ADAPTIVE FILTERING TECHNIQUES FOR AUDIO SOUND RECOVERY

By

DOAN THANH LONG

Dissertation submitted to the Electrical & Electronics Engineering Program in partial fulfillment of
the Requirements for the
Bachelor of Engineering (Hons)
(Electrical & Electronic Engineering)

December 2013

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

# CERTIFICATION OF APPROVAL

ADAPTIVE FILTERING TECHNIQUES FOR AUDIO SOUND RECOVERY

by

Doan Thanh Long

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Approved by:

_____________________________

Dr Vijanth Sagaya Asirvadam
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

December 2013

# CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

____________________________

Doan Thanh Long

# ABSTRACT

Noise is the common problem that affects not only the daily life but also the industrial manufacturing process. There are many different type of noise coming from various sources. A lot of work has been done to improve the audio control system, especially focus on the sound recovery and noise cancellation. Adaptive filter recently have been used as a best tool to eliminate the noise and give the good result on recovering the original pure audio signal. The number of applications applying adaptive techniques has increased tremendously on the fields like telecommunication, signal processing, biomedical and sonar.
The objective of this project is to applying adaptive filtering techniques to cancel the noise from the mixed signal and recover the clean original audio signal.

The project will be divided into two stages. For the first stage, the reference signal (original signal) will be given for the prediction and take part in noise reducing process. For the next stage, the noisy signal will be processed and filtered without the presence of reference signal (clean audio signal). In addition, the project also involve in matrix computation technique for the adaptive designing step. There are different types of adaptive filter configurations. For this project, adaptive linear prediction configuration will be implemented in the first stage. The adaptive system identification and adaptive noise cancellation configuration are used for the later part. Adaptive algorithm applied in this paper is Least Mean Square algorithm. The output of the filter is expected to be as closed as possible to the original signal. All the work will be done by using simulation software (Matlab).

# ACKNOWLEDGEMENTS

I would like to take this opportunity to express my gratitude to all the individuals and groups that give me a lot of supports in this Final Year Project. I would like to say many thanks to:

- Dr Vijanth Sagaya Asirvadam (Project Supervisor)
- Final Year project Committee
- Resource Center of University Technology PETRONAS
- University Technology PETRONAS lecturers.

Lastly I would like to thank my friends who have supported me and my family who encouraged me throughout my academic studies.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# Chapter I INTRODUCTION

## 1.1 Background:

For the last decades, a lot of researches and experiments have been implemented for people to learn and know more about the audio control systems. Among these systems, sound recovery and noise cancellation are the two aspects that are concentrated and carefully tested. The common technology used for the systems are digital signal processing. However, for the clear and better results, instead of using only the normal DSP techniques, the adaptive filters, which successfully used in different type of applications, are becoming the best choice. They have widely used in different industries and fields like telecommunication, signal processing, radar.

The special thing about adaptive filter is that it has the ability to self-adjusting its own filter coefficients to adapt the input signal with the support of the adaptive algorithm. For the adaptive filters to work properly, we need to design the filter base on the algorithm of adaptation. Most popular types of filters using nowadays are finite impulse response (FIR) and infinite impulse response (IIR) filters.

For this paper, only the Finite impulse response filter are discussed. Among different algorithms employed by adaptive filters such as recursive-least-squares (RLS), Normalized least mean squares (NLMS), the basic least-mean-squares (LMS) algorithm is the most effective one that will be used in the simulations.

In general, audio sound commonly interrupted by some noise from the surrounding environment. The task will be investigating on both white and colored noise in audio and the process of eliminating noise in audio signal using fast adaptive system identification techniques. The project involves the implementation of fast matrix computation techniques for filtering.

## 1.2 Objective:

To employ adaptive filtering techniques to cancel the noise and recover the original audio signal.

To use matrix computation technique to build adaptive filters. All steps are implemented via MATLAB simulations.

To development ensemble supervised and unsupervised technique for audio signal recovery.

## 1.3 Scope of study:

Noise cancellation using adaptive filtering technique has been implement for a very long time. In this paper, we mostly focus on the adaptive system identification technique with the support of Least Mean Square algorithm. All the experiment will be implemented by MATLAB simulation.

## 1.4 Problem Statement:

Most of the sounds in daily life are affected by the noise coming from surrounding environment or different type of noise sources. Filtering audio signal from environment noise which is sometimes mixed colored noise is a challenging task. The difficulty level can increase with the absence of reference signal (clean audio signal). Thus the project can be divided into 2 stages. The reference signal is provided for the first stage to predict and correct the noise corrupted audio signal to original form. Whereas in the second stage, prior signal pattern (in form clusters) will be used as a filtering (or classification) tool.

Designing adaptive filter with the matrix computation technique employed will recover the audio sound with and without reference signal.

# Chapter II LITERATURE REVIEW

The project is working on the noise cancellation and audio sound recovery. Most of the audio signal is interfered by variety of noise sources. The digital filtering is the common technique that helps in recovering the original audio signal. The mixed signal that corrupted by noise will be put through a filter and eliminate the noise to result in clean elementary signal.

Basically there are two types of filters: Fixed and adaptive filters. For the fixed filter, it is a need to have the reference of both signal and noise to process. Later, the filter will be designed with the passing frequencies is the frequencies of original signal. In addition, the frequency band of the noise will be stopped. In the real life application, finding a reference for both original signal and noise signal is not a easy job. This is the reason why fixed filter is not preferable.

Adaptive filter, on the other hand, is an electric filter whose frequency response varies with time, as a function of the input signal. The advantage of adaptive filters is that it does not require the reference of both signal and noise, or even both of them [1].

In this report, for the noise cancellation and signal recovery, we use the adaptive filtering technique to get the precise results in a short time. One filtering system will have at least following parameter: input x(n), output y(n), desired result d(n), error signal e(n) ( the different between the desired output and real output) and the weight of the adaptive filter w(n). These parameters will work together and change continuously to adapt for the desired result.

## 2.1 Filter configurations:

In general, there are four types of configurations for the adaptive filter: Adaptive system identification, Adaptive noise cancellation, Adaptive linear prediction, and Adaptive inverse system [2]. Each of these configurations has its own advantages and disadvantages when comparing to each other. They are all applied in different scenarios, based on the requirement of each project.

### 2.1.1 Adaptive system identification

The purpose of the Adaptive system Identification is originally to determine the transfer function of one unknown system.

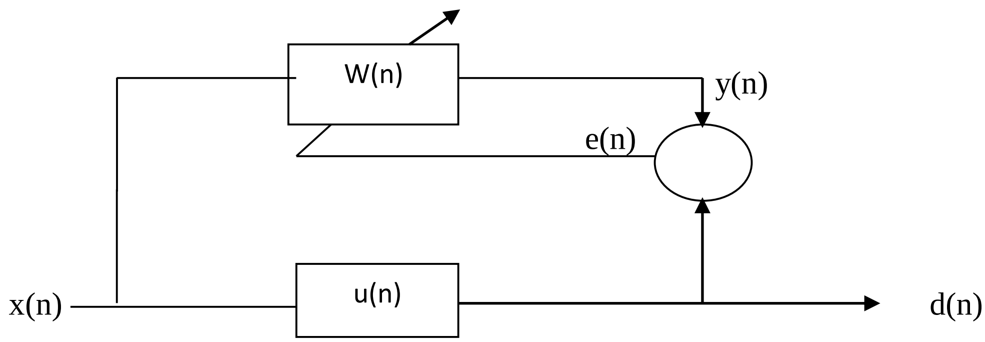


*Figure 2.1: Adaptive system identification*

Some of the iterations will be done until the filter transfer function reach to the unknown's one. The error signal in this case will converge to 0. The problem with adaptive system identification is that the use of the least mean square (LMS) is not stable in the low signal-to-noise ratio region [3].

### 2.1.2 Adaptive noise cancellation:

Main objective of this system is to cancel the noise from the mixed signal. We have the reference for the noise and the original signal will be achieved by subtracting the noise reference input from the corrupted signal.

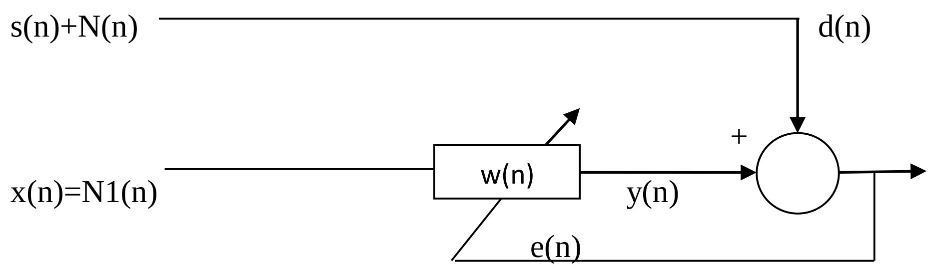


*Figure 2.2: Adaptive noise cancellation*

The condition for the system to work properly is that 2 noise source should be correlated to each other and not to the original signal. After several rounds, the error signal will go to the initial signal which is s(n).

### 2.1.3 Adaptive linear prediction

Nowadays, the application of the system has been spread in some fields like spectral analysis and speech encoding. By predicting the current value of signal using linear weighted sum of n delayed signals, the least mean square error will be reduced [4].

