ELE 659

Midterm Project
ANC Filtered X LMS Algorithm

Mustafa Oztoprak
Z1883688

Contents

1. Introduction

Active Noise Cancellation (ANC) is a method for reducing undesired noise. ANC is achieved by introducing a canceling antinoise wave through secondary sources. These secondary sources are interconnected through an electronic system using a specific signal processing algorithm for the particular cancellation scheme. The project is to write an algorithm in matlab for single channel feed-forward active noise control system based on the FxLMS. This report will demonstrate the approaches that we take on tackling the noise cancellation effects, along with results comparison.

1.1 Basic Concepts

Noise Cancellation makes use of the notion of destructive interference. When two sinusoidal waves superimpose, the resulting waveform depends on the frequency amplitude and relative phase of the two waves. If the original wave and the inverse of the original wave encounter at a junction at the same time, total cancellation occur. The challenges are to identify the original signal and generate the inverse without delay in all directions where noises interact and superimpose. We will demonstrate the solutions later in the report

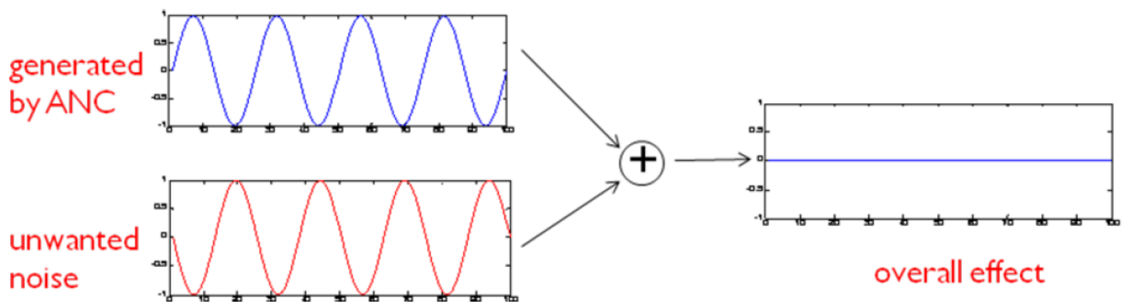

Figure 1: Signal Cancellation of two waves 180° out of phase.

2. Layout

In figure 2, the left model is our headset model. The circle symbolizes a microphone. There are two microphones on the headset, one outside the headset and the other inside the headset. The outside microphone called the reference microphone is used to measure the noise near the headset which is inputted to the DSK. DSK adaptively tries to produce an inverse noise by processing the input. The inside microphone called the error microphone will perceive the error between the noise from the noise source and the inverse signal generated by the DSK. This error signal will be fed back to the DSK, which updates the filter coefficients based on this feedback. The right model is the equivalent model of the headset. The reason we transfer from the headset model to the acoustic duct model is the humungous amount of literature available in this model.

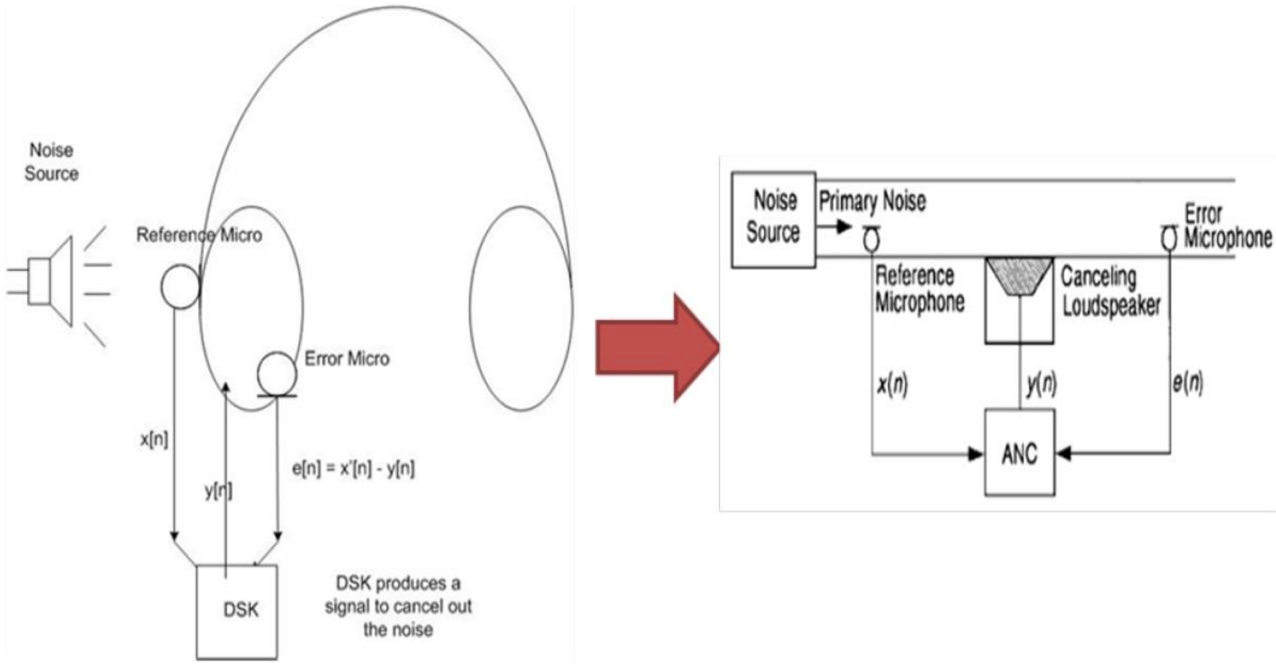

Figure 2: Layout of ANC Headset and its equivalent model

2.1 Adaptive Filter Framework

Since the characteristics of the acoustic noise source and the environment are time varying, the frequency content, amplitude, phase, and sound velocity of the undesired noise are nonstationary. An ANC system must therefore be adaptive in order to cope with these variations. Adaptive filters adjust their coefficients to minimize an error signal and can be realized as (transversal) finite impulse response (FIR), (recursive) infinite impulse response (IIR), lattice, and transform-domain filters. The most common form of adaptive filter is the transversal filter using the least mean-square (LMS) algorithm.

