FINAL PROJECT REPORT

Mohammad Yasser Zaki 6510

Omar Sherif Almas 6292

Karim Mohamed El-Baroudy 6424

Ahmed Saad Ahmed Mohamed   6060

Omar Hossam El-Din Ebraheem 6667

2021-2022

Part 1:
1.1 Code:

~~~
```

clear
clc
%% reading audio
% reading audio
[y,Fs] = audioread('eric.wav');

% fft application
L = length(y); % measure length of function for generation of frequency axis
Y = fftshift(fft(y)); % transform and shift the function
f = (-L/2:L/2-1)*(Fs/L); % generate frequency axis

% plotting the spectrum
plot(f,Y); % plot in frequency domain
title('Frequency Spectrum of eric.wav')
xlabel('Frequency/Hz')
figure;
%% filtering




% filtering
Yfiltered = Y;
Yfiltered(abs(f)>4000) = 0; % simulating ideal lpfilter set at 4000 Hz

% reversing fft and testing
yfiltered = ifft(ifftshift(Yfiltered));

% sounding filtered signal
sound(y,Fs);
pause(9);




%% modulation
% % resampling
yfilresampled = resample(yfiltered,125,12); % resampling to make sampling
% frequency = carrier frequency times 5

% % modulation
fc = 100000; % set carrier frequency
tcar = 0:1/(fc*5):8.567668; % generate carrier time tupple with overall
% function sampling frequency from 0 to length of audio tupple
tcar(end) = []; % remove final 'extra value'
carrier = cos(2*pi*fc*tcar)'; % generate carrier tupple

% % % DSB-SC
subplot(2,1,1);
~~~

```
sc = yfilresampled.*carrier; % multiply carrier and filtered+resampled function tupple
Ld = length(sc); % calculate legnth of modulated wave
SC = fftshift(fft(sc));
fam = (-Ld/2:Ld/2-1)*((5*fc)/Ld);

plot(fam,SC); % plot DSB-SC spectrum
title('Frequency Spectrum of DSB-SC')
xlabel('Frequency/Hz')

% % % DSB-TC
subplot(2,1,2);

M = max(abs(yfilresampled)); % find amplitude of sound function
tc = (2*M + yfilresampled).*carrier; % generate modulated DSB-TC wave
TC = fftshift(fft(tc));

plot(fam,TC); % plot DSB-TC spectrum
title('Frequency Spectrum of DSB-TC')
xlabel('Frequency/Hz')
figure;

subplot(2,1,1);
% % DSBSC envelope detections
scE = abs(hilbert(sc)); % envelope detection formula
plot(tcar, scE); % plot envelope in time domain
title('DSBSC envelope')
xlabel('Time/s')

subplot(2,1,2);
% % DSBTC envelope detection
tcE = abs(hilbert(tc)); % envelope detection formula
plot(tcar, tcE); % plot envelope in time domain
title('DSBTC envelope')
xlabel('Time/s')
figure

%% resampling and sounding

tcR = resample(tcE,12,125);
sound(tcR,Fs)
pause(9);
scR = resample(scE,12,125);
sound(scR,Fs)
pause(9);

%% coherent detection for DSB-SC

scWN0 = awgn(sc,0); % adds white noise with SNR = 0 then SNR = 10 then SNR = 30
scWN10 = awgn(sc,10);
scWN30 = awgn(sc,30);

% % dsbsc with snr = 0
s0 = scWN0.*carrier;
S0 = fftshift(fft(s0));
L = length(s0);
f = (-L/2:L/2-1)*(500000/L);
S0(abs(f)>1.5*fc) = 0;
```

```
s0 = ifft(ifftshift(S0));

s0R = resample(s0,12,125);
sound(s0R,Fs)
pause(9);

subplot(2,1,1)
plot(tcar,s0)
title('Coherent Detection SNR 0 waveform')
xlabel('Time/s')
subplot(2,1,2)
plot(f,S0)
title('Coherent Detection SNR 0 spectrum')
xlabel('Frequency/Hz')
figure

% % dsbsc with snr = 10
s10 = scWN10.*carrier;
S10 = fftshift(fft(s10));
S10(abs(f)>1.5*fc) = 0;
s10 = ifft(ifftshift(S10));

s10R = resample(s10,12,125);
sound(s10R,Fs)
pause(9);

subplot(2,1,1)
plot(tcar,s10)
title('Coherent Detection SNR 10 waveform')
xlabel('Time/s')
subplot(2,1,2)
plot(f,S10)
title('Coherent Detection SNR 10 spectrum')
xlabel('Frequency/Hz')
figure

