การประเมินเชิงประสบการณ์ของอัลกอริทึมแอ็ฟไฟน์โปรเจคชันสำหรับการกำจัดเสียงป้อนกลับใน

เครื่องช่วยฟัง

นางสาววิมลธา เขียววงศ์พระจันทร์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2547

ISBN 974-53-1522 -2

EMPIRICAL EVALUATION OF AFFINE PROJECTION ALGORITHM FOR ACOUSTIC FEEDBACK CANCELLATION IN HEARING AIDS

Miss Vimontha Khieovongphachanh

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering in Electrical Engineering
Department of Electrical Engineering
Faculty of Engineering
Chulalongkorn University
Academic year 2004
ISBN 974-53-1522-2

Thesis Title	EMPIRICAL EVALUATION OF AFFINE PROJECTION ALGORITHM FOR ACOUSTIC FEEDBACK CANCELLATION IN HEARING AIDS
By	Miss Vimontha Khieovongphachanh
Field of study	Electrical Engineering
Thesis Advisor	Assistant Professor Chedsada Chinrungrueng, Ph.D.

Accepted by the Faculty of Engineering, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

....Dean of the Faculty of Engineering
(Professor Direk Lavansiri, Ph.D.)

THESIS COMMITTEE

.... Chairman
(Associate Professor Somchai Jitapunkul, Dr.Ing.)

.... Thesis Advisor
(Assistant Professor Chedsada Chinrungrueng, Ph.D.)

.... Member
(Assistant Professor Manop Wongsaisuwan, D.Eng.)

.... Member
(Suvit Nakpeerayuth, M.Eng.)

วิมลธา เขียววงศ์พระจันทร์ : การประเมินเชิงประสบการณ์ของ อัลกอริทึมแอ็ฟไฟน์โปรเจคชัน สำหรับการกำจัดเสียงป้อนกลับในเครื่องช่วยฟัง (EMPIRICAL EVALUATION OF AFFINE PROJECTION ALGORITHM FOR ACOUSTIC FEEDBACK CANCELLATION IN HEARING AIDS) อ. ที่ปรึกษา: ผู้ช่วยศาสตรจารย์ ดร.เจษฎา ชินรุ่งเรือง, 54 หน้า. ISBN 974-53 -1522-2.

เครื่องช่วยฟังเป็นอุปกรณ์ที่ใช้สำหรับช่วยเพิ่มประสิทธิภาพทางการได้ยินของผู้ที่มีปัญหาในการรับรู้ข่าวสารทางเสียง ปัญหาหลักอย่างหนึ่งที่เกิดขึ้นกับผู้ใช้เครื่องช่วยฟังคือ การเกิดเสียงหอนอันเนื่องมาจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง วิธีการแก้ปัญหาดังกล่าวที่มีประสิทธิภาพและใช้กันโดยทั่วไปคือ การใช้วงจรกรองปรับตัวสร้างสัญญาณเลียนแบบสัญญาณป้อนกลับเพื่อหักล้างกับสัญญาณาป้อนกลับที่เกิดขึ้นจริง

อัลกอริทึมแอ็ฟไฟน์โปรเจคชันเป็นวงจรกรองปรับตัวแบบใหม่ อัลกอริทึมนี้มีคุณสมบัติอยู่ระหว่าง NLMS อัลกอริทึมและ RLS อัลกอริทึม ซึ่งมีความซับซ้อนน้อยกว่า RLS และมีอัตราการลู่เข้าที่เร็วกว่า NLMS อัลกอริทึม ทั้งนี้ขึ้นอยู่กับ projection order วิทยานิพนธ์ฉบับนี้ประเมินประสิทธิภาพของ อัลกอริทึมแอ็ฟไฟน์โปรเจคชัน ในแบบจำลอง fixed point พร้อมทั้งพิจารณาวิธีการอินเวอร์สเมตริกซ์อัตสัมพันธ์ด้วยระเบียบวิธี Levinson Durbin Recursion

ภาควิชา	วิศวกรรมไฟฟ้า	ลายมือชื่อนิสิต..
สาขาวิชา	วิศวกรรมไฟฟ้า	ลายมือชื่ออาจารย์ที่ปรึกษา......................................
ปีการศึกษา	2547	

4670638621: MAJOR ELECTRICAL ENGINEERING
KEY WORD: AFFINE PROJECTION ALGORITHM/FEEDBACK CANCELLATION/HEARING AIDS/RECURSIVE LEAST SQUARE ALGORITHM/LEAST MEAN SQUARE ALGORITHM

VIMONTHA KHIEOVONGPHACHANH, MISS.: EMPIRICAL EVALUATION OF AFFINE PROJECTION ALGORITHM FOR ACOUSTIC FEEDBACK CANCELLATION IN HEARING AIDS THESIS. ADVISOR: ASST. PROF. CHEDSADA CHINRUNGRUENG, Ph.D.54 pp. ISBN 974-53-1522-2.

A hearing aid equipment is used for amplifying the audio signal in order to enhance the hearing efficiency of people with hearing impairment. One major problem that hearing aids users usually encounter is screeching sound, which results from acoustic feedback in hearing aids. One effective and widely used solution for solving this problem is to employ an adaptive filter to produce signal for canceling out the acoustic feedback signal.

Affine Projection (AP) algorithm is a new adaptive filter. This algorithm has properties that lie between those of the normalized LMS algorithm and RLS algorithm. It has less complexity than RLS and much faster convergence than normalized LMS, which depends on the projection order. This thesis evaluates AP algorithm in fixed point arithmetic for determining number of bits, and also the use of Levinson Durbin Recursion for inverting the autocorrelation matrix.

Department Electrical Engineering Student’s signature…………………

Field of study Electrical Engineering Advisor’s signature…………………

Academic year 2004

Acknowledgements

This research was possible because of the help and support of numerous individuals. I wish to express my profound gratitude to all of them. I am greatly thankful to Assistant Professor Dr. Chedsada Chinrungrueng, the thesis advisor, for his continuous support, help and guidance throughout my time as his graduate student.

I deeply thank Associate Professor Dr. Somchai Jitapunkul (thesis chairman), Assistant Professor Dr. Manop Wongsaisuwan, and Mr Suvit Nakpeerayuth (thesis committee) for their valuable comments and suggestions.

I shall always be grateful to my family, for providing me with the best educational opportunities. They have been a source of incessant and encouragement, which have been vital to all of my accomplishments.

This research was enabled by the sponsorship of AUN/SEED-Net and JICA. I wish to express my sincere thanks to the AUN/SEED-Net and JICA for their financial support. I also thank International School of Engineering, Graduate School of Chulalongkorn University and Department of Electrical Engineering of Chulalongkorn University for their many helps. I also thank all of Electrical Engineering's staffs and friends in Digital Signal Processing laboratory.

I, particularly, thank Department of Electronic, Faculty of Engineering and Architecture, National University of Laos for giving the best opportunities to do this research through the AUN/SEED-Net and JICA project.

TABLES OF CONTENTS

Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

CHAPTER I

INTRODUCTION

Hearing aids are helping instrument for those who have a problem with hearing by increasing the volume of the sound. The magnitude of sound enters the ear. The basic component of hearing aids can be classified into microphone, amplifier, receiver and feedback path.

One factor that limits performance of hearing aids is acoustic feedback. It occurs when the aid's receiver produces an acoustic signal that leaks back to the microphone. Acoustic feedback is an important problem since it causes screeching sounds, which are greatly annoying to hearing aid user. So many researchers try to find the solution of acoustic feedback in hearing aid such as gain reduction or variation phase of acoustic feedback in hearing aid. Another way uses adaptive filter that generates the estimation of feedback path and subtracts out of the real feedback path. This paper focuses on the solution of acoustic feedback in hearing aids by using adaptive filter.

Adaptive filter measures the output signal of the filter, and compares it to a desired output signal dictated by the true system. By the observing the error between the output of adaptive filter and the desired output signal, an adaptation algorithm updates the filter coefficients with the aim to minimize an objective function. Figure1.1 shows the basic schematic diagram of an adaptive filter, where $x(n)$, $y(n)$, $d(n)$, $e(n)$ are input, output, desired response and error signals of the adaptive filter for time instant n.

In adaptation algorithm, the Least Mean Square (LMS) algorithm is widely used in adaptive filtering applications because of its simplicity in term of computation and implementation. However, the rate convergence is slow. On other hand, the Recursive Least Square (RLS) algorithm can significantly improve the convergence rate, but at a computational cost. It significantly requires more complex implementation.

Affine Projection (AP) algorithm is a new adaptive filter. This algorithm has properties that lie between those of the normalized LMS and RLS. It has less complexity than RLS but much faster convergence than normalized LMS as shown in figure 1.1. The compromise between

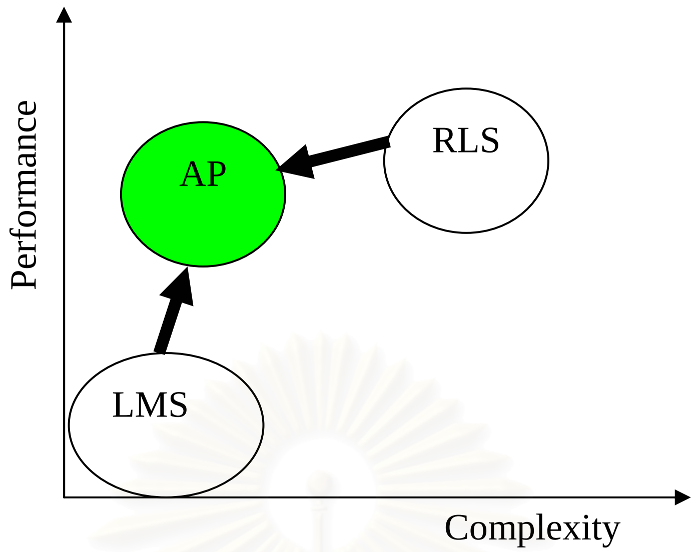


Figure 1.1 AP algorithm compare with LMS and RLS algorithm

RLS algorithm and normalized LMS algorithm depends on the projection order [1]. If the projection order equal to one, it produces the normalized LMS algorithm while projection order being equal to number of tap filter yields the windowed RLS algorithm, which means that has high flexible computational complexity.

The following of this thesis is organized as follows. In Chapter 2 we describe about hearing aids, acoustic feedback in hearing aids and some solution of acoustic feedback, including the model of acoustic feedback problem. In Chapter 3 we describe adaptive filter. The algorithms adapt the adaptive filter such as the Least Mean Square (LMS) algorithm, Recursive Least Square (RLS) algorithm, and Affine Projection (AP) algorithm, we compare the computational complexity of three algorithms and evaluate those algorithm in the problem of acoustic feedback in hearing aids. In Chapter 4 we determine the parameters of AP algorithm. How many projection orders that are used in the model of acoustic feedback problem. We also examine and evaluate of AP algorithm on floating point arithmetic and on fixed point arithmetic. In Chapter 5 we describe Levison Durbin recursion for inverting Toeplitz matrix for use in AP algorithm. This chapter will develop Autocorrelation matrix of AP algorithm that becomes to Toeplitz matrix. It also evaluates AP algorithm using inverting Toeplitz matrix which compares AP algorithm, and then evaluate AP in fixed point arithmetic. Finally, Chapter 6 concludes our work and the recommendation for the future.

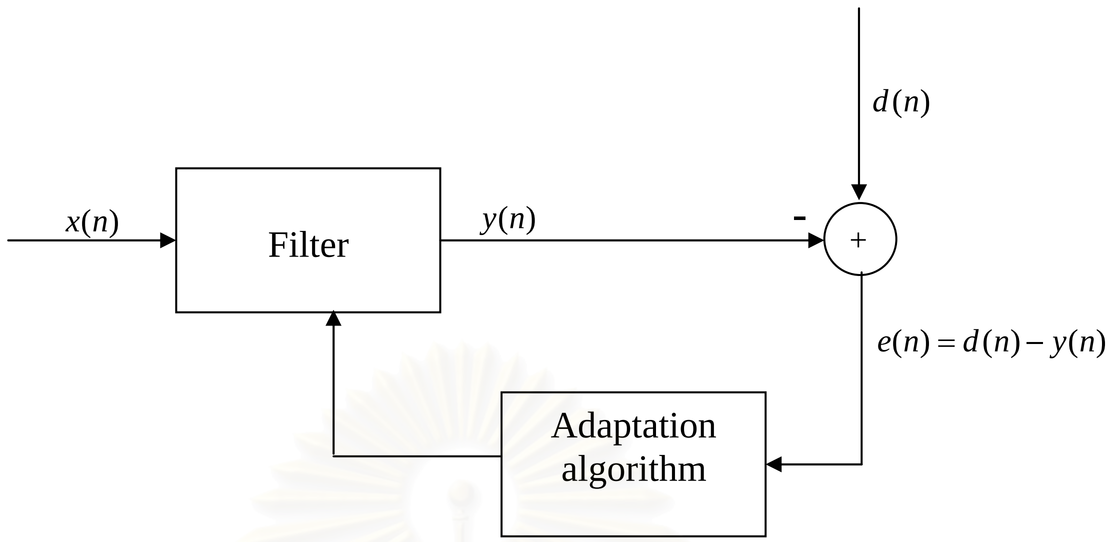


Figure 1.2 Block Diagram of an Adaptive Filter

1.1. OBJECTIVE

This paper is to empirically evaluate Affine Projection (AP) algorithm for acoustic feedback cancellation in hearing aids.

1.2. SCOPES OF RESEARCH

This research is based on simulation. This thesis uses MATLAB program to simulate non continuous adaptation in fixed point arithmetic. Its propose is to evaluate whether the AP algorithm is suitable for implementation in hardware with fixed point arithmetic.

1.3. RESEARCH PROCEDURE

- Study the paper that relates to reducing acoustic feedback in hearing aids.

- Study Affine Projection (AP) algorithm.

- Simulate AP algorithms.

- Study how to solve the problem of inversion.

- Simulate the AP algorithm using inverting toeplitz matrix.
- Simulate the AP algorithm using fixed point arithmetic.

1.4. CONTRIBUTION OF RESEARCH

This research will contribute to evaluating the practicality of implementing AP algorithm for acoustic feedback cancellation in hearing aids in hardware with fixed point arithmetic.

CHAPTER II

ACOUSTIC FEEDBACK IN HEARING AIDS

This chapter describes acoustic feedback in hearing aids. Section 2.1 describes hearing aids. The problem of hearing aids is described in section 2.2. Section 2.3 describes model of acoustic feedback problem in hearing aids.

2.1. HEARING AIDS

A hearing aid is a device consisting of a microphone, amplifier, and a receiver. The amplifier enhances a few or several frequencies depending on the needs of the user, and the receiver transmits the modified sounds to the middle ear.

Hearing impaired people usually have more hearing loss at some frequencies (pitch) than at others. Hearing aids therefore have to amplify more at some frequencies than at others. So internal hearing aids have to have more than one frequency as shown in figure2.1

Figure 2.1 shows block diagram of hearing aids. It has N channels, each channel has different frequency and also has different rate of amplifier. Amplification can change the volume. The levels of sound depend on hearing impaired. The adjustment of volume can control by hand or automatic control.

Another problem of hearing aids is noise signal (the sound is not wanted to listen such as the sound of air condition, children's sound, etc. those signal are amplified with original sound), this problem is different from the problem of acoustic feedback and the solution is also different. So this research chooses only the problem of acoustic feedback of hearing aids. The cause, the characteristic and the solution of the problem is described below.

