).

The quantized value of the gain vector $k(n)$ is written as

$$\begin{aligned}
k_q(n) &= r_q(n)\pi_q(n) \\
&= k(n) + \eta_k(n)
\end{aligned} \tag{6.30}$$

Where $\eta_k(n)$ is the gain vector quantization error, defined by

$$\eta_k(n) = r(n)(I - k(n)u^H(n))\eta_p(n-1)u(n) + O(\eta_p^2) \tag{6.31}$$

Finally, using Equation (6.26), we find that the quantization error incurred in computing in updated inverse-correlation matrix $P(n)$ is

$$\eta_p(n) = \lambda^{-1}(I - k(n)u^H(n))\eta_p(n-1)(I - k(n)u^H(n))^H \tag{6.32}$$

where the term $O(\eta_p^2)$ has been ignored.

On the basis of Equation (6.32), it would be tempting to conclude that $\eta_p^H(n) = \eta_p(n)$ and therefore the RLS algorithm of Table 6.1 is *Hermitian*-symmetry preserving, if we can assume that the condition $\eta_p^H(n-1) = \eta_p(n-1)$ holds at the previous iteration. We are justified in making this assertion by virtue of the fact there is no blow-up in this formulation of the RLS algorithm, as demonstrated in what follows (it is also assumed that there is no stalling).

Equation (6.32) defines the error propagation mechanism for the RLS algorithm summarized in Table 4.1 on the basis of a single quantization error in $P^{(n-1)}$.

The matrix $I - k(n) U^H(n)$ plays a crucial role in the way in which the single quantization error $\eta_p^{(n-1)}$ propagates through the algorithm.

$$k(n) = \Phi^{-1}(n)u(n) \quad (6.33)$$

We may write

$$I - k(n)u^H(n) = I - \Phi^{-1}(n)u(n)u^H(n) \quad (6.34)$$

Next, we have

$$\Phi(n) = \lambda\Phi(n-1) + u(n)u^H(n) \quad (6.35)$$

Multiplying both sides of Equation (6.35) by the inverse matrix $\Phi^{-1}(n)$ and rearranging terms, we get

$$I - \Phi^{-1}(n)u(n)u^H(n) = \lambda\Phi^{-1}(n)\Phi(n-1) \quad (6.36)$$

Comparing Equations (6.34) and (6.36), we readily deduce that

$$I - k(n)u^H(n) = \lambda\Phi^{-1}(n)\Phi(n-1) \quad (6.37)$$

Suppose now we consider the effect of the quantization error $\eta_p^{(n_0)}$ induced at time

$n_0 \leq n$. When the RLS algorithm of Table 6.1 is used and the matrix $P^{(n)}$ remains Hermitian,

then according to the error propagation model of Equation (6.32). The quantization error

$\eta_p(n_0)$ becomes modified at time n as follows:

$$\eta_p(n) = \lambda^{-(n-n_0)} \varphi(n, n_0) \eta_p(n_0) \varphi^H(n, n_0), \quad n \geq n_0 \quad (6.38)$$

Where $\varphi(n, n_0)$ is a transition matrix defined by

$$\varphi(n, n_0) = (I - k(n)u^H(n)) \cdots (I - k(n_0 + 1)u^H(n_0 + 1)) \quad (6.39)$$

The repeated use of Equation (6.37) in (6.39) leads us to express the transition matrix in the equivalent form

$$\varphi(n, n_0) = \lambda^{n-n_0} \Phi^{-1}(n) \Phi(n_0) \quad (6.40)$$

The correlation matrix $\Phi(n)$ is defined by

$$\Phi(n) = \sum_{i=1}^n \lambda^{n-i} u(i) u^H(i) \quad (6.41)$$

On the basis of this definition, the tap-input vector $u(n)$ is said to be uniformly persistently exciting for sufficiently large n if there exist some $a > 0$ and $N > 0$ such that the following condition is satisfied (Ljung and Ljung, 1985):

$$\Phi(n) \geq aI \quad \text{for } n \geq N \quad (6.42)$$

The notation used in Equation (6.42) is shorthand for saying that the matrix $\Phi(n)$ is positive definite. The condition for persistent excitation not only guarantees the positive definiteness of $\Phi(n)$, but also guarantees its matrix norm to be uniformly bounded for $n \geq N$, as shown by

$$\|\Phi^{-1}(n)\| \leq \frac{1}{a} \quad \text{for } n \geq N \quad (6.43)$$

Returning to the transition matrix $\varphi(n, n_0)$ of Equation (6.40) and invoking the mutual consistency property of a matrix norm. We may write

$$\|\varphi(n, n_0)\| \leq \lambda^{n-n_0} \|\Phi^{-1}(n)\| \|\Phi(n_0)\| \quad (6.44)$$

Next. Invoking the inequality of (6.43). We may rewrite that of Equation (6.44) as

$$\|\varphi(n, n_0)\| \leq \frac{\lambda^{n-n_0}}{a} \|\Phi(n_0)\| \quad (6.45)$$

finally. We may use the error propagation equation (6.38) to express the vector norm of $\eta_p(n)$ as

$$\|\eta_p(n)\| \leq \lambda^{-(n-n_0)} \|\varphi(n, n_0)\| \|\eta_p(n-1)\| \|\varphi^H(n, n_0)\|$$

Which, in light of (6.48), may be rewritten as

$$\|\eta_p(n)\| \leq \lambda^{n-n_0} M \quad n \geq n_0 \quad (6.46)$$

Where M is a positive number defined by

$$M = \frac{1}{a^2} \|\Phi(n_0)\|^2 \|\eta_p(n-1)\| \quad (6.47)$$

Equation (6.47) states that the RLS algorithm of Table 6.1 is exponentially *stable* in the sense that a single quantization error $\eta_p(n_0)$ occurring in the inverse correlation matrix $P(n_0)$ at time n_0 decays exponentially provided that $\lambda < 1$ (i.e., the algorithm has finite memory). In other words, the propagation of a single error through this formulation of the standard RLS algorithm with finite memory is contractive. Computer simulations validating this result are presented in Verhaegen (1989).

However, the single-error propagation for the case of growing memory (i.e. $\lambda = 1$) is *not* contractive. The reason for saying so is that when $\lambda = 1$, neither $\|\varphi(n, n_0)\| \leq 1$ nor $\|\varphi(n, n_0)\| \leq 1$ holds, even if the input vector $u(n)$ is persistently exciting. Consequently, the accumulation of numerical errors may cause the algorithm to be divergent (Yang 1994). In an independent study, Slock and Kalath (1991) also point out that the error propagation mechanism in the RLS algorithm with $\lambda = 1$ is unstable and of a random walk type. Moreover, there is experimental evidence for this numerical divergence. Which reported in (Ardalan and Alexander, 1987).

As with the LMS algorithm, a second form of divergence. Referred to as the stalling

phenomenon, occurs when the tap weights in the RLS algorithm stop adapting. In particular this phenomenon occurs when the quantized elements of the matrix $P(n)$ become very small, such that multiplication by $P(n)$ is equivalent to multiplication by a zero matrix (Bottomley and Alexander, 1989). Clearly, The stalling phenomenon may arise no matter how the RLS algorithm is implemented.

The stalling phenomenon is directly linked to the exponential weighting factor λ and the variance σ_u^2 of the input data $u(n)$. Assuming that λ is close to unity, we find from the definition of the correlation matrix $\Phi(n)$ that the expectation of $\Phi(n)$ is given by .

$$E[\Phi(n)] \approx \frac{R}{1-\lambda} \quad \text{Large } n \quad (6.48)$$

For λ close to unity, we have

$$E[P(n)] = E[\Phi^{-1}(n)] \approx (E[\Phi(n)])^{-1} \quad (6.49)$$

Hence, using Equation (6.48) in Equation (6.49), we get

$$E[P(n)] \approx (1-\lambda)R^{-1} \quad \text{Large } n \quad (6.50)$$

Where R^{-1} is the inverse of matrix R . Assuming that the tap-input vector $u(n)$ is drawn from a wide-sense stationary process with zero mean, we may write

$$\mathfrak{R} = \frac{1}{\sigma_u^2} R \quad (6.51)$$

Where \mathfrak{R} is a *normalized correlation matrix* with diagonal elements equal to 1 and off-diagonal elements less than or equal to 1 in magnitude, and σ_u^2 is the variance of an input data sample $u(n)$. We may therefore rewrite Equation (6.50) as

$$E[P(n)] \approx \left(\frac{1-\lambda}{\sigma_u^2} \right) \mathfrak{R}^{-1} \quad \text{For large } n \quad (6.52)$$

Equation (6.52) reveals that the RLS algorithm may stall if the exponential weighting factor λ is close to 1 and/or the input data variance σ_u^2 is large. Accordingly, we may prevent stalling

of the standard RLS algorithm by using a sufficiently large number of accumulator bits in the computation of the inverse correlation matrix $P(n)$.