% % dsbsc with snr = 30
s30 = scWN30.*carrier;
S30 = fftshift(fft(s30));
S30(abs(f)>1.5*fc) = 0;
s30 = ifft(ifftshift(S30));

s30R = resample(s30,12,125);
sound(s30R,Fs)
pause(9);

subplot(2,1,1)
plot(tcar,s30)
title('Coherent Detection SNR 30 waveform')
xlabel('Time/s')
subplot(2,1,2)
plot(f,S30)
title('Coherent Detection SNR 30 spectrum')
xlabel('Frequency/Hz')
figure

%% Error
% % with freq error 100.1 KH
fcf = 100100; % redefining carrier frequency
```

```
carrier = cos(2*pi*fcf*tcar)';
% % dsbsc with snr = 0
se0 = scWN0.*carrier;
Se0 = fftshift(fft(se0));
Se0(abs(f)>1.5*fc) = 0;
se0 = ifft(ifftshift(Se0));

se0R = resample(se0,12,125);
sound(se0R,Fs)
pause(9);

subplot(2,1,1)
plot(tcar,se0)
title('Coherent Detection SNR 0 waveform and freq error')
xlabel('Time/s')
subplot(2,1,2)
plot(f,Se0)
title('Coherent Detection SNR 0 spectrum and freq error')
xlabel('Frequency/Hz')
figure

% error calculation for snr = 0
errorArr0 = se0./s0;
plot(tcar,errorArr0);
title('Freq error for SNR = 0')
xlabel('Time/s')
figure

% % dsbsc with snr = 10
se10 = scWN10.*carrier;
Se10 = fftshift(fft(se10));
Se10(abs(f)>1.5*fc) = 0;
se10 = ifft(ifftshift(Se10));

se10R = resample(se10,12,125);
sound(se10R,Fs)
pause(9);

subplot(2,1,1)
plot(tcar,se10)
title('Coherent Detection SNR 10 waveform and freq error')
xlabel('Time/s')
subplot(2,1,2)
plot(f,Se10)
title('Coherent Detection SNR 10 spectrum and freq error')
xlabel('Frequency/Hz')
figure

% error calculation for snr = 10
errorArr10 = se10./s10;
plot(tcar,errorArr10);
title('Freq error for SNR = 10')
xlabel('Time/s')
figure

% % dsbsc with snr = 30
se30 = scWN30.*carrier;
Se30 = fftshift(fft(se30));
Se30(abs(f)>1.5*fc) = 0;
```

```
se30 = ifft(ifftshift(Se30));

se30R = resample(se30,12,125);
sound(se30R,Fs)
pause(9);

subplot(2,1,1)
plot(tcar,se30)
title('Coherent Detection SNR 30 waveform with freq error')
xlabel('Time/s')
subplot(2,1,2)
plot(f,Se30)
title('Coherent Detection SNR 30 spectrum with freq error')
xlabel('Frequency/Hz')
figure

% error calculation for snr = 30
errorArr30 = se30./s30;
plot(tcar,errorArr30);
title('Freq error for SNR = 30')
xlabel('Time/s')
figure

%% Error
% % with phase error 20
carrier = cos(2*pi*fc*tcar + 20)'; % generating carrier again with phase error
% % dsbsc with snr = 0
sp0 = scWN0.*carrier;
Sp0 = fftshift(fft(sp0));
L = length(sp0);
f = (-L/2:L/2-1)*(500000/L);
Sp0(abs(f)>1.5*fc) = 0;
sp0 = ifft(ifftshift(Sp0));

sp0R = resample(sp0,12,125);
sound(sp0R,Fs)
pause(9);

subplot(2,1,1)
plot(tcar,sp0)
title('Coherent Detection SNR 0 waveform with phase error')
xlabel('Time/s')
subplot(2,1,2)
plot(f,Sp0)
title('Coherent Detection SNR 0 spectrum with phase error')
xlabel('Frequency/Hz')
figure