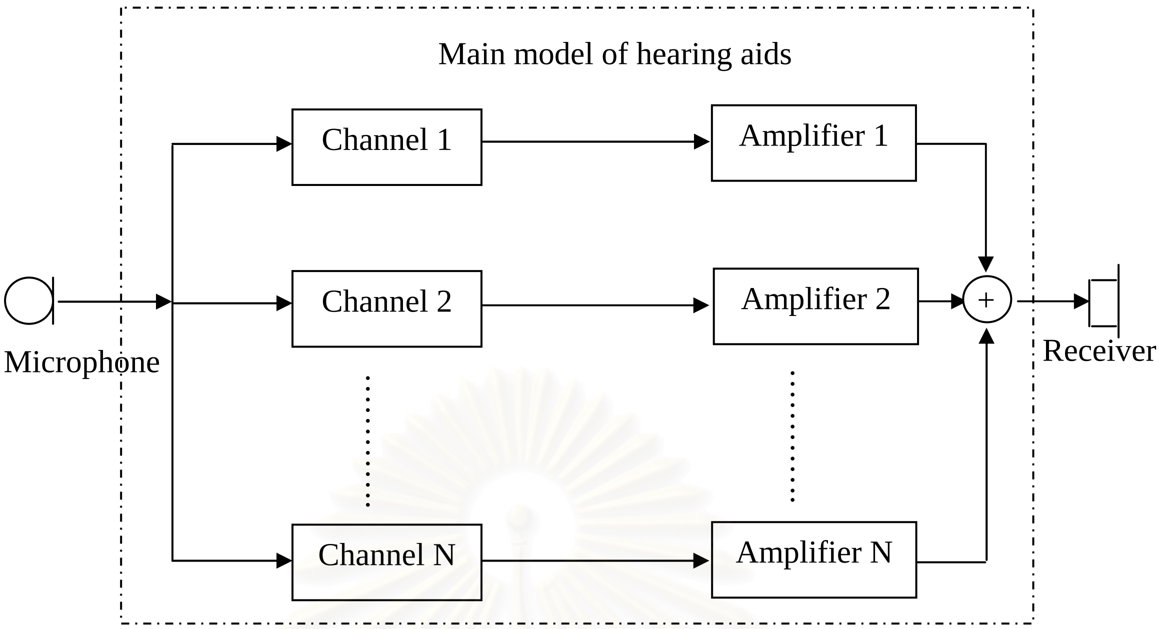


Figure 2.1 Block Diagram of Hearing Aids

2.2 ACOUSTIC FEEDBACK IN HEARING AIDS

Acoustic feedback in hearing aids occurs when the sound from receiver leaks back to microphone. It is called acoustic feedback as shown in figure 2.2 and the paths of acoustic feedback is called feedback path. Some of the feedback paths are as follows:

- Acoustics near the ear

Surfaces that may reflect sound, such as a telephone receiver or another person's head (as during a hug), can increase the likelihood of feedback.

- Leakage around the ear mold

A loose fitting or improperly formed ear mold or shell can create a path for sound to re-enter the microphone.

- Microphone /receiver distance

When the microphone and receiver are close, there is a greater likelihood that feedback will occur. This is especially true for smaller instrument styles.

- vent

While larger vents have the advantages of reducing occlusion, releasing low frequency energy, and introducing direct sound from the environment, they also create a path for feedback

Some solutions of feedback path are as follows:

- Taking an accurate impression and ensuring a tight fit of the ear mold or shell.

- Choosing appropriate vent sizes.

- Selecting an instrument appropriate for the amount of amplification needed.

Even through we can solve acoustic feedback using the aforementioned methods, there is still acoustic feedback. In [2], the solution of acoustic feedback in hearing aid can be divided into two parts

1. Suppression of feedback path

- Variation phase of feedback path is out of phase with incoming signal. In [3], they use Time Varying Delay, adding time varying delay in the signal path of hearing aids is generating a time varying phase response.
- Gain reduction, for protection of amplifying feedback rather than attenuating feedback. Example of this solution uses filter, it will decrease amplification for high frequency (inverse filtering [3], notch filter 4]).

2. Canceling of feedback path

- Separation between original sound and acoustic feedback. It can reduce noise signal by using more than one microphone, it enable a hearing aid to provide more amplification to sounds arriving from the front than to the sounds arriving from other directions(superdirective array[5]). This way can cancel feedback but it does not work well.

- Using adaptive filter that generates the estimation of feedback path and subtracts to the real feedback path (feedback cancellation [3], [4], [6]).

Suppression of feedback can not solve the problem well and the sound is distortion. This paper interests the solution of acoustic feedback in hearing aids by using adaptive filter. In [6] it divides adaptation in adaptive filter for canceling feedback into two types.

1. Continuous adaptation

Continuous adaptation continually adjusts the adaptive filter weight while simultaneously processing the input signal. The probe noise ($N_p(n)$) is always injected in the system .We can track the change of feedback path but it disturbs the user. It is shown in figure 2.3.

2. non continuous adaptation

When the normal signal path is broken and the filter is adapted or when a threshold change in gain is sensed so the probe noise ($N_p(n)$) is injected in the system very short. It does not disturb the user but we can not track the change of feedback at all time. It is shown in figure 2.4.

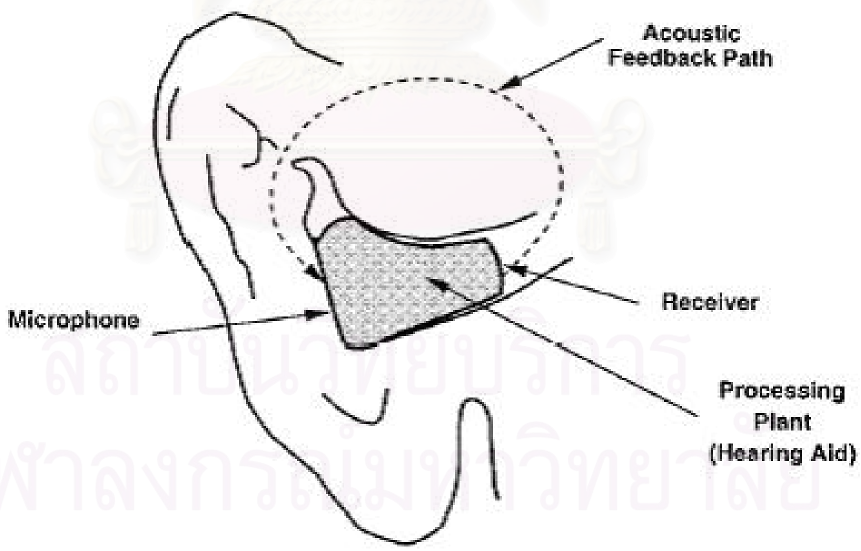


Figure 2.2 the Acoustic Feedback Path

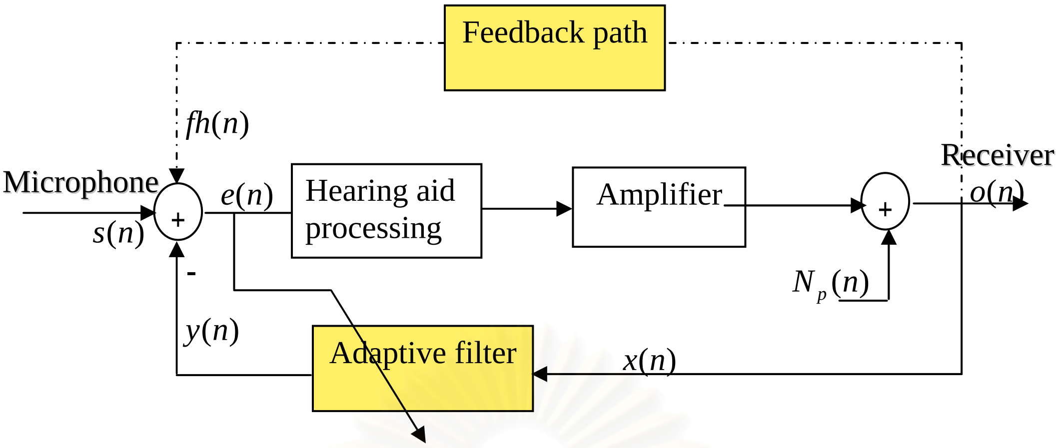


Figure 2.3 Block Diagram of a Continuous Adaptation Feedback Cancellation System

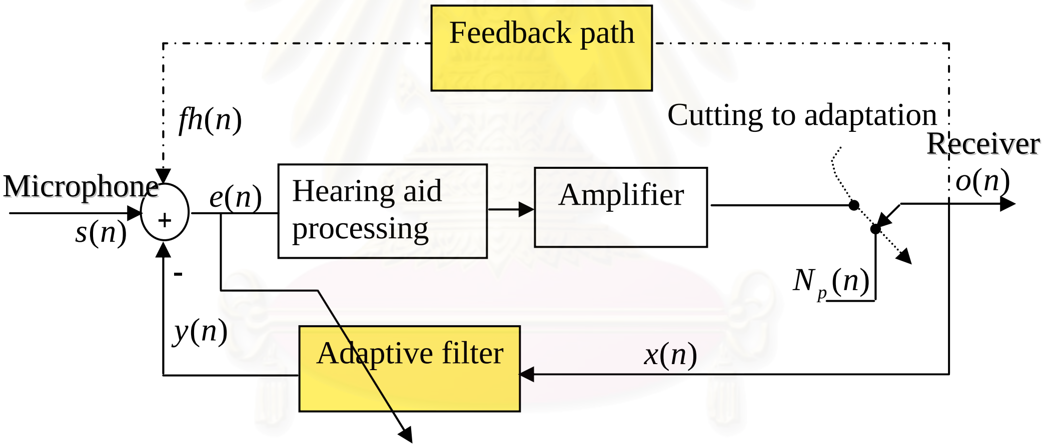


Figure 2.4 Block Diagram of a Non-continuous Adaptation Feedback Cancellation System

2.3 THE MODEL OF ACOUSTIC FEEDBACK PROBLEM IN HEARING AIDS

This research gives one channel and one amplifier in hearing aids; it decreases complication and facilitates computer simulation. Adaptation in adaptive filter uses non continuous adaptation as shown in figure 2.4.

This model uses impulse response of feedback path [2] that find from frequency response of feedback path [6], consists 32 taps. The impulse response as shown in figure 2.5

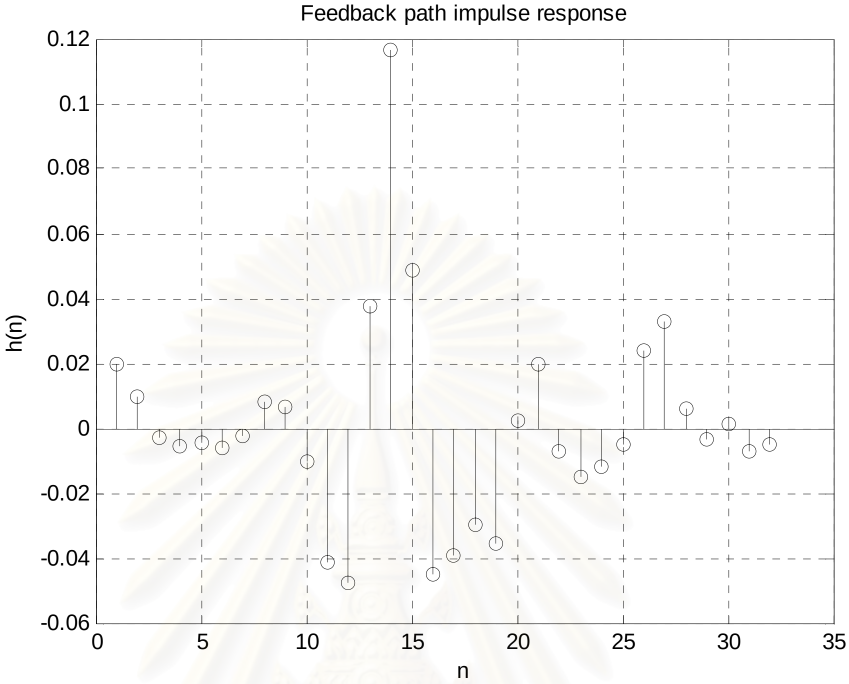


Figure 2.5 Impulse Response of Feedback Path

The solution of acoustic feedback considers non continuous adaptation as shown in figure2.4 and the symbol of signal as follow

$N_p(n)$, is random signal for help adjustment of adaptive filter, is zeros-mean uncorrelated Gaussian signal with variance 0.1

$s(n)$, is input signal of hearing aids, is zeros-mean uncorrelated Gaussian signal with variance 0.1 (the sound is soft)

$\vec{x}(n)$ is input vector of adaptive filter , the filter order of the filter is L

$$\vec{x}(n) = [x(n), x(n-1), ...x(n-L+1)]^T$$

$\vec{h}$ is feedback impulse response vector

$$\vec{h} = [h(0), h(1), ...h(L-1)]^T$$

$fh(n)$ is feedback signal in hearing aids

$$fh(n) = \vec{h}^T \vec{x}(n) \tag{2.1}$$

$\vec{w}(n)$ is tap weight vector of adaptive filter

$$\vec{w}(n) = [w_1(n), w_2(n), ..., w_L(n),]^T$$

$y(n)$ is the output signal of adaptive filter

$$y(n) = \vec{w}^T(n)\vec{x}(n) \tag{2.2}$$

$e(n)$ is error

$$e(n) = s(n) + fh(n) - y(n) \tag{2.3}$$

$o(n)$ is the output signal of hearing aids

$[.]^T$ is transpose

CHAPTER III

ADAPTIVE FILTER

This chapter describes the adaptive filter. Section 3.1 describes types of filters that use in this thesis. Section 3.2 describes adaptation algorithm, which has LMS algorithm, RLS algorithm, and AP algorithm. Section 3.3 shows the computational complexity of adaptation algorithm. Simulation of adaptation algorithm will be shown in section 3.4.

The problem of acoustic feedback in hearing aid uses adaptive filter. Adaptive filter has 2 parts. The first part is on structure of filters, and the second part is an adaptation algorithm.

3.1. FILTER

Filter is designed to produce an output in response to a sequence of input data. The filter can have either finite duration impulse response (FIR) or infinite duration impulse response (IIR).

This thesis uses Transversal Filter which is a finite memory response filter (FIR filter) with L coefficients as shown in figure3.1. Transversal filter consists of three basic elements: unit delay element, multiplier, and adder

The output of filter can be expressed as

$$y(n) = \vec{w}^T(n)\vec{x}(n) \qquad (3.1)$$

where

$\vec{w}(n) = [w_0(n), w_1(n), ..., w_{L-1}(n)]^T$ is the vector containing the coefficients of the adaptive filter.

$\vec{x}(n) = [x(n), x(n-1), ..., x(n-L+1)]^T$ is the vector containing the input of adaptive filter.

3.2. ADAPTATION ALGORITHMS

Adaptation algorithm updates the filter coefficients with the aim to minimize an objective function.

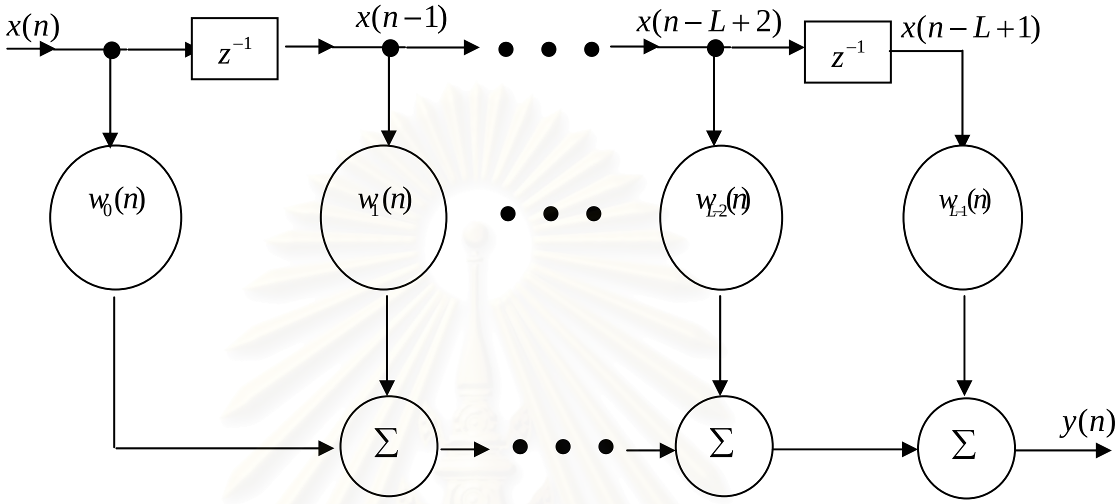


Figure 3.1 Filter Order L taps of Transversal Filter

This section introduces three adaptation algorithms that possess different qualities in term of the performance.

LMS algorithm to be described in section 3.2.1. Due to its low computational complexity, it still remains one of the most popular adaptive filtering algorithms. In section 3.2.2, we review the recursive least squares (RLS) algorithm, which is among the fastest adaptive filtering algorithm in term of convergence speed. The high computational complexity of the RLS algorithm can be significant in applications where the order of the adaptive filter is high. This inspired the development of algorithms with computational complexity some where in between those of the LMS and RLS algorithms. The affine projection (AP) algorithm [7] presented in section 3.2.3 utilizes the concept of reusing past information to improve the convergence speed.

