Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics and Communication Engineering

Speech Signal Denoising Using Adaptive Filter

A Project Report

submitted to

Faculty: Prof. KALAIVANI S

Slot: C1

In

Digital Signal Processing

(ECE2006)

By:

Abhirup Ghosh -19BEC0417

Deva Chandragiri -19BEC0227

Devakinandan Palla -19BEC0812

CERTIFICATE

This is to certify that the project report entitled “**Speech Signal Denoising Using Adaptive Filter”** submitted by **Abhirup Ghosh (19BEC0417), Deva Chandragiri (19BEC0227), Devakinandan Palla (19BEC0812)** to Vellore Institute of Technology is a record of bonafide project report undertaken by us under the supervision of **Prof. KALAIVANI S (Professor) School of Electronics and Communication Engineering**. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other Project Work in any other subject.

Signature of the Professor

TABLE OF CONTENT

Aim:

The aim of the project is to define an adaptive filter algorithm and use it to denoise real world signal and compare it with an existing algorithm. We will be using matlab to execute the algorithm. Through this project we hope to acquire more knowledge on practical denoising devices, adaptive filtering techniques and algorithms as well as coding practical algorithms in matlab.

Abstract:

Recent technological advancements have revolutionized the use of systems such as cellular phones, tablets, biomedical wearables, drones, satellites, and remote power management systems. To achieve the objective of handling more sophisticated attributes and services, the systems are becoming more and more complex. Design constraints are becoming imperative regarding system processing speed, precision, power consumption, and size. These systems should be efficient, especially in terms of power consumption, as they are powered by batteries. So, Signal Processing is a very important topic to make a signal efficient for the specific system. We are going to visualize it with signal denoising.

It is sometimes desirable to have circuits capable of selectively filtering one frequency or range of frequencies out of a mix of different frequencies in a circuit. A circuit designed to perform this frequency selection is called a *filter circuit*, or simply a *filter*.

A common need for filter circuits is in high-performance stereo systems, where certain ranges of audio frequencies need to be amplified or suppressed for best sound quality and power efficiency.

We are familiar with *equalizers*, which allow the amplitudes of several frequency ranges to be adjusted to suit the listener's taste and acoustic properties of the listening area.

So, if we can specify a certain amount of frequency in a signal, a filter can select the frequency. That way we can then work with that frequency level. We can even remove it from the signal, that's what our target is in this project.

Theory & Calculations:

- **Least Mean Squares Algorithm:**

Least mean squares (LMS) algorithms are a class of adaptive filter used to mimic a desired filter by finding the filter coefficients that relate to producing the least mean square of the error signal (difference between the desired and the actual signal). It is a stochastic gradient descent method in that the filter is only adapted based on the error at the current time. It was invented in 1960 by Stanford University professor Bernard Widrow and his first Ph.D. student, Ted Hoff.

The basic idea behind LMS filter is to approach the optimum filter weights, by updating the filter weights in a manner to converge to the optimum filter weight. This is based on the gradient descent algorithm. The algorithm starts by assuming small weights (zero in most cases) and, at each step, by finding the gradient of the mean square error, the weights are updated.

LMS algorithm is a type of adaptive filter known as stochastic gradient-based algorithms as it utilises the gradient vector of the filter tap weights to converge on the optimal wiener solution. It is well known and widely used due to its computational simplicity. It is this simplicity that has made it the benchmark against which all other adaptive filtering algorithms are judged. With each iteration of the LMS algorithm, the filter tap weights of the adaptive filter are updated according to the following formula.

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

Here x(n) is the input vector of time delayed input values, x(n) = [x(n) x(n-1) x(n-2) .. x(n-N+1)]T . The vector w(n) = [w0(n) w1(n) w2(n) .. wN-1(n)]T represents the coefficients of the adaptive FIR filter tap weight vector at time n. The parameter μ is known as the step size parameter and is a small positive constant. This step size parameter controls the influence of the updating factor. Selection of a suitable value for μ is imperative to the performance of the LMS algorithm, if the value is too small the time the adaptive filter takes to converge on the optimal solution will be too long; if μ is too large the adaptive filter becomes unstable and its output diverges.