% % dsbsc with snr = 10
sp10 = scWN10.*carrier;
Sp10 = fftshift(fft(sp10));
L = length(sp10);
f = (-L/2:L/2-1)*(500000/L);
Sp10(abs(f)>1.5*fc) = 0;
sp10 = ifft(ifftshift(Sp10));

sp10R = resample(sp10,12,125);
sound(sp10R,Fs)
```

```
pause(9);

subplot(2,1,1)
plot(tcar,sp10)
title('Coherent Detection SNR 10 waveform with phase error')
xlabel('Time/s')
subplot(2,1,2)
plot(f,Sp10)
title('Coherent Detection SNR 10 spectrum with phase error')
xlabel('Frequency/Hz')
figure

% % dsbsc with snr = 30
sp30 = scWN30.*carrier;
Sp30 = fftshift(fft(sp30));
L = length(sp30);
f = (-L/2:L/2-1)*(500000/L);
Sp30(abs(f)>1.5*fc) = 0;
sp30 = ifft(ifftshift(Sp30));

sp30R = resample(sp30,12,125);
sound(sp30R,Fs)

subplot(2,1,1)
plot(tcar,sp30)
title('Coherent Detection SNR 30 waveform with phase error')
xlabel('Time/s')
subplot(2,1,2)
plot(f,Sp30)
title('Coherent Detection SNR 30 spectrum with phase error')