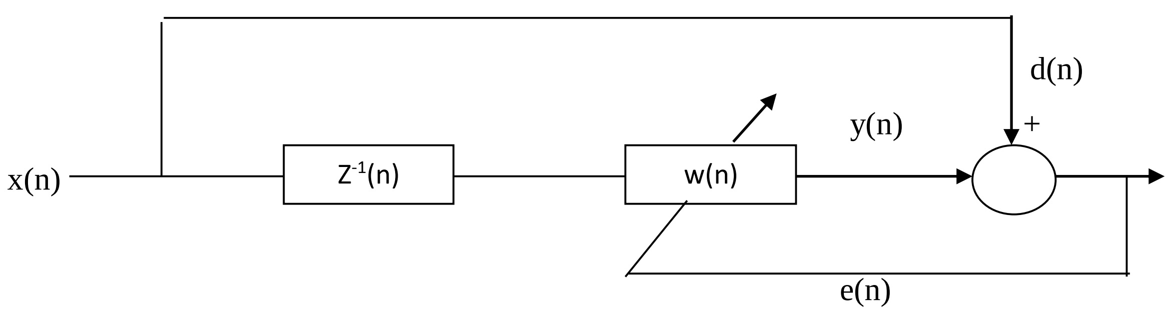


*Figure 2.3: Adaptive linear prediction.*

### 2.1.4 Adaptive inverse System:

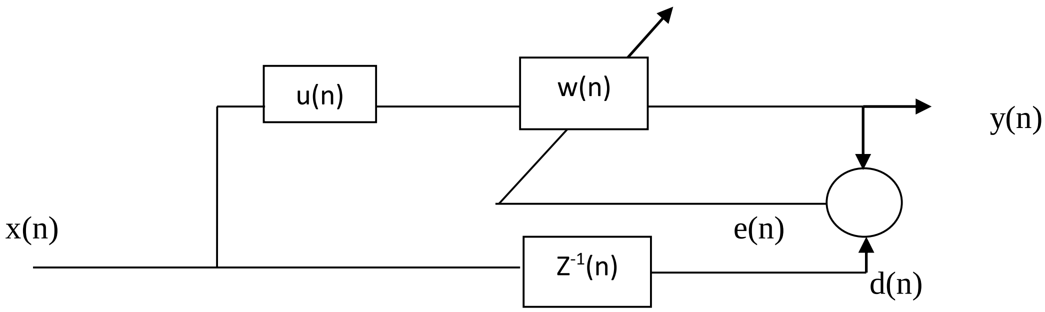


*Figure 2.4: Adaptive inverse system*

By using the adaptive inverse system, we can create the inverse of one unknown system (in this case is u(n)).

The input x(n) will be put through the unknown system first, then pass by the filter to create output y(n). In addition, x(n) goes with a delay to create signal d(n).

After some iterations, the error signal goes to zero. As the result, the transfer function of the filter is the inverse of the one of unknown system.

## 2.2 Type of Filters:

There are 2 types of adaptive filter: Finite impulse response (FIR) filter and Infinite impulse response (IIR) filter. A finite Impulse Response (FIR) filter is a digital filters that have weighting sequence (impulse response) with finite in length. Usually thereare no feedback for FIR filter. Infinite impulse response (IIR) filter, on the other hand, will have the feedback loop and the impulse response go continuously without reaching zero value.

In this paper we will use FIR filter due to its advantages over the IIR such as: coefficients are simple to calculate, liner design method and its high stability. We can apply different algorithms to the adaptive filter to control how the filter adjusts the adaptive coefficients. Two popular groups of algorithms that used popularly for the adaptive filter are Least Mean Squared (LMS) algorithms and Recursive Least Squares (RLS) algorithms.

LMS algorithms (App.I) basically will adjust the coefficient of the adaptive filter until it produces the least mean squared of the error signal e(n).

The drawback of this algorithm is the unstable performance. However, the stability can be enhanced by adding sufficient conditions [5]. Recently, there are some proposals to improve or upgrade the LMS algorithms like: Total Least Mean Squared algorithms [6], Extended Correlation Least Mean Squared algorithm [7].

RLS algorithms is an algorithm that instead of minimize the least mean squared of the e(n), it will recursively look for the filter coefficients which can reduce a weighted linear least squares cost function relating to the input signals. In general, recursive least squares (RLS) algorithms have advantages over LMS one in the convergence speed and it works really well in the time-varying environment. The disadvantage of RLS algorithms is the difficulty and complexity in calculation.

In this paper, we will employ the FIR filtering technique with the LMS algorithm to filter out the noise and recover the original audio signal with or without the reference.

## 2.3 Filter Order Selection:

There are many techniques that are used in the filter selection field. However, the most popular techniques is Mean Square Error Estimation technique. Mean Square Error (MSE) is the estimation of the difference between the real value and the predicted value. By using MSE, we can observe the performance of each filtering order and how effective that order is. The filtering method with the minimum mean square error is going to provide the output that is closest to the original expected value.

Besides MSE, there are some other improved methods also focus on the analysis of the performance of the estimator like: Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), Normalized Mean Square Error (NMSE). All of the methods above are the key tools to evaluate how good is the result of a filter and have been used as a criterion to select the best filter order [8].

## 2.4 Filter Performance Improvement:

There are several techniques that have been researched and implemented to enhance the performance of the adaptive filter. Known as one of the most effective technique in improvement of filter output, regularization is becoming a critical part of filter design. By adding the additional coefficient in to the model, it prevent the overfitting in predictive model, which lead to more accurate result. In addition, regularization can help to stable the filter and fasten the solution convergence process [9].

However, the work on this technique is still very limited and not focused by the researcher. Some of the work have perform the regularization technique in the noise cancellation with the main algorithms like Normalised Least Mean Squared algorithms (NLMS), Squared root NLMS (Sr-NLMS) [10].

## 2.5 Autoregressive-moving-average model:

Autoregressive-moving-average (ARMA) models are mathematical model that is used to predict the behavior of the time series system. ARMA can provide the suitable fit data and reduce the error term. It is widely used for the forecasting work in the economic and industrial system. The model consists of two parts: Auto-regression and Moving-average.

$$X_t = c + \varepsilon_t + \sum_{i=1}^{p} \alpha_i X_{t-i} + \sum_{i=1}^{q} \beta_i \varepsilon_{t-i} \tag{1}$$

Where $X_t$ is predict value. $\varepsilon_t$ is error. C is a constant, $\alpha$ and $\beta$ are parameters

There are many applications of the ARMA digital filter: Electroencephalographic signal simulation [11], Music analysis [12], Spotting system for telephone speech.

For the noise cancellation without reference, we will employ this model to enhance the result of the filter. The output is not expected as good as the work with the reference signal. However, ARMA model can give a quite precise result and minimize the error.

# Chapter III METHODOLOGY

The general idea of audio sound recovery project is to cancel the noise and bring back the original pure signal. For this project, we employ the adaptive linear prediction configuration. The Adaptive system Identification and adaptive noise cancellation configuration will be also used for the later improvement of the project.

## 3.1 Experimental Design:

All the experiment will be implement using simulation software. For FYP I, the work will be carried out with the presence of reference signal (clean original signal).

Basically, we have s(n) which is the reference signal. Generating some random noise N(n) by matlab function, we create the noisy signal mix(n) by adding the original s(n) and the noise N(n). The mix signal mix(n) later will be put through the delay system ($z^{-1}$(n)) then feed into the filtering sytem. We expect the output from the system which is xP(n) will be closed to the original signal s(n). The difference between xP(n) and s(n), error signal e(n) will show how close is the result from filtering work and the reference signal.

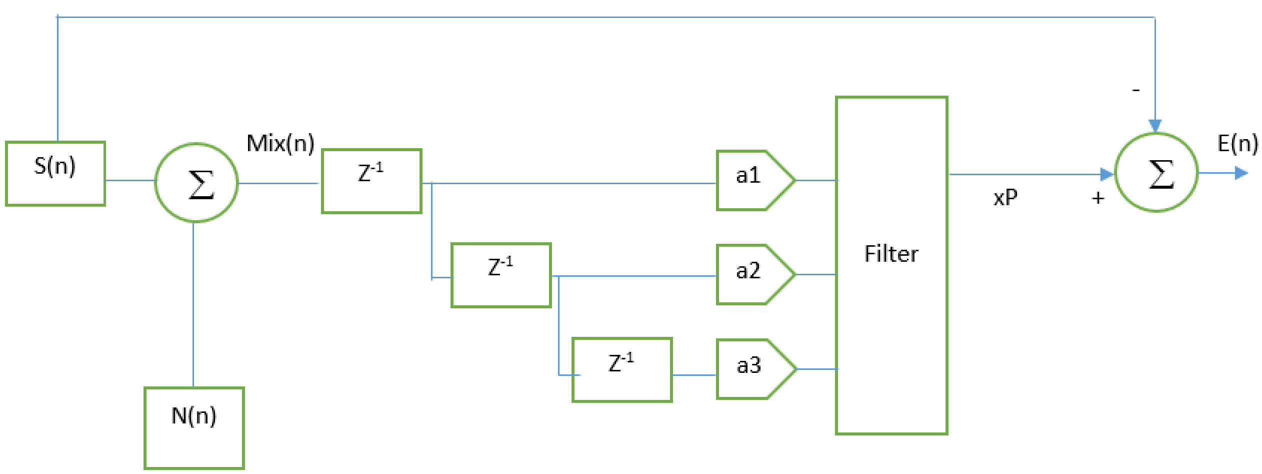


*Figure 3.1: Model Design*

## 3.2 Procedure:

We will divide the project into two stages. The first stage is the noise cancellation work with reference signal. For more challenge, in the second stage, the work will be implemented without reference signal.

### 3.2.1 Noise cancellation with reference signal

First, a portion the audio signal (in this case is a song) will be recorded by using Matlab tool.

Similarly, the random noise will be generated with random values. We have all the data of the original signal and the noise in form of matrix. By adding two signal, the mixed signal is the noisy signal that can be heard clearly by using audio convert command.

By putting the noisy signal mix(n) through a tap delay, we can create a number of delay vector depend on the order of the filter we design. The experiment will be implemented with 3 orders: order 3 and order 6 and order 10. The result will be compare to see if the order of the filter really affects the filtering performance.

Now we consider the steps for order 3. Same procedure will happened to order 6 and 10.

We will build a matrix including 3 vectors which are the delay of the noisy signal mix(n). The resulting vector will be $X(t) = \{x1, x2, x3\}$.