The main reason for the LMS algorithms popularity in adaptive filtering is its computational simplicity, making it easier to implement than all other commonly used adaptive algorithms. For each iteration the LMS algorithm requires 2N additions and 2N+1 multiplications (N for calculating the output, y(n), one for 2μe(n) and an additional N for the scalar by vector multiplication.

• NORMALISED LEAST MEAN SQUARE (NLMS) ALGORITHM

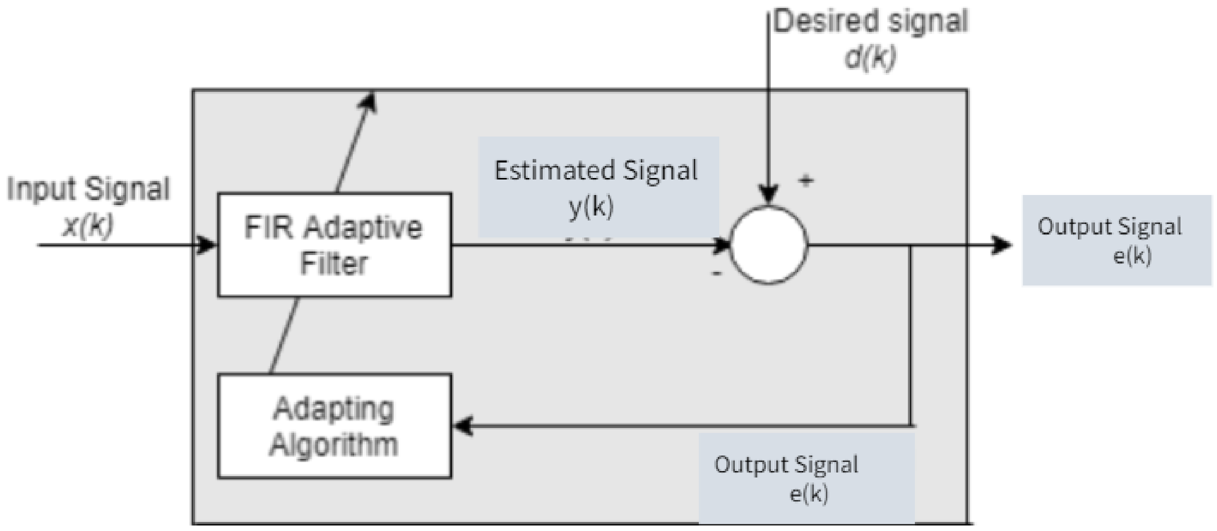
One of the primary disadvantages of the LMS algorithm is having a fixed step size parameter for every iteration. This requires an understanding of the statistics of the input signal prior to commencing the adaptive filtering operation. In practice this is rarely achievable. Even if we assume the only signal to be input to the adaptive echo cancellation system is speech, there are still many factors such as signal input power and amplitude which will affect its performance. The normalised least mean square algorithm (NLMS) is an extension of the LMS algorithm which bypasses this issue by calculating maximum step size value. Step size value is calculated by using the following formula. Step size=1/dot product (input vector, input vector) This step size is proportional to the inverse of the total expected energy of the instantaneous values of the coefficients of the input vector x(n). This sum of the expected energies of the input samples is also equivalent to the dot product of the input vector with itself, and the trace of input vectors auto-correlation matrix, R

$$\begin{aligned} tr[\mathbf{R}] &= \sum_{i=0}^{N-1} E\left[x^2(n-i)\right] \\ &= E\left[\sum_{i=0}^{N-1} x^2(n-i)\right] \end{aligned} \tag{5}$$

The recursion formula for the NLMS algorithm is stated in equation 6.

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \frac{1}{\mathbf{x}^T(n)\mathbf{x}(n)} e(n)\mathbf{x}(n) \tag{6}$$

Block Diagram



Implementation of the NLMS algorithm

The NLMS algorithm has been implemented in Matlab. As the step size parameter is chosen based on the current input values, the NLMS algorithm shows far greater stability with unknown signals. As the NLMS is an extension of the standard LMS algorithm, the NLMS algorithms practical implementation is very similar to that of the LMS algorithm. Each iteration of the NLMS algorithm requires these steps in the following order.

