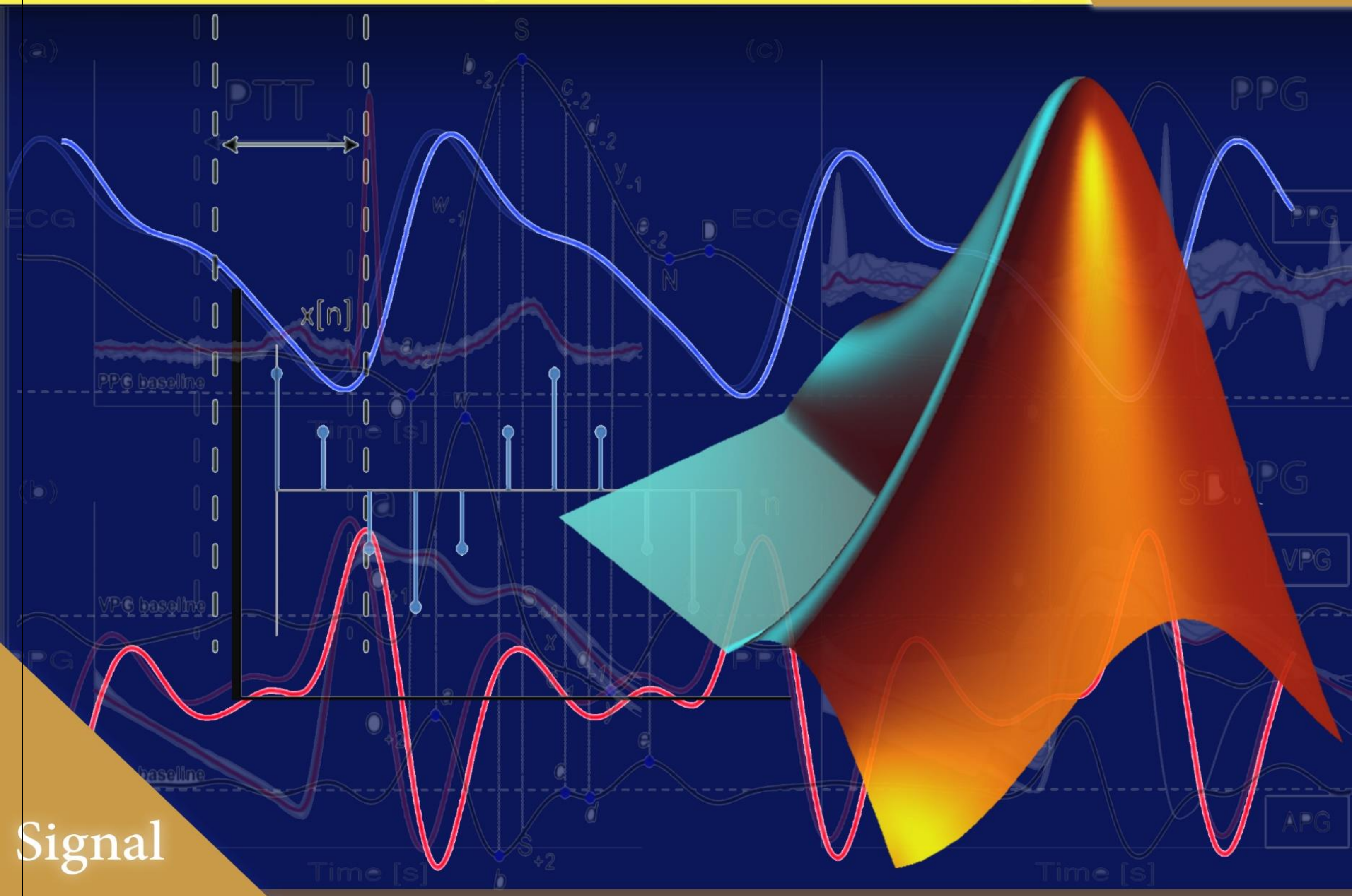
Final Project

Section 2

Semon Joseph Shehata Abdelmalak 18010835
Marina Fouad Aziz 18011310

Section 1

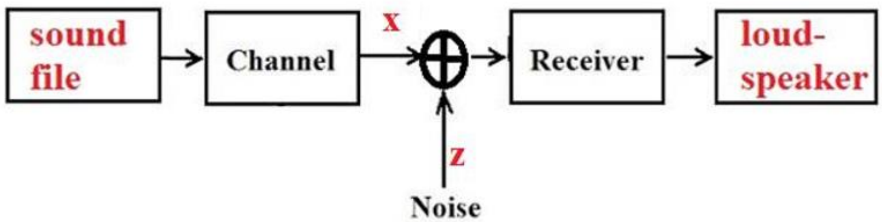
Nada Abdo Hanafy El-Sadany 18011976
Saeed Mabrouk Saeed El-Shaikh 18010788



Signal Processing

Second Year Communicatin

In this project we will implement a very simple communication systems with transmitter, Channel , noise And Receiver



1- Transmitter :
   In this stage " The first stage" we Enter a sound file and prepare it for the transmission over the channel.

First of all , we inserted the sound file by using " audioread" function and it returns sampled data " y " and a sample rate for this data, " Fs " , And to play the sound we used "Sound" function 'we paused the sound after 10 seconds' ..

To deal with the mono signal we defined the new sampled data 'y' to be "y(1: 32*fs, 1) " ' it's the same but with one column '

For the original signal :
We defined the time domain 't' with linspace :
t=linspace (0 , 32 , 32*fs); with number of samples equal to the sound's time * the sample rate

And defined the frequency domain 'f' also with linspace :
f=linspace(-fs/2 , fs/2 , length(y)); with number of samples equal to the y length

And we defined new linspaces to be used in the channels ' their range = (range of original sound*2)-1' :

For time domain 'ConvT' : convT= linspace (0 , 64 , 64*fs-1)