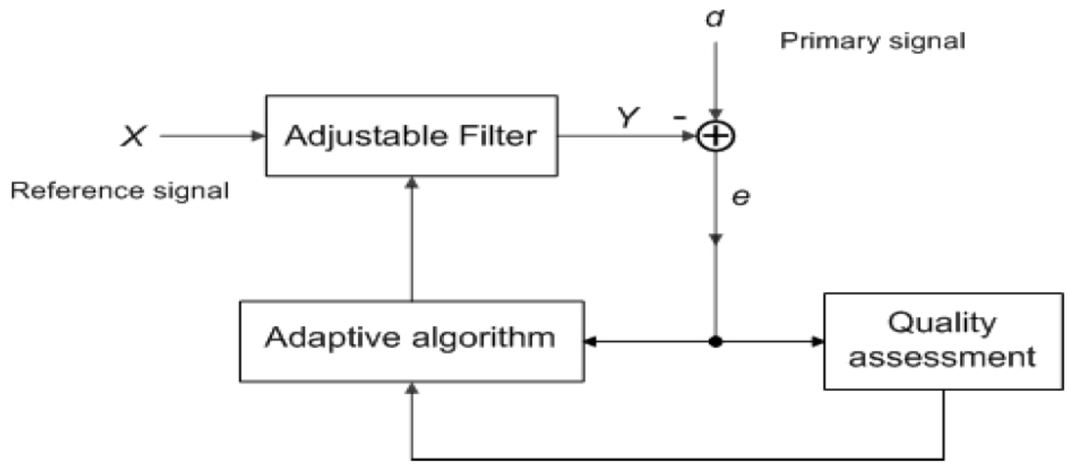

Figure 3: Adaptive Filter Framework

2.2 Basic outline of LMS and its variations

One of the main constraints in the choice of an adaptive algorithm is its computational complexity. For the application of ANC, it is desired to choose an algorithm which is computationally very fast. Taking this into consideration, LMS algorithm became an obvious choice. The update equation for the LMS algorithm is given by

$$w(n+1) = w(n) + \mu * e(n) * w(n)$$

where μ is the step size, $e(n)$ is the error at time n and $w(n)$ is the filter coefficients at time instant n.

2.3 FxLMS Algorithm

The basic LMS algorithm fails to perform well in the ANC framework. This is due to the assumption made that the output of the filter $y(n)$ is the signal perceived at the error microphone, which is not the case in practice. The presence of the A/D, D/A converters and anti aliasing filter in the path from the output of the filter to the signal received at the error microphone cause significant change in the signal $y(n)$. This demands the need to incorporate the effect of this secondary path function $S(z)$ in the algorithm. One solution is to place an identical filter in the reference signal path to the weight update of the LMS algorithm, which realizes the so-called filtered-X LMS (FXLMS) algorithm, see Figure 4.

The FXLMS Algorithm:

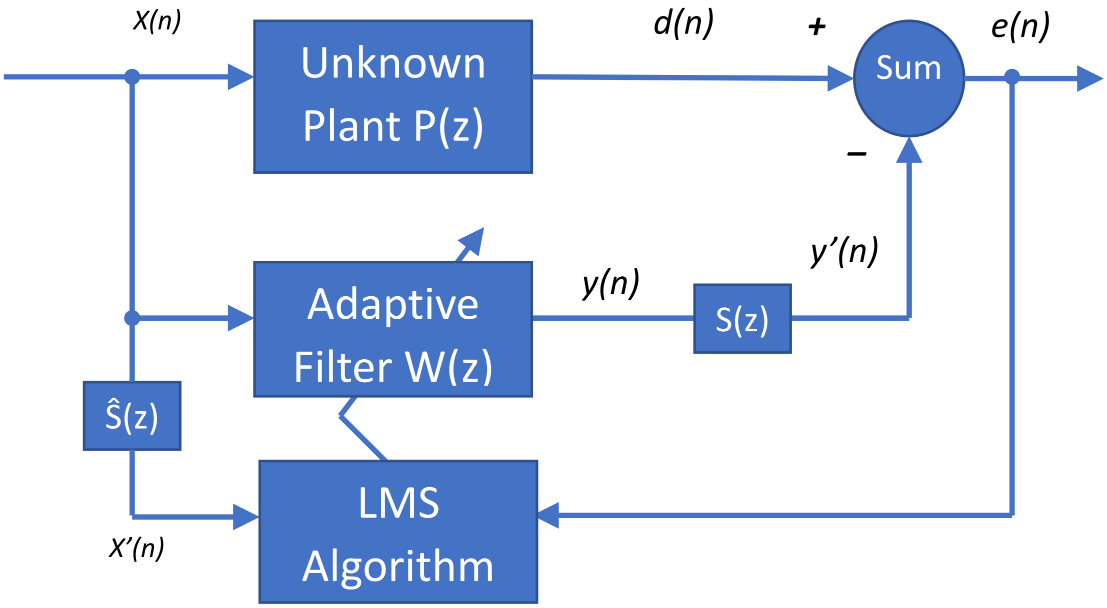

Figure 4:FxLms Algorithm Scheme

Imagine that the noise X(n) is propagating from the source to the sensor, through the fluid medium P(z). The sensor measures the arriving noise as d(n). To reduce noise, we generate another 'noise' y(n) using the controller W(z). We hope that it destructively interferes X(n). It means that the controller has to be a model of the propagation medium P(z). Least mean square algorithm is applied to adjust the controller coefficient/weight. However, there is also fluid medium S(z) that stay between the actuator and sensor. We called it the secondary propagation path. So, to make the solution right, we need to compensate the adjustment process using Ŝ(z) which is an estimate of S(z)

Equations:

y(n)= w^T (n)x(n)

y' (n)= $s(n) *$[w^T (n)x(n)]

$e(n) = d(n) - y'(n)$

$x'(n) = x(n)*s(n)$

$w(n+1)=w(n)+\mu e(n)x'(n)$

3. Programming Environment

Data Provided:

- 1. SEC13R.mat
- This can be used as a reference noise source x(n).
-
- 2. SEC18R.mat
- This can be used as a reference noise source x(n).
-
- 3. TF.mat
- This contains transfer function of primary path P(z) and secondary path S(z).
-
- P_z - coefficients of numerator of P(z)
- P_p - coefficients of denominator of P(z)
- (The leading coefficient is assumed to be equal to 1)
-
- S_z - coefficients of numerator of S(z)
- S_p - coefficients of denominator of S(z)
- (The leading coefficient is assumed to be equal to 1)

Procedure:

1. Offline training for secondary path estimation

Try different filter length and step size

Show the error e(n) vs. time (learning curve)

After converge, plot the magnitude response of your estimated S(Z) and compare it with the real secondary path

2. Online FXLMS

Try different filter length and step size

Show the error e(n) vs. time (learning curve)

After converge, plot the magnitude response of error signal e(n) and compare it with original noise d(n)

3.1 Approach 1: off-line estimation of S(z):

Figure 5 illustrates the block diagram of off-line estimation of S(z). We send a sequence of training data to estimate S(z) before the setup of noise cancellation. It's better to train the filter with white noise.