$\mu = \text{step size}$

Initialization: $\hat{\mathbf{h}}(0) = \mathrm{zeros}(p)$

Computation: For $n = 0, 1, 2, \ldots$

$$\mathbf{x}(n) = [x(n), x(n-1), \ldots, x(n-p+1)]^T$$

$$e(n) = d(n) - \hat{\mathbf{h}}^H(n)\mathbf{x}(n)$$

$$\hat{\mathbf{h}}(n+1) = \hat{\mathbf{h}}(n) + \frac{\mu\, e^*(n)\mathbf{x}(n)}{\mathbf{x}^H(n)\mathbf{x}(n)}$$

Where x(n), e(n), w(n) and μ represent the filter input vector, the output (or feedback) signal, the filter tap weight vector and the step size respectively.

MATLAB EXECUTION: (ADAPTIVE FILTER)

```
clc
clear
close all

[y,Fs] = audioread('D:\5th sem\DSP\LAB\1.wav');
D = y(4000:8000);% Desired Signal

n = numel(D); %Total count of elements of the signal
N = 0.9*randn(1,n);
A = D(1:n)+N(1:n); %Signal Corrupted with noise (The
actual signal input)

M = 25; %Constant Variable to define the filter
normalisation
wi = zeros(1,M); %initialise weight factors [array of
zeros]
E=[];
den = A(1,:)*A(1,:)';% mu = 1/den

for i=M:n
    E(i) = D(i)-wi*(A(i:-1:i-M+1)');
    wi = wi + (1/den)*(E(i)'*(A(i:-1:i-M+1)));
end

Est = zeros(n,1);
for i=M:n
    j = A(i:-1:i-M+1);  %Array of i to i-M+1 by
decreasing -1 at each step
    Est(i) = ((wi*(j)'));
end

Err = D - Est';

figure(1);
subplot(4,1,1); plot(D);
title('Desired Signal');

subplot(4,1,2); plot(A);
title('Signal corrupted with noise');

subplot(4,1,3); plot(Est);
```

```
title('Estimated signal');

subplot(4,1,4); plot(Err);
title('Output signal');

x=0;
h=[];
Sed = 0;
Se=0; Se2=0;
Sd=0; Sd2=0;
for i=1:n
    h(i) = ((Err(i)-D(i))^2);
    x = x + ((Err(i)-D(i))^2);
    Se = Se+Err(i); Se2 = Se2+Err(i)*Err(i);
    Sd = Sd+D(i); Sd2 = Sd2+D(i)*D(i);
    Sed = Sed+(Err(i)*D(i));
end

co = (Sed/n) - ((Se/n)*(Sd/n));
corrcoeff = co/(sqrt((Se2/n)-(Se/n)^2)*sqrt((Sd2/n)-
(Sd/n)^2));

p = 10^(-4);
mse = x/n;
fprintf('The mean squared error = %f \n', mse);
figure(2);
plot(h);
title('Mean squared error');

SNR = snr(Err,N);
s = 20*log10(SNR);
fprintf('The signal noise ratio (in dB) = %f \n', s);
fprintf('Correlation coeff = %f \n', corrcoeff);
```

