國 立 交 通 大 學

電信工程學系

碩 士 論 文

根據最佳收斂步伐與通道削減來提升非線性迴音消除的收斂性

Study on Fast Converging Nonlinear Echo Cancellation Based on Optimum Step Size and Channel Shortening Approaches

研究生:施嘉勝

指導教授：謝世福 博士

中 華 民 國 九十七 年 六 月

根據最佳收斂步伐與通道削減來提升非線性迴音消除的收斂性

學生：施嘉勝　　指導教授：謝世福

國立交通大學電信工程學系碩士班

中文摘要

爲了消除免持聽筒或者視訊會議上非線性的迴音，傳統上可以用 Volterra 濾波器或 Hammerstein 濾波器來追蹤非線性迴音的通道。然而這兩個濾波器最大的缺點就是收斂速度慢並需要付出高的計算量。

在此篇論文中，我們提出最佳的可調整式收斂步伐演算法並且應用在 Volterra 濾波器。其目的在於加快收斂速度，此收斂步伐是由估計濾波器與真實的最小閥係數誤差在均方誤差(MSE)。每一個閥，都隨著係數誤差改變而調整的收斂步伐，而由於此演算法需要知道真實的迴音通道，所以我們進一步提出模擬通道的實際的應用。

除了收斂步伐的控制，通道削減結構(channel shortening)也被用來解決 Hammerstein 濾波器收斂速度慢與高複雜度的問題。我們做了 Least-square 和適應性演算法角的理論分析，並且提出多級更新係數的方法來更加快收斂速度。最後用電腦模擬來支持驗證之前的分析討論。

Study on Fast Converging Nonlinear Echo Cancellation Based on Optimum Step Size and Channel Shortening Approaches

Student: C. S. Shih **Advisor: S. F. Hsieh**

Department of Communication Engineering

National Chiao Tung University

Abstract

In order to cancel nonlinear acoustics echo in hands-free telephones or teleconferencing system. In general, adaptive Volterra filter and Hammerstein model are known to track nonlinear echo path. However, their major drawbacks are slow convergence rate and high computation complexity.

In this thesis, we propose an optimum time–and tap– variant step-size for Volterra filter in order to speed up convergence rate. The step-size is based on the MMSE criterion of coefficients errors. As the optimum step-size needs to know the real echo path coefficient, we propose the exponential model for practical implementations.

In addition to adaptive step-size control , the channel shortening structure was proposed to overcome slow convergence rate and high computation complexity in Hammerstein structure, we perform the least-square and adaptive algorithm theoretical analysis in channel shortening structure in case of a linear loudspeaker.

From which a multiple stage update scheme is proposed in this structure to speed up convergence rate. Computer simulations justify our analysis and show the improved performance of the proposed nonlinear acoustic echo canceller.

Acknowledgement

I would like to express my deepest gratitude to my advisor Dr. S.F. Hsieh, for his enthusiastic guidance and great patience, especially the autonomy in research. Throughout the composition of this thesis, Dr Hsieh provides me with many enlightening viewpoints and insightful suggestions. My special thanks go to C.W Huang, S.H Weng, J.H Dai, and H.C Chen for their inspiration and encouragement. I also appreciate my friends for their inspiration and help. Finally, I would like to show my thanks to my parents, sisters, and girlfriend for their unceasing encouragement and love.

Study on Fast Converging Nonlinear Echo Cancellation Based on Optimum Step Size and Channel Shortening Approaches

Contents

List of Figures

List of tables

Chapter 1
Introduction

In these years, hands-free system telephone and teleconference systems are widely used. However, the systems usually suffer from the annoying acoustic echo problem; the phenomenon occurs that the far end speech is transmitted back to the microphone at the near end. A hands-free telephone system is shown in Fig 1.1. The main problem of acoustic echo cancellation (AEC) is to copy the unknown echo path and subtract the copied echo components from the microphone output. Since the echo path may be time-variant due to objects moving around the room, an adaptive filter is commonly used for tracking the echo path. If AEC estimates the echo path accurately, the echo would be cancelled and the communication quality would be enhanced.

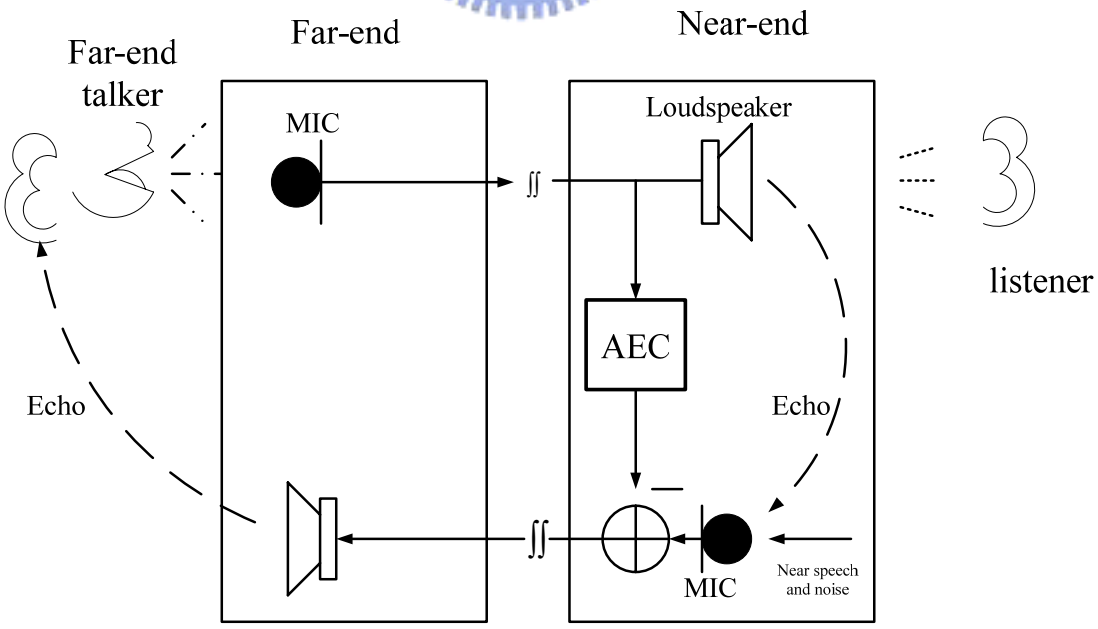


Fig 1.1 Hands-free telephone system

There are many adaptive algorithms that have been proposed [13]. The least-mean-square (LMS) algorithm is famous for its low computational cost, the affine projection algorithm (APA) and recursive least-squares (RLS) algorithm has its advantage of fast convergence rate, but they are higher computational complexity than LMS algorithm.

However, competitive audio consumer products require not only cheap signal processing hardware but also low-cost analog equipment and sound transducer, the echo path has nonlinear components caused by the capability of power amplifier (PA) [1], which can be overdriven and leads to nonlinear distortion in far-end speech. Therefore, the linear AEC is not sufficient to estimate the acoustic echo path. To overcome this problem, there are many methods that had been proposed. A general nonlinear AEC system is shown in Fig1.2. The far-end speech signal is passing through the nonlinear loudspeaker and the room impulse response and then picked up by microphone.

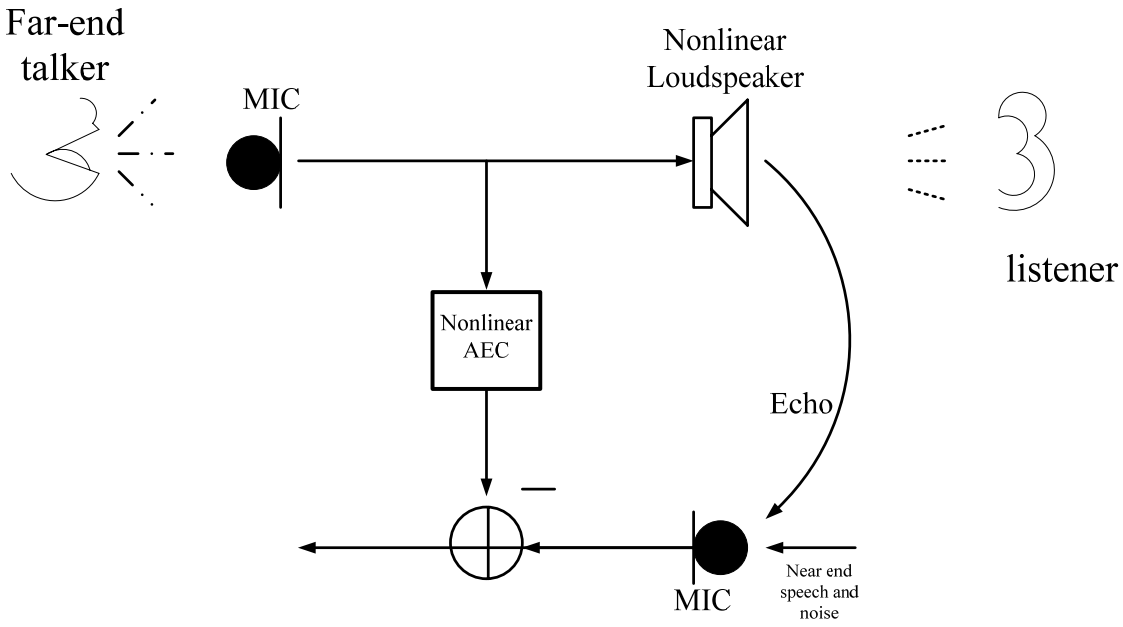


Fig1.2 Nonlinear acoustic echo cancellation system

In order to overcome the nonlinear acoustic echo caused by power amplifier, the popular method is via polynomial functions, i.e. Hammerstein model [5], Volterra [2]-[3], Wiener [17], Wiener and Hammerstein model [18]. Those four of nonlinear models will be introduced in Chapter 2.

Another approach is using echo suppression method, which is to increase the attenuation of the nonlinearly distorted residual echo and the convergent speed, but this approach will cause the near-end speech distortion and not eliminate acoustic echo completely.

In this thesis, in order to avoid near-end speech and eliminate acoustic completely, we focus on the nonlinear acoustic cancellation and employ the Volterra and Hammerstein model to track the nonlinear echo path. To overcome their slow convergence rate and high complexity, we propose an optimum time and tap – variant step-size for Volterra filter to speed up convergence rate in Chapter 3.

The channel shortening structure has been proposed in [14] to overcome high computational complexity disadvantage of Hammerstein structure. In chapter 4, we perform theoretical analysis in the senses of LMS and LS in case of a linear loudspeaker. In addition to theoretical analysis, we will propose multiple stage update scheme to speed up convergence rate.

We will provide computer simulations to justify our analysis and show the improved performance of the proposed nonlinear acoustic echo canceller in Chapter 5. Finally, we will give a conclusion of our work.

Chapter 2

Adaptive Nonlinear Acoustic echo cancellation

The loudspeakers for hands-free telephone or teleconferencing are usually small and cheap, so the loudspeaker will be saturated at high level speech. When the saturation effect happens, the loudspeaker is not linear any more [1]. The residual error using only linear acoustic echo cancellation is very large. We will discuss the nonlinear acoustic echo cancellation to overcome this question.

To some loudspeakers, the nonlinear effects have memory. If using memoryless structures to model that, the cancellations don't eliminate nonlinear echo perfect. The memory structures for canceling the memory echo are complex in general, i.e. Volterra model. As shown below, we will introduce and compare the several memoryless and memory structures in section 2.1.

However, the major drawback of nonlinear models lies in slow convergence rate and high computation complexity. In section 2.2, the algorithm to improve convergence rate will also be introduced.

2.1 Nonlinear AEC structure

2.1.1 Memoryless nonlinear AEC

- **Hammerstein structure**

In the section, we focus on the case that the nonlinearity in the echo can be considered to be memoryless. The Hammerstein structure [5], a famous memoryless nonlinearity model is a cascade of a memoryless polynomial filter and a FIR filter. As shown as Fig 2.1.

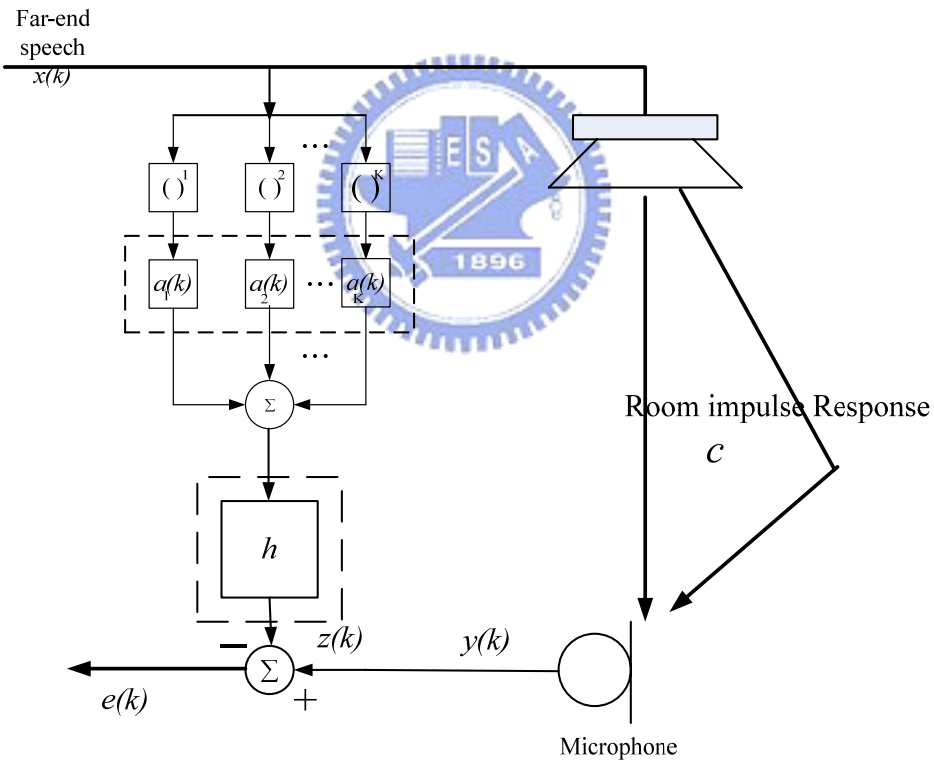


Fig 2.1 Hammerstein structure

In Fig 2.1, the output of *K*-order Hammerstein model $z(k)$ can be expressed by

$$z(k) = \sum_{l=0}^{M-1} h_l[u(k-l)] \tag{2.1}$$

where the $u(k)$ is polynomial filter output and M is FIR memory length

$$u(k) = \sum_{i=1}^{K} a_i [x(k)]^i \tag{2.2}$$

2.1.2 Memory nonlinear AEC

[A] Volterra structure

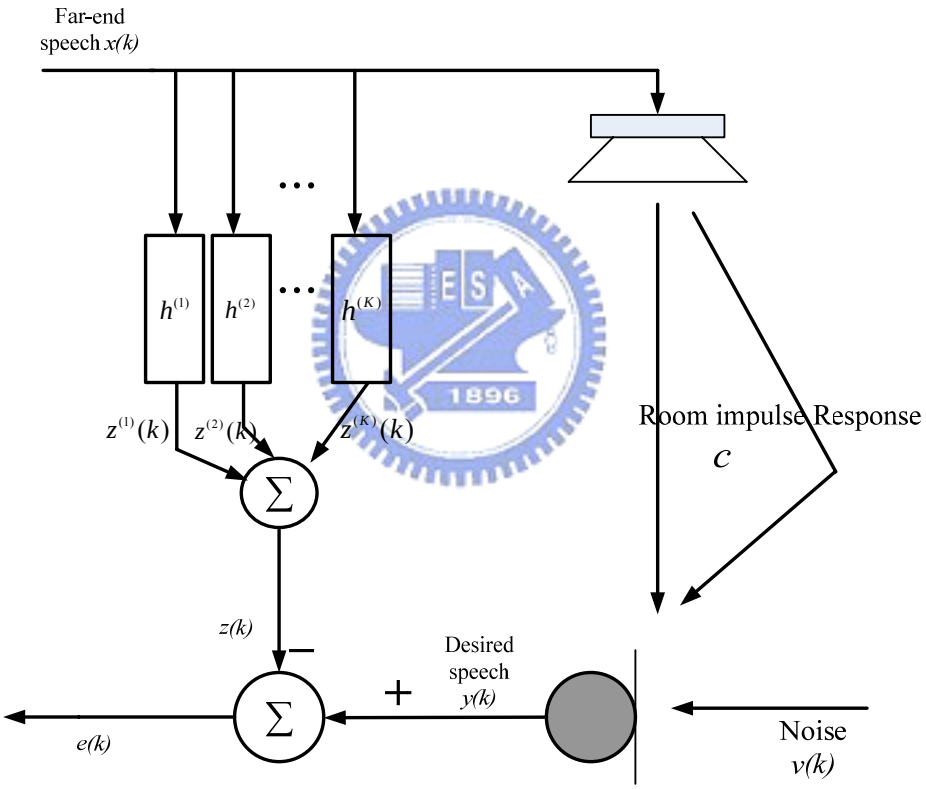


Fig 2.2 Volterra structure

As shown in Fig 2.2, another common approach to model the nonlinear behavior of loudspeakers is given by Volterra [2]-[3]. In the following we assume the unknown echo path, i.e. the cascade of nonlinear loudspeaker and room impulse response, can be modeled by a K^{th}-order Volterra filter, thus the output of Volterra filter $z(k)$ can

be expressed by

$$z(k)=\sum_{i=1}^{K} z^{(i)}(k) \tag{2.3}$$

where the I/O relation of the p'th order Volterra kernel with finite memory length N_p yields

$$z^{(p)}(k)=\sum_{l_{p,1}=0}^{N_p-1}\sum_{l_{p,2}=l_{p,1}}^{N_p-1}\cdots\cdots\sum_{l_{p,p}=l_{p,p-1}}^{N_p-1} h_{l_{p,1},l_{p,2},\ldots..l_{p,p}}\prod_{i=1}^{p} x(k-l_{p,i}) \tag{2.4}$$

[B] Wiener Model

In addition to Volterra model, the Wiener model can model memory loudspeaker [17], it consists of two parts, a cascade of a FIR filter and a memoryless polynomial filter, as shown in Fig 2.3.

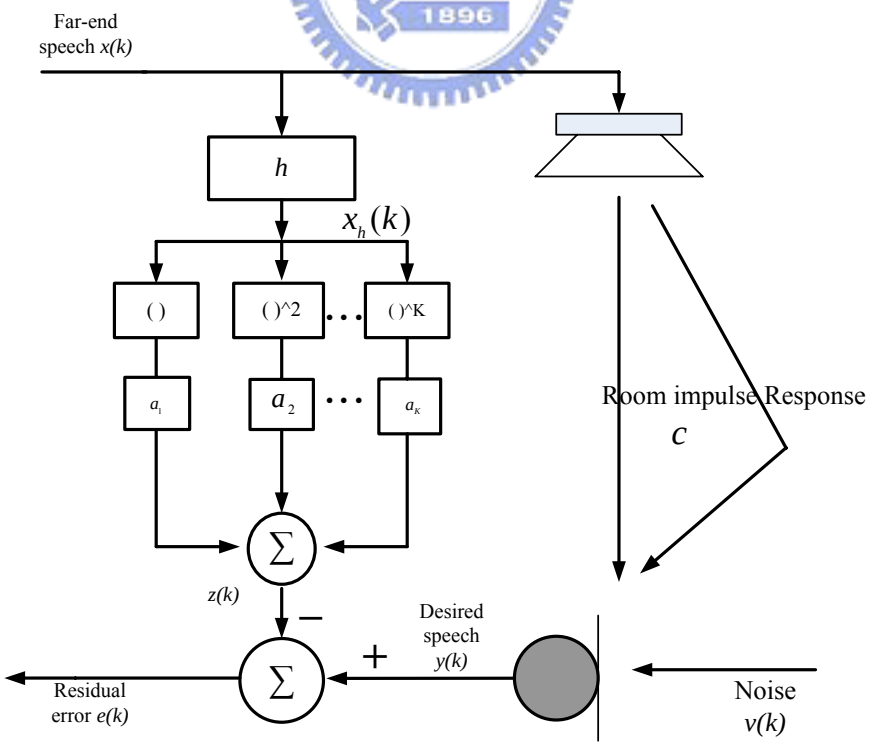


Fig 2.3 Wiener Model

The output of *K*-order Wiener model $z(k)$ can be expressed by

$$z(k) = \sum_{i=1}^{K} a_i [x_h(k)]^i \tag{2.5}$$

where the $x_h(k)$ is FIR output and *M* is FIR memory length

$$x_h(k) = \sum_{l=1}^{M} h_l x(k-l) \tag{2.6}$$

[C] Wiener and Hammerstein Model

We have already introduced the Wiener structure and the Hammerstein structure, Bershad [18] proposed the combination of the two structures, it cascades a FIR filter, a memoryless polynomial filter and a FIR filter.

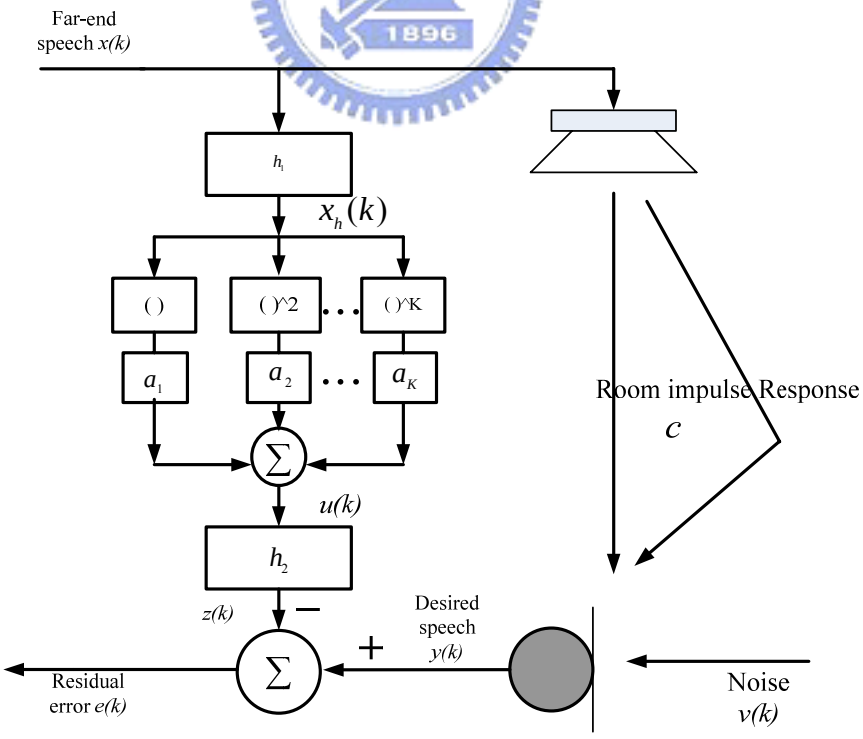


Fig 2.4 Wiener and Hammerstein structure

As shown in Fig 2.4, the $x_h(k)$ denote output of first FIR filter, M_1 and M_2 are memory length of h_1 and h_2, respectively. The $u(k)$ denote the output of memoryless polynomial filter with K^{th}-order.

$$x_h(k) = \sum_{l=0}^{M_1-1} h_{1l} x(k-l)$$

$$u(k) = \sum_{i=1}^{K} a_i [x_h(k)]^i$$

Thus the output of Wiener-Hammerstein structure can be expressed as:

$$z(k) = \sum_{l=0}^{M_2-1} h_{2l} u(k-l)$$

We have already introduced general nonlinear structures in this section, we can summarize and compare those structures. (a) The advantage of both Wiener and Hammerstein models is fewer parameters are needed, the disadvantage is low convergence rate because the parameters of nonlinear and linear (i.e. a and h) are dependent. (b) The advantage of Volterra model is that it can care all terms of distortion causes by nonlinear loudspeaker, thus the performance of Volterra model is the best to the other models. The disadvantage is that the computational complexity is most which causes low convergence rate. (c) All nonlinear models can be considered as a particular subclass of Volterra model.

In this thesis, we will focus on Hammerstein and Volterra models, which the main drawback is low convergence rate. To overcome low convergence rate, many work has been proposed, for example, input decorrelation[21], Orthogonal polynomial-basis [22-23], step size control [12],[19], and so on. We will introduce those approaches in next section.

2.2 Convergence rate speed-up algorithms For NAEC

To accelerate convergence speed, there have been algorithm such as input decorrelation[21], orthogonal polynomial basis [22], and step sized control [12],[19]. We will discuss the first two algorithms here. In Chapter3, we will introduce step size control approaches and propose new step size approach for NAEC.

2.2.1 Input Signal decorrelation

In the field of acoustic echo cancellation, such undesired signal components are removed by adaptive filtering. However, the adaptation performance of the LMS algorithm suffers form slow convergence if the input signal is strongly correlated.

A way to overcome this problem is first decorrlate the input signal, and then uses the decorrelated signal as excitation for the adaptation of the echo canceller.

Kuech [21] proposed an efficient configuration of decorrelation filters for use within nonlinear AEC is derived for second-order Volterra filter, it assumed that the unknown echo can be modeled by a finite-length second-order Volterra filter. It can be shown as follows:

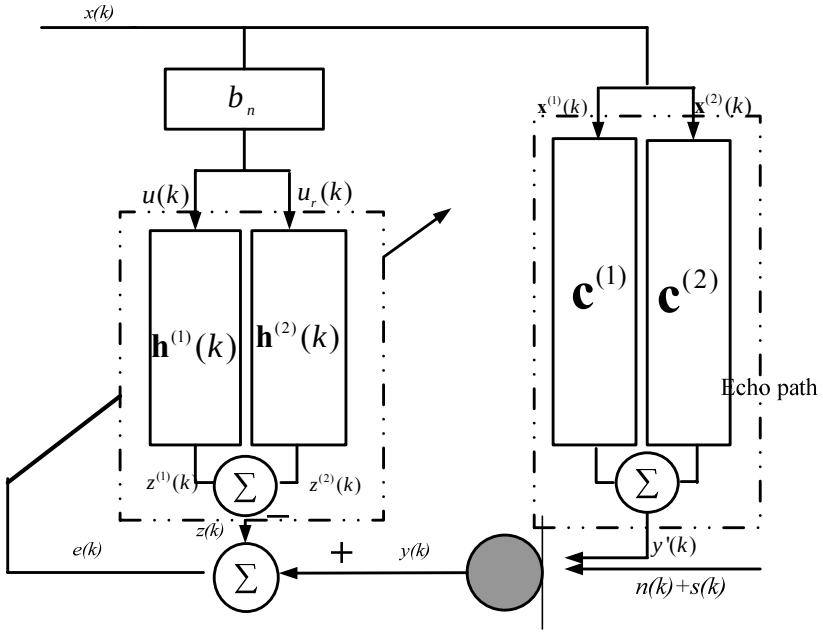


Fig 2.5 Second-order Volterra with decorrelation filter

An optimum decorrelation requires a signal in Kuech [21]

$$u(k) = \sum_{n=0}^{K_{AR}} b_n x(k-n)$$

where K_{AR} denotes AR (autoregressive) random process order, and

$$b_0 = 1$$
$$b_n = -b_{AR,n} \quad \forall 1 \le n \le K_{AR}$$

where b_n is used for the following orthogonality relations hold for $u(k)$ and its produces $u_r(k)$, respectively :

$$E\{u(k-a)u_r(k)\} = 0 \qquad \forall\ r$$
$$E\{u_r(k-a)u_s(k)\} = 0 \qquad \forall\ r \neq s$$
$$E\{u(k-a)u(k)\} = 0 \qquad \forall\ a \neq 0$$
$$E\{u_r(k-a)u_r(k)\} = 0 \qquad \forall\ a \neq 0 \wedge r \neq 0$$

Here, the adaptive equations by means of a joint normalize LMS algorithm read:

$$h_l^{(1)}(k+1) = h_l^{(1)}(k) + \mu e(k)u(k-l)$$
$$h_{l_1,l_2}^{(2)}(k+1) = h_{l_1,l_2}^{(2)}(k) + \mu e(k)u_{l_1}(k-l_2)$$

where the $u_r(k)$ denotes input of quadratic kernel

$$u_r(k) \triangleq u(k)u(k-r)$$

2.2.2 Orthogonal polynomial-basis

In [22-23], G. Y. Jiang and Kuech proposed an orthogonal polynomial adaptive filter to accelerate the convergence of the polynomial model. In general, the input signals of each channel are not mutually orthogonal, i.e. $E\{x^i(k)x^j(k)\} \neq 0, \forall i \neq j$. Thus, a new set of mutually orthogonal input signal has been introduced [23]:

$$p_1(k) = x(k)$$

$$p_u(k) = x^u(k) + \sum_{i=1}^{u-1} q_{u,i} x^i(k)$$

for $1 < u < K$. The orthogonalization coefficients $q_{u,i}$ can be determined using the Gram-Schmidt orthogonalization processing.

In addition to those approaches, step size control is also usually used to overcome the problem of low convergence rate. In Chapter 3, we will introduce several conventional step size algorithms and proposed new step size control approach.

Chapter 3

Optimum Step Size For Nonlinear AEC

In addition to [21-23], step size control is also usually used to overcome the problem of low convergence rate.

In this Chapter, we will focus on step size control in Volterra structure. We know tradeoff between fast convergence rate and small residual error power. In LMS algorithm, normally, large step size gives a faster convergence rate but large residual error power. Thus the optimum step size means that providing fast convergence rate and small residual error power at the same time.

In following sections, the conventional step size control is introduced in section 3.1. In Section 3.2, we will derive the optimum time-& tap-variant step-size LMS (OTTLMS) algorithm which is derived by introducing an optimality criterion which is given by MMSE between coefficients errors of real kernel and adaptive coefficients. Its practical implementations are proposed in Section 3.3. The echo path change and double talk conditions are considered in Section 3.4.

