INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

**ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600**

UMI[®]

Effects of Vocoder Distortion and Packet Loss on Network Echo Cancellation

submitted by

Ying Huang, B. Eng., M. Eng.

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of

Master of Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
Faculty of Engineering
Department of Systems and Computer Engineering

Carleton University
Ottawa, Ontario, Canada K1S 5B6
January 14, 2000

© copyright
2000, Ying Huang

**National Library
of Canada**

**Acquisitions and
Bibliographic Services**

**395 Wellington Street
Ottawa ON K1A 0N4
Canada**

**Bibliothèque nationale
du Canada**

**Acquisitions et
services bibliographiques**

**395, rue Wellington
Ottawa ON K1A 0N4
Canada**

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-66876-2

The undersigned recommend to the Faculty of Graduate Studies and Research
acceptance of the thesis

**Effects of Vocoder Distortion and Packet Loss on
Network Echo Cancellation**

submitted by

Ying Huang, B. Eng, M. Eng.

in partial fulfillment of the requirements for
the degree of Master of Engineering

Thesis Supervisor, Dr. R. A. Goubran

Chair, Department of Systems and Computer Engineering

Carleton University

January 14, 2000

Abstract

Voice services over asynchronous transfer mode (ATM) and Internet protocol (IP) networks are arousing keen interest in the communications industry. IP and ATM networks are packet-based networks and will have some effects on network echo cancellation. Among them are the effect of vocoder distortion and the effect of packet loss. In this thesis, we carry out simulations to investigate these two effects.

To study the effect of vocoder distortion, two standard vocoders, ITU Rec. G.723.1 and G.729, are used in our simulation models. These two low bit-rate vocoders are based on analysis-by-synthesis linear predictive coding and are the typical choices for speech coding in voice over ATM (VoATM) and voice over IP (VoIP). Three typical adaptive filtering algorithms, the Normalized Least Mean Square (NLMS) algorithm, the Fast Transversal Filters (FTF) algorithm, and the Fast Affine Projection (FAP) algorithm are implemented and used in our simulation. The simulation results show that the FAP algorithm outperforms the NLMS and FTF algorithms when there are vocoders in the echo path. The vocoder distortion is caused by several factors in the analysis-by-synthesis linear predictive coding, including the vector quantization of the line spectrum pairs, the formant post filtering, and the excitation modeling. Our simulation results show that the excitation modeling has the greatest effect on the performance of the echo canceller.

The effect of packet loss on network echo cancellation depends not only on the choice of adaptive filtering algorithms, but also on the choice of packet loss recovery

techniques. Three insertion-based recovery techniques are used to repair packet losses, namely silence substitution, noise substitution and repetition. The adaptive filtering algorithms adopted in the echo canceller are the NLMS, the Stabilized Fast Transversal Filter (SFTF), and the FAP. Among these three algorithms, we observe that the FAP algorithm demonstrates the best capability to depress the impact of packet loss on echo cancellation. A new noise substitution method is proposed which can significantly attenuate the effect of packet loss on echo cancellation as compared with the method which directly uses white noise as replacement.

Finally, we study the effect of vocoder distortion and effect of packet loss together to show their comprehensive effects. Simulation results show that the impact of packet loss on echo cancellation is smoothed by vocoders.

Acknowledgments

First and foremost, I would like to acknowledge and thank my supervisor, Dr. Rafik A. Goubran, for all his involvement, patience, criticisms and constant encouragement for my work for the whole duration of my program here.

I thank the Faculty of Graduate Studies and Research at Carleton University, the Department of Systems and Computer Engineering at Carleton University, Mitel Corporation, Communications and Information Technology Ontario (CITO) and Ontario Graduate Scholarship (OGS) for their financial support.

I thank all of my fellow graduate students in the DSP Lab for their helpful discussions and suggestions. Thanks also to Danny Lemay for always keeping the network of the DSP Lab operational.

To the Mitel staff, special thanks go to Dr. Dieter Schulz and Mirjana Popovic for their invaluable discussions and for providing the channel model and the source codes of G.723.1 and G.729.

Finally, I deeply acknowledge the constant support of my wife Ying Li.

Table of Contents

ABSTRACT	iii
ACKNOWLEDGMENTS	v
TABLE OF CONTENTS	vi
LIST OF FIGURES	viii
LIST OF TABLES	xi
 CHAPTER 1 INTRODUCTION	 1
1.1 NETWORK ECHO CANCELLATION	1
1.2 MOTIVATION	4
1.3 PROBLEM DEFINITION	5
1.3.1 Packet Voice Communication	5
1.3.2 Effect of Vocoder Distortion on Network Echo Cancellation	6
1.3.3 Effect of Packet Loss on Network Echo Cancellation	7
1.4 THESIS OUTLINE	8
 CHAPTER 2 BACKGROUND REVIEW	 10
2.1 INTRODUCTION	10
2.2 LMS AND NLMS ALGORITHMS	11
2.3 RLS ALGORITHM AND ITS FAST VERSIONS	16
2.3.1 Recursive Least-Squares (RLS) Algorithm	16
2.3.2 Fast Transversal Filters (FTF) Algorithm	19
2.4 FAST NEWTON TRANSVERSAL FILTERS	25
2.5 AFFINE PROJECTION ALGORITHMS	27
2.6 ANALYSIS-BY-SYNTHESIS LINEAR PREDICTIVE CODING	34
2.6.1 Principles	34
2.6.2 Introduction to G.723.1 and G.729	37

CHAPTER 3 EFFECT OF VOCODER DISTORTION ON NETWORK ECHO CANCELLATION	44
3.1 SIMULATION SYSTEM DESCRIPTION.....	45
3.2 SIMULATION RESULTS AND DISCUSSION	49
3.3 EFFECT OF VECTOR QUANTIZATION OF LPC PARAMETERS	54
3.4 EFFECT OF FORMANT POSTFILTERING.....	57
3.5 EFFECT OF EXCITATION MODELING.....	60
3.6 SIMULATIONS FOR G.729 CODEC	63
3.7 SUMMARY.....	65
CHAPTER 4 EFFECT OF PACKET LOSS ON NETWORK ECHO CANCELLATION	67
4.1 EFFECTS OF PACKET LOSS FOR DIFFERENT RECOVERY TECHNIQUES	67
4.1.1 Insertion-Based Recovery Techniques	68
4.1.2 Simulations and Discussion	69
4.2 PROPOSED INSERTION-BASED RECOVERY SCHEME	77
4.3 COMPARISON OF “PACKET LOSS (1)” AND “PACKET LOSS (2)”	82
4.4 SUMMARY.....	84
CHAPTER 5 COMBINATION OF EFFECT OF VOCODER DISTORTION AND EFFECT OF PACKET LOSS.....	85
5.1 SIMULATION SYSTEM	85
5.2 SIMULATION RESULTS AND DISCUSSION	87
CHAPTER 6 CONCLUSIONS.....	90
REFERENCES	94

LIST OF FIGURES

Figure 1.1 Generation of network echo	2
Figure 1.2 Echo suppressor	3
Figure 1.3 Echo cancellation in a telephone network	4
Figure 1.4 A configuration of VoIP	6
Figure 2.1 Transversal filter computation of RLS variables	20
Figure 2.2 Block diagram of conventional projection algorithm	28
Figure 2.3 Magnitude response of the IIR filter	32
Figure 2.4 Learning curves of NLMS and FAP	33
Figure 2.5 Discrete speech production model	35
Figure 2.6 Basic structure of analysis-by-synthesis linear predictive coding	36
Figure 2.7 Block diagram of the G.723.1 speech coder (From Reference [38])	39
Figure 2.8 Block diagram of the G.723.1 speech decoder (From Reference [38])	41
Figure 2.9 Block diagram of G.729 speech encoder (From Reference [39])	42
Figure 2.10 Block diagram of G.729 speech decoder (From Reference [39])	43
Figure 3.1 Simulation Model	45
Figure 3.2 Impulse response of channel model	46
Figure 3.3 Frequency response of band-pass filter	47
Figure 3.4 Speech excitation	48
Figure 3.5 Spectrogram of the speech input	49
Figure 3.6 ERLE performance for NLMS	50
Figure 3.7 ERLE performance for FTF	51
Figure 3.8 ERLE performance for FAP (N=5)	51
Figure 3.9 Comparative performance of NLMS, FTF and FAP in the vocoding case	52
Figure 3.10 Comparison of FAP for different projection orders	52
Figure 3.11 Comparative performance for different noise levels	54
Figure 3.12 ERLE performance for NLMS: VQ and no VQ cases	56
Figure 3.13 ERLE performance for FTF: VQ and no VQ cases	56
Figure 3.14 ERLE performance for FAP(N=5): VQ and no VQ cases	57
Figure 3.15 ERLE performance for NLMS: postfiltering and no postfiltering cases.	58
Figure 3.16 ERLE performance for FTF: postfiltering and no postfiltering cases.	59

Figure 3.17 ERLE performance for FAP(N=5): postfiltering and no postfiltering cases.	59
Figure 3.18 ERLE performance for NLMS: accurate and inaccurate excitation.	62
Figure 3.19 ERLE performance for FTF: accurate and inaccurate excitation.	62
Figure 3.20 ERLE performance for FAP: accurate and inaccurate excitation.....	63
Figure 3.21 Comparative performance for G.729 and G.723.1 (NLMS).....	64
Figure 3.22 Comparative performance for G.729 and G.723.1 (FTF).....	64
Figure 3.23 Comparative performance for G.729 and G.723.1 (FAP, N=5)	65
Figure 4.1 Packet loss simulation system for no vocoding case	69
Figure 4.2 Impact of packet loss on ERLE performance for NLMS and silence substitution	70
Figure 4.3 Impact of packet loss on ERLE performance for SFTF and silence substitution	71
Figure 4.4 Impact of packet loss on ERLE performance for FAP(N=5) and silence substitution	71
Figure 4.5 Impact of packet loss on ERLE performance for NLMS and repetition	72
Figure 4.6 Impact of packet loss on ERLE performance for SFTF and repetition	72
Figure 4.7 Impact of packet loss on ERLE performance for FAP(N=5) and repetition ...	73
Figure 4.8 Impact of packet loss on ERLE performance for NLMS and white noise substitution	73
Figure 4.9 Impact of packet loss on ERLE performance for SFTF and white noise substitution	74
Figure 4.10 Impact of packet loss on ERLE performance for FAP(N=5) and white noise substitution	74
Figure 4.11 Impact of packet loss on ERLE performance for FAP: N=2, 10, 20, silence substitution	75
Figure 4.12 Impact of packet loss on ERLE performance for FAP: N=2, 10, 20, repetition	76
Figure 4.13 Impact of packet loss on ERLE performance for FAP: N=2, 10, 20, white noise substitution.....	76
Figure 4.14 Block diagram of the new packet loss repair method	78

Figure 4.15 Comparative ERLE performance of “white noise” and “filtered noise” for NLMS.....	79
Figure 4.16 Comparative ERLE performance of “white noise” and “filtered noise” for SFTF.....	80
Figure 4.17 Comparative ERLE performance of “white noise” and “filtered noise” for FAP(N=5).....	80
Figure 4.18 Comparative ERLE performance of silence substitution, repetition and proposed method for NLMS	81
Figure 4.19 Comparative ERLE performance of silence substitution, repetition and proposed method for FAP (N=5)	81
Figure 4.20 Impact of packet loss for NLMS: comparative performance of “Packet loss (1)” and “Packet loss (2)”	82
Figure 4.21 Impact of packet loss for SFTF: comparative performance of “Packet loss (1)” and “Packet loss (2)”	83
Figure 4.22 Impact of packet loss for FAP(N=5): comparative performance of “Packet loss (1)” and “Packet loss (2)”	83
Figure 5.1 Simulation model for combined effects of vocoder distortion and packet loss	86
Figure 5.2 Simplified simulation model for combined effects of vocoder distortion and packet loss	87
Figure 5.3 Impact of packet loss on ERLE performance: with/without vocoders (NLMS: silence substitution).....	88
Figure 5.4 Impact of packet loss on ERLE performance: with/without vocoders (SFTF; silence substitution).....	89
Figure 5.5 Impact of packet loss on ERLE performance: with/without vocoders (FAP, N=5; silence substitution)	89

LIST OF TABLES

Table 2.1 LMS algorithm	13
Table 2.2 RLS algorithm	18
Table 2.3 FTF algorithm	22
Table 2.4 SFTF with improved tracking capability	24
Table 2.5 FNTF algorithm	27
Table 2.6 The regularized AP algorithm	29
Table 2.7 FAP algorithm	31
Table 2.8 Performance of adaptive filtering algorithms.....	34

Chapter 1

Introduction

1.1 Network Echo Cancellation

Echo is the phenomenon in which a delayed and possibly distorted version of the transmitted sound or signal is reflected back to the sender. Most telephone networks suffer from echoes due to impedance mismatches in their hybrids. These network echoes impair communication quality and need to be controlled effectively.