3.2.1 LEAST MEAN SQUARE (LMS) ALGORITHMS

The Least Mean Square (LMS) [8] algorithm is probably the most widely used adaptive filtering algorithm. It consists of two basic steps where first the filter output $y(n)$ is computed and compared against a desired response $d(n)$ to determine the filter output error. This error is then used to adjust the filter parameter vector $\vec{w}(n)$. The LMS algorithm is shown in equations (3.2) to (3.4) where the filter input vector containing the latest input values is denoted by $\vec{x}(n)$, and μ is the step size parameter.

$$y(n) = \vec{w}^T(n)\vec{x}(n) \tag{3.2}$$

$$e(n) = d(n) - y(n) \tag{3.3}$$

$$\vec{w}(n+1) = \vec{w}(n) + \mu\vec{x}(n)e(n) \tag{3.4}$$

The stability and convergence properties of the algorithm are determined by the step size parameter. If μ is too large then the algorithm will not be convergent in the mean square. If, on the other hand, μ is too small, then the algorithm will be very slow.

The LMS algorithm is very efficient computationally and is of complexity $O(L)$. This is the main reason for its popularity in echo cancellation although it has been found to have a slower convergence rate than many other algorithms, such as RLS algorithm. When implementing the LMS algorithm on a computer with finite precision it is important that the step size is not chosen too small to prevent the stalling phenomenon from occurring.

3.2.2 RECURSIVE LEAST SQUARE (RLS) ALGORITHMS

To overcome the problem of slow convergence of the LMS algorithm, one can implement the recursive least squares (RLS) algorithm. The RLS algorithm is a recursive implementation of the least square (LS) solution, example it minimizes the LS objective function. The recursions for the most common version of the RLS algorithm, which is presented in its standard form in TABLE 1, is a result of the weighted least square (WLS) objective function $\varepsilon(n) = \sum_{i=1}^{n} \beta(n,i)\left|e(i)\right|^2$.

$e(n)$ is the difference between the desired response $d(n)$ and the output $y(n)$ produced by a transversal filter whose tap input (at time n) equal $x(n), x(n-1), ..., x(n-L+1)$ as in figure3.1.

$$e(n) = d(n) - y(n) \\ = d(n) - \vec{w}^T(n)\vec{x}(n) \tag{3.5}$$

$\vec{x}(n)$ is the tap-input vector at time n , defined by

$$\vec{x}(n) = [x(n), x(n-1), ..., x(n-L+1)]^T$$

$\vec{w}(n)$ is the tap-weight vector at time n , defined by

$$\vec{w}(n) = \left[w_0(n), w_1(n), ..., w_{L-1}(n)\right]^T$$

Weighting factor $\beta(n,i)$, $0 < \beta(n,i) \le 1$

Differentiating the objective function $\varepsilon(n)$ with respect to $\vec{w}(n)$ and solving for the minimum yields the following equation.

$$\left[\sum_{i=1}^{n} \lambda^{n-i} \vec{x}(i)\vec{x}^T(i)\right]\vec{w}(n) = \sum_{i=1}^{n} \lambda^{n-i} \vec{x}(i) d^*(i)$$

where $0 < \lambda \le 1$ is an exponential scaling factor often referred to as the forgetting factor.

Defining the quantities:

$$\phi(n) = \sum_{i=1}^{n} \lambda^{n-i} \vec{x}(i)\vec{x}^T(i)$$

and

$$z(n) = \sum_{i=1}^{n} \lambda^{n-i} \vec{x}(i) d^*(i)$$

the solution is obtained as

$$\vec{w}(n) = \phi^{-1}(n) z(n)$$

The recursive implementation is a result of the formulations

$$\phi(n) = \lambda\phi(n-1) + \vec{x}(n)\vec{x}^T(n)$$

and

$$z(n) = \lambda z(n-1) + \vec{x}(n) d^*(n)$$

The inverse $\phi^{-1}(n)$ can be obtained recursively in terms of $\phi^{-1}(n-1)$ using *the matrix inversion lemma* thus avoiding direct inversion of $\phi(n)$ at each time instant n. The main problems with the RLS algorithm are potential divergence behavior in finite precision environment and high computational complexity, which is of order L^2 or $O(L^2)$.

From Matrix inversion lemma we can write

$$\phi^{-1}(n) = \lambda^{-1}\phi^{-1}(n-1) - \frac{\lambda^{-2}\phi^{-1}(n-1)\vec{x}(n)\vec{x}^{T}(n)\phi^{-1}(n-1)}{1+\lambda^{-1}\vec{x}^{T}(n)\phi^{-1}(n-1)\vec{x}(n)} \tag{3.6}$$

For convenience of computation let be

$$p(n) = \phi^{-1}(n)$$

and

$$k(n) = \frac{\lambda^{-1}p(n-1)\vec{x}(n)}{1+\lambda^{-1}\vec{x}^{T}(n)p(n-1)\vec{x}(n)} \tag{3.7}$$

we may write Equation (3.6) as follows:

$$p(n) = \lambda^{-1}p(n-1) - \lambda^{-1}k(n)\vec{x}^{T}(n)p(n-1) \tag{3.8}$$

$p(n)$: Inverse correlation matrix L by L

$k(n)$: The gain vector L by 1

We get the desired recursive equation for updating the tap weight vector

$$\begin{aligned}\vec{w}(n) &= \vec{w}(n-1) + k(n)\left[d^{*}(n) - \vec{x}^{T}(n)\vec{w}(n-1)\right] \\ &= \vec{w}(n-1) + k(n)\xi^{*}(n)\end{aligned} \tag{3.9}$$

$\xi^{*}(n)$: a priori estimation error defined by

$$\begin{aligned}\xi(n) &= d(n) - \vec{x}^{T}(n)\vec{w}(n-1) \\ &= d(n) - \vec{w}^{T}(n-1)\vec{x}(n)\end{aligned} \tag{3.10}$$

TABLE 3.1: the Recursive Least Square algorithm

Initialize the algorithm by setting

$p(0) = \delta^{-1} I,$ δ = small positive constant

$w(0) = 0$

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

$$k(n) = \frac{\lambda^{-1} p(n-1)\vec{x}(n)}{1 + \lambda^{-1}\vec{x}^{T}(n)\, p(n-1)\vec{x}(n)}$$

$$\xi(n) = d(n) - w^{T}(n-1)\vec{x}(n)$$

$$\vec{w}(n) = \vec{w}(n-1) + k(n)\xi^{*}(n)$$

$$p(n) = \lambda^{-1} p(n-1) - \lambda^{-1} k(n)\vec{x}^{T}(n)\, p(n-1)$$

Algorithms whose convergence rate and computational complexity are somewhere between those of the LMS and RLS algorithms are considered in the following section.

3.2.3 AFFINE PROJECTION (AP) ALGORITHM

Affine Projection algorithm is a new adaptive filtering algorithm, which includes RLS like convergence and normalized LMS like complexity. The Affine Projection (AP) algorithm [7] is a generalization of the well known Normalized Least Mean Square (NLMS) adaptive filtering algorithm.

The tap weight ($\underline{w}_n$) update equation of the affine projection algorithm is shown in equation (3.11). Assume that the filter order of the filter is L and the degree of projection order is N.

$$\underline{w}_n = \underline{w}_{n-1} + \mu X_n \underline{\varepsilon}_n \qquad (3.11)$$

X_n is a L by N matrix and has the structure

$X_n = \left[\underline{x}_n, \underline{x}_{n-1}, ...\underline{x}_{n-N+1}\right] = \left[\overline{x}_n, \overline{x}_{n-1}, ...\overline{x}_{n-L+1}\right]^t,$

where $\underline{x}_n = [x_n, ..., x_{n-L+1}]^t$ is input signal vector and $\underline{\bar{x}}_n = [x_n, ..., x_{n-N+1}]^t$

μ is the step size

$\underline{\varepsilon}_n$ is the normalized error vector, $\underline{\varepsilon}_n = [\varepsilon_n, ..., \varepsilon_{n-N+1}]^t_{N\times 1}$

$R_N = \left[X^t_n X_n + \delta I\right]^{-1}$, is autocorrelation matrix, N by N

The normalized error ($\underline{\varepsilon}_n$) is calculated as

$$\underline{\varepsilon}_n = \left[X^t_n X_n + \delta I\right]^{-1} \underline{e}_n = R_N \underline{e}_n \tag{3.12}$$

δ is convergence parameter

I is an N by N identity matrix

$\underline{e}_n$ is the error vector, $\underline{e}_n = [e_n, ..., e_{n-N+1}]^t_{N\times 1}$

$$\underline{e}_n = \underline{d}_n - X^t_n \underline{w}_{n-1} \tag{3.13}$$

The N dimension vector, $\underline{d}_n$, is a desired response consisting of the response of the echo path impulse response, $\underline{h}_{ep}$ to the input and the additive system noise, $\underline{s}_n$

$$\underline{d}_n = X^t_n \underline{h}_{ep} + \underline{s}_n \tag{3.14}$$

$\underline{h}_{ep}$ is a L by 1 vector

The scalar δ is the convergence parameter for the sample autocorrelation matrix inversion used in (3.12) in the calculation of the N dimension normalized error vector, $\underline{\varepsilon}_n$, where $X^t_n X_n$ may have eigenvalues close to zero, creating problems for the inverse. The matrix $X^t_n X_n + \delta I$ has δ as its smallest eigenvalue which, if large enough, yields a well behaved inverse. The step size parameter μ is the relation factor. As in NLMS, the algorithm is stable for $0 \le \mu < 2$.

If N is set to one, equation (3.11), (3.12), (3.13) reduce to the familiar NLMS algorithm thus, AP algorithm is a generalization of NLMS.

3.3 COMPUTATIONAL COMPLEXITY OF ADAPTATION ALGORITHMS

From above three (LMS, RLS, AP) algorithms, we can compare computational complexity in one iteration cycle, N is projection order; L is number of tap weight. K_{inv} is a constant associated with the complexity of the inverse required in equation (3.12). The lowest computational complexity is LMS algorithm in TABLE 3.2. The highest computational complexity is RLS algorithm in TABLE 3.3. The computational complexity of AP algorithm link between LMS algorithm and RLS algorithm as shown in TABLE 3.4, it is depend on the projection order.

According to computational complexity of adaptation algorithms, in the real time, as shown in TABLE 3.5, AP algorithm gives various projection orders.

TABLE 3.2: Computational complexity of LMS algorithm

LMS algorithm	The number of additions/subtraction	The number of multiplication
$y(n)=\vec{w}^T(n)\vec{x}(n)$	$L-1$	L
$e(n)=d(n)-y(n)$	1	
$\vec{w}(n+1)=\vec{w}(n)+\mu\vec{x}(n)e(n)$	$2L$	L
Total computational complexity	$3L$	$2L$

TABLE 3.3: Computational complexity of RLS algorithm

RLS algorithm	The number of additions/subtraction	The number of multiplication
$k(n)=\dfrac{\lambda^{-1}p(n-1)\vec{x}(n)}{1+\lambda^{-1}\vec{x}^T(n)p(n-1)\vec{x}(n)}$	L^2+2L	L^2
$\xi(n)=d(n)-w^T(n-1)\vec{x}(n)$	L	L
$\vec{w}(n)=\vec{w}(n-1)+k(n)\xi^*(n)$	L	L
$p(n)=\lambda^{-1}p(n-1)-\lambda^{-1}k(n)\vec{x}^T(n)p(n-1)$	$4L^2$	L^2-L+1
Total computational complexity	$5L^2+4L$	$2L^2+L+1$

TABLE 3.4: Computational complexity of AP algorithm

AP algorithm	The number of additions/subtraction	The number of multiplication
$\underline{e}_n = \underline{d}_n - X_n^t \underline{w}_{n-1}$	LN	LN
$\underline{\varepsilon}_n = \left[X_n^t X_n + \delta I \right]^{-1} \underline{e}_n$	$N^2 - N$	$K_{inv} N^2$
$\underline{w}_n = \underline{w}_{n-1} + \mu X_n \underline{\varepsilon}_n$	N	$LN + L$
Total computational complexity	$LN + N^2$	$2LN + K_{inv} N^2 + L$

TABLE 3.5 Computational complexity in the real time

Complexity / algorithms	The real time (second)/iteration				
Least Mean Square (LMS)	0.0002				
Recursive Least Square (RLS)	0.0004				
Affine Projection (AP)	$N = 2$	$N = 9$	$N = 10$	$N = 32$	$N = 40$
	0.0002	0.0002	0.0004	0.0008	0.0014

3.4 SIMULATION

Affine Projection algorithm is applied to the problem of hearing aids and uses the model of acoustic feedback problem in section 2.3 of chapter 2. If the projection order increases, the rate of convergence is very fast but it has more complexity than normalized LMS as shown in Figure 3.2. In Figure 3.3 comparing with RLS algorithm, LMS algorithm and AP algorithm (the projection order equal to 5), the rate convergence of AP algorithm closes to the rate convergence of RLS algorithm and less complexity. AP algorithm is useful for applying to acoustic feedback problem, if we wish to use AP algorithm to implement hardware of adaptive filter of hearing aids, we must consider the fixed point model of AP algorithm. Because the real number can not be used as the signal in hardware, we have to quantize it then convert to binary. Therefore, before we implement hardware, we have to simulate AP algorithm by fixed point arithmetic. It is presented in chapter 4.

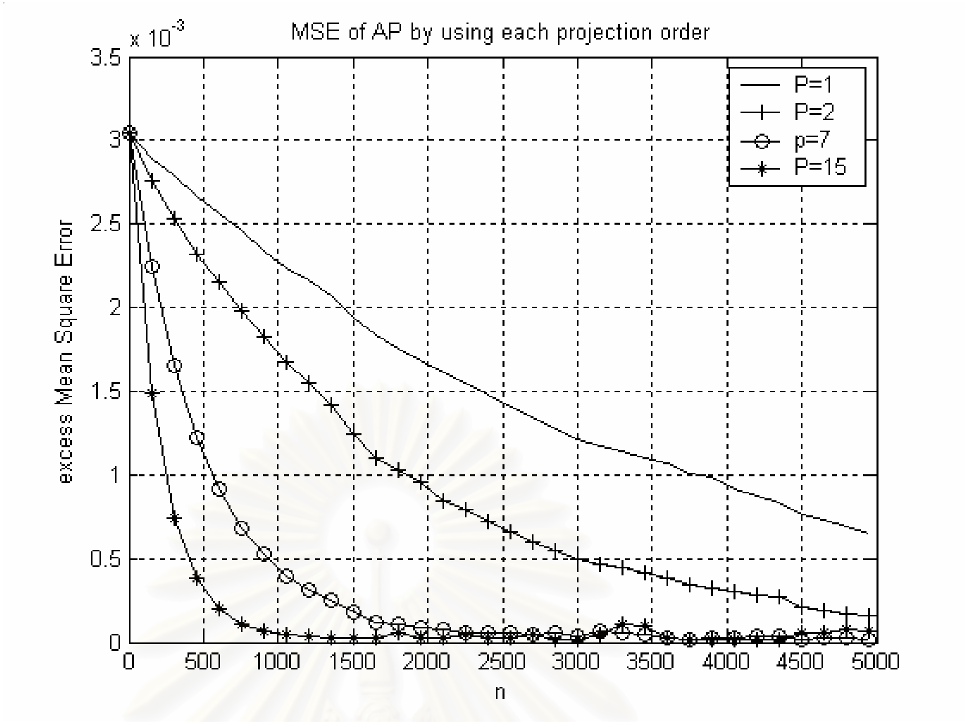


Figure 3.2 Graph of AP algorithms with each Projection Order

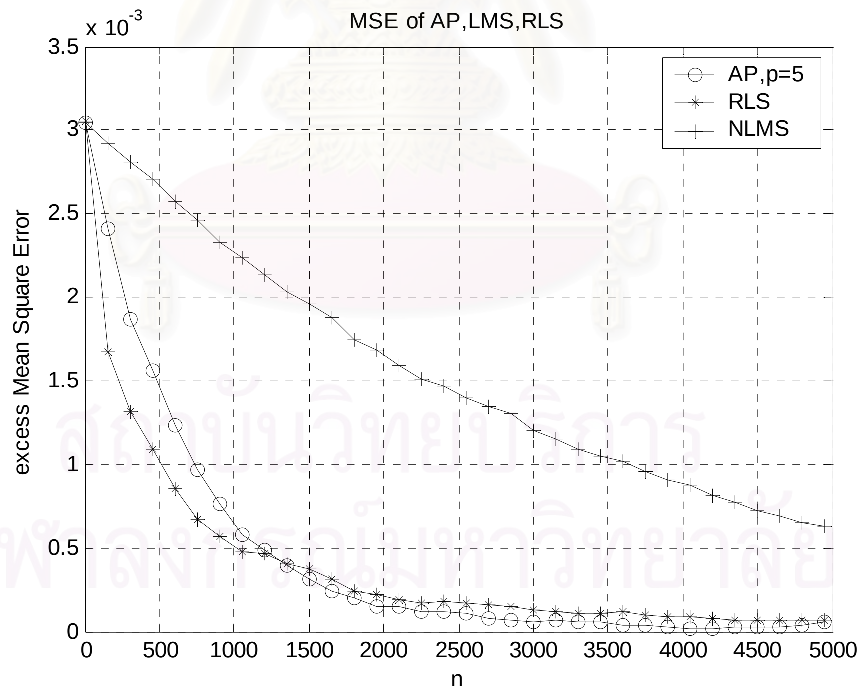


Figure 3.3 Graph of the Convergence Rate with Three Adaptation algorithms

CHAPTER IV

APPLICATION OF AFFINE PROJECTION (AP) ALGORITHM IN HEARING AIDS

This chapter will use AP algorithm to cancel acoustic feedback in hearing aids because it get both low computational complexity and fast convergence rate. However, it depends on the projection order. In section 4.1 we determine the projection order of AP algorithm to acoustic feedback problem in hearing aid. In section 4.2, we determine the number of bits.