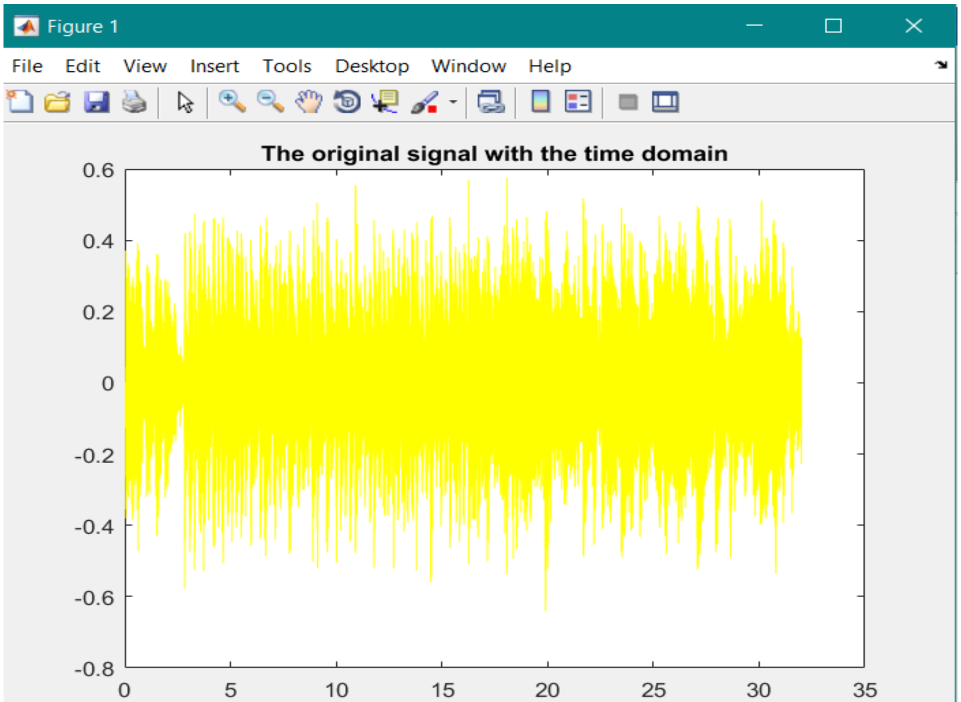
For Frequency domain 'ConvF' :
convF= linspace(-fs/2 , fs/2 , length(convT));
To convert to Frequency Domain we used Fourier transform

With ‘ FFt ‘ Function And the signal is brought to the middle by using ‘ FFtshift ‘ function

So we defined a new variable ‘X’ to be the signal in frecuency domain ‘x= fftshift(fft(y))’

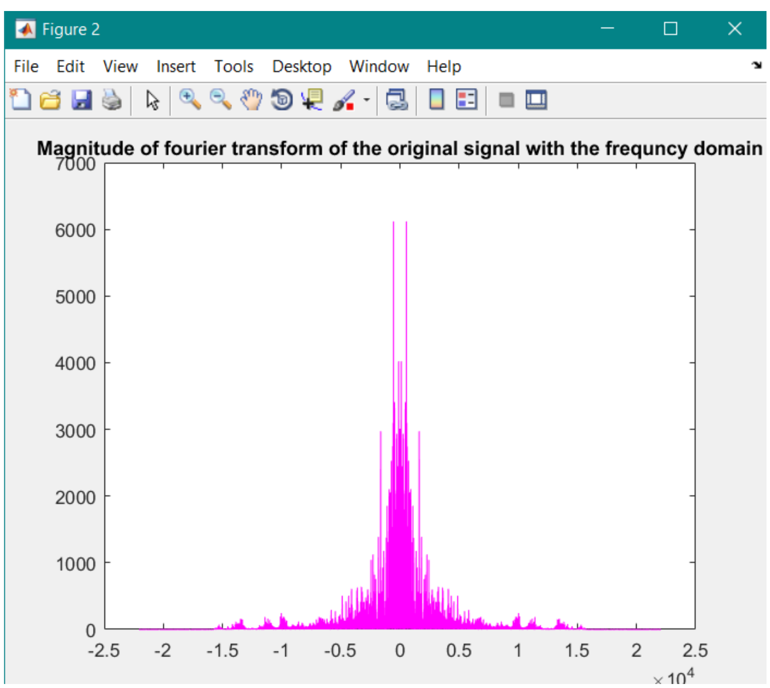
To get the signal magnitude we used ‘ abs ‘ function and stored it in new variable ‘ymag’ and to get the phase we used ‘ angle’ function and stored it in new variable ‘yphase’ “on the frequency domain”

Plot of the signal in the Time Domain:

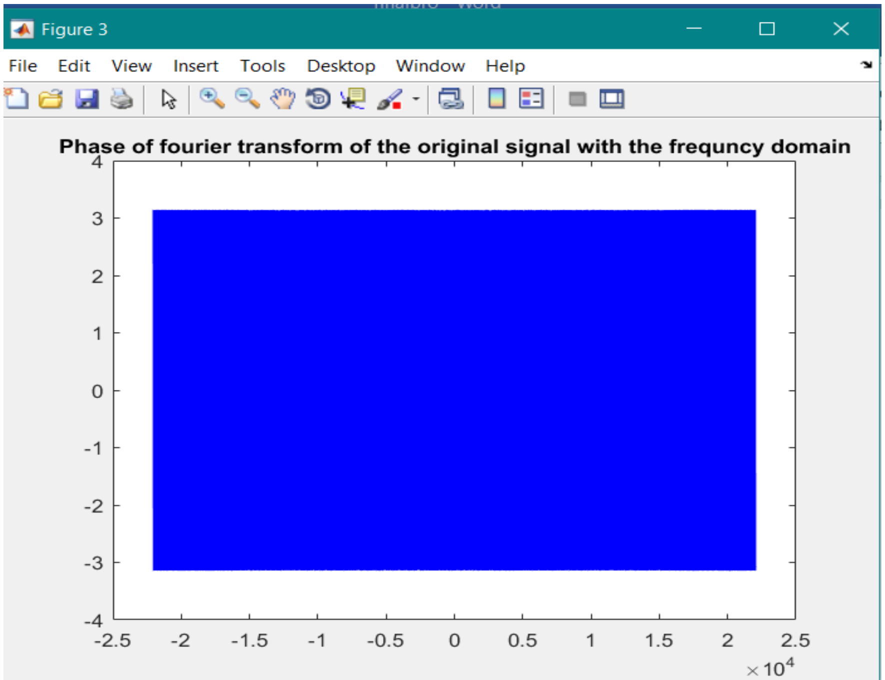


Plot of the signal in the Frequency Domain:

The Magnitude :



The phase :



2-The Channel :

At this stage, we will pass our sound message over the channel

We have 4 options for the channel and the user will choose from them :

We defined ‘m’ variable to store the user choice

And transposed ‘y’ to be used in the convolution

1- The First Channel is Delta Function :

In this channel we will draw the signal with the time and frequency’s linspace of the original signal :

```
convT=t; convF=f;
```