Figure 1.1 illustrates the generation of network echo. Every telephone set in a given area is connected to a local telephone central office (CO) by a two-wire line. This two-wire telephone circuit is called a local loop that carries the signals travelling in both directions simultaneously. In other words, it is a two-wire bidirectional channel. To connect a local CO to a remote CO, a four-wire circuit which consists of two unidirectional channels is usually used. This is because repeater amplifiers require separate circuits for the two directions respectively. Accordingly, we need a four-wire to two-wire coupling device called a *hybrid*. The single port on its two-wire side connects to a bidirectional transmission line, e.g. the local loop. There are two ports on the four-wire side: the input port and the output port. Ideally, the hybrid passes an incoming signal

from its input port to its “two-wire” port, and does not leak anything to its output port. However, impedance mismatch problems in the hybrid can cause imperfect separation of incoming and outgoing signals at the hybrid. As depicted in Figure 1.1, the input port of the hybrid becomes coupled to the output port, thereby giving rise to an echo.

Figure 1.1 Generation of network echo

A number of echo control methods are currently used in telephone networks. One solution is to control the echo by using an echo suppressor. An echo suppressor is essentially a voice-operated electronic switch that monitors and controls the voice signals travelling in both directions during a conversation, as shown in Figure 1.2. The suppressor detects which person is talking and blocks the signal travelling in the opposite direction. This prevents the echo from looping back to the speaker through the listener's transmission path. Thus the echo suppressor, inherently, allows only one-way communication (half duplex). While the echo suppressor is simple to implement, this solution clips the speech and impairs its quality. Furthermore, during double talk, i.e. when both parties talk simultaneously, echo suppressors become ineffective.

Figure 1.2 Echo suppressor

A more effective method is echo cancellation, as depicted in Figure 1.3. Typically, an echo canceller is an adaptive finite impulse response (FIR) filter. At a given end of the 4-wire trunk, the incoming signal is applied both to the hybrid and to the adaptive filter. The adaptive filter first attempts to estimate the impulse response of the echo path, and then generates a replica of the echo. The output of the adaptive filter is subtracted from the output of the hybrid to yield the error signal of the canceller. The error signal is used to adjust the tap weights of the adaptive filter. If the tap weights of the adaptive filter are selected such that its output is a synthesized copy of the echo signal, the echo can be effectively eliminated. Hence, the performance of the echo canceller depends on the choice of the adaptive filtering algorithm.

Figure 1.3 Echo cancellation in a telephone network

The idea of echo canceller was proposed in the mid-1960s for controlling long-distance telephone echoes [1]. In the 1970s, compact and economically feasible echo-canceller hardware became possible with advances in large-scale-integration and digital signal processor technologies. For example, Bell Laboratories announced a single-chip echo canceller in 1979 [2]. Echo cancellers have become a common method of controlling echoes in telephone systems and have been widely used in many applications [3].

1.2 Motivation

The transmission of compressed voice over packet networks, e.g. packet-switched IP networks and Asynchronous Transfer Mode (ATM) networks, has received considerable attention in recent years. Due to vocoder distortion and packet loss, voice transmission over packet networks exhibits non-linearity. The presence of vocoders and/or packet

losses in the echo path disturbs the operation of the echo cancellers, which are usually linear and cannot effectively model these two non-linear effects. Therefore, the performance of echo cancellers is degraded by these two effects.

Reviews of literature show that no attention has been paid to the effects of vocoder distortion and packet loss on network echo cancellation. However, we observe that due to these two effects, the achievable Echo Return Loss Enhancement (ERLE) of echo cancellers, defined as the ratio of the energy in the residual echo to the energy in the original echo, will be greatly decreased. Therefore, investigation of these effects is of significance and is helpful for designing a high-quality echo canceller for packet-based networks.

1.3 Problem Definition

1.3.1 Packet Voice Communication

Packet telephony is a newly developed cost-effective option for traditional circuit-based telephone networks [4]- [7]. To send packetized voice over data networks, such as asynchronous transfer mode (ATM) and Internet protocol (IP)-based networks, we need to convert analogue voice into digital samples. These digital samples must be coded using some form of compression, inserted into packets that have sequence numbers and creation timestamps, and then transported by the data network. On the receiver side, packetized voice is received and stored in a buffer, decoded in a sequential order, and played back. Vcoders (voice coders) are used to encode and decode the voice signals.

which allows significant reduction in the bandwidth needs. They play an important role in the packet telephony systems. The quality of voice service is also affected by bandwidth, packet delays and losses, etc.

IP networks are often combined with standard public switched telephone networks (PSTN). Figure 1.4 shows a possible configuration where the local access is PSTN and the long distance transmission is performed by an IP network. This thesis investigates the performance of echo cancellers in similar configurations when vocoders and/or packet loss are present in the echo path.

Figure 1.4 A configuration of VoIP

1.3.2 Effect of Vocoder Distortion on Network Echo Cancellation

Vocoders are speech-specific coders. Unlike waveform coders, which strive for facsimile reproduction of the signal waveform, vocoders rely on speech models and are focussed on producing perceptually intelligible speech without necessarily matching the

waveform. This means that vocoders will cause distortion on speech signals. In packet voice communications, if there are vocoders in the echo path, due to vocoder distortion, the echo generation process cannot be modeled exactly as a linear system. In other words, vocoders introduce non-linearity in the echo generation process. As a result, the performance of the echo canceller is degraded. In Chapter 4, we will use different adaptive algorithms to study the effects of vocoder distortion. The most popular class of vocoders for low and medium bit rates is the analysis-by-synthesis linear predictive coder. Several types of analysis-by-synthesis linear predictive coders have been adopted in telephony standards, such as G.723.1 and G.729, designed by the International Telecommunications Union (ITU). They are the main candidates for codecs used in voice over IP (VoIP) and voice over ATM (VoATM). We use both G.723.1 and G.729 codecs to carry out the experiments, and our discussion will focus on the first one.

1.3.3 Effect of Packet Loss on Network Echo Cancellation

When transmitting voice over a packet-based network, packet losses due to congestion or transmission errors in the network can produce significant impact on echo cancellers. The effects introduced by packet loss depend greatly on how a lost packet is recovered. Packet loss recovery techniques can be divided into two categories: sender-based repair and receiver-based repair [44][45]. The receiver-based repair is also called error concealment. The idea of error concealment is to generate a replacement for a lost packet which is similar to the original. Among error concealment techniques, the

insertion-based schemes are simple to implement and our studies will focus on methods belonging to this category.

1.4 Thesis Outline

This thesis is organized into 6 chapters.

Chapter 2 reviews the background materials, including adaptive filtering algorithms and analysis-by-synthesis linear predictive coding. For adaptive filtering algorithms, the scope of our review covers four families of algorithms. The first is the family of the least-mean-square (LMS) algorithms. We review the standard LMS and normalized LMS (NLMS). The second family belongs to the category of the least-squares (LS) algorithms. We discuss the recursive least-squares (RLS) algorithm and its fast version: the fast transversal filters (FTF) algorithm. The third is the fast Newton transversal filter (FNTF) which is an approximate RLS algorithm. The last one is the family of the affine projection (AP) algorithms. A brief introduction to the AP algorithm is presented, and the fast version of AP (FAP) proposed by S. Gay is discussed in detail [26][27]. For analysis-by-synthesis linear predictive coding, we introduce the principles and two standard vocoders (G.723.1 & G.729) [38][39] based on this method. We concentrate particularly on analysis-by-synthesis linear predictive coding because this technique has been the basis for most of the high quality speech coding standards created during the past decade.

In Chapter 3, the effect of vocoder distortion is studied. First, a simulation model with embedded G.723.1 or G.729 vocoders is established. Then, the NLMS, FTF and FAP algorithms are implemented to carry out the simulations. Simulation results for the

G.723.1 and G.729 vocoders are presented. We also look into the internal structure of vocoders to investigate effects of several processing methods adopted in the analysis-by-synthesis linear predictive coding. These methods include the vector quantization (VQ) of LPC parameters, the formant postfiltering, and the excitation modeling.

In Chapter 4, the effect of packet loss is studied. The effect of packet loss depends not only on the choice of the adaptive filtering algorithm but also on the choice of the packet loss recovery technique. For the adaptive filter, the NLMS, the stabilized FTF (SFTF) and the FAP algorithms are implemented. For the packet loss recovery technique, we concentrate our study on the insertion-based repair schemes, including silence substitution, noise substitution and repetition. A new noise substitution method is proposed to reduce the impact of packet loss on the performance of echo cancellers.

In Chapter 5, simulations are carried out under the situation where both vocoders and packet losses are present in the echo path. A simulation model is proposed and simplified to represent this situation.

Chapter 6 concludes the thesis and proposes some suggestions for further research.

Chapter 2

Background Review

2.1 Introduction

In this chapter, we present background material for our studies. The chapter consists of two parts. In the first part, we review several adaptive filtering algorithms commonly used in echo cancellation [8][9]. In the second part, we introduce analysis-by-synthesis linear predictive coding, including two standard coders (G.723.1 & G.729).

The theoretical basis for echo cancellers lies in the field of adaptive filtering. Adaptive filters generally consist of two distinct parts: a filter, whose structure is designed to perform a desired processing function, and an adaptive algorithm for adjusting the parameters of that filter. The many possible combinations of filter structures and the adaptive laws governing them lead to a variety of adaptive filters.

There are two basic families of algorithms: the least-mean-square (LMS) algorithm and the recursive-least-squares (RLS) algorithm. They are considered as typical of the two ends of the range of convergence speed that can be achieved for echo cancellation. A significant feature of the LMS algorithm is its simplicity, but it suffers from low

convergence speed. RLS has high complexity $O(L^2)$ with L being the filter length. The fast RLS (FRLS) algorithms reduce the complexity to $O(L)$. Further reduction of complexity can be achieved by adjusting the order of the predictor, which is L in FRLS, to a much lower order. This leads to the fast Newton transversal filters (FNTF) which balance the trade-off between the LMS and FRLS regarding performance and complexity. The family of affine projection algorithms lies between the NLMS and block RLS algorithms. The fast affine projection (FAP) algorithm also provides a large choice of tradeoffs between computational complexity and convergence rate.

The main objective in speech coding is to represent speech with a minimum number of bits while maintaining its perceptual quality. In the application of packet voice communications, low bit-rate speech coders are required. Low bit-rate speech coding can be achieved using the linear prediction based analysis-by-synthesis paradigm. The analysis-by-synthesis linear predictive coding is a hybrid method that is a meld of techniques from both waveform and linear predictive coding (LPC). In fact, this coding method provides state-of-the-art performance for bit rates in the range between 4 and 16 kb/s [36], and is widely adopted in packet voice communications.

2.2 LMS and NLMS algorithms

The LMS and related algorithms are widely used for echo cancellation. The original LMS algorithm was proposed by Widrow and Hoff in [11]. Being a gradient-based algorithm, it provides a solution to the optimal Wiener filter criterion, minimizing the mean square value of the filtering error in a stochastic approximation sense. Its simplicity

and ease of implementation made this algorithm an attractive solution for many practical problems.

The LMS algorithm can be derived from the steepest descent method. The steepest descent method is an optimization technique that uses the gradients of the surface to seek its minimum. It uses an iterative procedure to locate the optimum point, which is similar in concept to the qualitative approach of “rolling the marble towards the bottom”.

Let $J(n)$ denote the cost function which is chosen as the mean-square value of the estimation error. According to the method of steepest descent, the $(n+1)$ th tap-weight vector is the n th tap-weight vector plus a correction in the direction of the $J(n)$'s gradient. That is,

$$\underline{h}(n+1) = \underline{h}(n) + \frac{1}{2} \mu [-\nabla J(n)] \quad (2.1)$$

where $\underline{h}(n) = [h_0(n) \ h_1(n) \ \dots \ h_{L-1}(n)]^T$ is a tap-weight vector, μ is a positive real-valued factor that controls stability and rate of adaptation. The gradient of $J(n)$ is given by

$$\nabla J(n) = -2\mathbf{p} + 2\mathbf{R}\underline{h}(n) \quad (2.2)$$

where

$$\mathbf{R} = E[\underline{x}(n)\underline{x}^H(n)] \quad (2.3)$$

$$\mathbf{p} = E[\underline{x}(n)d^*(n)] \quad (2.4)$$

Here, $d(n)$ is the desired response and $\underline{x}(n) = [x(n), x(n-1), \dots, x(n-L+1)]^T$ is a tap-input vector. Substituting Eq.(2.2) into (2.1), we obtain the recursive relation

$$\underline{h}(n+1) = \underline{h}(n) + \mu[\mathbf{p} - \mathbf{R}\underline{h}(n)] \quad (2.5)$$

Since the parameter μ controls the size of the incremental correction applied to the tap-weight vector, we call it the step-size parameter.