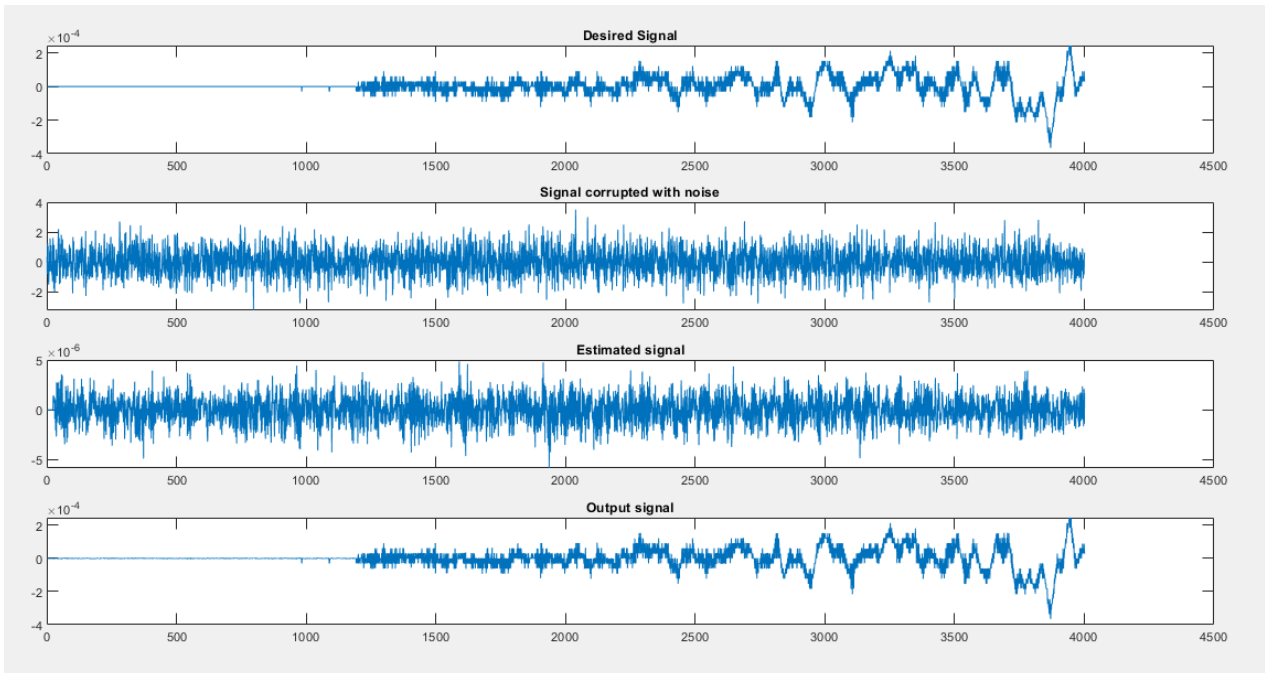
NORMAL LMS FILTER CODE:

```
clc
clear
close all

[y,Fs] = audioread('D:\5th sem\DSP\LAB\1.wav');
D = y(2000:8000);
%t=0.001:0.001:1;
%D = 2*sin(2*pi*50*t); % Desired Signal

n = numel(D); %Total count of elements of the signal
N = 0.9*randn(1,n);
A = D(1:n)+N(1:n); %Signal Corrupted with noise (The
actual signal input)

M = 25; %Constant Variable to define the filter
normalisation
wi = zeros(1,M); %initialise weight factors [array of
zeros]
E=[];
mu=0.05;

for i=M:n
    E(i) = D(i)-wi*(A(i:-1:i-M+1)');
    wi = wi + (mu)*(E(i)*(A(i:-1:i-M+1)));
end

Est = zeros(n,1);
for i=M:n
    j = A(i:-1:i-M+1);  %Array of i to i-M+1 by
decreasing -1 at each step
    Est(i) = ((wi*(j')));
end

Err = D-Est';

figure(1);
subplot(4,1,1); plot(D);
title('Desired Signal');

subplot(4,1,2); plot(A);
title('Signal corrupted with noise');
```