4.1. DETERMINATION OF PARAMETER

Acoustic feedback solution in hearing aids uses Affine Projection (AP) algorithm. The block diagram of AP algorithm is shown in figure 4.1, x_n is input signal. AP algorithm has properties that lie between those of the normalized LMS algorithm and RLS algorithm; it has less complexity than RLS but much faster convergence than normalized LMS as shown in chapter 3. But we have to determine how many projection orders that we use to get low computational complexity and good performance.

This thesis examines many projection orders. We use the model of acoustic feedback problem in section 2.3 of chapter 2. Figure 4.2 compares various projection orders (N =1, N = 2, N = 4, N = 8), plot excess Mean Square Error with step size $(\mu) = 0.005$, and the number of iteration is 5000. Each curves the average of 100 trials. While the projection order is increases, the convergence is fastest. Initially, we test to see whether AP algorithm is works in floating point arithmetic. If we wish to use AP algorithm to implement on hardware in hearing aids, we must consider the fixed point model of AP algorithm. Because the real number can not be used as the signal in hardware, we have to quantize the number and convert to binary. So, before we implement hardware, we have to simulate AP algorithm by fixed point arithmetic. We also have to determine how many bits that we use because if we choose wrong number of bits, the Mean Square Error will diverge instead converge. So, section4.2 calculates resolution of data and compares which bit is stable converge like floating point arithmetic.

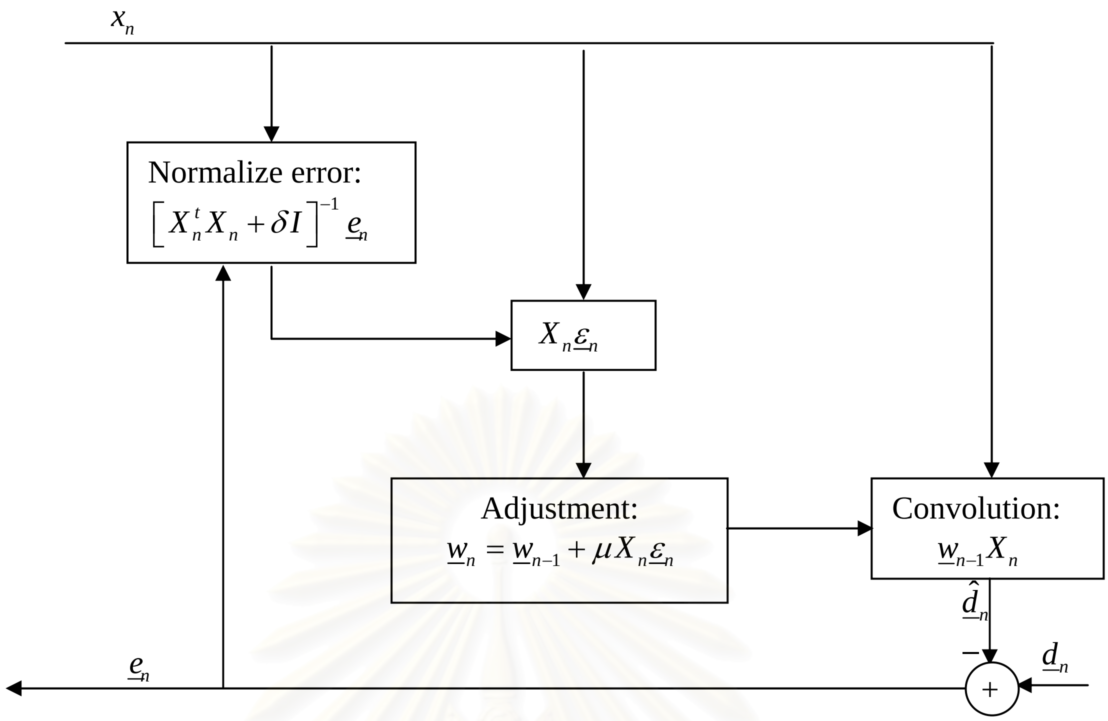


Figure 4.1 Block Diagram of AP algorithm [9]

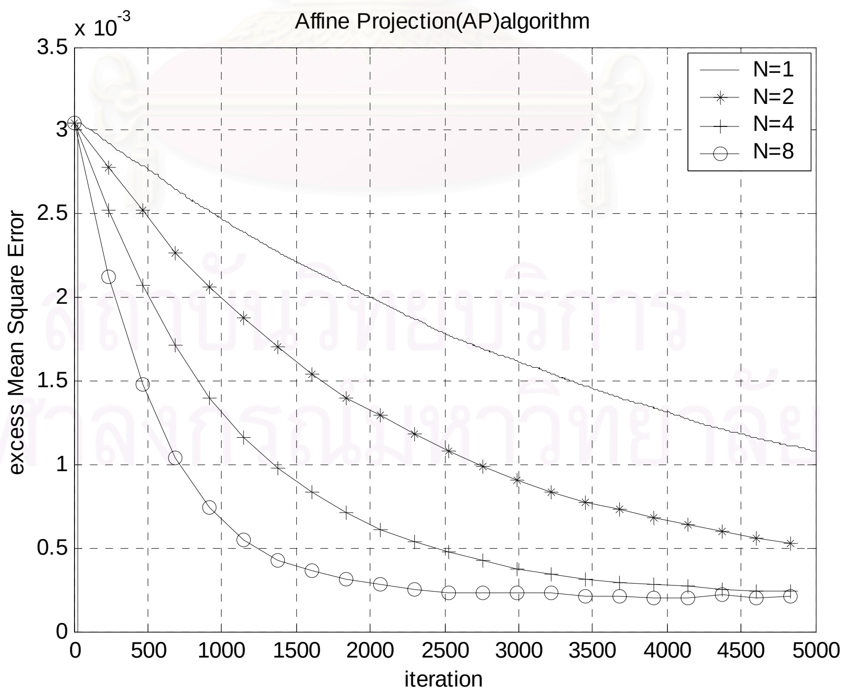


Figure 4.2 Convergence of (AP) algorithm with 100 trials

4.2. RESOLUTION OF DATA

Determination of number of bits as shown in figure 4.3 is for adaptive filter applied to acoustic feedback problem in hearing aids, and the maximum of each variable are shown in TABLE 4.1.

We use 10 bits of the input (x_n) and 19bits of the output (y_n), the step size ($\mu = 0.005$) is represented 10 bits where the sign bit is denoted by left most bit. Next position is Most Significant Bit (MSB), which is 2^{-8} and the right most bit is referred to as the Least Significant Bit (LSB), the position is 2^{-16}. Other positions are shown in figure 4.4

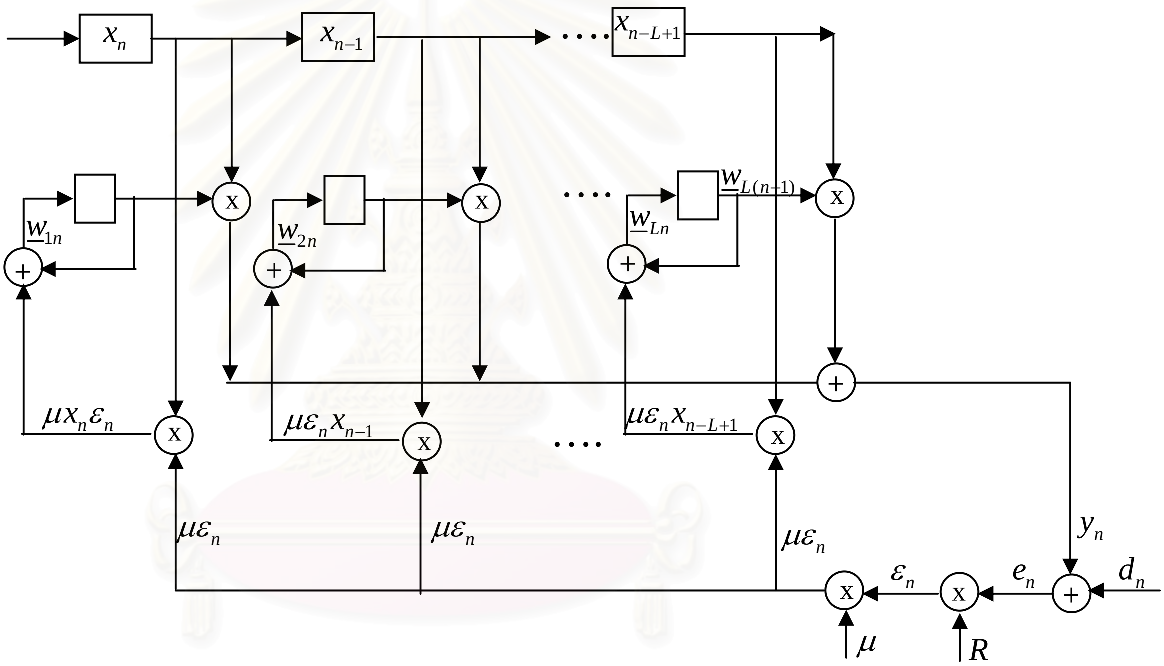


Figure 4.3 Adaptive Filter for Determination of Number of Bits

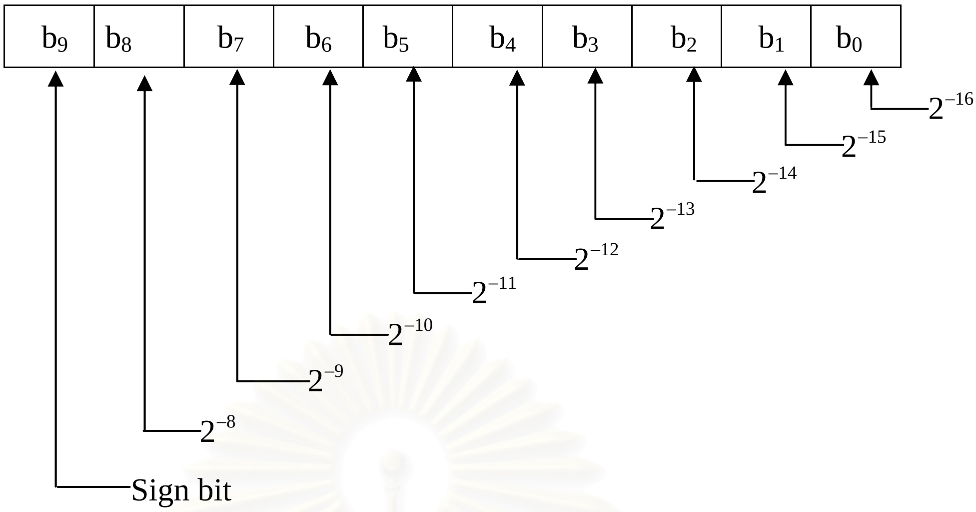


Figure 4.4 Data of Step Size

The tap weight of adaptive filter ($\underline{w}_n$) is represented by 19 bits, where every bit is shown in figure 4.5. (MSB is 2^{-4} and LSB is 2^{-21})

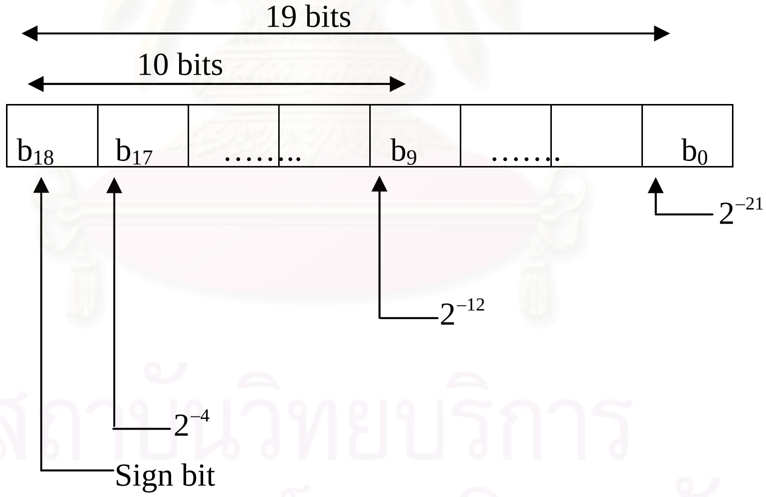


Figure 4.5 Data of Tap Weight

The input signal (x_n) is represented by 10 bits, where every bit is shown in figure 4.6. (MSB is 2^{-1} and LSB is 2^{-9}).

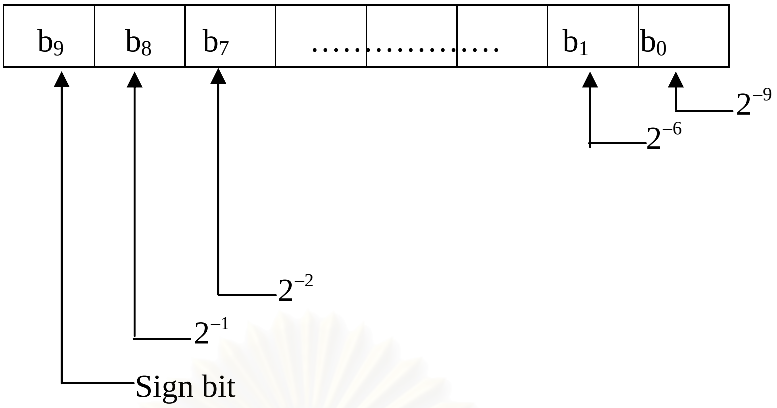


Figure 4.6 Data of Input Signal

The output of adaptive filter (y_n) is multiplication between tap weight (use only 10 bits) and input signal. It is represented by 19 bits as shown in figure 4.7.

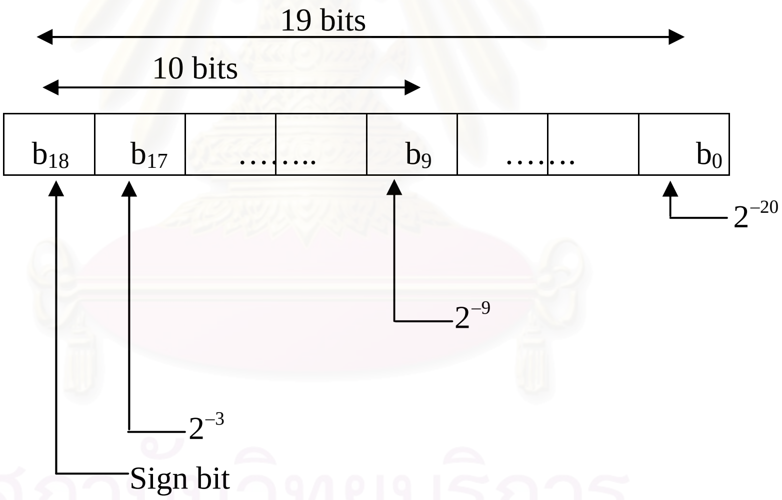


Figure 4.7 Data of Output

Desired response (d_n) is represented 10 bits, which is shown in figure 4.8. (MSB is 2^{-1} and LSB is 2^{-9}).