3.1 Conventional step size adjustment

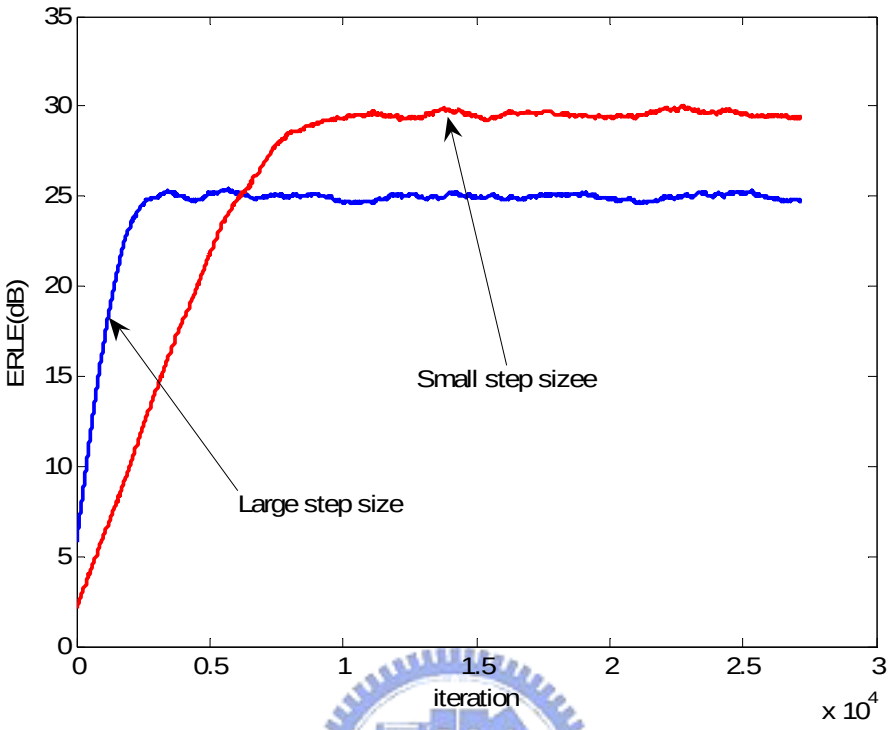


Fig 3.1 Trade off of LMS algorithm

In Fig 3.1, the evaluate of echo return loss enhancement; we can obtain that due to tradeoff between fast convergence rate and small residual error in traditional constant step size, various approaches employing varying step-size in linear echo cancellation have been proposed, including time-varying [9], tap-varying [10], and both time- & tap varying [11]. In this thesis, the word "time-varying" represents all taps use identical step size which is time variant. Similarly, the word "tap-varying" means each tap has individual and tine-invariant step size, and the word "time- & tap-varying" means each tap has its individual time-variant step size.

In addition to linear acoustic echo cancellation, Kuech proposed a time-& tap –varying approach in second- order Volterra structure [12] for nonlinear echo cancellation field.

These typical approaches of step size adjustment for linear AEC are summarized below:

3.1.1 Linear AEC

[A] Variable step size LMS algorithm

The VSLMS approach [9] employs a time-varying (time-variant) step-size which is controlled by the power of the error signal. This is based on using large step-size when the AEC filter coefficient is far from the optimal solution, thus speeding up the convergence rate. Similarity, when the AEC filter coefficient is near the optimum solution, small step-size is used to achieve lower MSE, thus achieving better overall performance. The variable step size LMS algorithm works as follows.

$$\mu'(k+1) = \alpha\mu'(k) + \gamma e^2(k) \quad \text{with } 0 < \alpha < 1,\ \gamma > 0$$

where the time-variant step size is controlled by

$$\mu(k+1) = \begin{cases} \mu_{\max} & \text{if } \mu'(k+1) > \mu_{\max} \\ \mu_{\min} & \text{if } \mu'(k+1) > \mu_{\min} \\ \mu'(k+1) & \text{otherwise} \end{cases}$$

The motivation is that a large residual error i.e., $e(k)$ will cause the large step-size to provide faster tracking the echo path. Similarity, when the residual error is small, the step size is decreased to yield smaller residual error.

The constant $\mu_{\max}$ is chosen to ensure that the MSE remains bounded and $\mu_{\min}$ is chosen to provide a minimum level of tracking ability.

[B] Exponentially weighted step size NLMS algorithm

The exponentially weighted step-size NLMS (ESNLMS) algorithm [10] uses a different step-size (tap-varying) for each tap of an adaptive filter. These step-sizes are time-invariant and weighted proportional to the expected variation of a room impulse response. As a result, the algorithm adjusts coefficients with large echo path variation in large steps, and coefficients with small echo path variation in small steps. The ESNLMS algorithm is expressed as:

$$\mathbf{h}(k+1) = \mathbf{h}(k) + \mathbf{U}_{ESNLMS} \frac{e(k)}{\|\mathbf{x}(k)\|_2^2} \mathbf{x}(k)$$

where $\mathbf{U}_{ESNLMS}$ is the diagonal step-size matrix to account for the tap-variant step-sizes:

$$\mathbf{U}_{ESNLMS} = \begin{bmatrix} \mu_1 & & & 0 \\ & \mu_2 & & \\ & & \ddots & \\ 0 & & & \mu_M \end{bmatrix}$$

where $\mu_l = \mu_0 \gamma^l$ for $l = 1, \ldots, M$ and γ is the room exponential attenuation factor $(0 < \gamma < 1)$.

The elements μ_l are time-invariant and decrease exponentially from μ_1 to μ_M with the same ratio γ that depends on the decay rate of the real room impulse response $\mathbf{c}$.

[C] Optimum time-& tap-variant step size algorithm

The OTTLMS approach [11] is employed to minimize each tap coefficient error variance at each iteration step (i.e. $g_l(k)$). The coefficient error is the difference

between real kernel and adaptive coefficients. The optimum step size can be obtained by setting the derivative of tap coefficient error variance formula with respect to $\mu_l(k)$ equal to zero. The OTTLMS algorithm is expressed as:

$$\mathbf{h}(k+1) = \mathbf{h}(k) + \mathbf{U}_{OTTLMS}(k)e(k)\mathbf{x}(k)$$

where

$$\mathbf{U}_{OTTLMS}(k) = \begin{bmatrix} \mu_{1,OTTLMS}(k) & & 0 \\ & \ddots & \\ 0 & & u_{M,OTTLMS}(k) \end{bmatrix},$$

$$\mu_{l,OTTLMS}(k) = \frac{g_l(k)}{2\sigma_x^2 g_l(k) + \sigma_x^2 \sum_{l=1}^{M} g_l(k) + \sigma_n^2},$$

$$g_l(k+1) = (1 - \sigma_x^2 \mu_{OTTLMS}(k)) g_l(k)$$

As the optimum step-size needs to know the room impulse response to evaluate coefficient error, it is not accessible in general, thus the author employed the recursive relation of second moment coefficient error and used the room impulse response exponential decay model for practical implementation [11].

3.1.2 Nonlinear AEC

We have already introduced step size control applied to linear AEC. However, in the nonlinear AEC application, it faced low convergence rate badly due to high computational complexity. In this section, we want to introduce several step size control approaches applied to nonlinear AEC

[A] Proportionate NLMS for second-order Volterra filters

For acoustic echo cancellation, it is reasonable to assume that the echo path is sparse, i.e., many coefficients are zeros, therefore only the nonzero active coefficients

need to be identified (updated). This is the idea behind the proportionate NLMS (PNLMS) [20] algorithm. It exploits the sparseness of such impulse response to achieve significantly faster adaptation than NLMS.

Kuech [19] proposed an extension of the proportionate NLMS to second-order Volterra filters. It assumes that the unknown echo can be modeled by a finite-length second-order Volterra filter. The nonlinear echo cancellation system model is summarized in Figure 3.2; the microphone signal $y(k)$ is composed of echo signal $y'(k)$, the noise signal $n(k)$ accounting for background noise, and the speech signal of a near-end talker $s(k)$.

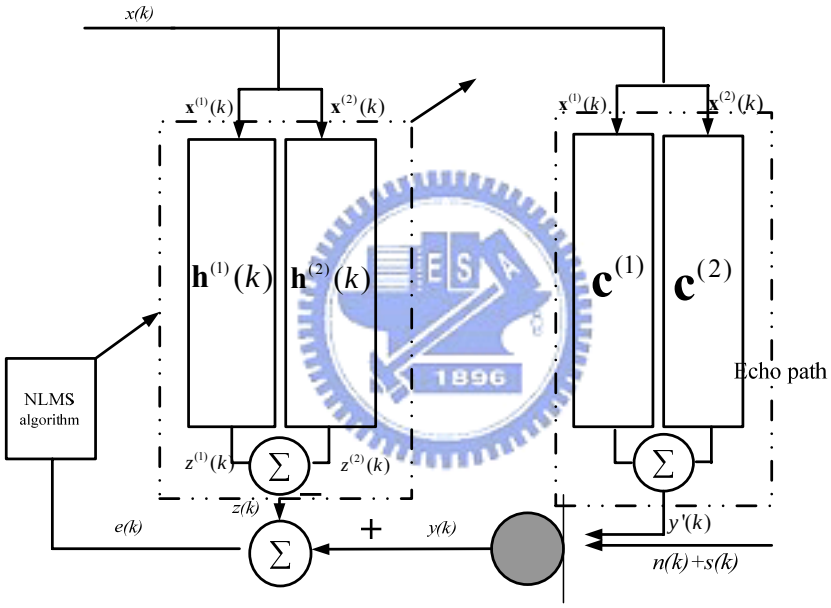


Fig 3.2 Second-order Volterra acoustic echo canceller

By (2.3) (2.4), the input/output relation of a second-order Volterra filter is given by

$$\begin{aligned} z(k) &= z^{(1)}(k) + z^{(2)}(k) \\ &= \sum_{l=0}^{M-1} h_l^{(1)}(k)x(k-l) + \sum_{l_1=0}^{N_2-1}\sum_{l_2=l_1}^{N_2-1} h_{l_1,l_2}^{(2)}(k)x(k-l_1)x(k-l_2) \\ &\triangleq \sum_{l=1}^{M} h_l^{(1)}(k)x_l^{(1)}(k) + \sum_{j=1}^{L_2} h_j^{(2)}(k)x_j^{(2)}(k) \\ &\triangleq [\mathbf{h}^{(1)}]^T(k)\mathbf{x}^{(1)}(k) + [\mathbf{h}^{(2)}]^T(k)\mathbf{x}^{(2)}(k) \end{aligned}$$

To obtain a compact vector representation, we define

$$\mathbf{x}^{(1)}(k) = [x_1^{(1)}(k), x_2^{(1)}(k), ...x_M^{(1)}(k)]^T$$
$$= [x(k), x(k-1), ...x(k-M+1)]^T$$
$$\mathbf{x}^{(2)}(k) = [x_1^{(2)}(k), x_2^{(2)}(k), ...x_{L_2}^{(2)}(k)]^T$$
$$= [x^2(k), x(k)x(k-1), ...x^2(k-N_2+1)]^T$$
$$\mathbf{h}^{(1)}(k) = [h_1^{(1)}(k), h_2^{(1)}(k), ...h_M^{(1)}(k)]^T$$
$$\mathbf{h}^{(2)}(k) = [h_1^{(1)}(k), h_2^{(1)}(k), ...h_{L_2}^{(1)}(k)]^T$$
$$= [h_{0,0}^{(2)}(k), h_{0,1}^{(2)}(k), ...h_{N_2-1,N_2-1}^{(2)}(k)]^T$$

M and N_2 represent memory lengths of linear and quadratic kernel, the lengths of $\mathbf{x}^{(2)}(k)$ and $\mathbf{h}^{(2)}(k)$ are both $L_2 = N_2(N_2+1)/2$

The PNLMS algorithm updates each coefficient of the filter independently of the others by adjusting the adaptation step-size in proportion to the estimated filter coefficient. Thus the extension of the proportionate NLMS to second-order Volterra filters is summarized:

$$\mathbf{h}^{(i)}(k+1) = \mathbf{h}^{(i)}(k) + \mu \frac{\mathbf{P}^{(i)}(k)\mathbf{x}^{(i)}\hat{e}_i(k)}{[\mathbf{x}^{(i)}]^T \mathbf{P}^{(i)}(k)\mathbf{x}^{(i)}}$$
$$\mathbf{P}^{(1)}(k) = diag\left\{p_1^{(1)}(k), ..., p_M^{(1)}(k)\right\}$$
$$\mathbf{P}^{(2)}(k) = diag\left\{p_1^{(2)}(k), ..., p_{L_2}^{(2)}(k)\right\}$$

$$p_l^{(1)}(k) = \frac{1-\alpha}{2M} + (1+\alpha)\frac{|h_l^{(1)}(k)|}{2\|\mathbf{h}^{(1)}(k)\|_1}$$
$$p_l^{(2)}(k) = \frac{1-\alpha}{2L_2} + (1+\alpha)\frac{|h_l^{(2)}(k)|}{2\|\mathbf{h}^{(2)}(k)\|_1}$$

For $i \in \{1,2\}$, $\hat{e}_i(k)$ is used to avoid unstable behavior [19], α is a scalar.

The step-sizes are calculated from the last estimate of the filter coefficients so that a large coefficient receives a large step-size, it is intuitive that if the someone tap of adaptive filter coefficient (i.e. $h_l^{(i)}(k)$) is large value, the coefficient error of this tap should be large, thus if we give large step size to update, it may be increase the convergence rate. Hence, PNLMS converges much faster than NLMS.

We observe that both ESNLMS and PNLMS algorithms rely on the concept of using large step-size for large tap. It is quite intuitive that large tap will produce large estimate tap coefficient error and should use a large step-size for fast tracking. This is appropriate at the stage of initial adaptation.

[B] Optimum step-size for adaptive second-order Volterra filters

The approach [A] is the direct concept of using adaptive filter coefficients to control step size. In approach [B], Kuech [12] derived the optimum step size theoretically and proposed approximated model to practical application.

The concept of the optimum step size in Kuech approach [12] is identical to OTTLMS approach which is derived by introducing minimum MSE between the coefficient errors of Volterra filter and real echo path.

The desired optimum step sizes for linear and quadratic kernel are shown below respectively

$$\mu_{l,opt}^{(1)}(k) = \frac{E\left\{[v_l^{(1)}(k)]^2\right\}}{E\left\{\varepsilon^2(k) + n^2(k) + s^2(k)\right\}},$$

$$\mu_{j,opt}^{(2)}(k) = \frac{E\{[v_j^{(2)}(k)]^2\}}{E\{\varepsilon^2(k) + n^2(k) + s^2(k)\}} \tag{3.1}$$

The linear and quadratic kernel coefficient error in time *k* can be defined by

$$\mathbf{v}^{(1)}(k) = \mathbf{h}^{(1)}(k) - \mathbf{c}^{(1)} \tag{3.2}$$

$$\mathbf{v}^{(2)}(k) = \mathbf{h}^{(2)}(k) - \mathbf{c}^{(2)} \tag{3.3}$$

where

$$\mathbf{v}^{(1)}(k) = [v_1^{(1)}(k), v_2^{(1)}(k), \ldots\ldots v_M^{(1)}(k)]^T$$
$$\mathbf{v}^{(2)}(k) = [v_1^{(2)}(k), v_2^{(2)}(k), \ldots\ldots v_{L_2}^{(2)}(k)]^T$$

The residual echo comes from filter coefficient errors of the linear and quadratic kernel

$$\begin{aligned}\varepsilon(k) &= \varepsilon^{(1)}(k) + \varepsilon^{(2)}(k) \\ &= \mathbf{x}^{(1)T}(k)\mathbf{v}^{(1)}(k) + \mathbf{x}^{(2)T}(k)\mathbf{v}^{(2)}(k)\end{aligned}$$

For a better understanding of the optimum step size, in [12], the author introduced the auxiliary step size factors:

$$\mu_{dt}(k) = \frac{E\{\varepsilon^2(k) + n^2(k)\}}{E\{\varepsilon^2(k) + n^2(k) + s^2(k)\}}, \quad \mu_{bn}(k) = \frac{E\{\varepsilon^2(k)\}}{E\{\varepsilon^2(k) + n^2(k)\}}$$

$$\mu_{\varepsilon_i}(k) = \frac{E\{[\varepsilon^{(i)}(k)]^2\}}{E\{\varepsilon^2(k)\}} \quad i \in \{1,2\} \quad , \quad \alpha_l^{(1)}(k) = \frac{E\{[v_l^{(1)}(k)]^2\}}{\sum_{l=1}^{M} E\{[v_l^{(1)}(k)]^2\} E\{x_j^{(1)}(k)\}}$$

$$\alpha_j^{(2)}(k) = \frac{E\{[v_j^{(2)}(k)]^2\}}{\sum_{j=1}^{L_2} E\left\{\left[v_j^{(2)}(k)\right]^2\right\} E\left\{\left[x_j^{(2)}(k)\right]^2\right\}}$$

The above definitions of step-size factors are used to factorize the optimum step sizes according to

$$\mu_{l,opt}^{(1)}(k) = \mu_{dt}(k)\mu_{bn}(k)\mu_{\varepsilon_1}(k)\alpha_l^{(1)}(k)$$
$$\mu_{j,opt}^{(2)}(k) = \mu_{dt}(k)\mu_{bn}(k)\mu_{\varepsilon_2}(k)\alpha_j^{(2)}(k)$$

As the parameters $E\{[\varepsilon^{(i)}(k)]^2\}$, $E\{[v_l^{(1)}(k)]^2\}$ and $E\{[v_j^{(2)}(k)]^2\}$ are not accessible in general, so the author introduce models for estimating those parameters :

1. The $E\{[\varepsilon^{(i)}(k)]^2\}$ are proportionate to the adaptive filter output of linear and quadratic kernel, respectively

   $$E\{[\varepsilon^{(i)}(k)]^2\} \approx \gamma(k)[\delta_i + \beta_i(k)\overline{|z^{(i)}(k)|}] \quad i \in \{1,2\}$$

2. The second-moment of coefficient error is proportionate to the magnitude of the corresponding adaptive coefficient

   $$E\left\{[v_l^{(1)}(k)]^2\right\} \approx \gamma_1(k)[\rho_1 + \lambda_1(k)|h_l^{(1)}(k)|]$$
   $$E\left\{[v_j^{(2)}(k)]^2\right\} \approx \gamma_2(k)[\rho_2 + \lambda_2(k)|h_j^{(2)}(k)|]$$

For comparison with [11], our work is extending [11] to nonlinear system, second-order Volterra. We employed the advantage of (second moment of coefficient error) recursive relation in [11] and proposed a different practical implementation from Kuech approach [12].

Next, in Section 3.2, we will derive the optimum time-& tap-variant step-size LMS (OTTLMS) algorithm which is derived by introducing an optimality criterion which is given by MMSE between coefficients errors of real kernel and adaptive coefficients. The practical implementation is proposed in section 3.3. The echo path change and double talk conditions are considered in section 3.4.

3.2 Derivation of optimum time-& tap-variant step-size LMS (OTTLMS) algorithm

In this section, we will extend [11] to second-order Volterra filter by getting recursive relation of coefficient errors. By this extension, we not only speed up the convergence rate in linear acoustics echo problem, but also in nonlinear echo cancellation. There our notations are identical to section 3.1 (see Fig 3.2)

We want to find out the step size in time k which can minimize each tap coefficient error variance in time $k+1$ i.e. MSE for each iteration step. Hence, we use diagonal matrixes $\mathbf{U}^{(1)}(k)$ and $\mathbf{U}^{(2)}(k)$ to replace the step size of conventional LMS algorithm [13], thus it corresponding LMS algorithm can be rewritten as

$$\mathbf{h}^{(1)}(k+1) = \mathbf{h}^{(1)}(k) + \mathbf{U}^{(1)}(k)e(k)\mathbf{x}^{(1)}(k) \tag{3.4}$$

$$\mathbf{h}^{(2)}(k+1) = \mathbf{h}^{(2)}(k) + \mathbf{U}^{(2)}(k)e(k)\mathbf{x}^{(2)}(k) \tag{3.5}$$

$$e(k) = y(k) - \mathbf{h}^{(1)T}(k)\mathbf{x}^{(1)}(k) - \mathbf{h}^{(2)T}(k)\mathbf{x}^{(2)}(k) \tag{3.6}$$

The $\mathbf{U}^{(1)}(k)$, $\mathbf{U}^{(2)}(k)$ denote linear and quadratic step size matrices our interest:

$$\mathbf{U}^{(1)}(k) = \begin{bmatrix} \mu_1^{(1)}(k) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mu_M^{(1)}(k) \end{bmatrix} \quad \mathbf{U}^{(2)}(k) = \begin{bmatrix} \mu_1^{(2)}(k) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mu_{L_2}^{(2)}(k) \end{bmatrix}$$

where the l^{th} element of step size matrices is chosen to minimize l^{th} coefficient error variance in time $k+1$. The criterion is summarized as

$$\mu_l^{(i)}(k) = \underset{\mu_l^{(i)}(k)}{\arg\min}\, E\left\{[h_l^{(i)}(k+1) - c_l^{(i)}]^2\right\}$$

where $i \in 1,2$, and it means the linear and quadratic kernel, respectively.

By (3.2), (3.3), (3.6), we get recursive relation of linear kernel:

$$\begin{aligned} \mathbf{h}^{(1)}(k+1) &= \mathbf{h}^{(1)}(k) + \mathbf{U}^{(1)}(k)e(k)\mathbf{x}^{(1)}(k) \\ &= \mathbf{h}^{(1)}(k) + \mathbf{U}^{(1)}(k)\left\{y'(k) - \left[\mathbf{v}(k) + \mathbf{c}\right]^T \mathbf{x}(k) + n(k) + s(k)\right\}\mathbf{x}^{(1)}(k) \end{aligned} \tag{3.7}$$

where

$$\mathbf{x}(k) = [\mathbf{x}^{(1)T}(k) \quad \mathbf{x}^{(2)T}(k)]^T$$

$$\mathbf{v}(k) = [\mathbf{v}^{(1)T}(k) \quad \mathbf{v}^{(2)T}(k)]^T$$

Using (3.2) and (3.7), we may rewrite the linear kernel coefficient error $\mathbf{v}^{(1)}(k)$

$$\begin{aligned} \mathbf{v}^{(1)}(k+1) = {} & \mathbf{v}^{(1)}(k) - \mathbf{U}^{(1)}\mathbf{v}^T(k)\mathbf{x}(k)\mathbf{x}^{(1)}(k) + \mathbf{U}^{(1)}\left[n(k) + s(k)\right]\mathbf{x}^{(1)}(k) \\ = {} & [\mathbf{I} - \mathbf{U}^{(1)}(k)\mathbf{x}^{(1)}(k)\mathbf{x}^{(1)T}(k)]\mathbf{v}^{(1)}(k) - \mathbf{U}^{(1)}\mathbf{x}^{(1)}(k)\mathbf{x}^{(2)T}(k)\mathbf{v}^{(2)}(k) \\ & + \mathbf{U}^{(1)}n(k)\mathbf{x}^{(1)}(k) + \mathbf{U}^{(1)}s(k)\mathbf{x}^{(1)}(k) \end{aligned} \tag{3.8}$$

Similar to processing in [11], we can derive the autocorrelation matrix of the linear kernel coefficient errors as follows, and by the direct-average method [13]

$$\begin{aligned}\mathbf{R}_{\mathbf{v}^{(1)}}(k+1) \approx & [\mathbf{I}-2\mathbf{U}^{(1)}(k)\mathbf{R}_{\mathbf{x}^{(1)}}(\mathbf{k})]\mathbf{R}_{\mathbf{v}^{(1)}}(k) \\ & +[\mathbf{U}^{(1)}(k)]E\{\mathbf{x}^{(1)}(k)\mathbf{x}^{(1)T}(k)\mathbf{v}^{(1)}(k)\mathbf{v}^{(1)T}(k)\mathbf{x}^{(1)}(k)\mathbf{x}^{(1)T}(k)\}[\mathbf{U}^{(1)}(k)]^T \\ & +[\mathbf{U}^{(1)}(k)]E\{\mathbf{x}^{(1)}(k)\mathbf{x}^{(2)T}(k)\mathbf{v}^{(2)}(k)\mathbf{v}^{(2)T}(k)\mathbf{x}^{(2)}(k)\mathbf{x}^{(1)T}(k)\}[\mathbf{U}^{(1)}(k)]^T \\ & +[\mathbf{U}^{(1)}(k)]\sigma_n^2\mathbf{R}_{\mathbf{x}^{(1)}}(k)+\sigma_s^2\mathbf{R}_{\mathbf{x}^{(1)}}(k)[\mathbf{U}^{(1)}(k)]^T\end{aligned}$$

(3.9)

From formula (3.9), the $E\{\ .\ \}$ denotes expectation. By assumption of the mutual independence of $x(k)$, $n(k)$ and $s(k)$, and probability density function of $x(k)$ is an even function, as then $E\{x^3(k)\}=0$. Thus cross products terms $[\mathbf{I}-\mathbf{U}^{(1)}(k)\mathbf{x}^{(1)}(k)\mathbf{x}^{(1)T}(k)]\mathbf{v}^{(1)}(k)$, $\mathbf{U}^{(1)}\mathbf{x}^{(1)}(k)\mathbf{x}^{(2)T}(k)\mathbf{v}^{(2)}(k)$, $\mathbf{U}^{(1)}n(k)\mathbf{x}^{(1)}(k)$ and $\mathbf{U}^{(1)}s(k)\mathbf{x}^{(1)}(k)$ in formula (3.9) could be neglected.

The l'th diagonal term of autocorrelation matrix, denoting l'th mean-square of linear coefficient error, can be written as:

$$\begin{aligned}g_l^{(1)}(k+1) &= E\left\{\left[v_l^{(1)}(k+1)\right]^2\right\} \\ &\approx (1-2\mu_l^{(1)}\sigma_x^2)g_l^{(1)}(k) \\ &\quad +\mu_l^{(1)2}(k)m_{x^4}g_l^{(1)}(k)+\mu_l^{(1)2}(k)\sigma_x^4\sum_{p=1,p\neq l}^{M}g_p^{(1)}(k) \\ &\quad +\mu_l^{(1)2}(k)m_{x^6}g_j^{(2)}(k)+\mu_l^{(1)2}(k)\sigma_x^6\sum_{q=0,q\neq j}^{L_2}g_q^{(2)}(k) \\ &\quad +\mu_l^{(1)2}(k)\sigma_x^2\sigma_n^2\ +\mu_l^{(1)2}(k)\sigma_s^2\sigma_x^2\end{aligned}$$

(3.10)

where σ_x^2 is the far-end input variance. $m_{x^4}=E\{x^4(k)\}$ and $m_{x^6}=E\{x^6(k)\}$ denoting the 4^{th} and 6^{th} moment of $x^{(1)}(k)$. As the length of linear and quadratic kernel in Volterra M and L_2 is sufficiently large, we can approximate $m_{x^4}\approx\sigma_x^4$ and $m_{x^6}\approx\sigma_x^6$, Thus the Eq(3.10) can be rewritten as

$$g_l^{(1)}(k+1) \approx (1-2\mu_l^{(1)}(k)\sigma_x^2)g_l^{(1)}(k) + \mu_l^{(1)2}(k)\sigma_x^2\left[\sigma_x^2\sum_{l=1}^{M} g_l^{(1)}(k) + \sigma_x^4\sum_{j=0}^{L_2} g_j^{(2)}(k) + \sigma_n^2 + \sigma_s^2\right] \tag{3.11}$$

The optimum time-& tap variant step-size can be obtain by taking derivative of Eq (3.11) with respect to $\mu_l^{(1)}(k)$ and setting the result equal to zero.

$$\nabla_{\mu_l^{(1)}(k)} g_l^{(1)}(k+1) = -2\sigma_x^2 g_l^{(1)}(k) + 2\mu_l^{(1)}(k)\sigma_x^2\sum_{l=1}^{M} g_l^{(1)}(k) + 2\mu_l^{(1)}(k)\sigma_x^2\sigma_n^2 + 2\mu_l^{(1)}(k)\sigma_x^2\sigma_s^2 + 2\mu_l^{(1)}(k)\sigma_x^6\sum_{j=1}^{L_2} g_j^{(2)}(k) \triangleq 0$$

Thus we can get the optimum time-&tap-variant step-size of linear kernel

$$\mu_{l,OTTLMS}^{(1)}(k) = \frac{g_l^{(1)}(k)}{\sigma_x^2\sum_{l=1}^{M} g_l^{(1)}(k) + \sigma_n^2 + \sigma_s^2 + \sigma_x^4\sum_{l=1}^{L_2} g_j^{(2)}(k)} \tag{3.12}$$

Analogously to linear kernel, we can get the optimal step size of quadratic kernel is given by

$$\mu_{j,OTTLMS}^{(2)}(k) = \frac{g_j^{(2)}(k)}{\sigma_x^4\sum_{j=1}^{L_2} g_j^{(2)}(k) + \sigma_n^2 + \sigma_s^2 + \sigma_x^2\sum_{l=1}^{M} g_l^{(1)}(k)} \tag{3.13}$$

From the result of (3.12) and (3.13), we can obtain that the optimum step sizes are direct proportion to the coefficient error variance. If the coefficient error variance large (i.e. initial state), the optimum step sizes are large; and if the coefficient error variance small, the optimum steps are become small to get small residual error, the result fits our intuition.

The numerator of (3.12) and (3.13) mean that the second moment coefficient error of linear and quadratic kernel, respectively (i.e. $g_l^{(1)}(k) \triangleq E\left\{\left[v_l^{(1)}(k)\right]^2\right\}$, $g_j^{(2)}(k) \triangleq E\left\{\left[v_j^{(2)}(j)\right]^2\right\}$), and the denominator of (3.12) and (3.13) mean the summation of residual error power and near-end speech power, i.e.

$$\sigma_x^2 \sum_{l=1}^{M} g_l^{(1)}(k) \triangleq E\left\{\left[\varepsilon^{(1)}(k)\right]^2\right\}$$

$$\sigma_x^4 \sum_{j=1}^{L_2} g_j^{(2)}(k) \triangleq E\left\{\left[\varepsilon^{(2)}(k)\right]^2\right\}$$

Thus we can find that the results of (3.12) and (3.13) fit the work (3.1) in [12].

Similar to processing in [11], we substitute the optimum time-&tap-variant step size of linear and quadratic kernel back to (3.10), thus we can get that the relationship mean-square coefficient errors

$$\begin{aligned} g_l^{(1)}(k+1) &= (1-\mu_{l,OTTLMS}^{(1)}(k)\sigma_x^2)g_l^{(1)}(k) \\ g_j^{(2)}(k+1) &= (1-\mu_{j,OTTLMS}^{(2)}(k)\sigma_x^4)g_j^{(2)}(k) \end{aligned} \tag{3.14}$$

for $l = 1,....,M$, $j = 1,....,L_2$.

We found the results fit the works on tradition AEC [11].

