Faculty of Engineering, Alexandria University  
  
FM MATLAB Assignment

Name: Rania Hamada Muhammad ID: 83

First, we used MATLAB to read the attached audio file and get the spectrum of this signal

# Code

clc

clear

%%

%{

read audio file and find the spectrum

%}

fileName = 'eric.wav';

[y, Fs] = audioread(fileName); % read audio file y >> sampled data

sound(y, Fs); % play sound signal

% find the spectrum

Y = fftshift(fft(y));

Ymag = abs(Y); % magnitude spectrum

Yphase = angle(Y); % phase spectrum

Fvec = linspace(-Fs/2, Fs/2, length(Y)); % frquency vector

% plot the signal in time domain

figure;

subplot(4, 2, [1 2]); plot(y);

title('The message signal in time domain');

% plot the spectrum

subplot(4, 2, 3); plot(Fvec, Ymag);

title('Magnitude Spectrum of the message signal');

subplot(4, 2, 4); plot(Fvec, Yphase);

title('Phase Spectrum of the message signal');

# Output

Second, we used an ideal Filter, remove all frequencies greater than 4 KHz. Then Obtained the filtered signal in time domain.

# Code

%%

%{

apply an ideal low pass filter with BW = 4kHz

%}

SPHz = length(Y)/Fs; % sample per Hz

% low pass filter edges

lp\_edge1 = round(20000\*SPHz); % 20kHz = 24kHz - 4kHz

lp\_edge2 = round(length(Y)-(20000\*SPHz)+1);

Y([1:lp\_edge1 lp\_edge2:length(Y)]) = 0; % apply the low pass filter

Ymag\_filtered = abs(Y); % magnitude spectrum after LPF

Yphase\_filtered = angle(Y); % phase spectrum after LPF

% Plot spectrum after LPF

subplot(4, 2, 5); plot(Fvec, Ymag\_filtered);

title('Magnitude Spectrum of the message signal after LPF');

subplot(4, 2, 6); plot(Fvec, Yphase\_filtered);

title('Phase Spectrum of the message signal after LPF');

% obtain the filtered signal in time domain

y\_filtered = real(ifft(ifftshift(Y)));

% plot the signal in time domain after LPF

subplot(4, 2, [7 8]); plot(y\_filtered);

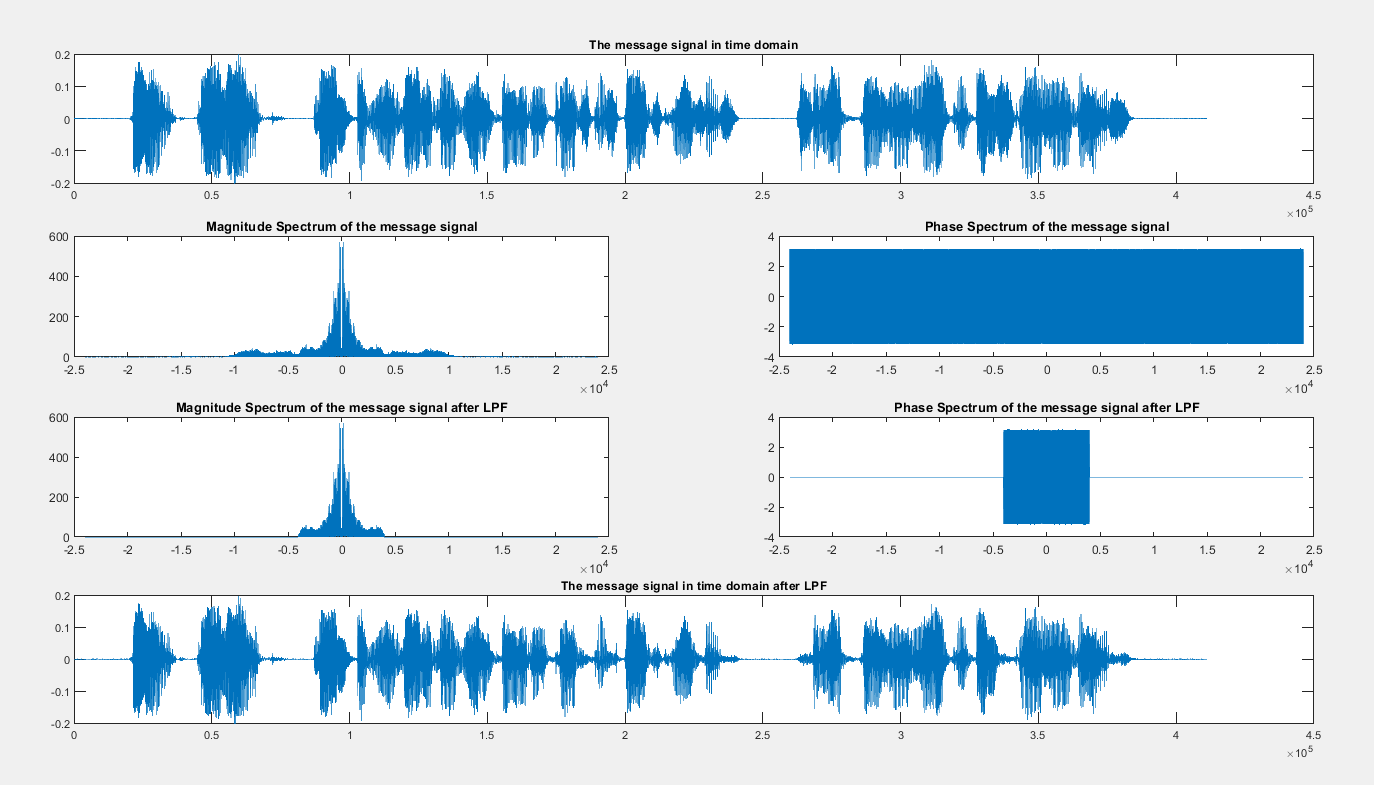
title('The message signal in time domain after LPF');