Since vector X(t) is go through the filter with the matrix coefficient A to get the output xP. xP(t) will be :

$$[xP] = \{x1, x2, x3\} * [A] \tag{2}$$

Where [A] = $\begin{bmatrix} a1 \\ a2 \\ a3 \end{bmatrix}$

The matrix A can be found by multiplying Moore-Penrose pseudo-inverse of matrix [X(t)] with the reference signal s(t).

As the result, we can calculate the matrix [xP(t)] which is the product of input [Xt] and [A].

From the data of [xPt], we can create an audio signal as well as graph to check for the noise cancellation process.

For the improvement of output from filter, we start to apply the regularization technique to bring more accuracy. By adding a same size small-value matrix to the original matrix A, the output xP of the filter will be a bit closer to the reference signal.

The error as the result will be reduced. In this case the new matrix A will be:

A = (inv(Xt'.Xt) + 0.001.eye(3)).Xt'.s_1 (3)

Thus, the new output xP will be calculated by multiplying input Xt and A as previous step.

For the next step, the Least Mean Square algorithm is employed. The step size $\lambda$ is chosen as 0.05.

The coefficients of the filter will be updated continuously by using the For loop with the length of input matrix Xt. Hence, the output can fastly converge to the optimum value.

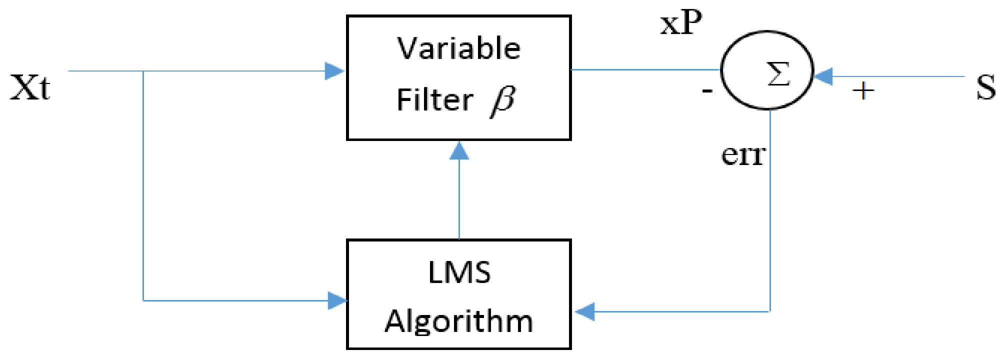


*Figure 3.2: Adaptive Filter Mechanism*

We will employ graph plotting and error estimation method to assess the performance of the adaptive filter.

### 3.2.2 Noise cancellation without reference signal

Filter out the noise without the presence of the reference is a big challenge. Most of the previous filtering work can only cancel the noise due to the known specific characteristic of the original signal like: periodicity, constant magnitude… However, in this paper, we work with the random audio signal. The ARMA model will be carry out to see if it can improve the result.

First, we use the normal adaptive filter with LMS algorithm. By converge the output of the filter to the mixed signal, the noise is expected to cancel out a bit and the output can step closer to the unknown original signal.

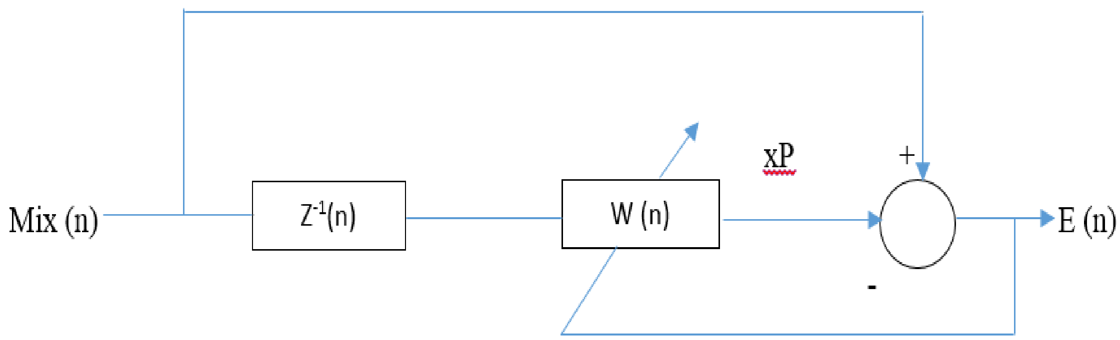


*Figure 3.3: Adaptive Filter Without Reference*

The design is similar to the previous work. The only difference is that we use the mix signal instead of the clean original signal as a parameter to find the error E(n).

For the improvement of the result, ARMA model is employed. The input Xt for the adaptive filter now is modified by adding the error signal into it.

Xt = {x1,x2,x3,err}

With initial value of the error: err = 0.1*E (4)

E is the random matrix with the size (82126,3).

We apply the NLMS algorithm. The stepsize is chose as 0.00008. By reducing the error, the output of the filter is expected to have the similar form to the original clean signal. The error estimation process also be carried out to observe the improvement of the work.

## 3.3 Milestones:

Following is the milestone for this project.