Double talk condition is not considered in this section, we set $s(k) = 0$, the double talk and echo path change conditions will be considered in section 3.5, thus the approximated OTTLMS algorithm for second-order Volterra filter is summarized in Table 3.1:

$$e(k) = d(k) - \mathbf{x}^{(1)T}(k)\mathbf{h}^{(1)}(k) - \mathbf{x}^{(2)T}(k)\mathbf{h}^{(2)}(k)$$

$$\mathbf{h}^{(1)}(k+1) = \mathbf{h}^{(1)}(k) + \Delta\mu \begin{pmatrix} g_1^{(1)}(k) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & g_M^{(1)}(k) \end{pmatrix} \mathbf{x}^{(1)}(k)$$

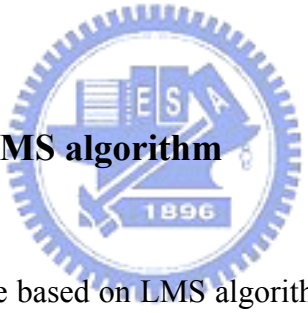
$$\mathbf{h}^{(2)}(k+1) = \mathbf{h}^{(2)}(k) + \Delta\mu \begin{pmatrix} g_1^{(2)}(k) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & g_{L_2}^{(2)}(k) \end{pmatrix} \mathbf{x}^{(2)}(k)$$

$$\Delta\mu = \frac{e(k)}{\sigma_x^4 \sum_{j=1}^{L_2} g_j^{(2)}(k) + \sigma_n^2 + \sigma_x^2 \sum_{l=1}^{M} g_l^{(1)}(k)}$$

$$g_l^{(1)}(k+1) = (1 - \mu_{l,OTTLMS}^{(1)}(k)\sigma_x^2) g_l^{(1)}(k)$$

$$g_j^{(2)}(k+1) = (1 - \mu_{j,OTTLMS}^{(2)}(k)\sigma_x^4) g_j^{(2)}(k)$$

Table 3.1: OTTLMS algorithm



3.3 Extension to OTTNLMS algorithm

The above discussions are based on LMS algorithm. However, when the input is large, the LMS algorithm suffers from a gradient noise amplification problem. In order to overcome this difficulty, we extend it to the normalized LMS (NLMS) algorithm. By the approximation [13] of $\mathbf{x}^{(1)T}(k)\mathbf{x}^{(1)}(k) = M\sigma_x^2$ and $\mathbf{x}^{(2)T}(k)\mathbf{x}^{(2)T}(k) = L_2\sigma_x^4$, the step size of OTTNLMS can be shown to be $\mu_{l,OTTNLMS}^{(1)}(k) = (M\sigma_x^2 + L_2\sigma_x^4)\mu_{l,OTTLMS}^{(1)}(k)$ and $\mu_{j,OTTNLMS}^{(2)}(k) = (M\sigma_x^2 + L_2\sigma_x^4)\mu_{j,OTTLMS}^{(2)}(k)$. So, we can rewrite (3.12) and (3.13) as:

$$\mu_{l,OTTNLMS}^{(1)}(k) = \frac{\left[M\sigma_x^2 + L_2\sigma_x^4\right] g_l^{(1)}(k)}{\sigma_x^2 \sum_{l=1}^{M} g_l^{(1)}(k) + \sigma_n^2 + \sigma_x^4 \sum_{l=1}^{L_2} g_j^{(2)}(k)}$$

$$\mu_{j,OTTNLMS}^{(2)}(k)=\frac{\left[M\sigma_x^2+L_2\sigma_x^4\right]g_j^{(2)}(k)}{\sigma_x^4\sum_{j=1}^{L_2}g_j^{(2)}(k)+\sigma_n^2+\sigma_x^2\sum_{l=1}^{M}g_l^{(1)}(k)}$$

Similarly, the (3.14) can be rewritten as:

$$g_l^{(1)}(k+1)=(1-\frac{\mu_{l,OTTLMS}^{(1)}(k)\sigma_x^2}{M\sigma_x^2+L_2\sigma_x^4})g_l^{(1)}(k)$$

$$g_j^{(2)}(k+1)=(1-\frac{\mu_{j,OTTLMS}^{(2)}(k)\sigma_x^4}{M\sigma_x^2+L_2\sigma_x^4})g_j^{(2)}(k)$$

for $l=1,....,M$, $j=1,....,L_2$.

Thus the OTTNLMS algorithm for second-order Volterra filter is summarized in Table 3.2:

$$e(k)=d(k)-\mathbf{x}_1^T(k)\mathbf{h}^{(1)}(k)-\mathbf{x}_2^T(k)\mathbf{h}^{(2)}(k)$$

$$\mathbf{h}^{(1)}(k+1)=\mathbf{h}^{(1)}(k)+\Delta\mu\begin{pmatrix}g_1^{(1)}(k)&\cdots&0\\ \vdots&\ddots&\vdots\\ 0&\cdots&g_M^{(1)}(k)\end{pmatrix}\frac{\mathbf{x}^{(1)}(k)}{\|\mathbf{x}(k)\|_2^2}$$

$$\mathbf{h}^{(2)}(k+1)=\mathbf{h}^{(2)}(k)+\Delta\mu\begin{pmatrix}g_1^{(2)}(k)&\cdots&0\\ \vdots&\ddots&\vdots\\ 0&\cdots&g_{L_2}^{(2)}(k)\end{pmatrix}\frac{\mathbf{x}^{(2)}(k)}{\|\mathbf{x}(k)\|_2^2}$$

$$\Delta\mu=\frac{\left[M\sigma_x^2+L_2\sigma_x^4\right]e(k)}{\sigma_x^4\sum_{j=1}^{L_2}g_j^{(2)}(k)+\sigma_n^2+\sigma_x^2\sum_{l=1}^{M}g_l^{(1)}(k)}$$

$$g_l^{(1)}(k+1)=(1-\frac{\mu_{l,OTTNLMS}^{(1)}(k)\sigma_x^2}{M\sigma_x^2+L_2\sigma_x^4})g_l^{(1)}(k)$$

$$g_j^{(2)}(k+1)=(1-\frac{\mu_{j,OTTNLMS}^{(2)}(k)\sigma_x^4}{M\sigma_x^2+L_2\sigma_x^4})g_j^{(2)}(k)$$

Table 3.2 OTTNLMS algorithm

3.4 Practical implementations of OTTLMS algorithm

In Section 3.2, we have already derived optimum time-&tap variant step-size for LMS and NLMS algorithm in Table 3.1 and Table 3.2. Here, the OTTLMS and OTTNLMS not only need prior statistics knowledge σ_x^2, σ_x^4, and σ_n^2, but also the prior knowledge of second moment of coefficient error $g_l^{(1)}(k)$ and $g_j^{(2)}(k)$. Thus we must be known the real room impulse response $\mathbf{c}^{(1)}$ and second order kernel caused by nonlinear loudspeaker $\mathbf{c}^{(2)}$. In general case, the echo path $\mathbf{c}^{(1)}$ and $\mathbf{c}^{(2)}$ are not accessible. In section 3.3.1, we propose a model function to estimate those parameters for the application in nonlinear acoustics echo cancellation.

3.4.1 Exponential models of linear and quadratic kernel

Unlike the approximation approach of Kuech approach [12] in Section 3.1.2, we introduced the recursive formula (3.14), thus we only need to know the real envelope of real echo path (i.e. $g_l^{(1)}(0) \triangleq E\{\left[h_l^{(1)}(0) - c_l^{(1)}\right]^2\} \triangleq \left[c_l^{(1)}\right]^2$), thus we proposed an exponentially models for implementation.

Here, we will assume reasonably the real linear and quadratic kernel $\mathbf{c}^{(1)}$, $\mathbf{c}^{(2)}$ can be modeled as an exponentially decaying envelope shown in Fig 3.3, and Fig 3.5. Let the linear and quadratic envelope functions modeled as:

$$w_l^{(1)} = w_0^{(1)}(r^{(1)})^l \quad \text{for } l = 1 \sim M \tag{3.15}$$

$$w_{l_1,l_2}^{(2)} = w_0^{(2)}(r^{(2)})^{(l_1+l_2)} \quad \text{for } l_1, l_2 = 1 \sim N_2 \tag{3.16}$$

where $r^{(1)}$ and $r^{(2)}$ are linear and quadratic kernel exponential decay factors respectively.

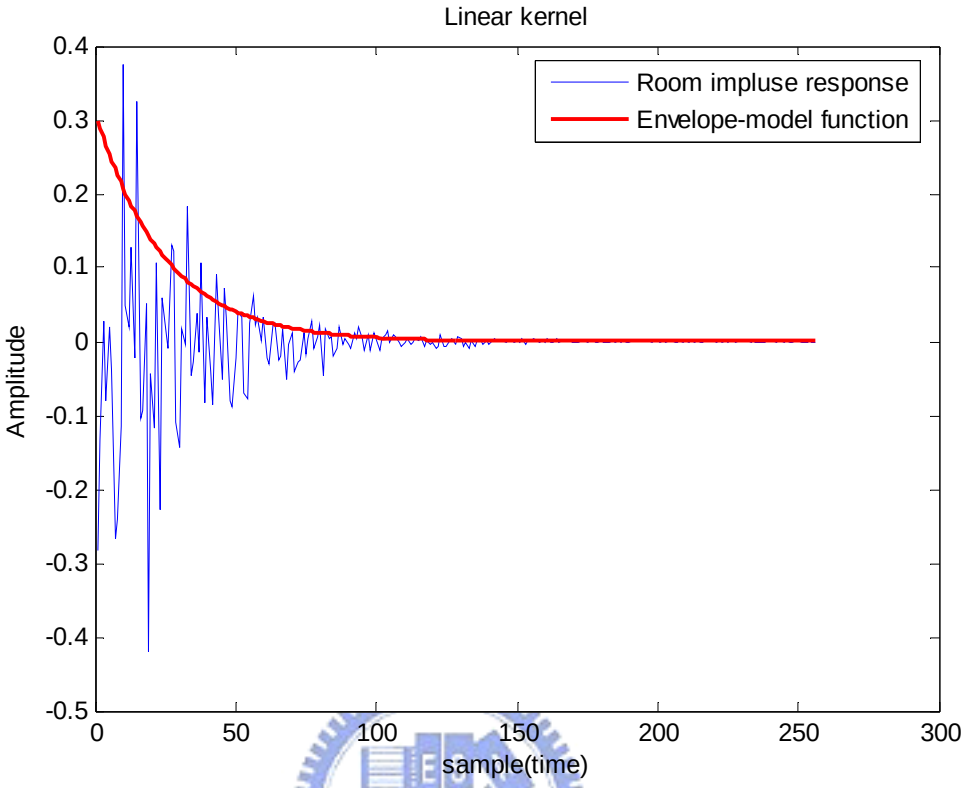


Fig 3.3 Real Linear kernel and exponential model of the envelope

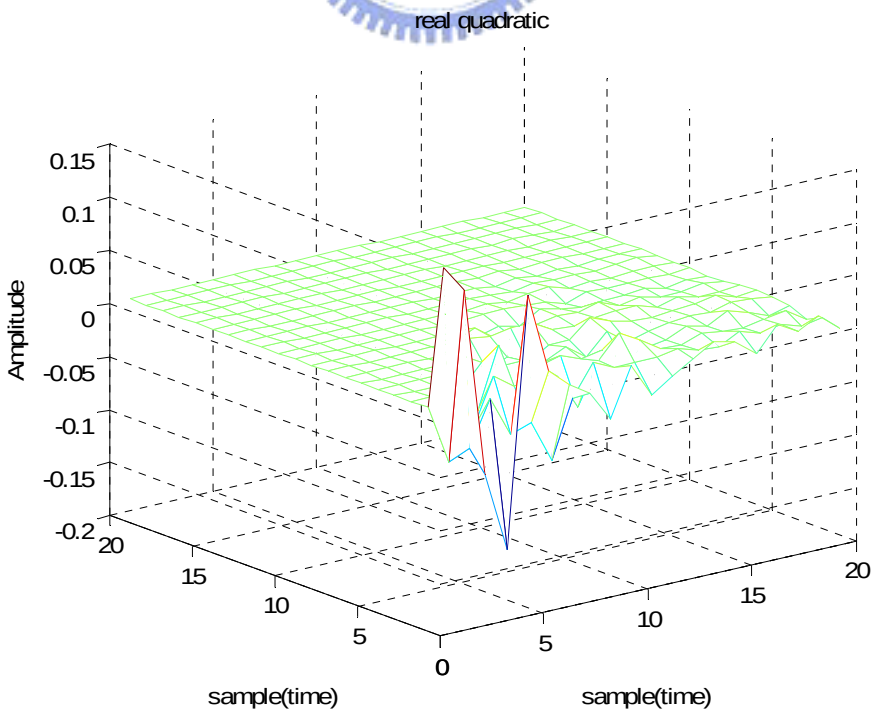


Fig 3.4 Real quadratic kernel of the nonlinear loudspeaker

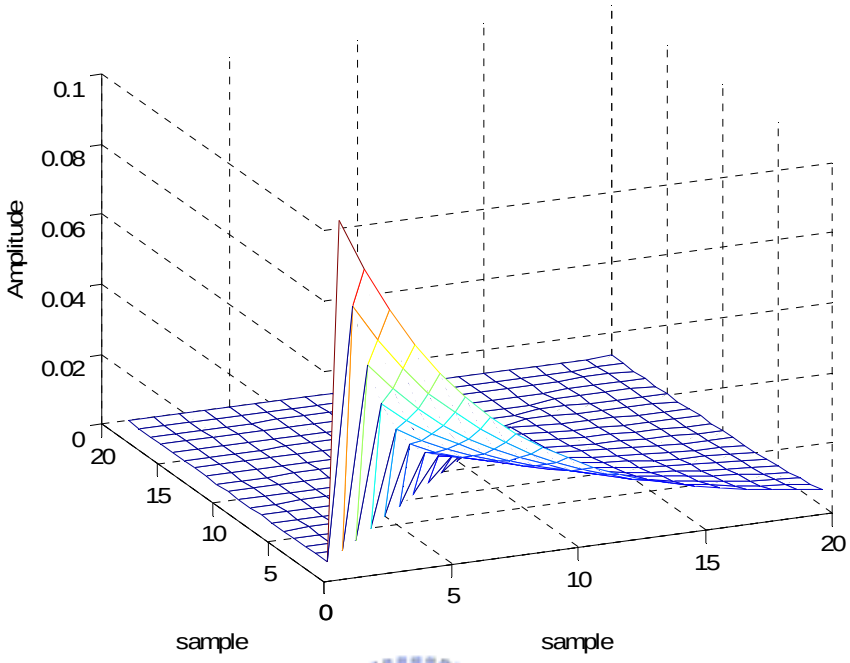


Fig 3.5 Exponential model of the envelope of the quadratic kernel

The diagonal elements of tap coefficient error variance matrix $R_{\mathbf{v}^{(1)}}(k)$ and $R_{\mathbf{v}^{(2)}}(k)$ are $g_l^{(1)}(k) = E[(h_l^{(1)}(k) - c_l^{(1)})^2]$ and $g_j^{(2)}(k) = E[(h_j^{(2)}(k) - c_j^{(2)})^2]$, respectively. We let the initial linear and quadratic tap coefficients to be zero. i.e. $h_l^{(1)}(0) = 0$, $h_j^{(2)}(0) = 0$, so $g_l^{(1)}(0) = E[(h_l^{(1)}(0) - c_l^{(1)})^2] = \left[c_l^{(1)}\right]^2 \approx \left[w_l^{(1)}\right]^2$ and $g_j^{(2)}(0) = E[(h_j^{(2)}(0) - c_j^{(2)})^2] = \left[c_j^{(2)}\right]^2 \approx \left[w_{l_1,l_2}^{(2)}\right]^2$. By (3.12) and (3.13), if we have $g_l^{(1)}(0)$ and $g_j^{(2)}(0)$, we can get the initial step-sizes of linear $\mu_{l,OTTLMS}^{(1)}(0)$ and quadratic kernel filter $\mu_{j,OTTLMS}^{(2)}(0)$, with initial step-size plugged into (3.13) we can get $g_l^{(1)}(1)$ and $g_j^{(2)}(1)$, and so forth. Thus, we can find $\mu_{l,OTTLMS}^{(1)}(k)$ and $\mu_{j,OTTLMS}^{(2)}(k)$, recursively. The practical OTTLMS algorithm with exponentially envelope model functions can be summarized as follows:

1. Measure the exponential decay factor of linear and quadratic kernel $r^{(1)}$ and $r^{(2)}$ to get $w_l^{(1)} = w_0^{(1)}(r^{(1)})^l$ and $w_{l_1,l_2}^{(2)} = w_2^{(2)}(r^{(2)})^{(l_1+l_2)}$.

2. Set up initial value $g_l^{(1)}(0) \approx \left[w_l^{(1)} \right]^2$ for $l = 1,\ldots,M$ and $g_j^{(2)}(0) \approx \left[w_{l_1,l_2}^{(2)} \right]^2$ for for $l_1, l_2 = 1,\ldots,N_2$

3. According to table.3.1

By using the exponential function to model the linear and quadratic kernel, we can practically implement the OTTLMS algorithm, whose performance will be verified in Chapter 5.

3.4.2 Exponentially approximated temporal function of step-size in OTLMS and OTNLMS

In Section 3.3, we proposed practical implement in OTTLMS. Now we would further obtain property of step size in adaptive processing.

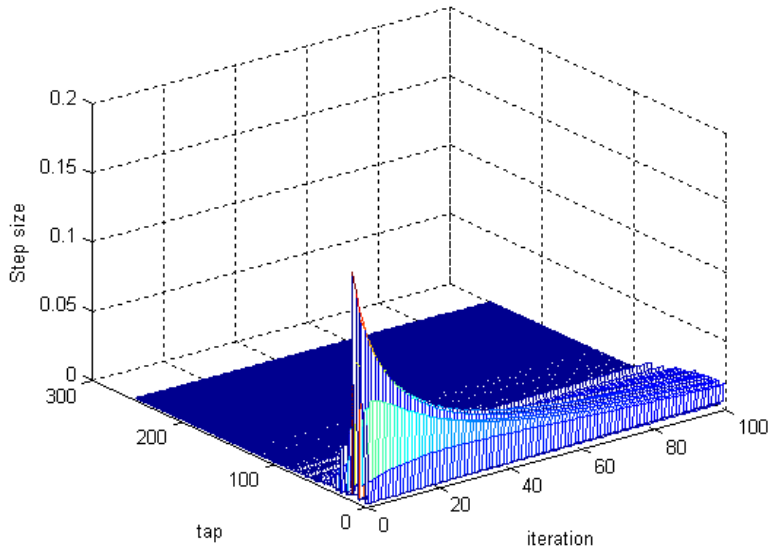


Fig 3.6 Linear kernel step size temporal function

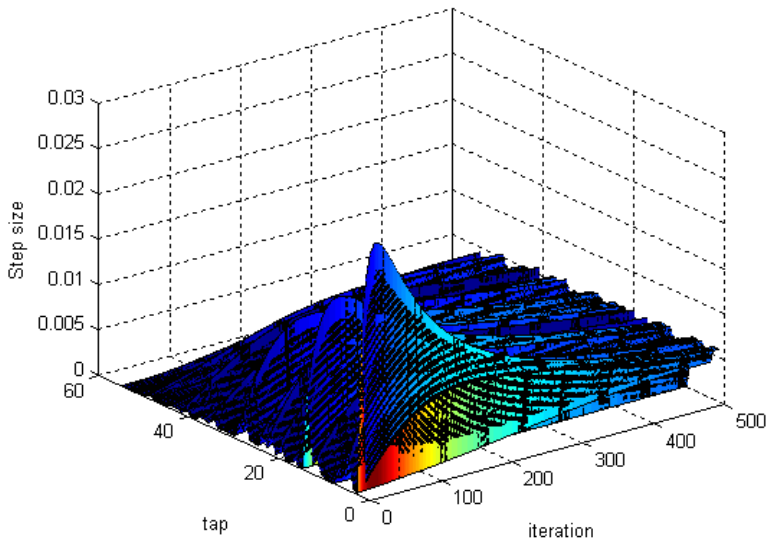


Fig 3.7 Quadratic kernel step size temporal function

By Fig 3.6 and 3.7, for our optimum step size in AEC, we obtain that large step size in initial times and small in converged times.

To reduce computation complexity, we want to derive the exponentially approximation temporal function of step size, with the approximated step size function, we can abbreviate the calculation of (second moment) coefficient error.

To simplify calculation, we assume that all taps of step size are equal, thus we can rewrite optimum step sizes (3.12) and (3.13) as

$$\mu_{OTTLMS}^{(1)}(k)=\frac{g^{(1)}(k)}{\sigma_x^2 M g^{(1)}(k)+\sigma_n^2+\sigma_x^4 L_2 g^{(2)}(k)} \tag{3.17}$$

$$\mu_{OTTLMS}^{(2)}(k)=\frac{g^{(2)}(k)}{\sigma_x^4 L_2 g^{(2)}(k)+\sigma_n^2+\sigma_x^2 M g^{(1)}(k)} \tag{3.18}$$

Similarly, the mean-square coefficient error could be rewritten as

$$\begin{aligned} g^{(1)}(k+1)&=(1-\mu_{OTTLMS}^{(1)}(k)\sigma_x^2)g^{(1)}(k)\\ g^{(2)}(k+1)&=(1-\mu_{OTTLMS}^{(2)}(k)\sigma_x^4)g^{(2)}(k)\end{aligned} \tag{3.19}$$

We put the recursive relation of mean-square coefficient error (3.19) into (3.17), the OTLMS of linear kernel can be expressed as :

$$\begin{aligned}\mu^{(1)}(k)&=\frac{(1-\mu^{(1)}(k-1)\sigma_x^2)g^{(1)}(k-1)}{\sigma_x^2 M(1-\mu^{(1)}(k-1)\sigma_x^2)g^{(1)}(k-1)+\sigma_n^2+\sigma_x^4 L_2(1-\mu^{(2)}(k-1)\sigma_x^4)g^{(2)}(k-1)}\\ &=\frac{\alpha^{(1)}(k-1)g^{(1)}(k-1)}{\sigma_x^2 M\alpha^{(1)}(k-1)g^{(1)}(k-1)+\sigma_n^2+\sigma_x^4 L_2\alpha^{(2)}(k-1)g^{(2)}(k-1)}\\ &=\frac{\prod_{i=0}^{k-1}\alpha^{(1)}(i)g^{(1)}(0)}{\sigma_x^2 M\prod_{i=0}^{k-1}\alpha^{(1)}(i)g^{(1)}(0)+\sigma_n^2+\sigma_x^4 L_2\prod_{i=0}^{k-1}\alpha^{(2)}(i)g^{(2)}(0)}\end{aligned} \tag{3.20}$$

where common ratios of mean-square coefficient error are defined as

$\alpha^{(1)}(i)=(1-\mu^{(1)}(i)\sigma_x^2)$ and $\alpha^{(2)}(i)=(1-\mu^{(2)}(i)\sigma_x^4)$,for time $i=1\sim k-1$

By assumption of the common ratios are identical for the whole time (i.e. $\alpha^{(1)}(k-1)=\alpha^{(1)}(k-2)=\ldots=\alpha^{(1)}(0)=\alpha^{(1)}$ and $\alpha^{(2)}(k-1)=\alpha^{(2)}(k-2)=\ldots=\alpha^{(2)}(0)=\alpha^{(2)}$, and $g^{(1)}(0)=g^{(2)}(0)$, thus we rewrite (3.16) as

$$\mu^{(1)}(k)\ =\ \frac{[\alpha^{(1)}]^k g^{(1)}(0)}{(M[\alpha^{(1)}]^k+L_2[\alpha^{(2)}]^k)g^{(1)}(0)+\sigma_n^2+\sigma_s^2} \tag{3.21}$$

Analogously to linear kernel,

$$\mu^{(2)}(k) = \frac{[\alpha^{(2)}]^k g^{(1)}(0)}{(M[\alpha^{(1)}]^k + L_2[\alpha^{(2)}]^k)g^{(1)}(0) + \sigma_n^2 + \sigma_s^2} \tag{3.22}$$

Similarly, the exponential approximated OTLMS algorithm for second-order Volterra filter is summarized in Table 3.3:

$$e(k) = y(k) - \mathbf{x}^{(1)T}(k)\mathbf{h}^{(1)}(k) - \mathbf{x}^{(2)T}(k)\mathbf{h}^{(2)}(k)$$

$$\mu^{(1)}(k) = \frac{[\alpha^{(1)}]^k g^{(1)}(0)}{(M[\alpha^{(1)}]^k + L_2[\alpha^{(2)}]^k)g^{(1)}(0) + \sigma_n^2}$$

$$\mu^{(2)}(k) = \frac{[\alpha^{(2)}]^k g^{(1)}(0)}{(M[\alpha^{(1)}]^k + L_2[\alpha^{(2)}]^k)g^{(1)}(0) + \sigma_n^2}$$

$$\mathbf{h}^{(1)}(k+1) = \mathbf{h}^{(1)}(k) + \mu^{(1)}(k)e(k)\mathbf{x}^{(1)}(k)$$

$$\mathbf{h}^{(2)}(k+1) = \mathbf{h}^{(2)}(k) + \mu^{(2)}(k)e(k)\mathbf{x}^{(2)}(k)$$

Table 3.3 Exponentially approximated temporal function of step-size in OTLMS

Similarly, for NLMS algorithm, we have:

$$\mu_{OTTNLMS}^{(1)}(k) = (M\sigma_x^2 + L_2\sigma_x^4)\mu_{OTTLMS}^{(1)}(k)$$

$$\mu_{OTTNLMS}^{(2)}(k) = (M\sigma_x^2 + L_2\sigma_x^4)\mu_{OTTLMS}^{(2)}(k)$$

3.5 Double Talk and Echo Path change conditions

We have already proposed OTTLMS algorithm and assumed single talk case in Section 3.2. In this section, we consider the double talk (i.e. $s(k) \neq 0$) and echo path change conditions.

From the (3.12) and (3.13), the determination of $\mu_{l,OTTLMS}^{(1)}(k)$ and $\mu_{j,OTTLMS}^{(2)}(k)$, as statistics knowledge of near-end σ_s^2 are not accessible, it is intuitive that residual error variance $\sigma_e^2(k)$ is near to $\sigma_n^2 + \sigma_s^2$ in converged condition. Thus we

introduced the estimated residual error variance $\hat{\sigma}_e^2(k)$ to model the background noise and near end speech variance for practical implementation which is using the smoothed recursive algorithm from square of residual error:

$$\hat{\sigma}_e^2(k) = \lambda \hat{\sigma}_e^2(k-1) + (1-\lambda)e^2(k)$$

where λ is constant.

Next, we look at the echo path change condition. When the echo path changes, our proposed optimum step size is not robust because our step sizes are still restricted to small value in converged single-talk.

In OTTLMS algorithm, we introduce the recursive relation to evaluate the optimum step size, rather than calculate step size repetitively. These introduction constraints that our optimum step sizes are to be small in converged condition even echo path changes suddenly.

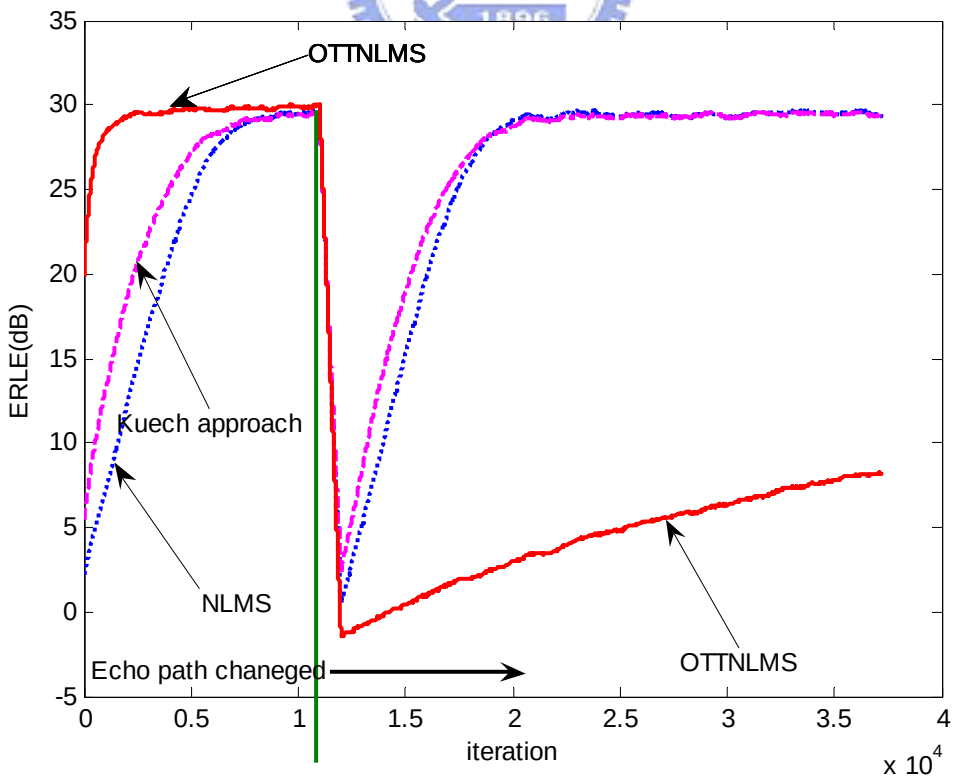


Fig 3.8 OTTLMS during echo-path variations

Fig 3.8 shows the comparison of OTTLMS and LMS during echo-path variations. The room response is changed after 12000 iterations. According to the simulation, the convergence rate of OTTLMS algorithm is increased as compared to the LMS algorithm; however, it can be observed from Fig 3.8, the result is different after echo path changed. The reason is the recursive characteristic of (3.14), whether echo path changes or not, the mean-square coefficient errors of linear and quadratic kernel are smaller and smaller, it leads to our proposed optimum step sizes are smaller and smaller, so even if the echo path changes after 12000 iterations, the step sizes are still very small at convergence.