7.1 Design procedure

The design steps in the construction of a FIR filter are:

1. Choose a value of L . This is arbitrary, and we may set $L=1$.

2. Choose a value of T . This is arbitrary, and we may set $T=1$.

3. Choose the cutoff frequencies ω_p and ω_s , that is, calculate ω_p and ω_s using

$$\omega_p = \frac{2}{T} \tan^{-1} \left(\frac{\omega_p}{\omega_s} \right) \quad \text{and} \quad \omega_s = \frac{2}{T} \tan^{-1} \left(\frac{\omega_s}{\omega_p} \right) \quad (7.1)$$

4. Design a window filter $H_w(z)$ to meet the specifications ω_p , ω_s , R_p and R_s .

5. Finally, set

$$H(z) = H_w \left(\frac{z - z^{-L}}{1 + z^{-L}} \right)$$

7.2 Matlab implementation

MATLAB provides a function called `firpmord` to implement this design. Its invocation

is similar to the original function, but it also takes several more arguments to specify the

desired filter.

Here is the design process for a digital FIR filter implemented in MATLAB:

1. Choose the following specifications:

7. PRACTICAL CONSIDERATIONS AND DESIGN FILTERS

7.1 Design procedure

Given the digital filter specification W_p , W_s , R_p and R_s we want to determine $H(z)$.

The design steps in the procedure are following.

1. Choose a value of T . This is arbitrary, and we may set $T=1$.
2. Prewarp the cutoff frequencies W_p and W_s ; that is, calculate Ω_p and Ω_s using

$$\Omega_s = \frac{2}{T} \tan\left(\frac{\omega_s}{2}\right), \quad \Omega_p = \frac{2}{T} \tan\left(\frac{\omega_p}{2}\right) \quad (7.1)$$

3. Design an analog filter $H_a(s)$ to meet the specifications Ω_p , Ω_s , R_p and R_s .
4. Finally, set

$$H(z) = H_a\left(\frac{21 - z^{-1}}{T1 + z^{-1}}\right)$$

7.2 Matlab implementation

MATLAB provides a function called `bilinear` to implement this mapping. Its invocation is similar to the `impinvar` function, but it also takes several forms for different in-put out-put quantities.

Here is the design procedure of digital IIR filters(butterworth, chebyshev and elliptic) under the following specifications:

$W_p=0.2\pi$, $W_s=0.3\pi$, $R_p=1\text{dB}$ and $R_s=15\text{dB}$.

» % **DIGITAL FILTER SPECIFICATIONS**

» $W_p=0.2*\pi$; $W_s=0.3*\pi$; $R_p=1$; $R_s=15$;

» % inverse mapping for freq:

» $T=1$; $F_s=1/T$;

» $Op=(2/T)*\tan(W_p/2)$; % prewrap prototype passband freq

» $Os=(2/T)*\tan(W_s/2)$; % prewrap prototype stopband freq

» % **Butterworth** filter order calculation:

» $[N,W_n]=\text{buttord}(Op,Os,R_p,R_s,'s')$;

» $[\text{num},\text{den}]=\text{butter}(N,W_n,'s')$;

» %bilinear transformation

» $[\text{b},\text{a}]=\text{bilinear}(\text{num},\text{den},F_s)$;

» %magnitude and phase response

» $\text{freqz}(\text{b},\text{a},512)$;

As Shown in figure 7.1.

Magnitude Response (dB)

Phase (degrees)

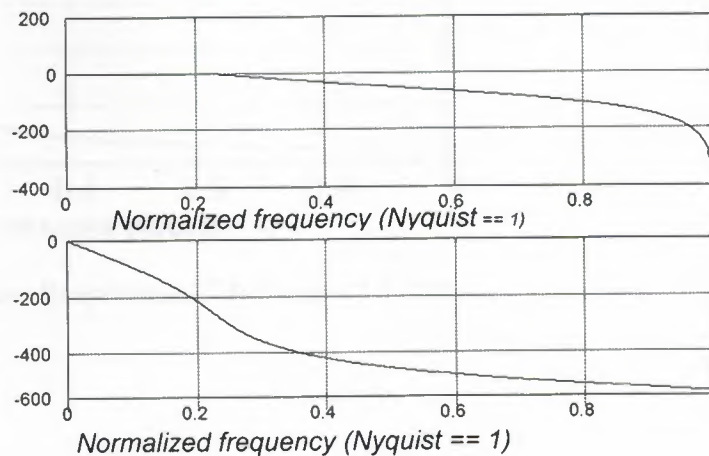


Figure 7.1 Magnitude and Phase Response of ButterWorth Filter.

```
»% Chebyshev1 filter order calculation:
```

```
» [N,Wn]=cheblord(Op,Os,Rp,Rs,'s');
```

```
» [num,den]=cheby1(N,Rp,Wn,'s');
```

```
» %bilinear transformation
```

```
» [b,a]=bilinear(num,den,Fs);
```

```
» %magnitude and phase response
```

```
» freqz(b,a,512);
```

As Shown in figure 7.2.

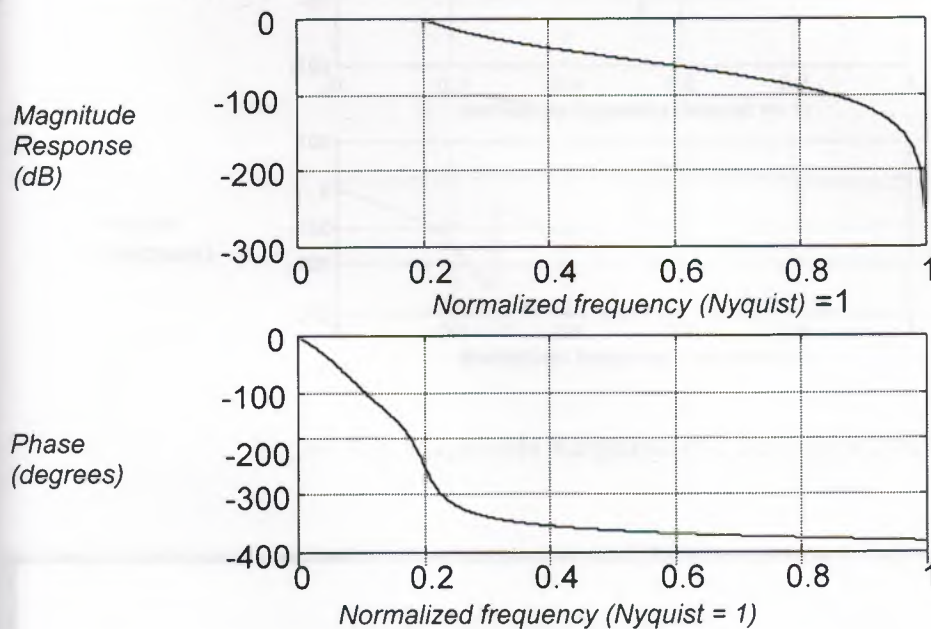


Figure 7.2 Magnitude and Phase Response of Chebyshev 1 Filter.

```
»%Chebyshev type 2 order calculation:
```

```
» [N,Wn]=cheb2ord(Op,Os,Rp,Rs,'s');
```

```
» [num,den]=cheby2(N,Rs,Wn,'s');
```

```
» [b,a]=bilinear(num,den,Fs);
```

```
» freqz(b,a,512);
```

As Shown in Figure 7.3.