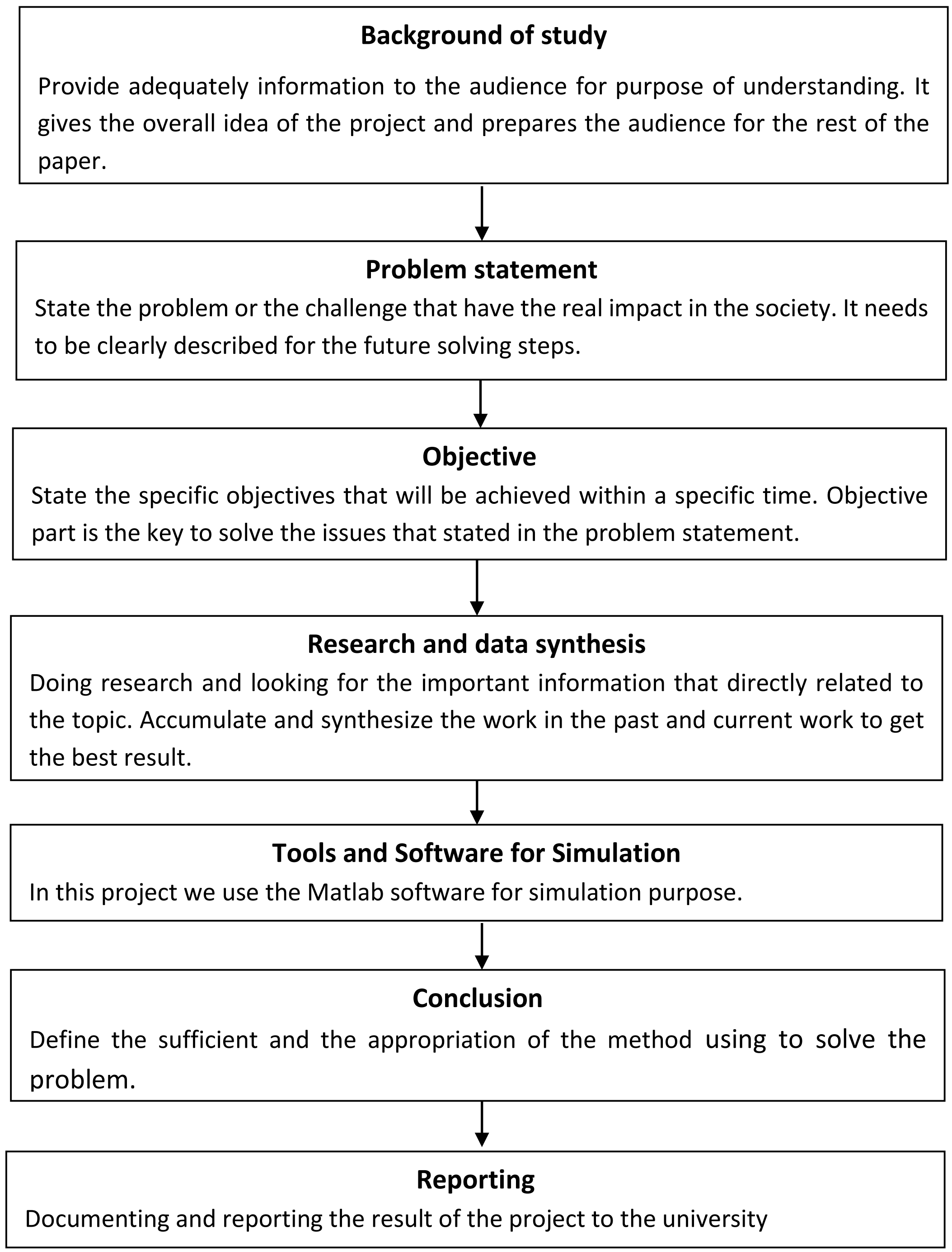


*Figure 3.4: Milestone*

## 3.4 Gantt Chart:

Below is the Gantt Chart

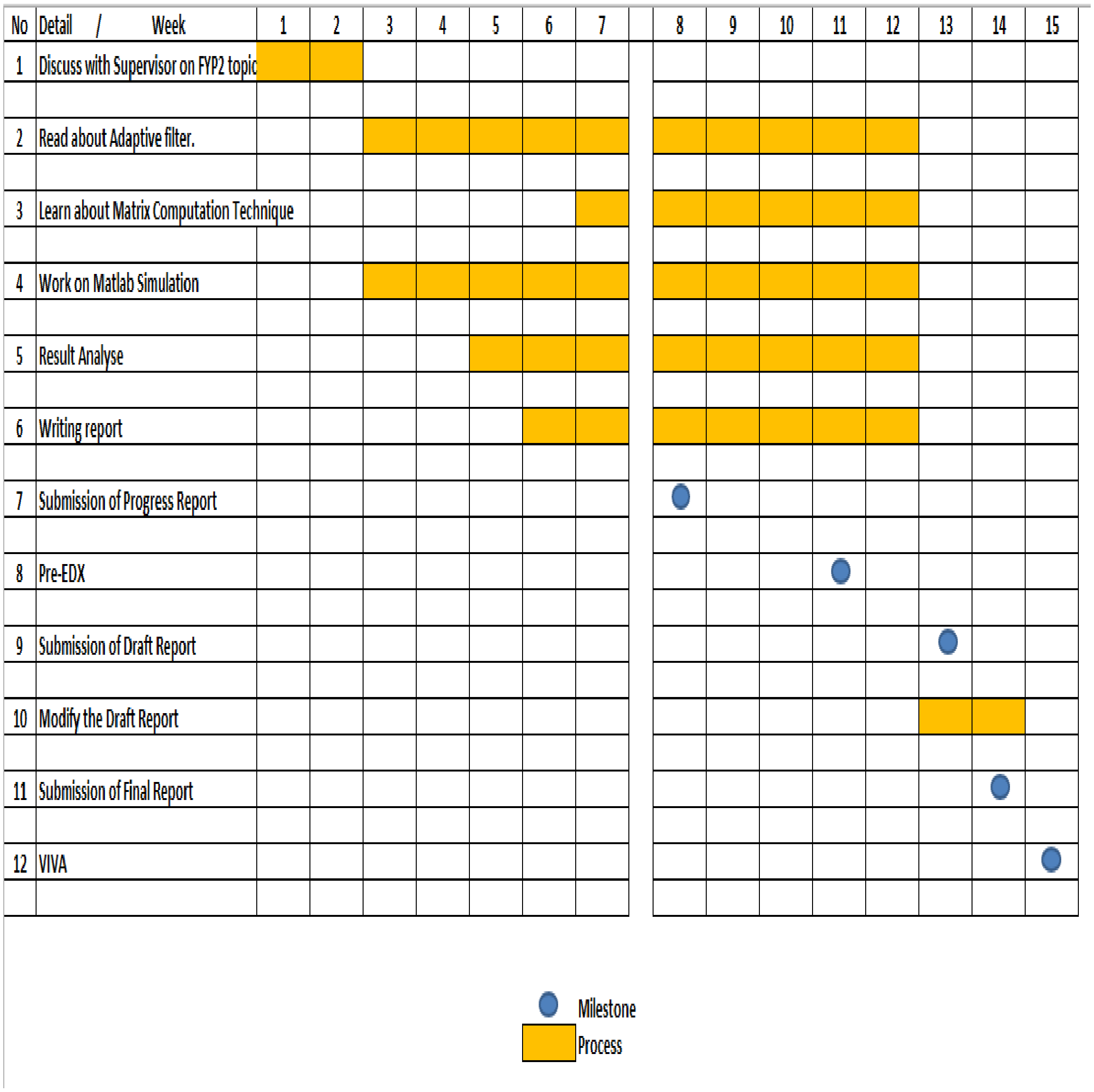


*Figure 3.5: Gantt Chart*

## 3.5 Tool & Software:

Matlab is a very useful software where the data can be arrange in the array. For the project that deal with the matrix calculation and filtering technique, Matlab offer sufficient tool to calculate and simulate effectively. Matlab programming is simple and helpful with a lot of supports.

In addition, the interface in Matlab software is quite friendly. That the reason why we choose Matlab for this project simulation.

# Chapter IV RESULTS & DISCUSSION

## 4.1 Noise cancellation with reference signal

With the reference signal, the results below show the performance of the filter in two modes: Offline and Online.

### 4.1.1 Offline Filter's Result on basic design:

With the fixed coefficients, an amount of noise is expected to be cancel out. However, the output from the filter will not be as good as compared to the adaptive filter which will be shown later on.

#### 4.1.1.1 FIR model order 3:

The main purpose of the filter design is to minimize the error function. The complexity or filter type is defined by the filters order. By applying 3 tap delays for the filter, we have the model of the order 3th filter.

By simulating the process on Matlab (code attached in the Appendix), we got the following graph:

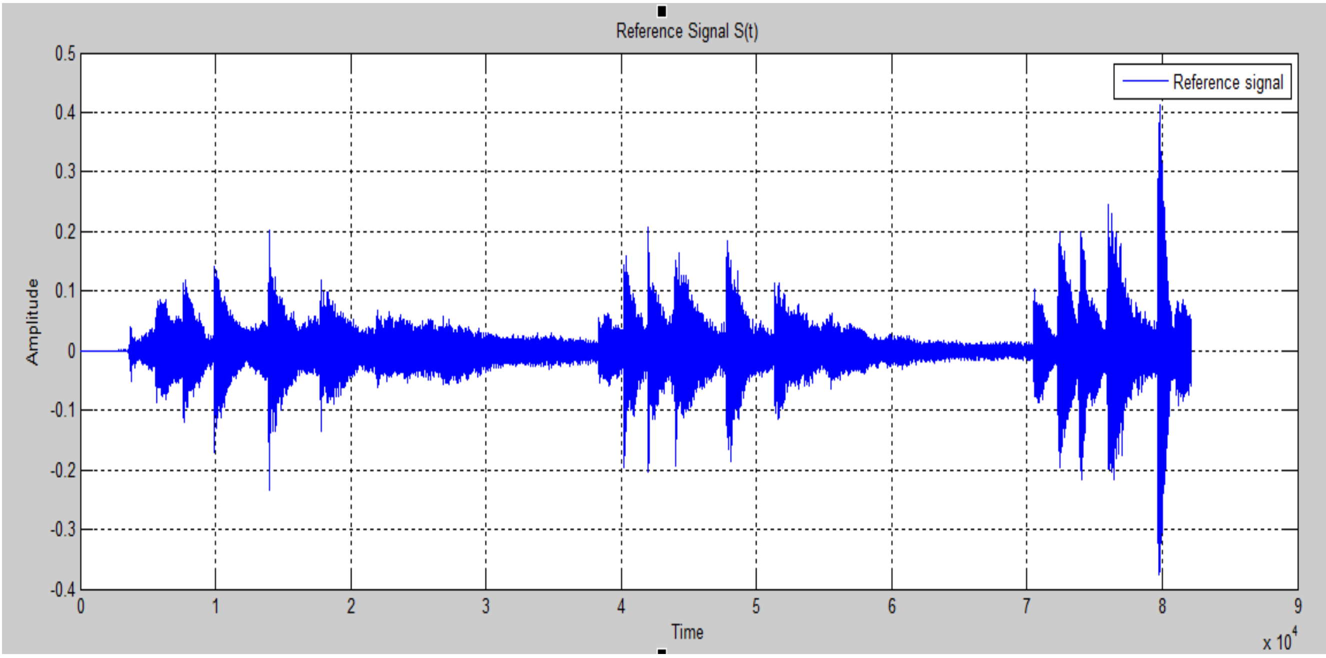


*Figure 4.1: Reference signal*

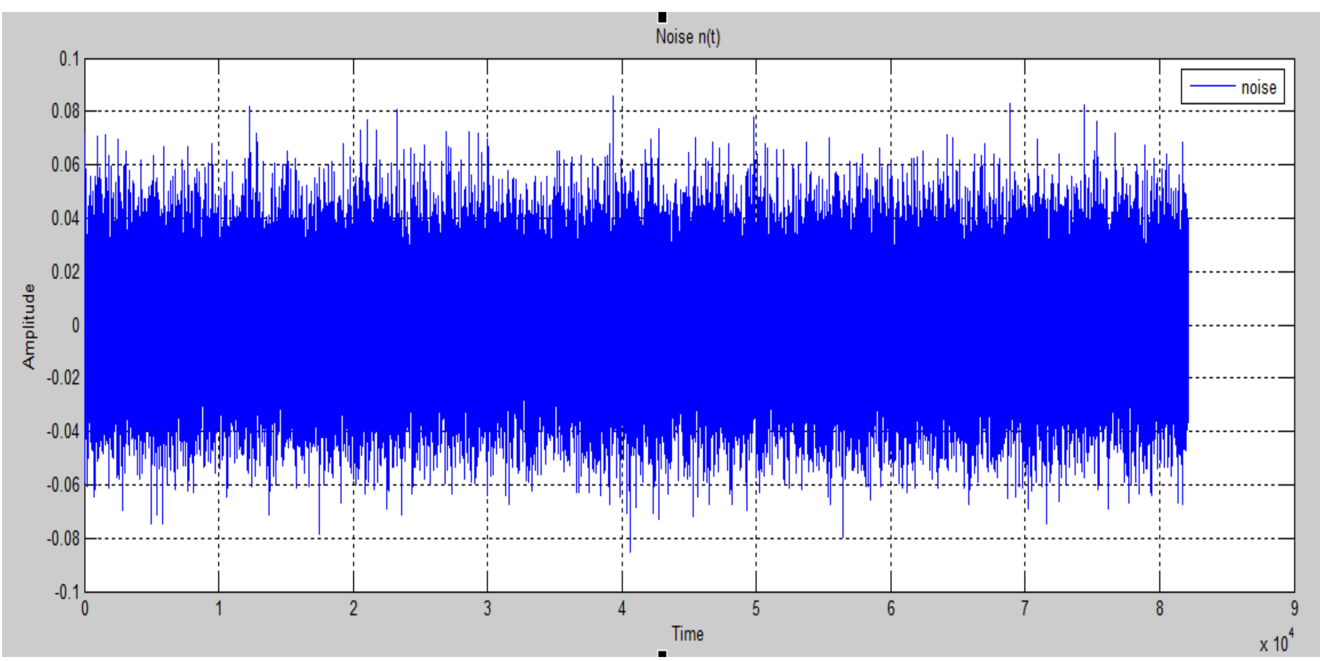


*Figure 4.2: Noise signal*

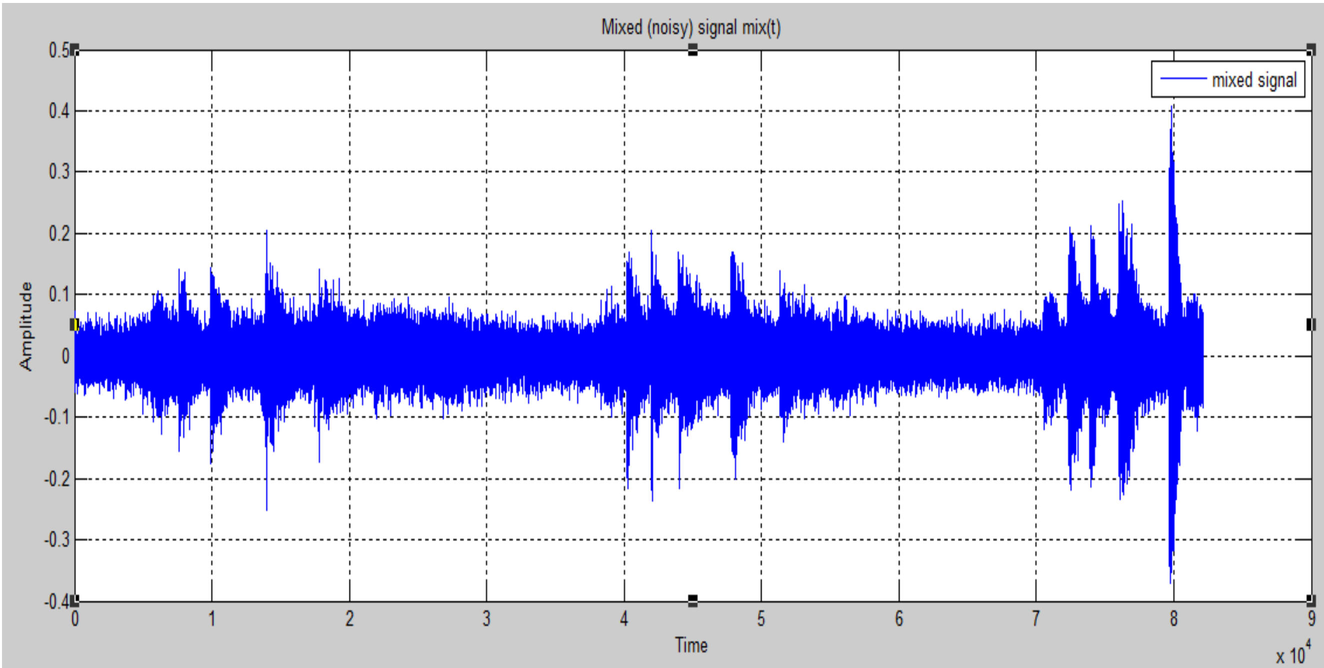


*Figure 4.3: Mixed signal*

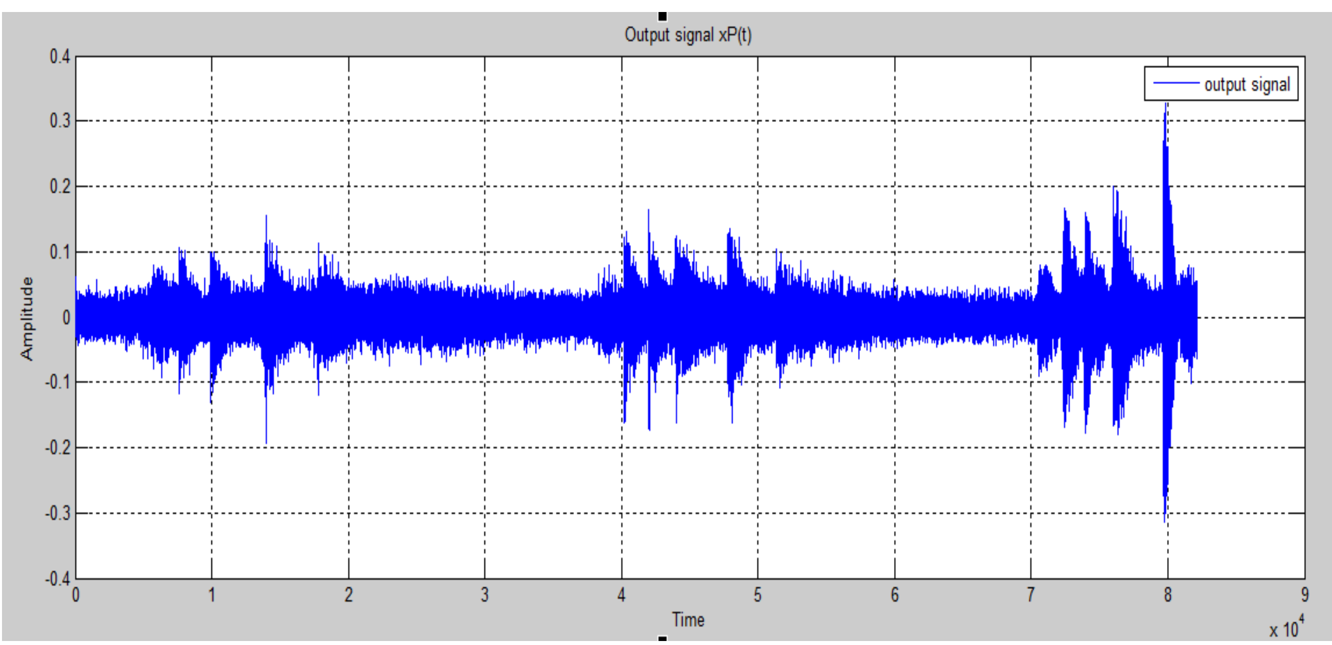


*Figure 4.4: Output signal*

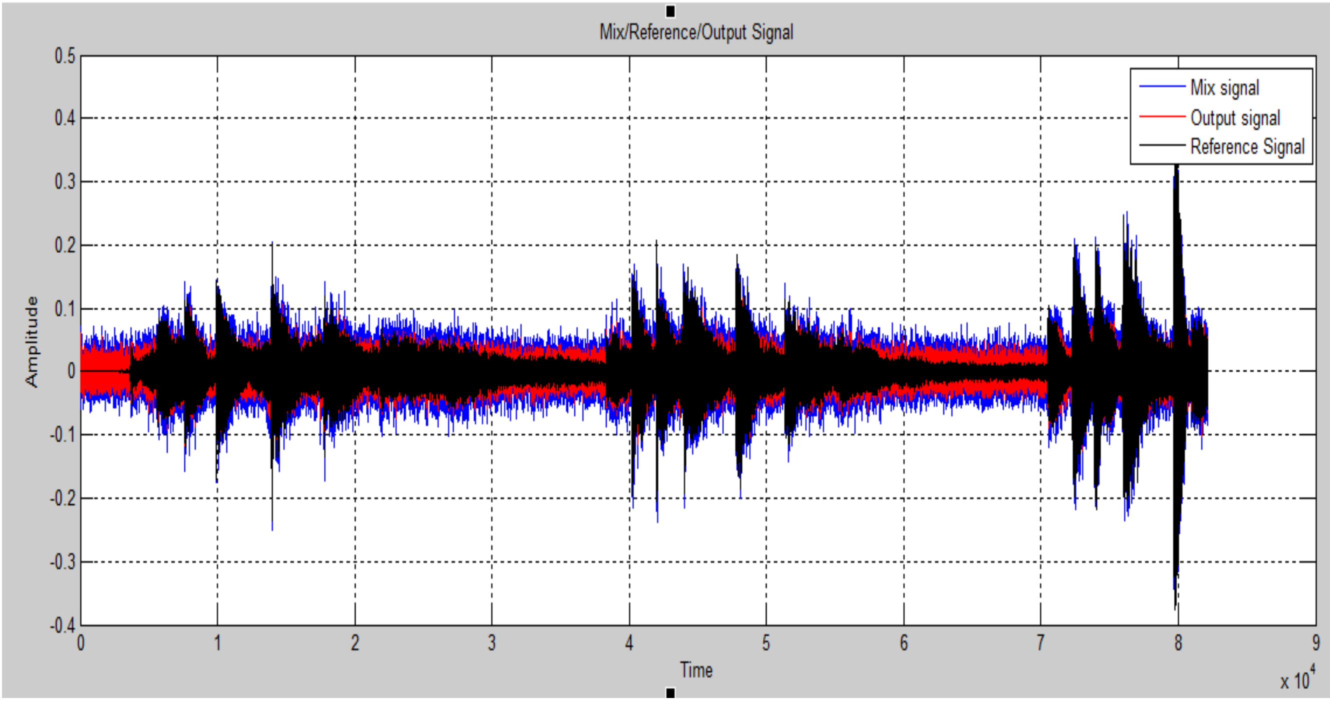


*Figure 4.5: Mix/Reference/Output signal( order 3)*

### 4.1.1.2 FIR model order 6:

The higher the order of the filter, the better performance it will get. By increasing the number of tap delay, we design the order 6th filter.

Similar to order 3 experiment, the graph below show the difference between the output signal and the reference signal as well as noisy signal.