Thus we introduce a detector [16] to detect the echo path change, by a direct measure of the adaptive filter's convergence. Referring to Fig 3.2, the cross correlation between the desired signal $y(k)$, and the residual error $e(k)$ is given by:

$$\begin{aligned} r_{ey}(k) &= E\left\{e(k)y^T(k)\right\} \\ &= E\left\{\left\{\left[\mathbf{c}^{(1)} - \mathbf{h}^{(1)}(k-1)\right]^T \mathbf{x}^{(1)}(k) + n(k)\right\}\left\{\mathbf{c}^{(1)T}\mathbf{x}^{(1)}(k) + n(k)\right\}\right\} \\ &= \left[\mathbf{c}^{(1)} - \mathbf{h}^{(1)}(k-1)\right]^T \mathbf{R}_{\mathbf{x}^{(1)}}(k)\mathbf{c}^{(1)} + \sigma_n^2 \end{aligned} \tag{3.23}$$

The variance of desired signal $y(k)$ and residual error $e(k)$ are expressed by

$$\sigma_y^2 = E\left\{y^2(k)\right\} = \mathbf{c}^{(1)T}\mathbf{R}_{\mathbf{x}^{(1)}}(k)\mathbf{c}^{(1)} + \sigma_n^2 \tag{3.24}$$

$$\sigma_e^2 = E\left\{e^2(k)\right\} = \left[\mathbf{c}^{(1)} - \mathbf{h}^{(1)}(k-1)\right]^T \mathbf{R}_{\mathbf{x}^{(1)}}(k)\left[\mathbf{c}^{(1)} - \mathbf{h}^{(1)}(k-1)\right] + \sigma_n^2 \tag{3.25}$$

The proposed convergence statistic definition from [16] is given by:

$$\xi(k) = \left|\frac{r_{ey}(k) - \sigma_e^2}{\sigma_y^2 - r_{ey}(k)}\right| = \left|\frac{\left(\mathbf{c}^{(1)} - \mathbf{h}^{(1)}(k-1)\right)^T \mathbf{R}_{\mathbf{x}^{(1)}}\mathbf{h}^{(1)}(k-1)}{\left(\mathbf{c}^{(1)}\right)^T \mathbf{R}_{\mathbf{x}^{(1)}}\mathbf{h}^{(1)}(k-1)}\right| \tag{3.26}$$

We can observe that If the adaptive AEC filter converged (i.e. $c^{(1)} \approx \mathbf{h}^{(1)}(k-1)$), the convergence statistic $\xi(k)$ in (3.29) is approximately to zero, and if $c^{(1)} \neq \mathbf{h}^{(1)}(k-1)$, the statistic $\xi(k)$ is lager than zero ($\xi(k)>0$). Thus the proposed statistic is a good

measure convergence of the adaptive filter.

When the echo path change detector detects the echo path change, we re-initialize from $g_l^{(1)}(k)$ to $g_l^{(1)}(0)$, it means that we re-update the linear adaptive filter $\mathbf{h}^{(1)}$, thus we can overcome the echo path change condition. The simulation result will be shown in Section 5.2.7.

3.6 Computational complexity

We have already discussed OTTLMS algorithm with an increased the convergence rate. This section we make a computation comparison of OTTLMS, LMS, and Kuech approach, we examine the number of multiplications required to make once complete iteration of the algorithm. The recursive relation of second moment coefficient error in (3.13) is need $2M+2L_2$ multiplications, the update equation of OTTLMS in Table 3.1 is need $2M+2L_2$ multiplications, thus the total requirement multiplication of OTTLMS is $5M+5L_2$

Similarly, the multiplications of [12] are about $4M+4L_2$, and the EAOTTLMS (Exponentially approximation temporal function of OTTLMS) in Section 3.4.2, it needs only about $2M+2L_2$.

The computation complexity for OTTLMS are summarized in Table 3.3

Algorithm	Multiplications/sample
LMS	$2M+2L_2$
Kuech[12]	$4M+4L_2$
OTTLMS	$5M+5L_2$
EAOTTLMS	$2M+2L_2$

Table.3.3 Computation complexity comparison of different algorithms

3.7 Summary

In this chapter, we propose the optimum step size in second-order Volterra structure, in section 3.1, we introduced the conventional step size control algorithm, in section 3.2, and the optimum step-size is derived by introducing an optimality criterion which is given by MMSE between coefficients errors of real kernel and adaptive coefficients.

In Section 3.3 we extend to NLMS algorithm. In Section 3.4 we propose exponentially model function to practical implement because the prior knowledge of echo path is not easy to be acquired. To save the computational complexity, the exponentially approximated temporal function was derived in Section 3.5.

In Section 3.6, the echo path change and double talk conditions were considered, and the computational complexity was summarized in Section 3.7. The overall of discussion will be verified in Chapter5.

In addition to step size control of second-order Volterra, the higher-order Volterra model was not considered here. Because the optimum step size deriving processing is more complicated, for example, (3.8) will be have many cross term which is leading to hard to get the second moment of coefficient error in (3.10).

In Hammerstein model, as it is cascade structure, the joint error term produced by linear and nonlinear term, thus it is difficult to perform its optimum step size.

Chapter 4
Channel Shortening Structure For Nonlinear AEC

Another alternative is the channel shortening technique that has been proposed in [14] to overcome high computational complexity and low convergence rate disadvantages of Hammerstein structure.

In Chapter 4, we will investigate the issues of channel shortening approach. The channel shortening approach will be introduced in Section 4.1, in Section 4.2, we will perform theoretical analysis in the senses of LMS and LS in case of a linear loudspeaker to obtain the converged tendency.

In addition to theoretical linear analysis, we will propose multiple nonlinear stage update scheme to accelerate convergence rate in Section 4.3, and finally we apply the channel shortening with second-order Volterra filer in Section 4.4.

4.1 Channel shortening approach

Kun Shi [14] proposed a novel algorithm based on Hammerstein model, Fig.4.1 shows the structure of nonlinear acoustics echo cancellation, it introduced an FIR shortening filter $\mathbf{w}(k)$ is introduced after the acoustics path. The purpose of shortening filter $\mathbf{w}(k)$ is to "shorten" the channel, which is the convolution of the room impulse response and $\mathbf{w}(k)$ to have fewer number of non-negligible taps.

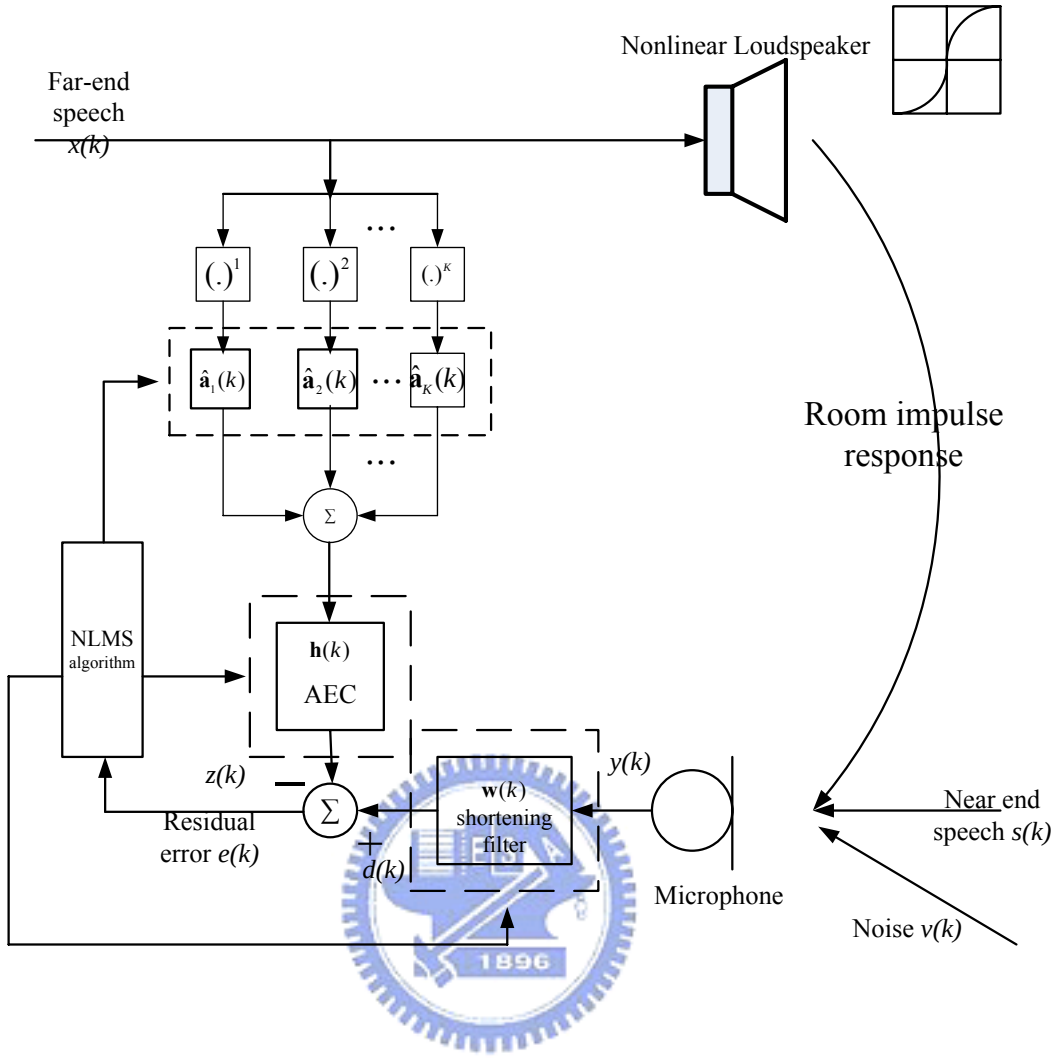


Fig4.1 Channel shortening structure for nonlinear AEC

In [14], the author performed the RLS algorithm for nonlinear polynomial coefficient $\hat{\mathbf{a}}(k)$, and NLMS algorithm for adaptive AEC $\mathbf{h}(k)$ and shortening filter $\mathbf{w}(k)$. Here we will focus the NLMS algorithm.

Suppose that the lengths of shortening filter *w(k)* and the AEC filter $\hat{h}(k)$ are L_w and L_s, respectively. Define the vectors

$$\mathbf{w}(k) = \left[w_1(k), w_2(k), \ldots, w_{L_w}(k) \right]^T$$

$$\mathbf{y}(k) = \left[y(k), y(k-1), \ldots, y(k-L_w+1) \right]^T$$

$$\mathbf{h}(k) = \left[h_1(k), h_2(k), \ldots, h_{Ls}(k) \right]^T$$

$$\mathbf{x}(k) = \left[x(k), x(k-1), \ldots, x(k-L_s+1)\right]^T$$

The reference signal (output of shortening filter) *d(k)* can be defined as

$$d(k) = \mathbf{w}^T(k)\mathbf{y}(k)$$

The nonlinear AEC output signal $z(k)$ can be written as

$$z(k) = \mathbf{h}^T(k)\hat{\mathbf{s}}(k)$$

where the $\hat{\mathbf{s}}(k)$ is the output vector of the nonlinear filter

$$\hat{\mathbf{s}}(k) = [\hat{s}(k),\ \hat{s}(k-1), \ldots, \hat{s}(k-L_s+1)]^T$$

Each $\hat{s}(k)$ is given by

$$\begin{aligned}\hat{s}(k) &= [x^1(k)\ x^2(k)\cdots x^K(k)][\hat{a}_1(k), \hat{a}_2(k)\cdots\hat{a}_K(k)]^T \\ &= \mathbf{x}(k)^T\hat{\mathbf{a}}(k)\end{aligned}$$

where $\hat{\mathbf{a}}(k)$ is the estimated coefficients vector of the nonlinear processor. Therefore, $\hat{\mathbf{s}}(k)$ can be expressed by

$$\begin{aligned}\hat{\mathbf{s}}(k) &= [\hat{s}(k),\ \hat{s}(k-1), \ldots, \hat{s}(k-L_s+1)]^T \\ &= [\ \mathbf{x}(k)^T\hat{\mathbf{a}}(k),\ \mathbf{x}(k-1)^T\hat{\mathbf{a}}(k-1), \ldots,\ \mathbf{x}(k-L_s+1)^T\hat{\mathbf{a}}(k-L_s+1)\ ]^T\end{aligned}$$

where p_i is the polynomial basis of order i, for example $p_1(k) = x(k)$ and $p_2(k) = x^2(k)$ in case of a power series expansion basis. K is the order of the polynomials. The estimated error is

$$\begin{aligned}e(k) &= d(k) - z(k) \\ &= \mathbf{w}^T(k)\mathbf{y}(k) - \mathbf{h}^T(k)\hat{\mathbf{s}}(k)\end{aligned}$$

The gradient of the error power $e^2(k)$, as derived in [15], can be calculated according to:

$$\nabla_h\left[e^2(k)\right] = \frac{\partial e^2(k)}{\partial\hat{\mathbf{h}}(k)} = -2e(k)\hat{\mathbf{s}}(k)$$

$$\nabla_a\left[e^2(k)\right]=\frac{\partial e^2(k)}{\partial \hat{\mathbf{a}}(k)}=-2e(k)\mathbf{P}^T(k)\mathbf{h}(k)$$

$$\nabla_w\left[e^2(k)\right]=\frac{\partial e^2(k)}{\partial w(k)}=2e(k)\mathbf{y}(k)$$

where $\mathbf{P}(k)$ is the expanded nonlinear matrix defined by

$$\mathbf{P}(k)=\begin{bmatrix} p_1(k) & p_2(k) & \cdots & p_K(k) \\ p_1(k-1) & p_2(k-1) & \cdots & p_K(k-1) \\ \vdots & \vdots & \ddots & \vdots \\ p_1(k-L_s+1) & p_2(k-L_s+1) & \cdots & p_K(k-L_s+1) \end{bmatrix}$$

If the coefficients vectors are updated with step size μ_h, μ_a and μ_w, a joint NLMS-type adaptive algorithm is given by

$$\mathbf{h}(k+1)=\mathbf{h}(k)+\frac{\mu_h}{\left\|\hat{\mathbf{s}}(k)\right\|_2^2}\hat{\mathbf{s}}(k)e(k) \tag{4.1}$$

$$\hat{\mathbf{a}}(k+1)=\hat{\mathbf{a}}(k)+\frac{\mu_a}{\left\|\mathbf{P}^{\mathrm{T}}(k)\mathbf{h}(k)\right\|_2^2+\delta}\mathbf{P}^T(k)\mathbf{h}(k)e(k) \tag{4.2}$$

$$\mathbf{w}(k+1)=\mathbf{w}(k)-\frac{\mu_w}{\left\|\mathbf{y}(k)\right\|_2^2}e(k)\mathbf{y}(k) \tag{4.3}$$

In order to avoid trivial solutions, the author constrained that two-norm of polynomial and linear filter are equal to one (i.e. $\left\|\hat{\mathbf{a}}\right\|_2=1$, $\left\|\hat{\mathbf{h}}\right\|_2=1$). Thus, unlike the RLS [14], the NLMS algorithm for channel shortening structure was summarized as

- $\hat{\mathbf{a}}(k+1)=\hat{\mathbf{a}}(k)+\frac{\mu_a}{\left\|\mathbf{P}^{\mathrm{T}}(k)\hat{\mathbf{h}}(k)\right\|_2^2+\delta}\mathbf{P}^T(k)\mathbf{h}(k)e(k)$
- $\hat{\mathbf{a}}(k+1)=\frac{\hat{\mathbf{a}}(k+1)}{\left\|\hat{\mathbf{a}}(k+1)\right\|_2}$
- $\mathbf{h}(k+1)=\mathbf{h}(k)+\frac{\mu_h}{\left\|\hat{\mathbf{s}}(k)\right\|_2^2}\hat{\mathbf{s}}(k)e(k)$
- $\mathbf{h}(k+1)=\frac{\mathbf{h}(k+1)}{\left\|\mathbf{h}(k+1)\right\|_2}$

- $\mathbf{w}(k+1) = \mathbf{w}(k) - \frac{\mu_w}{\|\mathbf{y}(k)\|_2^2} e(k)\mathbf{y}(k)$

We did some simulations to verify that faster converged rate in this structure. The far end signal $x(k)$ was generated according to an i.i.d Gaussian distribution. The room impulse response was generated by a random number generator with an exponential damping factor and we assume the length of room impulse response is equal to 350.

The nonlinear loudspeaker is modeled by polynomial function

$$f(x) = .89x + 0.002x^2 - 0.3x^3 + 0.001x^4 + 0.5x^5$$

The length of shortening filter L_w and linear adaptive L_s are equal to 250 and 100, respectively, and the nonlinear polynomial filter order is $K = 5$.

To evaluate system performance, residual error power, performance measure of echo return loss enhancement (ERLE), and coefficient misalignment are major system performance measures for comparison purposes. With the assumption of high SNR, the (ERLE) can be formulated as

$$ERLE(k) = 10\log_{10}(\frac{[d(k)]^2}{[e(k)]^2})$$

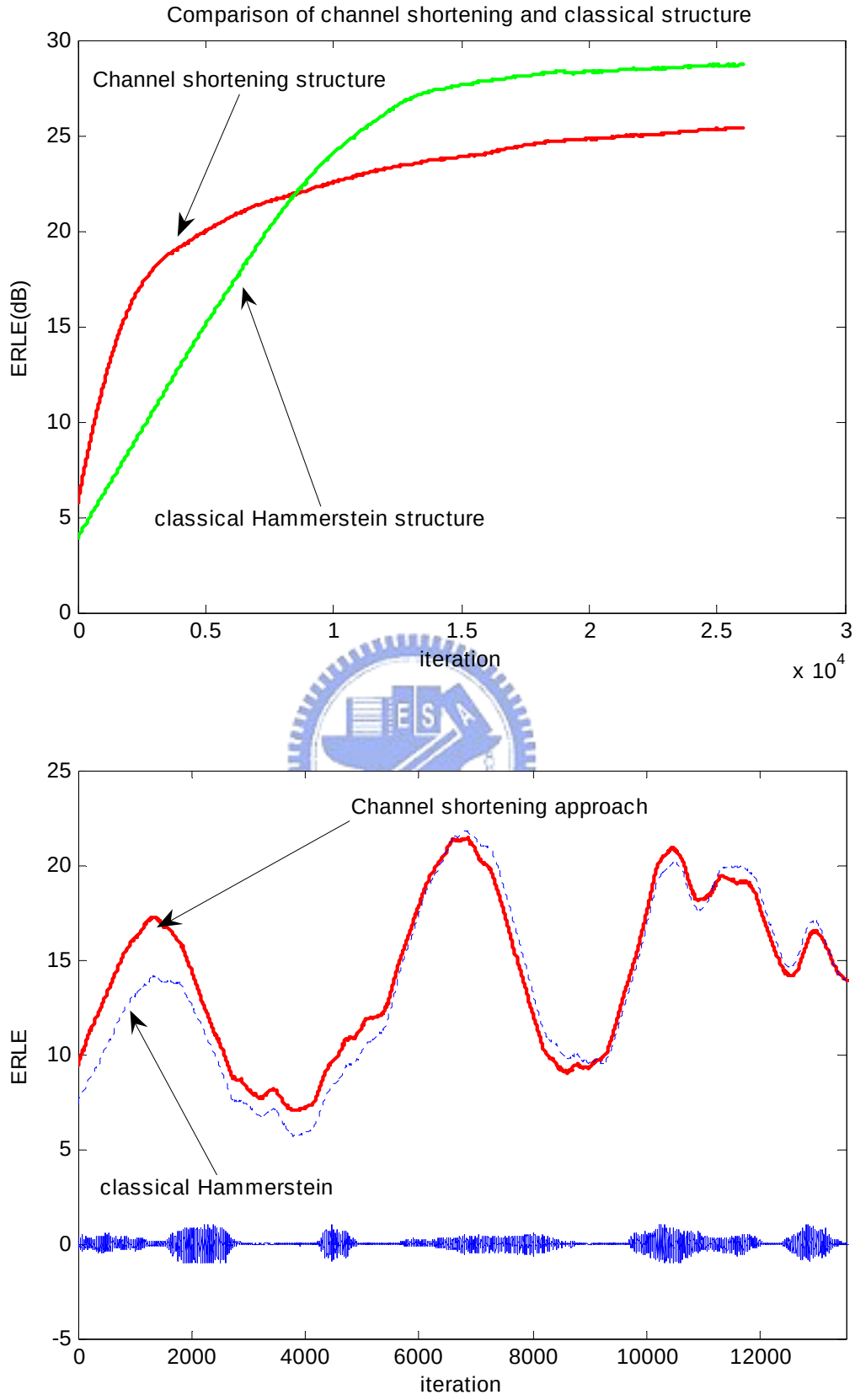


Fig 4.2 Comparison of classical Hammerstein and channel shortening structure

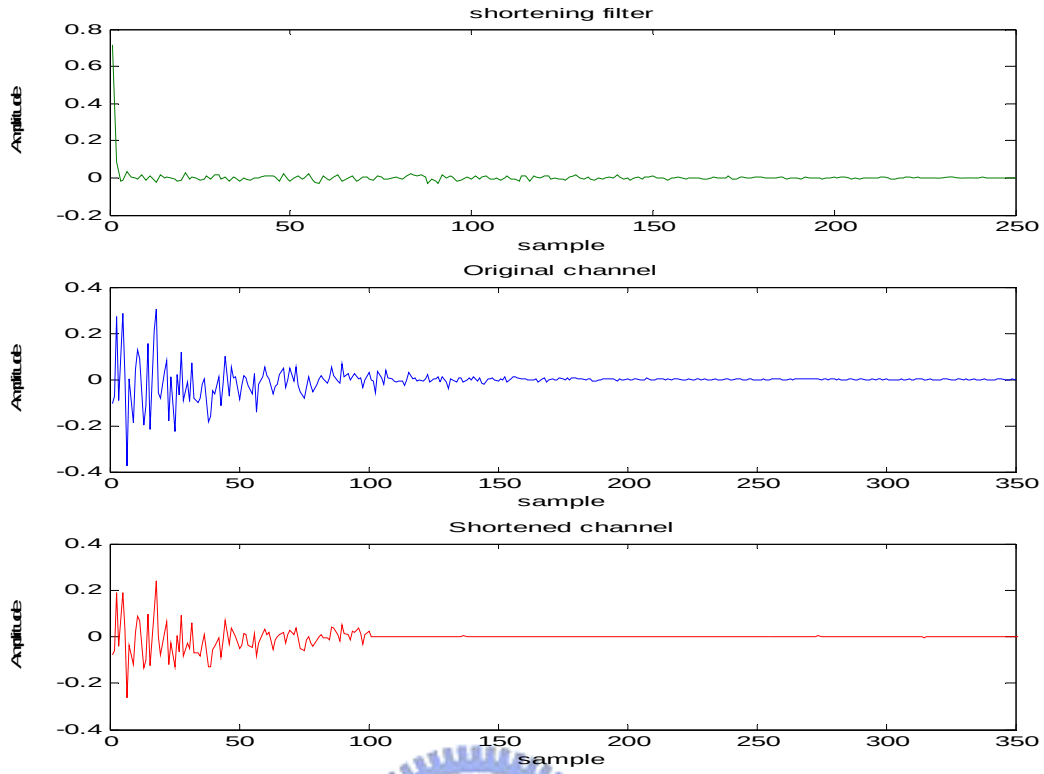


Fig 4.3 Shortening filter, original channel and channel shortened channel

In Fig 4.3, the coefficients of shortening, original and shortened channels were displayed; the shortened channel is the convolution of shortening filter and real original channel, we omit the shortened channel from 351 to 599 taps, which the amplitude value can be neglected.

We can obtain that shortening filter reduces the length of room impulse response from about 250 to 100 taps. As the reduction, the shortening structure reduces length of adaptive filter $\mathbf{h}$ from 350 to 100.

For computational complexity, we examine the number of multiplications required to make once complete iteration of the algorithm (4.1), (4.2) and (4.3). $\hat{\mathbf{s}}(k)$ in (4.1) and its 2-norm need nonlinear order K and linear filter tap M multiplications respectively thus the total requirement multiplication of (4.1) is about $2M + K$. For (4.2), $\mathbf{P}^T(k)\mathbf{h}(k)$ and its 2-norm need MK and K multiplications respectively thus

the total requirement of (4.2) is about $MK+2K$. Thus we know that the total requirement multiplication of classical Hammerstein structure is about $MK+2M+3K$. In channel shortening structure, we reduce the linear FIR filter from M (i.e. $Ls+Lw$) to Ls. The requirement multiplication of (4.3) is about Lw. In order to avoid trivial solutions, the renormalized term in channel shortening approach are added, the requirement multiplication is about $Ls+K$.

	Number of multiplication
Classical Hammerstein structure	$(Ls+Lw)K+2M+3K$
Channel shortening structure	$KLs+2Ls+3K+Lw+Ls+K$

Table 4.1 Computation complexity comparison of classical and shortening structure

By Table 4.1, it can be obtained that the main computation lies in (4.1) and (4.2), thus although it increases multiplication due to shortening filter, it still can reduce the multiplication complexity, because the dominate term of computation complexity of classical Hammerstein structure is MK, which was reduced to KLs.

4.2 Theoretical analysis of linear echo channel

In this section, in order to discuss convergent behavior in the structure, we analyze the coefficient error under the assumption of linear loudspeaker and. By the assumption, we simply the system model in Fig 4.4.

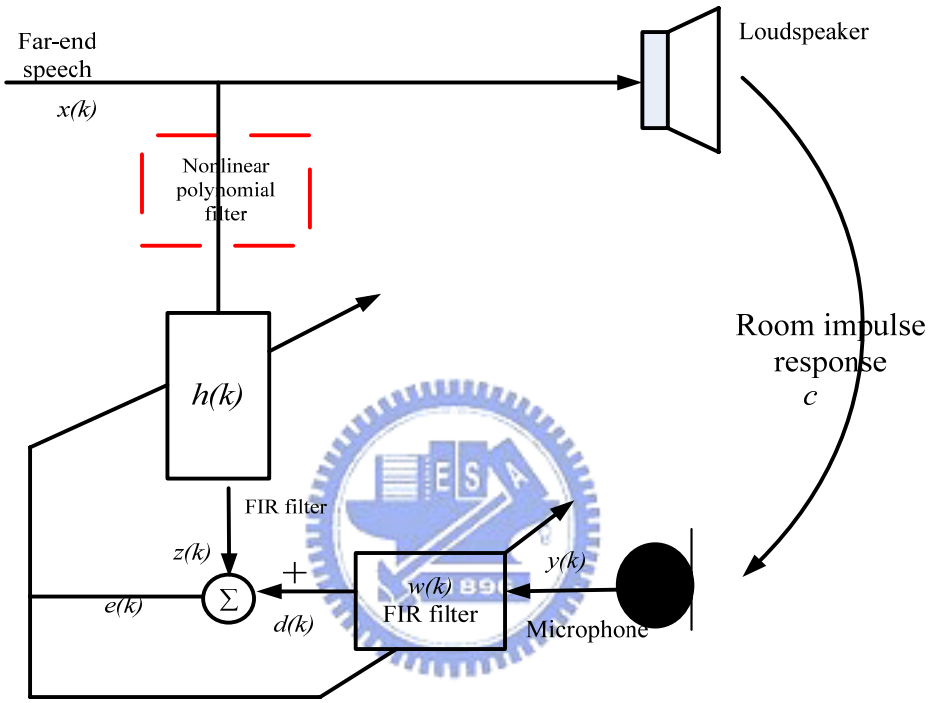


Fig 4.4 Shortening structure for the linear loudspeaker

4.2.1 Least-square solutions

In order to discuss the Least-Square solution in this structure, we don't take account of time index *k*.

In our analysis, we use the residual error $\mathbf{e}$ defined by the following equation.

$$\mathbf{e} = \mathbf{x} * \mathbf{w} * \mathbf{c} - \mathbf{x} * \mathbf{h} + \mathbf{n} * \mathbf{w} \tag{4.4}$$

where $*$ is linear convolution operator , $\boldsymbol{c}$ is room impulse response with finite length M , and the $\mathbf{n}$ denotes the background noise.

$$\mathbf{c} = [c_1 \ldots\ldots c_M]^T$$

$$\mathbf{w} = [w_1 \ldots\ldots w_{L_w}]^T$$

$$\mathbf{h} = [h_1 \ldots\ldots h_{L_s}]^T$$

$$\mathbf{e} = [e_1, e_2, \ldots\ldots]^T$$

c_i represent i'th tap of $\mathbf{c}$, $\mathbf{h}$ is FIR filter with length L_s to track the shortened echo path, h_q represent q'th tap of $\mathbf{h}$, and the $\mathbf{w}$ is shortening filter with length L_w, w_j represent j'th tap of $\mathbf{w}$.

We extend (4.4) to matrix form

$$\begin{aligned}\mathbf{e} &= \mathbf{x} * \mathbf{c} * \mathbf{w} - \mathbf{x} * \mathbf{h} + \mathbf{n} * \mathbf{w} \\ &= (\mathbf{x} * \mathbf{c} + \mathbf{n}) * \mathbf{w} - \mathbf{x} * \mathbf{h} \\ &= (\mathbf{y}) * \mathbf{w} - \mathbf{x} * \mathbf{h} \\ &= \bar{\mathbf{Y}}\mathbf{w} - \bar{\mathbf{X}}\mathbf{h}\end{aligned} \tag{4.5}$$

where $\bar{\mathbf{Y}}$ and $\bar{\mathbf{X}}$ are convolution matrix version of microphone input signal $\mathbf{y}$ and far end signal, respectively, and $L_q = M + L_w - 1$

$$\bar{\mathbf{Y}} = \begin{bmatrix} y_1, 0, \ldots\ldots\ldots 0 \\ y_2, y_1, 0, \ldots\ldots 0 \\ \vdots \quad \ddots \\ 0, \ldots\ldots y_1, \ldots\ldots 0 \\ \vdots \quad \ddots \\ 0 \ldots\ldots\ldots\ldots\ldots y_1 \end{bmatrix}, \quad \bar{\mathbf{X}} = \begin{bmatrix} x_1, 0, \ldots\ldots\ldots 0 \\ x_2, x_1, 0, \ldots\ldots 0 \\ \vdots \quad \ddots \\ 0, \ldots\ldots x_1, \ldots\ldots 0 \\ \vdots \quad \ddots \\ 0 \ldots\ldots\ldots\ldots\ldots x_1 \end{bmatrix}.$$

By cascading up the unknown vectors $\mathbf{w}$ and $\mathbf{h}$, the (4.5) can be written as:

$$\mathbf{e} = \left[\begin{array}{c|c} \bar{\mathbf{Y}} & \bar{\mathbf{X}} \\ & \mathbf{0} \end{array}\right] \begin{bmatrix} \mathbf{w} \\ -\mathbf{h} \end{bmatrix}$$

In order to get the least-square solution of $\mathbf{w}$ and $\mathbf{h}$, we assume that the first element of shortening filter $w_1 = 1$ to separate (4.5) into two term form, Thus (4.5) can be rewritten as:

$$\mathbf{e} = \mathbf{b} + \mathbf{A}\mathbf{s} \tag{4.6}$$

where

$$\mathbf{b}=\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \mathbf{A}=\left[\begin{array}{l|l} 0,\ 0,\ldots\ldots\ldots 0 & x_1,0\ldots\ldots\ldots 0 \\ y_1,\ 0,\ldots\ldots\ldots 0 & x_2,x_1\ldots\ldots\ldots 0 \\ y_2,y_1,\ldots\ldots\ldots 0 & \vdots \quad \ddots \\ \vdots \quad \ddots & \vdots \ \ldots\ldots\ x_1\ldots 0 \\ 0,\ldots y_1,0\ldots\ldots 0 & 0\ldots\ldots\ldots\ldots\ldots \\ 0 \quad \ddots \quad \vdots & 0\ldots\ldots\ldots\ldots 0 \\ \vdots \ \ldots\ldots y_1 \quad \vdots & \vdots \quad \ddots \quad \vdots \\ 0,\ldots\ldots\ldots\ldots & 0\ldots\ldots\ldots\ldots 0 \end{array}\right], \mathbf{s}=\begin{bmatrix} w_2 \\ w_3 \\ \vdots \\ w_{Lw} \\ -h_1 \\ \vdots \\ -h_{Ls} \end{bmatrix}$$

The normal equation of (4.6) can be written as

$$\mathbf{A}^T\mathbf{A}\mathbf{s} = -\mathbf{A}^T\mathbf{b} \tag{4.7}$$

Therefore, in order to minimize the coefficient error, the least-square solution of $\mathbf{x}$ can given by:

$$\mathbf{s} = -(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b} \tag{4.8}$$

4.2.2 Adaptive LMS algorithm and its convergence analysis

In section 4.2.1 , we analyzed the coefficient error in least-square sense , we will derived the theoretical coefficient error based on the adaptive LMS algorithm , we want to observe and discuss the result of LMS algorithm in shortening structure to see if it can achieve the least-square solution or not.