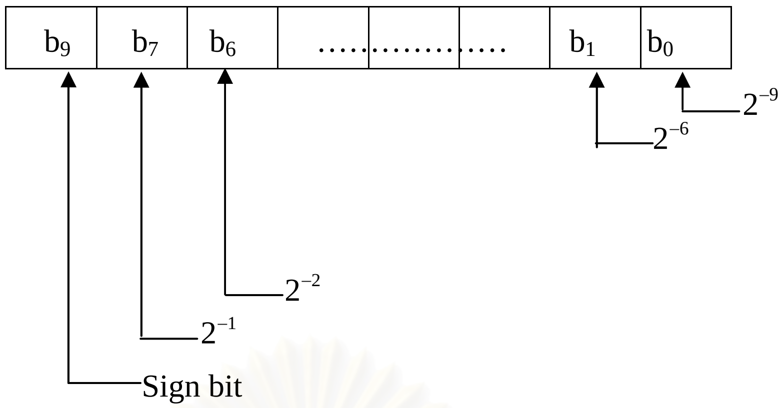


Figure 4.8 Data of Desired Response

Addition between the out put (y_n) and desired response (d_n) has to be the same bit and have the same position. So y_n is truncated the first most significant bit 7 bits of y_n and y_n extends the sign bit 3 bits (ext). The new output uses 10 bits as shown in figure 4.9

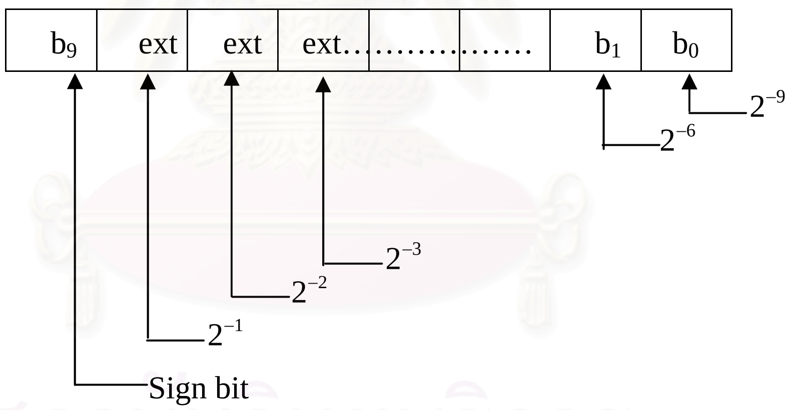


Figure 4.9 Data of New Output

From figure 4.3, normalized error (ε_n) multiplies between error (e_n) and inversion of auto correlation (R_N). It is shown in figure 4.10 and used only 10 bits for next step

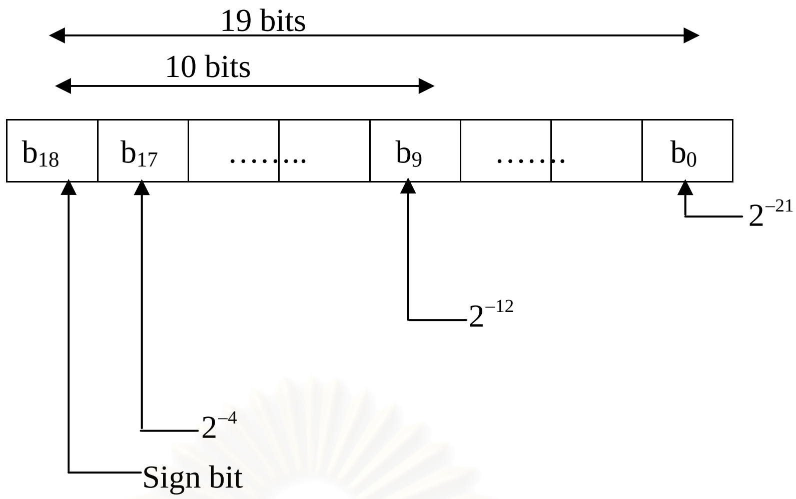


Figure 4.10 Data of Normalized Error

$\mu\varepsilon$ is multiplication between normalized error and step size , it is shown in figure 4.11and it is used only 10 bits for multiplication with input signal.

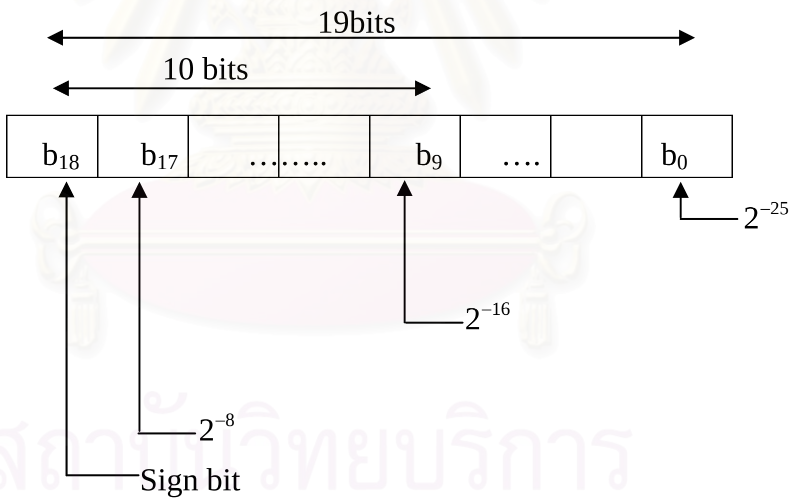


Figure 4.11 Data of $\mu\varepsilon$

$\mu\varepsilon x$ is multiplication between $\mu\varepsilon$ and input signal, it is shown in figure 4.12 . $\mu\varepsilon x$ has to be the same position with tap weight. So $\mu\varepsilon x$ is truncated the first most significant bit of 15 bits and it extends sign bit 4 bits (ext), the new $\mu\varepsilon x$ is shown in figure 4.13

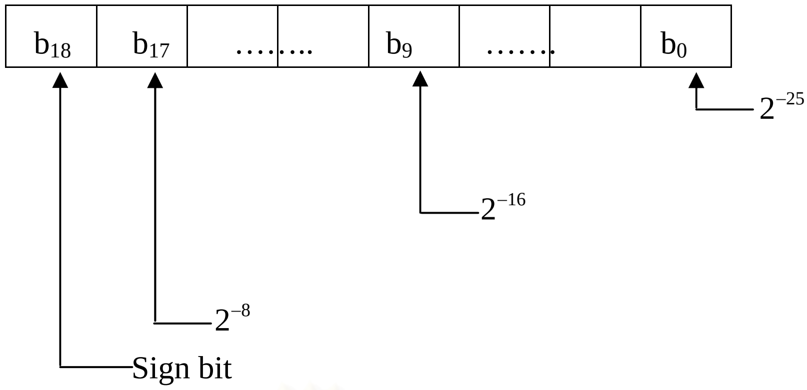


Figure 4.12 Data of $\mu \varepsilon x$

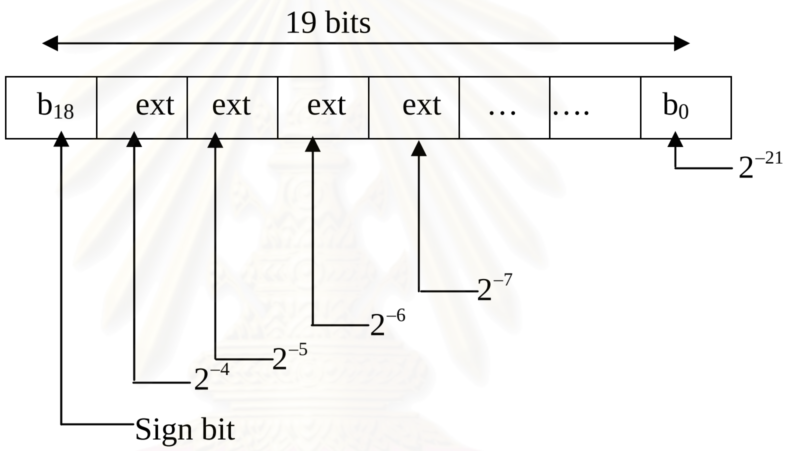


Figure 4.13 Data of new $\mu \varepsilon x$

TABLE 4.1 the Sizes of Signal in Adaptive Filter

Signals	Sizes of signal
input $x(n)$	using number bits :10 bits minimum value = -1.3329 maximum value =1.3329
tap weight $w(n)$	using number bits :10 bits minimum value = - 0.045131 maximum value =0.084209
output $y(n)$	using number bits :10 bits

	minimum value = -0.13486 maximum value =0.13486
desired response $d(n)$	using number bits :10 bits minimum value = -1.2399 maximum value =1.2399
error $e(n)$	using number bits :10 bits minimum value = -1.183 maximum value =1.183
step size μ	using number bits :10 bits value =0.005
normalized error $\varepsilon(n)$	using number bits :10 bits minimum value = -1.183 maximum value =0.40775
inversion of auto correlation R	using number bits :10 bits minimum value = 0 maximum value =1
multiplication of $\mu\ \varepsilon(n)$	using number bits :10 bits minimum value = -0.0018047 maximum value =0.0020388
multiplication of $\mu\ \varepsilon(n)x(n)$	using number bits :19 bits minimum value = -0.0014792 maximum value =0.0014792

Now we can compare floating point arithmetic and fixed point arithmetic that the projection order set to one, two, and four in section 4.2.1, section 4.2.2, and section 4.2.3 respectively.

4.2.1. PROJECTION ORDER SET TO ONE

Projection order set to one, equation(3.11), (3.12), (3.13) and (3.14) become Normalized LMS (NLMS) algorithm

Input signal, L by 1 vector

$$X_n = \left[x_n, \ldots., x_{n-L+1}\right]^t$$

$R_n = \left[X_n^t X_n + \delta I\right]^{-1}$ is the inversion of autocorrelation

error (e_n) is scalar
normalized error (ε_n) is scalar
desired response(d_n) is scalar

We evaluated AP algorithm in floating point arithmetic by one projection order as shown in figure 4.14. Then, we evaluate in fixed point arithmetic (the input is 8 bits, 9 bits, 10 bits, and 13 bits) as shown in figure 4.15. While the input gives 10 bits up to 13 bits, the excess Mean Square Error converge the same as floating point arithmetic.

4.2.2. PROJECTION ORDER SET TO TWO

The projection order equal to two ($N = 2$).

Input signal, L-by-2 matrix

$$X_n = \begin{bmatrix} x_n & x_{n-1} \\ x_{n-1} & x_{n-2} \\ . & . \\ . & . \\ x_{n-L+1} & x_{n-L} \end{bmatrix}_{L\times 2}$$

$$R_n = \left[X_n^t X_n + \delta I\right]^{-1} = \begin{bmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{bmatrix}$$ is the inversion of auto correlation

$\underline{e}_n = [e_n, e_{n-1}]^t_{2\times 1}$: error vector
$\underline{\varepsilon}_n = [\varepsilon_n, \varepsilon_{n-1}]^t_{2\times 1}$: normalized error
$\underline{d}_n = [d_n, d_{n-1}]^t_{2\times 1}$: desired response

We evaluated AP algorithm in floating point arithmetic by two projection orders as shown in figure 4.16 then, we evaluated in fixed point arithmetic (the input is 8 bits, 9 bits, 10 bits, and 13 bits) as shown in figure 4.17. While the input gives 10 bits up to 13 bits, the excess Mean Square Error converge the same as floating point arithmetic.

4.2.3. PROJECTION ORDER SET TO FOUR

Resolution of data is the same projection order that equal to two but we have four errors, four outputs and four normalized errors

Input signal, L-by-4 matrix

$$X_n = \begin{bmatrix} x_n & x_{n-1} & x_{n-2} & x_{n-3} \\ x_{n-1} & x_{n-2} & x_{n-3} & x_{n-4} \\ . & . & . & . \\ . & . & . & . \\ x_{n-L+1} & x_{n-L} & x_{n-L-1} & x_{n-L-2} \end{bmatrix}_{L\times 4}$$

$$R_n = \left[X_n^t X_n + \delta I \right]^{-1}$$

$$= \begin{bmatrix} R_{11} & R_{12} & R_{13} & R_{14} \\ R_{21} & R_{22} & R_{23} & R_{23} \\ R_{31} & R_{32} & R_{33} & R_{34} \\ R_{41} & R_{42} & R_{43} & R_{44} \end{bmatrix}$$ is the inversion of auto correlation

$\underline{e}_n = [e_n, .., e_{n-3}]^t_{4\times 1}$: error vector

$\underline{\varepsilon}_n = [\varepsilon_n, .., \varepsilon_{n-3}]^t_{4\times 1}$: normalized error vector

$\underline{d}_n = [d_n, .., d_{n-3}]^t_{4\times 1}$: desired response vector

We evaluated AP algorithm in floating point arithmetic by four projection orders as shown in figure 4.18 then, we evaluated in fixed point arithmetic (the input is 8 bits, 9 bits, 10 bits, and 13 bits) as shown in figure 4.19. While the input gives 10 bits up to 13 bits, the excess Mean Square Error converge the same as floating point arithmetic. On floating point arithmetic, the iteration is around 3000, which has a little bit fluctuation. Therefore, on fixed point arithmetic. It has much fluctuation because the input is not enough to use it (the input is 8bits).

4.2.4. PROJECTION ORDER SET TO EIGHT

Resolution of data is the same projection order equal to two but we have eight errors, eight outputs, eight normalized errors

Input signal, L-by-8 matrix

$$X_n = \begin{bmatrix} x_n & x_{n-1} & . & . & x_{n-7} \\ x_{n-1} & x_{n-2} & . & . & x_{n-8} \\ . & . & & & \\ . & . & & & \\ x_{n-L+1} & x_{n-L} & . & . & x_{n-L-6} \end{bmatrix}_{L\times 8}$$

$$R_n = \left[X_n^t X_n + \delta I \right]^{-1}$$

$$= \begin{bmatrix} R_{11} & . & . & R_{18} \\ . & & & . \\ . & & & . \\ R_{81} & . & . & R_{88} \end{bmatrix}$$ is the inversion of auto correlation

$\underline{e}_n = [e_n, .., e_{n-7}]^t_{8\times 1}$: error vector

$\underline{\varepsilon}_n = [\varepsilon_n, .., \varepsilon_{n-7}]^t_{8\times 1}$: normalized error vector

$\underline{d}_n = [d_n, .., d_{n-7}]^t_{8\times 1}$: desired response vector

We evaluated AP algorithm in floating point arithmetic by eight projection orders as shown in figure 4.20 then, we evaluated in fixed point arithmetic (the input is 8 bits, 9 bits, 10 bits, and 13 bits) as shown in figure 4.21. While the input gives 10 bits up to 13 bits, the excess Mean Square Error converge the same as floating point arithmetic. On floating point arithmetic, the iteration is around 3000, which has a little bit fluctuation. Therefore, on fixed point arithmetic. It has much fluctuation because the input is not enough to use it (the input is 8bits).