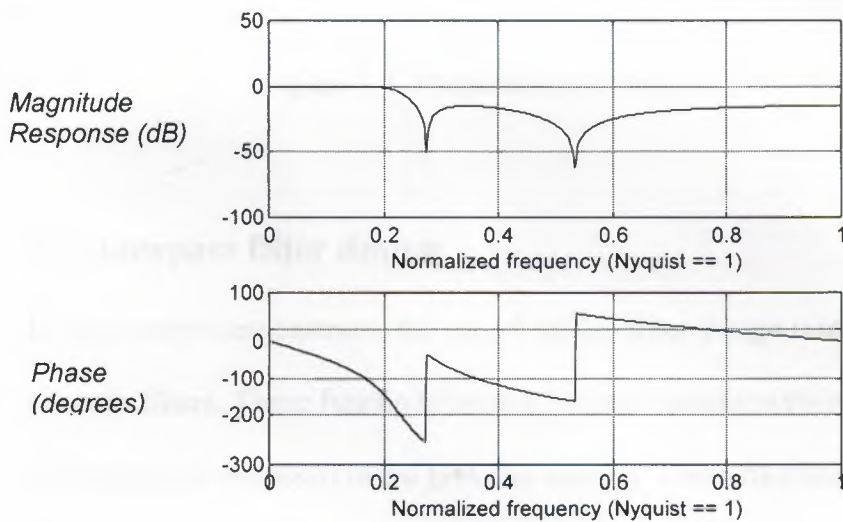


Figure 7.3. Magnitude Response of Chebyshev 2 Filter.

```
»%Elliptic filter order calculation:
```

```
» [N,Wn]=ellipord(Op,Os,Rp,Rs,'s');
```

```
» [num,den]=ellip(N,Rp,Rs,Wn,'s');
```

```
» [b,a]=bilinear(num,den,Fs);
```

```
» freqz(b,a,512);
```

As shown in figure 7.4.

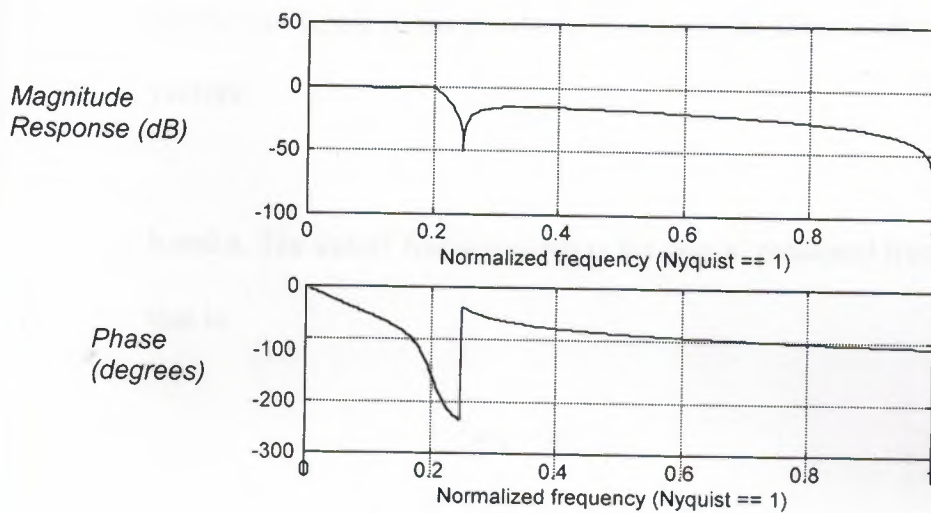


Figure 7.4. Magnitude and Response of Elliptic Filter.

7.3 Lowpass filter design

In this section demonstrates the use of matlab filter design routines to design digital lowpass filters. These functions use the bilinear transformation because of its desirable advantages as discussed in the previous section. These functions are as follows:

1. $[b,a]=bntttr(N,wn)$

This function designs an Nth-order lowpass digital Butterworth filter and returns the filter coefficients in length N+1 vectors b and a. In MATLAB all digital frequencies are given in unit of λ . Hence wn is computed by using the following relation:

$$\omega_n = \frac{2}{\Pi} \tan^{-1} \left(\frac{\Omega_c T}{2} \right)$$

2. $[b,a]=cheby1(N,Rp,wn)$

This function designs an Nth-order lowpass digital Chebyshev-I filter with R_p decibels of ripple in the passband. It returns the filter coefficients in length $N + 1$ vectors

b and a. The cutoff frequency ω_n is the digital passband frequency in units of λ ; that is,

$$\omega_n = \omega_p / \pi$$

3. [b,a]=cheby2[N,As,Wn)

This function designs an Nth-order lowpass digital Chebyshev-II filter with the stopband attenuation A_s decibels. It returns the filter coefficients in length $N + 1$ vectors b and a. The cutoff frequency ω_n is the digital stopband frequency in units of λ ; that is,

$$\omega_n = \omega_s / \pi$$

4. [b,a]=ellip[N,Rp,As,Wn)

This function designs an Nth-order lowpass digital elliptic filter with the passband ripple of A_p decibels and a stopband attenuation of A_s decibels. It returns the filter coefficients in length $N + 1$ vectors b and a. The cutoff frequency ω_n is the digital pass band frequency in units of λ ; that is,

$$\omega_n = \omega_p / \pi$$

Here is matlab implementation of above functions for the designing of lowpass Butterworth, Chebyshev and Elliptic filters under the following

specifications. Passband normalized edge frequency $W_p=0.2\pi$, stopband edge frequency $W_s=0.3\pi$, passband ripples $R_p=1\text{dB}$ and stopband attenuation $R_s=15\text{dB}$.

At the end a comparison is given to choose a best one.

a. Butterworth lowpass filter:

```

» Wp=0.2*pi;      Ws=0.3*pi;      Rp=1;      Rs=15;

% analog prototype specification

» T=1;

» Op=(2/T)*tan(Wp/2); %prewrap prototype passband freq.

» Os=(2/T)*tan(Ws/2); %ptewrap prototype stopband freq.

» % analog butterworth prototype order calculation

» N=ceil((log10((10^(Rp/10)-1)/(10^(Rs/10)- 1)))/(2*log10(Op/Os)));

» fprintf('**Butterworth filter order=%2.0f\n',N);

**Butterworth filter order= 6

» % analog Butterworth prototype cutoff frequency

» Oc=Op/((10^(Rp/10)-1)^(1/(2*N))); %analog cutoff freq.

» Wn=2*atan((Oc*T)/2); %digital cutoff freq.

» %Digital filter design

» Wn=Wn/pi; %cutoff freq. In pi unit

» [b,a]=butter(N,Wn);

» disp(a);

1.0000  2.3505  2.8579  2.0069  0.8539  0.2034  0.0211

» disp(b);

0.1452  0.8713  2.1782  2.9043  2.1782  0.8713  0.1452

```


b. Chebyshev -1 lowpass filter :

```

» % Analog chebyshev-1 prototype order calculation
» ep=sqrt(10^(Rp/10)-1);      %passband ripple factor
» Oc=Op;                      %analog cutoff freq.
» Or=Os/Op;                   %Transition ratio
» A=10^(Rs/20);               %stopband attenuation
» g=sqrt(A*A-1)/ep;           %Intermediate cal.
» N=ceil(log10(g+sqrt(g*g-1))/log10(Op+sqrt(Or*Or-1)));
» fprintf('chebyshev-1 filter order =%2.0f\n',N);
chebyshev-1 filter order = 4
» % digital chebyshev-1 filter design
» Wn=Wp/pi;                   %passband freq. In unit of pi.
» [b,a]=cheby1(N,Rp,Wn);
» disp(a);
1.0000 -3.9634 6.6990 -5.9815 2.8111 -0.5558
» disp(b);
0.0003 0.0015 0.0029 0.0029 0.0015 0.0003

```

c. Chebyshev-2 lowpass filter design:

```

» % analog chebyshev-2 order calculation

» ep=sqrt(10^(Rp/10)-1);

» A=10^(Rs/20);

» g=sqrt(A*A-1)/ep;

» Oc=Op;

» Or=Os/Op;

» N=ceil(log10(g+sqrt(g*g-1))/log10(Or+sqrt(Or*Or-1)));

» fprintf('***chebyshev-2 filter order is=%2.0f',N);

***chebyshev-2 filter order is= 4

» % digital chebyshev-2 filter design

» Wn=Ws/pi;

» [b,a]=cheby2(N,Rs,Wn);

» disp(a);

    1.0000  -1.5508   1.3423  -0.4707   0.1079

» disp(b);

    0.1797  -0.0916   0.2525  -0.0916   0.1797