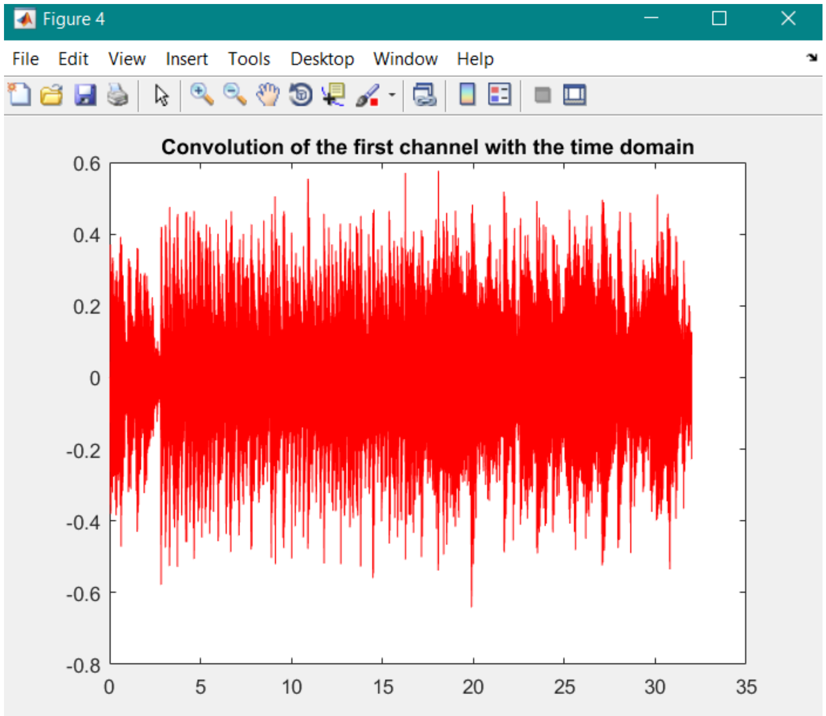
we convoluted a signal defined as

x1=[1 zeros(1,length(y)-1)]; with our signal x2=[y];

and converted them to the frequency domain

```
x1=[1 zeros(1,length(y)-1)];
x2=[y];
X1=(fft(x1));
X2=(fft(x2));
Y= X1.*X2;
```

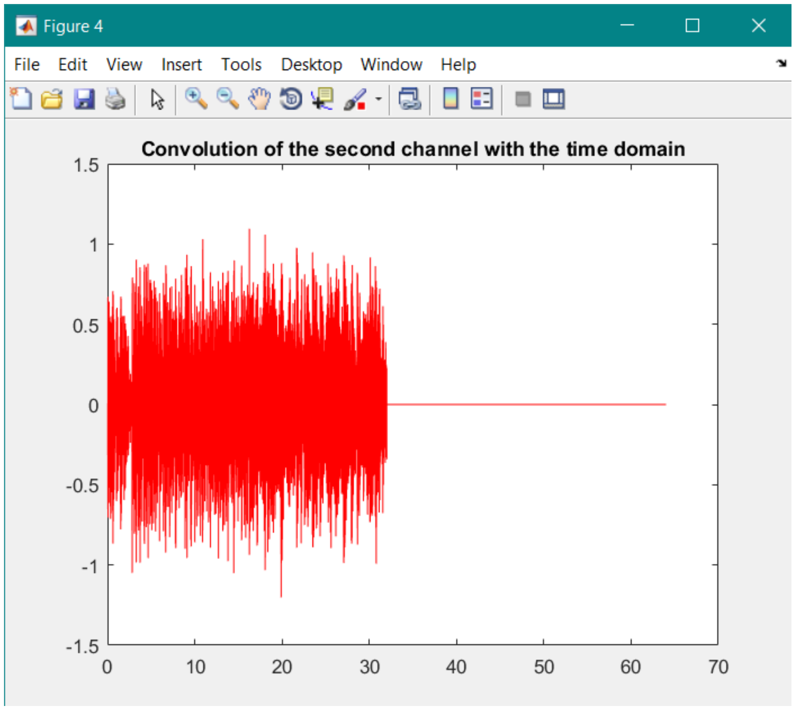
We defined a new variable for the channel ‘CH’ to return the signal to time domain and plot it with ‘convt’



2- The channel of exp(-2pi*5000t) :
 We did convolution using exponential (exp(-2*pi*5000*t)) function

```
        case 2
            %Convolution using exponential (exp(-2*pi*5000*t)) function
            x1= exp(-2*pi*5000*t);
            x1=[x1 zeros(1,length (y)-1)];
            x2=[y  zeros(1,length (y)-1) ];
            X1=(fft(x1));
            X2=(fft(x2));
            Y= X1.*X2;
            CH=real(ifft((Y)));
```

And the plot in the time domain :

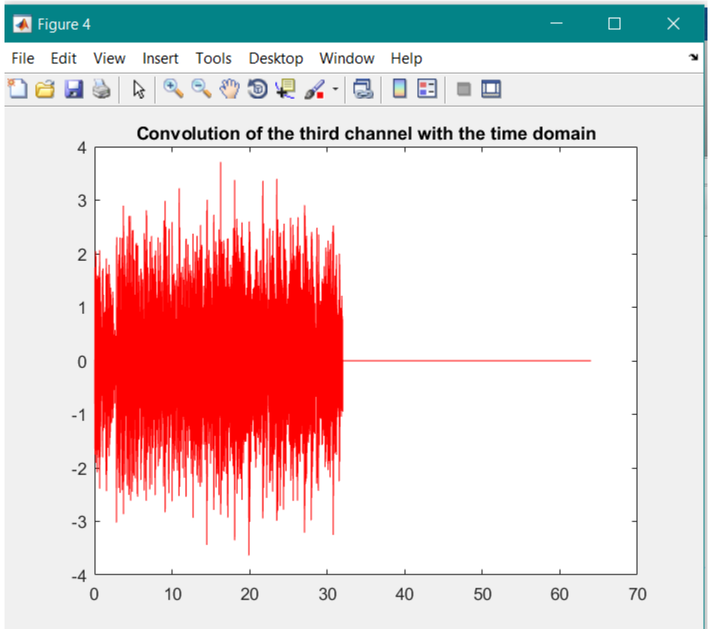


3- The third channel exp(-2pi*1000t) :