In the steepest-descent algorithm, to calculate the gradient vector requires prior knowledge of both the correlation matrix $\mathbf{R}$ and the cross-correlation vector $\mathbf{p}$. In most applications, only an estimate of these second-order statistics can be obtained. In order to implement the method of steepest descent, the gradient must be estimated from the available data. Let the estimates of the $\mathbf{R}$ and the $\mathbf{p}$ be given simply as

$$\hat{\mathbf{R}}(n) = \underline{x}(n)\underline{x}^H(n) \quad (2.6)$$

and

$$\hat{\mathbf{p}} = \underline{x}(n)d^*(n) \quad (2.7)$$

Then, the instantaneous estimate of the gradient is

$$\hat{\nabla}J(n) = -2\underline{x}(n)d^*(n) + 2\underline{x}(n)\underline{x}^H(n)\underline{h}(n) \quad (2.8)$$

Substitution of Eq.(2.8) into (2.1) leads to the LMS algorithm:

$$\underline{h}(n+1) = \underline{h}(n) + \mu\underline{x}(n)[d^*(n) - \underline{x}^H(n)\underline{h}(n)] \quad (2.9)$$

Equivalently, the standard LMS algorithm may be written as follows:

Table 2.1 LMS algorithm

$e(n) = d(n) - \underline{h}(n)^T \underline{x}(n)$ $\underline{h}(n+1) = \underline{h}(n) + \mu\underline{x}(n)e^*(n)$

where $e(n)$ is the estimation error. The step-size parameter μ controls stability and rate of adaptation. The LMS algorithm is convergent if and only if [12]

$$0 < \mu < \frac{2}{\lambda_{\max}} \quad (2.10)$$

The tap-weight vector $\underline{h}(n)$ does not converge exactly to the minimum point of the error-performance surface. Instead, $\underline{h}(n)$ exhibits a random motion around the minimum point due to the presence of gradient noise. This results in a mean-squared error $J(n)$ that is always greater than the minimum mean-squared error $J_{\min}$ pertaining to the Wiener solution. The difference between $J(n)$ and $J_{\min}$ is defined as the *excess mean-squared error*. Denoting the steady-state value of the excess mean-squared error by $J_{\text{ex}}(\infty)$, we define the *misadjustment* as the ratio of $J_{\text{ex}}(\infty)$ to $J_{\min}$. The misadjustment ψ is approximately given by [13]

$$\psi \approx \frac{\mu L \lambda_{\min}}{2} \quad (2.11)$$

From Eq.(2.11), we observe that the misadjustment of the LMS algorithm increases linearly with the filter length L and the step-size parameter μ .

In the standard LMS algorithm described above, when $\underline{x}(n)$ is large, there will be a gradient noise amplification problem. This is because the correction $\mu \underline{x}(n) e^*(n)$ applied to $\underline{h}(n)$ is directly proportional to $\underline{x}(n)$. To solve this problem, we may use the normalized LMS (NLMS) algorithm which uses the following iteration:

$$\underline{h}(n+1) = \underline{h}(n) + \frac{\mu}{a + \|\underline{x}(n)\|^2} \underline{x}(n) e^*(n) \quad (2.12)$$

where the positive constant α is introduced to avoid division by a small value. We may also view the NLMS as a time-varying step-size LMS.

The necessary and sufficient condition for the convergence of the NLMS algorithm is [12]

$$0 < \mu < 2 \quad (2.13)$$

The fastest convergence occurs when $\mu=1$ [15]. From Eq. (2.13), we see that the convergent condition of the NLMS algorithm is independent of the correlation statistics of the input data. It has been indicated that the convergence rate of the NLMS algorithm is potentially faster than that of the standard LMS algorithm [14]-[16].

The main advantage of NLMS is its computational simplicity (of the order of $2L$ multiply-accumulate operations per iteration). Furthermore, it is numerically stable. Its major drawback is that the convergence properties degrade with coloured signal input such as speech.

2.3 RLS Algorithm and Its Fast Versions

The LMS algorithm uses the actual data to compute predictions and update their coefficients. This leads to the main deficiency of this method: they have been derived using a statistical error measure, whereas their implementation utilizes the exact data, available in the specific application. Therefore, it is quite logical to consider adaptive filters based upon error measures derived from the exact data acquired. This approach leads to the family of extremely powerful adaptive signal processing techniques known collectively as the recursive-least-squares (RLS) algorithms.

2.3.1 Recursive Least-Squares (RLS) Algorithm

In the exponentially weighted recursive-least-squares (RLS) algorithm, the cost function to be minimized is chosen as:

$$\xi(n) = \sum_{i=1}^n \lambda^{n-i} |e(i)|^2 \quad (2.14)$$

where λ is the exponential weighting factor or forgetting factor. From the principle of orthogonality based on time average, we may derive the following normal equations:

$$\Phi(n) \underline{h}(n) = \underline{z}(n) \quad (2.15)$$

where

$$\Phi(n) = \sum_{i=1}^n \lambda^{n-i} \underline{x}(i) \underline{x}^H(i) \quad (2.16)$$

$$\mathbf{z}(n) = \sum_{i=1}^n \lambda^{n-i} \underline{x}(i) d^*(i) \quad (2.17)$$

Actually, $\Phi(n)$ represents the time-averaged autocorrelation matrix of the tap inputs, and $\mathbf{z}(n)$ represents the cross-correlation vector between the tap inputs and the desired response. The recursions for updating $\Phi(n)$ and $\mathbf{z}(n)$ are given by

$$\Phi(n) = \lambda \Phi(n-1) + \underline{x}(n) \underline{x}^H(n) \quad (2.18)$$

$$\mathbf{z}(n) = \lambda \mathbf{z}(n-1) + \underline{x}(n) d^*(n) \quad (2.19)$$

To solve for the least-square estimate $\hat{h}(n)$ in Eq.(2.15), we need to compute the inverse of the correlation matrix $\Phi(n)$. However, it is too complex to compute the inverse of $\Phi(n)$ in practical applications, especially when the order L is high. In order to avoid this operation, we use the *matrix inversion lemma* to derive the recursive form for the inverse of $\Phi(n)$ and obtain

$$\Phi^{-1}(n) = \lambda^{-1} \Phi^{-1}(n-1) - \frac{\lambda^{-2} \Phi^{-1}(n-1) \underline{x}(n) \underline{x}^H(n) \Phi^{-1}(n-1)}{1 + \lambda^{-1} \underline{x}^H(n) \Phi^{-1}(n-1) \underline{x}(n)} \quad (2.20)$$

For convenience of computation, the above equation may be rewritten as

$$\mathbf{P}(n) = \lambda^{-1} \mathbf{P}(n-1) - \lambda^{-1} \mathbf{k}(n) \underline{x}^H(n) \mathbf{P}(n-1) \quad (2.21)$$

where we define

$$\mathbf{P}(n) = \Phi^{-1}(n) \quad (2.22)$$

$$\mathbf{k}(n) = \frac{\lambda^{-1} \mathbf{P}(n-1) \underline{x}(n)}{1 + \lambda^{-1} \underline{x}^H(n) \mathbf{P}(n-1) \underline{x}(n)} \quad (2.23)$$

Using Eq.(2.15), (2.19), (2.21), and (2.22), we obtain the recursive equation for updating the tap-weight vector as follows:

$$\underline{h}(n) = \underline{h}(n-1) + \underline{k}(n)\underline{\varepsilon}^*(n) \quad (2.24)$$

where $\underline{\varepsilon}(n)$ is the *a priori* estimation error defined by

$$\underline{\varepsilon}(n) = d(n) - \underline{h}^H(n-1)\underline{x}(n) \quad (2.25)$$

The RLS algorithm is summarized in Table 2.2 [13].

Table 2.2 RLS algorithm

<p><i>Riccati Equation :</i></p> $\underline{k}(n) = \frac{\lambda^{-1}\underline{\mathbf{P}}(n-1)\underline{x}(n)}{1 + \lambda^{-1}\underline{x}^H(n)\underline{\mathbf{P}}(n-1)\underline{x}(n)}$ $\underline{\mathbf{P}}(n) = \lambda^{-1}\underline{\mathbf{P}}(n-1) - \lambda^{-1}\underline{k}(n)\underline{x}^H(n)\underline{\mathbf{P}}(n-1)$ <p><i>Filtering :</i></p> $\underline{\varepsilon}(n) = d(n) - \underline{h}^H(n-1)\underline{x}(n)$ $\underline{h}(n) = \underline{h}(n-1) + \underline{k}(n)\underline{\varepsilon}^*(n)$

The inverse correlation matrix $\underline{\mathbf{P}}(n)$ is initialized by setting

$$\underline{\mathbf{P}}(0) = \delta^{-1}\underline{\mathbf{I}} \quad (2.26)$$

where δ is a small positive constant, and $\underline{\mathbf{I}}$ is an identity matrix.

The RLS algorithm described above has zero excess mean-squared error. The RLS algorithm converges in about $2L$ iterations. This means that the RLS algorithm typically converges faster than the LMS algorithm. It can be shown that the convergence behavior of the RLS algorithm is independent of the spectral characteristics of the input signal.

The RLS algorithm performs well in many situations. The main reason that limits the usefulness of this algorithm in some applications is its computational complexity which increases in proportion to the square of the filter order. This is often unacceptable for high-order filters. To solve this problem, several fast versions of the recursive least-squares algorithms, namely the fast Kalman [17], the fast *a posteriori* error sequential technique (FAEST) [18], and the fast transversal filters (FTF) [19] algorithms have been proposed. They lead to a computational complexity of the order of $O(L)$. In the following part, we will discuss the FTF algorithm briefly.

2.3.2 Fast Transversal Filters (FTF) Algorithm

In the FTF algorithm, the propagation of the inverse correlation matrix $\mathbf{P}(n)$ is replaced by the recursive computation of a ‘Kalman gain’ within a prediction framework. This algorithm uses four transversal filters to compute all quantities of interest in the RLS problem. One filter is the adaptive filter whose tap values are the desired parameter values in system identification. This filter also appears in the LMS algorithm. The other three filters are called auxiliary filters. They perform forward linear prediction, backward linear prediction, and computation of the gain vector, respectively. Here we denote the tap-weight vectors of these three filters at time n , with filter length L , as $A_L(n)$, $B_L(n)$ and $C_L(n)$. Figure 2.1 shows the transversal filter computation of RLS variables. $e_L(n)$, $r_L(n)$, $\hat{u}_L(n)$ and $\hat{\varepsilon}_L(n)$ are the filtered errors. $e_L^p(n)$, $r_L^p(n)$ and $\varepsilon_L^p(n)$ are the predicted errors. The residual powers are denoted as $\alpha_L(n)$, $\beta_L(n)$ and $\xi_L(n)$. The detailed description can be found in [19].

Figure 2.1 Transversal filter computation of RLS variables

The basic form of the FTF algorithm using prewindowing [19] is recalled in Table

2.3. $\tilde{C}_L(n)$ is the normalized gain vector, defined as

$$\tilde{C}_L(n) = \frac{C_L(n)}{\gamma_L(n)} \quad (2.27)$$

$\tilde{C}_{L+1}^{L+1}(n)$ is the $(L+1)$ th component of $\tilde{C}_{L+1}(n)$. The factor μ is used for soft constrained initialization. The per-iteration computational requirement of this FTF algorithm is $7L$ multiply-accumulate operations.

A serious limitation of the FTF algorithm is that it suffers from numerical instabilities. This problem arises when the FTF algorithm is implemented on a machine with finite word-length. The rescue variable described in Table 2.3 is introduced to solve

this problem. If the rescue variable is negative, $\underline{h}(n)$ is saved as the initial condition to restart the algorithm. Thus, it is a quantity indicator of the level of the numerical errors. We observe it until a threshold is reached, at which point the algorithm is restarted. Even though the restart can include the current solution as prior information by means of a soft-constrained initialization, such a restart leads to sub-optimal performance because of a significant discontinuity in the sample covariance matrix in the normal equations that determine the solution.

Table 2.3 FTF algorithm**Initialization**

$$A_L(-1) = [1 \ 0, \dots, 0], \ B_L(-1) = [0, \dots, 0 \ 1], \ \tilde{C}_L(-1) = [0, \dots, 0]$$

$$\underline{h}(-1) = \underline{h}_0, \ \alpha_L(-1) = \lambda^L \mu, \ \beta_L(-1) = \mu, \ \gamma_L(-1) = 1$$

Predictions :

$$\mathbf{e}_L^P(n) = A_L(n-1)\underline{x}_{L+1}(n)$$

$$e_L(n) = e_L^P(n)\gamma_L(n-1)$$

$$\alpha_L(n) = \lambda\alpha_L(n-1) + e_L^P(n)e_L(n)$$

$$\gamma_{L+1}(n) = \frac{\lambda\alpha_L(n-1)}{\alpha_L(n)}\gamma_L(n-1)$$

$$\tilde{C}_{L+1}(n) = [0 \ \tilde{C}_L(n-1)] - \lambda^{-1}e_L^P(n)\alpha_L^{-1}(n-1)A_L(n-1)$$

$$A_L(n) = A_L(n-1) + e_L(n)[0 \ \tilde{C}_L(n-1)]$$

$$r_L^P(n) = -\lambda\beta_L(n-1)\tilde{C}_{L+1}^{L-1}(n)$$

$$\gamma_L(n) = [1 + r_L^P(n)\gamma_{L+1}(n)\tilde{C}_{L+1}^{L-1}(n)]^{-1}\gamma_{L+1}(n)$$

$$\text{Rescue variable} = [1 + r_L^P(n)\gamma_{L+1}(n)\tilde{C}_{L+1}^{L-1}(n)]$$

$$r_L(n) = r_L^P(n)\gamma_L(n)$$

$$\beta_L(n) = \lambda\beta_L(n-1) + r_L^P(n)r_L(n)$$

$$[\tilde{C}_L(n) \ 0] = \tilde{C}_{L+1}(n) - \tilde{C}_{L+1}^{L-1}(n)B_L(n-1)$$

$$B_L(n) = B_L(n-1) + r_L(n)[\tilde{C}_L(n) \ 0]$$

Filtering :

$$\varepsilon_L^P(n) = d(n) - \underline{h}^T(n-1)\underline{x}(n)$$

$$\varepsilon_L(n) = \varepsilon_L^P(n)\gamma_L(n)$$

$$\underline{h}(n) = \underline{h}(n-1) - \varepsilon_L(n)\tilde{C}_L(n)$$

Stabilized versions of FTF (SFTF) have been proposed in [20][21]. They include a control of the propagation of numerical errors; hence stable operation is obtained when the algorithm is implemented in finite precision. The additional complexity is of the order of L multiplications and additions per-iteration. A modified SFTF with improved tracking capability has been proposed in [22]. The basic idea under this algorithm is that the forgetting factor for the adaptation of the filtering part is modified to be smaller, while the forgetting factor in the prediction part is kept to a value which ensures the numerical stability. The modified STFT is recalled in Table 2.4.