Using (4.5), the coefficient error at time $k+1$.

$$\boldsymbol{\varepsilon}(k+1) = \bar{\mathbf{C}}\mathbf{w}(k+1) - \begin{bmatrix} \mathbf{h}(k+1) \\ \mathbf{0} \end{bmatrix}$$

In the following analysis, we assume the step sizes of shortening filter $\mathbf{w}$ and FIR

filter $\mathbf{h}$ are equal, i.e. $\mu_w = \mu_h = \mu$. Using LMS algorithm, we have

$$\begin{aligned}\boldsymbol{\varepsilon}(k+1) &= \bar{\mathbf{C}}\mathbf{w}(k+1) - \begin{bmatrix}\mathbf{h}(k+1)\\ \mathbf{0}\end{bmatrix} \\ &= \bar{\mathbf{C}}[\mathbf{w}(k) - \mu\mathbf{y}(k)e(k)] - \begin{bmatrix}\mathbf{h}(k) - \mu\mathbf{x}(k)e(k)\\ \mathbf{0}\end{bmatrix} \\ &= \boldsymbol{\varepsilon}(k) - \mu[\bar{\mathbf{C}}\mathbf{y}(k) + \mathbf{x}_{Ls}(k)]e(k) \qquad (4.9)\end{aligned}$$

By the noise-free assumption, the residual error $\mathbf{e}(k)$ can be written as

$$\begin{aligned}\mathbf{e}(k) &= d(k) - z(k) \\ &= \mathbf{y}^T(k)\mathbf{w}(k) - \mathbf{x}^T(k)\mathbf{h}(k) \\ &= \mathbf{x}_{Lq}^T(k)\bar{\mathbf{C}}\mathbf{w}(k) - \mathbf{x}^T(k)\mathbf{h}(k) \\ &= \mathbf{x}_{Lq}^T(k)\boldsymbol{\varepsilon}(k) \qquad (4.10)\end{aligned}$$

where

$$L_q = M + L_w - 1$$

$$\mathbf{x}_{Ls}^T(k) = \underbrace{[x(k), x(k-1), \ldots, x(k-L_s+1), 0, \ldots 0]}_{1\times L_q}$$

$$\mathbf{x}_{Lq}^T(k) = [x(k), x(k-1), \ldots.., x(k-L_q+1)]$$

Using (4.9), (4.10), we rewrite the coefficient error as

$$\begin{aligned}\boldsymbol{\varepsilon}(k+1) &= \boldsymbol{\varepsilon}(k) - \mu[\bar{\mathbf{C}}\mathbf{y}(k) + \mathbf{x}_{Ls}(k)]\mathbf{x}_{Lq}^T(k)\boldsymbol{\varepsilon}(k) \\ &= \left[\mathbf{I}_{L_q\times L_q} - \mu[\bar{\mathbf{C}}\mathbf{y}(n)\mathbf{x}_{Lq}^T(k) + \mathbf{x}_{Ls}(k)\mathbf{x}_{Lq}^T(k)]\right]\boldsymbol{\varepsilon}(k) \qquad (4.11)\end{aligned}$$

According to the direct averaging method [13], when μ is very small, the coefficient error $\boldsymbol{\varepsilon}(k+1)$ can be approximated as follows:

$$\boldsymbol{\varepsilon}(k+1) \cong \left[\mathbf{I}_{L_q\times L_q} - \mu\sigma_x^2\mathbf{R}_q\right]\boldsymbol{\varepsilon}(k) \qquad (4.12)$$

where $\mathbf{R}_q = \bar{\mathbf{C}}\mathbf{H}_q + \mathbf{I}_{L_s \times L_s}$, $\mathbf{H}_q = \begin{bmatrix} c_1, c_2, \ldots\ldots & c_{tap} & , & 0 \ldots\ldots 0 \\ 0, c_1, \ldots\ldots & c_{tap-1} & , & 0 \ldots\ldots 0 \\ \vdots \quad \ddots & & & \vdots \quad \ddots \quad \vdots \\ 0 \ldots\ldots c_1, \ldots & c_{tap-L_w+1} & , & 0 \ldots\ldots 0 \end{bmatrix}$

By applying the similarity transformation, $\mathbf{R}_q$ is transformed into a simpler form:

$$\mathbf{S}^{-1}\mathbf{R}_q\mathbf{S} = \mathbf{E}$$

where $\mathbf{S}$ is a matrix consisting of eigenvectors of $\mathbf{R}_q$ and $\mathbf{E}$ is a diagonal matrix consisting of the eigen-values λ_i. Let $\boldsymbol{\varphi}(k) = \mathbf{S}^{-1}\boldsymbol{\varepsilon}(k)$ then we may transform (4.12) into the form

$$\mathbf{S}^{-1}\boldsymbol{\varepsilon}(k+1) = \mathbf{S}^{-1}\left[\mathbf{I} - \mu\sigma_x^2\mathbf{R}_q\right]\boldsymbol{\varepsilon}(k)$$

$$\boldsymbol{\varphi}(k+1) = \left[\mathbf{I} - \mu\sigma_x^2\mathbf{E}\right]\boldsymbol{\varphi}(k)$$

The natural mode $\varphi_i(k)$ denotes i'th entry of $\boldsymbol{\varphi}(k)$. Let $\varphi_i(0)$ denote the initial value of $\varphi_i(k)$. We may rewrite $\varphi_i(k)$ as follows.

$$\begin{aligned} \varphi_i(k) &= (1 - \mu\sigma_x^2\lambda_i)\varphi_i(k-1) \\ &= (1 - \mu\sigma_x^2\lambda_i)^k \varphi_i(0) \end{aligned} \tag{4.13}$$

Hence, we get the theoretical coefficient error in LMS algorithm sense. In chapter 5, we would simulate it and compare with the least-square solution.

We will note that the theoretical mean-square error is

$$J(k) = E\left\{e^2(k)\right\} = \sigma_x^2 \left\|\varepsilon(k)\right\|_2^2 \tag{4.14}$$

4.2.3 Non-unique converged value

To further understand the converged behavior, we discuss the converged value of

adaptive filter $\mathbf{h}(k)$ and shortening filter $\mathbf{w}(k)$, we want to discuss the convergence toward the optimal parameters in the system. Indeed, the mean-square error $J(k) = E\{e^2(k)\}$ produces some local minima, implying the convergence toward incorrect parameters depending on the initialization of $\mathbf{h}(k)$ and $\mathbf{w}(k)$. We study this phenomenon in this section. From (4.10), the square of residual error $e^2(k)$ can be expressed as:

$$e^2(k) = \mathbf{w}^T(k)\mathbf{y}(k)\mathbf{y}^T(k)\mathbf{w}(k) + \mathbf{h}^T(k)\mathbf{x}(k)\mathbf{x}^T(k)\mathbf{h}(k) - 2\mathbf{w}^T(k)\mathbf{y}(k)\mathbf{x}^T(k)\mathbf{h}(k) \tag{4.15}$$

The squared error can be used to get the optimal parameter set in the minimum mean-square error sense (MMSE). To simplify analysis, we assume that the other filter is quasi-constant when we analyses one filter.

For the linear AEC filter $\mathbf{h}(k)$, the gradient of $J(k)$ respect to $\mathbf{h}(k)$ is given by

$$\nabla_{\mathbf{h}}\{J(k)\} = 2E\left[\mathbf{x}(k)\mathbf{x}^T(k)\right]\mathbf{h}(k) - 2E\left[\mathbf{w}^T(k)\mathbf{y}(k)\mathbf{x}(k)\right] \tag{4.16}$$

For the shortening filter $\mathbf{w}(k)$, the gradient of $J(k)$ respect to $\mathbf{w}(k)$ is given by

$$\nabla_{\mathbf{w}}\{J(k)\} = 2E\left[\mathbf{y}(k)\mathbf{y}^T(k)\right]\mathbf{w}(k) - 2E\left[\mathbf{h}^T(k)\mathbf{x}(k)\mathbf{y}(k)\right] \tag{4.17}$$

$\mathbf{h}_{MMSE}$ and $\mathbf{w}_{MMSE}$, both satisfy the equality $\nabla_{\hat{\mathbf{h}}}\{J(k)\}\Big|_{\hat{\mathbf{h}}=\hat{\mathbf{h}}_{MMSE}} = 0$ and $\nabla_{\mathbf{w}}\{J(k)\}\Big|_{\mathbf{w}=\mathbf{w}_{MMSE}} = 0$, respectively. Thus from (4.13) and (4.14), we can obtain that $\mathbf{w}_{MMSE}$ is function of $\mathbf{h}(k)$, and $\mathbf{h}_{MMSE}$ is a function of $\mathbf{w}(k)$.

Thus we can see that initial value of $\mathbf{h}(k)$ and $\mathbf{w}(k)$ will affect the final converged value.

4.3 Multiple stage update in channel shortening structure

As seen (4.16) and (4.17), in linear echo path channel, we can point out several things. First, the different initial value of filter (whatever AEC or shortening filter) will cause the shortening structure converge to different value; second, the system is not able to identify the optimal set of parameters, unless one of filter is know.

It has identical situations in nonlinear channel shortening structure (i.e. Fig 4.1). In this section, we will try to change the way of updating in channel shortening structure, rather than joint update.

The Guerin [3] proposed the two-staged strategy which it starts with one filter, and joint adaption of all filters (i.e. polynomial filter $\hat{\mathbf{a}}(k)$, adaptive AEC filter $\mathbf{h}(k)$, and shortening filter $\mathbf{w}(k)$) once the linear filter has sufficiently converged in first stage.

In channel shortening structure, the two-staged strategy means change the initial value of filters, as when the first stage finished, the initial value is identical to the converged value in first stage.

By the idea, we want to obtain and compare that the performance of different multiple update strategies (i.e. different initial value of filters). The overall of discussion will be verified in Section 5.3.

4.4 Volterra with channel shortening and OTTLMS

In this section, we try to implement the shortening structure to second order Volterra, we hope to the shortening filter shorten the linear kernel of echo path and lead to improve the convergence rate in Volterra structure.

It is identical idea with Hammerstein, the purpose of shortening filter $w(k)$ is to "shorten" the linear kernel, which is the convolution of the room impulse response and $w(k)$ to have shorter taps.

In addition to channel approach for Volterra structure, we implement the OTTLMS algorithm to the combination (i.e. channel shortening in Volterra filter), The overall of discussion will be verified in Section 5.3.

Chapter 5
Computer Simulation

To evaluate the performance of our proposed nonlinear AEC algorithm, we provide computer simulations. In Section 5.1, we introduce the parameters of our simulation. A series of simulations and experiments on the optimum time-&tap-variant step size, will be compared and discussed in Section 5.2. In Section 5.3 we will compare simulations results with theoretical analyses in channel shortening structure.

5.1 Simulation parameters introduction

The signal to noise ratio at microphone is defined as

$$SNR = 10\log_{10}(\frac{P_{echo}}{P_{noise}})$$

where P_{echo} is power of the nonlinear echo and the P_{noise} is the power of the background noise.

For simplicity, we use a 256-tap room impulse response as shown in Fig 5.1.1. It

is generated by a random number generator with an exponential damping factor.

Nonlinear memory echo path is shown in Fig 5.1.2. In this thesis, we use a 20-memory kernel.

In our experiment, we not only use an i.i.d white Gaussian signal, but also speech signal as the input signal to examine the performance. The speech signal is sampled with 8 KHz sampling rate shown in Fig 5.1.3.

To evaluate system performance, residual error power, ERLE, and coefficient misalignment are major system performances for comparison purposes. The performance measure of echo return loss enhancement (ERLE) can be formulated in single talk condition as:

$$ERLE(dB) = 10\log_{10}\left\{\frac{y(k)^2}{e(k)^2}\right\}$$

where the $y(k)$ is the microphone signal

In section 5.3, the nonlinear loudspeaker in our simulation is modeled as the polynomial function:

$$f(x) = .89x + 0.002x^2 - 0.3x^3 + 0.001x^4 + 0.5x^5$$

In [14], Usually ERLE is usually defined as the ratio of microphone received echo power to the residual echo power. In Fig4.1, since shortening filter may change the power of receive echo, the ERLE is redefined as

$$ERLE(dB) = 10\log_{10}\left\{\frac{d(k)^2}{e(k)^2}\right\}$$

where $d(k)$ and $e(k)$ represent the "filtered" received echo signal and residual echo, respectively.

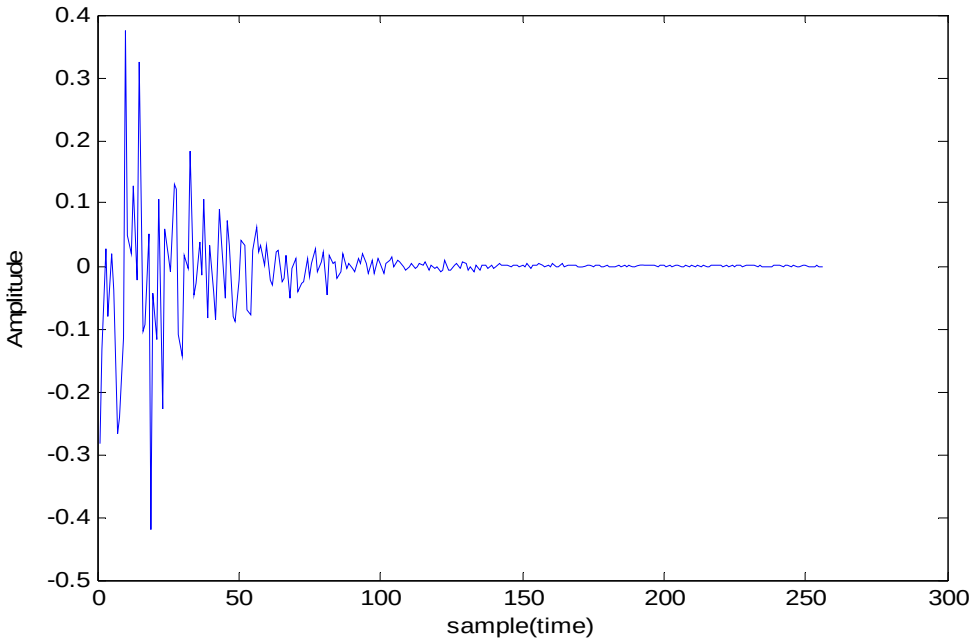


Figure 5.1.1 Room impulse response

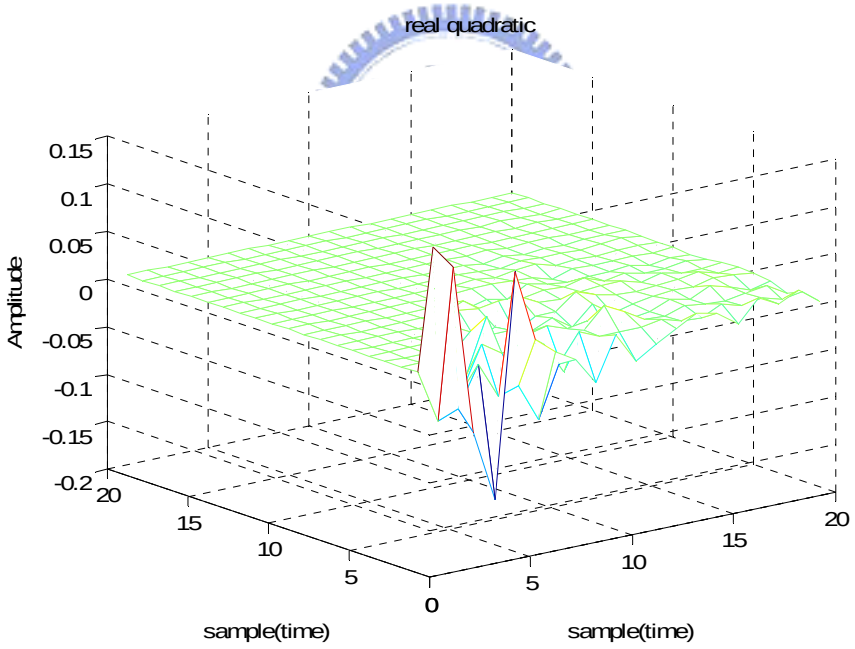


Figure 5.1.2 Quadratic kernel

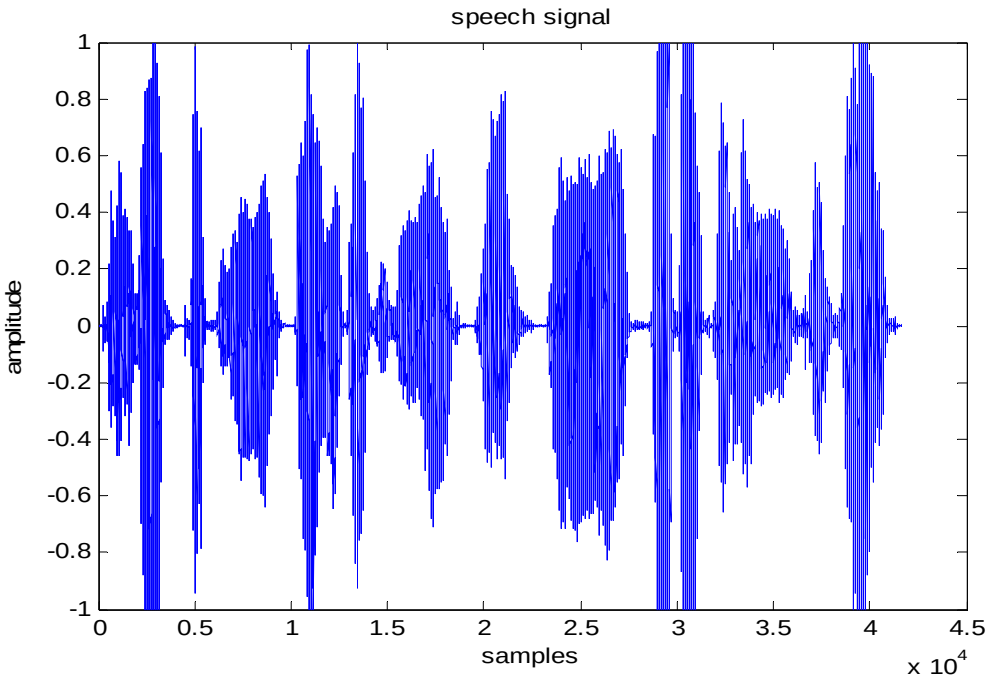


Fig 5.1.3 Speech signal

5.2 ERLE convergent rate comparison

5.2.1 Comparison of OTTLMS, only linear OTTLMS and LMS

Figure 5.2.1 shows the convergent rate curves of LMS and OTTLMS algorithms. The parameters settings chosen for Figure 5.2.1 are that the order of linear and quadratic kernel equal to 256 and 20 respectively, and the signal-to-noise ratio is 30dB. We simulated large and small step size in LMS algorithm, We can see that large step size provides fast convergence rate but low ERLE performance, and vice versa. At initial state, LMS algorithm has the slowest convergence rate compared to OTTLMS with variant step-size algorithms.

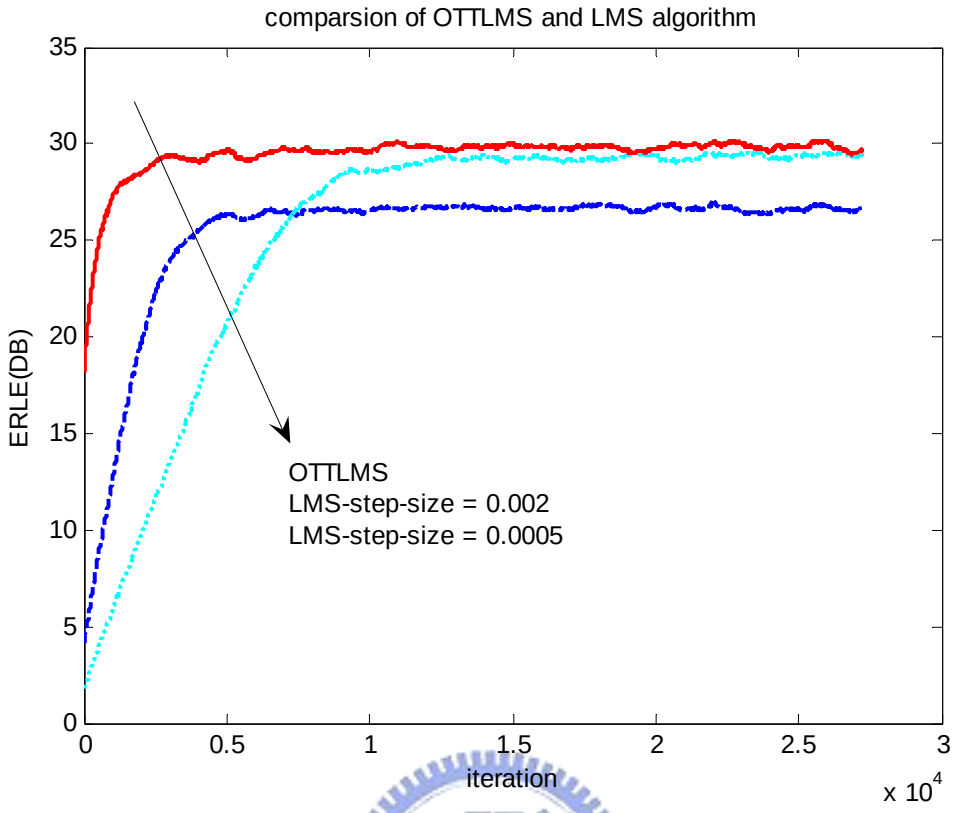


Fig 5.2.1 Comparison of OTTLMS and LMS algorithms (with white Gaussian input)

From result of Fig 5.2.1, in small step size that has identical convergent ERLE value to OTTLMS, we can obtain that our OTTLMS algorithm provides faster convergent rate, our approach converged after 3000 iterations which is faster than 10000 iterations in LMS algorithm. Besides compare with small step size, we can see that even if we use large step size in LMS algorithm, the OTTLMS algorithm still has a faster convergence rate than LMS algorithm.

In fig 5.2.2, we use the real speech signal in Fig 5.1.3, the parameters settings chosen for Figure 5.2.2 are as follows:

- $M = 256$ N_2=20 SNR=25 dB
- LMS : $u = 0.01$

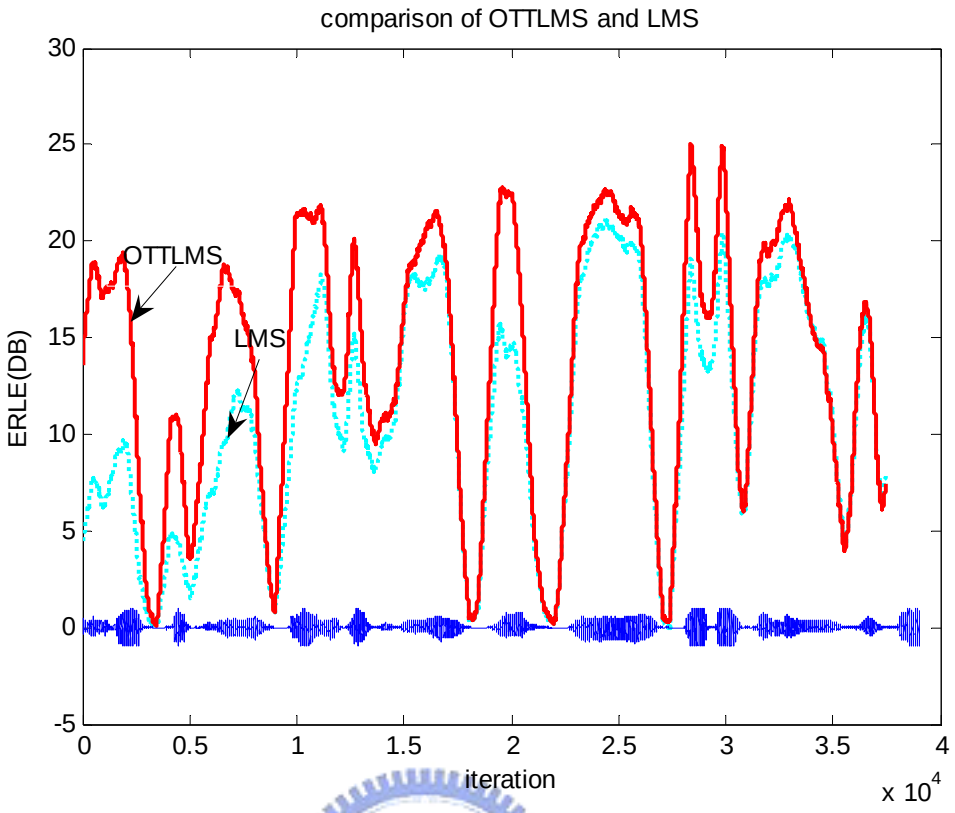


Fig 5.2.2 Comparison of OTTLMS and LMS algorithm (with real speech)

Fig 5.2.2 show that the optimum time-& tap-variant step-size LMS (OTTLMS) provides faster convergence rate than LMS algorithm in real speech input. We can obtain that our OTTLMS algorithm enhance the LMS algorithm about 10dB in initial state, after 15000 iterations, the OTTLMS still improve LMS algorithm about 5dB.

After the comparison of OTTLMS and LMS algorithm, we want to look the effect of optimal step size in quadratic kernel, we compare OTTLMS algorithm into both linear and quadratic kernels with only into linear kernel.

In this thesis, the word “only-linear OTTLMS” represents that the step sizes of linear adaptive filter are optimal, and the quadratic kernel is fixed (i.e. LMS algorithm for quadratic kernel).

Fig 5.2.3 shows the convergence rate curve of OTTLMS and only-linear-OTTLMS algorithm. The parameters settings chosen for Figure 5.2.3 are the same as Fig 5.2.1.

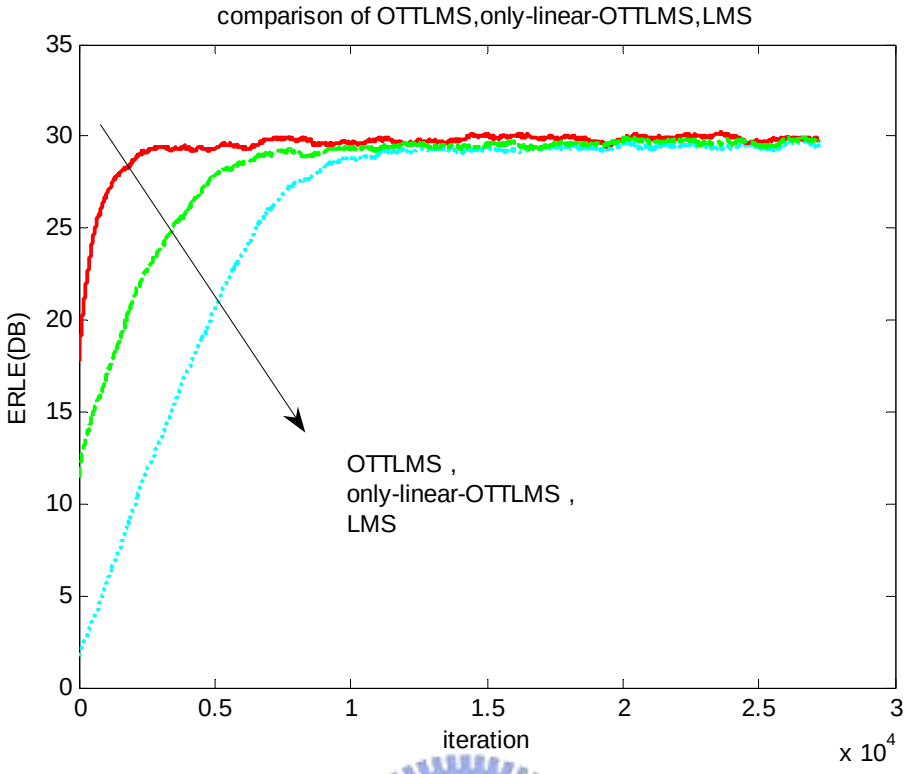


Fig 5.2.3 comparison of OTTLMS, only-linear-LMS, and LMS algorithm (with white Gaussian input)

From result of Fig 5.2.3, we find the convergent rate of OTTLMS for all kernels is higher than only-linear OTTLMS, we can obtain that the OTTLMS for all kernels converged after about 3000 iterations which is faster than 5000 iterations in only –linear OTTLMS.

In addition to Gaussian input signal, we also simulated in real speech signal to compare OTTLMS with only-linear-OTTLMS algorithm. The parameters settings chosen for Figure 5.2.4 are the same as Fig 5.2.2.

