2102473 Communication Engineering Laboratory

Introduction to Adaptive Filters for Acoustic Echo Cancellation in Hands-free Mobile Communications

Dr Nisachon Tangsangiumvisai
http://www.ee.eng.chula.ac.th/~ntang

Aim: To introduce the students the use of linear adaptive filters in the application of acoustic echo cancellation. The experiments are based on computer simulations via MATLAB. Some introductory examples are given on how to use MATLAB.

The adaptive filters are widely used in many applications nowadays, such as acoustic echo cancellation, channel equalisation in communications, ECG monitoring in medicine, etc. This is because of the capacity of the adaptive filters to vary with changing statistical properties of the signals.

The problem of acoustic echo cancellation (AEC) in hands-free mobile communications, where people are free to move anywhere, is considered here. In this situation, the mobile unit, which contains the loudspeaker and microphone, is placed at some distance from the local (near-end) speaker. When the remote (far-end) speaker is talking, there is acoustic coupling between the loudspeaker and microphone of the mobile unit, due to the reflection of the signals within the enclosures. This leads to the far-end speaker hearing his/her own voice, i.e. **echo**, which interferes the conversation. The purpose of the adaptive filter is to eliminate the undesired echos and noise during conversation. The situation can be illustrated in Fig.1.

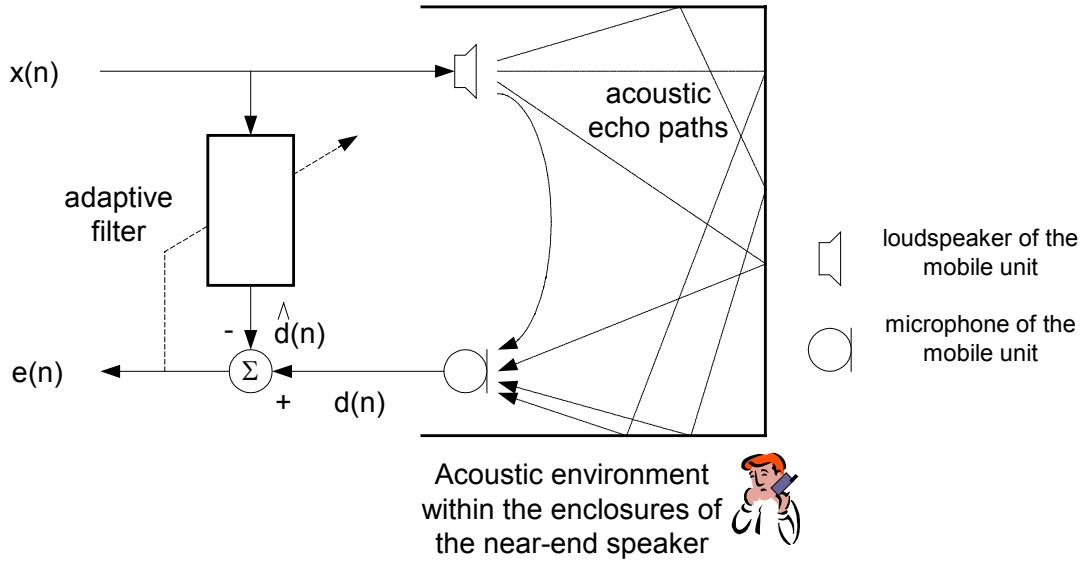

Fig.1 Application of Acoustic Echo Cancellation

The input signal of the adaptive filter, $x(n)$, where n represents the sample index, corresponds to the loudspeaker signal (the far-end speech). The adaptive filter is trying to identify the acoustic echo paths between the loudspeaker and microphone at the near-end site, which are unknown. The output of the adaptive filter, $\hat{d}(n)$, is

subtracted from the microphone signal (the echo signal), $d(n)$, which is called the *desired signal* of the adaptive filter. The error signal, $e(n)$, is the difference between the actual and estimate of the desired signal. To eliminate the undesired echo, the error signal, which is sent back to the far-end speaker, has to be made close to zero as possible.

The adaptive filter is based upon a Finite Impulse Response (FIR) digital filter structure and an adaptive filtering algorithm to adjust its coefficients as time goes by, in order to minimise the cost function of the error, $e(n)$. The function of the adaptive filter is known as the *system identification,* as depicted in Fig.2.

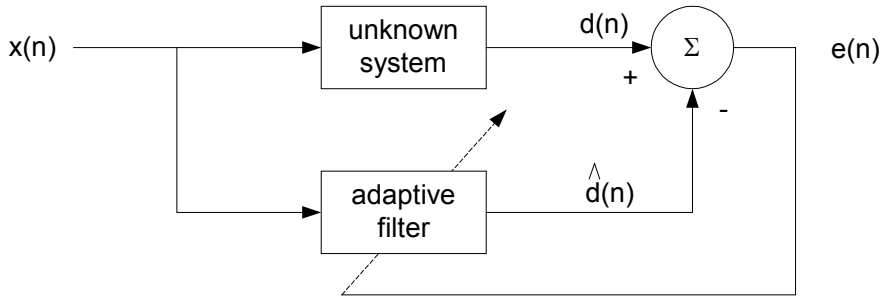

Fig.2 A diagram of a system identification

One of the adaptive filtering approach used in the application of AEC, is the Least Mean Square (LMS) algorithm. This algorithm minimises the Mean Square Error (MSE), $J = E\{e^2(n)\}$, where the error signal is defined by

$$e(n) = d(n) - \mathbf{w}^T(n)\mathbf{x}(n) \qquad \ldots(1)$$

and $E\{.\}$ is the expectation operator and $<.>^T$ is the transposition of any vector or matrix. The output of the adaptive filter, $\hat{d}(n)$, is obtained from the convolution of the input signal, $x(n)$, and the coefficients of the adaptive filter. The parameter $\mathbf{w}(n) = [w_1(n)\ w_2(n) \ldots w_L(n)]^T$ is the coefficient vector of the adaptive filter and $\mathbf{x}(n) = [x(n)\ x(n-1) \ldots x(n-L+1)]^T$ is the input data vector, both of length L. The update equation of the adaptive filter employing the LMS algorithm is given by