```

d. Elliptic lowpass filter:

```

» % analog ellip prototype order calculation
» ep=sqrt(10^(Rp/10)-1);
» A=10^(Rs/20);
» g=sqrt(A*A-1)/ep;
» k=Op/Os;
» k1=ep/sqrt(A*A-1);
» capk=ellipke([k.^2 1-k.^2]);
» capk1=ellipke([(k1.^2) 1-(k1.^2)]);
» N=ceil(capk(1)*capk1(2)/(capk(2)*capk1(1)));
» fprintf('***\n ellip filter order=%2.0f,N);
*** ellip filter order= 3

» % Digital Elliptic filter design
» Wn=Wp/pi;
» [b,a]=ellip(N,Rp,Rs,Wn);
» disp(a);
1.0000  3.0000  3.0000  1.0000

» disp(b);
1.0000  3.0000  3.0000  1.00

```


7.4 Comparison of three filters

The filters compared in terms of order N and stopband attenuation R_s under the specification they were designed ,the comparison is shown in table 7.1

Prototype	Order N	Stopband Attenuation (dB)
Butterworth	6	15
Chebyshev-1	4	25
Elliptic	3	27

Clearly the Elliptic prototype gives the best design. However, if we compare their phase response ,elliptic design has the most non linear phase response in the passband.

To compare different filters suppose we have the following input signal

$$S(t) = \sin 2\pi(5)t + \sin 2\pi(15)t + \sin 2\pi(30)t.$$

Using MATLAB Implementation as

```
% Here's an example of filtering with the Signal Processing
% Toolbox. First make a signal with three sinusoidal
% components (at frequencies of 5, 15, and 30 Hz).

Fs=100;
t=(1:100)/Fs;

s1=sin(2*pi*t*5); s2=sin(2*pi*t*15); s3=sin(2*pi*t*30);

s=s1+s2+s3;

plot(t,s);
```

The signal is shown in figure 7.5

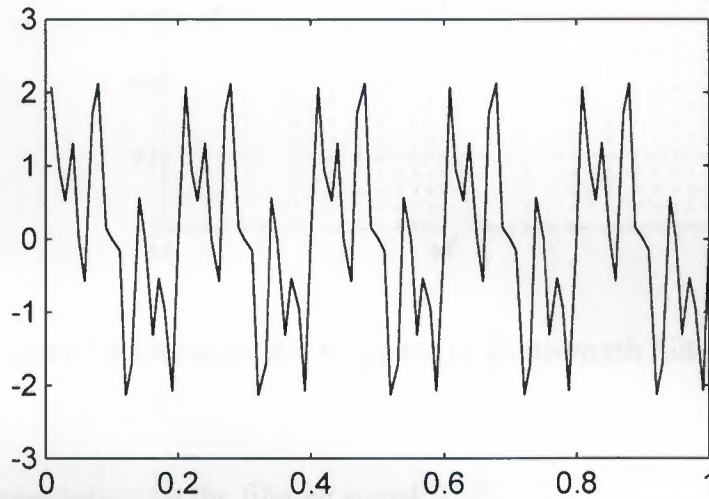


Figure 7.5. The distorted signal.

To design a filter to keep the 15 Hz sinusoid and get rid of the 5 and 30 Hz sinusoids, we create an eighth order IIR filter with a passband from 10 to 20 Hz. Here is its frequency response. The filter was created with the `ELLIP` command.

to find the poles and zeros and the gain of an order n analog filter of the appropriate type with cutoff frequency 1 rad/sec. And to plot the filter

To design a Butterworth filter

```
[b,a] = butter(4,0.1,40,[10 20]*2/fs);
```

```
[h,w] = freqz(b,a,512);
```

```
plot(w*fs/(2*pi),abs(h));
```

The following figure 7.6. shows the charecteristic response of the butterwoth filter

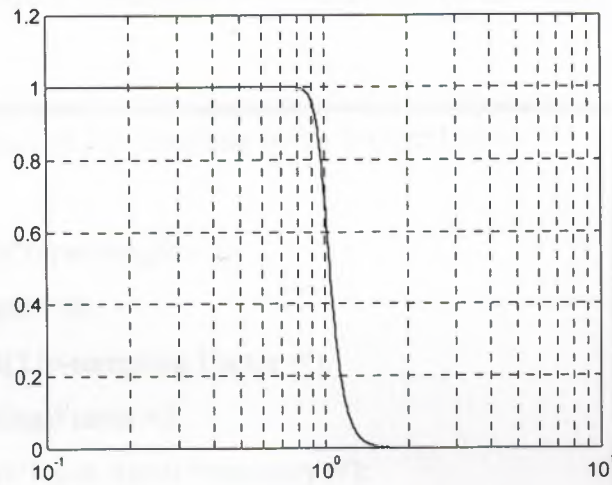
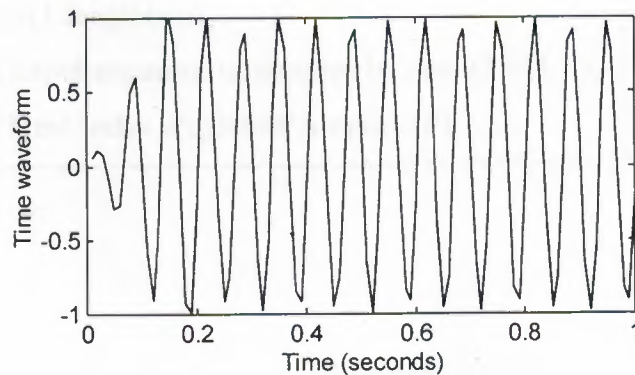


Figure 7.6 Characteristic Response of Butterworth Filter.

MATLAB Implementation for the filtered signal

```
sf=filter(b,a,s);
» plot(t,sf);
» xlabel('Time (seconds)');
» ylabel('Time waveform');
» axis([0 1 -1 1]);
```

The filtered signal is shown in figure 7.7 below.



MATLAB Implementation for Illustration of Up-sampling by an integer Factor as shown in figure 7.8.

```

» %Illustrate of Up-sampling by an Integer Factor
» clf;
» N=input('Input length =');
Input length =50
» L=input('Up-sampling Factor =');
Up-sampling Factor =3
» fo=input('Input signal frequency =');
Input signal frequency =0.12
» %Generate the input sinusoidal sequence
» n= 0:N-1;
» x= sin(2*pi*fo*n);
» %Generate the Up-sampled sequence
» y= zeros(1, L*length(x));
» y([1: L: length(y)])= x;
» %Plot the input and the output sequences
» subplot(2,1,1)
» stem(n,x);
» title('InputSequence');
» xlabel('Time index n');ylabel('Amplitude');
» subplot(2,1,2)
» stem(n,y(1:length(x)));
» title(['Output sequence upsampled by', num2str(L)]);
» xlabel('Time index n');ylabel('Amplitude');