In the algorithm described in Table 2.4, the parameters μ^y and μ^p are used to control the propagation of numerical errors. If we choose $\mu^y = \mu^p = 1$, the algorithm is numerically stable under the following condition:

$$\lambda > 1 - 1/(2L + 3.5) \quad (2.28)$$

The parameter ρ is introduced to control the forgetting factor in the filtering part, which is given by:

$$\lambda_H = \frac{(1 - \rho)\lambda}{1 - \lambda\rho} \quad (2.29)$$

The parameter ρ must be less than or equal to one. If $\rho = 0$, we get $\lambda_H = \lambda$, and the modified algorithm is equivalent to the normal SFTF algorithm.

Table 2.4 SFTF with improved tracking capability**Initialization**

$$A_L(-1) = [1 \ 0, \dots, 0], \ B_L(-1) = [0, \dots, 0 \ 1], \ C_L(-1) = [0, \dots, 0]$$

$$\underline{h}(-1) = \underline{h}_0, \ \alpha_L(-1) = \lambda^L \mu, \ \beta_L(-1) = \mu, \ \gamma_L(-1) = 1$$

Predictions :

$$\begin{aligned} \mathbf{e}_L^P(n) &= A_L(n-1) \underline{x}_{L+1}(n) \\ e_L(n) &= e_L^P(n) \gamma_L(n-1) \\ \alpha_L(n) &= \lambda \alpha_L(n-1) + e_L^P(n) e_L(n) \\ \gamma_{L+1}(n) &= \frac{\lambda \alpha_L(n-1)}{\alpha_L(n)} \gamma_L(n-1) \\ \tilde{C}_{L+1}(n) &= [0 \ \tilde{C}_L(n-1)] - \lambda^{-1} e_L^P(n) \alpha_L^{-1}(n-1) A_L(n-1) \\ A_L(n) &= A_L(n-1) + e_L(n) [0 \ \tilde{C}_L(n-1)] \\ r_L^c(n) &= B_L(n-1) \underline{x}_{L+1}(n) \\ \xi_L(n) &= r_L^c(n) + \lambda^{-L+1} \alpha_L(n-1) \gamma_L(n-1) \tilde{C}_{L+1}^{L+1}(n) \\ r_L^z(n) &= r_L^c(n) + \mu^z \xi_L(n) \\ r_L^b(n) &= r_L^c(n) + \mu^b \xi_L(n) \\ \gamma_L(n) &= [1 + r_L^z(n) \gamma_{L+1}(n) \tilde{C}_{L+1}^{L+1}(n)]^{-1} \gamma_{L+1}(n) \\ [\tilde{C}_L(n) \ 0] &= \tilde{C}_{L+1}(n) - \tilde{C}_{L+1}^{L+1}(n) B_L(n-1) \\ B_L(n) &= B_L(n-1) + r_L^b(n) \gamma_L(n) [\tilde{C}_L(n) \ 0] \end{aligned}$$

Filtering :

$$\begin{aligned} \varepsilon_L^P(n) &= d(n) - \underline{h}^T(n-1) \underline{x}(n) \\ \varepsilon_L(n) &= \varepsilon_L^P(n) \gamma_L(n) \\ \underline{h}(n) &= \underline{h}(n-1) - \varepsilon_L(n) \tilde{C}_L(n) / (1 - \rho \gamma_L(n)) \end{aligned}$$

2.4 Fast Newton Transversal Filters

For some applications, the fast RLS algorithms are disqualified because of their computational complexity. To solve this problem, a new class of Newton-type estimation algorithms known as fast Newton transversal filters (FNTF) has been proposed in [23].

Newton algorithms are a very general class of adaptive algorithms unifying LMS and RLS. The class can be defined as follows:

$$C_L(n) = -Q_L(n)\underline{x}(n) \quad (2.30)$$

$$\varepsilon(n) = d(n) - \underline{h}^T(n-1)\underline{x}(n) \quad (2.31)$$

$$\underline{h}(n) = \underline{h}(n-1) - \mu\varepsilon(n)C_L(n) \quad (2.32)$$

where $C_L(n)$ is the gain vector. Notice that Eq.(2.31) and Eq.(2.32) are common to all algorithms belonging to this class and are known as the “filtering part” of the algorithm. In order to compute this part we have to perform $2L$ multiply-accumulate operations per iteration, which is the lower bound for the computational complexity of any algorithm in this class. The part that discriminates the algorithms is the definition of the matrix $Q_L(n)$. If $Q_L(n)=I_L$ (I_L is a $L \times L$ identity matrix), the resulting algorithm is LMS. If $\mu=1$ and $Q_L(n)=R_L^{-1}(n)$ with $R_L(n)$ being the covariance matrix of the sample sequence $\underline{x}(n)$, the resulting algorithm is the RLS. Thus, the LMS and RLS are two special cases in this class.

We have seen that $Q_L(n)$ is selected as the inverse covariance matrix of the input sequence in RLS. If the input sequence is modeled as white noise, we can expect that $Q_L(n)$ is a diagonal matrix and we obtain LMS. The FNTF is derived by generalizing this idea. Instead of modeling the input sequence as white noise, we assume that the input

sequence can be modeled as an AR process of order N with $0 \leq N \leq L$. If N takes values 0 and L , we obtain the LMS and RLS as two extremes. In the FNTF, the predictable information of the input sequence is extracted using predictors of order N , and the estimation of the covariance matrix of order $L \times L$ is extrapolated from a lower order $N \times N$ estimate. The FNTF using forward and backward predictors is summarized in Table 2.5. $A_N(n)$, $B_N(n)$ are the tap-weight vectors of the forward and backward predictors and $e_N^f(n)$, $e_N^b(n)$ are their corresponding prediction errors. $\underline{0}_{L-N}$ is a zero vector of size $(L-N)$. Two sets of the FNTF prediction variables are needed: one computed at time n and the other at time $k=n-L+N$. $S_{N+1}(n)$ and $U_{N+1}(n)$ are defined as follows:

$$S_{N+1}(n) = \frac{e_N^f(n)}{\lambda \alpha_N(n-1)} \begin{bmatrix} 1 \\ -A_N(n-1) \end{bmatrix} \quad (2.33)$$

$$U_{N+1}(n) = \frac{e_N^b(n)}{\lambda \beta_N(n-1)} \begin{bmatrix} -B_N(n-1) \\ 1 \end{bmatrix} \quad (2.34)$$

where $\alpha_N(n)$ and $\beta_N(n)$ are the forward and backward residual powers, and λ is the forgetting factor.

If we use the fast versions of RLS to solve prediction problems of order N , the FNTF will have complexity $2L+(5\sim 8)N$. In the applications of echo cancellation, since the input speech can be modeled as an AR process of order 8~20, while the filter size L is much larger, the complexity of FNTF will be LMS-like.

The simplified fast Newton algorithm was developed by Mavridis *et al.* [24]. Instead of using the fast RLS schemes to calculate the predictors, this algorithm uses the LMS algorithm to obtain the coefficients of the predictors. As a result, the complexity is reduced to $2L+3N$ multiplications and $2L+4N$ additions per iteration. Furthermore, its

overall algorithmic structure is simple (LMS type) and it does not have robustness problems due to finite precision.

Table 2.5 FNTF algorithm

Available from any least - squares algorithm: $S_{N+1}(n), e_N^f(n), U_{N+1}(k), e_N^b(k)$

$$C_{L+1}(n) = \begin{bmatrix} 0 \\ C_L(n-1) \end{bmatrix} - \begin{bmatrix} S_{N+1}(n) \\ \underline{0}_{L-N} \end{bmatrix}$$

$$\begin{bmatrix} C_L(n) \\ 0 \end{bmatrix} = C_{L+1}(n) + \begin{bmatrix} \underline{0}_{L-N} \\ U_{N+1}(k) \end{bmatrix}$$

$$\gamma_{L,N}(n) = \gamma_{L,N}(n-1) + S_{N+1}^1(n) e_N^f(n) - U_{N+1}^{N+1}(k) e_N^b(k)$$

Filtering:

$$\varepsilon(n) = d(n) - \underline{h}^T(n-1) \underline{x}(n)$$

$$e(n) = \varepsilon(n) / \gamma_{L,N}(n)$$

$$\underline{h}(n) = \underline{h}(n-1) - e(n) C_L(n)$$

2.5 Affine Projection Algorithms

The affine projection (AP) algorithm [25] can be considered as a generalization of the NLMS algorithm. Under the affine projection interpretation, each tap update in the NLMS is viewed as a one-dimensional projection [26]. The general AP algorithm is based on a multiple dimension projection. The algorithm applies update directions that are orthogonal to the last N input vectors, and thus decorrelates the input signal.

Therefore, convergence can be improved for a highly correlated input signal, such as speech. A block diagram of the conventional projection algorithm is shown in Figure 2.2 [31]. It consists of four parts: calculation of the decorrelation filter coefficients, decorrelation of the input signal, convolution of $x(n)$ and $\underline{h}(n)$ to generate an estimate value of $d(n)$, and adjustment of the filter $\underline{h}(n)$.

The tap-weight vector $\underline{h}(n)$ is adjusted as follows:

$$\underline{h}(n) = \underline{h}(n-1) + \mu \mathbf{X}(n) \underline{e}(n) \quad (2.35)$$

Here, μ and $\underline{e}(n)$ denote the step size and the tap-weight vector of the decorrelation filter.

Figure 2.2 Block diagram of conventional projection algorithm

$\mathbf{X}(n)$ is a L -by- N matrix defined as

$$\mathbf{X}(n) = [\underline{x}(n), \underline{x}(n-1), \dots, \underline{x}(n-N+1)] \quad (2.36)$$

$$\underline{x}(n) = [x(n), x(n-1), \dots, x(n-L+1)]^T \quad (2.37)$$

where N is the projection order, and L is the filter length. In the basic AP algorithm, the decorrelation filter vector $\underline{g}(n)$ is calculated as

$$\underline{g}(n) = [\mathbf{X}(n)^T \mathbf{X}(n)]^{-1} \underline{e}(n) \quad (2.38)$$

The basic AP algorithm exhibits a large residual error if the input signal is non-stationary and if some noise is added to the reference signal. This is due to the fact that the noise gets amplified when the covariance matrix $\mathbf{X}(n)^T \mathbf{X}(n)$ is ill-conditioned. To solve this problem, the AP algorithm is regularized by using the following equation to calculate the $\underline{g}(n)$:

$$\underline{g}(n) = [\mathbf{X}(n)^T \mathbf{X}(n) + \delta \mathbf{I}]^{-1} \underline{e}(n) \quad (2.39)$$

where δ is the regularization parameter. In Eq. (2.39), it is easily seen that, even if $\mathbf{X}(n)^T \mathbf{X}(n)$ has eigenvalues close to zero, $\mathbf{X}(n)^T \mathbf{X}(n) + \delta \mathbf{I}$ has δ as its smallest eigenvalue. Hence, if δ is large enough, it yields a better conditioned inverse. The regularized AP algorithm is summarized in Table 2.6.

Table 2.6 The regularized AP algorithm

$\underline{e}(n) = \underline{d}(n) - \mathbf{X}(n)^T \underline{h}(n-1)$ $\underline{g}(n) = [\mathbf{X}(n)^T \mathbf{X}(n) + \delta \mathbf{I}]^{-1} \underline{e}(n)$ $\underline{h}(n) = \underline{h}(n-1) + \mu \mathbf{X}(n) \underline{g}(n)$
--

As in the NLMS algorithm, this algorithm is stable for $0 < \mu < 2$.

The basic AP algorithm involves high computational complexity because of the matrix inversion, especially when the projection order is high. The fast version of the AP (FAP) algorithm has been proposed by Gay [26][27]. For the FAP algorithm, computational complexity has been reduced to $2L+20N$. If L is much larger than N , which is usually the case in echo cancellation, FAP has an LMS-like complexity. If step-size parameter μ is set to one, the computational complexity can be further reduced to $2L+14N$. The FAP algorithm is listed in Table 2.7.