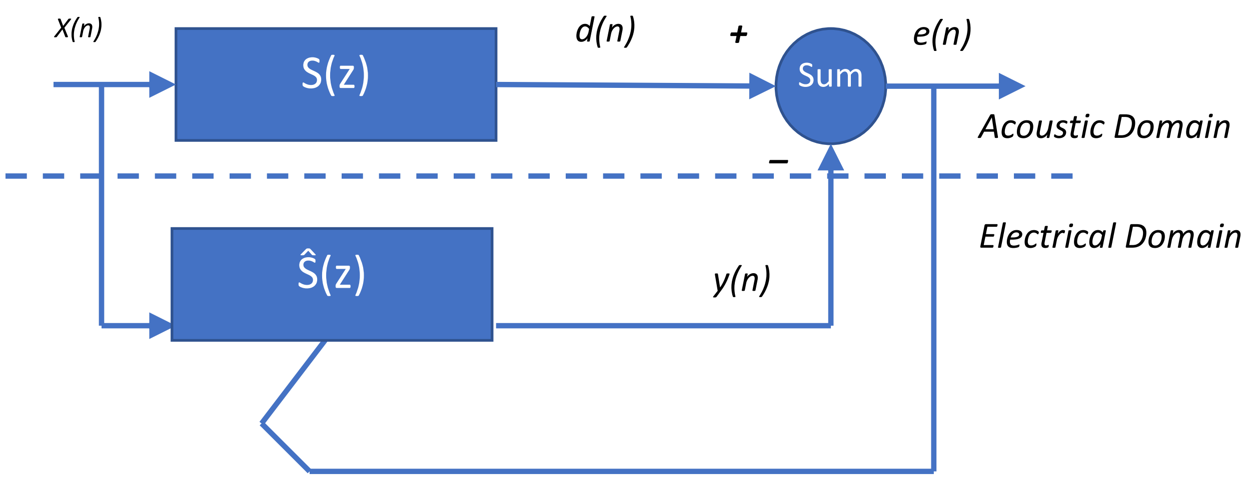

Figure 6: Block diagram of Approach 1: off-line estimation of S(z).

Off-line Training part codes and graphs:

```
% white noise signal,
x_noise=randn(1,T);
y_d=filter(S_z,S_p, x_noise);

%  the identification process
Shx=zeros(1,L);    % the state of Sh(z)
Shz=zeros(1,L);    % the weight of Sh(z)
e_z=zeros(1,T);  % data buffer for the identification error

%Applying least mean square algorithm

for k=1:T                   % discrete time k
   Shx=[x_noise(k) Shx(1:L-1)];  % update the state
   Shy=Shx*Shz';          % calculate output of Sh(z)
   e_z(k)=y_d(k)-Shy;   % calculate error
   Shz=Shz+mu*Shx*e_z(k);  % adjust the weight
end
```