```

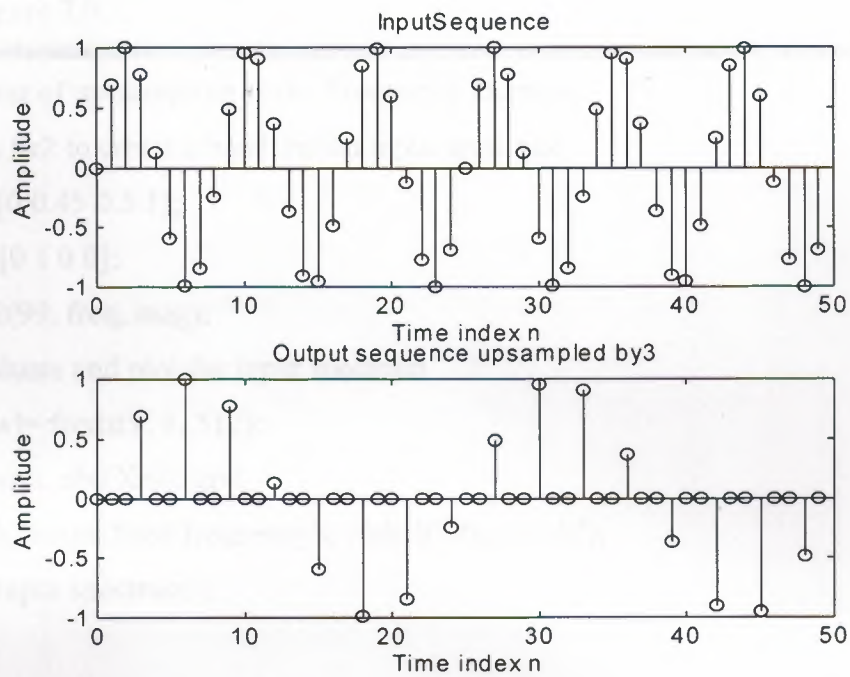


Fig. 7.8. Illustrate of Up-sampling by an Integer Factor

MATLAB Implementation for the Effect of Up-sampling in the Frequency Domain as shown in figure 7.9.

```
%Effect of up-sampling in the Frequency Domain
%Use fir2 to create a bandlimited input sequence
freq=[0 0.45 0.5 1];
mag=[0 1 0 0];
x=fir2(99, freq, mag);
%Evaluate and plot the input spectrum
[Xz, w]= freqz(x, 1, 512);
plot(w/pi, abs(Xz)); grid
xlabel('Normalized frequency'); ylabel('Magnitude');
title('Input spectrum');
pause
%Generate the up-sampled sequence
L=input('Type in the up-sampling factor=');
Type in the up-sampling factor=5
y=zeros(1, L*length(x));
y([1:L:length(y)])= x;
```

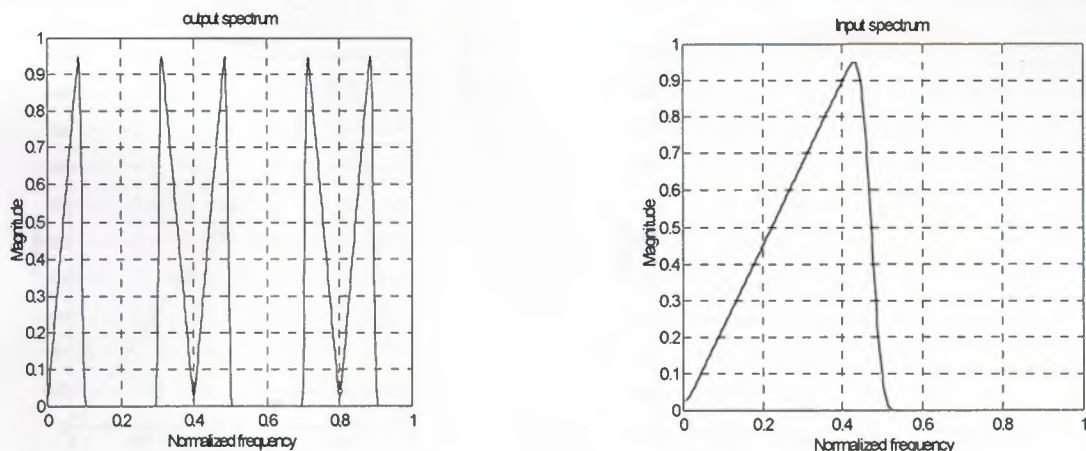


Fig 7.9. MATLAB generated input and output spectrum of a factor-of-5 up-sampler.

MATLAB Implementation for the Effect of Down-sampling in the frequency Domain is shown in figure 7.10.


```

» %Effect of Down-sampling in the Frequency Domain
» %Use fir2 to create a badlimited input sequence
» freq=[0 0.42 0.48 1];
» mag=[0 1 0 0];
» x=fir2(101, freq, mag);
» %Evaluate and plot the input spectrum
» [Xz, w]=freqz(x, 1, 512);
» plot(w/pi, abs(Xz)); grid
» xlabel('Normalized frequency');ylabel('Magnitude');
» title('Input spectrum');
» pause
» %Generate the down-sampled sequence
» M=input('Type in the down-sampling factor =');
Type in the down-sampling factor =2
» y=x([1: M: length(x)]); %Evaluate and plot the output
spectrum
» [Yz, w]=freqz(y, 1, 512); plot(w/pi, abs(Yz));grid
» xlabel('Normalized frequency'); ylabel('Magnitude');
title('Output spectrum');
    
```

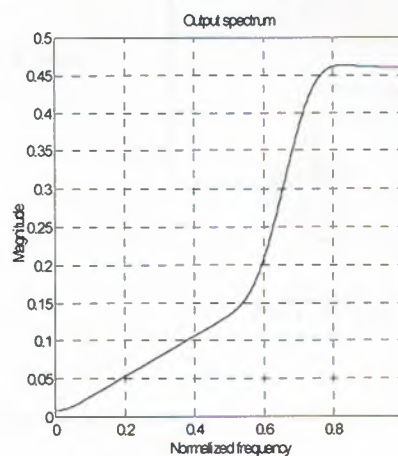
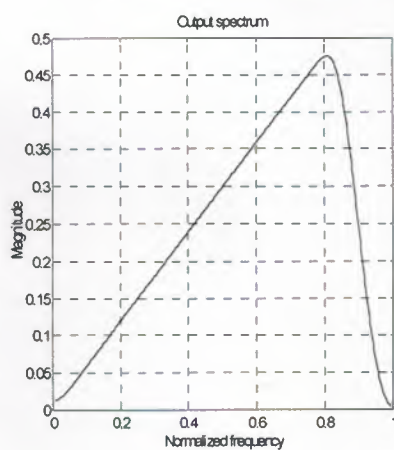


Figure.7.10. Effect of Down-sampling in the Frequency Domain use fir2 to create a bad-limited input sequence

7.5 Adaptive LMS

The following Block Diagram represents the Adaptive Noise Cancellation Using LMS Algorithm.