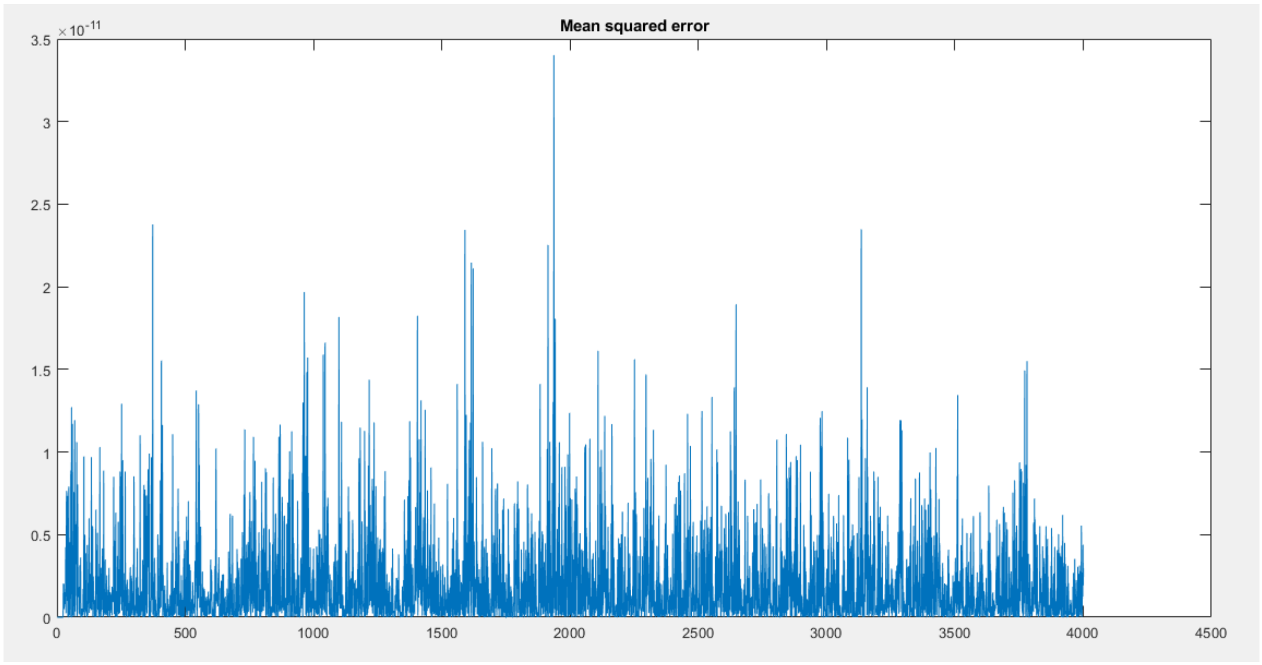
```
subplot(4,1,3); plot(Est);
title('Estimated signal');

subplot(4,1,4); plot(Err);
title('Output signal');

x=0;
h=[];
Sed = 0;
Se=0; Se2=0;
Sd=0; Sd2=0;
for i=1:n
    h(i) = ((Err(i)-D(i))^2);
    x = x + ((Err(i)-D(i))^2);
    Se = Se+Err(i); Se2 = Se2+Err(i)*Err(i);
    Sd = Sd+D(i); Sd2 = Sd2+D(i)*D(i);
    Sed = Sed+(Err(i)*D(i));
end

co = (Sed/n) - ((Se/n)*(Sd/n));
corrcoeff = co/(sqrt((Se2/n)-(Se/n)^2)*sqrt((Sd2/n)-
(Sd/n)^2));

p = 10^(-4);
mse = x/n;
fprintf('The mean squared error = %f \n', mse);
figure(2);
plot(h);
title('Mean squared error');

SNR = snr(Err,N);
s = 20*log10(SNR);
fprintf('The signal noise ratio (in dB) = %f \n', s);
fprintf('Correlation coeff = %f \n', corrcoeff);
```

<u>Result and Inference:</u> (Input: song with bass)