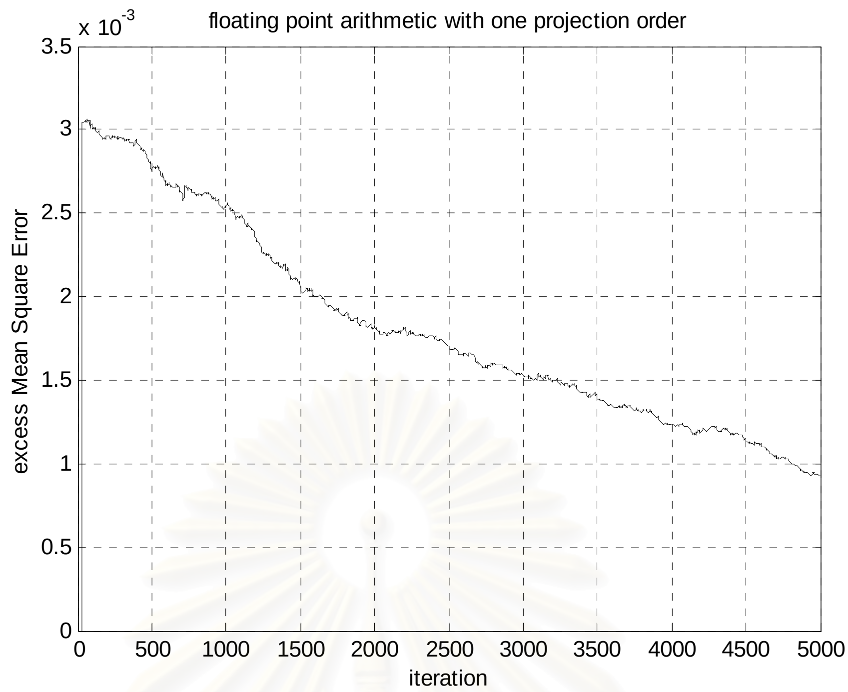


Figure 4.14 Floating Point Arithmetic with Projection Order Set to One

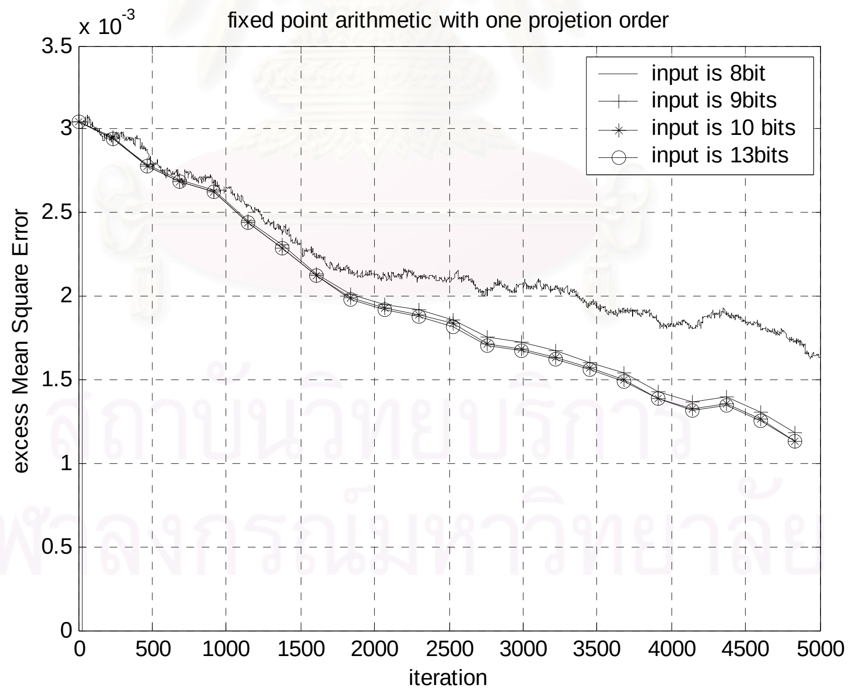


Figure 4.15 Fixed Point Arithmetic with Projection Order Set to One

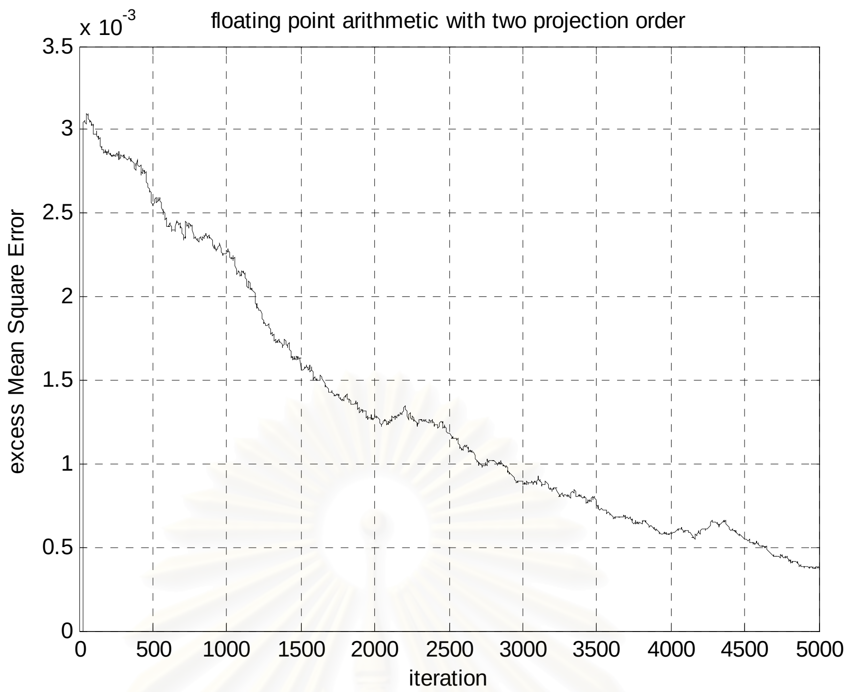


Figure 4.16 Floating Point Arithmetic with Projection Order Set to Two

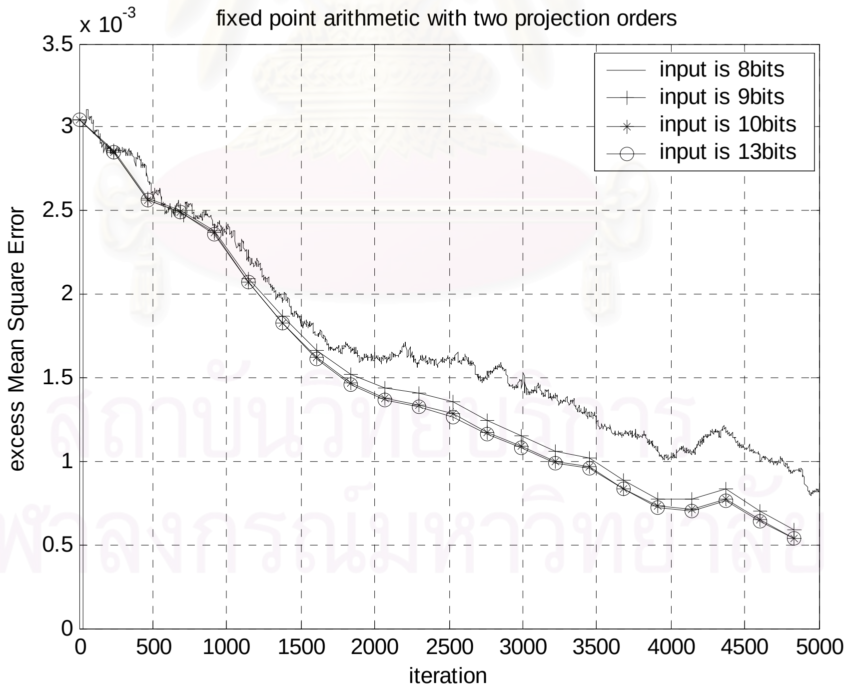


Figure 4.17 Fixed Point Arithmetic with Projection Order Set to Two

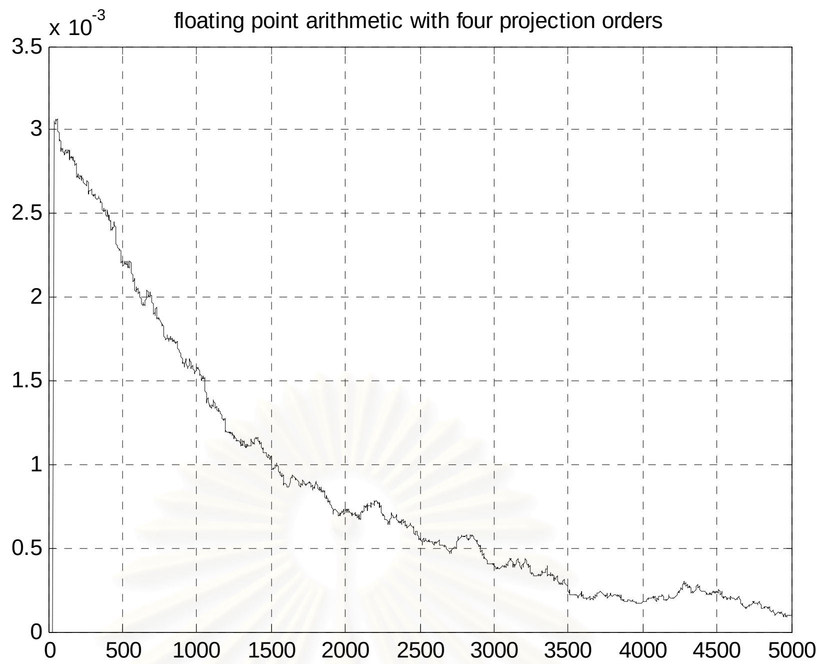


Figure 4.18 Floating Point Arithmetic with Projection Order Set to Four

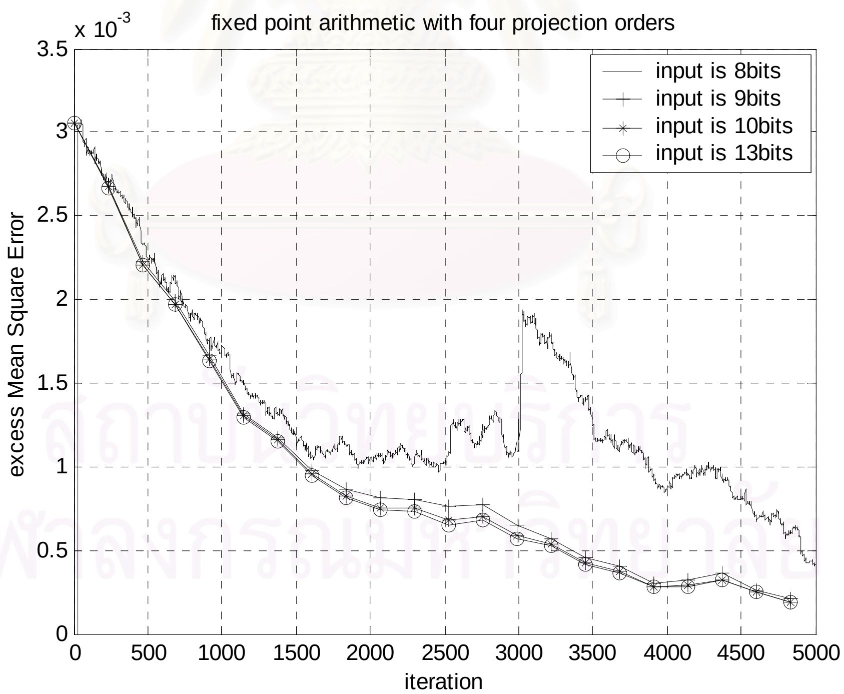


Figure 4.19 Fixed Point Arithmetic with Projection Order Set to Four

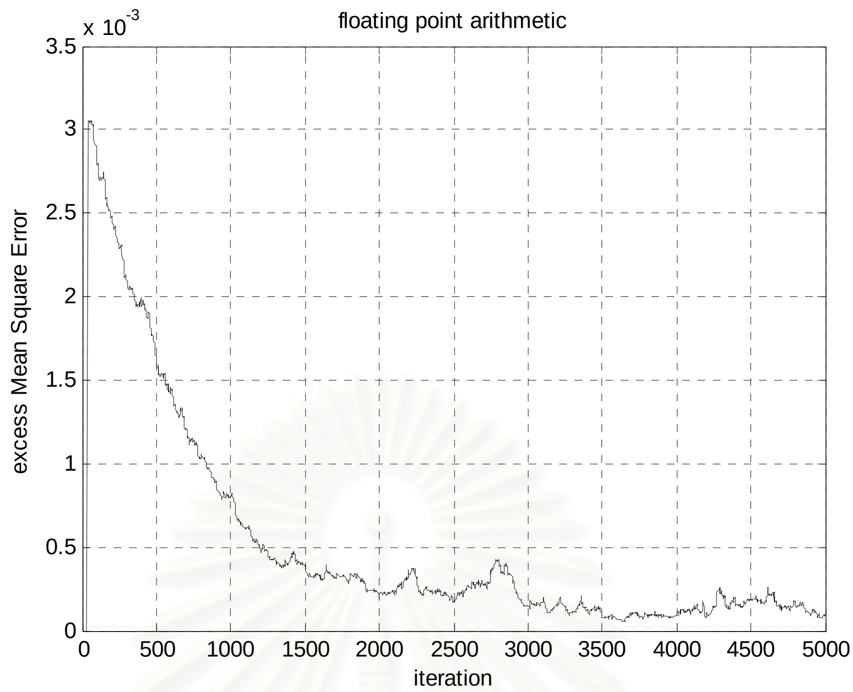


Figure 4.20 Floating Point Arithmetic with Projection Order Set to Eight

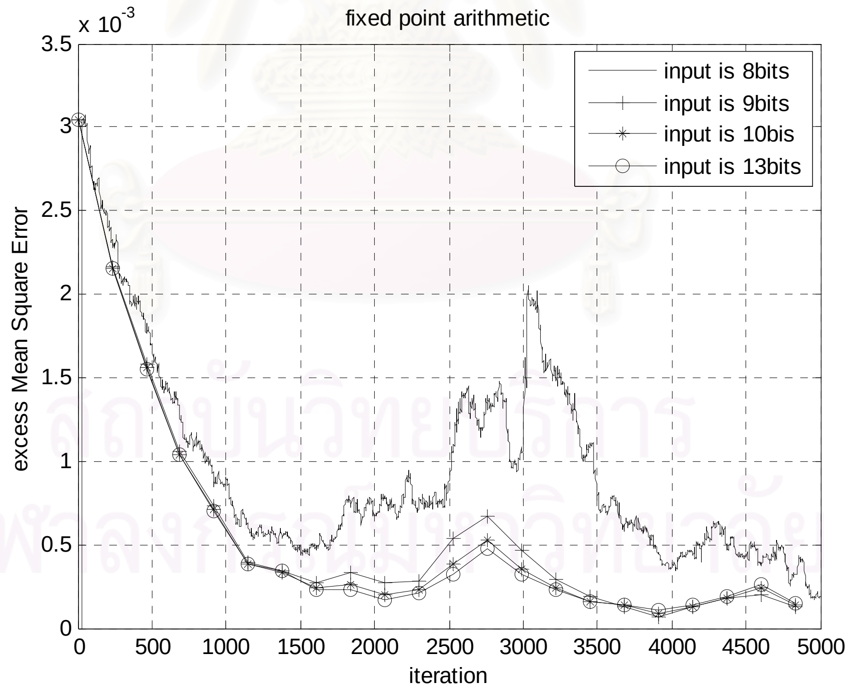


Figure 4.21 Fixed Point Arithmetic with Projection Order Set to Eight

CHAPTER V

MODIFICATION OF AFFINE PROJECTION ALGORITHM

This chapter modifies AP algorithm to cancel acoustic feedback in hearing aids, because matrix inversion is difficult to implement. So, we discuss how to avoid inversion by using inverting Teoplitz matrix. Section 5.1 introduces inverting Toeplitz matrix using Levinson Durbin Recursion. Section 5.2 develops autocorrelation of AP algorithm which becomes to Toeplitz matrix. Section 5.3 evaluates AP algorithm using Toeplitz matrix in fixed point arithmetic.

5.1 THE LEVINSON DURBIN RECURSION

Levinson [10] presented a recursive for solving a general set of linear symmetric Toeplitz matrix equations

$$R_x a = b \tag{5.1}$$

Durbin improved the Levinson recursion for the special case in which the right hand side of the Toeplitz equations is a unit vector. In this section we describe developing this algorithm, known as the *Levinson Durbin recursion* and inverting Toeplitz matrix.