pause(length(y)/Fs); % wait until the sound stops

sound(y\_filtered, Fs);

# Output

# Overall output



Then, we generated the NBFM signal using a carrier frequency of 100kHz and a sampling

frequency of 𝐹𝑠 = 5𝐹𝑐.

# Code

%%

%{

Generate NBFM signal

fc = 100kHz

Fs\_NBFM = 5 \* fc

plot the resulting spectrum

%}

a = 10;

fc = 100000;

Fs\_NBFM = 5\*fc;

kf = 0.5; % use it small enough to get NBFM signal

wc = 2\*pi\*fc;

Ns = length(Y)\*Fs\_NBFM/Fs;

duration = Ns/Fs\_NBFM;

t = transpose(linspace(0, duration, Ns));

% resampling the filtered signal

y\_resampled = resample(y\_filtered, Fs\_NBFM, Fs);

% using general FM eqn: a \* cos((wc\*t) + (kf\*cumsum(y\_filtered)))

% using NBFM eqn:(a\*cos(wc\*t)) - (a \* kf \* sin(wc.\*t) .\* cumsum(y\_filtered))

% got the same output

s\_NBFM = a\*cos(wc\*t)-a\*kf\*sin(wc\*t).\*cumsum(y\_resampled(1:end-1));

% plot the modulated signal in time domain

figure;

subplot(2, 2, [1 2]);plot(s\_NBFM);

title('S\_N\_B\_F\_M(t): Modulated signal');

% find the spectrum of the modulated signal

S\_NBFM = fftshift(fft(s\_NBFM));

S\_NBFM\_mag = abs(S\_NBFM); % magnitude spectrum of the modulated signal

S\_NBFM\_phase = angle(S\_NBFM); % phase spectrum of the modulated signal

F\_NBFM\_vec = linspace(-Fs\_NBFM/2, Fs\_NBFM/2, length(S\_NBFM));

%plot the spectrum of the modulated signal

subplot(2, 2, 3); plot(F\_NBFM\_vec, S\_NBFM\_mag);

title('Magnitude Spectrum of S\_N\_B\_F\_M(f)');

subplot(2, 2, 4); plot(F\_NBFM\_vec, S\_NBFM\_phase);

title('Phase Spectrum of S\_N\_B\_F\_M(f)');

# Output

Here we noticed that the modulated signal is like the DSB-TC signal with   
bandwidth = 2\*fm. And we could use discriminator detector to detect the FM signal in simple and cheapest way. And this can be done with a differentiator and an envelope detector as following:

# Code

%%

%{

Demodulation

%}

figure;

subplot(3, 2, 1); plot(y);

title('Message Signal');

subplot(3, 2, 2); plot(s\_NBFM);

title('Modulated Signal');

