Digital Signal Processing Lab

Name: Deep C. Patel

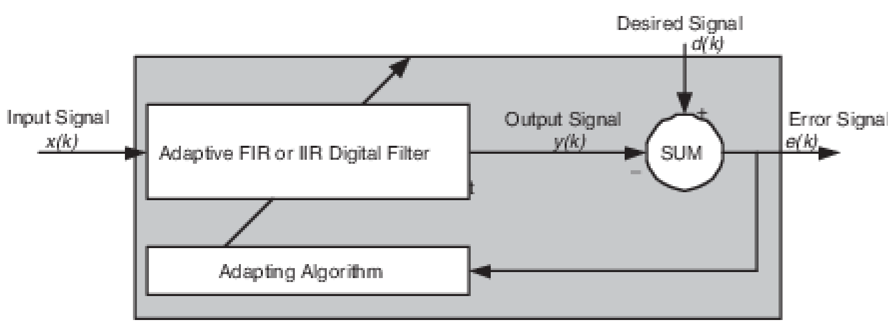
Roll No: 1401010

Lab Report - 1

Theory:-

Adaptive Filtering

Adaptive filtering involves the changing of filter parameters (coefficients) over time, to adapt to changing signal characteristics. An adaptive filter self-adjusts the filter coefficients according to an adaptive algorithm. One such algorithm is LMS Algorithm. We have implemented LMS Algorithm in MATLAB in this lab.



(Diagram for generic Adaptive Filter provided by The MathWorks, Inc. in MATLAB Documentation)