5.1.1 DEVELOPMENT OF THE RECURSION

All poles modeling using prony's method or the autocorrelation method requires that we solve the normal equations which, for a N^{th} order model, are

$$r_x(k) + \sum_{l=1}^{N} a_N(l) r_x(k-l) = 0 \qquad ; \quad k = 1, 2, .., N \tag{5.2}$$

where the modeling error is

$$\varepsilon_N = r_x(0) + \sum_{l=1}^{N} a_N(l) r_x(l) \tag{5.3}$$

Combining equation(5.2) and (5.3) into matrix form we have

$$\begin{bmatrix} r_x(0) & r_x(1) & r_x(2) & . & . & . & r_x(N) \\ r_x(1) & r_x(0) & r_x(1) & . & . & . & r_x(N-1) \\ r_x(2) & r_x(1) & r_x(0) & . & . & . & r_x(N-2) \\ . & . & . & & & & . \\ . & . & . & & & & . \\ . & . & . & & & & . \\ r_x(N) & r_x(N-1) & r_x(N-2) & . & . & . & r_x(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_N(1) \\ a_N(2) \\ . \\ . \\ . \\ a_N(N) \end{bmatrix} = \varepsilon_N \begin{bmatrix} 1 \\ 0 \\ 0 \\ . \\ . \\ . \\ 0 \end{bmatrix} \qquad (5.4)$$

which is a set of $N+1$ linear equations in the $N+1$ unknowns $a_N(1), a_N(2), ..., a_N(N)$ and ε_N. Equivalently, equation (5.4) may are written as

$$R_N a_N = \varepsilon_N \mathrm{u}_1 \qquad (5.5)$$

where R_N is a $(N+1)\times(N+1)$ Hermitian Toeplitz matrix and $\mathrm{u}_1 = [1, 0, ..., 0]^T$ is a unit vector with 1 in the first position

The Levison Durbin recursion for solving equation (5.5) is an algorithm that is recursive in the model order. In the other words, the coefficients of the $(j+1)^{st}$ order all pole model, a_{j+1}, are found from the coefficients of the j pole model, a_j. We begin, therefore, by showing how the solution to the j^{th} order normal equations may be used to derive the solution to the $(j+1)^{st}$ order equations. Let $a_j(i)$ be the solution to the j^{th} order normal equations

$$\begin{bmatrix} r_x(0) & r_x(1) & r_x(2) & . & . & . & r_x(j) \\ r_x(1) & r_x(0) & r_x(1) & . & . & . & r_x(j-1) \\ r_x(2) & r_x(1) & r_x(0) & . & . & . & r_x(j-2) \\ . & . & . & & & & . \\ . & . & . & & & & . \\ . & . & . & & & & . \\ r_x(j) & r_x(j-1) & r_x(j-2) & . & . & . & r_x(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_j(1) \\ a_j(2) \\ . \\ . \\ . \\ a_j(j) \end{bmatrix} = \begin{bmatrix} \varepsilon_j \\ 0 \\ 0 \\ . \\ . \\ . \\ 0 \end{bmatrix} \qquad (5.6)$$

which, in matrix notation is

$$R_j a_j = \varepsilon_j \mathrm{u}_1 \qquad (5.7)$$

given a_j, we want to derive the solution to the $(j+1)^{st}$ order normal equations,

$$R_{j+1}a_{j+1} = \varepsilon_{j+1}\mathrm{u}_1 \qquad (5.8)$$

The procedure for doing this is a follows. Suppose that we append a zero to the vector a_j and multiply the resulting vector by R_{j+1}. The result is

$$\begin{bmatrix} r_x(0) & r_x(1) & r_x(2) & . & . & . & r_x(j) & r_x(j+1) \\ r_x(1) & r_x(0) & r_x(1) & . & . & . & r_x(j-1) & r_x(j) \\ r_x(2) & r_x(1) & r_x(0) & . & . & . & r_x(j-2) & r_x(j-1) \\ . & . & . & & & & . & \\ . & . & . & & & & . & \\ r_x(j) & r_x(j-1) & r_x(j-2) & . & . & . & r_x(0) & r_x(1) \\ r_x(j+1) & r_x(j) & r_x(j-1) & . & . & . & r_x(1) & r_x(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_j(1) \\ a_j(2) \\ . \\ . \\ a_j(j) \\ 0 \end{bmatrix} = \begin{bmatrix} \varepsilon_j \\ 0 \\ 0 \\ . \\ . \\ . \\ \gamma_j \end{bmatrix} \qquad (5.9)$$

where the parameter γ_j is

$$\gamma_j = r_x(j+1) + \sum_{i=1}^{j} a_j(i) r_x(j+1-i) \qquad (5.10)$$

The key step in the derivation of the Levinson Durbin recursion is to note that the Hermitian Toeplitz property of R_{j+1} allow us to rewrite equation (5.9) in the equivalent form

$$\begin{bmatrix} r_x(0) & r_x(1) & r_x(2) & . & . & . & r_x(j) & r_x(j+1) \\ r_x(1) & r_x(0) & r_x(1) & . & . & . & r_x(j-1) & r_x(j) \\ r_x(2) & r_x(1) & r_x(0) & . & . & . & r_x(j-2) & r_x(j-1) \\ . & . & . & & & & . & \\ . & . & . & & & & . & \\ r_x(j) & r_x(j-1) & r_x(j-2) & . & . & . & r_x(0) & r_x(1) \\ r_x(j+1) & r_x(j) & r_x(j-1) & . & . & . & r_x(1) & r_x(0) \end{bmatrix} \begin{bmatrix} 0 \\ a_j(j) \\ a_j(j-1) \\ . \\ . \\ a_j(1) \\ 1 \end{bmatrix} = \begin{bmatrix} \gamma_j \\ 0 \\ 0 \\ . \\ . \\ . \\ \varepsilon_j \end{bmatrix} \qquad (5.11)$$

Combining the resulting equation with equation (5.9)

$$R_{j+1}\left\{\begin{bmatrix}1\\a_j(1)\\a_j(2)\\.\\.\\a_j(j)\\0\end{bmatrix}+\Gamma_{j+1}\begin{bmatrix}0\\a_j(j)\\a_j(j-1)\\.\\.\\a_j(1)\\1\end{bmatrix}\right\}=\begin{bmatrix}\varepsilon_j\\0\\0\\.\\.\\0\\\gamma_j\end{bmatrix}+\Gamma_{j+1}\begin{bmatrix}\gamma_j\\0\\0\\.\\.\\0\\\varepsilon_j\end{bmatrix} \qquad (5.12)$$

Since we want to find the vector a_{j+1} which, when multiplied by R_{j+1}, yield a scaled unit vector, note that if we set

$$\Gamma_{j+1}=-\frac{\gamma_j}{\varepsilon_j} \qquad (5.13)$$

The equation (5.12) becomes

$$R_{j+1}a_{j+1}=\varepsilon_{j+1}\mathrm{u}_1$$

where

$$a_{j+1}=\begin{bmatrix}1\\a_j(1)\\a_j(2)\\.\\.\\a_j(j)\\0\end{bmatrix}+\Gamma_{j+1}\begin{bmatrix}0\\a_j(j)\\a_j(j-1)\\.\\.\\a_j(1)\\1\end{bmatrix} \qquad (5.14)$$

which is the solution to the $(j+1)^{st}$ order normal equations. Furthermore,

$$\varepsilon_{j+1}=\varepsilon_j+\Gamma_{j+1}\gamma_j=\varepsilon_j\left[1-\left|\Gamma_{j+1}\right|^2\right] \qquad (5.15)$$

equation (5.14), referred to as the *Levinson order update equation*, may be expressed as

$$a_{j+1}(i)=a_j(i)+\Gamma_{j+1}a_j(j-i+1) \quad ; \quad i=0,1,\ldots,j+1 \qquad (5.16)$$

All that is required to complete the recursion is to define the condition necessary to initialize the recursion. These conditions are given by the solution for the model of order $j = 0$

$$
\begin{aligned}
a_0(0) &= 1 \\
\varepsilon_0 &= r_x(0)
\end{aligned}
\tag{5.17}
$$

5.1.2 THE CHOLESKY DECOMPOSITION

We have seen the Levinson Durbin recursion is an efficient algorithm for solving the autocorrelation normal equation. It may also be used to perform Cholesky decomposition of the Hermitian Toeplitz autocorrelation matrix R_N. The Cholesky (LDU) decomposition of a Hermitian matrix C is a factorization of the form

$$C = LDL^H \tag{5.18}$$

where L is a *lower triangular* matrix with ones along the diagonal and D is a *diagonal* matrix . With the Cholesky decomposition of the autocorrelation matrix we will easily be able to establish the equivalence between the positive definiteness of R_N, the positivity of the error sequence ε_j, and the unit magnitude constraint on the reflection coefficient Γ_j. In addition, we will be able to derive a closed form expression for the inverse of the autocorrelation matrix as well as a recursive algorithm for inverting a Toeplitz matrix.

To derive the Cholesky decomposition of R_N, consider the $(N+1)\times(N+1)$ *upper triangular* matrix

$$
A_N = \begin{bmatrix}
1 & a_1(1) & a_2(2) & . & . & . & a_N(N) \\
0 & 1 & a_2(1) & . & . & . & a_N(N-1) \\
0 & 0 & 1 & . & . & . & a_N(N-2) \\
. & . & . & & & & . \\
. & . & . & & & & . \\
. & . & . & & & & . \\
0 & 0 & 0 & . & . & . & 1
\end{bmatrix}
\tag{5.19}
$$

This matrix is formed from the vectors $a_0, a_1, ..., a_N$ that are produced when the Levinson Durbin recursion is applied to the autocorrelation sequence $r_x(0),...,r_x(N)$. Note that the j^{th} column of A_N contains the filter coefficients, a^R_{j-1}, padded with zeros. Since

$$R_j a^R_j = \varepsilon_j u_j \tag{5.20}$$

where $u_j = [0,0,...,1]^T$ is a unit vector of length $j+1$ with a one in the final position then

$$R_N A_N = \begin{bmatrix} \varepsilon_0 & 0 & 0 & . & . & . & 0 \\ * & \varepsilon_1 & 0 & . & . & . & 0 \\ * & * & \varepsilon_2 & . & . & . & 0 \\ . & . & . & & & & . \\ . & . & . & & & & . \\ . & . & . & & & & . \\ * & * & * & . & . & . & \varepsilon_N \end{bmatrix} \tag{5.21}$$

which is a lower triangular matrix with the prediction error ε_j along the diagonal (an asterisk is used to indicate those elements that are, in general, nonzero).

Since the product of two lower triangular matrices is lower triangular matrix, if we multiply $R_N A_N$ on the left by the lower triangular matrix A^H_N, then we obtain another lower triangular matrix, $A^H_N R_N A_N$ .note that since the terms along the diagonal of A_N are equal to one, the diagonal of $A^H_N R_N A_N$ will be the same as that $R_N A_N$, and $A^H_N R_N A_N$ will also have the form

$$A^H_N R_N A_N = \begin{bmatrix} \varepsilon_0 & 0 & 0 & . & . & . & 0 \\ * & \varepsilon_1 & 0 & . & . & . & 0 \\ * & * & \varepsilon_2 & . & . & . & 0 \\ . & . & . & & & & . \\ . & . & . & & & & . \\ . & . & . & & & & . \\ * & * & * & . & . & . & \varepsilon_N \end{bmatrix} \tag{5.22}$$

Even more importantly, however, is the observation that since $A^H_N R_N A_N$ is Hermitian, then the matrix on the right side of the equation (5.22) must also be Hermitian. Therefore, the term below the diagonal are zero and

$$A_N^H R_N A_N = D_N \tag{5.23}$$

where D_N is a diagonal matrix

$$D_N = \text{diag}\{\varepsilon_0, \varepsilon_1, ..., \varepsilon_N\}$$

5.1.3 INVERTING A TOEPLITZ MATRIX

In this section, deriving a recursive algorithm for inverting a Toeplitz matrix R_N and show how this recursion may be used to derive the Levinson recursion for solving a general set of Toeplitz equations $R_N x = b$. Showing how the decomposition derived in section5.1.2 may be used to express the inverse of a Toepliz matrix in terms of the vectors a_j and the error ε_j.the Levinson Durbin recursion will then applied to this expression for R_N^{-1} to derive the *Toeplitz matrix inversion recursion.*

Let R_N be a nonsingular Hermitian Toeplitz matrix. Using the decomposition given in equation (5.23), taking the inverse of both sides we have

$$\left(A_N^H R_N A_N\right)^{-1} = A_N^{-1} R_N^{-1} A_N^{-H} = D_N^{-1} \tag{5.24}$$

multiplying both sides of this equation by A_N on the left and by A_N^H on the right give the desired expression for R_N^{-1}

$$R_N^{-1} = A_N D_N^{-1} A_N^H \tag{5.25}$$

D_N is a diagonal matrix, D_N^{-1} is easily computed. Therefore, finding the inverse of R_N simply involves applying the Levinson Durbin recursion to the sequence $r_x(0),...,r_x(N)$ forming the matrix A_N, and performing the matrix product in equation (5.25).

5.2 AFFINE PROJECTION ALGORITHM USING INVERTING TOEPLITZ MATRIX

In equation (3.12), the main difficulty of implementing an AP algorithm is how to invert the correlation matrix R_N effectively [11] and [7] suggested to use the sliding window fast RLS type algorithm and suggested use of the matrix inversion lemma twice. As [12] pointed out, the FTF (Fast Transversal Filter) type of algorithm is too complicate to implement and numerically unstable, and hence, it is difficult to use in the real environment. The use of the matrix inversion lemma is also not guaranteed to be stable. It is worthwhile to mention that [13] used approximation in transform domain. This method also has an inherit problem as [13] proposed to use either periodic restart or leaky integration to reduce the error accumulation in the fixed point implementation.

In [13] to develop a new algorithm, consider the matrix R_N as

$$R_N = \begin{bmatrix} \hat{r}_0(n) & \hat{r}_1(n) & . & . & . & . & \hat{r}_{N-1}(n) \\ \hat{r}_1(n-1) & \hat{r}_0(n-1) & . & . & . & . & \hat{r}_{N-2}(n-1) \\ \hat{r}_2(n-2) & \hat{r}_1(n-2) & . & . & . & . & \hat{r}_{N-3}(n-2) \\ . & . & & & & & . \\ . & . & & & & & . \\ . & . & & & & & . \\ \hat{r}_{N-1}(n-N+1) & \hat{r}_{N-2}(n-N+1) & . & . & . & . & \hat{r}_0(n-N+1) \end{bmatrix} \quad (5.26)$$

where

$\hat{r}_\tau(n) = \sum_{i=0}^{L-1} x(n-i)x(n-i-\tau)$ is the estimate of the autocorrelation at lag τ and at time instant n based on the past L input data.

Therefore, if $L \gg N$ (and it is true for most applications), the following holds:

$$\hat{r}_\tau(n) = \hat{r}_\tau(n-1) = \ldots = \hat{r}_\tau(n-N+1) \quad (5.27)$$

thus the equation (5.26) can be approximated as

$$\tilde{R}_N = \begin{bmatrix} \hat{r}_0(n) & \hat{r}_1(n) & . & . & . & . & \hat{r}_{N-1}(n) \\ \hat{r}_1(n) & \hat{r}_0(n) & . & . & . & . & \hat{r}_{N-2}(n) \\ \hat{r}_2(n) & \hat{r}_1(n) & . & . & . & . & \hat{r}_{N-3}(n) \\ . & . & & & & & . \\ . & . & & & & & . \\ . & . & & & & & . \\ \hat{r}_{N-1}(n) & \hat{r}_{N-2}(n) & . & . & . & . & \hat{r}_0(n) \end{bmatrix} \quad (5.28)$$

And the autocorrelation matrix becomes a Toeplitz matrix. The inversion of a Toeplitz matrix is calculated by equation (5.25). Computational complexity of the inversion of a Toeplitz matrix is less than the computational complexity of normal inversion matrix as shown in TABLE 5.1, which is difficult to compare it. That's why, TABLE 5.2 give various matrix.

5.3 SIMULATION RESULT

Comparison between normal inversion matrix and the inversion of Toeplitz matrix, it is shown in figure 5.1, it compares various projection orders (N =1, N = 2, N = 4, N = 8), plot excess Mean Square Error with step size $(\mu) = 0.005$, iteration is 5000, each curve the average of 100 trials.

AP algorithm using inverting Toeplitz matrix converges similar to AP algorithm. So we will use inversion of Toeplitz matrix with AP algorithm that is not difficult to inverse if the projection order increase.

For the resolution of data the same AP algorithm is applied and the input is 10 bits, it is enough for convergence like floating point arithmetic.

5.3.1 PROJECTION ORDER SET TO TWO

Affine Projection algorithm uses Toeplitz matrix for inversion. It compares between floating point arithmetic as shown in figure5.2 and fixed point arithmetic (input is 8 bits, 9bits, 10bits, 13bits) as shown in figure5.3. While the input gives 10bits up to 13bits, the excess Mean Square Error converge same as floating point arithmetic.

5.3.2 PROJECTION ORDER SET TO FOUR

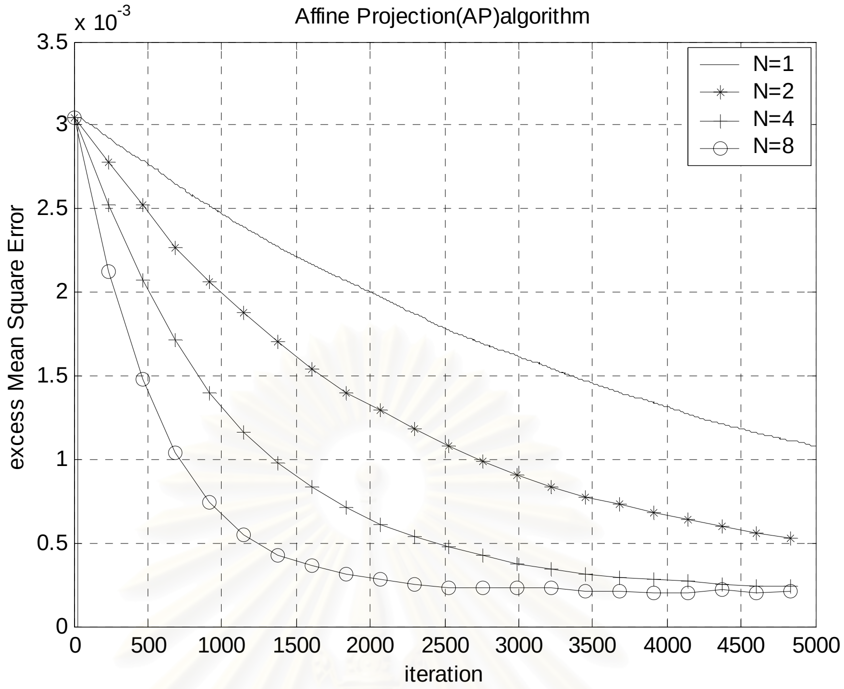
Affine Projection algorithm uses Toeplitz matrix for inversion. It compare between floating point arithmetic as shown in figure 5.4 and fixed point arithmetic (input is 8 bits, 9 bits, 13 bits) as shown in figure 5.5. While the input gives 10 bits up to 13 bits, the excess Mean Square Error converge the same as floating point arithmetic.