We did Convolution using exponential (exp(-2*pi*1000*t)) function

```
    case 3
        %Convolution using exponential (exp(-2*pi*1000*t)) function
        x1= exp(-2*pi*1000*t);
        x1=[x1 zeros(1,length (y)-1)];
        x2=[y zeros(1,length (y)-1) ];
        X1=(fft(x1));
        X2=(fft(x2));
        Y= X1.*X2;
        CH=real(ifft((Y)));
```

The plot in the time domain:

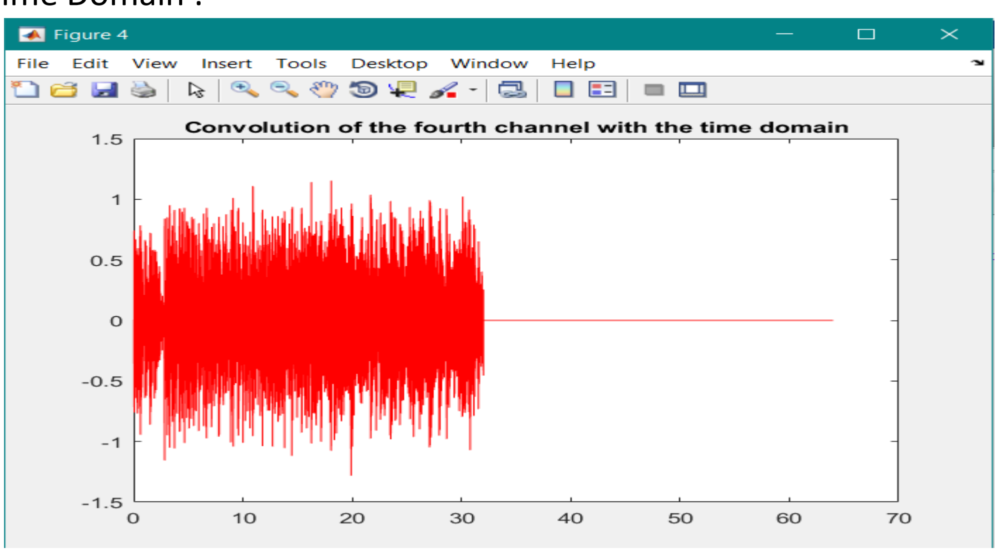


4 – The last Channel :

We did Convolution using two delta functions as following :

```
        case 4
            %Convolution using two delta functions
             x1= 2*(t==0)+0.5*(t==1);
             x1=[x1 zeros(1,length (y)-1)];
             x2=[y zeros(1,length (y)-1) ];
             X1=(fft(x1));
             X2=(fft(x2));
             Y= X1.*X2;
             CH=real(ifft((Y)));
```

And the plot in Time Domain :



If we compare the effect of the first three channels on the sound signal :

- the first channel “ The Delta function” had no change it’s the same as the original sound.

- The second channel “exp(-2pi*5000t)” its voice is higher than the original sound

- The Third Channel “exp(-2pi*1000t)” its voice increased more than the original one and the second Channel and it nearly has a noise

3-The third stage is to add a noise to the sound :

The program have the ability to add noise (simply random signal) to the output of the channel
The random signal generation is done as following

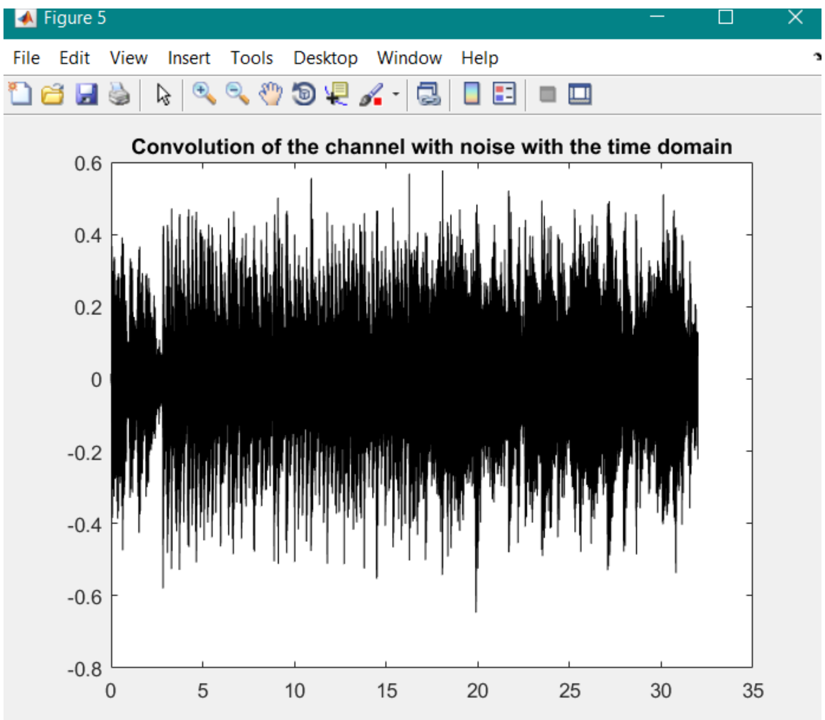
Z(t) =sigma*randn(1,length(CH)) Where CH is a vector represents the output of the channel

We asked the user to add a value of sigma and we add it to the signal

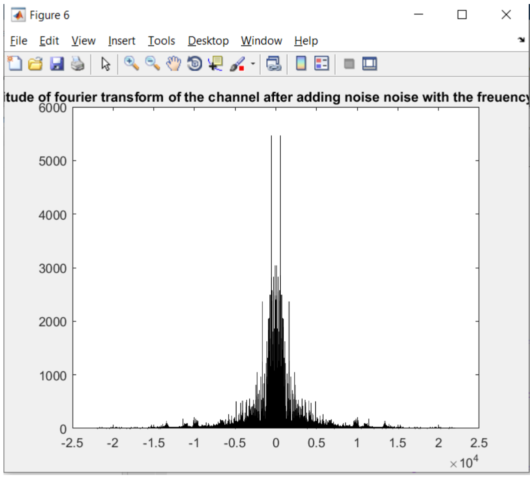
```
%Adding noise
sigma=input('Please enter th value of sigma:'); %  STANDARD DEVIATION
z=sigma*randn(1,length(CH));%Noise function
CH=CH+z;%Adding noise to the desired channel result
sound (CH,fs);%Playing audio after adding noise
pause (10);
clear sound;
figure(5);
plot(convT,CH,'k');
title('Convolution of the channel with noise with the time domain','color','k');
CH =real(fftshift (fft(CH)));
```