In Table 2.7, the optimum forward and backward linear predictors $A_N(n)$ and $B_N(n)$, and their respective prediction error energies $\alpha(n)$ and $\beta(n)$, are obtained using the sliding windowed FTF [28]. A vector with bar hat is an $N-1$ length vector containing the $N-1$ upper elements of the original one. $\tilde{\underline{x}}(n)$ is defined as

$$\tilde{\underline{x}}(n) = [x(n-1), x(n-2), \dots, x(n-N+1)]^T \quad (2.40)$$

$E_{N-1}(n)$ is the $(N-1)$ th element in $\underline{E}(n)$. The filter $\hat{\underline{h}}(n)$ is an approximation of the exact filter coefficient vector $\underline{h}(n)$. They hold the following relationship:

$$\underline{h}(n) = \hat{\underline{h}}(n) + \mu \bar{\underline{X}}(n) \bar{\underline{E}}(n) \quad (2.41)$$

The sliding-window FTF algorithm embedded within the FAP algorithm can be too complex to implement in some cases. The approximate FAP algorithm based on the discrete cosine transform (DCT) has been proposed in [29]. Simulations show that the performance of this simplified algorithm is nearly as good as the original one when the orthogonal transform is chosen properly.

Table 2.7 FAP algorithm

$$\begin{aligned}
&\text{Initialization: } \alpha(0) = \beta(0) = \delta, \quad A_N(0) = [1, \mathbf{0}^T]^T, \quad B_N(0) = [\mathbf{0}^T, 1]^T \\
&\text{Use sliding windowed FTF [28] to update } \alpha(n), \beta(n), A_N(n), B_N(n) \\
&\tilde{\mathbf{r}}_{xx}(n) = \tilde{\mathbf{r}}_{xx}(n-1) + x(n)\tilde{\mathbf{x}}(n) - x(n-L)\tilde{\mathbf{x}}(n-L) \\
&\mathbf{e}(n) = d(n) - \mathbf{x}(n)^T \hat{\mathbf{h}}(n-1) - \mu \tilde{\mathbf{r}}_{xx}^T(n) \bar{\mathbf{E}}(n-1) \\
&\underline{\mathbf{e}}(n) = \begin{bmatrix} \mathbf{e}(n) \\ (1-\mu)\bar{\mathbf{e}}(n-1) \end{bmatrix} \\
&\underline{\boldsymbol{\varepsilon}}(n) = \begin{bmatrix} 0 \\ \tilde{\boldsymbol{\varepsilon}}(n) \end{bmatrix} + \frac{1}{\alpha(n)} A_N(n) A_N^T(n) \underline{\mathbf{e}}(n) \\
&\begin{bmatrix} \bar{\boldsymbol{\varepsilon}}(n) \\ 0 \end{bmatrix} = \underline{\boldsymbol{\varepsilon}}(n) - \frac{1}{\beta(n)} B_N(n) B_N^T(n) \underline{\mathbf{e}}(n) \\
&\bar{\mathbf{E}}(n) = \begin{bmatrix} 0 \\ \bar{\mathbf{E}}(n-1) \end{bmatrix} + \underline{\boldsymbol{\varepsilon}}(n) \\
&\hat{\mathbf{h}}(n) = \hat{\mathbf{h}}(n-1) + \mu \mathbf{x}(n-N+1) E_{N-1}(n) \\
&\bar{\mathbf{E}}(n+1) = (1-\mu)\bar{\mathbf{E}}(n)
\end{aligned}$$

To demonstrate the convergence behavior of the FAP, we carry out the following computer simulations. The experimental conditions are: filter length is 150; the power level of additive noise is -40 dB relative to the average power of pure echo; the step-size parameter $\mu=1$. The input signal of $x(n)$ is coloured noise generated through a 3-order IIR filter given by $x(n)=0.9x(n-1)-0.5x(n-2)-0.2x(n-3)+u(n)$, where $u(n)$ is a zero-mean uncorrelated Gaussian signal. The magnitude response of the IIR filter is shown in Figure 2.3.

Figure 2.4 shows the convergence behaviors of NLMS and FAP with $N=2, 3$ and 8 . It can be seen that the FAP demonstrates better convergence speed than the NLMS. The convergence speed of the FAP is improved as the projection order N increases from 2 to

3. However, no improvement is gained when N increases from 3 to 8. This is because the FAP with projection order N is capable of decorrelating an autoregressive (AR) process of order N [30], and our input signal is a 3rd-order AR process.

The FAP algorithm provides a large choice of trade-offs between computational complexity and convergence rate. Although similar in spirit to the FNTF, the new algorithm is not a block version of the FNTF: it is mainly a stochastic gradient algorithm where the direction of search is given by some projection, while the FNTF is an approximate RLS algorithm. Simulations reported by Montazeri *et al.* [32] show that the AP algorithm can outperform both the LMS and RLS algorithms with respect to tracking capability. In our simulations, the FAP algorithm will be employed to improve the performance of the echo canceller.

Figure 2.3 Magnitude response of the IIR filter

Figure 2.4 Learning curves of NLMS and FAP

In Sections 2.1 to 2.4, we have discussed several adaptive filtering algorithms. They are classified into four families: LMS algorithms, LS algorithms, fast Newton transversal filters, and affine projection algorithms. We summarize the computational complexity (multiply-accumulate operations per iteration) and convergence rate of some of them in Table 2.8.

Table 2.8 Performance of adaptive filtering algorithms

<i>Algorithms</i>	<i>Computational Complexity (multiply-accumulate operations per iteration)</i>	<i>Convergence Rate</i>
NLMS	2L	Slow
FTF	7L	Fast
SFTF	8L	Fast
FNTF	$2L+(5-8)N$	Medium
FAP	$2L+20N$	Medium

2.6 Analysis-by-Synthesis Linear Predictive Coding

2.6.1 Principles

Vocoder methods are based on speech models. A rather successful model of speech production is the linear speech source-system production model shown in Figure 2.5 [33][34]. This model consists of a linear time-varying system (for the vocal tract model) excited by periodic impulse train excitation (for voiced speech) and random excitation (for unvoiced speech). If we assume that the vocal tract transfer function in the model of Figure 2.5 is of the form of an all-pole filter, the source-system model becomes associated with autoregressive (AR) time-series methods and its parameters are obtained by linear prediction (LP) analysis. Over the years, the most successful speech coding

techniques have been based on linear predictive coding (LPC). The main limitation of this model is the assumption that the source of excitation of the synthesis all-pole filter is either a train of pulses (for voiced speech) or random noise (for unvoiced speech). This gross simplification degrades the speech quality and intelligibility.

Figure 2.5 Discrete speech production model

A new class of source-system coders is the analysis-by-synthesis linear predictive coder. This new generation of coders also determines the parameters of all-pole synthesis filters by linear prediction. However, the excitation sequence is determined by minimizing the perceptually weighted error between the original and synthesized speech. The basic structure of the analysis-by-synthesis linear predictive coders is shown in Figure 2.6. The synthesis filters include the short-term predictor synthesis filter and the long-term predictor synthesis filter. The short-term synthesis filter is an all-pole time-varying filter for modeling the short-time spectral envelope of the speech. Its coefficients are computed using the method of linear prediction. The long-term synthesis filter models the fine structure of the speech spectrum. The long-term predictor is also called the pitch predictor since it removes the pitch periodicity. The speech can be decoded by filtering

the signal produced by the excitation generator through both the short-term synthesis filter and the long-term synthesis filter.

(a) Encoder

(b) Decoder

Figure 2.6 Basic structure of analysis-by-synthesis linear predictive coding

All analysis-by-synthesis coders share the same basic structure. They differ in the way the excitation sequence is modeled. The excitation models for analysis-by-synthesis linear predictive coding include the multi-pulse excitation, regular-pulse excitation, code-excited linear prediction, self-excited vocoder, and binary-pulse excitation, etc. The analysis-by-synthesis linear predictive coders are capable of producing toll-quality speech at bit rates below 8 kbits/s [35].

2.6.2 Introduction to G.723.1 and G.729

During the past decade, many digital speech coding standards have been defined for network and wireless applications. Most of these standards belong to the class of analysis-by-synthesis linear predictive coders. In the 1995-1996 time period, ITU standardized three low bit-rate speech coders: G.729, G.729A, and G.723.1 [37]. Presently, G.723.1 and G.729 are two popular vocoders used in packet telephony. In the following section, we introduce G.723.1 and G.729 vocoders which will be used to carry out the simulations in Chapter 3 and Chapter 5.

(1) G.723.1

ITU-T Recommendation G.723.1 specifies a coded representation that can be used for compressing speech or other audio signal component of multimedia services at a very low bit-rate [38]. This coder has two bit-rates associated with it, 5.3 and 6.3 kbps. The higher bit-rate provides greater quality, while the lower bit-rate gives good quality and provides system designers with additional flexibility. Both rates are a mandatory part of the encoder and decoder. The excitation model for the high rate coder is the multipulse model and for the low rate coder is the code excitation model. It is possible to switch between the two rates at the boundary of any frame. An option for variable rate operation using discontinuous transmission and noise fill during non-speech intervals is also available.

The G.723.1 coder is designed to operate with a digital signal by first performing telephone bandwidth filtering (Recommendation G.712) of the analogue input, then sampling at 8000 Hz and then converting to 16-bit linear pulse-code modulation (PCM)

for the input to the encoder. The output of the decoder should be converted back to analogue by similar means.

The block diagram of the encoder is shown in Figure 2.7. The G.723.1 vocoder encodes speech in 30ms frames. Since the sampling rate is 8kHz, each frame contains 240 samples. In addition, there is a look-ahead of 7.5ms, resulting in a total algorithmic delay of 37.5ms. Each frame is first high-pass filtered to remove the DC component and then divided into four subframes of 60 samples each. For every subframe, a 10th order Linear Prediction Coder (LPC) is computed using the unprocessed input signal. The LPC filter for the last subframe is quantized using a Predictive Split Vector Quantizer (PSVQ). The unquantized LPC coefficients are used to construct the short-term perceptual weighting filter, which is used to filter the entire frame and to obtain the perceptually weighted speech signal.

Figure 2.7 Block diagram of the G.723.1 speech coder (From Reference [38])

The perceptual weighting filter is used to reduce the perceived noise. Frequency-masking experiments have shown that greater amounts of quantization noise are undetectable by the auditory system in the frequency bands where the speech signal has high energy [10]. Thus, a large part of the perceived noise in a coder comes from the frequency bands where the speech signal level is low. The perceptual weighting filter de-emphasizes the importance of noise in the formant region and emphasizes its importance in spectral valleys. The spectrum of the output noise will be inversely proportional to the perceptual weighting filter's frequency response.

At the decoder, the LPC information, adaptive codebook information, and fixed codebook information are demultiplexed and then used to reconstruct the output signal. Postfiltering is used to reduce the perceptible noise. The adaptive postfilter consists of a formant and a pitch postfilter. The pitch postfilter is applied to the excitation signal before passing it through the LPC synthesis filter and the formant postfilter. The block diagram of the decoder is shown in Figure 2.8.

Silence compression techniques are used to reduce the transmitted bit-rate during silent intervals of speech. They are described in G.723.1 Annex A. Systems allowing discontinuous transmission are based on a Voice Activity Detection (VAD) algorithm and a Comfort Noise Generator (CNG) algorithm that allows the insertion of an artificial noise during silence periods. This feature is necessary to avoid noise modulation introduced when the transmission is switched off: if the background acoustic noise that was present during active periods abruptly disappears, this very unpleasant noise modulation may even reduce the intelligibility of the speech.

Figure 2.8 Block diagram of the G.723.1 speech decoder (From Reference [38])

(2) G.729

In the G.729 coding scheme, speech signals are coded at 8 kbit/s using conjugate-structure algebraic code-excited Linear Prediction (CS-ACELP). Like G.723.1, this coder is also designed to operate with a digital signal obtained by first performing telephone bandwidth filtering of the analogue input signal, then sampling it at 8000Hz, followed by conversion to 16-bit linear PCM for the input to the encoder. The block diagrams of encoder and decoder are shown in Figure 2.9 and Figure 2.10 [39], where G_C and G_P represent the fixed-codebook gain and the adaptive-codebook gain, respectively.

Figure 2.9 Block diagram of G.729 speech encoder (From Reference [39])

Figure 2.10 Block diagram of G.729 speech decoder (From Reference [39])

The G.729 coder operates on 10ms speech frames. For each frame, the speech signal is analyzed to extract the parameters of the CELP model. These parameters are encoded and transmitted to the decoder. At the decoder, these parameters are used to retrieve the excitation and synthesis filter parameters. The speech is reconstructed by filtering this excitation through the short-term synthesis filter which is based on a 10th order linear prediction filter. The long-term, or pitch synthesis filter is implemented using an adaptive-codebook approach. The reconstructed speech is then further enhanced by a postfilter. The G.729 has a 15 ms algorithmic codec delay, which is smaller than that of G.723.1. However, it is of higher complexity than G.723.1.

Chapter 3

Effect of Vocoder Distortion on Network Echo Cancellation

In packet voice communications, vocoders encode the speech signals at the transmitter and then reproduce them at the receiver. Vocoders compress speech signals so they can be transmitted at lower rates. However, the synthesized speech produced by vocoders is distorted to some degree. In this chapter, the effect of vocoder distortion on network echo cancellation is investigated using a simulation model. In this simulation model, NLMS, FTF and FAP algorithms are implemented in the echo canceller, and G.723.1 and G.729 standards are used as the speech coding schemes. Vocoder distortion may be due to the inaccurate representation of the excitation sequence, the postfiltering, and the quantization of LPC parameters, etc. Effects of these factors on echo cancellation are studied separately.