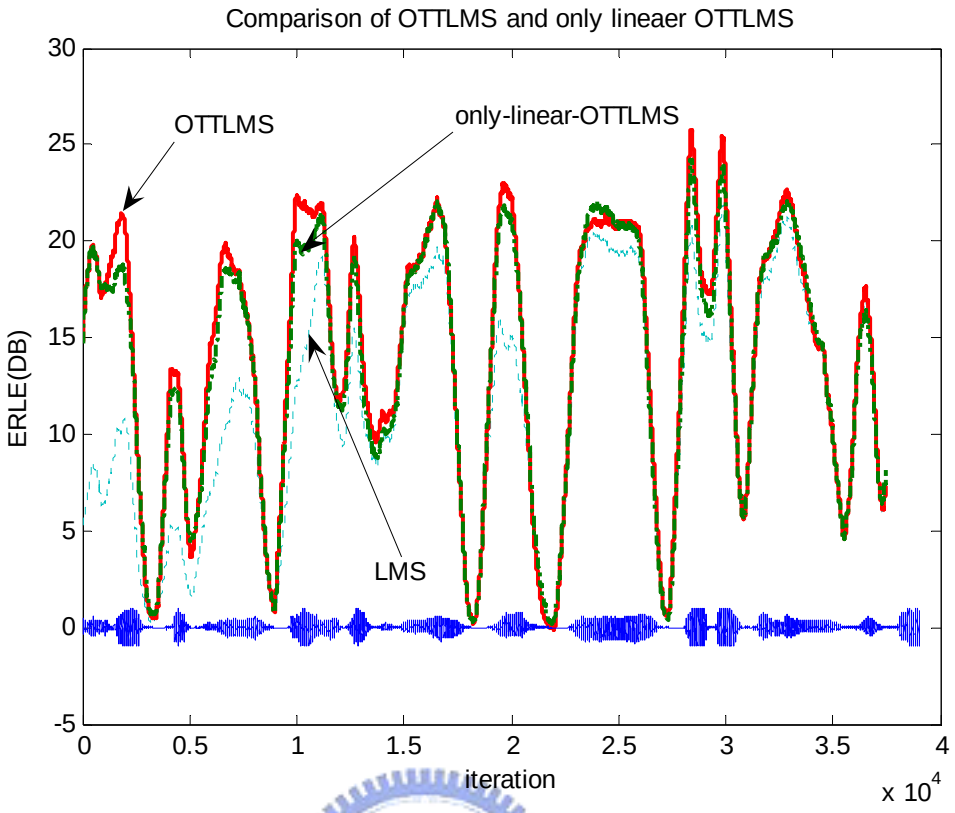


Fig 5.2.4 comparison of OTTLMS and only linear OTTLMS (with real speech)

Fig 5.2.4 shows that the OTTLMS algorithm provides faster convergence rate than only-linear OTTLMS algorithm in real speech input. We can obtain that our OTTLMS algorithm enhance the only-linear-OTTLMS about 4dB in ERLE performance.

5.2.2 Comparison of OTTLMS and Different parameters of model function

In this section, we will simulate the ERLE performance using different modeling parameters of model function by (3.14), (3.15), we named “matched model” if model function close to the real kernel; “under model”, if decay rate of model function faster than real kernel; “over model” if decay rate of model function is slower than real kernel.

We will discuss the case of mismatched model. By Fig5.1.1 and Fig5.1.3, we use three slope parameters of model function in (3.14), (3.15), by assumption of the slopes of real linear $r^{(1)}$ and quadratic kernel $r^{(2)}$ are approximately equal to 0.96 and 0.85.

The parameters settings chosen for Figure 5.2.5 and Figure 5.2.6 are as follows:

- under model : $r_{under}^{(1)} = 0.93$, $r_{under}^{(2)} = 0.80$
- upper model : $r_{over}^{(1)} = 0.99$, $r_{over}^{(2)} = 0.90$

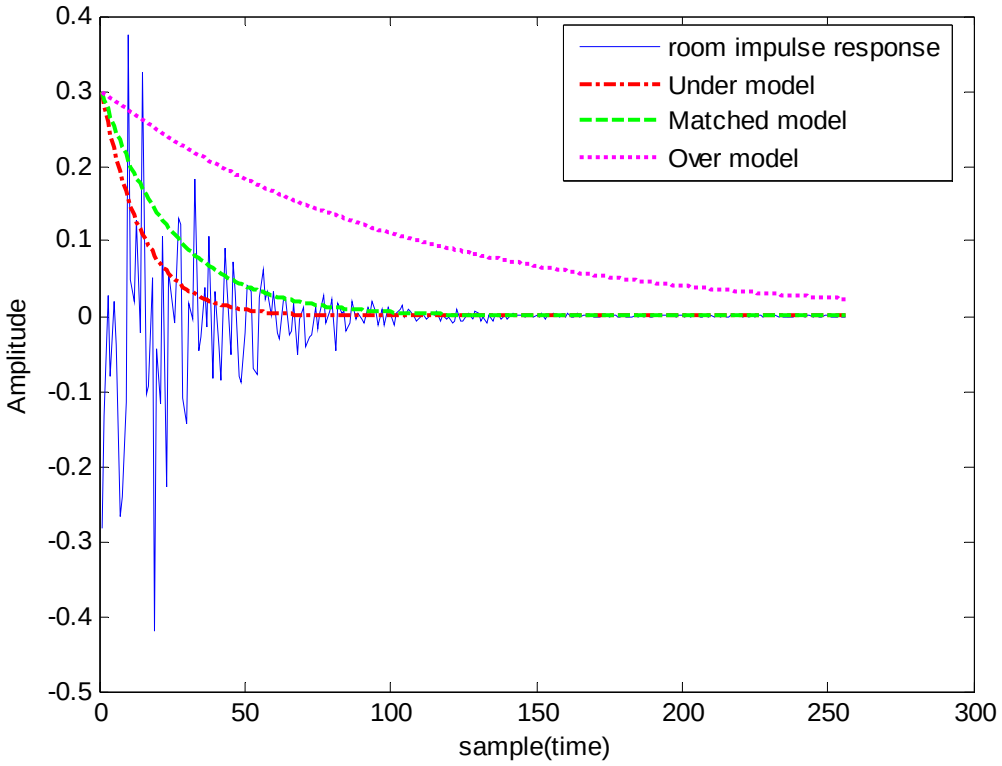


Fig5.2.5 RIR and Model function

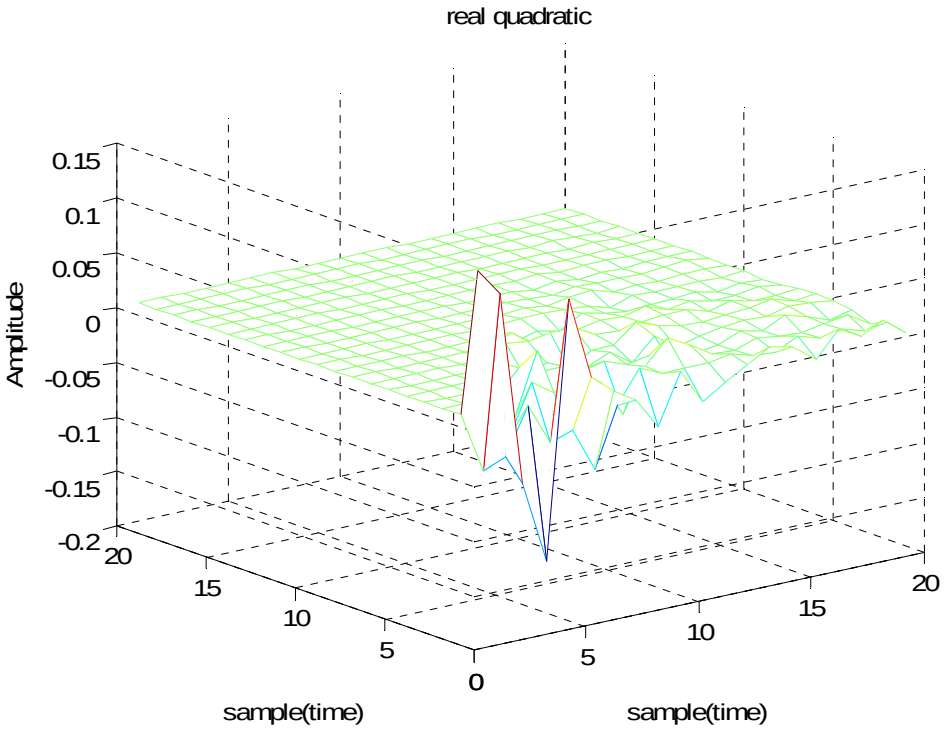


Fig 5.2.6 (a) Quadratic kernel

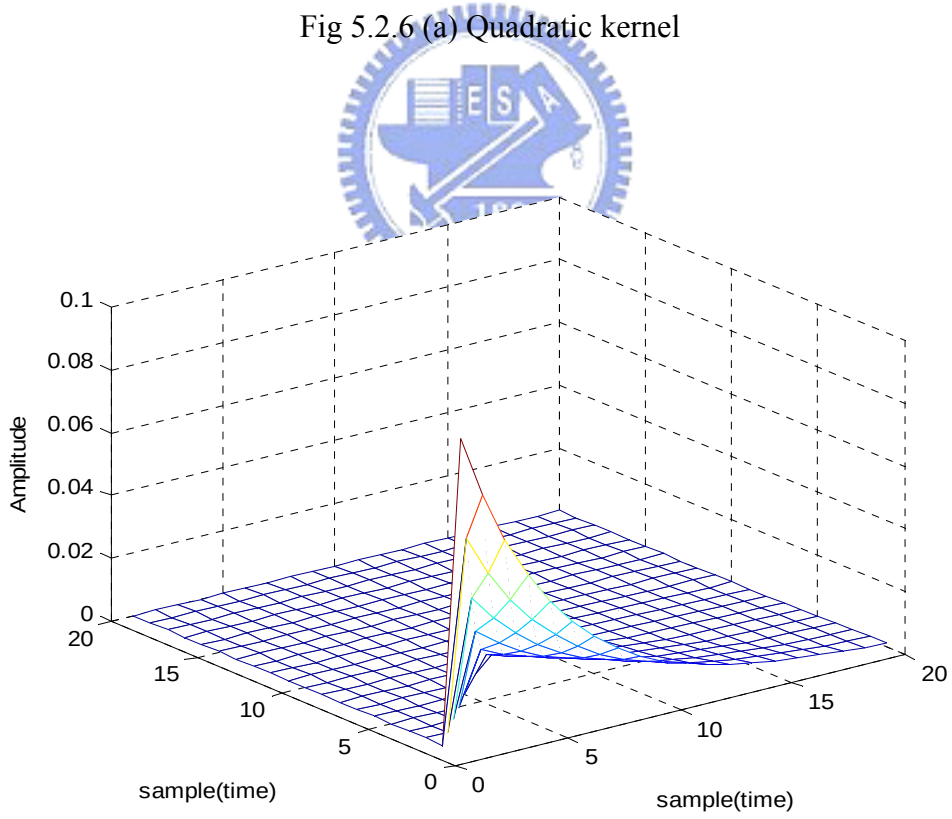


Fig 5.2.6 (b) Under Model

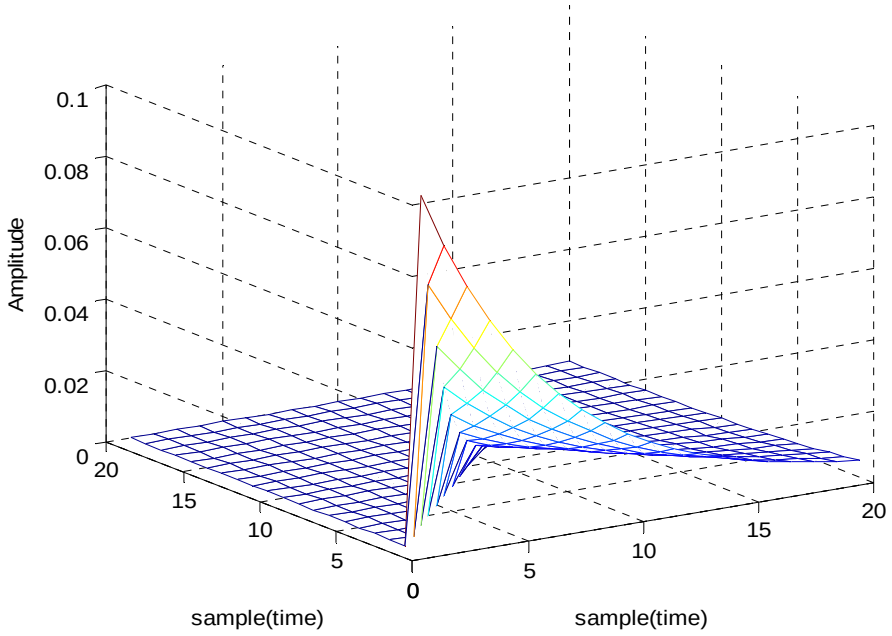


Fig 5.2.6 (c) Matched Model

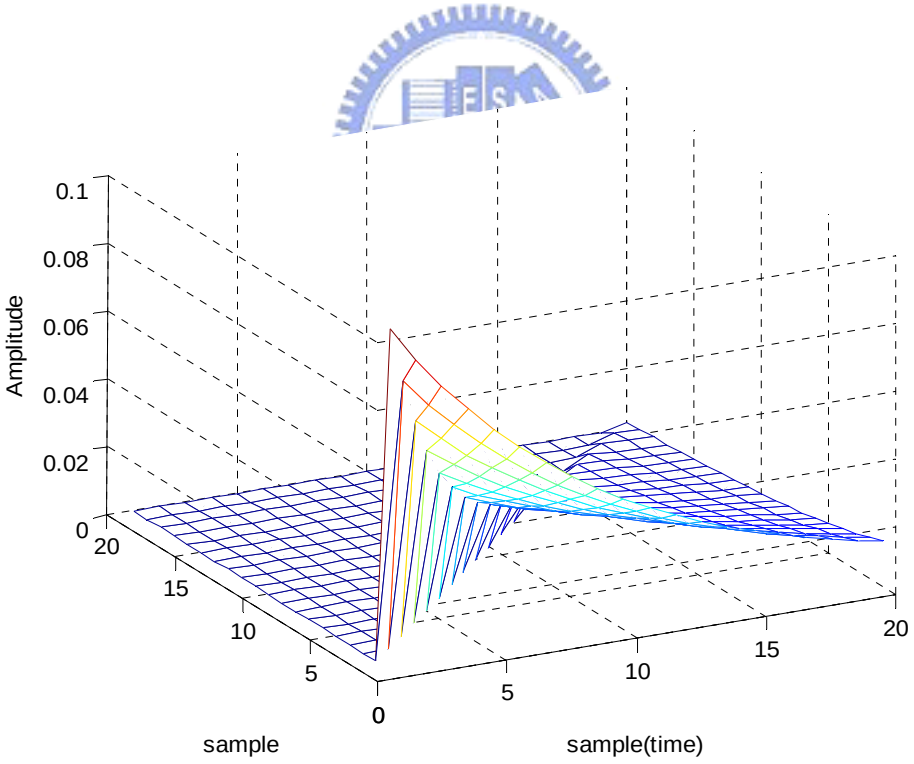


Fig 5.2.6 (d) Over Model

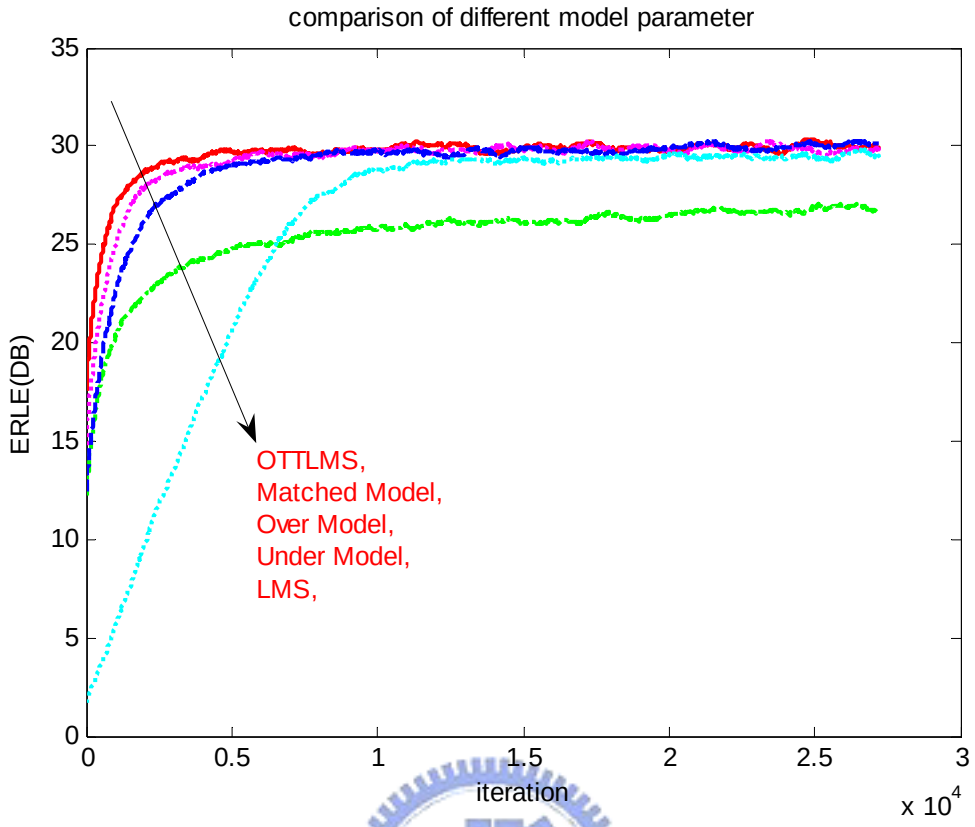


Fig 5.2.7 Comparison of mismatched model functions (with white Gaussian input)

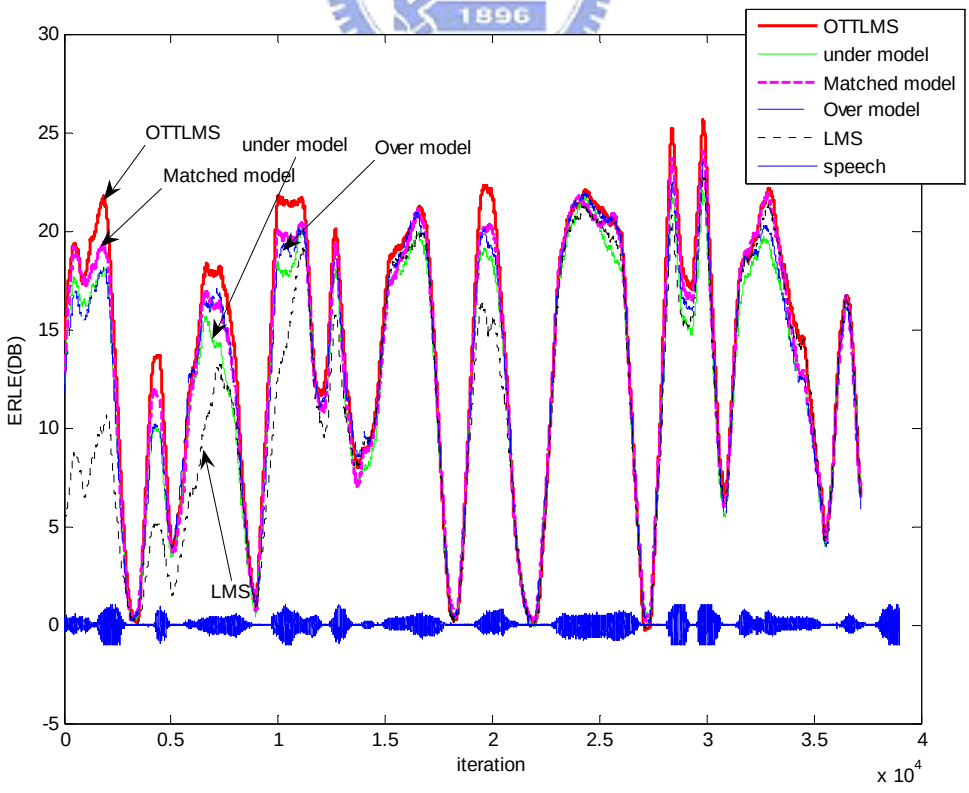


Fig 5.2.8 Comparison of Mismatched model functions (with real speech)

From result of Fig 5.2.7 and Fig 5.2.8, we obtain that ERLE performance of closed and Over model are similar to optimum model, even if we use the Under model, the converged rate still faster than LMS algorithm.

5.2.3 Comparison of OTTLMS and OTLMS

In this section, we want to examine the effect of tap-variant step size. We will compare tap-variable step size with tap-invariable. Thus we fixed the

The parameters settings chosen for OTLMS are as follows:

- OTLMS : $r_{OTLMS}^{(1)} = 1$, $r_{OTLMS}^{(2)} = 1$

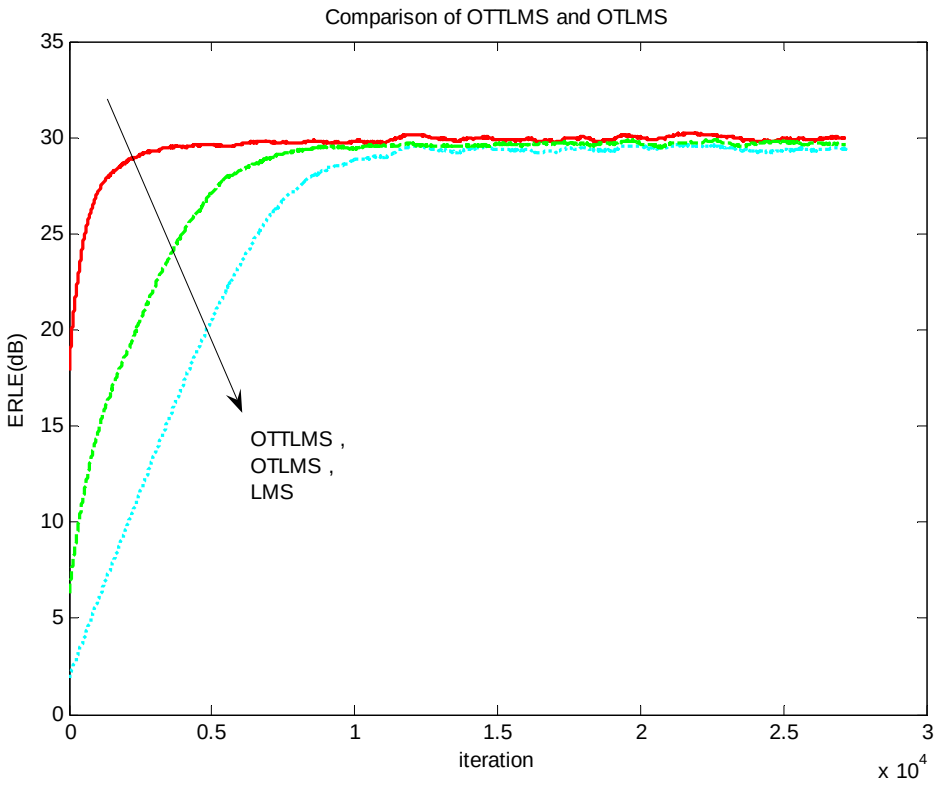


Fig 5.2.9 Comparison of OTTLMS and OTLMS (with Gaussian input)

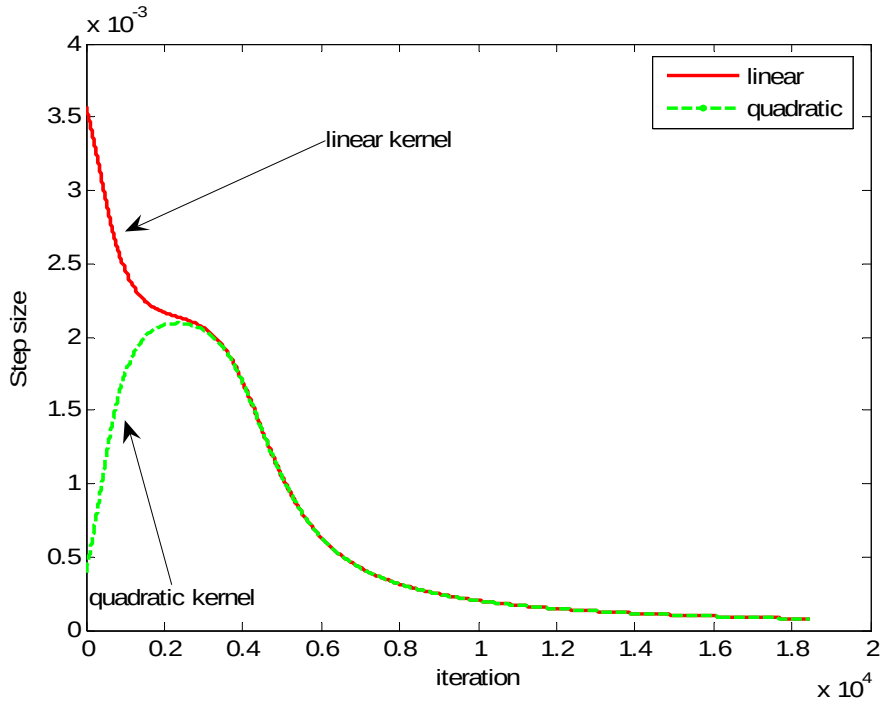


Fig 5.2.10 Step size of practical OTLMS (with white Gauss input)

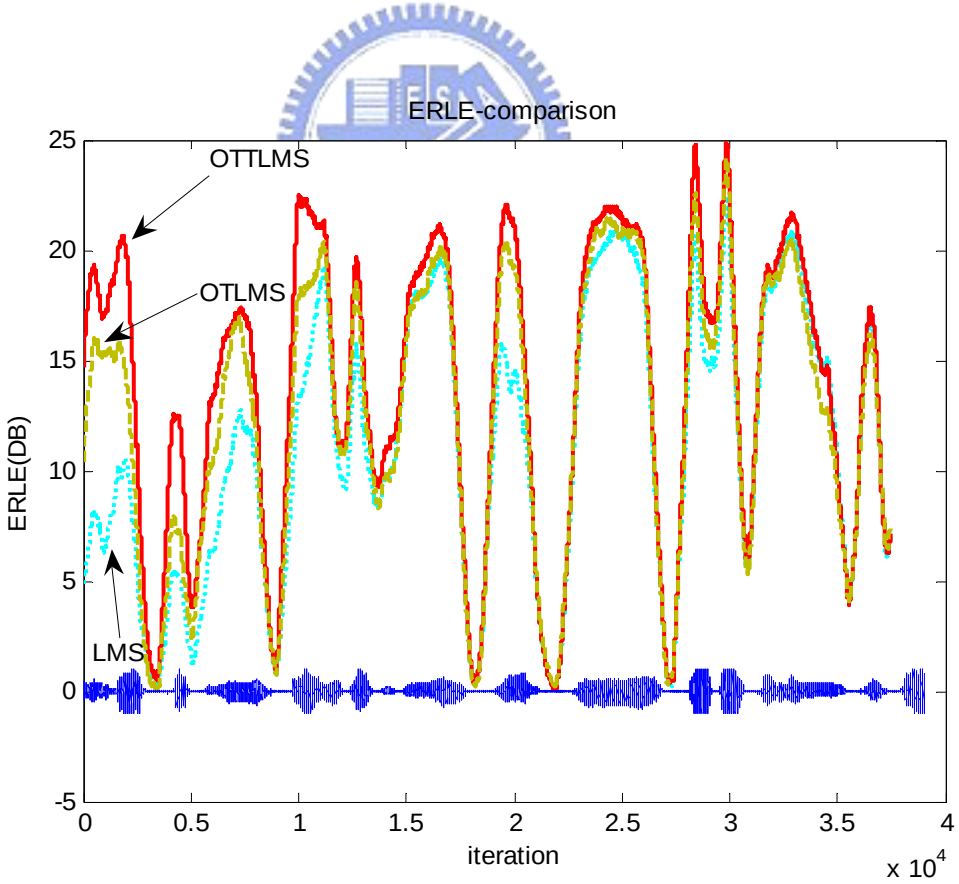


Fig 5.2.11 Comparison of OTTLMS and OTLMS (with real speech)

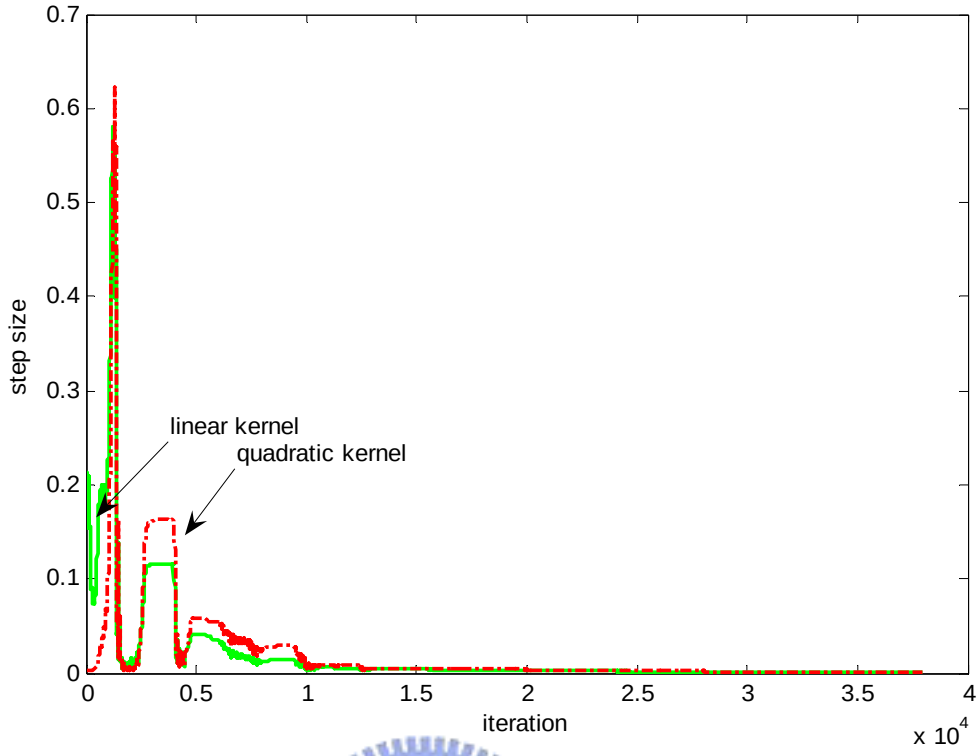


Fig 5.2.12 Step size of practical OTLMS (with real speech)

Fig 5.2.9 and Fig 5.2.11, show in either case of white Gaussian or speech signal, OTTLMS performs better than OTLMS, it also demonstrate that the time and tap-variable step size provides faster convergence rate. Fig 5.2.10 and Fig5.2.12 are reasonable since the value of step size is large at initial state (i.e. before 5000 iterations), and small if it converged.