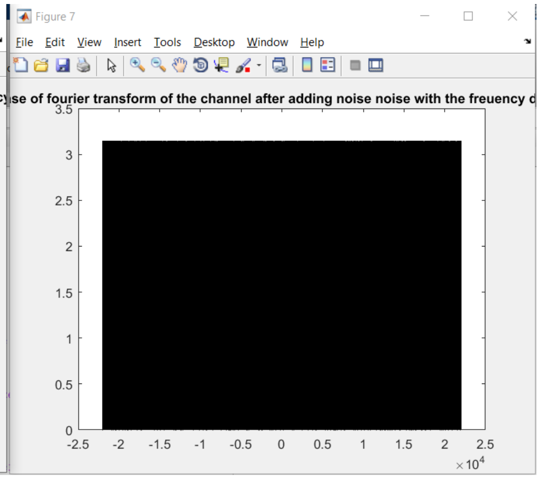
If we add the noise to the first channel with sigma= .005

The plot in time domain :



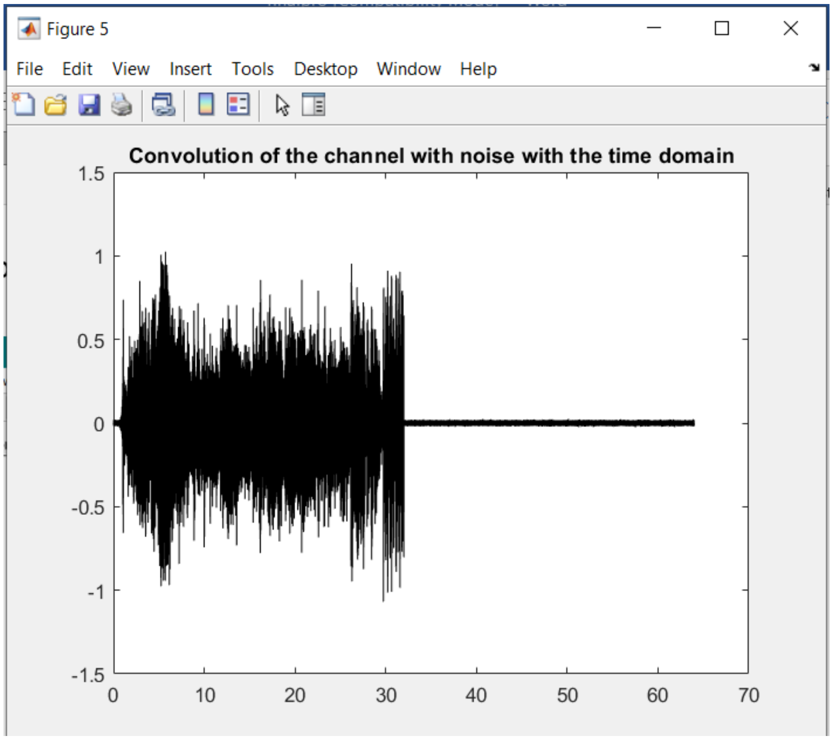
The plot in Frequency domain : Magnitude - Phase



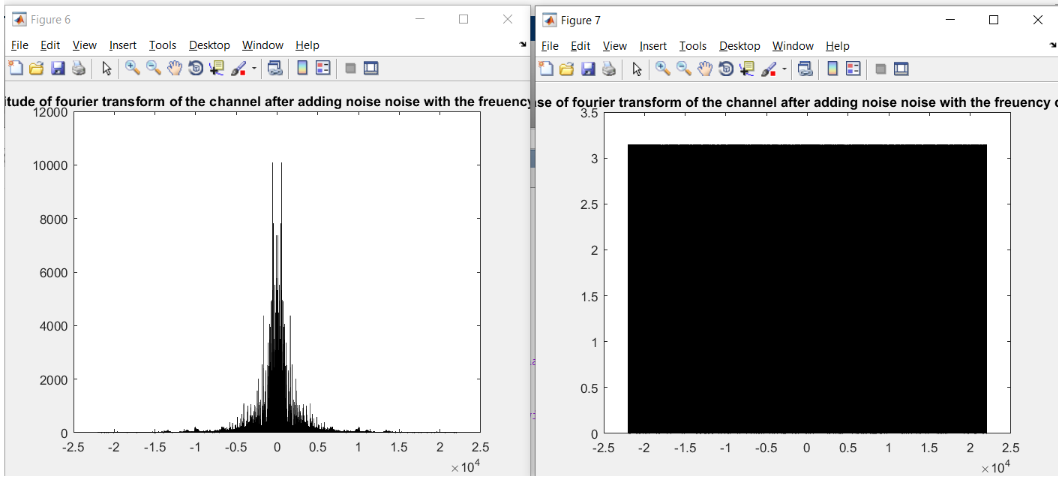


Noise(sigma = 0.005) to the second channel :