xlabel('Frequency/Hz')


```
```
```

1.2 plots:

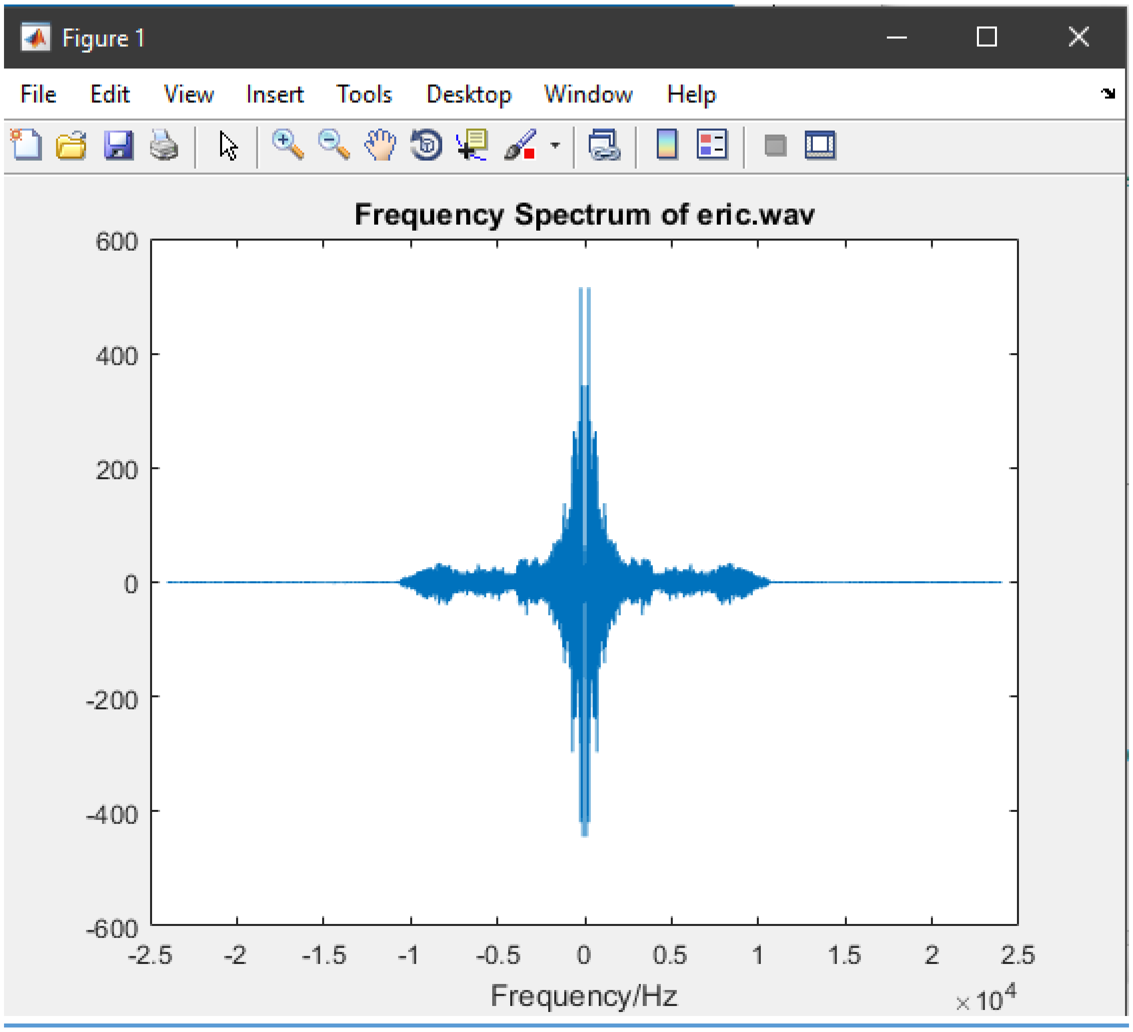


Figure 1
File
Edit
View
Insert
Tools
Desktop
Window
Help
Frequency Spectrum of eric.wav
600
400
200
0
-200
-400
-600
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Frequency/Hz
$\times 10^4$

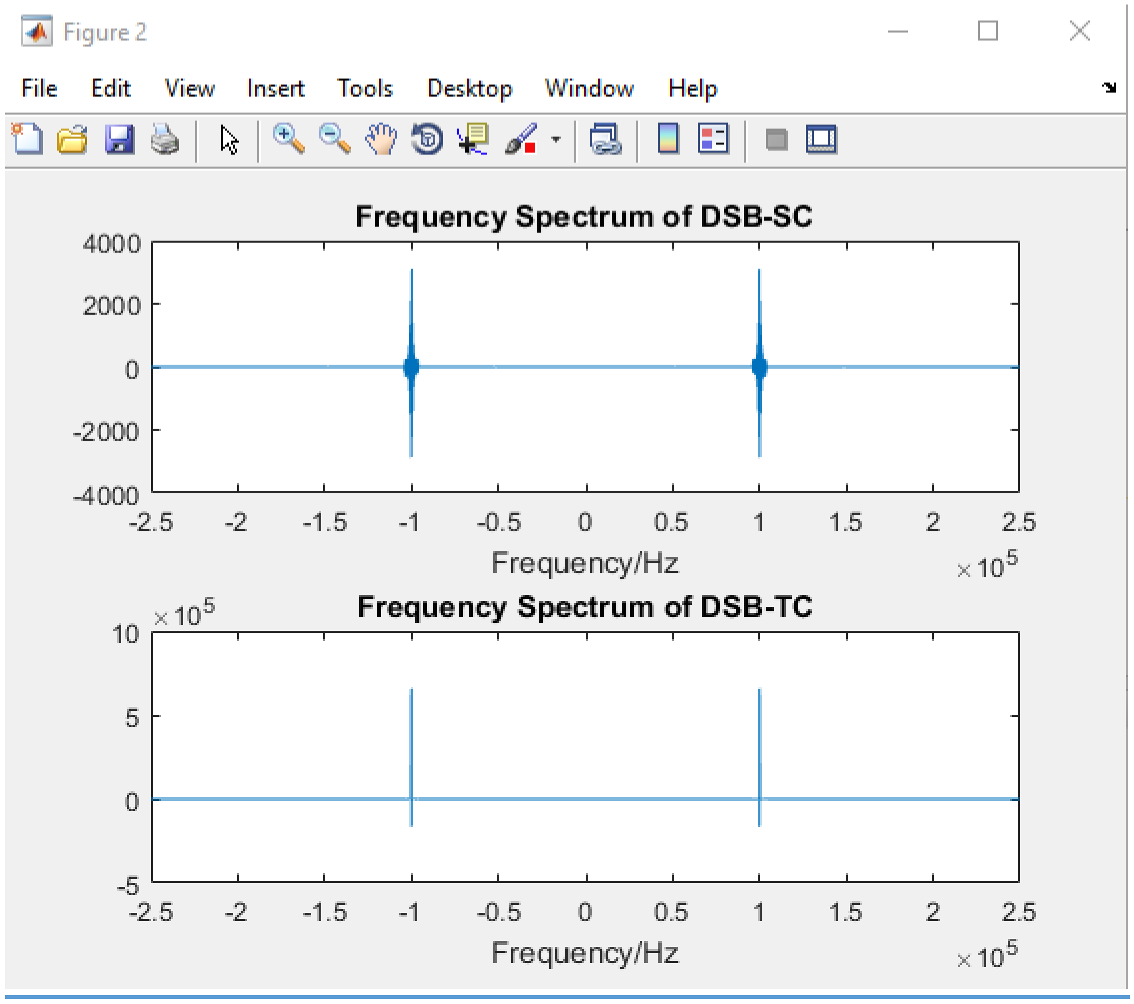


Figure 2
File Edit View Insert Tools Desktop Window Help
Frequency Spectrum of DSB-SC
4000
2000
0
-2000
-4000