% apply envelope detecttion

env\_det\_s = sqrt(a^2 + (a\*kf\*cumsum(y\_resampled(1:end-1)).^2));

subplot(3, 2, 3); plot(env\_det\_s);

title('Demodulated Signal after envelope detector');

diff\_s = diff(env\_det\_s); % differentiation of env\_det\_s

subplot(3, 2, 4); plot(diff\_s);

title('Diff of the Demodulated Signal after envelope detector');

signal = resample(diff\_s, Fs, Fs\_NBFM);

subplot(3, 2, 5); plot(signal);

title('Demodulated Signal');

%

% amplification

signal = signal\*kf;

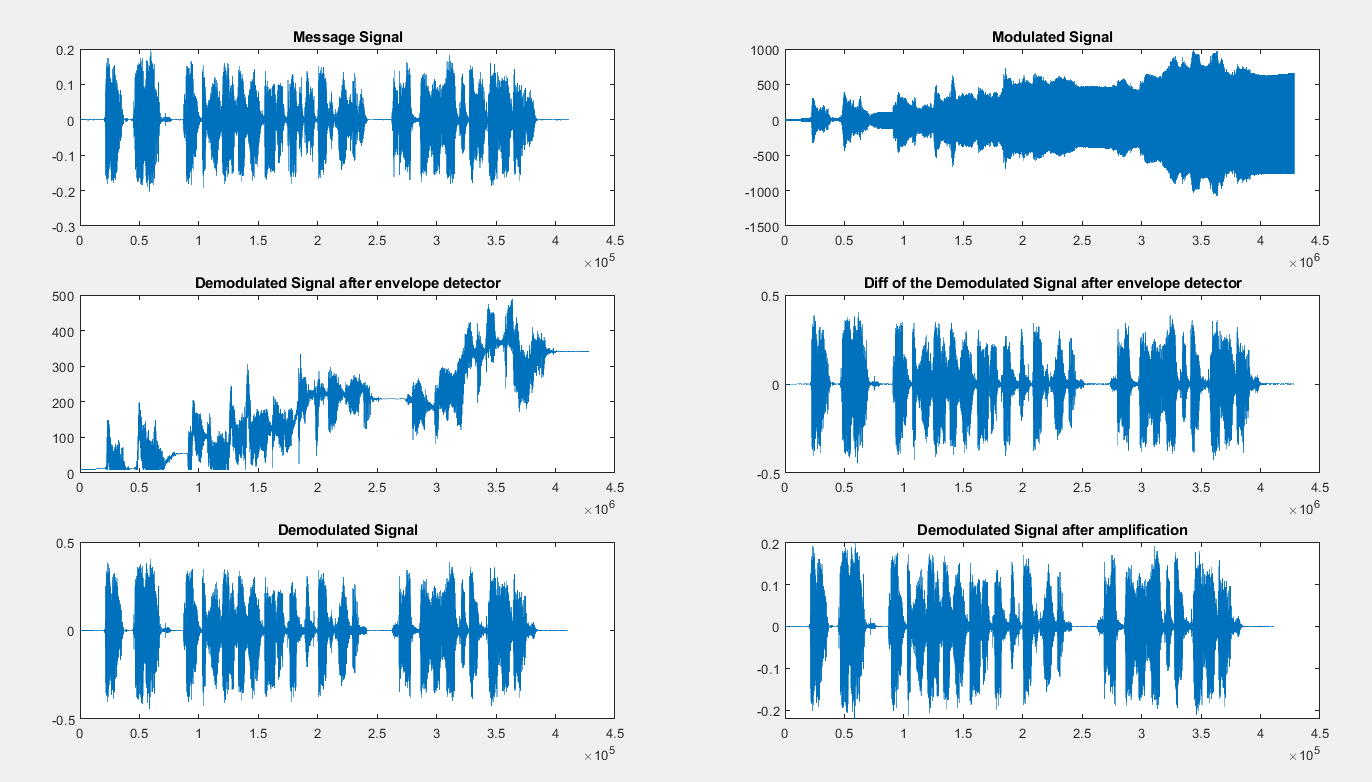
subplot(3, 2, 6); plot(signal);

title('Demodulated Signal after amplification');

pause(length(y)/Fs);

sound(signal, Fs);

# Output



## Conclusion

At first we read the sound signal and got the spectrum of the signal. Then we apply a LPF with bandwidth = 4kHz to remove high frequencies and when we play the sound we feel that the noise is lower than before.

Then we generated the FM signal and reduce the value of kf to get a NBFM signal and when plotting its spectrum, we found it like DSB-TC so we used ED to detect the FM signal and differentiate it later then amplify it to get the message signal.