** The plot in time domain :

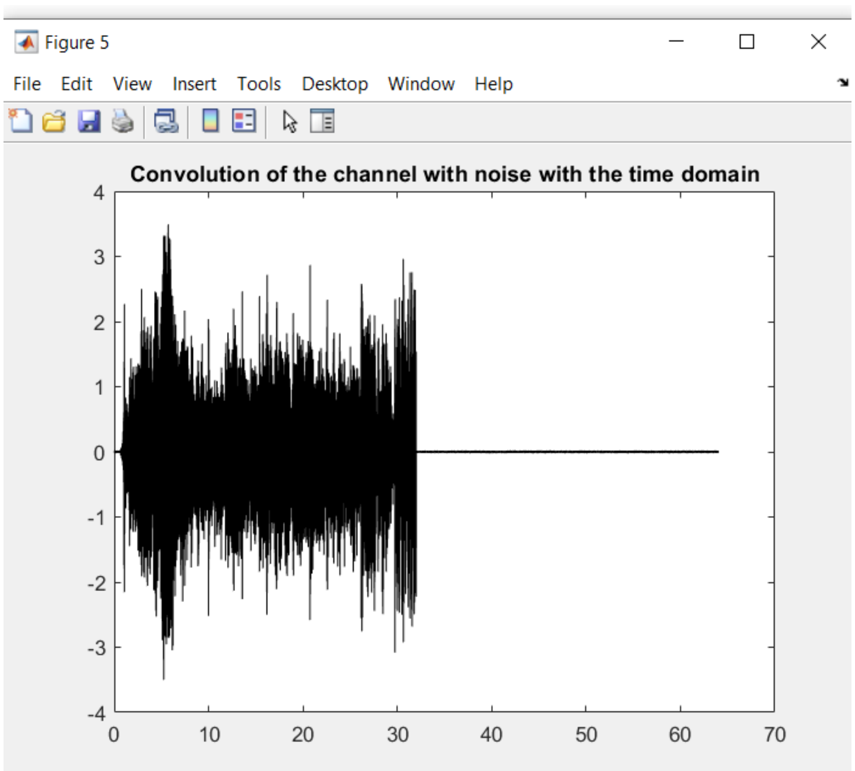


**The plot in Frequency domain: Magnitude - Phase

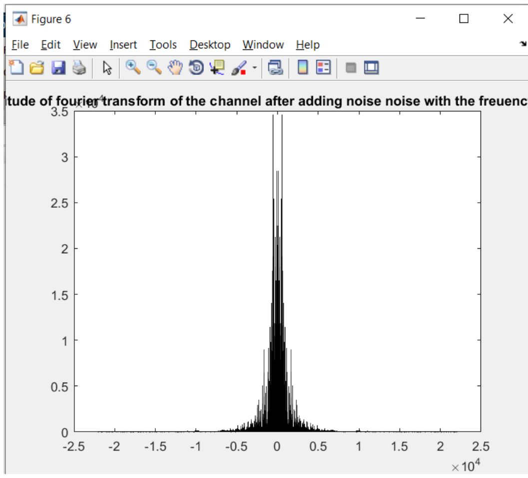


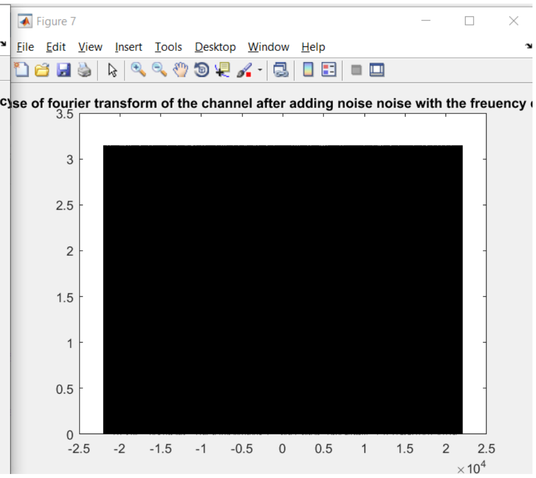
Noise(sigma = 0.005) to the third channel :

** The plot in time domain:



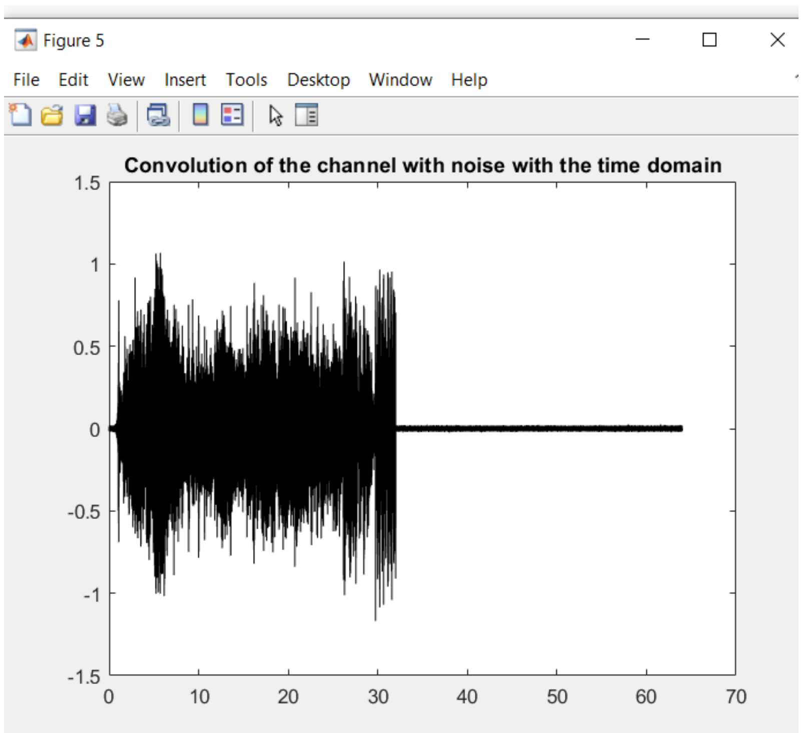
**The plot in Frequency domain: Magnitude - Phase



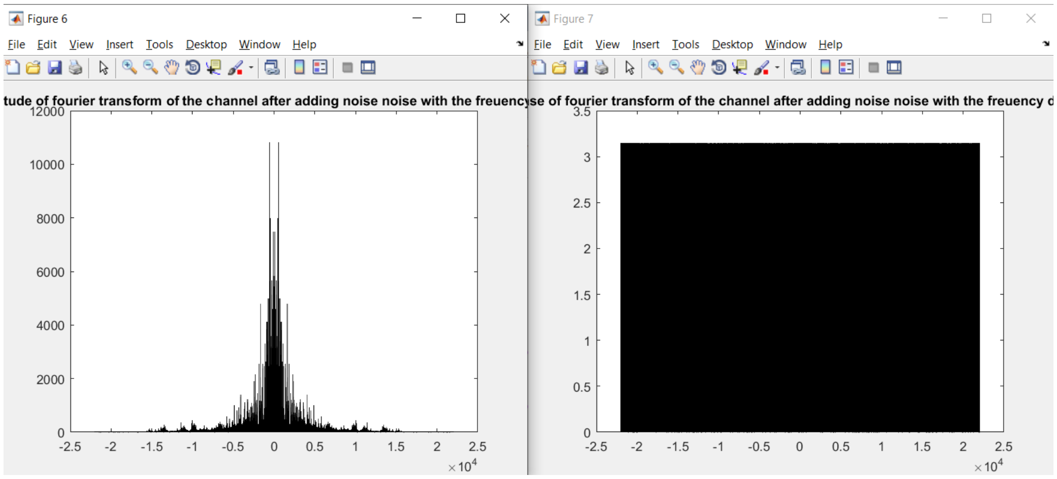


Noise(sigma = 0.005) to the last channel :