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5
Frequency/Hz
×10^5
Frequency Spectrum of DSB-TC
×10^5
10
5
0
-5
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5
Frequency/Hz
×10^5

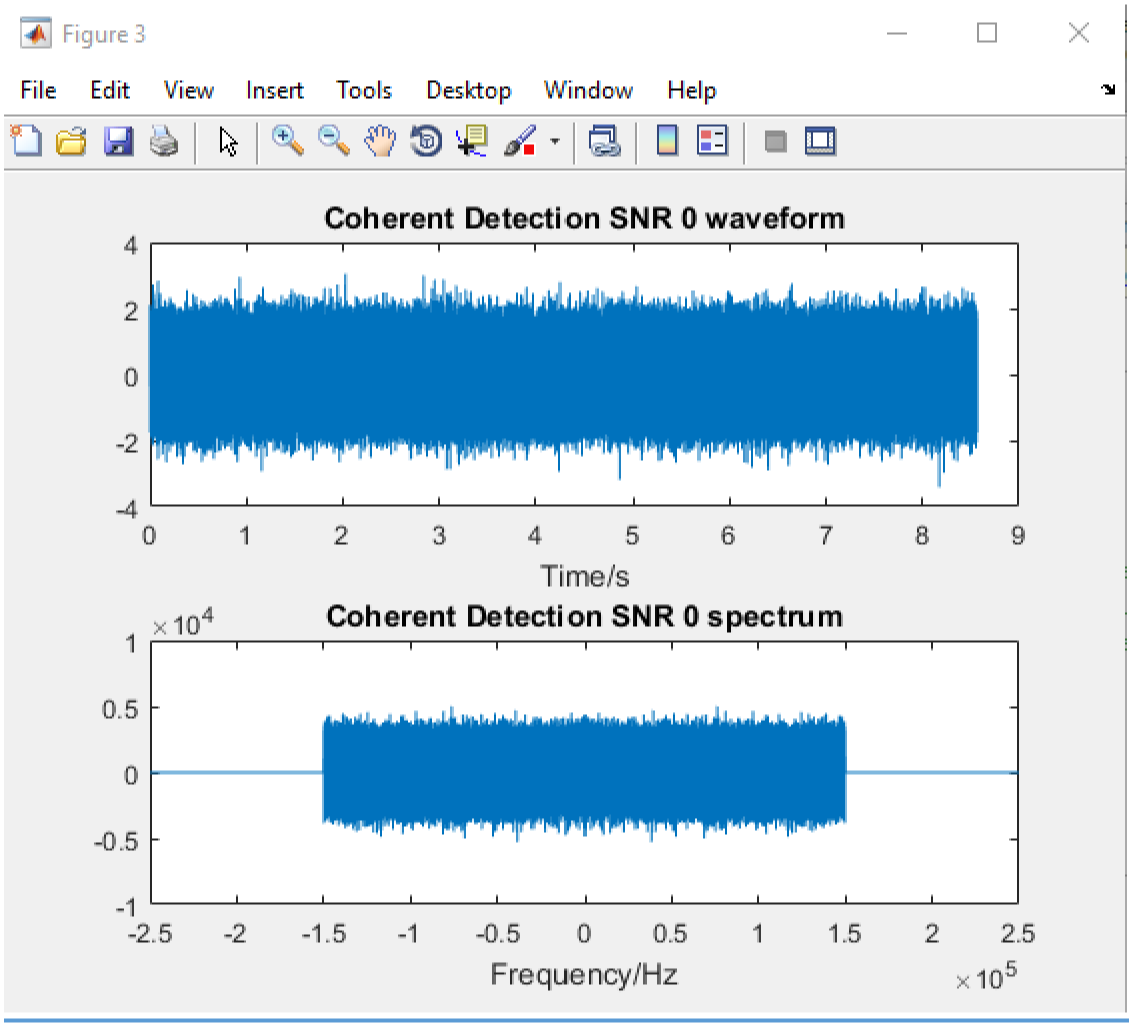


Figure 3
File
Edit
View
Insert
Tools
Desktop
Window
Help
Coherent Detection SNR 0 waveform
4
2
0
-2
-4
0
1
2
3
4
5
6
7
8
9
Time/s
Coherent Detection SNR 0 spectrum
× 10^4
1
0.5
0
-0.5
-1
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Frequency/Hz