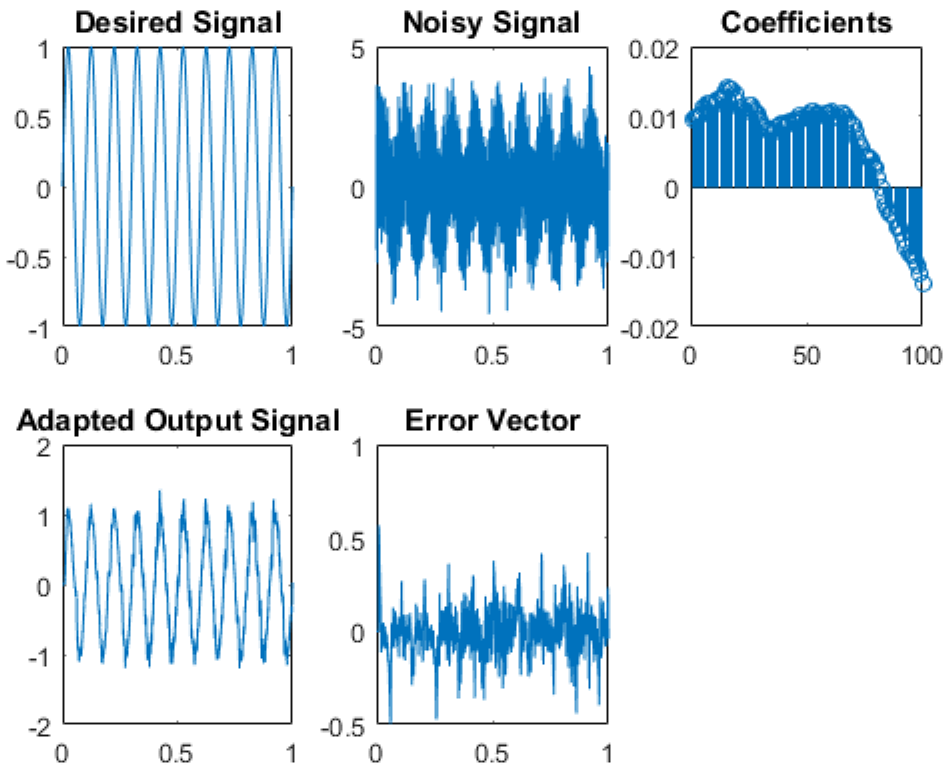
Lab Work:-

Lab – 0

1).LMS Test:-

```
%Lab_1_1 LMS Test

clear;
```

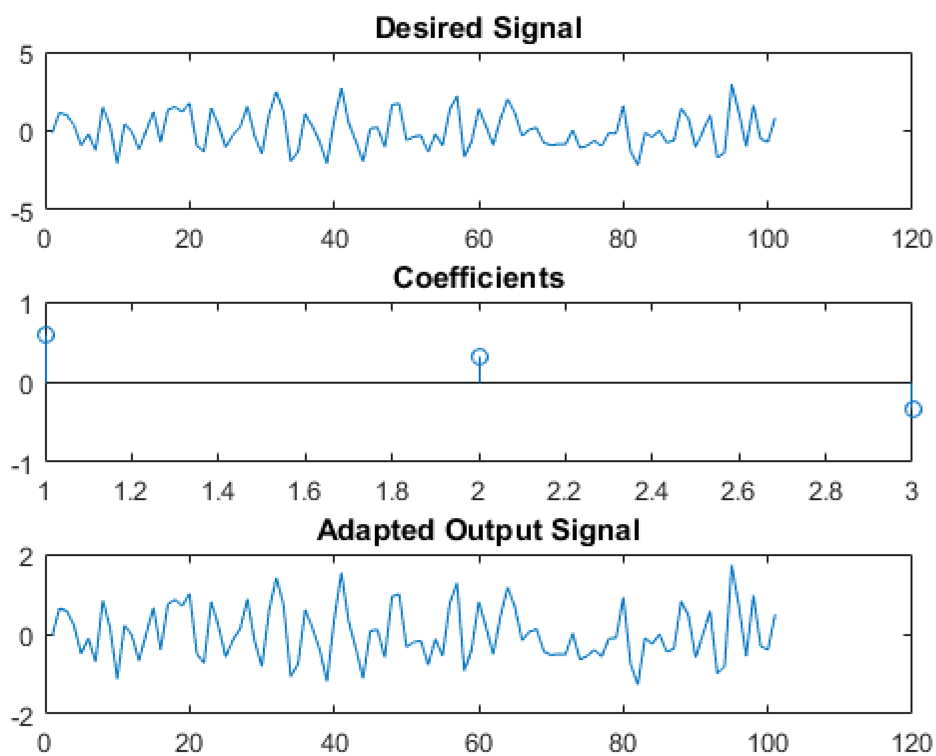
```
clc;

% Time specifications:
Fs = 10300;                  % samples per second or Sampling frequency
dt = 1/Fs;                   % seconds per sample
stop = 1;                    % seconds
t = 0:dt:stop;               % seconds

% Sine wave:
Fc = 10;  % hertz
d = sin(2*pi*Fc*t);
x = d + randn(1,Fs+1);
N=100;                      %Number of Coefficients

delta=0.001;

[h, Y, E] = LMS(x,d,delta,N);

figure;

subplot(2,3,1);
plot(t,d);
title('Desired Signal');
%xtitle('Time');
%ytitle('Signal Value(d)');

subplot(2,3,2);
plot(t,x);
title('Noisy Signal');
%xtitle('Time');
%ytitle('Signal Value(x)');

subplot(2,3,3);
stem(h);
title('Coefficients');
%ytitle('h');

subplot(2,3,4);
plot(t,Y);
title('Adapted Output Signal');
%xtitle('Time');
%ytitle('Signal Value(Y)');

subplot(2,3,5);
plot(t,E);
title('Error Vector');
%ytitle('E');
```



2).<u>FIR Filter</u>:-

```
%Lab_1_2 FIR Filter

clear;
clc;

b=[1 0.5 -0.6];              %Unknown Filter Coefficients
a=1;                         %Unknown Filter Coefficients
x=randn(1,1000+1);

d=filter(b,a,x);             %Simulates the signal from unknown system

N=3;                         %Number of Coefficients

delta=0.001;

[h, Y, E] = LMS(x,d,delta,N);

figure;
```

```
subplot(3,1,1);
plot(d(900:1000));
title('Desired Signal');

subplot(3,1,2);
stem(h);
title('Coefficients');

subplot(3,1,3);
plot(Y(900:1000));
title('Adapted Output Signal');
```



3).LMS Algorithm:-

```
%Linear Mean Square Algorithm (LMS Algorithm)

function [h,Y,E] = LMS(x,d,delta,N)

M=length(x);
Y=zeros(1,M);
h=zeros(1,N);
E=zeros(1,M);

for n=N:M

    x1=x(n:-1:n-N+1);
    Y(n)=h*x1';
    e=d(n)-Y(n);
    h=h+(delta*e*x1);
    E(n)=e;

end

end
```