** The plot in time domain:



**The plot in Frequency domain: Magnitude - Phase



4- The last stage 'Receiver':
In this stage we will pass the noisy sound over the ideal filter and receive the new sound

```
if (m==1)
    v=zeros (1 , ((length (CH)) /2)-108800);
    vv= ones (1,217600);
else
    v=zeros (1 , ((length (CH)-1) /2)-108800);
    vv= ones (1,217601);
end

 H=[v vv v];
 CH = H .* CH;
 CHmag= abs(CH);
 CHphase= angle(CH);
 figure (7);
 plot (convF,CHmag,'g');
 title('Magnitude of fourier transform of the receiver signal with the frequncy domain','color','k')
 figure (8);
 plot (convF,CHphase,'y');
 title('Phase of fourier transform of the receiver signal with the frequncy domain','color','k');
 res=real(ifft(ifftshift(CH)));
 figure (9);
 plot (convT,res,'b');
 title('Convolution of the signal after filtering with the time domain','color','k');
 sound (res,fs);
 pause (10);
 clear sound;
```

We need to define frequency of graph the signal with the frequency so we divide 141120/44100= 32 so now we need to determine the number of samples of range from -3400 to 3400 it equal the difference * frequency of sampling = 6800*32 = 217600 we need half of them before zero and the rest after zero so we divide it by 2 =108800

Our idea to make a new range contain zero samples and ones . we need to put ones in the range will the filter let signal allow which equal 217600 and 108800 in the left and 108800 in the right of ones

We have 2 cases :

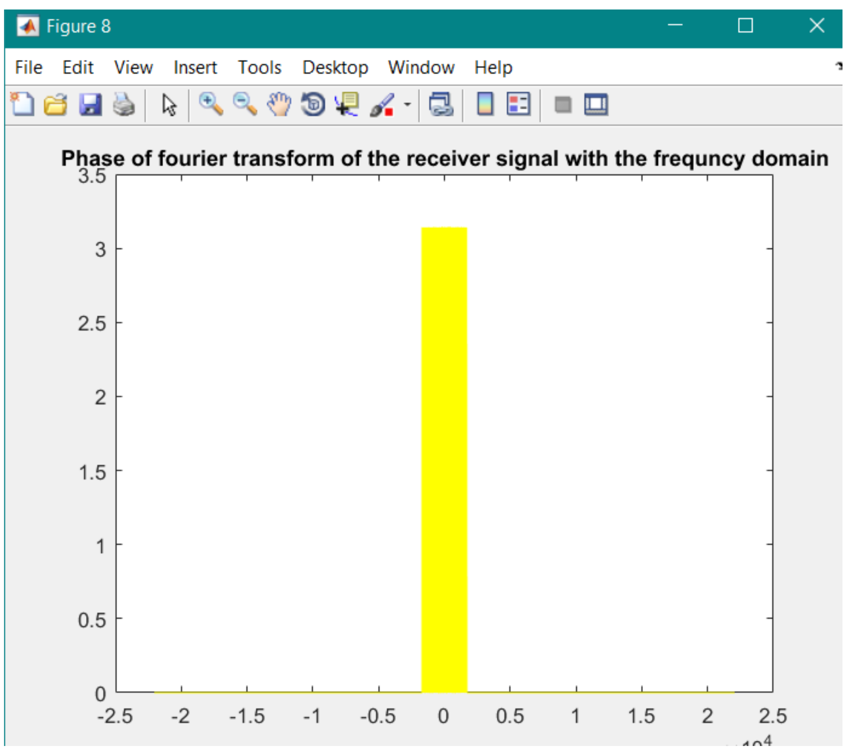
First: in the first channel we did convolution with one sample so number of samples of channel is even so we divide the length by 2 and subtract 108800 and put the ones in the center

Second: in the other channels we did convolution with number of samples equal the same of the original so we have odd samples (samples of original*2 -1) so we subtract 1 from the length (this sample we subtract at zero) now we have the samples in +ve and in –ve so we divide by 2 to get the half and subtract 108800 and put the ones in the center .

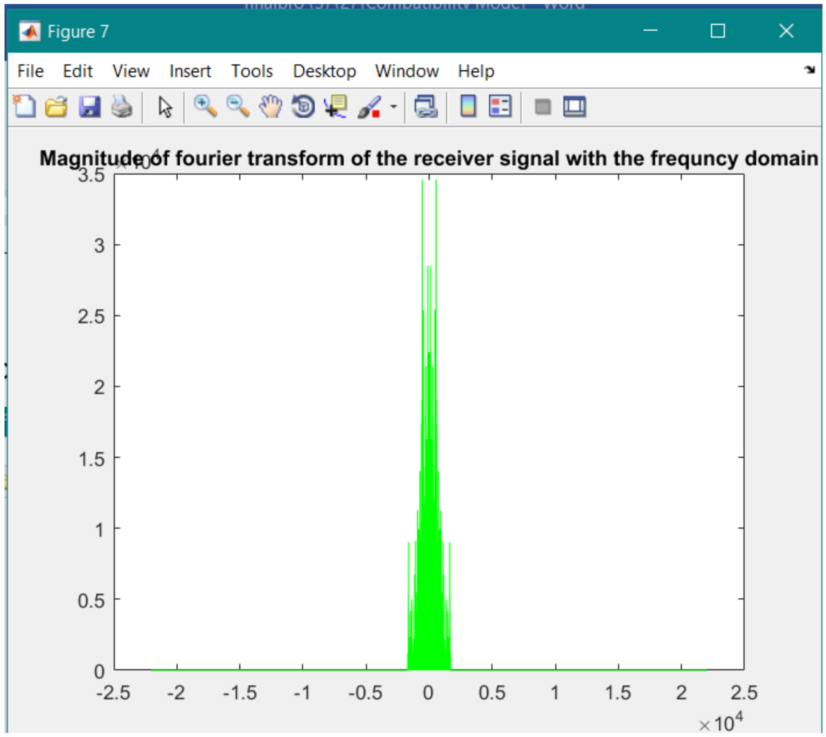
Now we have the new range

We multiply the signal by new range element by element so we pass the samples in range of -3400 to 3400