5.2.4 Exponentially approximated temporal function and LMS

From Fig 3.5, 3.6, we know that the step size of OTTLMS algorithm is exponential decay by time, thus we derived the (3.21), (3.22) and compared the temporal curve with exponentially approximated function. In this section we simulated the ERLE performance to compare them.

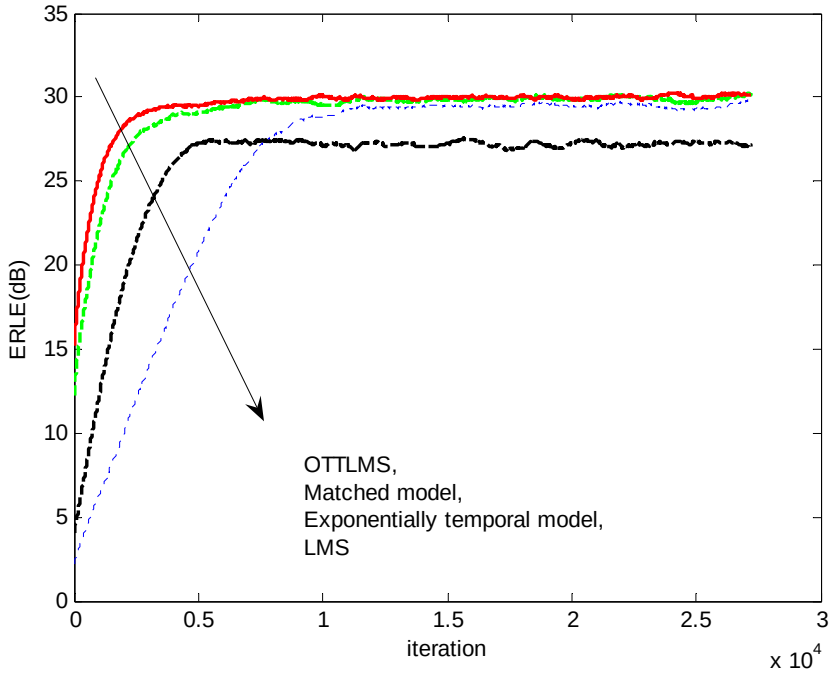


Fig 5.2.13 Comparison of OTTLMS and Exponentially approximation function (with white Gaussian input)

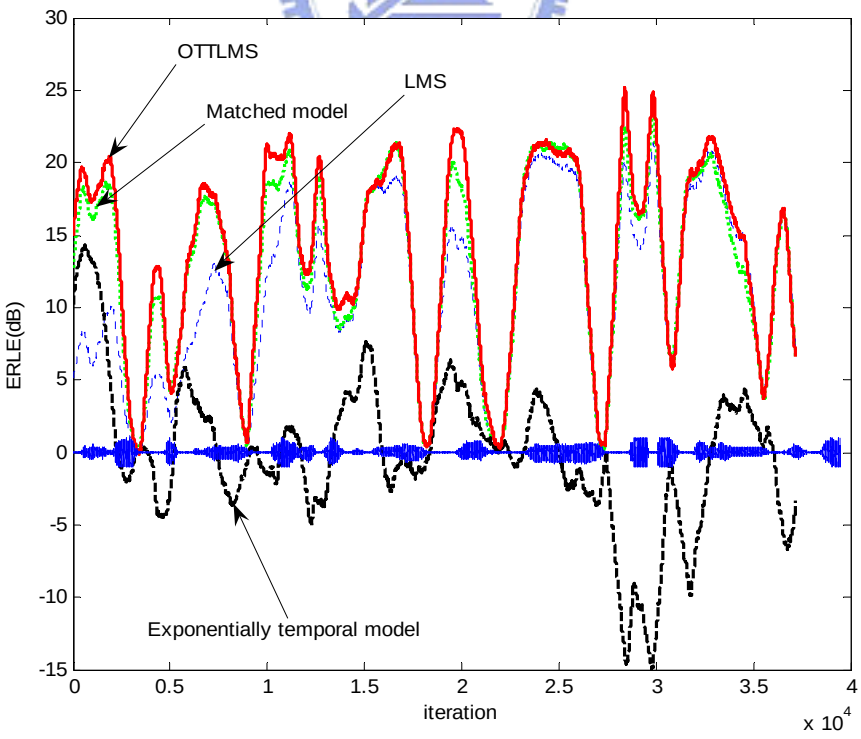


Fig 5.2.14 Comparison of OTTLMS and Exponentially approximation function (with real speech signal)

From Fig 5.2.13 and Fig 5.2.14, we can obtain that the exponentially temporal model still accelerate convergent rate of LMS algorithm in white Gaussian input, however, when the far end input is real speech, the EAOTTLMS(exponentially approximated temporal of OTTLMS) is failed.

We obtain further the temporal function of step size, in Fig 5.2.15, we can obtain that step size of linear and quadratic are too large when the speech volume is small, this out of step size control behavior cause that the EAOTTLMS failed.

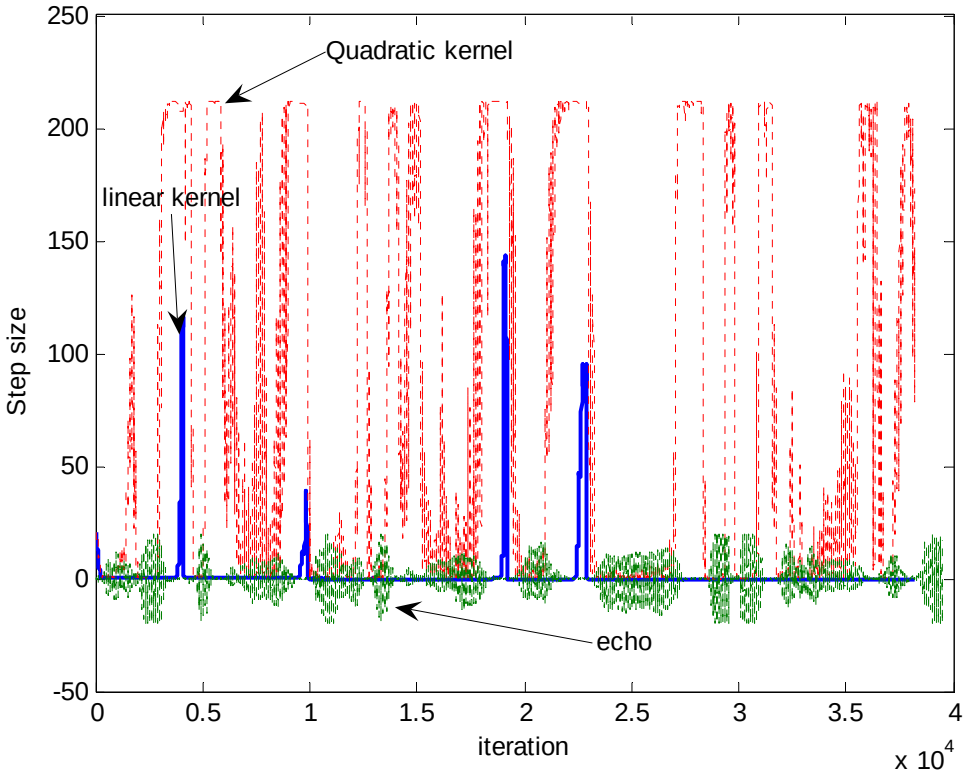


Fig 5.2.15 Step size of EAOTTLMS

5.2.5 Comparison of OTTNLMS and Kuech's approach

In section 2, we introduce Kuech approach [14]. OTTNLMS algorithm and practical implement were proposed in section 3.2 and section 3.3 respectively .This section we want to make a comparison of them, to compare fairly, we choose some parameter which make the steady state of Kuech's ERLE equal to our proposed. The parameters settings chosen for Figure 5.2.16 are same as Fig 5.2.2, the step sizes of NLMS are fixed to 0.2, and the parameter of Kuech [14] (i.e. μ_{bn}) is equal to 0.2.

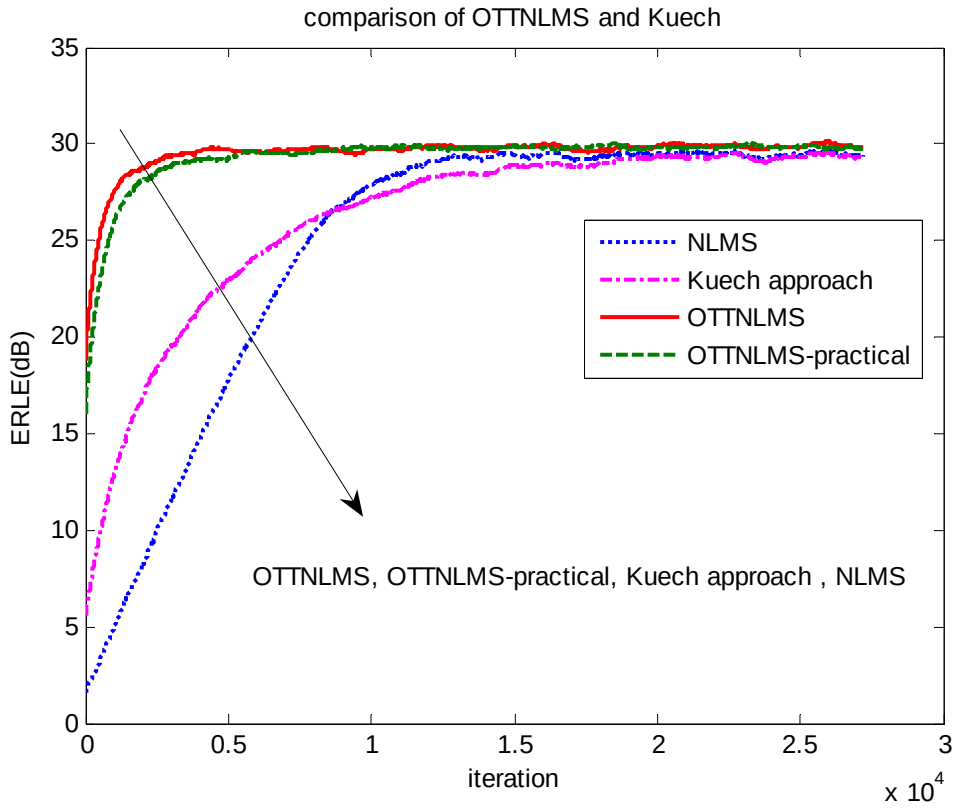


Fig 5.2.16 Comparison of OTTNLMS and Kuech approach (with white Gauss input)

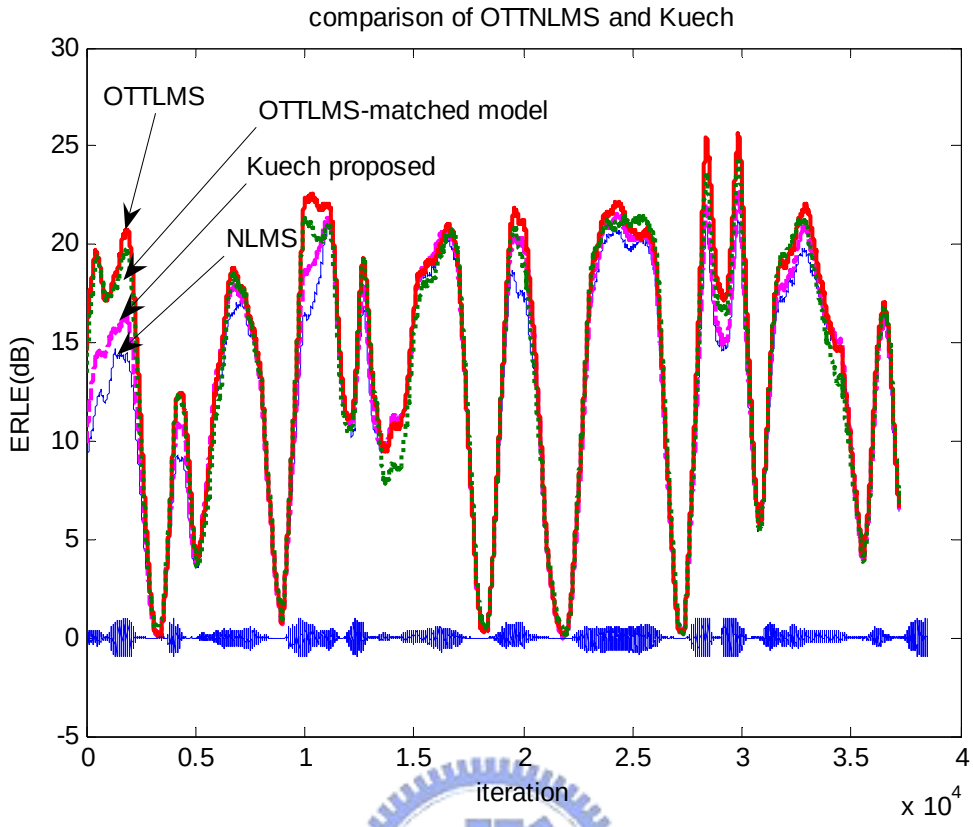


Fig 5.2.17 Comparison of OTTNLMS and Kuech approach (with real speech)

5.2.6 Imperfectly model condition

In below sections, we have already verified that our OTTLMS algorithm provide faster convergence rate than LMS and Kuech approach by computer simulations.

However, we assumed that our filter can perfect model the echo path (i.e. filter order equals to echo path order), it is not critical in general. Thus in this section, we want look the case of imperfectly model condition (i.e. filter order smaller than echo path order), under this condition, we want to compare OTTNLMS to others algorithm (Kuech approach, NLMS). The memory of linear and quadratic kernel is equal to 100 and 10 respectively, which smaller the real echo path orders 200 and 20. The other parameters of Fig 5.2.18 are the same as 5.2.16.

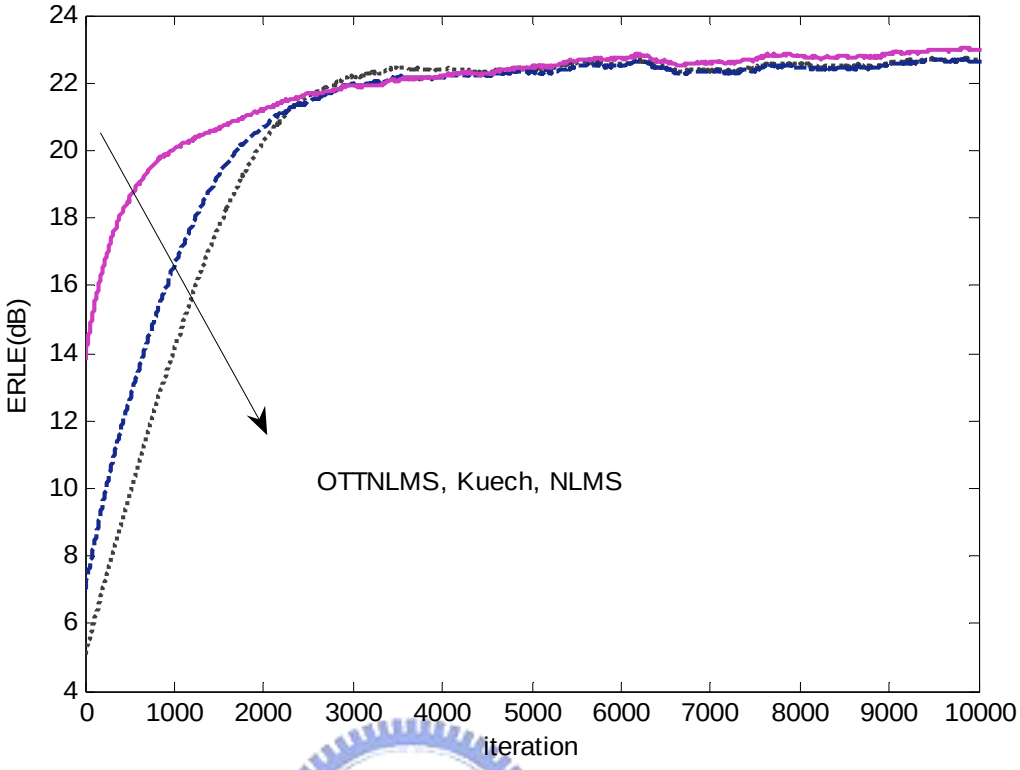


Fig 5.2.18 Comparison of OTTNLMS and Kuech approach in imperfectly model case

From result of Fig 5.2.18, we can obtain that convergent rate of OTTLMS is still faster than Kuech, even in imperfectly model condition**.**

5.2.7 Echo path and Double talk conditions

We have already discussed the echo path and double talk effects in Section 3.5, which our OTTLMS is not robust to echo path change, thus we added echo path detector into our OTTLMS to overcome it. In this section, we will provide the computer simulation to verify the performance of OTTNLMS.

In our simulation, we changed the room impulse response after 14000 iterations, and added near end speech from 25000 to 27000 iterations; both far and near end are white Gaussian noise.

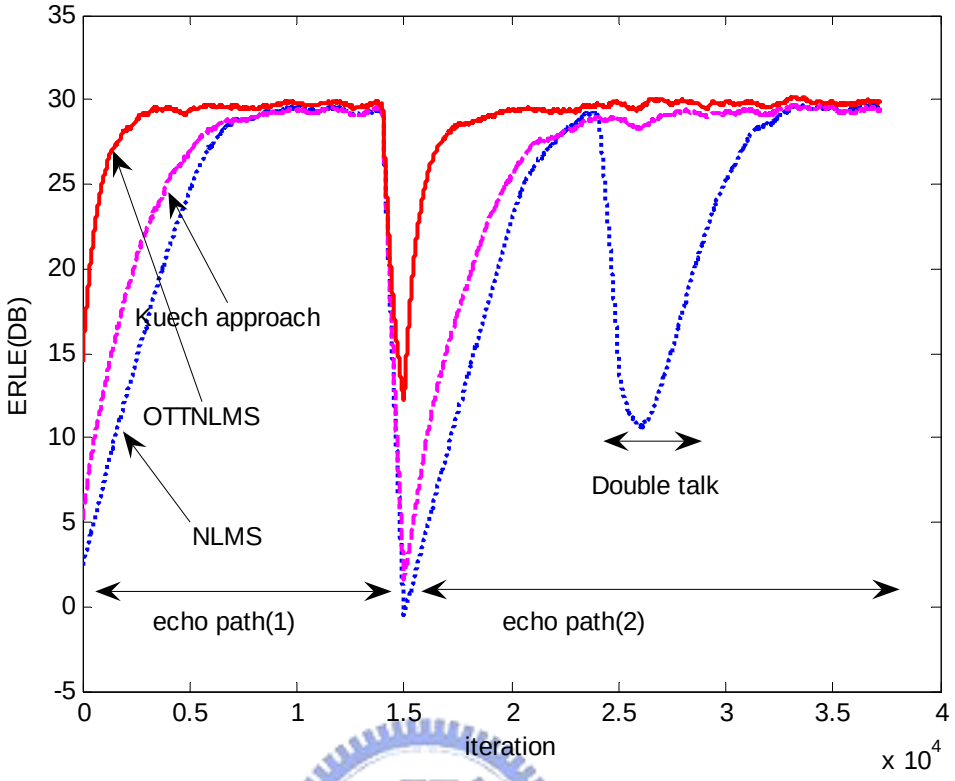


Fig 5.2.19 Comparison of OTTNLMS and Kuech approach in EPC and DT condition

In Fig 5.2.19, we assume that the echo path changed after 15000 iterations, and double talk happed from 25000 to 27000 iterations.

The Kuech approach, it did not considered the double talk condition in [12], we assume that the approach knows when the double talk happened here (i.e. know the real $s(k)$ in (3.1)).

As to our proposed algorithm, we can obtain it maintain on 30 dB, thus our proposed algorithm still work in double talk period.

In echo path condition, we have already introduced the echo path detector [16] into our proposed algorithm in Section 3.5. With the echo path detector, our proposed algorithm can work in echo path condition.

5.3 Performance comparison for channel shortening structure

We have already proposed the least-square solution and theoretical convergent analyses in Section 4.2, in this section we want to use the computer simulations to verify our analyses.

First, we will examine that the least-square solution adaptive AEC filter $\mathbf{h}$ and shortening filter $\mathbf{w}$ in (4.10), and make convolution operation of the shortening filter and RIR (i.e. real room impulse response), to see if the RIR can be shorten or not. To examine it more accurately, we define a quantifiable parameter χ to obtain it.

The χ is denoted as normalized power:

$$\chi = \frac{\left\|\mathbf{f}_{Ls}\right\|_2^2}{\left\|\mathbf{f}\right\|_2^2}$$

where the $\mathbf{f}$ means someone response, and $\mathbf{f}_{Ls}$ means the first L_s elements of $\mathbf{f}$.

In Section 5.3.1, we want to look the length effect of the adaptive AEC filter and shortening filter. By the length effect, we can obtain that how to portion out the resource of filter order.

In Section 5.3.2, we simulate (4.13), we want to see that curves of simulated and theoretical are consistent or not. Next, we will observe that the convergent value of adaptive LMS algorithm and see if it can achieve the least-square solution or not.

5.3.1 Theoretical shortening and original channel

This section corresponds to our analyses in Section 4.2.1, we want to check accuracy of analysis in least-square solution in Figure 5.1.3 shown the RIR, we made convolution of room impulse response and shortening filter from least-square solution

in (4.7).The result of convolution is shown below:

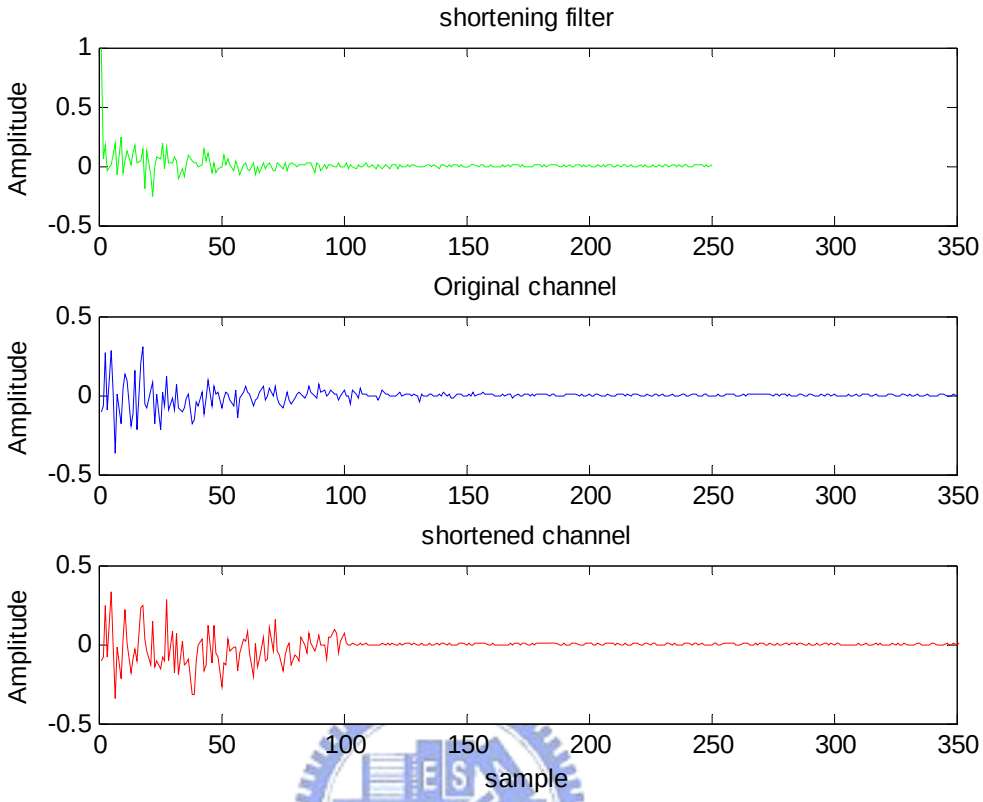


Fig 5.3.1 Original channel and shortened channel from Least-Square solution coefficient

	Normalized power
Original channel	0.9852
Shortened channel	≈1

Table 5.1 Normalized comparison of original channel and shortened channel

In Fig 5.3.1, the coefficients of shortening, original and shortened channels were displayed; the shortened channel is the convolution of shortening filter and real original channel, we omit the shortened channel from 351 to 599 taps, which the amplitude value can be neglected. In Table 5.1, the effective lengths of original channel are equal to about 200, and normalized power is equal to 0.9852. After

passing shortening filter, the effective lengths become 100 (i.e. L_s); and the normalized power is approximately to 1.Thus we can obtain the energy of RIR are shorten to the first L_s .

5.3.2 Different length effect

In this section, we will simulate the performance of coefficient error in different length of shortening filter $\mathbf{w}$ and adaptive AEC filter $\mathbf{h}$, we want to verify the filter length effect to coefficient.

The coefficient error power was defined as $\|\boldsymbol{\varepsilon}\|_2^2$, which the $\boldsymbol{\varepsilon}$ was defined in (4.4), (i.e. difference of $\mathbf{h}$ and the convolution of RIR and $\mathbf{w}$).

Fig 5.3.2 shows that the effects of the length of adaptive AEC filter $\mathbf{h}$, we fixed shortening filter lengths L_w to 100, 200, and 300, and we increased the L_s from 10 to 340 to see if the coefficient error power will decrease or not.

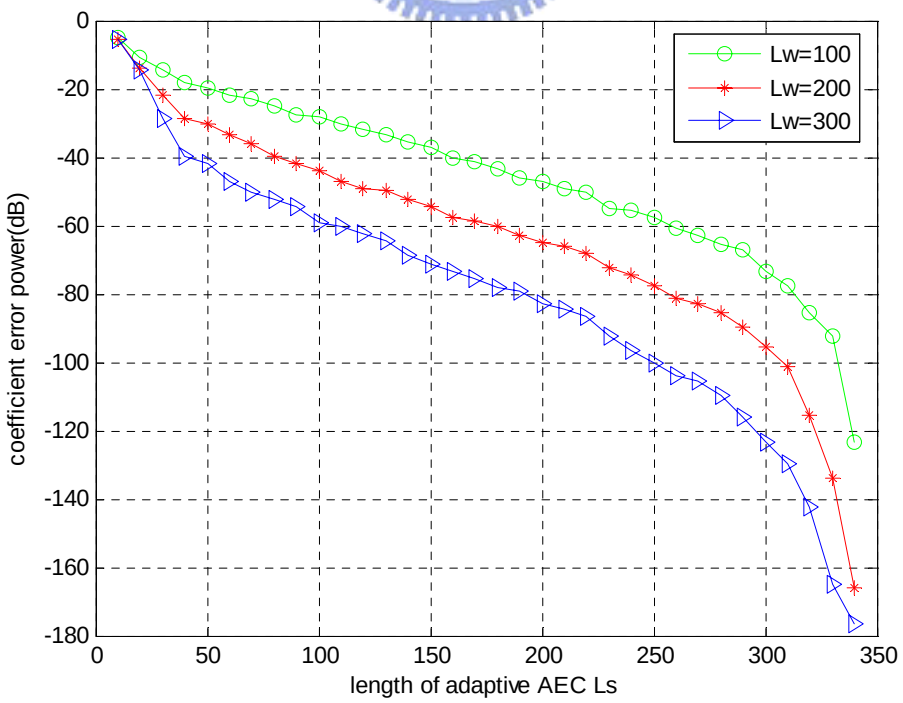


Fig 5.3.2 Coefficient error power effect of different length in FIR filter

Fig 5.3.3 shows that the effects of the length of shortening filter $\mathbf{w}$, we fixed adaptive AEC filter $\mathbf{h}$ lengths L_s to 100, 200, and 300, and we increased the L_w from 10 to 340 to see if the coefficient error power will decrease or not.

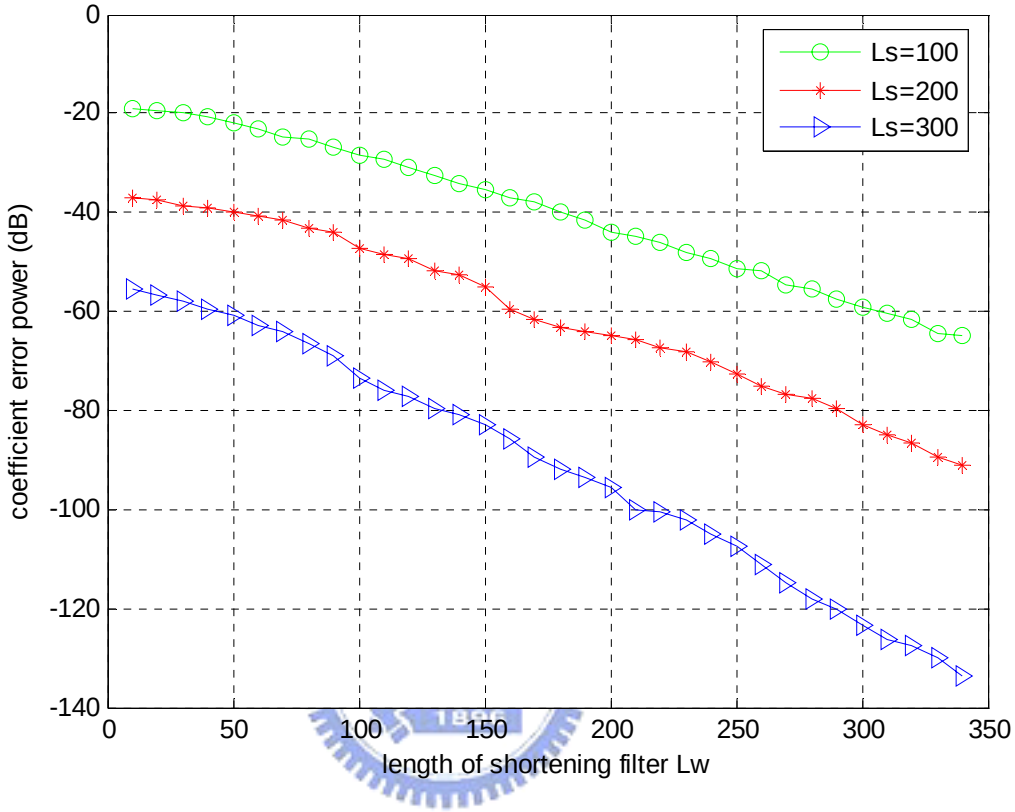


Fig 5.3.3 Coefficient error effect of different length in shortening filter

From Fig 5.3.2 and Fig 5.3.3, we can find that if we increase the length of shortening filter L_w and adaptive AEC filter L_s, the coefficient error power would decease, but it has much computational complexity every iteration.