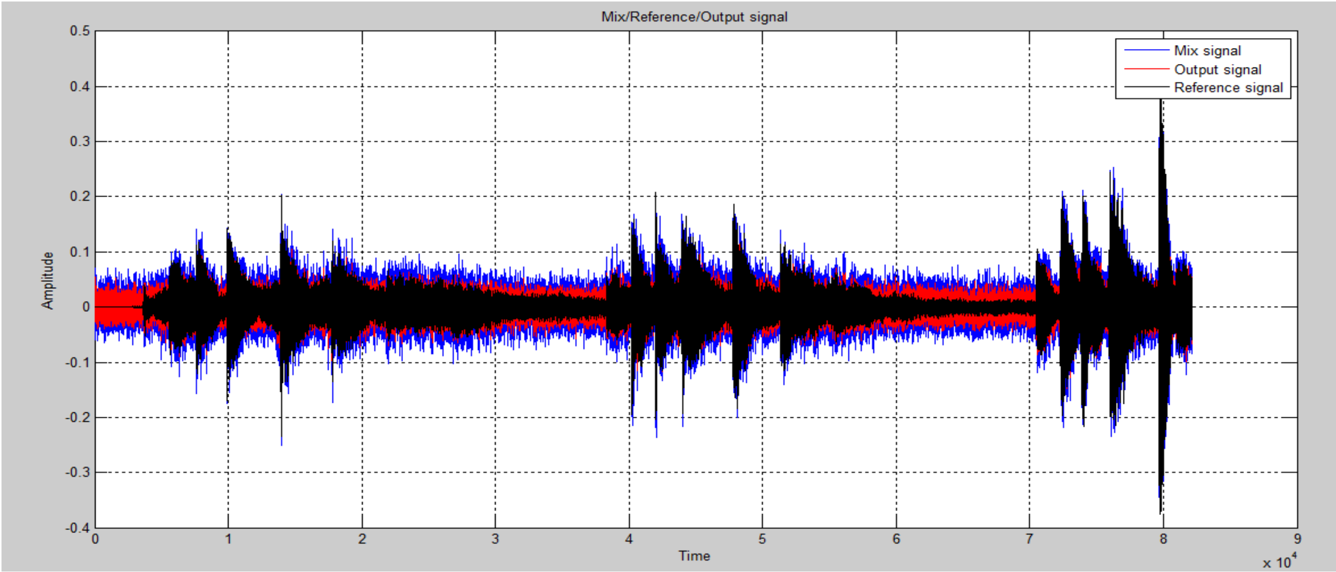


*Figure 4.6: Mix/Reference/Output signal (order 6)*

#### 4.1.1.3 FIR model order 10:

Increase the number of tap delay to 10, we have the order 10th filter. The order 10th filter is expected to give the best result among 3 filters. Below is the graph for order 10 filter.

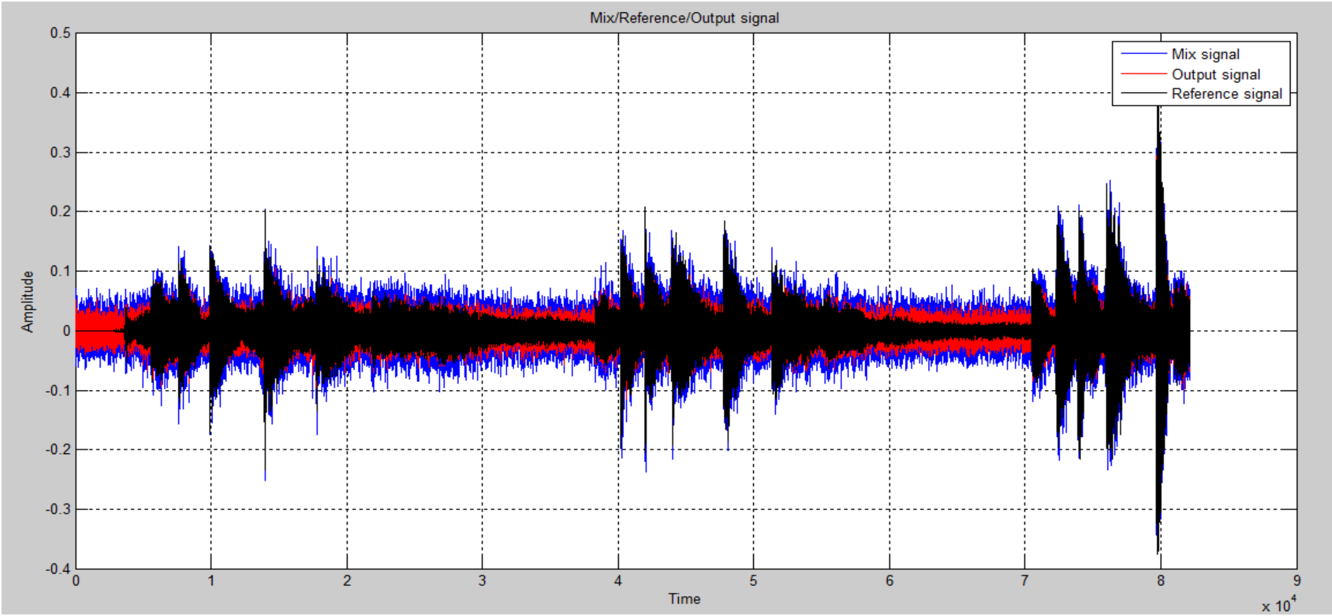


*Figure 4.7: Mix/Reference/Output signal (order 10)*

#### 4.1.1.4 Result comparison:

Looking at the graph we can see there is not much difference between the outputs from different order filters. More work need to be done to select the best order for the filtering. By using the different type of error estimation (Appendix II), we can decide which model gives the best prediction.

| Type of Error / Filter Order | Mean Squared Error | Root Mean Squared Error | Normalized Mean Squared Error |
|---|---|---|---|
| 3 | 6.7974 | 2.6072 | 0.2152 |
| 6 | 3.3100 | 1.8193 | 0.2096 |
| 10 | 1.9188 | 1.3852 | 0.2025 |

*Table 4.1: Result comparison*

### 4.1.2 Adaptive filter's result applying LMS Algorithms and Regularization technique:

For the online mode, we implement two basic techniques to improve the output result: Regularization and LMS algorithm. There is no doubt that the output signal will get closer to the reference signal compare to the offline mode.

#### 4.1.2.1 Filtering result with Regularization technique:

By comparing the Mean Squared Error before and after using the regularization, we can see the effect of this technique in correcting the output. It is expected that the higher the order of the filter, the more the technique work.

| Filter Order | MSE ( Without Regularization) | MSE ( With Regularization) |
|---|---|---|
| 3 | 6.7974 | 6.8019 |
| 6 | 3.3100 | 3.3007 |
| 10 | 1.9188 | 1.471 |

*Table 4.2: MSE in filter with regularization*

### 4.1.2.2 Filtering result with LMS Algorithm:

When we apply the LMS algorithm, the coefficient will be updated continuously until the output xP converge to the optimum value. The step size is chosen to be 0.05. The drawback of the LMS is that it provide a good output but cannot guarantee the stable performance.

Show here is the combination of all 3 signals (mix, reference and output signals). We can see how much improvement the filtering work has gain.

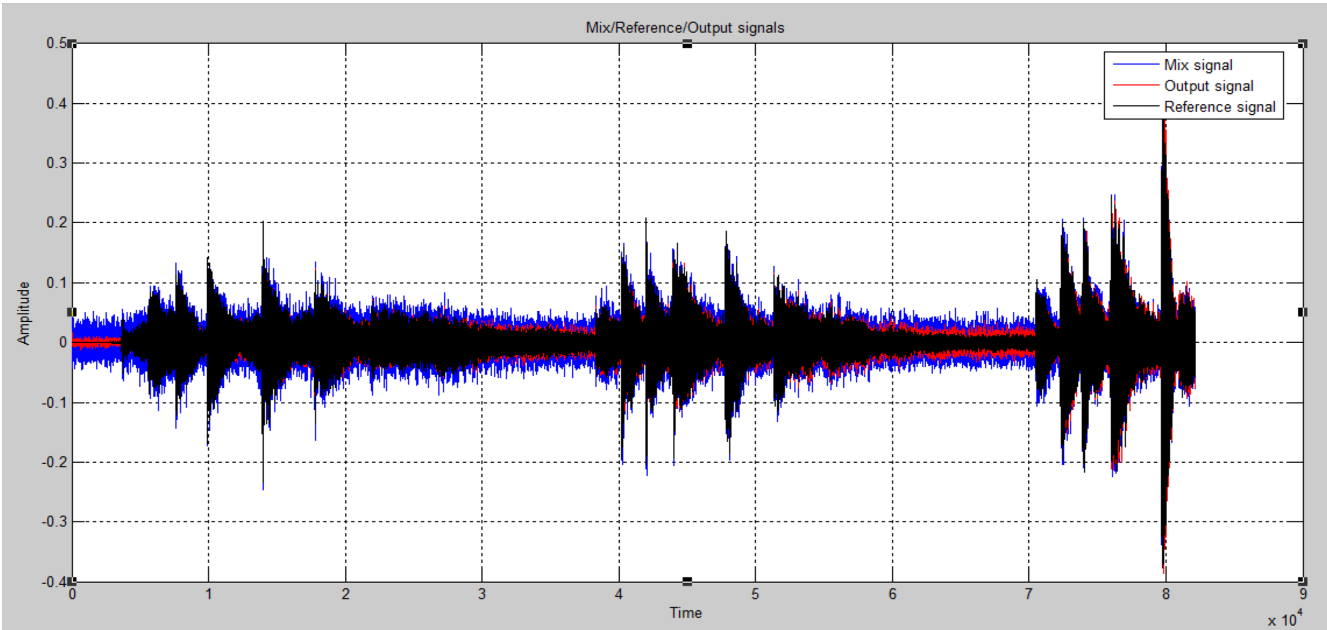


*Figure 4.8: Mix/Reference/Output signal (with LMS)*

For the clearer observation on the improvement of the filter, we also calculate the accuracy with the error estimation techniques (See appendix II). The results are shown below:

| Type of Error / Filter Order | Mean Squared Error | Root Mean Squared Error | Normalized Mean Squared Error |
|---|---|---|---|
| 3 | 6.6989 | 2.5882 | 0.2121 |
| 6 | 2.7736 | 1.6654 | 0.1756 |
| 10 | 1.1055 | 1.0514 | 0.1167 |

*Table 4.3: Error Estimation in adaptive filter with LMS algorithm*

## 4.2 Noise cancellation without reference signal.

Cancelling out the noise without the reference signal is a big challenge. Without the prior knowledge of the characteristic of the original signal, we only can estimate the form of the signal by reducing the error

### 4.2.1 Adaptive filter's result with NLMS algorithm:

The disadvantage of the LMS algorithm is that it is not easy to choose the step-size for the stable performance. Compare to LMS, NLMS will provide more stable and accurate results. However the algorithms is more complex. Following is the result of the filter using NLMS:

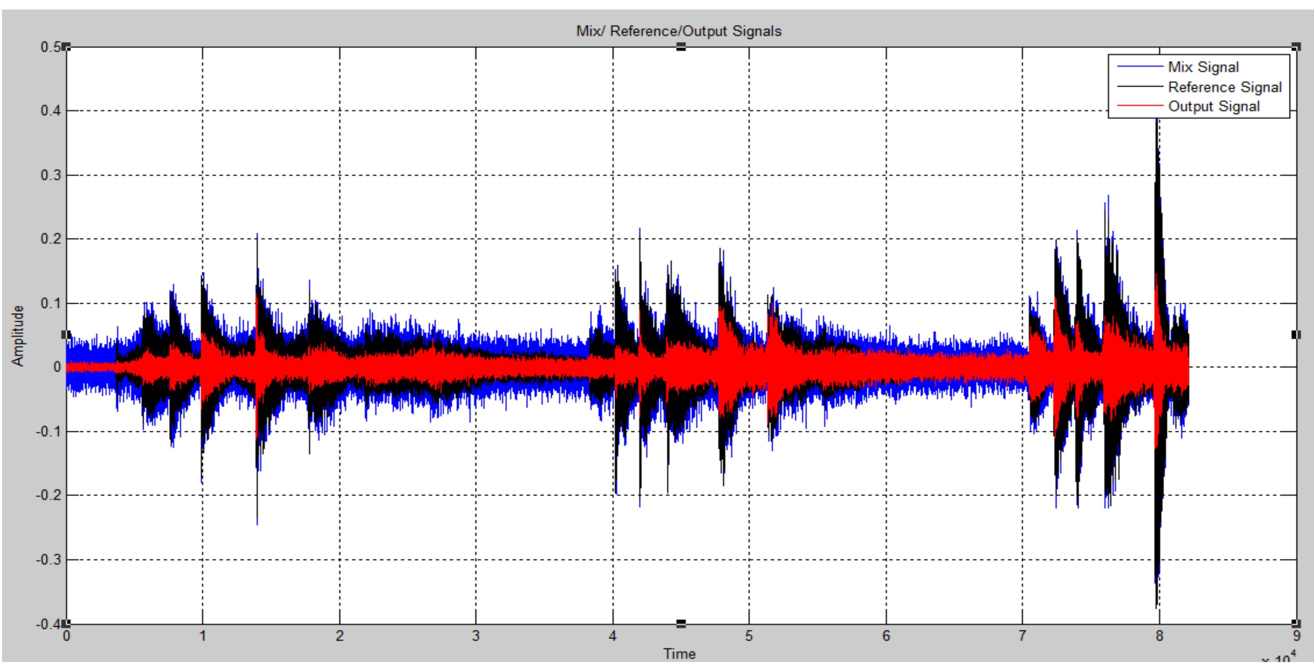


*Figure 4.9: Mix/Reference/Output signal (without reference signal)*

### 4.2.2 Adaptive filter's result with ARMA model:

ARMA model is a very effective tool in understanding and predicting the behavior of a system. We implement the ARMA model here to check if it can recover the form of the original signal by filter out the noise. Following is the graph of the filter's output:

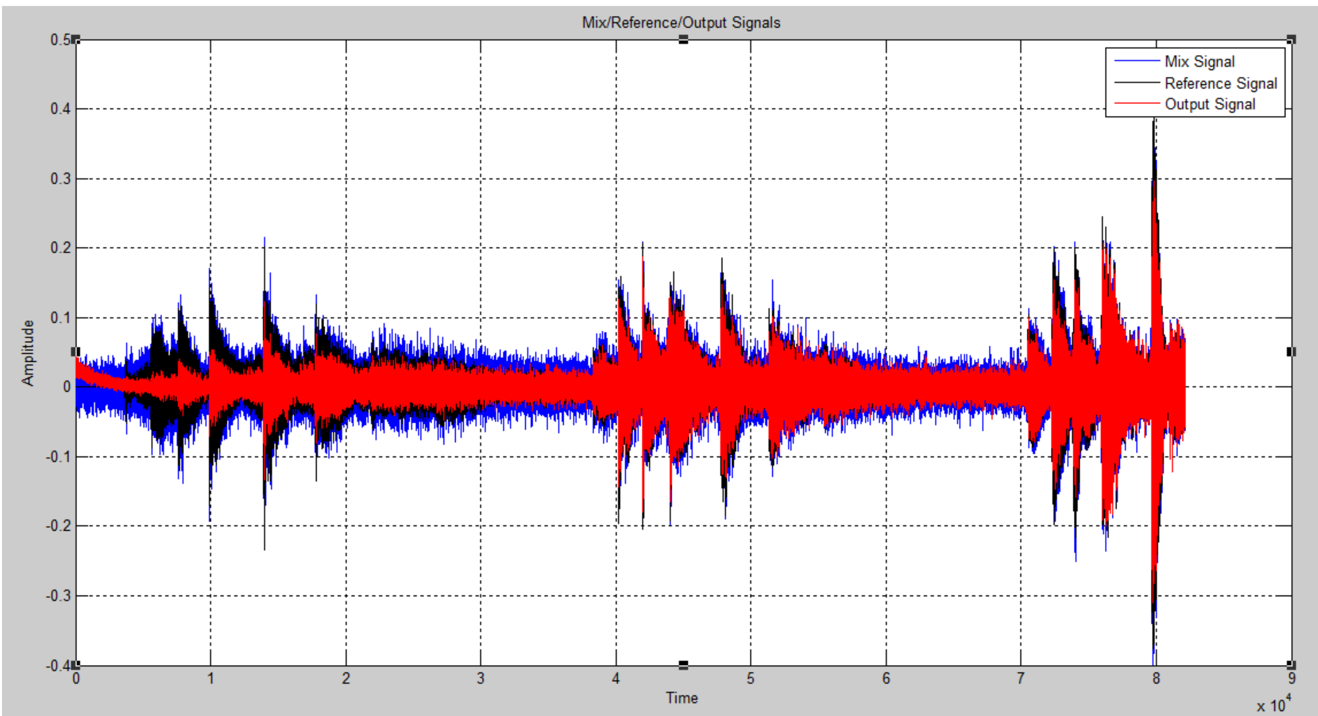


*Figure 4.10: Mix/Reference/Output signal (ARMA model)*

We can see that the form of the output signal is very similar to the original signal. To check how different between the original audio signal and the filter's output, we tabulate the errors estimation of the work with and without ARMA technique.

| Filter \ Error | Mean Squared Error | Root Mean Squared Error | Normalized Mean Squared Error |
|---|---|---|---|
| Adaptive filter without Reference | 8.6775 | 2.9458 | 0.9161 |
| Adaptive filter without Reference (With ARMA modelling technique) | 2.4333 | 1.5599 | 0.2568 |

*Table 4.4: Error Estimation in adaptive filters without reference*

## 4.3 Result Discussion:

For the basic filter design without adaptive technique, according to table 4.1, order 10 results in the minimum error for all the estimation. It means it will provide the best output which is closest to the reference clean signal. It makes a lot of sense since the higher the order of filter, the more accurate the filter is. According to the graphs, the work has brought some results. It can be clearly observed from the figure 4.5 that the noise is cancel out when mix signal pass through the filter. However, the amount of noise remaining is still high. In order to effectively eliminate the noise, the feedback loop needs to be implemented for coefficient adapting process.

When we apply the regularization in the design, the result is improved. Base on the table 4.2, with order 3 and 6 there are not much enhancement. But it work very well for order 10. The result is significantly improved when we apply the Least Mean Square Algorithm. Observing the graph 4.8, we can see that a lot of noise has been filter out, the red signal which is the output is very close to the reference signal (black). Moreover, based on the error estimation in table 4.3, the difference between the clean signal and the output from the filter are very small, especially with the high order (order 10). In this case the LMS work really well in cancelling the noise and recover the original signal.

For the work without reference signal, without the ARMA model, the output from adaptive filter can have shape of the original signal. However, there are a lot of errors. From figure 4.9, we can see that the results is improved a lot when we use ARMA technique. The output signal is very similar to the original. The error also reduces significantly according to table 4.4. It proves that ARMA model is the suitable technique for noise cancellation without references.

# Chapter V CONCLUSION & RECOMMENDATION

## 5.1 Conclusion:

Due to the advantage of adaptive filter, various applications of adaptive filter are studied and widely used in many industries, especially audio interference removal, voice signal processing. Using adaptive filter to eliminate the noise and bring back the clean signal has been done for several decades. Besides choosing the suitable configuration, filter need to be design with the correct algorithm to bring out the best result.

The work from the Matlab simulation has given out specific result. For the first part of the project which is cancelling the noise with reference signal, with the support of the adaptive algorithms, the final output of the filter is very close to the clean original signal, mean that the big portion of the noise has been canceled out. For the work without the reference signal, we employ the ARMA model to achieve the best output.

## 5.2 Recommendation for future work:

As stated above, for the improvement of the project, some of other adaptive algorithms as well as filter configuration can be employed to bring more stable performance and the better results.

We can also integrate the simulation with the real signal processing hardware to test in real time system.

# *REFERENCES:*

[1] A. Singh, "Adaptive Noise Cancellation,", thesis, Electron. & Commun., Netaji Subhas Inst., 2001.

[2] T. Drumright, "Adaptive Filtering," 1998.

[3] G. Gui, W. Peng, and F. Adachi, "Adaptive system identification using robust LMS/F algorithm" *International Journal of Communication Systems*. Available: http://onlinelibrary.wiley.com/doi/10.1002/dac.2517/abstract

[4] D. R. Morgan and S. Craig, "Real-time adaptive linear prediction using the least mean square gradient algorithm," *Acoustics, Speech and Signal Processing, IEEE Transactions on,* vol. 24, pp. 494-507, 1976.

[5] L. Guo, L. Ljung, and W. Guan-Jun, "Necessary and sufficient conditions for stability of LMS," *Automatic Control, IEEE Transactions on,* vol. 42, pp. 761-770, 1997.