3.1 Simulation System Description

Figure 3.1 shows the block diagram of the system used to carry out the study. Each block is described as follows.

(1) Voice encoder and decoder

The encoder and decoder are specified by ITU-T Recommendation G.723.1 or ITU-T Recommendation G.729 [38][39]. We mainly use the G.723.1 vocoder to do the simulations. However, we also show some simulation results for the G.729 vocoder.

Figure 3.1 Simulation Model

(2) Channel

This block represents the impulse response of the analogue echo path, as depicted in Figure 3.2. The channel model used was provided by Mitel Corporation and represents a typical telephone channel.

(3) Band pass filter

This is a band-pass filter with the pass band 0.3 ~ 3.4 kHz. It is used to perform telephone bandwidth filtering of both the echo stream and the noise stream. The band pass filter is a 46th order FIR filter. Its frequency response is shown in Figure 3.3.

Figure 3.2 Impulse response of channel model

Figure 3.3 Frequency response of band-pass filter

In our simulations, we use Echo Return Loss Enhancement (ERLE) as the performance index of the algorithms. ERLE is defined as the ratio of the energy in the original echo to the energy in the residual echo. We may express it as

$$ERLE(dB) = 10 \log \frac{E[d^2(n)]}{E[e^2(n)]} \cong 10 \log \frac{\sum_{k=0}^{n_w} d^2(n-k)}{\sum_{k=0}^{n_w} e^2(n-k)} \quad (2.1)$$

where $d(n)$ is the original echo at time n , $e(n)$ is the residual echo (error signal) and n_w represents the size of the window average. Since the vocoders process data based on each frame, we also estimate ERLE frame by frame. That is, we choose n_w as 240, which is the size of a frame in the G.723.1 vocoder.

The excitation signal is a speech segment sampled at 8kHz, which is shown in Figure 3.4. It consists of 5 periods: A, B, C, D, and E, as labeled in Figure 3.4. The corresponding spectrogram is shown in Figure 3.5. The experimental conditions are: filter length L is 300; the power level of additive noise is -40dB relative to the average power of pure echo; two G.723.1 vocoders are set in the high bit rate. For both LMS and FAP, the step-size parameter is set to 1. For FTF algorithm, we choose the forgetting factor $\lambda=1-1/3L$.

Figure 3.4 Speech excitation

Figure 3.5 Spectrogram of the speech input

3.2 Simulation Results and Discussion

The simulations are carried out under two situations: with and without vocoders in the echo path. Figure 3.6, Figure 3.7 and Figure 3.8 present the comparative ERLE of these two cases by using NLMS, FTF and FAP, respectively. The projection order of FAP, denoted as N , is set to 5. We observe that the presence of vocoders causes much degradation in ERLE for all three algorithms, especially in periods C and E. From the spectrogram, we can see that the speech input in periods C and E has more complicated formant structure than that in periods B and D. Thus, G.723.1 vocoders reconstruct the

speech in periods C and E less faithfully and cause more distortion of the speech and eventually more degradation of the ERLE performance.

The FTF algorithm converges fastest among the three algorithms. If there are no vocoders in the echo path, the FTF has the best ERLE performance, and the performance of FAP is somewhere between those of FTF and NLMS. However, with the presence of vocoders, the performance of the FTF degrades the most. In this situation, the FAP outperforms both NLMS and FTF. To make a comparison among the three algorithms, we replot the results for the vocoding case in Figure 3.9. It can be seen that the FAP algorithm provides the best ERLE performance in the vocoding case. Compared with NLMS, about 5dB improvement is gained by using FAP with $N=5$.

Figure 3.6 ERLE performance for NLMS

Figure 3.7 ERLE performance for FTF

Figure 3.8 ERLE performance for FAP (N=5)

Figure 3.9 Comparative performance of NLMS, FTF and FAP in the vocoding case

Figure 3.10 Comparison of FAP for different projection orders

For the FAP algorithm, the change of projection order N affects the ERLE performance. Figure 3.10 shows the ERLE performance with various projection orders. Once again, there are vocoders in the echo path. We observe that noticeable improvement is gained when N is increased from 2 to 10. However, if N is further increased to 20, there is no significant improvement. The reason is that the speech can be adequately modeled by an AR process of 10th order, and the 10th order decorrelation filter in the AP algorithm adequately decorrelates the speech.

In the above simulations, the band-limited additive noise is 40dB down from the echo. To study the effect of near end noise, we carry out the simulation with zero additive noise. The FAP algorithm with $N=5$ is adopted. In Figure 3.11, the curve indicated as “zero noise” shows the simulation results. We can see that the results are quite similar between these two cases showed in Figure 3.11. This means that the effect of vocoder distortion is a dominant factor causing the degradation of ERLE performance when the signal-to-noise ratio is higher than 40 dB.

Figure 3.11 Comparative performance for different noise levels

3.3 Effect of Vector Quantization of LPC Parameters

The set of LPC parameters can be quantized using either scalar or vector quantization methods. In scalar quantization, the LPC parameters are quantized individually using either uniform or non-uniform quantization. In vector quantization (VQ), the set of LPC parameters is considered as one entity, and they are quantized using a large trained codebook by minimizing a specified spectral distortion measure [43]. Generally speaking, vector quantizers yield smaller quantization error than scalar quantizers at the same bit rate, however, we pay the price of high complexity of VQ

algorithms. In the G.723.1 vocoding scheme, the LPC parameters are quantized using VQ to save bit-rate.

When quantizing the set of LPC parameters, we have to ensure the stability of the LPC synthesis filter. Therefore, it is necessary to transform the coefficients of the LPC synthesis filter into another set of parameters which are related in a one-to-one manner to the coefficients of a stable synthesis filter. In the G.723.1 codec, the prediction coefficients are converted into the line spectrum pairs (LSP). The LSP coefficients are then quantized using a predictive split vector quantizer.

In order to study the effect of this vector quantizer, we remove the vector quantizer in the coder and send the LSP coefficients directly to the decoder. Using this modified G.723.1 vocoder, we obtain the simulation results showed in Figure 3.12, Figure 3.13 and Figure 3.14. From these three figures, we can see that vector quantization of LSP coefficients does not have significant effects on the ERLE performance.

Figure 3.12 ERLE performance for NLMS: VQ and no VQ cases.

Figure 3.13 ERLE performance for FTF: VQ and no VQ cases

Figure 3.14 ERLE performance for FAP(N=5): VQ and no VQ cases

3.4 Effect of Formant Postfiltering

When the bit-rate of the vocoder is below 8 kb/s, the quality of the reconstructed speech starts to degrade and formant postfiltering techniques should be used to improve the speech quality. A conventional autoregressive-moving average (ARMA) postfilter is represented as follows:

$$F(z) = \frac{1 - \sum_{k=1}^p a_k \lambda_1^k z^{-k}}{1 - \sum_{k=1}^p a_k \lambda_2^k z^{-k}} (1 - \beta z^{-1}), \quad 0 \leq \lambda_1 < \lambda_2 \leq 1 \quad (3.1)$$

where p is the predictor order and a_k is the LPC parameter. The filter spectrum is controlled by the values of λ_1 , λ_2 and β . The parameters λ_1 and λ_2 are used to increase the bandwidth of the spectrum resonances. The parameter β is used to remove the spectral tilt. For the G.723.1 vocoder, we have $\lambda_1=0.65$ and $\lambda_2=0.75$ and β is estimated according to the speech characteristics [38].

To study the effect caused by the distortion of the formant postfiltering, we remove the formant postfilter in the G.723.1 vocoder and obtain simulation results showed in Figure 3.15, Figure 3.16 and Figure 3.17. We see that, with no formant postfiltering, the ERLE performance is a little bit improved for NLMS and FTF algorithms. For the FAP algorithm, this effect is negligible.

Figure 3.15 ERLE performance for NLMS: postfiltering and no postfiltering cases.

Figure 3.16 ERLE performance for FTF: postfiltering and no postfiltering cases.

Figure 3.17 ERLE performance for FAP(N=5): postfiltering and no postfiltering cases.

3.5 Effect of Excitation Modeling

The excitation sequence of analysis-by-synthesis linear predictive coding carries all the information that has not been captured by LP and pitch analyses. It is determined by an optimization process which minimizes a measure of the weighted difference between the input speech and the synthesized speech. The modeling of the excitation sequence plays a dominant role in determining the coder performance and complexity. The three most common excitation models are: the multi-pulse excitation (MPE), regular-pulse excitation (RPE), and code-excited linear prediction (CELP). The MPE model was proposed by Atal and Remde [40]. This method assumes that the excitation signal is modeled by a definite number of pulses in a short time period. The pulses are non-uniformly spaced. The RPE model proposed by Kroon *et al* [41] also assumes that the excitation signal consists of multiple pulses. The difference of this approach from the MPE model is that the pulses in the RPE model are uniformly spaced. The CELP algorithm is the leading technique for analysis-by synthesis linear predictive coders. It was first introduced by Atal and Schroeder [42]. The CELP algorithm encodes the excitation using a codebook of Gaussian sequences. The basic idea under the CELP algorithm is that the residual signal after short-term and long-term prediction becomes noise-like, and can be modeled by a zero-mean Gaussian process with slowly varying power spectrum.

In the G.723.1 vocoder, the excitation model is the MPE for the higher bit-rate and CELP for the lower bit-rate. To determine the excitation vector, firstly, the residual signal

$r(n)$ is computed for each subframe (60 samples). $r(n)$ is obtained by subtracting the zero-input response of the weighted synthesis filter and the contribution of the pitch predictor from the harmonic weighted speech vector [38]. Then, treated as a target vector, $r(n)$ is approximated by $r'(n)$:

$$r'(n) = \sum_{j=0}^n h(j)v(n-j), \quad 0 \leq n \leq 59 \quad (3.2)$$

where $v(n)$ is the excitation vector and $h(n)$ is the combined filter which consists of the short-term synthesis filter, the formant perceptual weighting filter and the harmonic noise shaping filter. $v(n)$ is selected to minimize the mean square error between $r(n)$ and $r'(n)$. For the higher bit-rate, $v(n)$ consists of 5 or 6 pulses, and we need to determine their positions and amplitudes. For the lower bit-rate, $v(n)$ is a codeword multiplied by a codebook gain. The index of the codeword is obtained by a search approach.

Using the MPE, CELP or other modeling approaches, we cannot make $r(n)$ exactly equal to $r'(n)$. Hence, we obtain inaccurate excitation sequences that will cause non-linear distortion on the synthesized speech. To study the effect of excitation modeling, we let $r'(n)$ equal to $r(n)$ and obtain the accurate excitation vector from Eq.(4.2):

$$v(0) = r(0) / h(0), \quad (3.3)$$

$$v(n) = \left[r(n) - \sum_{j=1}^n h(j)v(n-j) \right] / h(0), \quad 1 \leq n \leq 59 \quad (3.4)$$

Implementing the accurate excitation vector, we obtain the simulation results presented in Figure 3.18, Figure 3.19 and Figure 3.20. For all three algorithms, we observe significant improvement when using accurate excitation. This means that excitation modeling is an important factor causing the degradation of ERLE.

Figure 3.18 ERLE performance for NLMS: accurate and inaccurate excitation.

Figure 3.19 ERLE performance for FTF: accurate and inaccurate excitation.

Figure 3.20 ERLE performance for FAP: accurate and inaccurate excitation.

3.6 Simulations for G.729 codec

In the simulation system, we replace the G.723.1 codec by the G.729 codec and carry out the simulations under the same conditions. Figure 3.21, Figure 3.22 and Figure 3.23 show the ERLE performance for the G.729 and G.723.1 using NLMS, FTF and FAP algorithms, respectively. It can be seen that G.729 and G.723.1 codecs cause similar amounts of degradation on ERLE performance.

Figure 3.21 Comparative performance for G.729 and G.723.1 (NLMS)

Figure 3.22 Comparative performance for G.729 and G.723.1 (FTF)

Figure 3.23 Comparative performance for G.729 and G.723.1 (FAP, N=5)

3.7 Summary

In this chapter, we first proposed a simulation model to study the effect of vocoder distortion on network echo cancellation. Two standard vocoders were used in our simulation systems. They are G.723.1 and G.729, which are typically used in packet voice communications.

In our simulations, we focus on the G.723.1 vocoder. We use the NLMS, FTF and FAP algorithms to implement the echo canceller. For all three algorithms, we observe that the ERLE performance of the echo canceller degrades significantly due to the effect of vocoder distortion. We also observe that FAP outperforms both NLMS and FTF in the presence of vocoders. This may be because the FAP algorithm has outstanding tracking

capability. For FAP projection order N , we find that $N=10$ is adequate to exploit the potential of the FAP algorithm.

The vocoder distortion is caused by several factors, including the vector quantization of LSP coefficients, the formant postfiltering, and the excitation modeling, etc. Our simulations show that the vector quantization of LSP coefficients in the vocoders does not have significant effects on the echo canceller's performance. For the formant postfiltering, we find that it has little effect on echo cancellation for the NLMS and FTF algorithms, and almost no effect for the FAP algorithm. Finally, we observe that the inaccurate representation of the excitation sequence produces the most significant effect on the echo canceller's performance.

In addition, using the G.729 codec instead of the G.723.1 codec in our simulation system, we observe that the G.729 codec has similar effects on echo cancellation as compared the with G.723.1 codec.