Next, we want see that how to portion out the resource (i.e. length) of filters, we fixed the summation length of adaptive AEC filter $\mathbf{h}$ and shortening filter $\mathbf{w}$ to M which equal to 100, 250, 350. We increase L_s from 10 to 340 (i.e. decrease L_w 340 to 10) and obtain the value of coefficient error power.

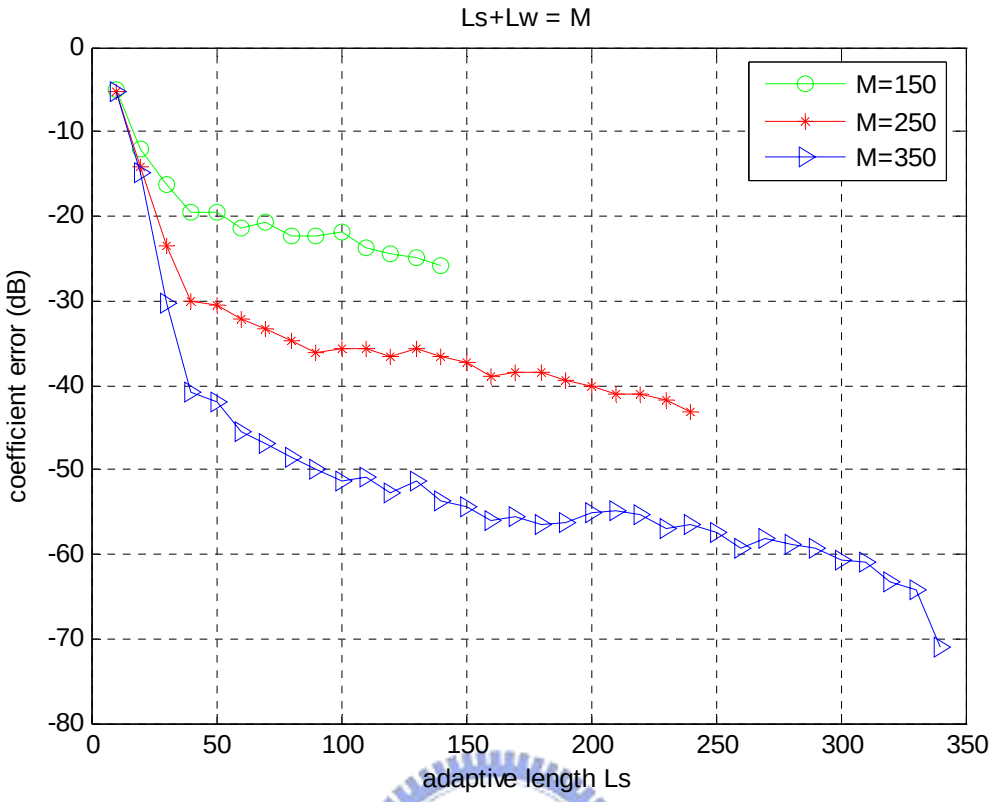


Fig 5.3.4 Coefficient error effect of different length in two filters

From Fig 5.3.4, we generalize a conclusion that the lengths of adaptive AEC filter $\mathbf{h}$ (i.e. L_s) dominate the performance of coefficient error. In this fig, even though the lengths of shortening filter L_w decrease, the coefficient error power is decease as long as increase the length of L_s in assumption of linear echo channel. However, in nonlinear echo path case (i.e. Fig 4.1), from (4.2), we can find that the length of $\mathbf{h}$ also dominate the computation complexity.

5.3.3 Comparison of LMS convergent analysis and simulated

In Section 5.3.2, we have already checked accuracy of analysis in least-square solutions. In this section, we want to examine the accuracy of LMS convergent analysis corresponds to Section 4.2.2. We fixed L_w and L_s to 250 and 100 respectively, and set the step size in LMS algorithm is equal to 0.0005 and 0.002.

As shown, the first moment of coefficient error (i.e. $\|\boldsymbol{\varepsilon}(k)\|_2$) and residual error power in Fig 5.3.5 and Fig 5.3.6, the theoretical curves are plotted from (4.13) and (4.14), respectively, the simulation results agree well with the theoretical convergent curves.

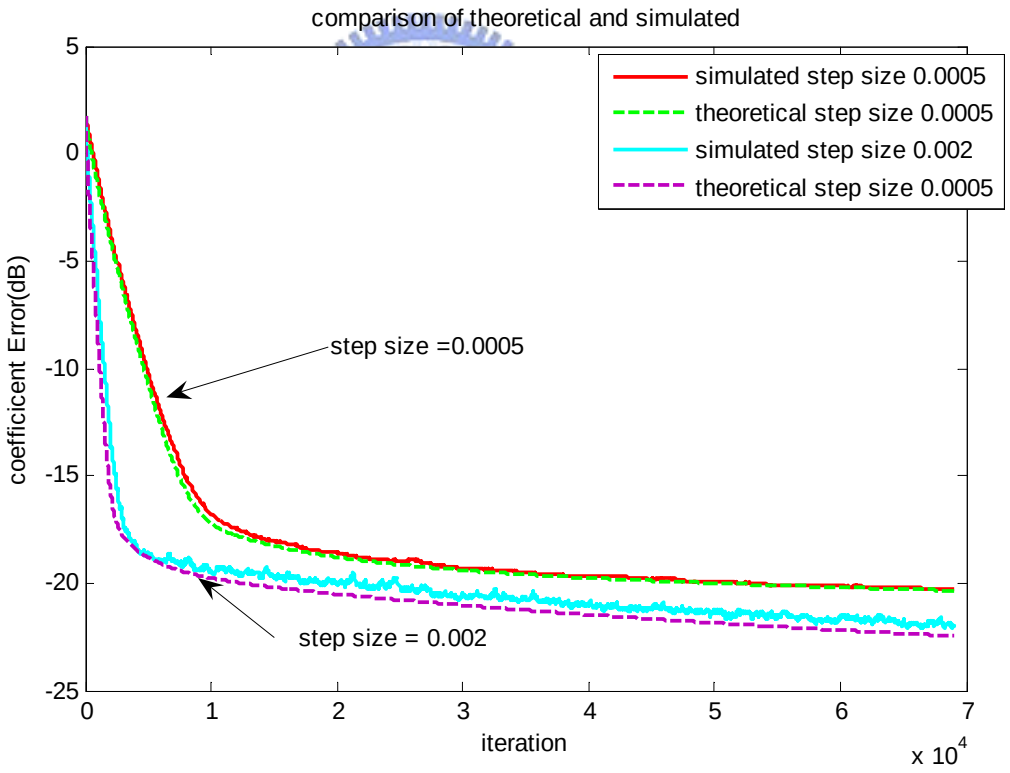


Fig 5.3.5 Comparison of theoretical and simulated (coefficient error)

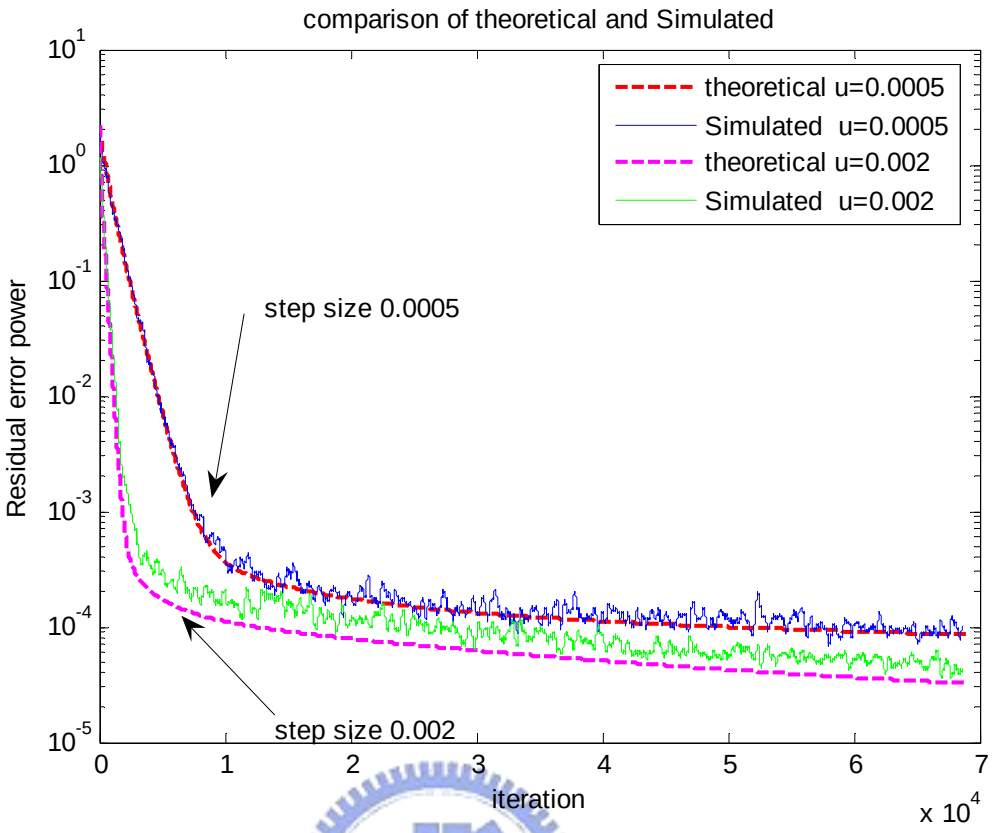


Fig 5.3.6 Comparison of theoretical and simulated (Mean-square error)

5.3.4 Comparison of adaptive LMS algorithm and least-square solution

We have already demonstrated the accuracy of least-square solutions and LMS convergent analysis, in this section, we want to compare their convergent values to discuss if the LMS algorithm can achieve the least-square solution or not. The length of adaptive AEC filter and shortening filter are equal to 100 and 250 respectively, and signal to noise ratio (SNR) is equal to 20 dB.

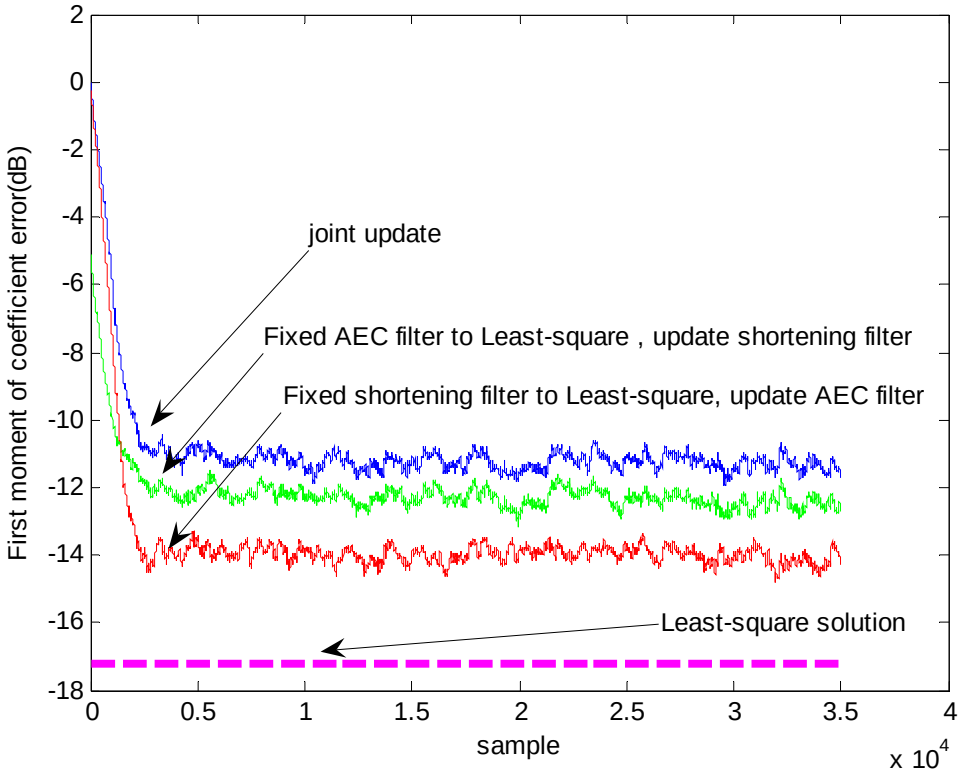


Fig 5.3.7 Coefficient error comparison of LMS algorithm and least-square solution

In Fig 5.3.7, we can obtain that LMS algorithm could not achieve the least-square solution. The first moment of coefficient error of LMS algorithm only converged to about -11 dB, but the least-square solution is about -17.5 dB.

Next, to further obtain the convergent behavior in channel shortening approach, we experimentally fixed one of adaptive AEC filter $\mathbf{h}(k)$ and shortening filter $\mathbf{w}(k)$ to least-square solution, and update the other filter.

And we can obtain that if fix shortening filter $\mathbf{w}(k)$ and just update AEC filter $\mathbf{h}(k)$, the coefficient error LMS algorithm could achieve -14dB, we may conclude that if two filter(i.e. $\mathbf{h}(k)$ and $\mathbf{w}(k)$) joint update, the LMS algorithm can not converge to the least-square solution.

5.3.5 Comparison of multiple stage and joint update

The concept of multiple stage have been introduced in Section 4.3, in Section 5.3.4, we have guaranteed that the LMS algorithm can not converge to least-square solution in joint update, thus we try to perform the multiple stage update in this section.

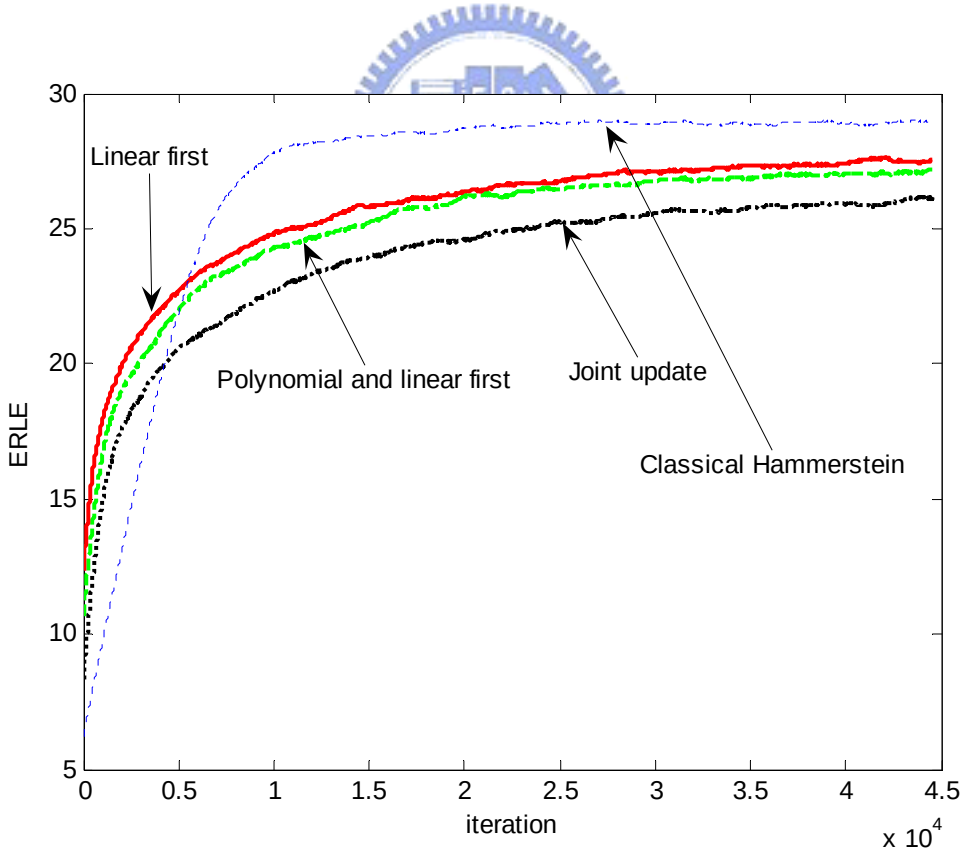


Fig 5.3.8 Comparison of different multiple stage update strategies (with white input Gaussian)

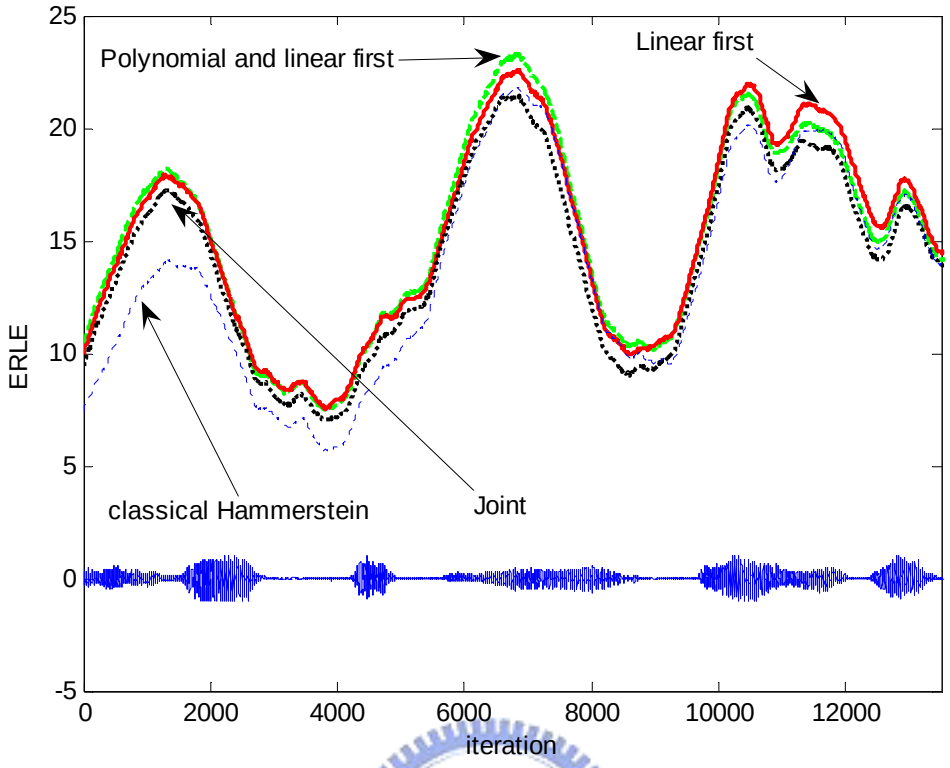


Fig 5.3.9 Comparison of different multiple stage update strategies (with real speech)

In Fig 5.3.8 and Fig 5.3.9, the "linear first" means that we only update the coefficient of linear AEC filter $\mathbf{h}(k)$ in first stage, then all filters joint update in the second stage; the "polynomial and linear first" means that update the Hammerstein polynomial filter (i.e. polynomial filter $\hat{\mathbf{a}}(k)$ and linear AEC filter $\mathbf{h}(k)$) in first stage, the joint update in second stage; "Joint update" means all filter joint update for the whole time.

We can obtain that the multiple stage strategy could not only enhances the convergent rate, but also the converged value. The ERLE performance of "linear first" strategy is best; it has 1 dB and about 3 dB enhancement compare with "polynomial and linear first" and "joint update", respectively.

5.3.6 Volterra with channel shortening and OTTLMS

We already introduced the Volterra model and channel approach in Chapter 2 and Chapter 4 respectively. In this section we want to extend the channel shortening approach to Volterra model, and combine our optimum step size into second order Volterra.

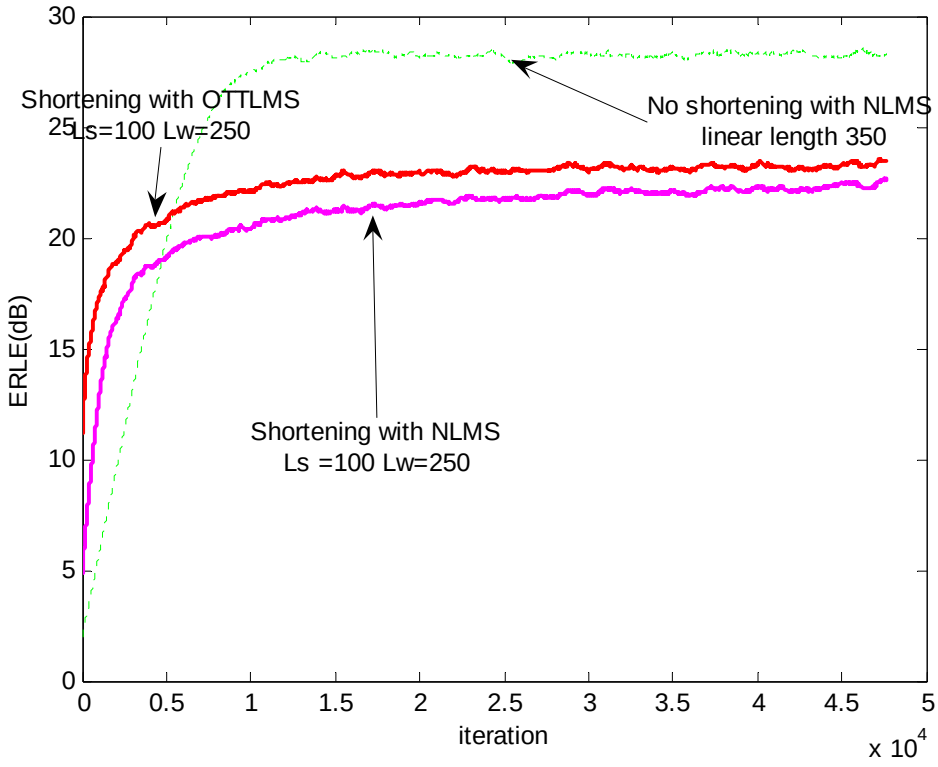


Fig 5.3.10 Channel shortening for second-order Volterra structure (with white Gaussian input)

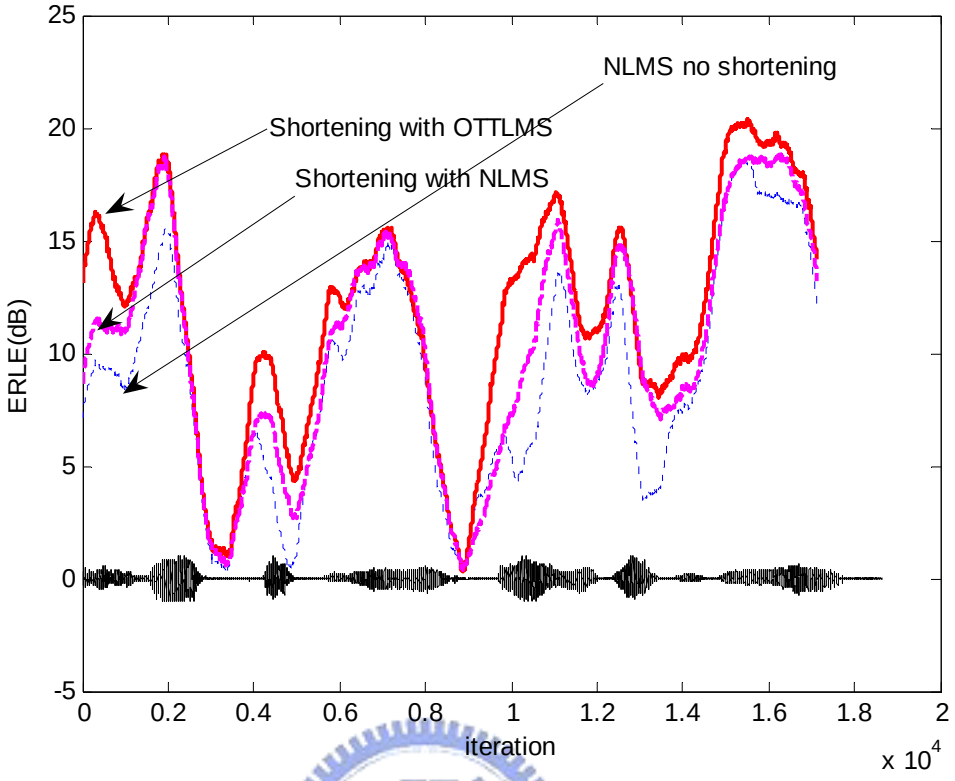


Fig 5.3.11 Channel shortening for second-order Volterra structure (with real speech)

By Fig 5.3.10 and Fig 5.3.11, we can obtain that the shortening structure still work in second order Volterra filter. The combination of Volterra filter and channel shortening filter improve the convergent rate of classical Volterra structure. If we further add the OTTLMS algorithm into this structure, we can obtain not only accelerate the shortening Volterra structure convergence rate, but also ERLE performance about 3dB.

Chapter 6

Conclusion

In this thesis, we employ the Volterra and Hammerstein model to track the nonlinear echo path.

To overcome their slow convergence rate and high complexity, we propose an optimum time and tap– variant step-size for Volterra filter in order to speed up convergence rate in Chapter 3. The step-size is derived by introducing an optimality MMSE criterion between coefficients errors of real and adaptive coefficients. As the optimum step-size need to know the real echo path coefficient, we propose the exponential model for practical implementations.

In addition to adaptive step-size control, the channel shortening structure [14] was proposed to overcome slow convergence rate and high computation complexity.

In Chapter 4, we perform the least-square and adaptive algorithm sense theoretical analysis in channel shortening structure in case of a linear loudspeaker to obtain the convergent behavior. We also propose multiple stage update in this structure to speed up convergence rate.

In Chapter 5, Computer simulations justify our analysis and show the improved performance of the proposed nonlinear acoustic echo canceller.

Bibliography

[1] A. N. Birkett and R. A. Goubran , "Limitations of handsfree acoustic echo cancellers due to nonlinear loudspeaker distortion and enclosure vibration effects ," *ICASSP*, pp.103-106,Oct.1995

[2] A. Stenger, L. Trautmann, and R. Rabenstein, "Nonlinear acoustic echo cancellation with 2nd order adaptive Volterra filters," *ICASSP proceeding*, vol. 2, pp. 877–880, Nov. 1999.

[3] A. Guerin, G. Faucon, and Le Bouquin-Jeannes, R., "Nonlinear acoustic echo cancellation based on Volterra filters," *IEEE Trans. Speech and Audio Processing* vol. 11, no. 6, pp. 672-683, Nov. 2003.

[4] A. N. Birkett and R. A. Goubran, "Acoustic echo cancellation using NLMS-neural network structures," *ICASSP*, pp.3035-3038 vol. 5, May 1995.

[5] K. Narendra and P. Gallman, "An iterative method for the identification of nonlinear systems using a Hammerstein model," *IEEE Trans. Automatic Control*, vol. 11, issue 13, pp. 546-550. Jul. 1966.

[6] K. S. H. Ngia and J. Sjöbert, , "Nonlinear acoustic echo cancellation using a Hammerstein model," *ICASSP,* vol. 2 pp.1229-1232 May. 1998.

[7] N. J. Bershad, “On the optimum gain parameter in LMS adaptation,” *IEEE Trans. Speech and Audio Processing,* vol. ASSP-35, NO. 7, p.p 1065-1068, July 1987.

[8] R. Y. Chen and C. L. Wang, “On the optimum step size for the adaptive sign and LMS algorithms,” *IEEE Trans. Circuits Syst*., vol. 7, NO. 6, p.p. 836-840, June 1990.

[9] R.H. Kwong and E. W. Johnston, “A variable step size LMS algorithm , ” *IEEE Trans. Signal Processing* ,vol. 40,pp. 1633-1642, July 1992

[10] S. Makino, Y. Kaneda and N. Koizumi, “Exponentially weighted step size NLMS adaptive filter based on the statistics of a room impulse response,” *IEEE Trans. Speech and Audio Processing,* vol. 1, NO. 1, p.p. 101-108, January 1993.

[11] J. M. Chen, “Step-Size Adjustments of NLMS Algorithm for Acoustic Echo Cancellation,” *NCTU Master Thesis* ,June 2005.

[12] F. Kuech and W. Kellermann, “Coefficient-dependent step-size for adaptive second-order Volterra filters ”.In *Proc. European Signal Processing Conference (EUSIPCO),*Vienna, September. 2004.

[13] Simon Haykin, *Adaptive Filter Theory*, 4^{th} -edition, Prentice-Hall, 2002.

[14] Kun Shi , Xiaoli Ma , and G. Tong Zhou , “A Channel Shortening Approach For Nonlinear Acoustic Echo Cancellation ”. *Statistical Signal Processing,* 2007. SSP '07. IEEE/SP 14th Workshop on 26-29 Aug. 2007 Page(s):351 – 354

[15] A. Stenger and W. Kellermann ,“Nonlinear acoustic echo cancellation with fast

converging memoryless preprocessor," *ICASSP*, vol. 2, pp. 805–808, Jun. 2000.

[16] Mohammad Asif Iqbal and Steven L. Grant, "A novel normalized cross-correlation based on echo-path change detector," in *Proceedings of IEEE Region 5 conference,* Arkansas, April 2007.

[17] J. P. Costa, A. Lagrange, and A. Arliaud, " Acoustic echo cancellation using nonlinear cascade filters," ICASSP, pp.v389-v392, 2003.

[18] N. J. Bershad, P. Celka, and S. McLaughlin, "Analysis of stochastic gradient identification of Wiener–Hammerstein systems for nonlinearities with Hermite polynomial expansions," IEEE Trans. Signal Process, vol. 49, pp. 1060–1071, May 2001.

[19] F. Kuech and W. Kellermann, "Proportionate NLMS algorithm for second-order Volterra filters and its application to nonlinear echo cancellation," in *Proc. Workshop on Acoustic Echo and Noise Control,* Kyoto, Step. 2003.

[20] D. L. Duttweiler, "Proportionate normalized least mean square adaptation in echo cancellers," *IEEE Trans. Speech and Audio Processing*, vol. 8, pp. 508-518, September 2000.

[21] F. Kuech, M. Zeller, and W. Kellermann, "Input signal decorrelation applied to adaptive second-order Volterra filters in the time domain," in *IEEE Digital Signal Processing Workshop, 12th - Signal Processing Education Workshop, 4th*, pp. 348-353, December 2006.

[22] G. Y. Jiang and S. F. Hsieh, “Nonlinear acoustic eco cancellation using orthogonal polynomial,” *Proc. IEEE ICASSP*, pp. 273-276, Toulouse, France, May 2006

[23] F. Kuech, A. Mitnacht, and W. Kellermann, “Nonlinear Acoustic Echo Cancellation Using Adaptive Orthogonalized Power Filters,” *IEEE Proc. ICASSP’05*, pp. III-105-108, 2005