mu = **0.006/*L*:100**

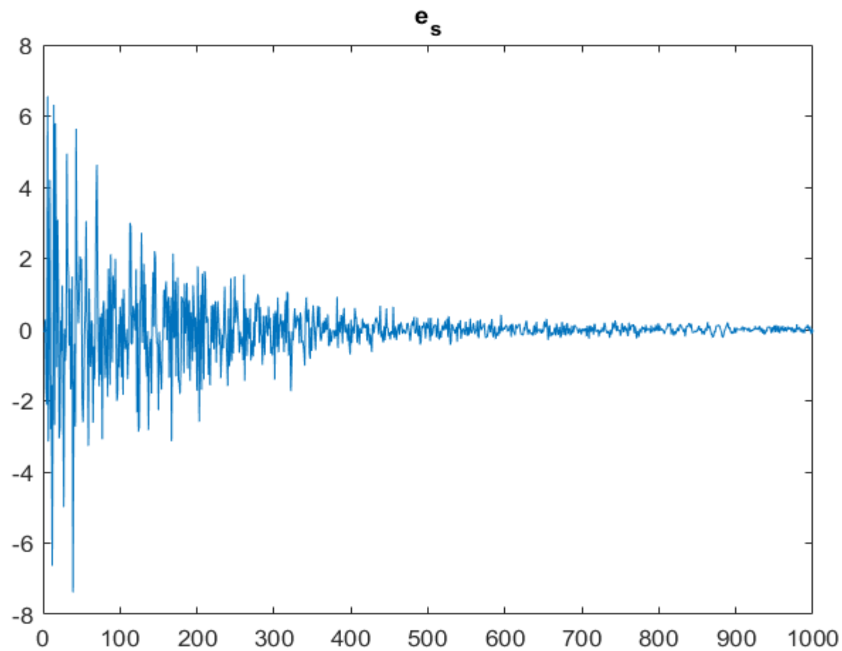

Figure 7: Error

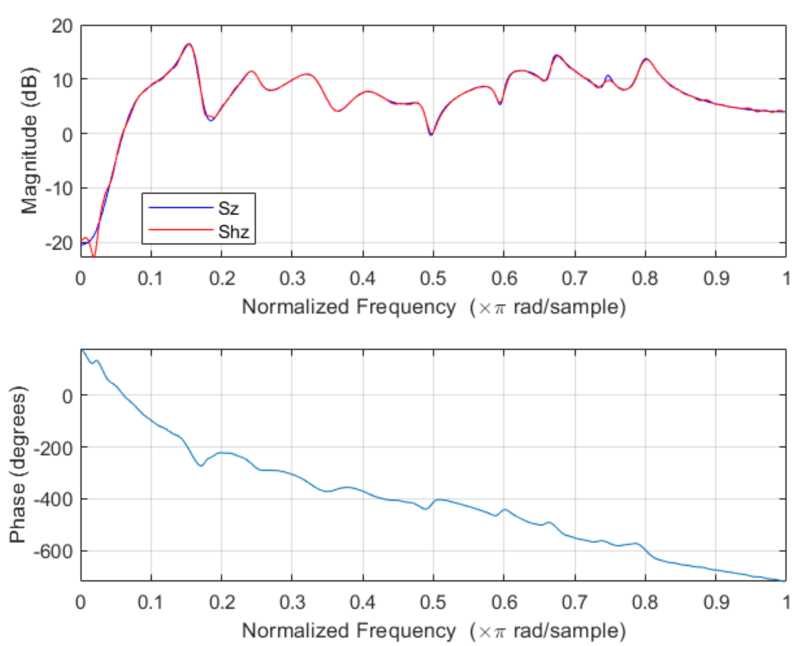

Figure 8: Magnitude Response Comparison of Sz and Shz

mu = **0.008/*L*:200**

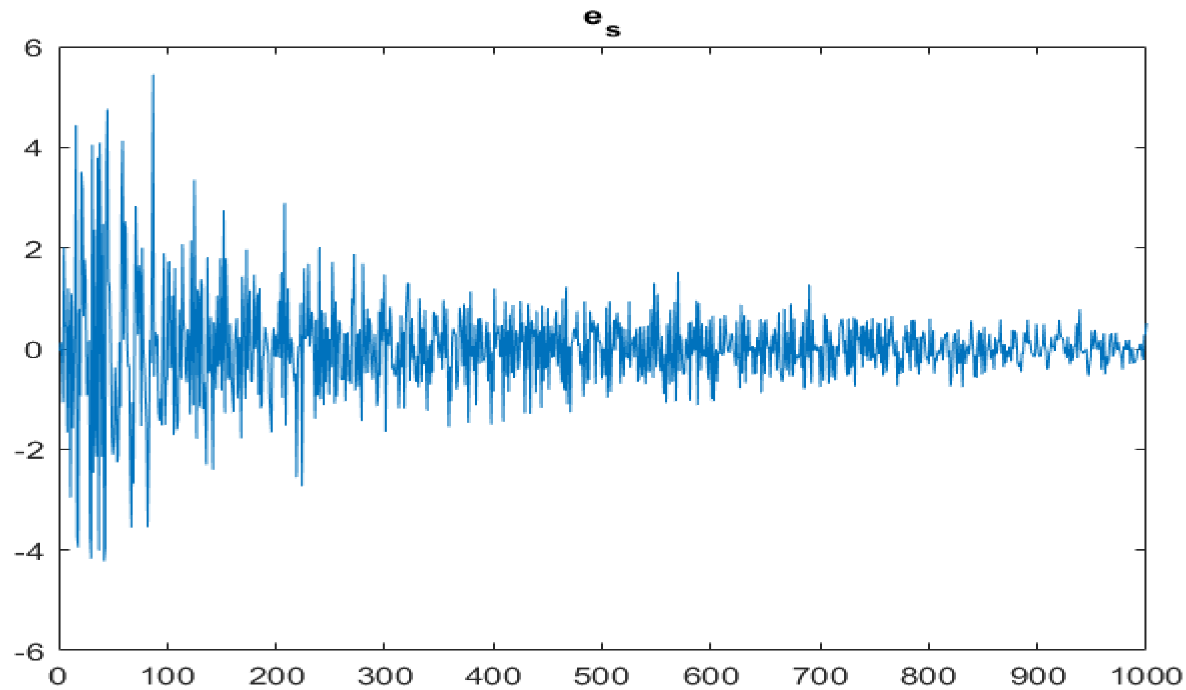

Figure 7: Error

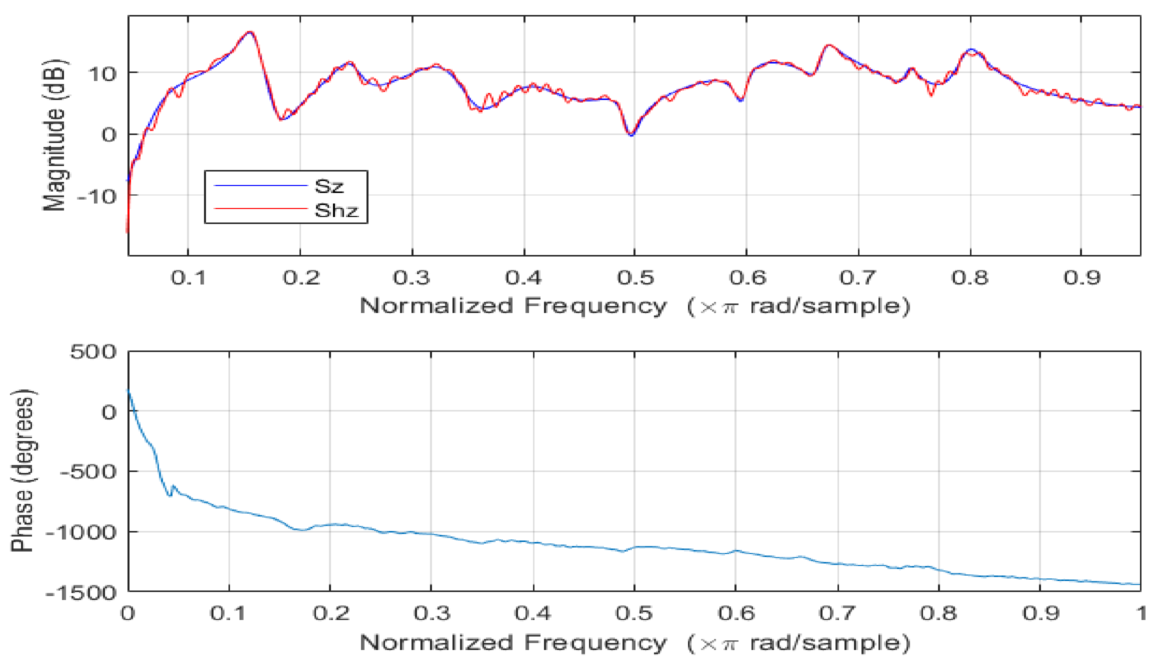

Figure 8: Magnitude Response Comparison of Sz and Shz

mu **= 0.001/*L*:100**

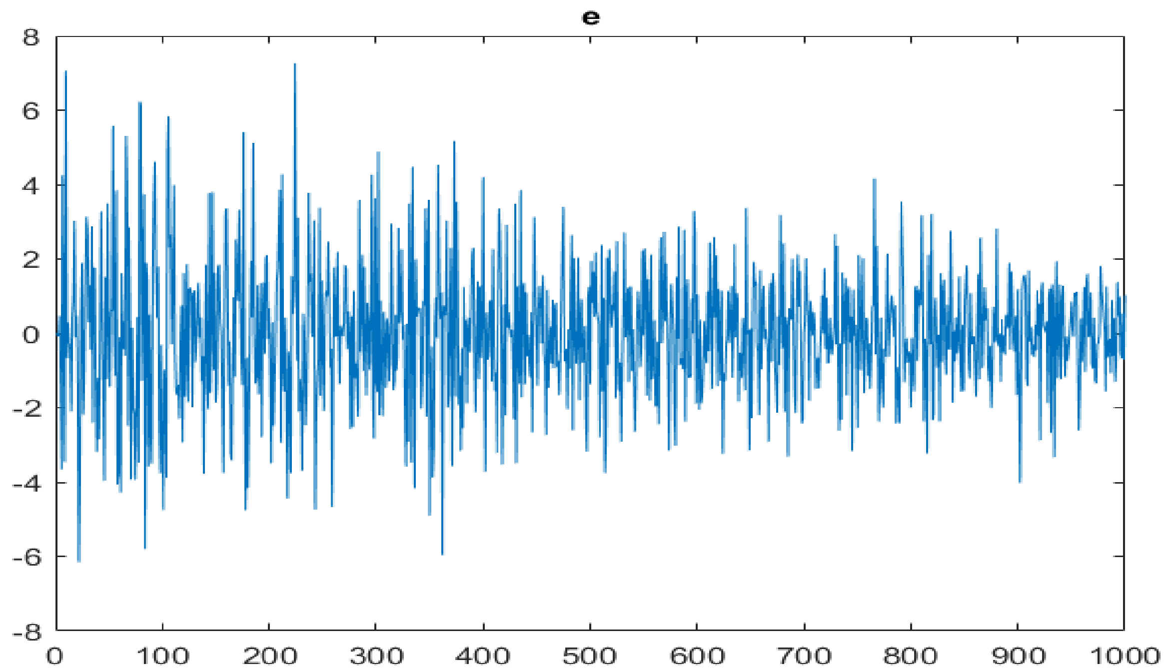

Figure 7: Error

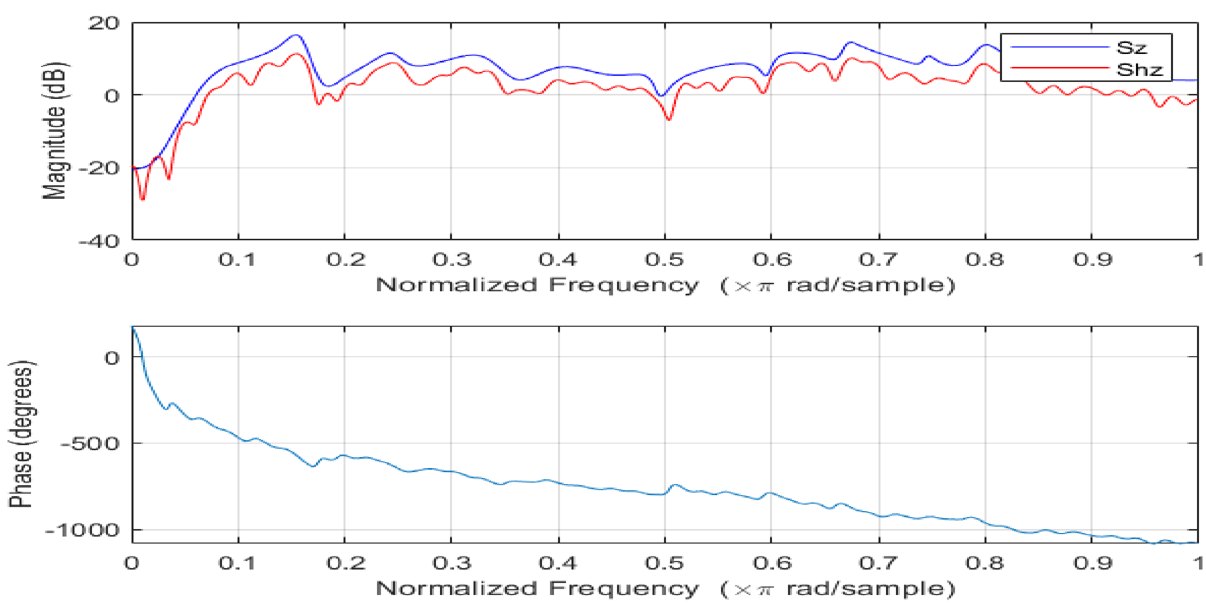

Figure 9: Magnitude Response Comparison of Sz and Shz

Conclusion:

A small mu(step size) causes slow convergence rate but high steady state performance. On the other hand, large mu(step size) converge fast but low steady state performance. In order to rape with this trade off a variable step size can be used. In the transient conditions, the step size set to relatively large number and it is decreased while the system converges to its steady state. The filter length(L) is too large it will not converge fast enough. The longer filter length efficiency is related to longer iterations. I got the best result with L=200 and mu=0.006 with 1000 iterations.

3.2 Approach 2: Online FxLMS

Online FxLMS part codes and graphs:

```
X=SEC18R;
% and measure the arriving noise at the sensor position,
d=filter(P_z, P_p, X);
% Initiate the system,
d=d';
Z = length(X);
xn = zeros(1,L);
y=zeros(1,L);
W = zeros(1,L);
e = zeros(1,Z);
 X = filter(Shz,1,X) ;  %xhat
mu1 =0.000000001;        %step size
%Real ANC part
for k=1:Z
   xn = [X(k), xn(1:L-1)]; % update the state of xhat
   y=xn*W'; % calculate output of yhat
e(k) = d(k)-y;% calculate error
W =W +mu1*xn*e(k); % adjust the weight

end
```