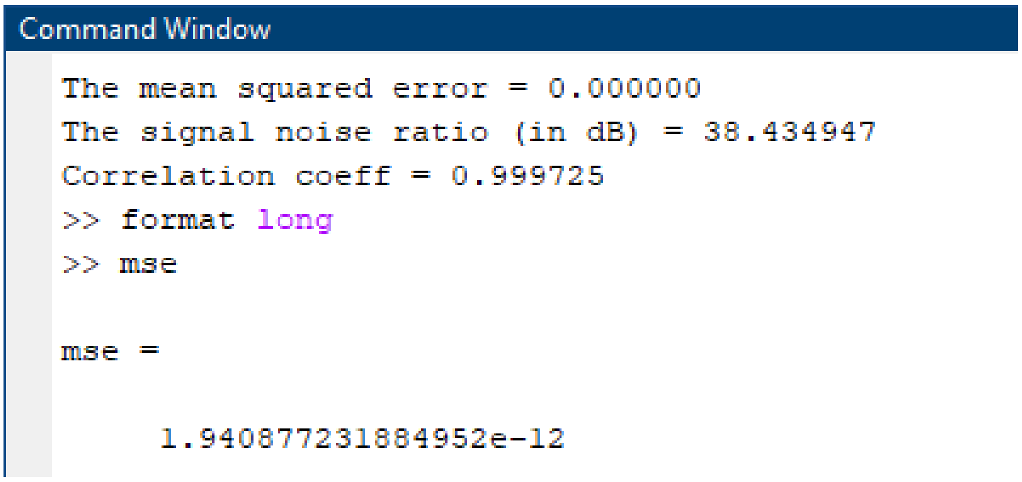
ADAPTIVE FILTER OUTPUT



We are reading a real world audio signal through audioread() function and are adding random noise to it. After processing through the algorithm we are getting an output quite similar to our desired signal. We are also getting a mean squared error of 10^(-12) scale which is very small with respect to the desired signal of 10^(-4) scale.



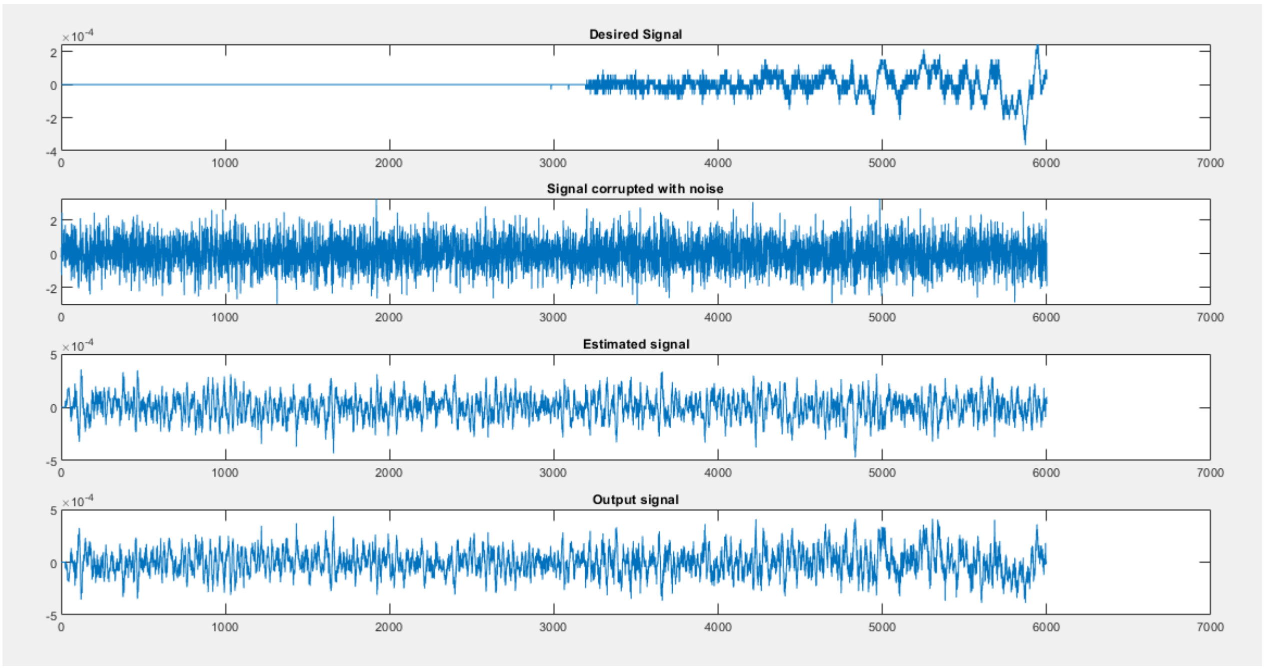
Mean squared error (in 10^(-11) scale)