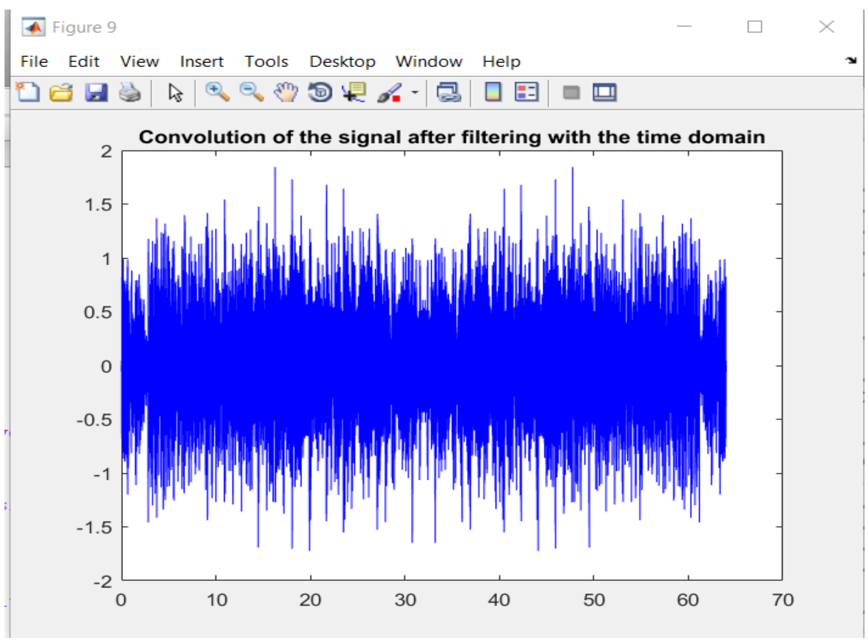
The Plot of the phase in frequency domain:



The Plot of the magnitude in frequency domain:



The Plot in the time domain



The code :

```
%transmitter
[y,fs]=audioread('project.mp3');
sound(y,fs);%playing the audio file
pause (10);
clear sound;%puase the audio after 10 seconds then clear it
y = y(1: 32*fs, 1);%to deal with mono signal
t=linspace ( 0 , 32 , 32*fs );%Time domain of the original signal
f=linspace( -fs/2 , fs/2 , length(y));%Frequency domain of the
original signal
% we create new linspace its range = (range of original sound*2)-1 :
convT= linspace ( 0 , 64 , 64*fs-1);
convF=  linspace( -fs/2 , fs/2 , length(convT));
x= fftshift(fft(y));%Fourier transform to the signal
figure (1);
plot(t, y,'y');
title('The original signal with the time domain','color','k');
ymag= abs(x);%Magnitude of fourier transform
yphase= angle (x);%Phase of fourier transform
figure (2);
plot(f , ymag,'m');
title('Magnitude of fourier transform of the original signal with the
frequncy domain','color','k');
figure (3);
plot(f, yphase,'b');
title('Phase of fourier transform of the original signal with the
frequncy domain','color','k');


%Channels
m=input('please enter th number the channel impulse response:');
y = transpose(y);

switch m
        case 1
            %Convolution using delta function
            convT=t;
            convF=f;
            x1=[1 zeros(1,length(y)-1)];
            x2=[y];
            X1=(fft(x1));
            X2=(fft(x2));
            Y= X1.*X2;
            CH=real(ifft(Y));
            figure (4);
            plot (convT,CH,'r');
            title('Convolution of the first channel with the time
domain','color','k');
```

```
            sound(CH,fs);
            pause (10);
            clear sound;
        case 2
           %Convolution using exponential (exp(-2*pi*5000*t))
function
           x1= exp(-2*pi*5000*t);
           x1=[x1 zeros(1,length (y)-1)];
           x2=[y  zeros(1,length (y)-1) ];
           X1=(fft(x1));
           X2=(fft(x2));
           Y= X1.*X2;
           CH=real(ifft((Y)));
           figure (4);
           plot (convT,CH,'r');
           title('Convolution of the second channel with the time
domain','color','k');
           sound(CH,fs);
           pause (10);
           clear sound;
       case 3
           %Convolution using exponential (exp(-2*pi*1000*t))
function
           x1= exp(-2*pi*1000*t);
           x1=[x1 zeros(1,length (y)-1)];
           x2=[y zeros(1,length (y)-1) ];
           X1=(fft(x1));
           X2=(fft(x2));
           Y= X1.*X2;
           CH=real(ifft((Y)));
           figure (4);
           plot (convT,CH,'r');
           title('Convolution of the third channel with the time
domain','color','k');
           sound(CH,fs);
           pause (10);
           clear sound;
      case 4
          %Convolution using two delta functions
           x1= 2*(t==0)+0.5*(t==1);
           x1=[x1 zeros(1,length (y)-1)];
           x2=[y zeros(1,length (y)-1) ];
           X1=(fft(x1));
           X2=(fft(x2));
           Y= X1.*X2;
           CH=real(ifft((Y)));
           figure (4);
           plot (convT,CH,'r');
```

```
           title('Convolution of the fourth channel with the time
domain','color','k');
           sound(CH,fs);

end




%Adding noise
 sigma=input('Please enter th value of sigma:'); %  STANDARD
DEVIATION
 z=sigma*randn(1,length(CH));%Noise function
 CH=CH+z;%Adding noise to the desired channel result
 sound (CH,fs);%Playing audio after adding noise
 pause (10);
 clear sound;
 figure(5);
 plot(convT,CH,'k');
 title('Convolution of the channel with noise with the time
domain','color','k');
 CH =real(fftshift (fft(CH)));
 CHnmag= abs(CH);
 CHnphase= angle(CH);
 figure (6);
 plot (convF,CHnmag,'k');
 title('Magnitude of fourier transform of the channel after adding
noise noise with the freuency domain','color','k');
 figure(7);
 plot (convF,CHnphase,'k');
 title('Phase of fourier transform of the channel after adding noise
noise with the freuency domain','color','k');
```

```
%Filter and receiver

 %we can obtain the freqency of graph ( our sound ,f ) =
141120/44100= 32 so
 %if we need to get number of samples in period -3400 to 3400 it
equal
 %6800*32 = 217600 we need half of them before zero and the other
after zero
 %so we divide by 2 =108800

if (m==1)
     v=zeros (1 , ((length (CH)) /2)-108800);
     vv= ones (1,217600);
else
     v=zeros (1 , ((length (CH)-1) /2)-108800);
     vv= ones (1,217601);
end

 H=[v vv v];
 CH = H .* CH;
 CHmag= abs(CH);
 CHphase= angle(CH);
 figure (8);
 plot (convF,CHmag,'g');
 title('Magnitude of fourier transform of the receiver signal with
the frequncy domain','color','k');
 figure (9);
 plot (convF,CHphase,'y');
 title('Phase of fourier transform of the receiver signal with the
frequncy domain','color','k');
 res=real(ifft(ifftshift(CH)));
 figure (10);
 plot (convT,res,'b');
 title('Convolution of the signal after filtering with the time
domain','color','k');
 sound (res,fs);
 pause (10);
 clear sound;
```