mu1 = **1e-9/*L*:100**

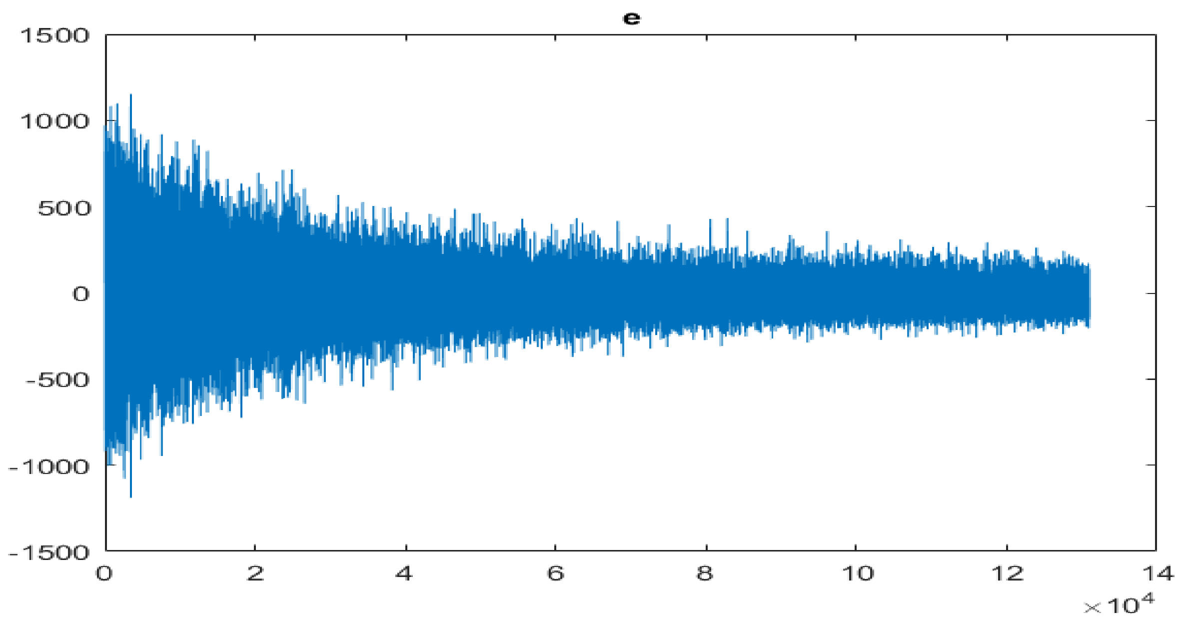

Figure 10: Error

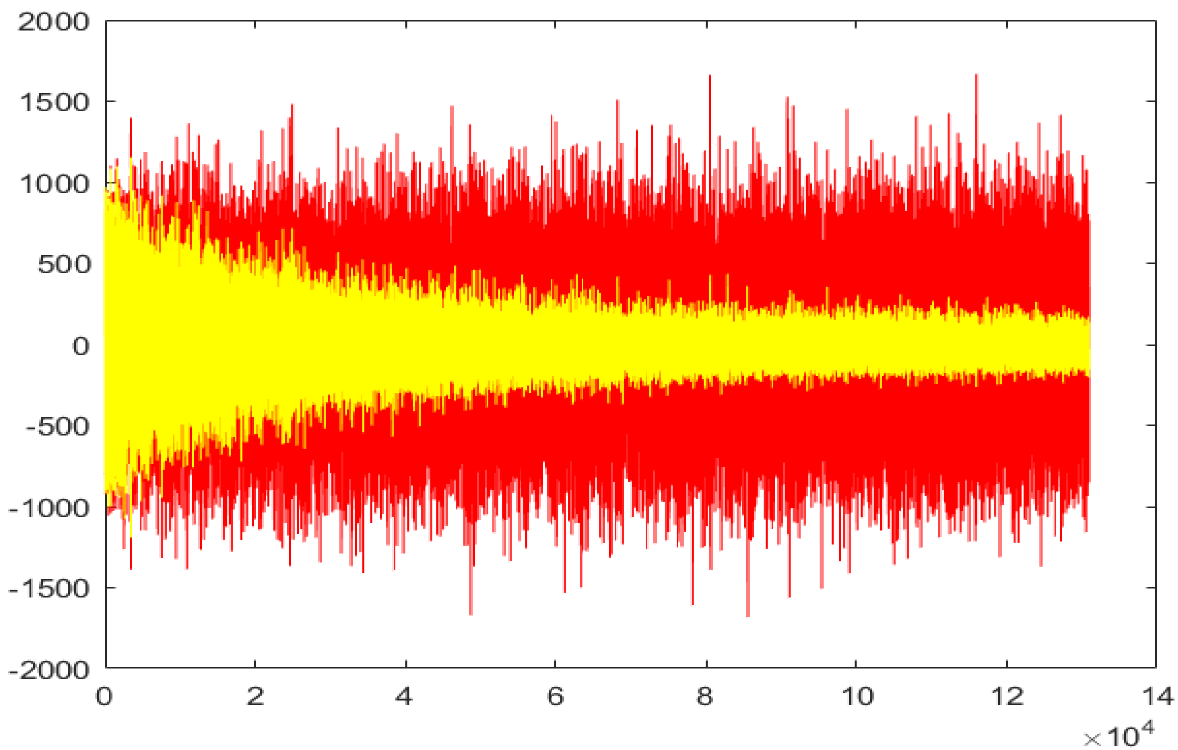

Figure 11: d(n) and e(n) ANC diagram

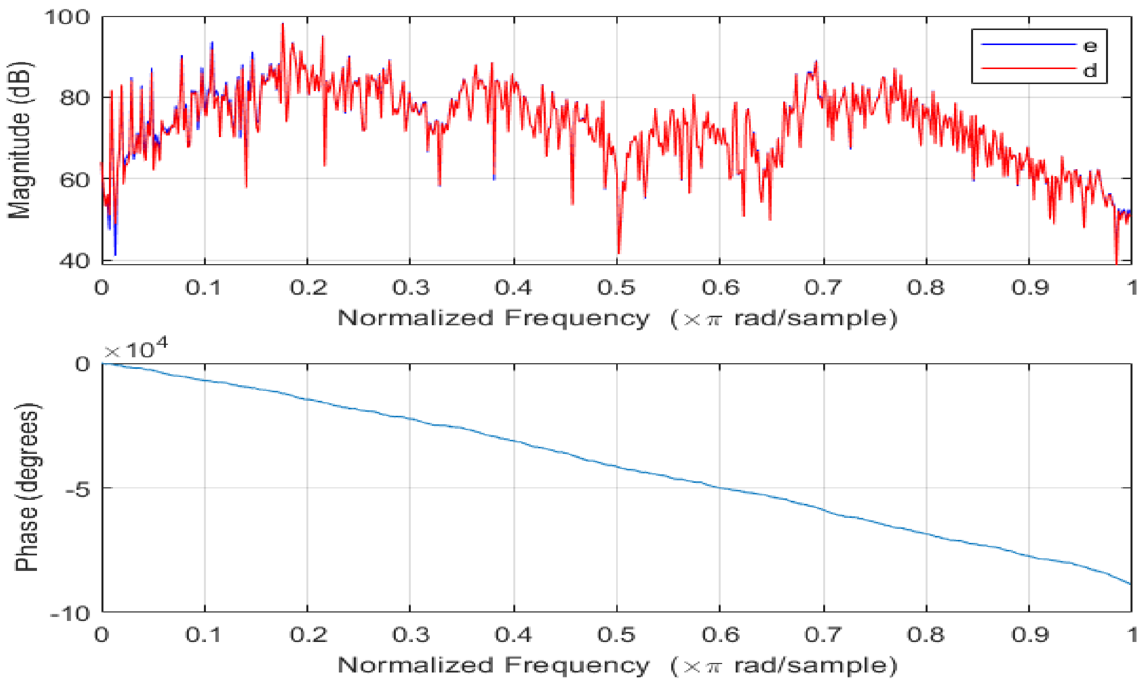

Figure 12: Magnitude Response Comparison of d and e

mu1 = **5e-10/*L*:100**

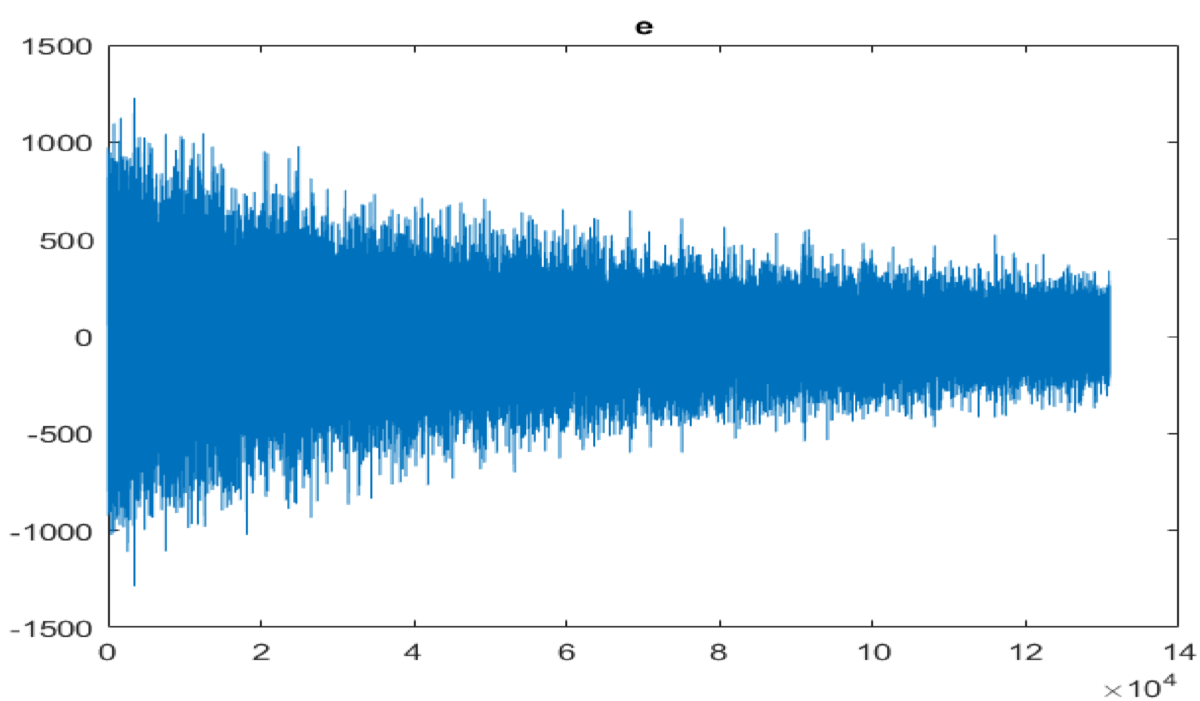

Figure 13: Error

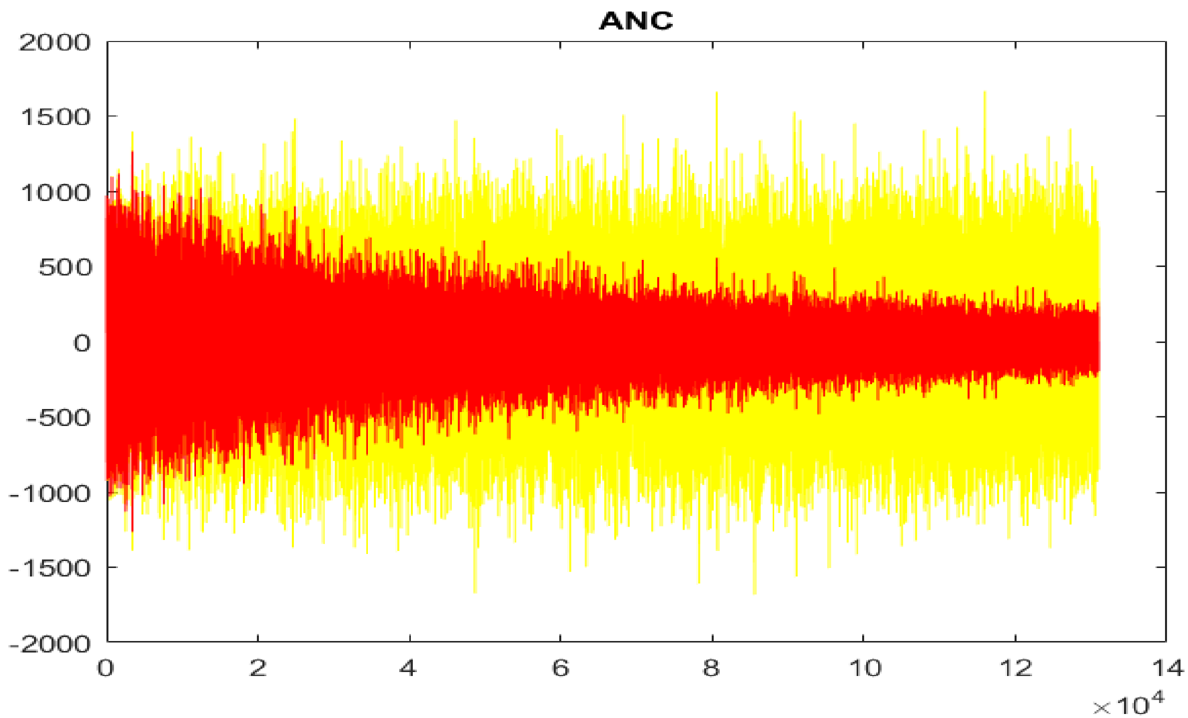

Figure 14: d(n) and e(n) ANC diagram