Chapter 4

Effect of Packet Loss on Network Echo Cancellation

For a packet-based network, packet losses due to congestion and transmission errors in the network can significantly degrade the performance of echo cancellers. In this chapter, we study the impact of packet losses on the echo cancellers in which the NLMS, SFTF and FAP algorithms are adopted.

4.1 Effects of Packet Loss for Different Recovery Techniques

The effects introduced by packet loss greatly depend on how packet losses are recovered. Packet loss recovery techniques can be divided into two categories: sender-based repair and receiver-based repair [44][45]. Sender-based repair requires the participation of the sender to achieve recovery from packet loss and is beyond the scope of this thesis. The receiver-based repair is also called error concealment. The idea of error concealment is to generate a replacement for a lost packet which is similar to the original.

Among error concealment techniques, the insertion-based schemes are simple to implement and our studies will focus on methods belonging to this category.

4.1.1 Insertion-Based Recovery Techniques

Insertion-based recovery techniques insert a replacement packet to fill the gap left by a lost packet. The fill-in packet can be:

- Silence
- White noise
- Previous packet

The silence substitution fills the gap left by a lost packet with silence in order to maintain the timing relationship between the surrounding packets. The performance of silence substitution degrades rapidly as packet sizes increase. Although silence substitution has been shown to perform poorly, the use of silence substitution is widespread, primarily because it is simple to implement. For noise substitution, instead of filling in the gap left by a lost packet with silence, background noise is inserted instead. Some studies have shown that inserting white noise, instead of silence, can give subjectively better quality [46] and improve intelligibility [47]. Among these three methods, the repetition of the previous packet yields the best voice quality since speech signals exhibit large amounts of short-term self-similarity. However, it will disturb the determination of the propagation delay of echoes.

4.1.2 Simulations and Discussion

The simulation system is illustrated in Figure 4.1. The test speech signal and the experimental conditions used to carry out the simulations are the same as those in Chapter 3. We assume that the 38th frame is lost in the *Packet Loss (1)* block, that is, samples 8881~9120 are lost.

Figure 4.1 Packet loss simulation system for no vocoding case

Figure 4.2, Figure 4.3 and Figure 4.4 plot the ERLE for the NLMS, SFTF and FAP algorithms respectively when using silence substitution. We see that the impact of packet

loss on the echo canceller's performance is the greatest when using the SFTF, and is the smallest when using the FAP. The degradation for SFTF can be as much as 35dB, while it is only 15dB for FAP. As compared with the NLMS, the ERLE for FAP not only degrades less but also recovers more quickly.

Figure 4.5, Figure 4.6 and Figure 4.7 plot the simulation results for the repetition recovery scheme. Again, we observe that the impact of packet loss on the echo canceller's performance is smallest when using the FAP algorithm.

The simulation results for white noise substitution are shown in Figure 4.8, Figure 4.9 and Figure 4.10. In this case, the ERLE can be reduced below 0dB for all three algorithms. Comparing with the previous results, we see that the impact of packet loss is much greater for the NLMS and the FAP. For the SFTF, the result is similar to those using the other two recovery techniques.

Figure 4.2 Impact of packet loss on ERLE performance for NLMS and silence substitution

Figure 4.3 Impact of packet loss on ERLE performance for SFTF and silence substitution

Figure 4.4 Impact of packet loss on ERLE performance for FAP(N=5) and silence substitution

Figure 4.5 Impact of packet loss on ERLE performance for NLMS and repetition

Figure 4.6 Impact of packet loss on ERLE performance for SFTF and repetition

Figure 4.7 Impact of packet loss on ERLE performance for FAP(N=5) and repetition

Figure 4.8 Impact of packet loss on ERLE performance for NLMS and white noise substitution

Figure 4.9 Impact of packet loss on ERLE performance for SFTF and white noise substitution

Figure 4.10 Impact of packet loss on ERLE performance for FAP(N=5) and white noise substitution

To make comparisons of FAP for different projection orders with packet loss, we carry out the simulations using FAP with $N=2, 10, 20$. The results are presented in Figure 4.11, Figure 4.12 and Figure 4.13. For silence substitution and repetition, we see that the ERLE recovers more quickly with higher N . This may be due to the fact that the convergence rate of FAP is increased as the projection order N is increased. Similar to the results in Chapter 3, we observe that no significant improvement is gained when N surpasses 10. For white noise substitution, we find that increasing N does not have the effect of reducing the impact of packet loss.

Figure 4.11 Impact of packet loss on ERLE performance for FAP: $N=2, 10, 20$, silence substitution

Figure 4.12 Impact of packet loss on ERLE performance for FAP: N=2, 10, 20, repetition

Figure 4.13 Impact of packet loss on ERLE performance for FAP: N=2, 10, 20, white noise substitution

4.2 Proposed insertion-based recovery scheme

In section 4.1, we observe that inserting white noise degrades the ERLE performance most. However, noise substitution can improve intelligibility of voice, compared with silence substitution and does not have the problem of confusing the determination of the echo's delay as in the case of repetition. In this section, we propose a new noise substitution approach to handle packet loss.

The repetition technique explores the similarities between neighboring packets in the time domain. Reasonably, we can assume that similarities between neighboring packets also exist in the frequency domain. Instead of directly inserting white noise, we insert coloured noise that has the same power spectrum as the previous packet, to replace the lost packet. The coloured noise is obtained by filtering white noise. The new method is implemented in the following steps:

1. Estimate the power spectrum of the previous frame;
2. Design a filter with the transfer function $H(f)$, where $|H(f)|^2$ is equal to the estimated power spectrum;
3. Use the filter designed in the second step to filter white noise;
4. Generate the replacement frame using filtered noise.

The filtered noise has the same power spectrum as the previous frame. Due to the similarities between the neighboring frames [45], the filtered noise is more similar to the lost packet as compared with white noise in a statistical sense. Thus, we can expect that

this method will provide better voice quality and echo cancellers will have better performance than with the white noise substitution.

The block diagram for this new method is illustrated in Figure 4.14

Figure 4.14 Block diagram of the new packet loss repair method

Figure 4.15, Figure 4.16 and Figure 4.17 provide comparative ERLE performance of the new method and that of inserting white noise for NLMS, SFTF and FAP respectively. The power spectrum is estimated using Welch's averaged periodogram method. The

length of the FFT is 64, as is the filter length. It can be seen that much improvement is gained for NLMS and FAP algorithms, especially for FAP. However, no significant improvement is observed for the SFTF. This is because SFTF is not a stochastic gradient algorithm, but a least-squares algorithm. It is deterministic in approach. Thus, similarities in statistics, such as power spectrum, are of no help to SFTF's performance.

Figure 4.18 and Figure 4.19 present comparative ERLE performance of silence substitution, repetition and the new method for NLMS and FAP respectively. We can see that the new method has similar ERLE performance compared with the other two packet loss recovery techniques.

Figure 4.15 Comparative ERLE performance of “white noise” and “filtered noise” for NLMS

Figure 4.16 Comparative ERLE performance of “white noise” and “filtered noise” for SFTF

Figure 4.17 Comparative ERLE performance of “white noise” and “filtered noise” for FAP(N=5)

Figure 4.18 Comparative ERLE performance of silence substitution, repetition and proposed method for NLMS

Figure 4.19 Comparative ERLE performance of silence substitution, repetition and proposed method for FAP (N=5)

4.3 Comparison of “Packet Loss (1)” and “Packet Loss (2)”

As shown in the simulation model (see Figure 4.1), packet loss can happen in two locations on the echo path: block “*Packet Loss (1)*” and block “*Packet Loss (2)*”. In previous simulations, we only implement “*Packet Loss (1)*”. To make comparisons between these two cases, we implement “*Packet Loss (2)*” under the same circumstances. Repetition is used as the packet loss recovery technique. Figure 4.20, Figure 4.21 and Figure 4.22 show that the impact of packet loss is similar in these two cases.

Usually we have knowledge of “*Packet Loss (2)*”. Thus we can freeze the echo canceller when “*Packet loss (2)*” happens. In other words, “*Packet Loss (2)*” does not pose a problem to the echo canceller.

Figure 4.20 Impact of packet loss for NLMS: comparative performance of “Packet loss (1)” and “Packet loss (2)”

Figure 4.21 Impact of packet loss for SFTF: comparative performance of “Packet loss (1)” and “Packet loss (2)”

Figure 4.22 Impact of packet loss for FAP(N=5): comparative performance of “Packet loss (1)” and “Packet loss (2)”

4.4 Summary

In this chapter, we have studied the impact of packet loss on the performance of echo cancellers. The impact depends on the adaptive filtering methods used in the echo canceller and the packet loss recovery techniques. In our simulations, the NLMS, SFTF and FAP algorithms are implemented for the echo canceller. For packet loss recovery, several insertion-based repair schemes are used to generate the replacement packets. They are silence substitution, white noise substitution and repetition. For the NLMS and FAP algorithms, simulation results show that ERLE performance degrades the most when white noise is used for replacement. For the SFTF, simulation results show that ERLE is similar in the three cases. In regard to silence substitution and repetition, we observe that packet loss has the least impact on the ERLE performance when using the FAP algorithm. We also notice that, when the projection order of FAP is less than 10, increasing the projection order can reduce the effect of packet loss. However, no noticeable improvement is gained when the projection order surpasses 10.

To suppress the impact of packet loss on ERLE performance, a new noise substitution method is proposed. We first estimate the power spectrum of the previous packet. Then, we design a filter with the transfer function $H(f)$, where $|H(f)|^2$ is equal to the estimated power spectrum. After that, we use this filter to filter white noise and generate the replacement packet using filtered noise. Simulations show that the proposed method is better than that of directly inserting white noise with respect to ERLE performance of echo cancellers, when NLMS and FAP algorithms are used.

Simulation results also show that if packet loss happens in the block "Packet loss (2)", the impact on ERLE performance is similar to that of "Packet loss (1)".

Chapter 5

Combination of Effect of Vocoder Distortion and Effect of Packet Loss

In Chapter 3 and 4, the effect of vocoder distortion and the effect of packet loss are investigated separately. In this chapter, we combine them together to study their comprehensive effects on network echo cancellation.

5.1 Simulation system

The model shown in Figure 5.1 simulates network echo cancellation when both vocoders and packet losses are present in the echo path. In this model, packet losses happen between the coders and decoders. Hence they are involved in the protocol level specification. In order to simplify the implementation of the simulation model, we put the “Packet Loss” blocks after the voice codecs to approximate the real situation. The simplified simulation model is shown in Figure 5.2.

Figure 5.1 Simulation model for combined effects of vocoder distortion and packet loss

Figure 5.2 Simplified simulation model for combined effects of vocoder distortion and packet loss

5.2 Simulation Results and Discussion

In this section, we combine both factors together to carry out the simulations. The simulation results for NLMS, SFTF and FAP algorithms are presented in Figure 5.3, Figure 5.4 and Figure 5.5. Silence substitution is used as the packet loss recovery technique. Comparison of the simulation results obtained in Section 4.1 and in this

section reveals that when there are vocoders in the echo path, the degradation of ERLE performance due to the packet loss is less than that when there are no vocoders in the echo path. In other words, vocoders smooth the impact of packet loss on the echo cancellers. The reason for this is that both the next frame and the previous frame are involved in reconstructing the present speech frame due to the following facts:

1. In the encoder, the LPC analysis for each subframe (60 samples) is performed using a window of 180 samples that is centered on the subframe;
2. In the decoder, to obtain an LSP vector for each subframe, linear interpolation is performed between the present decoded LSP vector and the previous decoded LSP vector.

Figure 5.3 Impact of packet loss on ERLE performance: with/without vocoders (NLMS; silence substitution)

Figure 5.4 Impact of packet loss on ERLE performance: with/without vocoders (SFTF; silence substitution)

Figure 5.5 Impact of packet loss on ERLE performance: with/without vocoders (FAP, N=5; silence substitution)

Chapter 6

Conclusions

In packet based networks, such as IP and ATM, vocoders and packet losses may be present in the echo path. The research work presented in this thesis is to investigate the performance of echo cancellers affected by vocoders and packet losses. We first define a simple simulation model. Then we implement the model to test existing adaptive filtering algorithms, including NLMS, FTF and FAP. ERLE is used as the index of performance.

For the effect of vocoder distortion, we observe that the degradation of ERLE caused by vocoder distortion varies with the speech input. In the speech periods with more formants, since vocoders synthesize the speech less faithfully, they cause more distortion of the speech signals and thus more degradation of ERLE. The effect of vocoder distortion also largely depends on the adaptive filtering algorithms used in echo cancellers. Our simulations show that FAP outperforms NLMS and FTF with respect to the ERLE performance if there are vocoders in the echo path. Under our experimental conditions, FAP with $N=5$ achieves 5dB improvement as compared with NLMS. The FTF algorithm provides the best ERLE performance in the no vocoding case. However, its performance in the vocoding case is the worst. Increasing the projection order of FAP

from 2 to 10, we observe several dB improvement in the vocoding case. As the projection order is further increased, no significant improvement is observed. This means that the 10th order decorrelation filter in the FAP algorithm is good enough to decorrelate the speech signals, which usually can be modeled as the 10th order autoregressive (AR) time-series.