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu\,\mathbf{x}(n)e(n) \qquad \ldots(2)$$

This can be seen that the new set of the coefficients of the adaptive filter is obtained recursively at each iteration from the sum of the old set of the coefficients of the adaptive filter and the instantaneous estimate of the negative gradient of the MSE. The parameter μ is called the adaptation gain and is the key factor to control the performance of the LMS algorithm. The adaptation gain normally lies within the range of

$$0 < \mu < \frac{2}{\sum_{k=0}^{L-1} x(n-k)^2} \qquad \ldots(3)$$

where $\sum_{k=0}^{L-1} x(n-k)^2$ is the called tap-input power of the input data vector.

Experiment 1: Implement the LMS algorithm using MATLAB. In this experiment, the input signal of the adaptive filter is a zero mean White Gaussian Noise, generated in MATLAB by the command **randn** of 1000 samples. The desired signal of the adaptive filter, $d(n)$, is found from the convolution of the input signal and the unknown system. The unknown acoustic echo paths **h** are assumed to be time-invariant and is modeled as the following.

$$h_k = \frac{1}{k}\exp\left(\frac{-(k-6)^2}{20}\right)\cdot\sin\left(\pi\left(\frac{k-1}{6}\right)\right), \quad \text{for } k = 1{:}20 \qquad \ldots(4)$$

where 20 is the number of coefficients of the unknown acoustic echo paths. The desired signal can be found using the command **filter**. The coefficients of the adaptive filter are initialised to zero.

(a) Write a MATLAB code for the LMS algorithm. The input parameters of the program should contain the input signal $x(n)$, the desired signal $d(n)$, the adaptation gain μ, and the adaptive filter length L. The output of the LMS algorithm should contain the error signal and the final coefficient vector of the adaptive filter.
(b) Investigate the LMS algorithm when the adaptive filter length is chosen to be the same as the length of the unknown system, i.e. L=20, whereas the adaptation gain is within the range given in Eq.(3). Use the command **plot** and **semilogy** to observe the results, i.e. $e(n)$, $e^2(n)$ and **w**(1000). (or **stem** for comparing **w**(1000) with **h.** Also check the command **hold**)
(c) Repeat the experiment in (b) for 20 times independent input sequences. (This means that you have to find the desired signal $d(n)$ for each set of input signal $x(n)$). Plot the ensemble average (averaged squared error). Comment on the results.

In order to evaluate the performance of the adaptive filtering algorithm, one measurement is terms of the so-called Weight Error Vector Norm (WEVN) (in *dB*):

$$\text{WEVN(n)} = 10\times\log_{10}\frac{\left\|\mathbf{h}-\mathbf{w}(n)\right\|_2^2}{\left\|\mathbf{h}\right\|_2^2} \qquad \ldots(5)$$

where $\boldsymbol{w}(n)$ is the coefficient vector of the adaptive filter at sample number n and the squared norm $(\|.\|_2)^2$ represents the sum of the squared values of the vector. The plot of the WEVN gives an indication of the learning rate of the adaptation of the algorithm.

Experiment 2: Use the same environment as described in Experiment 1, but this time, the background noise is added to the desired signal. This is achieved by adding a zero mean, white Gaussian noise (another sequence, **not** $x(n)$) to $d(n)$ so that the signal-to-

noise ratio (SNR) of the desired signal is 30 *dB*, i.e. the ratio of the power of $d(n)$ to the power of the background noise is 1000. Discuss the results when:

(a) The adaptive filter length is chosen to be of 3 cases; (1) L=10, (2) L=20, (3) L=30.

(b) What are the effects of increasing and decreasing the adaptation gain μ? Comment of the speed of convergence of the LMS algorithm (from the curve of WEVN) and its misadjustment performance (final value of WEVN).

Experiment 3: In the real situation, the loudspeaker signal is speech. Use the signal s1.mat as the input of the adaptive filter $x(n)$ by the command **load** in MATLAB. Comment upon the choice of the adaptation gain for the LMS algorithm when the input signal is a speech signal. (Do not forget the background noise of 30dB SNR)

Now, try normalizing the input signal s1.mat (subtract by its mean, and divide by its standard deviation) and observe the choice and the effect of the adaptation gain.

(Optional : If you want to listen to the speech signal and the error signal after employing the LMS algorithm, try wavread.m)

Experiment 4: The LMS algorithm is therefore modified to its normalised version (NLMS) with the update equation of the form

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \frac{\mu}{1+\left\|\mathbf{x}(n)\right\|_2^2}\mathbf{x}(n)e(n) \qquad \ldots(6)$$

(a) Write a MATLAB code and simulate this program when the input signal of the adaptive filter is a normalized version of the speech signal (s1.mat). Comment on its performance. Compare the computational complexities of the LMS and NLMS algorithms for real-time implementation.

(b) Explain why the NLMS algorithm is more suitable for use with real speech signals?

Reference:

[1] Haykin, S. "Adaptive Filter Theory", 3rd Edition, Prentice Hall, 1996, Chapter 9.