Adaptive Noise Cancellation

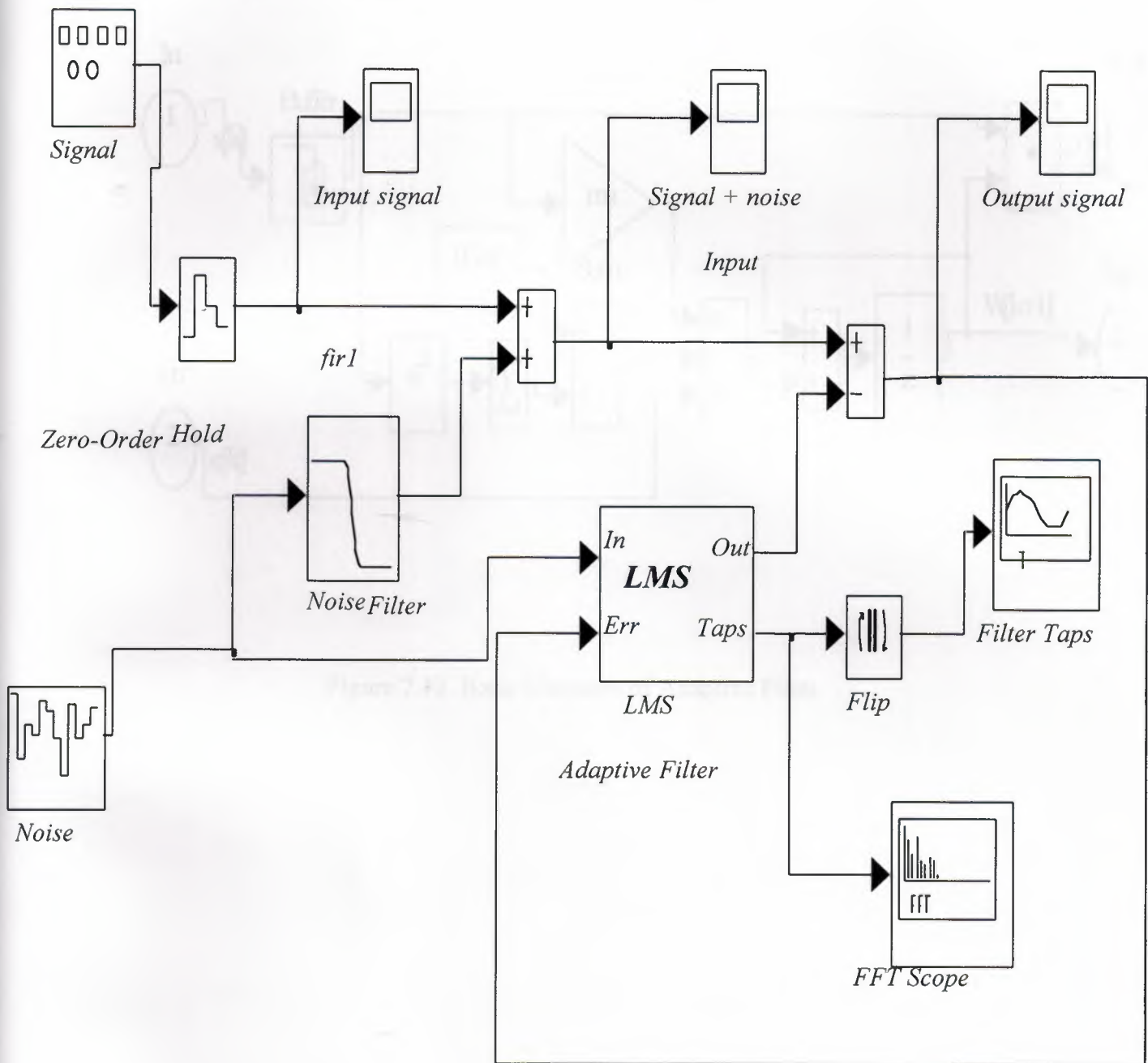


Figure 7.12. Block Diagram of Adaptive Noise Cancellation.

Basic Elements of the Adaptive Noise Cancellation are:

7.5.a. LMS Adaptive Filter:

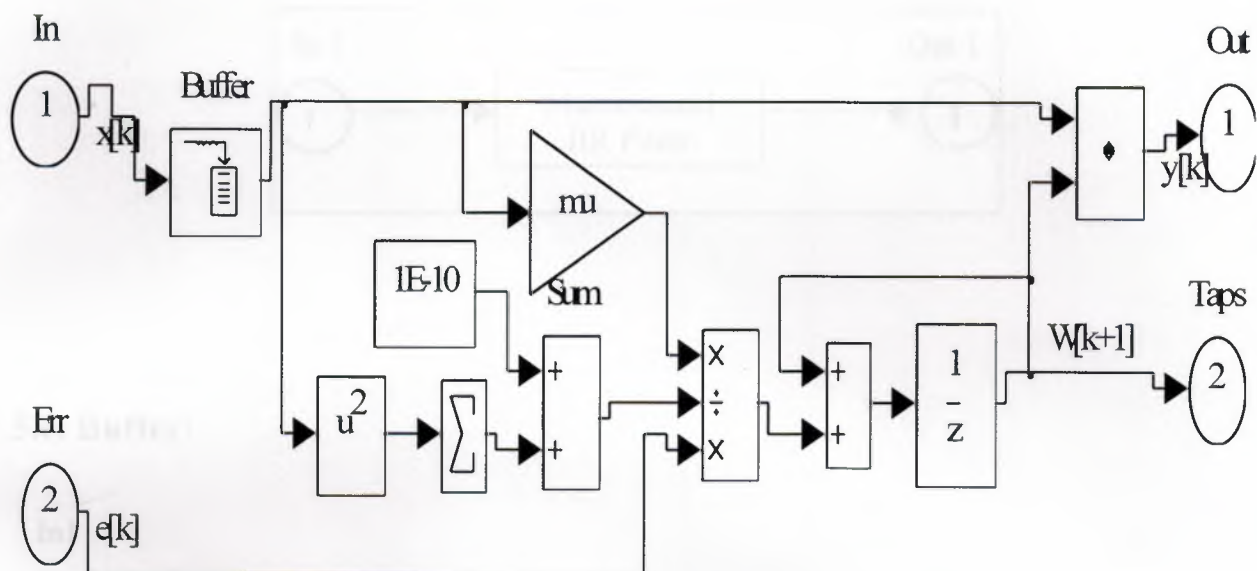


Figure 7.13. Basic Elements of Adaptive Filter.

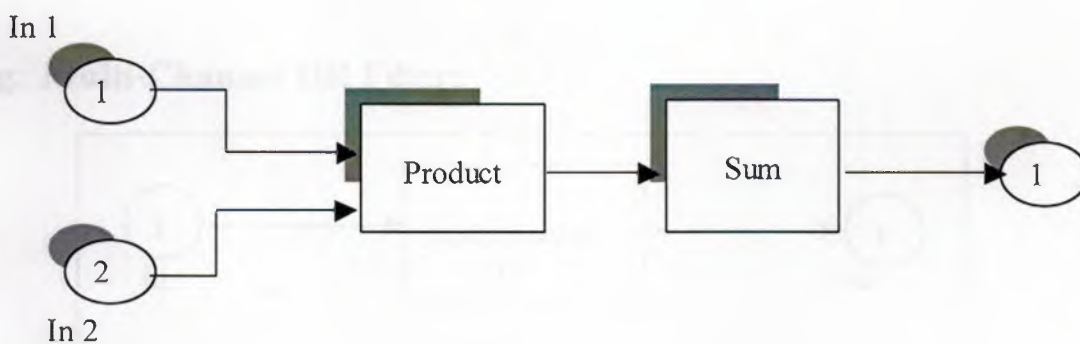
7.5.b. Digital FIR Filter Design:



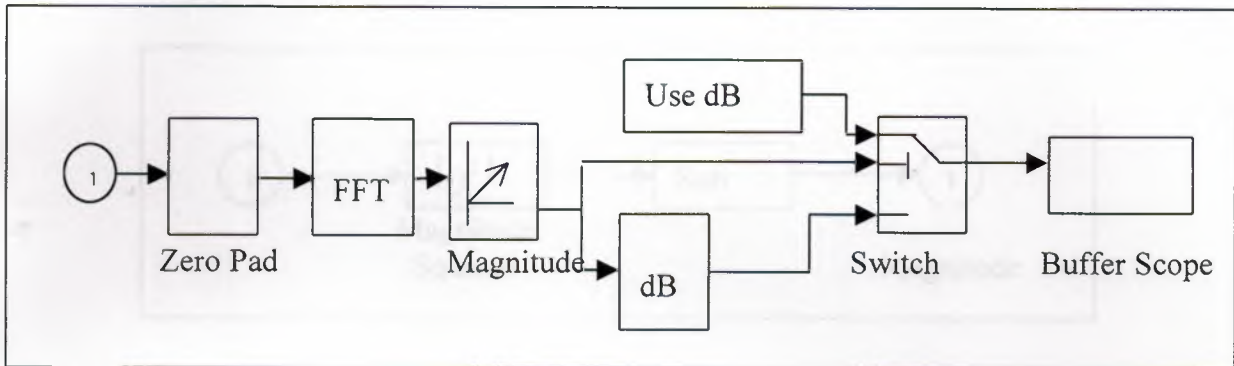
7.5.c. Buffer:



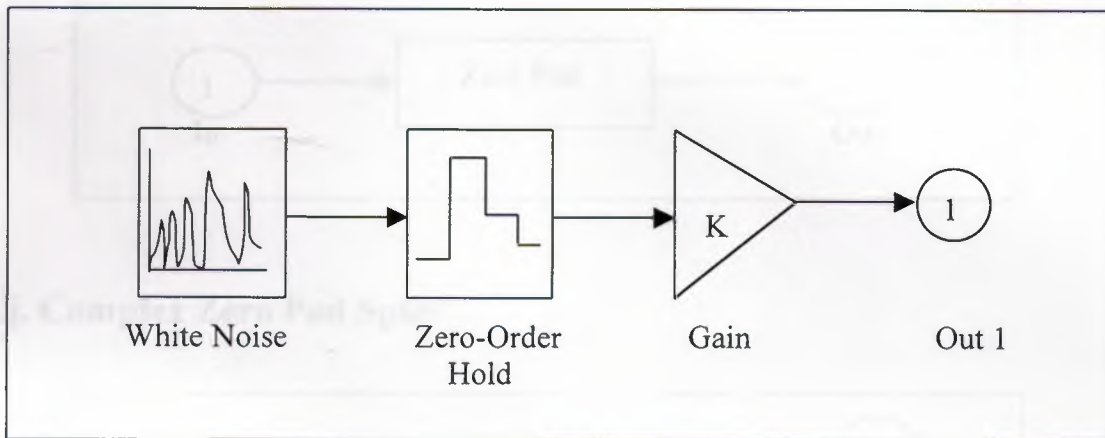
7.5.d. Dot Product Unit:



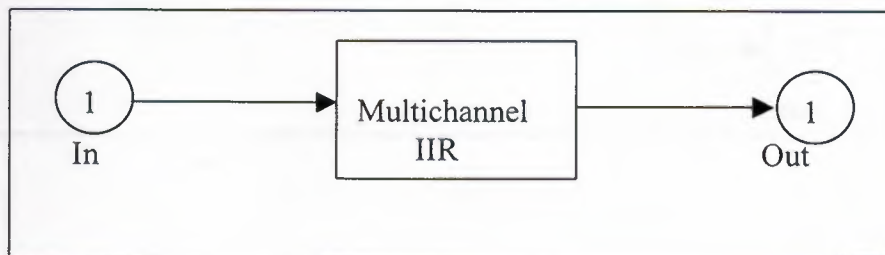
7.5.e FFT Scope:



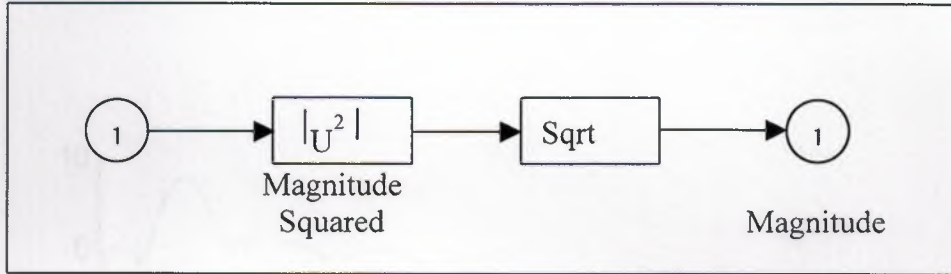
7.5.f. Band Limited White Noise:



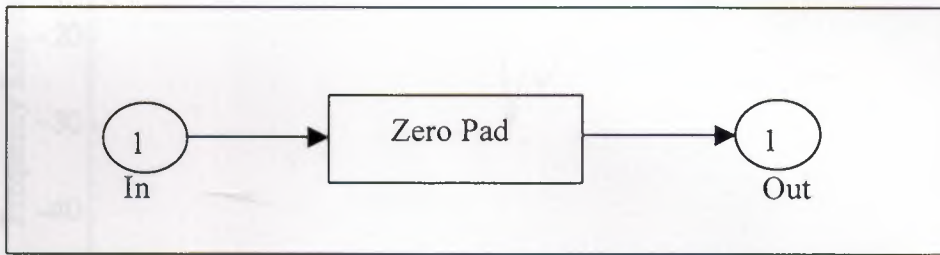
7.5.g. Multi-Channel IIR Filter:



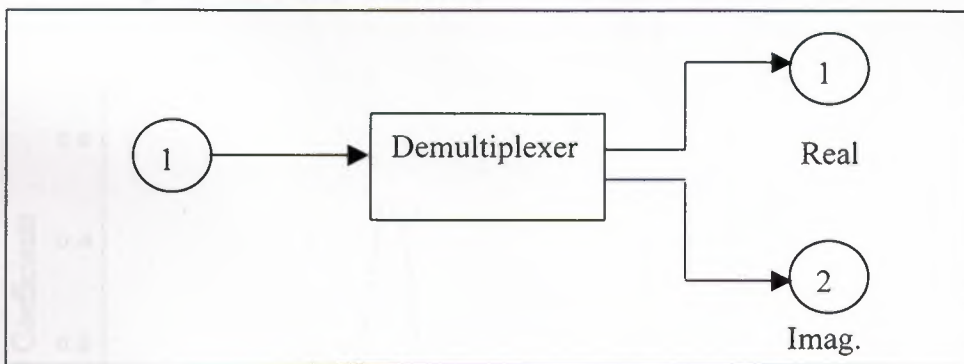
7.5.h. Magnitude:



7.5.i. Zero Pad:



7.5.j. Complex Zero Pad Split:



The computer results, Frequency Response of the Adaptive Filter as shown in fig. 7.14,
Adaptation Coefficients of the Adaptive Filter as shown in fig. 7.15,
Signals Obtained from Adaptive Filter Using LMS Algorithm as shown in fig. 7.16.

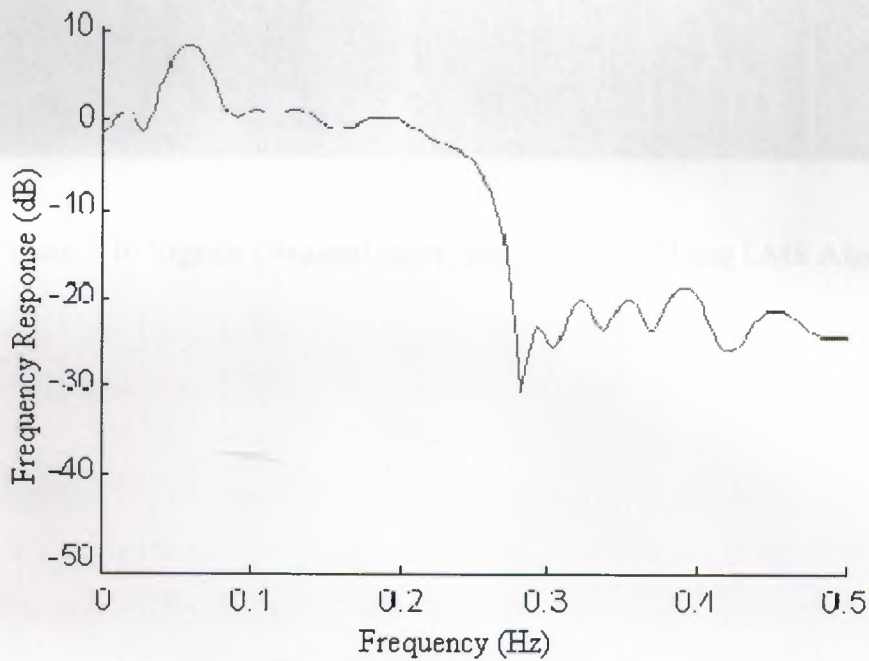


Figure 7.14. Frequency Response of the Adaptive Filter.