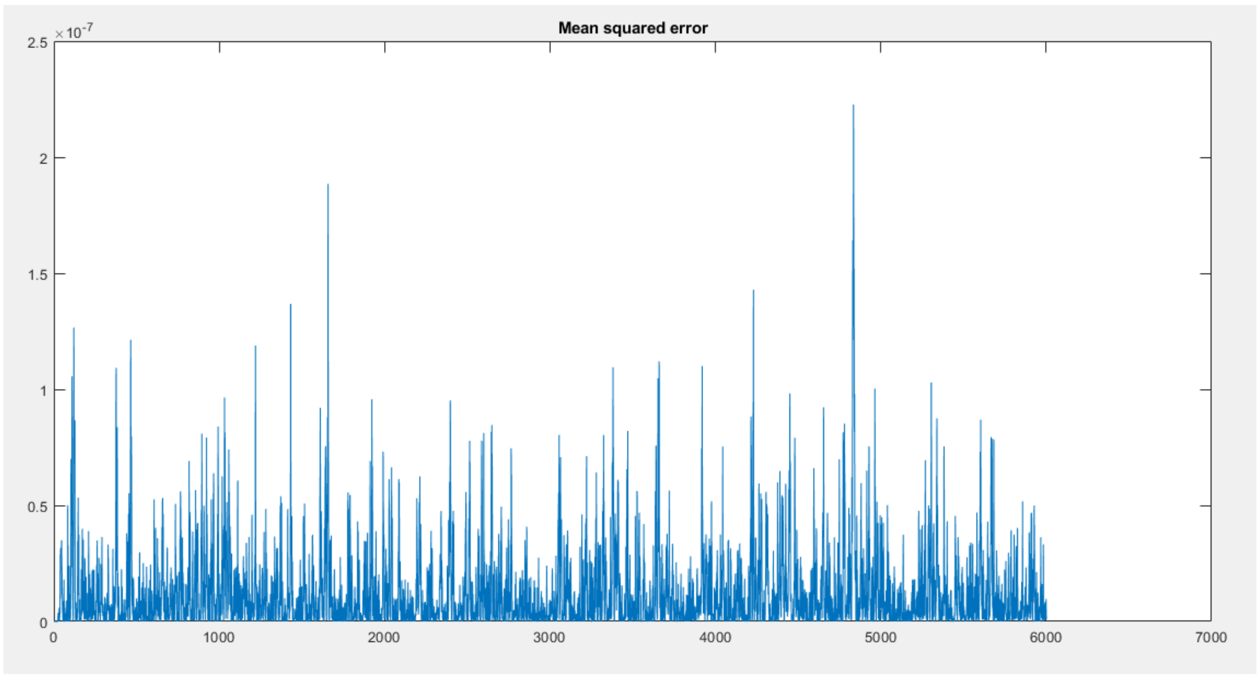
```
The mean squared error = 0.000000
The signal noise ratio (in dB) = 38.434947
Correlation coeff = 0.999725
>> format long
>> mse

mse =

     1.940877231884952e-12
```

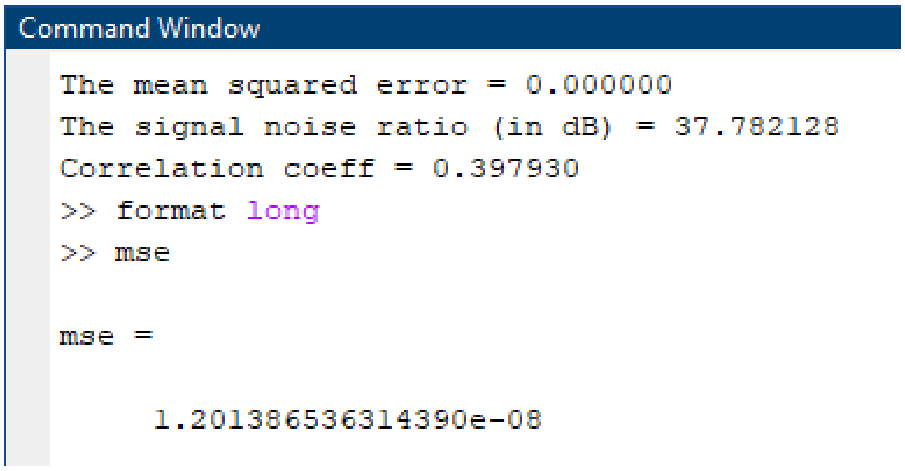
The value of mean squared error upon calculation is also proving the plot by showing 0 upto 5th decimal point. Also, the signal to noise ratio of 38.434947 dB (close to 40 dB) and a correlation coefficient of 0.999725 (close to 1) proves that the output from the adaptive filter is very strong and effective and the filter efficiency is very high.