[6] F. Da-Zheng, B. Zheng, and J. Li-Cheng, "Total least mean squares algorithm," *Signal Processing, IEEE Transactions on,* vol. 46, pp. 2122-2130, 1998.

[7] C. Rui, K. Shaoli, and M. Reza Asharif, "Adaptive Filter of Extended Correlation Least Mean Squared Algorithm Based on Tap-Length Variable Method," in *Information Science and Engineering (ICISE), 2009 1st International Conference on*, pp. 472-476, 2009.

[8] J. Gunther, T. Bose, and S. Wang, "Filter Bank Design for Minimizing Mean-Squared Estimation Error in Subband Adaptive Filtering," in *Signals, Systems and Computers, 2005. Conference Record of the Thirty-Ninth Asilomar Conference on*, pp. 535-539, 2005.

[9] A. Faza, S. Grant, and J. Benesty, "Adaptive regularization in frequency-domain NLMS filters," in *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, 2012, pp. 2625-2628.

[10] J. Benesty, C. Paleologu, and S. Ciochina, "On Regularization in Adaptive Filtering," *Audio, Speech, and Language Processing, IEEE Transactions on,* vol. 19, pp. 1734-1742, 2011.

[11] C. Janeczko and H. S. Lopes, "A genetic approach to ARMA filter synthesis for EEG signal simulation," in *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, pp. 373-378 vol.1, 2000.

[12] V. L. Stonick and D. Massie, "ARMA filter design for music analysis/synthesis," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, pp. 253-256 vol.2, 1992.

# APPENDICES

## Appendix I: Least Mean Squared Algorithms

The least mean squares (LMS) algorithms are the techniques that adjust the coefficients of the adaptive filter to minimize the cost function. As the result, the output from the filter will converge into to reference input.

We have the following scheme:

Input vector: $\vec{x}(n)$

Output: y(n)

Reference Input (desired output): s(n)

Step size of adaptive filter: $\mu$

Filter coefficient vector: $\vec{w}(n)$

Coefficient updated vector: $\vec{w}(n+1)$

The LMS algorithms are perform by following steps:

_ Output from filter: $y(n) = \vec{w}^T(n).\vec{x}(n)$

_ Calculating the different between the desired output and the real output, error: $e(n) = s(n) - y(n)$

_ Updating the filter coefficients: $\vec{w}(n+1) = \vec{w}(n) + 2\mu e(n)\vec{x}(n)$

The LMS algorithms have not only provided a simple calculating but also given out the good and stable result.

## Appendix II: Error Estimation Techniques:

Mean Square Error (MSE): $$MSE = \frac{1}{n}\sum_{t=1}^{n}(x_t - \widehat{x}_t)^2$$

Root Mean Square Error (RMSE): $$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(x_t - \widehat{x}_t)^2}$$

Normalized Mean Square Error (NMSE): $$NMSE = \frac{\sum_{t=1}^{n}(x_t - \widehat{x}_t)^2}{\sum_{t=1}^{n}(x_t - \overline{x}_t)^2}$$

With: $x_t$ is the actual value at time t

$\widehat{x}_t$ is the forecasting value at time t

n is the forecasting period

$\overline{x}_t$ is the mean of $x_t$

# Appendix III: Matlab Code:

## 1. Offline mode ( Fixed filter)

```
wavread('D:\Long\song.wav');
r=wavread('D:\Long\song.wav');
s=r(:,1);
sa=audioplayer(s,8000);
m=randn(82129,1);
n=m/50;
mix=s+n;
mixa=audioplayer(mix,8000);
wavwrite(mix,'D:\Long\mix.wav');

xt=mix(11:82129);
xt_1=mix(10:82128);
xt_2=mix(9:82127);
xt_3=mix(8:82126);
xt_4=mix(7:82125);
xt_5=mix(6:82124);
xt_6=mix(5:82123);
xt_7=mix(4:82122);
xt_8=mix(3:82121);
xt_9=mix(2:82120);
xt_10=mix(1:82119);
Xt=[xt_1 xt_2 xt_3 xt_4 xt_5 xt_6 xt_7 xt_8 xt_9 xt_10 ];
s_1=s(1:82119);

A = pinv(Xt)*s_1;
xP = A(1)*xt_1 + A(2)*xt_2 + A(3)*xt_3 + A(4)*xt_4 + A(5)*xt_5 + A(6)*xt_6 + A(7)*xt_7 + A(8)*xt_8 +
A(9)*xt_9 + A(10)*xt_10;
xPa=audioplayer(xP,8000);

1/10*(sum((s_1-xP).^2))
sqrt(1/10*(sum((s_1-xP).^2)))
(sum((s_1-xP).^2))/(sum((s_1-mean(s_1)).^2))

plot(mix);
hold
plot(xP,'r');
plot(s,'k');
```

## 2. Online mode ( Adaptive filter) with reference

```
wavread('D:\Long\song.wav');
r=wavread('D:\Long\song.wav');
s=r(:,1);
sa=audioplayer(s,8000);
m=randn(82129,1);
n=m/70;
mix=s+n;
mixa=audioplayer(mix,8000);
wavwrite(mix,'D:\Long\mix.wav');
```

```
xt=mix(11:82129);
xt_1=mix(10:82128);
xt_2=mix(9:82127);
xt_3=mix(8:82126);
xt_4=mix(7:82125);
xt_5=mix(6:82124);
xt_6=mix(5:82123);
xt_7=mix(4:82122);
xt_8=mix(3:82121);
xt_9=mix(2:82120);
xt_10=mix(1:82119);
Xt=[xt_1 xt_2 xt_3 xt_4 xt_5 xt_6 xt_7 xt_8 xt_9 xt_10 ];
s_1=s(1:82119);

A = (inv(Xt'*Xt) + 0.001*eye(10))*Xt'*s_1
xP = A(1)*xt_1 + A(2)*xt_2 + A(3)*xt_3 + A(4)*xt_4 + A(5)*xt_5 + A(6)*xt_6 + A(7)*xt_7 + A(8)*xt_8 +
A(9)*xt_9 + A(10)*xt_10;
xPa=audioplayer(xP,8000);

k=size(Xt);
beta=0.1*rand(k(2),1);
lamd =0.05;
for i=1:k(1)-1
    err=s_1(i)-xP(i);
    beta=beta+(lamd*Xt(i,:)'*err);
    xP(i+1)=Xt(i,:)*beta;
end

1/10*(sum((s_1-xP).^2))
sqrt(1/10*(sum((s_1-xP).^2)))
(sum((s_1-xP).^2))/(sum((s_1-mean(s_1)).^2))

plot(mix);
hold
plot(xP,'r');
plot(s,'k');
```

### 3. Online mode ( Adaptive filter) without reference

```
noise=randn(82129,1);
noise3=noise/70;
wavread('D:\Long\song.wav');
r=wavread('D:\Long\song.wav');
s=r(:,1);
sa=audioplayer(s,8000);
s_1=s(1:82100);
mix13=s+noise3;
wavwrite(mix13,'D:\Long\mix13.wav');

xt=mix13(30:82129);
xt_1=mix13(29:82128);
xt_2=mix13(28:82127);
xt_3=mix13(27:82126);
xt_4=mix13(26:82125);
xt_5=mix13(25:82124);
xt_6=mix13(24:82123);
xt_7=mix13(23:82122);
```

```
xt_8=mix13(22:82121);
xt_9=mix13(21:82120);
xt_10=mix13(20:82119);
Xt=[xt_1 xt_2 xt_3 xt_4 xt_5 xt_6 xt_7 xt_8 xt_9 xt_10];

k=size(Xt);
beta=0.1*rand(k(2),1);
xP=Xt*beta;
lamd =0.0008;
for i=1:k(1)-1
    err=mix13(i)-xP(i);
    beta=beta+(lamd*Xt(i,:)'*err)/((norm(Xt(i,:))+0.0000000000000001)^2);
    xP(i+1)=Xt(i,:)*beta;
end

1/10*(sum((s_1-xP).^2))
sqrt(1/10*(sum((s_1-xP).^2)))
(sum((s_1-xP).^2))/(sum((s_1-mean(s_1)).^2))

plot(mix13)
hold
plot(s,'k')
plot(xP,'r')
```

## 4. Online mode ( Adaptive filter with ARMA model) without reference

```
wavread('D:\Long\song.wav');
r=wavread('D:\Long\song.wav');
s=r(:,1);
sa=audioplayer(s,8000);
m=randn(82129,1);
n1=m/70;
mix7=s+n1;
mixa=audioplayer(mix7,8000);
wavwrite(mix7,'D:\Long\mix7.wav');

s_1=s(1:82119);
 err=0.1*rand(82119,10);

xt=mix7(11:82129);
xt_1=mix7(10:82128);
xt_2=mix7(9:82127);
xt_3=mix7(8:82126);
xt_4=mix7(7:82125);
xt_5=mix7(6:82124);
xt_6=mix7(5:82123);
xt_7=mix7(4:82122);
xt_8=mix7(3:82121);
xt_9=mix7(2:82120);
xt_10=mix7(1:82119);
Xt=[xt_1 xt_2 xt_3 xt_4 xt_5 xt_6 xt_7 xt_8 xt_9 xt_10 err];

k=size(Xt);
beta=0.1*rand(k(2),1);
xP=Xt*beta;

lamd =0.00008;
```

```
for i=1:k(1)-1
    for j=1:10
    err(i,j)=mix7(i,1)-xP(i,1);
    beta=beta+(lamd*Xt(i,:)'*err(i,j))/((norm(Xt(i,:))+0.0000000000000001)^2);
    xP(i+1)=Xt(i,:)*beta;
    end
 end

1/10*(sum((s_1-xP).^2))
sqrt(1/10*(sum((s_1-xP).^2)))
(sum((s_1-xP).^2))/(sum((s_1-mean(s_1)).^2))

plot(mix7);
hold
plot(s,'k');
plot(xP,'r');
```