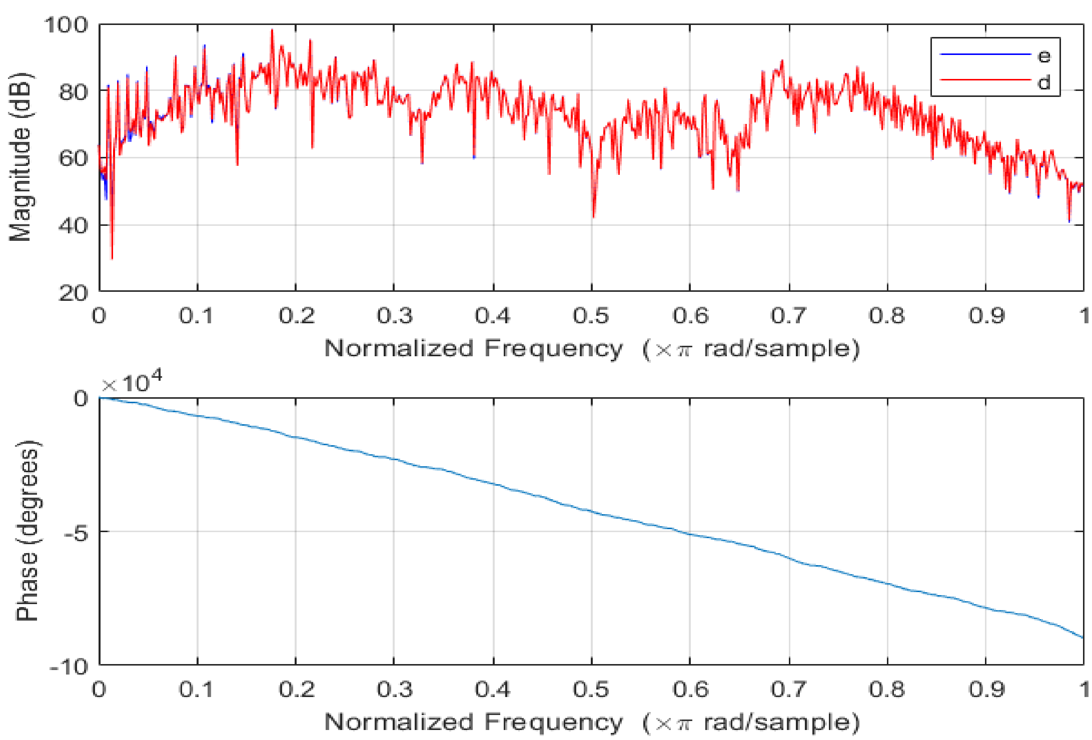

Figure 15: Magnitude Response Comparison of d and e

mu1 = **6e-9/*L*:50**

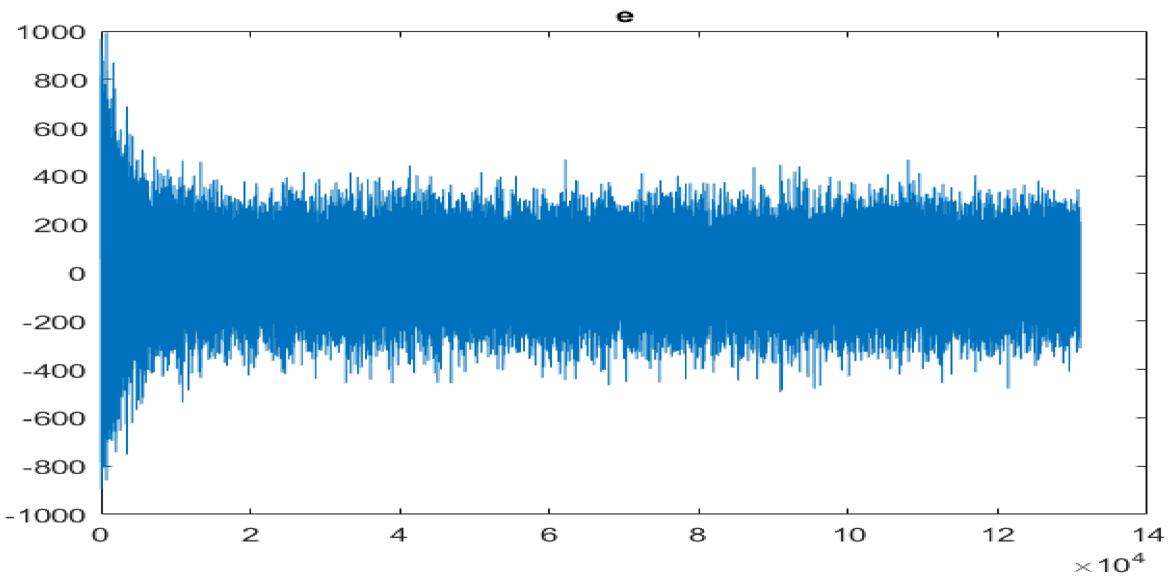

Figure 16: Error

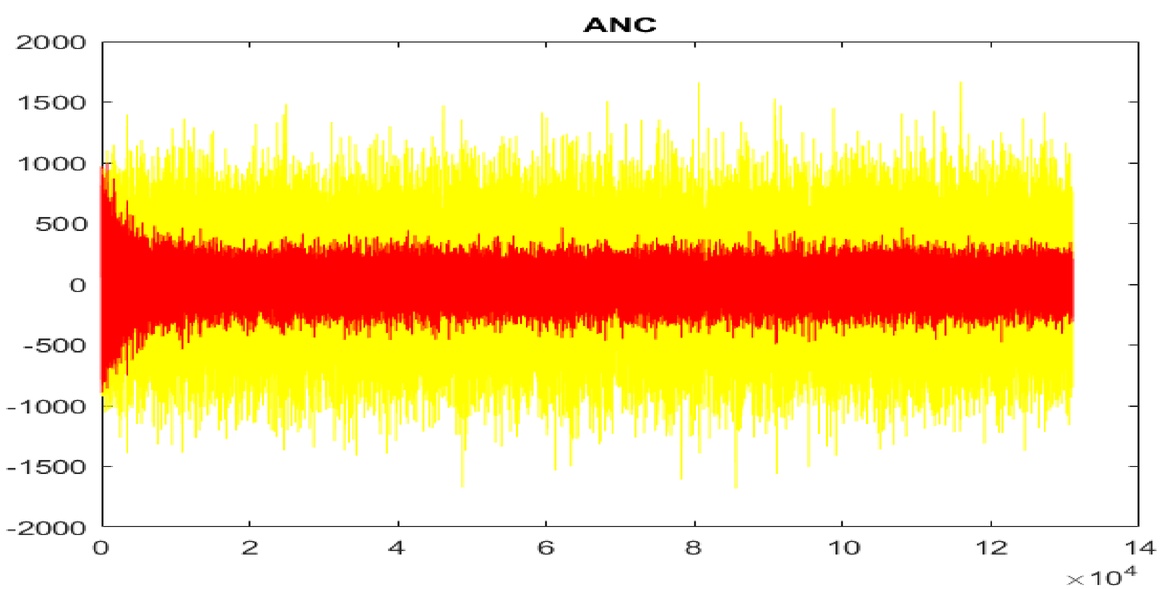

Figure 17: d(n) and e(n) ANC diagram

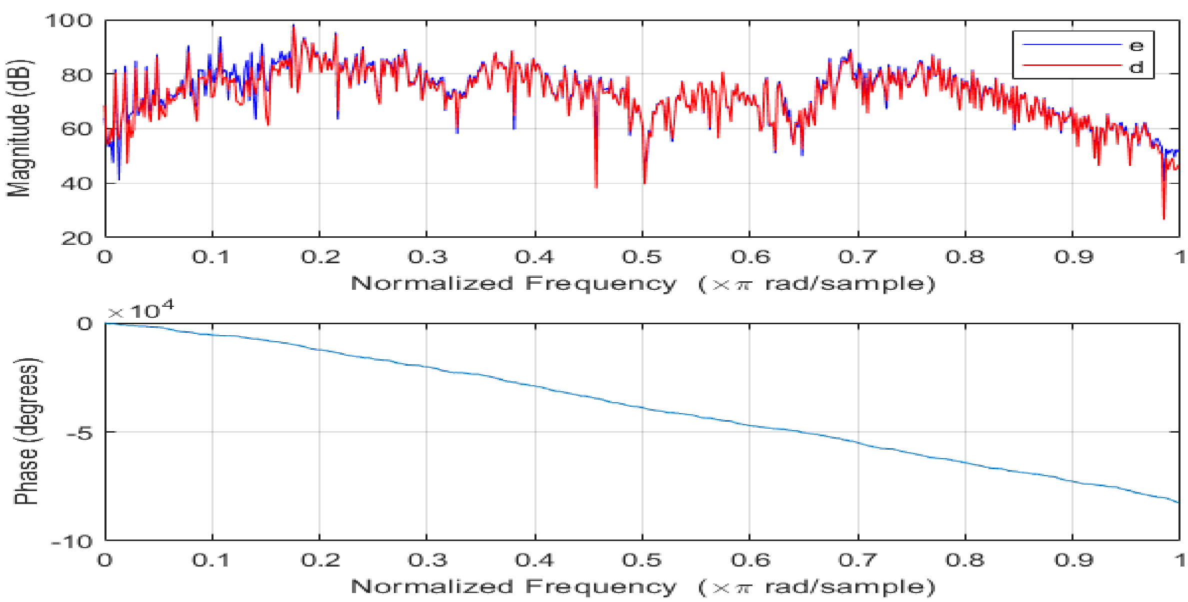

Figure 18: Magnitude Response Comparison of d and e

Conclusion:

Step size is much lower in online part.The reason is the filter weights will be very off at the beginning aand error will quickly get to large and not converge. For this reason the step size needs to be smaller.

The error when its converges is considerably larger than the secondary estimation error. Same filter length(L) used in online part. But the step size and filter length needs to be compatible for converges. Online secondary path modeling enhance the performance of ANC system.