TABLE 5.1 Computational complexity of inversion matrix

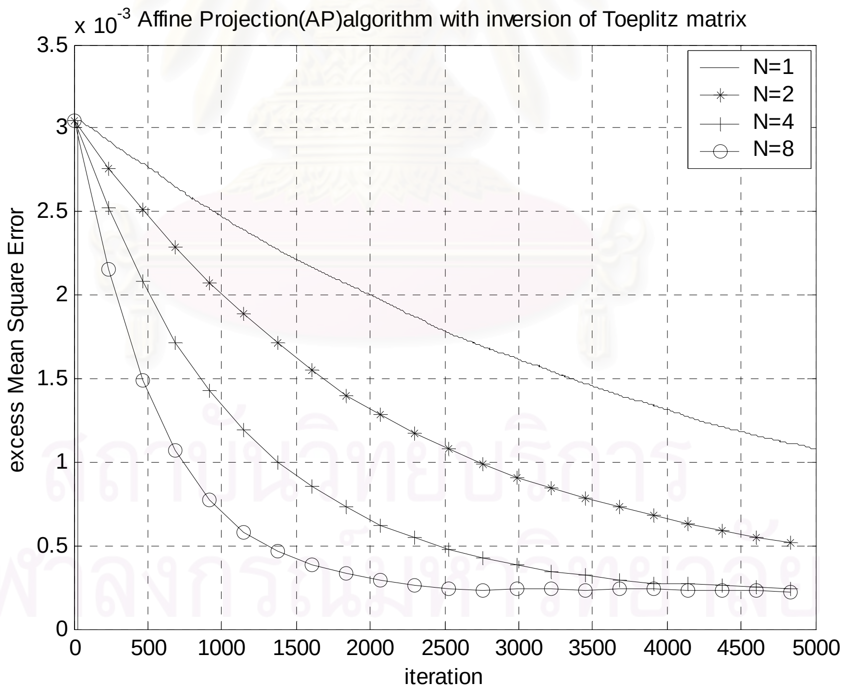
	Number of multiplication	Number of addition/subtraction	Number of division
Normal inversion matrix	$2^{N-1}\prod_{i=0}^{N-2}(N-i)+2^{N-3}N^{2}\prod_{i=0}^{N-3}(N-i-1)$, $\mathrm{N}\geq 3$	$N^{2}f(N-1)+f(N)$ $f(N)=(N-1)+Nf(N-1)$ $f(2)=1$ and $\mathrm{N}\geq 3$	N
Inversion of Toeplitz matrix	$3N+\sum_{j=0}^{N}(N-j)+2\sum_{j=1}^{N}(N-j)+2\left(\sum_{j=2}^{N}\sum_{i=j}^{N}(N-i)\right)+\sum_{j=1}^{N}j$	$2N+\sum_{j=0}^{N}(N-j)+2\left(\sum_{j=2}^{N}\sum_{i=j}^{N}(N-i)\right)+\sum_{j=1}^{N}j$	$2N$

TABLE 5.2 Computational complexity of various matrixes

	Number of multiplication						Number of addition/subtraction					
	$N=3$	$N=4$	$N=5$	$N=6$	$N=7$	$N=8$	$N=3$	$N=4$	$N=5$	$N=6$	$N=7$	$N=8$
Normal inversion matrix	30	288	3360	46080	725760	12902400	14	103	694	5003	40270	362815
Inversion of Toeplitz matrix	*29*	*52*	*85*	*130*	*189*	*264*	*20*	*36*	*60*	*94*	*140*	*200*



(a) Convergence of (AP) algorithm



(b) Convergence of (AP) algorithm using Inverting Toeplitz matrix

Figure 5.1 Comparison between AP algorithm and AP algorithm using Inverting Toeplitz Matrix

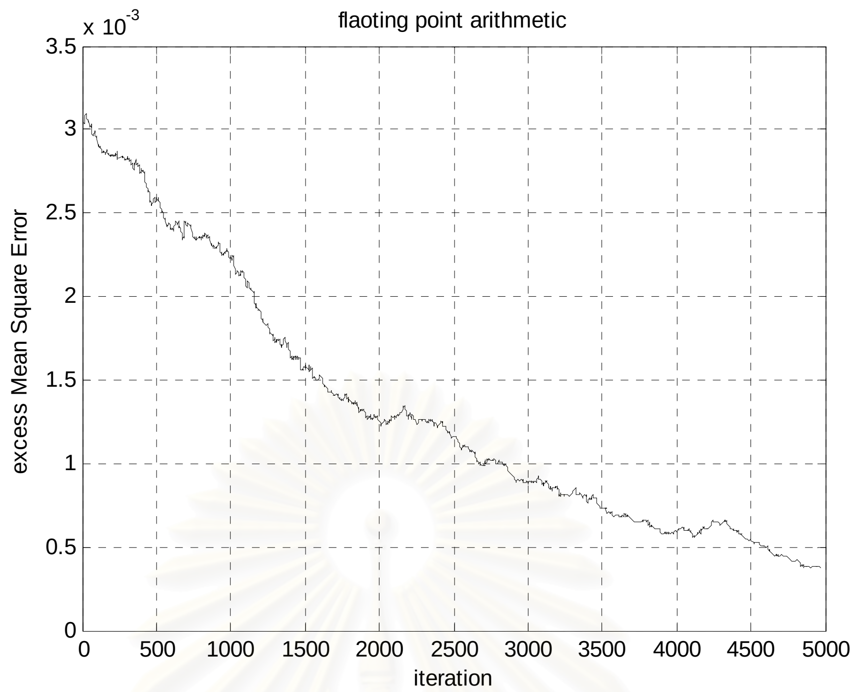


Figure 5.2 Floating Point Arithmetic using Inverting Toeplitz Matrix
(Projection Order equal to two)

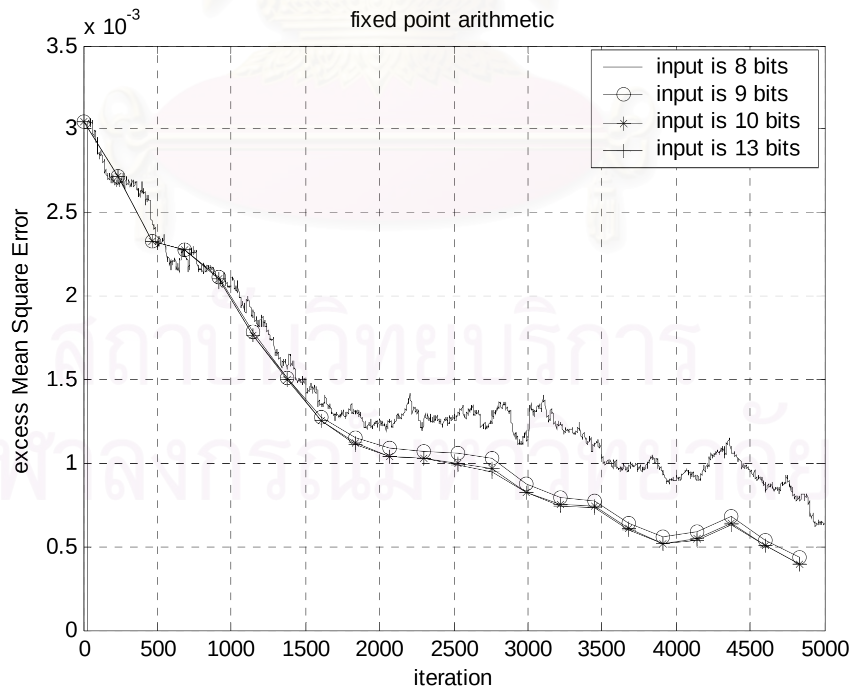


Figure 5.3 Fixed Point Arithmetic using Inverting Toeplitz Matrix
(Projection Order equal to two)

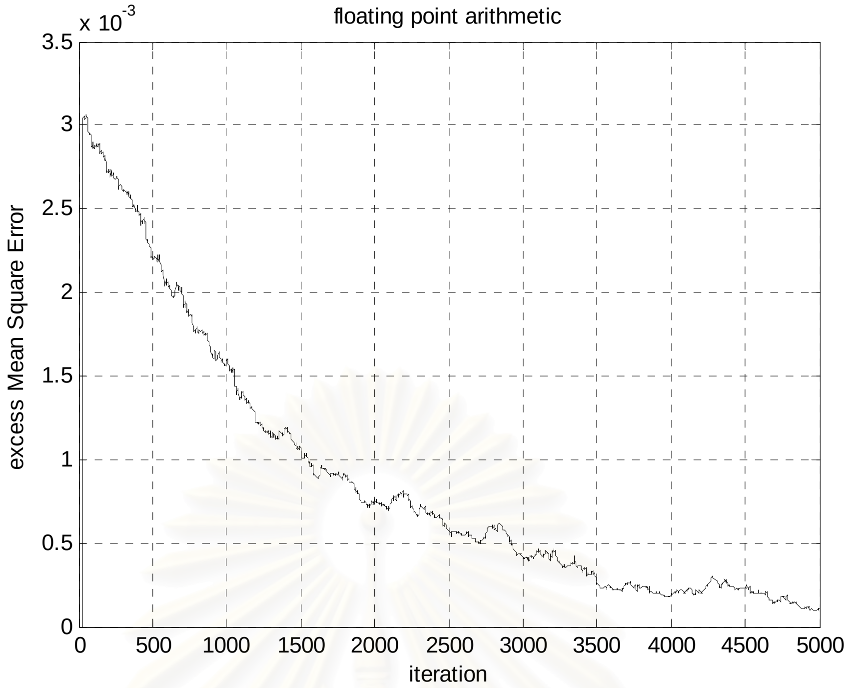


Figure 5.4 Floating Point Arithmetic using Inverting Toeplitz Matrix
(Projection Order equal to four)

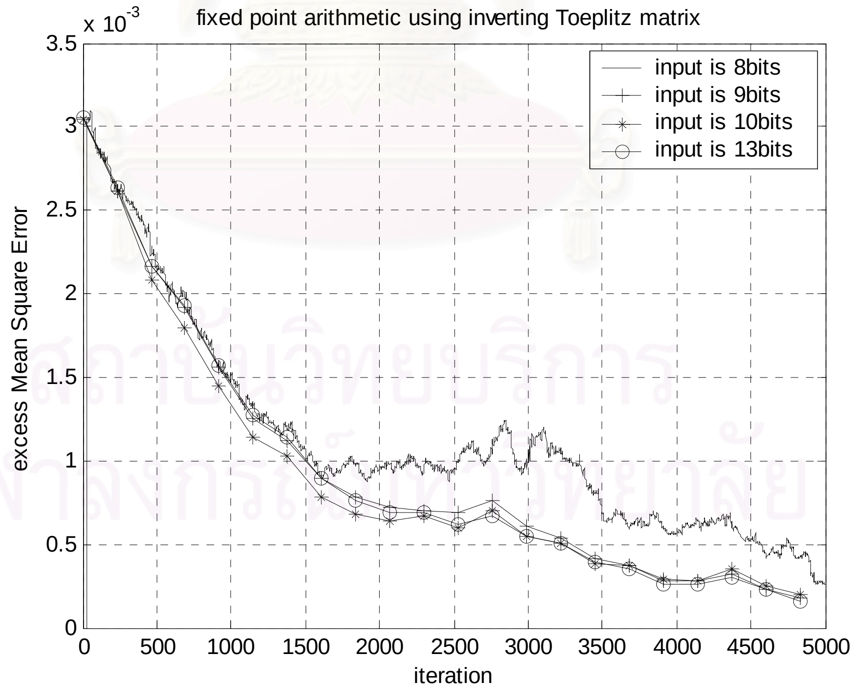


Figure 5.5 Fixed Point Arithmetic using Inverting Toeplitz Matrix
(Projection Order equal to four)

CHAPTER VI

CONCLUSION AND RECOMMENDATION

This thesis uses Affine Projection (AP) algorithm to adapt the coefficient of the adaptive filter. It gets both low computational complexity and fast convergence rate, thus making useful to implement in hardware since it can reduce complexity of hardware and save the area of hearing aids. Therefore, it makes a small hearing aid. Before implementing hardware, we have to evaluate this algorithm in fixed point arithmetic.

This thesis evaluated AP algorithm by many projection orders (projection order equal to one, two, and four). Each projection order by fixed point arithmetic are 8 bits' input, 9 bits' input, 10 bits' input and 13 bits' input. And at least number bits is 10bits' input, fixed point arithmetic converge the same as floating point arithmetic. One problem is inversion of autocorrelation matrix that is difficult to implement, if the projection order increase. Therefore, this thesis also evaluated AP algorithm using inverting Toeplitz matrix.

However, we have to consider number of projection orders, if projection orders increases, the computational complexity of AP algorithm may equal to RLS algorithm. In this thesis, number of projection orders is smaller than nine. It means that computational complexity is not so much. It is the fast convergence rate, which is goal of this thesis.

In this thesis, AP algorithm is simulated in fixed point arithmetic and 10 bits converges like floating point arithmetic. In the next work, AP algorithm should be implemented on FPGA chip and its hardware complexity should be investigated in the future research.

REFERENCES

[1] Zheng, Y.R., Goubran, R.A. *Adaptive Beamforming using AffineProjection Algorithm*. Signal Processing Proceedings. WCCC-ICSP. 5th International Conference on, 3 (2000):1929 – 1932.

[2] S. Thippayathethana. *Evaluation of Step Size Adjustments of the Least Mean Square Algorithm for Reducing Acoustic Feedback in Hearing Aids*. Master's thesis, Department of Electrical Engineering, Faculty of Engineering, Chulalongkorn University, 1999.

[3] Bustamante, D.K.; Worrall, T.L.; Williamson, M.J. *Measurement and Adaptive Suppression of Acoustic Feedback in Hearing Aids.* IEEE Trans. On Acoust, Speech, and Signal Processing 3, 3(1989): 2017- 2020.

[4] Kate, J. M. senior member. *Feedback Cancellation in Hearing Aids: Result from a Computer Simulation***.** IEEE Tran. On Signal Processing 39, 3 (March 1991):553-562.

[5] Kates, J.M. *Superdirective Arrays for Hearing Aids*. IEEE Applications of Signal Processing to Audio and Acoustics (17-20 Oct. 1993):73 – 76.

[6] Maxwell, J. A and Zurek, P. M. *Reducing Acoustic Feedback in Hearing Aids*. IEEE Trans .on Speech and Audio Processing, 3, 4(July 1995): 304 – 313.

[7] Steven L. Gay and Sanjeev Tvathia. *The Fast Affine Projection algorithm*. Proc. IEEE International Conf. Acoust., Speech, and Signal Processing, Detroit, MI, 5 (May 1995):3023 – 3026.

[8] Haykin. S. *Adaptive filter theory*. Prentice Hall, 1996.

[9] M. Tanaka, Y.Kaneda, S.makio, and J.Kojima. *Fast Projection Algorithm and its Step Size Control*. ICASSP-95, 1995.

[10] Monson H.Hayes. *Statistical Digital Signal Processing and Modeling.* (n. p): John Wiley & Sons,1996.

[11] M.J Reed and M.O.J Hawksford. *Acoustic Echo Cancellation with the Fast Affine Projection.* Audio and Music Technology: The Challenge of Creative DSP (Ref. No. 1998/470), IEE Colloquiumon, (18 Nov. 1998): 16/1 - 16/8.

[12] Oh, S.; Linebarger, D.; Priest, B.; Raghothaman, B. A *Fast Affine Projection Algorithm for an Acoustic Echo Canceller using a Fixed-point DSP Proces56sor.* ICASSP-97, 5 (21-24 April 1997): 4121 – 4124.

[13] S.C. Douglas. *Efficient Approximate Implementations of the Fast Affine Projection Algorithm using Orthogonal Transforms*. IEEE International Conf. on Acoustic, Speech, and Signal Processing, 3 (1996): 1656 – 1659.

Biography

Vimontha Khieovongphachanh was born in 1980 in Vientiane Municipality, Lao P.D.R. In 2002, she received her Eng, In Electronic Engineering from National University of Laos. From 2002 to 2003, she worked at Department of Electronic Engineering, Faculty of Engineering and Architecture as an instructor. Her research interest in adaptive filter