NORMAL LMS FILTER OUTPUT:



We are reading the same audio signal through audioread() function and are adding random noise to it. After processing through the algorithm we are getting an output quite that is very different than the desired signal. We are also getting a mean squared error of 10^(-8) scale which is problematic as the desired signal is of 10^(-4) scale.





```
The mean squared error = 0.000000
The signal noise ratio (in dB) = 37.782128
Correlation coeff = 0.397930
>> format long
>> mse

mse =

     1.201386536314390e-08
```

A correlation coefficient of 0.397930 is also proving that the output is very weak with respect to desired signal and also very ineffective with respect to our adaptive algorithm.

I also tried to apply the same filters in case of a different audio input. The noted values will be mentioned in the below table -

Signal (Input to the filter)		Adaptive Filter	Normal LMS Filter
Audio 1 - Song	**MSE**	0.194 (10^(-11) scale)	0.120 (10^(-7) scale)
	SNR (in dB)	38.434	37.782
	Correlation coeff.	0.999725	0.397930
Audio 2 – Birds Chirping	**MSE**	0.195 (10^(-9) scale)	0.149 (10^(-6) scale)
	SNR (in dB)	36.599	35.288
	Correlation coeff.	0.999307	0.625082

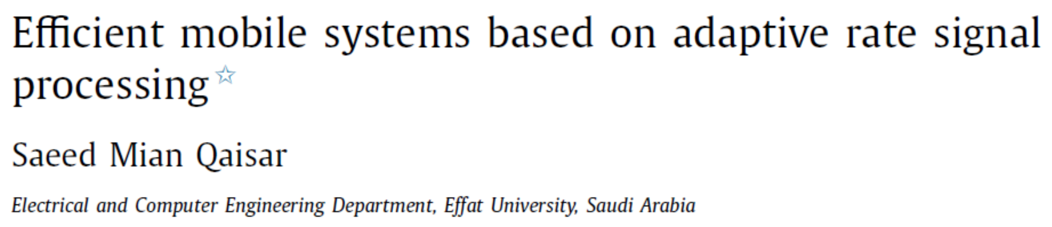
From this table we can easily visualise that our adaptive algorithm is way more efficient than the normal LMS algorithm.

Conclusion:

The whole algorithm is based on the theory of normalised lean mean squared filter theory. We found the output with a very miniscular mean squared error, and a correlation coefficient close to 1, which pretty much concludes our target to find a successfull speech signal or audio signal denoising using adaptive filtering technique. But the algorithm do have a drawback. Whenever we tried to merge two complex audio signals to create the noise, the input signal is somehow coming out to be disorganised. We found the reason to be signal misalignment, which happens due to different characteristics (amplitude, periodic movement etc.) in the signals.

References:

1.

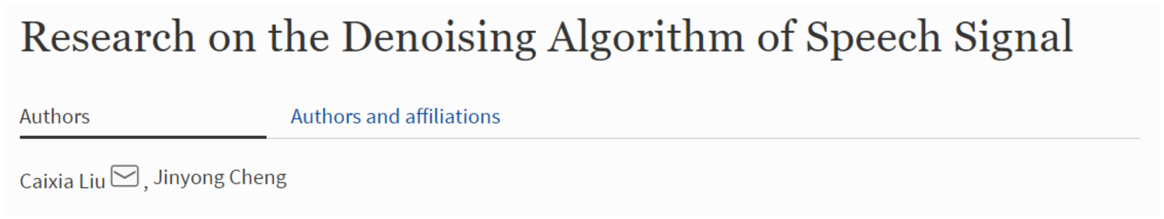


Efficient mobile systems based on adaptive rate signal processing

Saeed Mian Qaisar

Electrical and Computer Engineering Department, Effat University, Saudi Arabia

2.



Research on the Denoising Algorithm of Speech Signal

Authors | Authors and affiliations

Caixia Liu, Jinyong Cheng

3. https://en.wikipedia.org/wiki/Least_mean_squares_filter