-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-

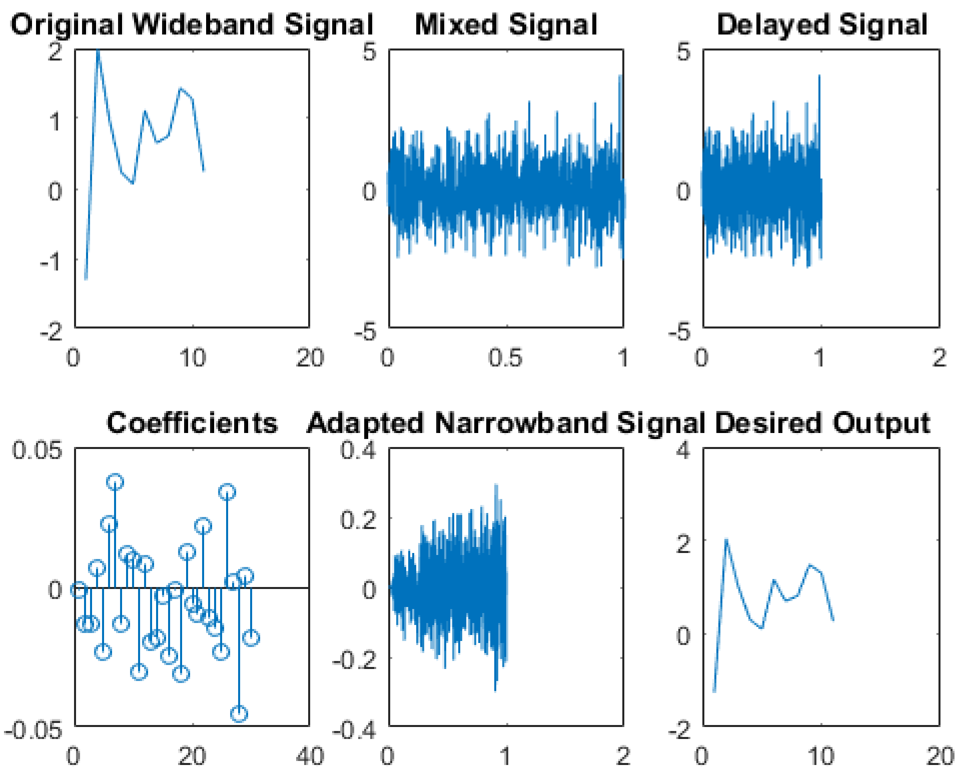
Test – 1

1).Test-1:-

```
%Test_1; Suppressing the Narrowband Signal
%Fact:-Correlation of the Wideband Signal is low and that of Narrowband
%Signal is high

clear;
clc;

% Time specifications:
Fs = 1030;                  % Samples per Second or Sampling frequency
dt = 1/Fs;                  % Seconds per Sample
stop = 1;                   % Seconds
t = 0:dt:stop;              % Seconds
```

```
t_new = 0:dt:stop+dt;       % Seconds

% Sine wave:
Fc = 10;                    % hertz

d = 5*sin(2*pi*Fc*t);
abs_d=sqrt(d*d');
d_m=d./abs_d;               %Normalizing d vector

w=randn(1,Fs+1);

x = d_m + w;
N=30;                       %Number of Coefficients
D=1;                        %Delay Parameter

delay_x=Delay(x,D);         %Delayed X
x_new=[x zeros(1,D)];

delta=0.001;                %Step Parameter

[h, Y, E] = LMS(delay_x,x_new,delta,N);

figure;

subplot(2,3,1);
plot(w(30:40));
title('Original Wideband Signal');
%xtitle('Time');
%ytitle('Signal Value(d)');

subplot(2,3,2);
plot(t,x);
title('Mixed Signal');
%xtitle('Time');
%ytitle('Signal Value(x)');

subplot(2,3,3);
plot(t_new,delay_x);
title('Delayed Signal');
%xtitle('Time');
%ytitle('Signal Value(x)');

subplot(2,3,4);
stem(h);
title('Coefficients');
%ytitle('h');

subplot(2,3,5);
plot(t_new,Y);
title('Adapted Narrowband Signal');
```

```
%xtitle('Time');
%ytitle('Signal Value(Y)');

subplot(2,3,6);
plot(E(30:40));
title('Desired Output');
%ytitle('E');
```



2).<u>LMS Algorithm</u>:-

```
%Linear Mean Square Algorithm (LMS Algorithm)

function [h,Y,E] = LMS(x,d,delta,N)

M=length(x);
Y=zeros(1,M);
h=zeros(1,N);
E=zeros(1,M);

for n=N:M
```

```
    x1=x(n:-1:n-N+1);
    Y(n)=h*x1';
    e=d(n)-Y(n);
    h=h+(delta*e*x1);
    E(n)=e;

end

end
```

3).Delay Function:-

```
%Delay Function

function [delayed_x] = Delay(x,D)

    delayed_x=[zeros(1,D) x];     %Delayed X

end
```

-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-