× 10^5

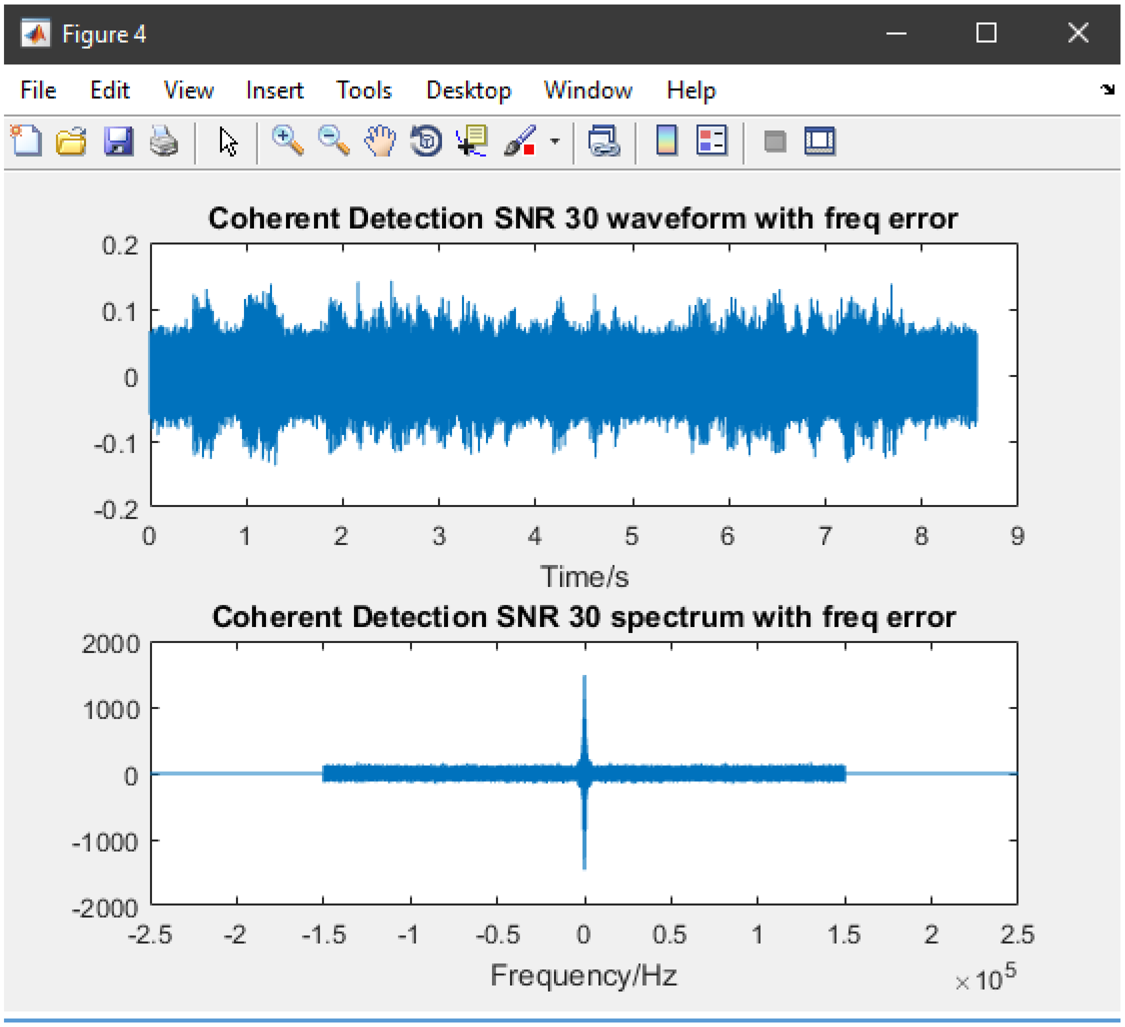


Figure 4
File Edit View Insert Tools Desktop Window Help
Coherent Detection SNR 30 waveform with freq error
0.2
0.1
0
-0.1
-0.2
0 1 2 3 4 5 6 7 8 9
Time/s
Coherent Detection SNR 30 spectrum with freq error
2000
1000
0
-1000
-2000
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5
Frequency/Hz
× 10^5

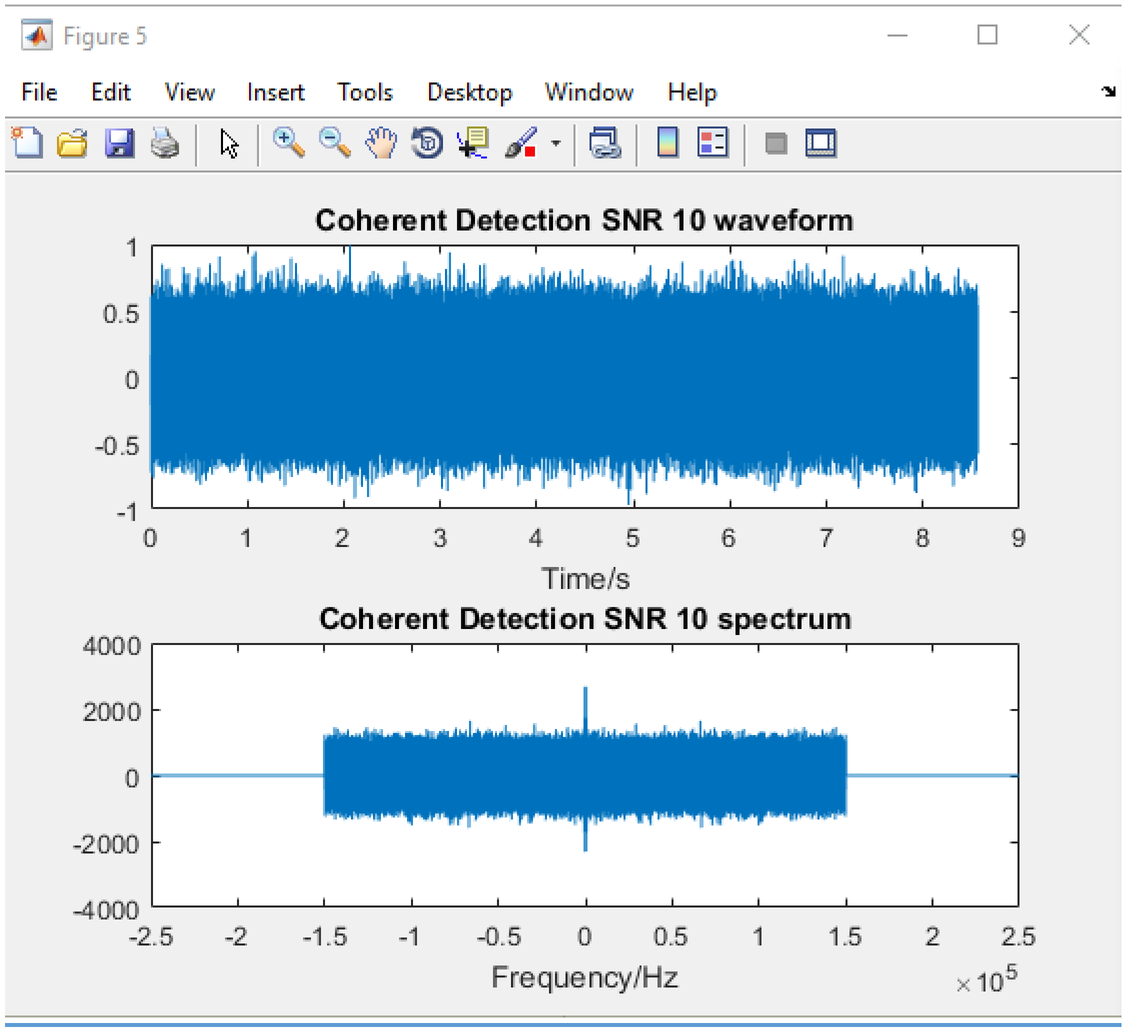


Figure 5
File
Edit
View
Insert