The conclusions described above apply to both the G.723.1 vocoder and the G.729 vocoder.

The vocoder distortion is caused by many factors. One of them is the quantization of the LPC parameters. For the G.723.1 vocoder, the LPC coefficients are transformed into the line spectrum pairs (LSP). Then, the LSP parameter set is quantized using a vector quantization (VQ) method. We remove the vector quantizer in the G.723.1 vocoder and carry out the simulation to study the effect of vector quantization of the LPC parameters. We get similar results compared with the normal G.723.1 vocoder. This means that vector quantization of the LPC parameters does not cause much distortion in the G.723.1 vocoder.

The formant postfilter is used for perceptual masking. However, it also causes some distortion. In the G.723.1 vocoder, a conventional ARMA postfilter is employed. Simulations show that the formant postfiltering affects echo cancellation slightly when using the NLMS and FTF algorithms, and has almost no effect on the echo canceller using the FAP algorithm.

In order to decrease the bit rates of vocoders, the excitation sequence is compressed by using an excitation model. The excitation models used in analysis-by-synthesis linear predictive coding include MPE, RPE, and CELP, etc. Our simulations show that the

inaccuracy of the excitation sequence due to excitation modeling causes significant degradation in the ERLE performance of echo cancellers.

The effect of packet loss on the performance of echo cancellers not only depends on the adaptive filtering algorithms adopted in echo cancellers, but also depends on the packet loss recovery techniques. There are many kinds of packet loss recovery techniques. Our studies focus on insertion-based repair schemes, including silence substitution, noise substitution and repetition. These three methods fill the gap left by a lost packet with silence, white noise and the previous packet, respectively. The adaptive filtering algorithms implemented in the simulation model for packet loss are NLMS, SFTF and FAP. We find that FAP has the best ERLE performance among the three algorithms when we use silence substitution and repetition to repair packet loss. What we observe is that the ERLE for FAP recovers more quickly than that for NLMS and degrades least among the three algorithms. As the projection order of FAP is increased, the impact of packet loss is reduced. Again, we observe that when the projection order surpasses 10, there is no noticeable improvement afterwards.

Meanwhile, for the FAP and NLMS algorithms, using white noise substitution results in more degradation of ERLE than the other two recovery techniques. To solve this problem, we propose a new noise substitution method. We first estimate the power spectrum of the frame that immediately precedes the lost frame. Then we design a filter with the transfer function $H(f)$, where $|H(f)|^2$ is equal to the estimated power spectrum of the previous frame. After that, we use this filter to filter white noise. The output of the filter is used to generate the replacement frame. Thus, the fill-in noise has the same power spectrum as the previous frame. Due to the similarities between the neighboring

speech segments, the characteristics of the fill-in noise are closer to that of the lost frame compared with white noise. Simulation shows that the new method results in less impact of packet loss than does directly inserting white noise, especially for the FAP algorithm.

Studies in Chapter 4 also show that the packet loss occurring before the analogue channel ("Packet loss (2)") has similar impact compared with the packet loss occurring after the analogue channel ("Packet loss (1)").

Combining the effect of packet loss and the effect of vocoder distortion, we find that packet loss causes less degradation of ERLE than in the case of no vocoders.

During the process of this research, a couple of ideas have arisen but not yet been implemented, due to time limitations. The recommendations for further research are summarized as follows:

1. Investigate the effect of vocoder distortion and effect of packet loss using other adaptive filtering algorithms, such as Fast Newton Algorithm, IIR Adaptive Filter, Nonlinear Adaptive Filtering, etc.
2. Investigate the effect of packet loss using some other packet loss recovery schemes, such as interpolation-based repair and regeneration-based repair.

References

- [1] M. Sondhi, "An Adaptive Echo Canceller", *Bell System Tech. Journal*, Vol. 46, pp. 497-511, March 1967.
- [2] D. L. Duttweiler and Y. S. Chen, "A Single-Chip VLSI Echo Canceller," *Bell System Tech. Journal*, Vol. 59, No. 2, pp. 149-160, February 1980.
- [3] Kazuo Murano, Shigeyuki Unagami, Fumio Amano. "Echo Cancellation and Applications," *IEEE Communications Magazine*, pp. 49 – 57, January 1990.
- [4] Thomas J. Kostas *et al.*, "Real-Time Voice Over Packet-Switched Networks". *IEEE Network*, Vol. 12, No. 1, pp. 18-27, Jan.-Feb. 1998.
- [5] Bharat T. Doshi *et al.*, "Protocols, Performance, and Controls for Voice over Wide Area Packet Networks," *Bell Labs Technical Journal*, Vol. 3, No. 4, pp. 297-337, Oct.-Dec. 1998.
- [6] Sudhir R. Ahuja and Krishna G. Murti, "Packet Telephony," *Bell Labs Technical Journal*, Vol. 2, No.2, pp. 5-14, Spring 1997.
- [7] Uyless Black, *Voice over IP*, Prentice Hall Inc., Upper Saddle River, New Jersey, 1999.
- [8] Andre Gilloire, "Recent Advances in Adaptive Filtering Algorithms for Acoustic Echo Cancellation," International Workshop on Acoustic Echo and Noise Control, Roros, Norway, pp.115 – 134, June 1995.

- [9] Mark Kahrs (Ed.), "The Past, Present, and Future of Audio Signal Processing," *IEEE Signal Processing Magazine*, pp. 30 – 57, September 1997.
- [10] H. Fletcher, "The mechanism of hearing as revealed through experiment on the masking effect of thermal noise", *Pro. of National Academy of Sciences*, pp. 265-274, April 1938.
- [11] B. Widrow, M. E. Hoff, "Adaptive switching circuits," *IRE WESCON Conv. Rec.*, pt. 4, pp.96-104, 1960.
- [12] T. C. Hsia, "Convergence analysis of LMS and NLMS adaptive algorithms," *Proceeding of ICASSP'83*, Vol. 2, pp. 667-670, 1983.
- [13] Simon Haykin. *Adaptive Filter Theory*, 3rd ed., Prentice Hall Inc., Upper Saddle River, New Jersey, 1996.
- [14] J. Nagumo and A. Noda, "A learning method for system identification," *IEEE Trans. Automat. Contr.*, Vol. AC-12, No. 3, pp. 282-287, June 1967.
- [15] Dirk T. M. Slock, "On the convergence behavior of the LMS and the normalized LMS algorithms," *IEEE Transactions on Signal Processing*, Vol. 41, No. 9, pp.2811-2825, September 1993.
- [16] M. Tarrab and A. Feuer, "Convergence and performance analysis of the normalized LMS algorithm with uncorrelated Gaussian data," *IEEE Trans. Inform. Theory*, Vol. 34, No. 4, pp. 680-691, July 1988.
- [17] L. Ljung, M. Morf, and D. Falconer, "Fast calculation of gain matrices for recursive estimation schemes," *Int. J. Contr.*, Vol. 27, pp. 1-19, Jan. 1978.

- [18] G. Carayannis, D. Manolakis, and N. Kalouptsidis, "A fast sequential algorithm for least square filtering and prediction," *IEEE Trans. Acoust. Speech, Signal Processing*, Vol. ASSP-31, No. 6, pp. 1394-1402, Dec. 1983.
- [19] John M. Cioffi, Thomas Kailath, "Fast, recursive-least-squares transversal filters for adaptive filtering," *IEEE Trans. on Acoustic, Speech and Signal Processing*, Vol. 32, No. 2, pp. 304 – 337, April 1984.
- [20] Dirk T. M. Slock, Thomas Kailath, "Numerically stable fast transversal filters for recursive least squares adaptive filtering", *IEEE Trans. on Signal Processing*, Vol. 39, No. 1, pp. 92 – 114, January 1991.
- [21] Ahmed Benallal, Andre Gilloire, "A new method to stabilize fast RLS algorithms based on a first-order model of the propagation of numerical errors," *Proceedings of ICASSP'88*, Vol. 3, pp. 1373 – 1376, 1988.
- [22] Ahmed Benallal, Andre Gilloire, "Improvement of the tracking capability of the numerically stable fast RLS algorithms for adaptive filtering," *Proceedings of ICASSP'89*, Vol. 2, pp. 1031-1034, 1989.
- [23] George V. Moustakides, Sergios Theodoridis, "Fast Newton transversal filters – a new class of adaptive estimation algorithms", *IEEE Trans. on Signal Processing*, Vol. 39, No. 10, pp. 2184 – 2193, October 1991.
- [24] Panagiotis P. Mavridis, George V. Moustakides, "Simplified Newton-type adaptive estimation algorithms", *IEEE Trans. on Signal Processing*, Vol. 44, No. 8, pp. 1932 – 1940, August 1996.

- [25] K. Ozeki and T. Umeda, " An Adaptive Filtering Algorithm Using an Orthogonal Projection to an Affine Subspace and Its Properties," *Elec. and Com. in Japan*, Vol. J67-A, No. 2, pp. 126-132, Feb. 1984.
- [26] Steven L. Gay, Sanjeev Tavathia, "The fast affine projection algorithm," *Proceedings of ICASSP'95*, Vol. 5, pp. 3023 – 3026, 1995.
- [27] Steven. L. Gay, "A fast converging, low complexity adaptive filtering algorithm," in *Proc. IEEE Acoust., Speech, Signal Process. Workshop Appl. Signal Process. Audio Acoust.*, New Paltz, NY, Oct. 1993.
- [28] John M. Cioffi, Thomas Kailath, "Windowed fast transversal filters adaptive algorithms with normalization", *IEEE Trans. on Acoustic, Speech and Signal Processing*, Vol. 33, No. 3, pp. 607 – 625, June 1985.
- [29] S. C. Douglas, "Efficient approximate implementations of the fast affine projection algorithm using orthogonal transforms," *Processing of ICASSP'96*, Vol.3, pp. 1656-1659, 1996.
- [30] Markus Rupp, "A family of adaptive filter algorithms with decorrelating properties," *IEEE Trans. on Signal Processing*, Vol. 46, No. 3, pp. 771 – 775, March 1998.
- [31] Masashi Tanaka, Yutaka Kaneda, Shoji Makino Junji Kojima, "Fast projection algorithm and its step size control," *Proceedings of ICASSP'95*, Vol. 2, pp. 945 – 948, 1995.
- [32] Mohsen Montazeri, Pierre Duhamel, "A set of algorithms linking NLMS and block RLS Algorithms," *IEEE Trans. on Signal Processing*, Vol. 43, No. 2, pp. 444 – 453, Febuary 1995.

- [33] Ronald W. Schafer, Lawrence R. Rabiner, "Digital Representations of Speech Signals", *Proceedings of The IEEE*, Vol. 63, No. 4, pp. 662 – 677, April 1975.
- [34] John Makhoul, "Linear Prediction: A Tutorial Review," *Proceedings of The IEEE*, Vol. 63, No. 4, pp. 561 – 580, April 1975.
- [35] Andreas S. Spanias, "Speech Coding: A Tutorial Review," *Proceedings of The IEEE*, Vol. 82, No. 10, pp. 1541 – 1582, October 1994.
- [36] Peter Kroon and W. Bastiaan Kleijin, "Linear Predictive Analysis by Synthesis Coding," in *Modern Methods of Speech Processing*, Ravi P. Ramachandran and Richard Mammone. Eds., Kluwer Academic Publishers, Norwell, Massachusetts. 1995.
- [37] Richard V. Cox, Peter Kroon, "Low Bit-Rate Speech Coders for Multimedia Communication," *IEEE Communications Magazine*, pp. 34 – 41, December 1996.
- [38] ITU-T, Recommendation G.723.1 Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 kbit/s, 1996.
- [39] ITU-T, Recommendation G.729 Coding of Speech at 8 kbit/s using Conjugate-Structure Algebraic-Code-Excited Linear-Prediction (CS-ACELP), 1995.
- [40] B. Atal and J. Remde, "A new model for LPC excitation for producing natural sounding speech at low bit rates," *Proceeding of ICASSP'82*, pp. 614-617, Apr. 1982.
- [41] P. Kroon, E. Deprettere, and R. J. Sluyeter, "Regular-pulse excitation - A novel approach to effective and efficient multipulse coding of speech," *IEEE Trans. Acoust., Speech, Signal Process.*, ASSP-34, No. 5, Oct. 1986.

- [42] M. R. Schroeder and B. S. Atal, "Code-excited linear prediction (CELP): high quality speech at very low bit rates, " *Proceeding of ICASSP'85*, pp. 937-940, 1985.
- [43] Allen Gersho, Robert M. Gray, *Vector quantization and signal compression*, Kluwer Academic Publishers, Norwell, Massachusetts, 1992.
- [44] Georg Carle, Ernst W. Biersack, "Survey of error recovery techniques for IP-based audio-visual multicast applications," *IEEE Network*, pp. 24 – 36, November/December 1997.
- [45] Colin Perkins, Orion Hodson, Vicky Hardman, " A survey of packet loss recovery techniques for streaming audio," *IEEE Network*, pp. 40 – 48, Spetember/October 1998.
- [46] G. A. Miller and J. C. R. Licklider, "The intelligibility of interrupted speech," *J. Acoust. Soc. Amer.*, Vol. 22, No. 2, pp. 167-173, 1950.
- [47] R. M. Warren, *Auditory Perception*, Pergamon Press, 1982.