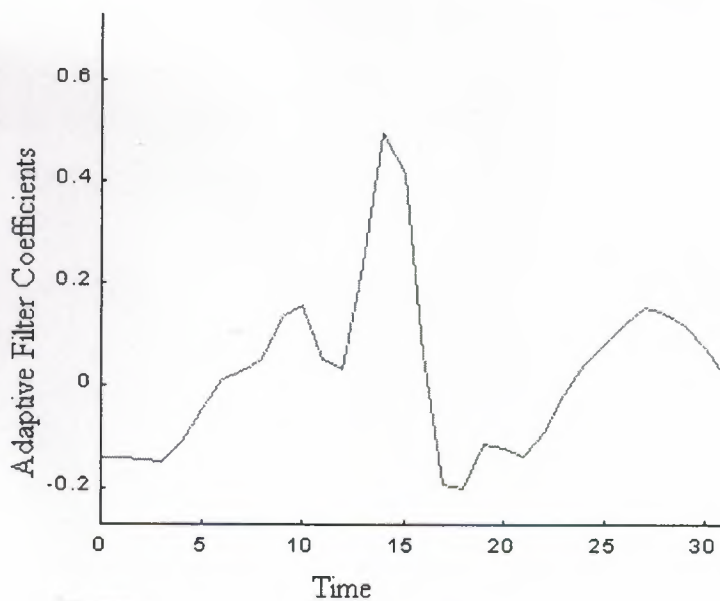


Figure 7.15. Adaptation Coefficients of the Adaptive Filter

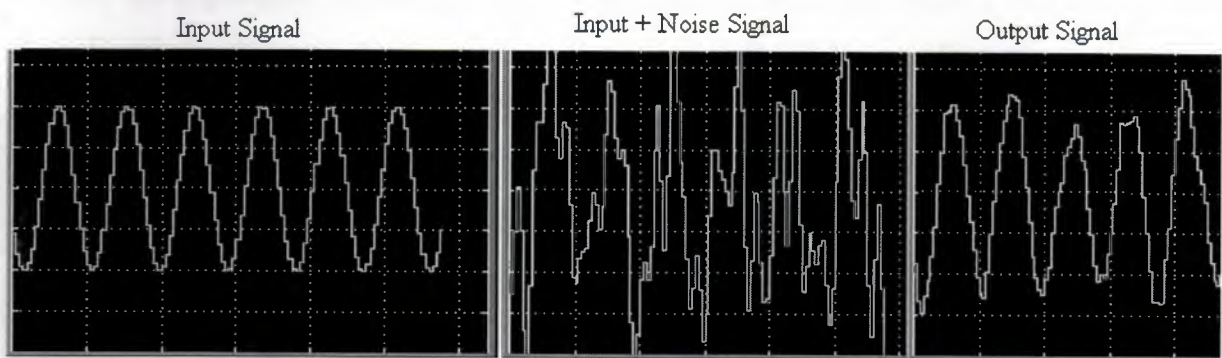


Figure 7.16 Signals Obtained from Adaptive Filter Using LMS Algorithm.

CONCLUSION

* While the importance of analog filters is continuously being continuously reduced by their digital counterparts, they remain an important study, if for no other reason than they provide a gateway to the study of digital filters. An anti-aliasing filter introduced to the signal processing system is based on analog filtering. Analog and digital conventional filters provide filtering if there is no overlap between spectrum signals and noise.

* The design of a Wiener filter requires a priori information about the statistics of the data to be processed. The filter is optimum only when the statistical characteristics of the input data match the priori information on which the design of the filter is based.

When this information is not known completely, however, it may not be possible to design the Wiener filter or else the design may no longer be optimum.

* An adaptive filter having self organizing structure based on recursive algorithm make it possible to perform satisfactory filtering in an environment where complete knowledge of the relevant signal characteristics are not available.

* Comparison between LMS algorithm and other algorithms show that LMS algorithm is simple for realization and computation, and it does not require off-line gradient estimations of the data. But instead knowledge of signal statistics, it uses instantaneous estimation. These estimates improve gradually with time as the weights are adjusted and the filter learns the characteristics of the signal.

* The performance limits the adaptive echo cancellation techniques are investigated. In particular was analyzed the effects of signal characteristics such as auto-and cross-correlation on the achievable echo suppression. Techniques to enhance signal characteristics such as linearity and signal to noise ratio are identified.

* Acoustic echoes generated by telephone hand-set and line echo, results obtained based on the analysis and interpretation theory and design of adaptive filters were recommended for echo cancellation in the digital telephone communication system.

Simulation echo cancellator by MATLAB, show that it satisfy the requirements for real time noise cancellation.

* Wireless and personal communication systems are increasingly being regarded as essential communication tool for future. From this point of view as new network infrastructure are implemented and the competition between them increases.

* The requirements of high voice quality from network provider are required, solution of this need investigation of adaptive filtering in the wireless and personal communication systems where frequency scattering is one of the important problems.

- [1] Haykin, S. *Adaptive Filter Theory* Prentice-Hall, 1996.
- [2] Lick, Z. *On synthesis of an complexity in adaptive parameter estimation*, *IEEE Trans. on Signal Processing*, 40(3):720-729, March 1992.
- [3] Mayyas and T. K. Alouane, *Least LMS algorithm for analysis of gaussian data*, *IEEE Trans. on Signal Processing*, 45(4):1017-1024, April 1997.
- [4] Moraw, R. S. Ungar and P. J. Schmitt, *Adaptive cancellation and applications*, *IEEE Trans. on Signal Processing*, 35(1):1-14, Jan. 1987.
- [5] Pothier, J. A. G. and J. L. T. *Adaptive cancellation filter: An efficient adaptive filter for noise cancellation*, *IEEE Trans. on Signal Processing*, 43(1):100-110, Jan. 1995.
- [6] Soderstrom and T. S. H. *Adaptive filtering in the time domain*, *IEEE Trans. on Signal Processing*, 35(1):1-14, Jan. 1987.
- [7] Makh and T. S. H. *Adaptive filtering in the time domain*, *IEEE Trans. on Signal Processing*, 35(1):1-14, Jan. 1987.

REFERENCES

- [1] Cioffi and J. J. M. A. C. Bingham, A datadriven multitone echo canceller, *IEEE Trans. on Signal Processing*, 42(10):2853–2869, Oct. 1994.
- [2] Chao, J. S. Kawabe, and S. Tsujii, A new IIR adaptive echo canceler: GIVE, *IEEE Trans. on Comm.*, 42(12):3090–3094, Dec. 1994.
- [3] Eriksson, A. G. Eriksson, J. Karlsen, A. Roxstrom, and T. V. Hulth, Ericsson echo cancellers, *Ericsson Review*, (1):25–33, 1996.
- [4] Haykin, S. *Adaptive Filter Theory*. Prentice-Hall, Upper Saddle River, NJ, 1996.
- [5] Liu, Z. *Qr methods of on complexity in adaptive parameter estimation*, *IEEE Trans. on Signal Processing*, 43(3):720–729, March 1995.
- [6] Mayyas and T. K. Aboulnasr, Leaky LMS algorithm: Mse analysis for gaussian data, *IEEE Trans. on Signal Processing*, 45(4):927–934, April 1997.
- [7] Murano, K. S. Unagami, and F. Amano, Echo cancellation and applications, *IEEE Comm. Magazine*, 28(1):49–55, Jan. 1990.
- [8] Petillon, T. A. Gilloire, and S. Theodoridis, The fast newton transversal filter: An efficient scheme for acoustic echo cancellation in mobile radio, *IEEE Trans. on Signal Processing*, 42(3):509–518, March 1994.
- [9] Sayed and T. A. H. Kailath, A state-space approach to adaptive RLS filtering, *IEEE Signal Processing Magazine*, 11(3):18–60, July 1994.
- [10] Slock and T D. T. M.. Kailath, Numerically stable fast transversal filters for recursive least squares adaptive filtering, *IEEE Trans. on Signal Processing*, 39(1):92–114, Jan. 1991.

- [11] Sondhi and W. M. M. Kellermann. *Advances in Speech Signal Processing*, chapter Adaptive echoes cancellation for speech signals, pp. 327–356. Marcel-Decker, 1991.
- [12] Vaseghi, S. *Advanced Signal Processing and Digital Noise Reduction*. John Wiley & Sons, Chichester, NY, 1996.
- [13] Vinay K. Ingle, *Digital Signal Processing*. The PWS BookWare Companion Series, International Thomson Publishing, ITP, 1997.
- [14] Yasukawa, H. I. Furukawa, and Y. Ishiyama, Acoustic echo control for high quality audio teleconferencing, In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pp. 2041–2044, 1989.
- [15] The LMS Adaptive Filter --<http://lcavwww.epfl.ch/~prandoni/dsp/echo/node3.html>
- [16] A.T.M. MAHMUDUR RAHMAN Previous slide Next slide Back to first slide View graphic version --http://www.eas.asu.edu/~speech/research/sbc/pres5_5/tsld1...
- [17] Adaptive Signal Processing, 1/e Bernard Widrow, Stanford University Samuel Stearns, University of New Mexico Published March, 1985 by Prentice Hall Engineering, Science & Math Copyright 1985, pp. Cloth ISBN 0-13-004029-0 Price not available Sign up.--<http://www.disc.ua.es/~emartin/webdoce/Biblio.docen/widro...>
- [18] IEEE PIMRC '96 IEEE International Symposium on Personal, Indoor and Mobile Radio Communications Adaptive Algorithms for Asynchronous DS-CDMA Receivers. Graeme Woodward, Predrag Rapajic and Branka Vucetic. Department of Electrical Engineering....--<http://wwwsyseng.amu.edu.au/~prapajic/pap/con/1996/96PIMR...>