Tools
Desktop
Window
Help
Coherent Detection SNR 10 waveform
1
0.5
0
-0.5
-1
0
1
2
3
4
5
6
7
8
9
Time/s
Coherent Detection SNR 10 spectrum
4000
2000
0
-2000
-4000
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Frequency/Hz
$\times 10^5$

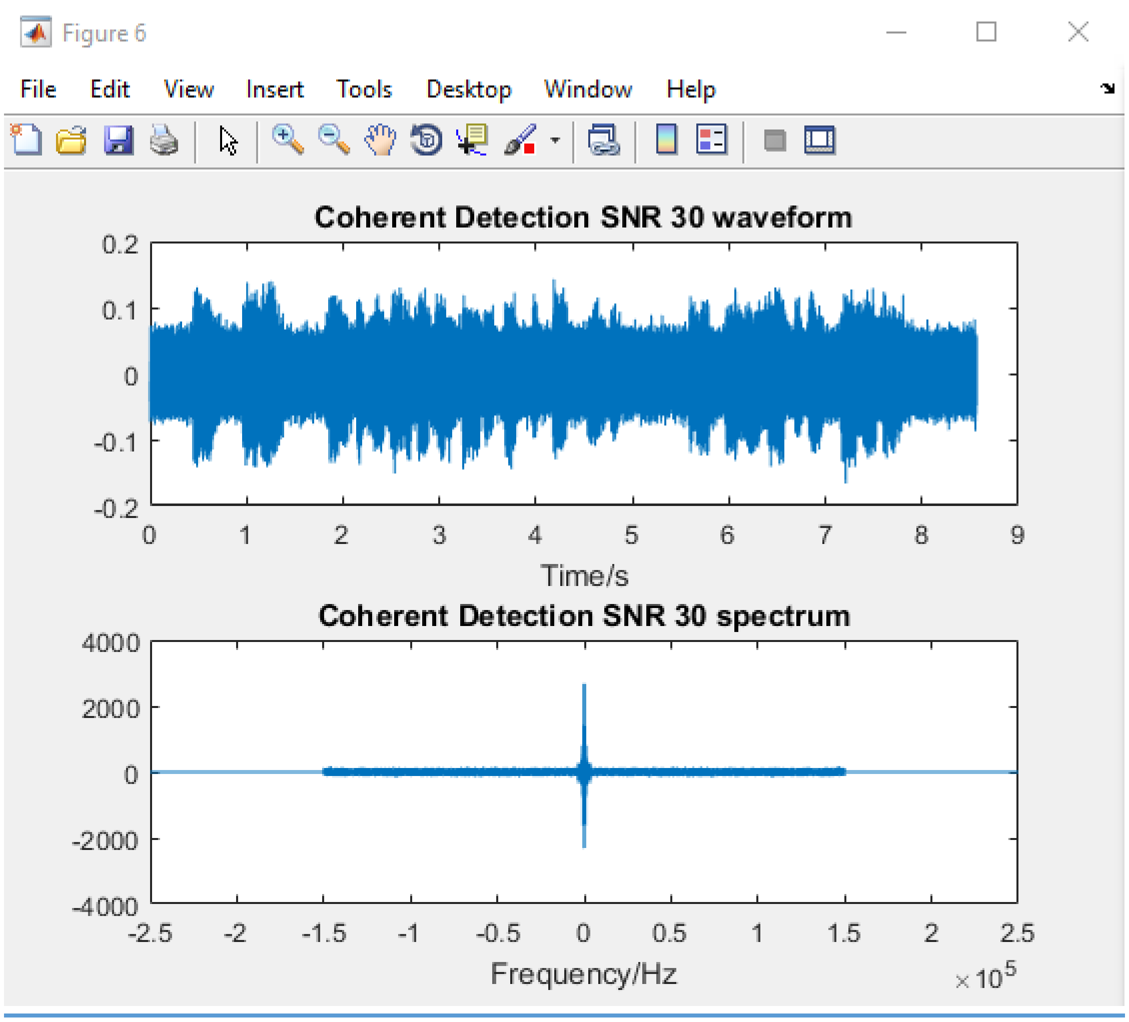


Figure 6
File
Edit
View
Insert
Tools
Desktop
Window
Help
Coherent Detection SNR 30 waveform
Time/s
Coherent Detection SNR 30 spectrum
Frequency/Hz
$\times 10^5$

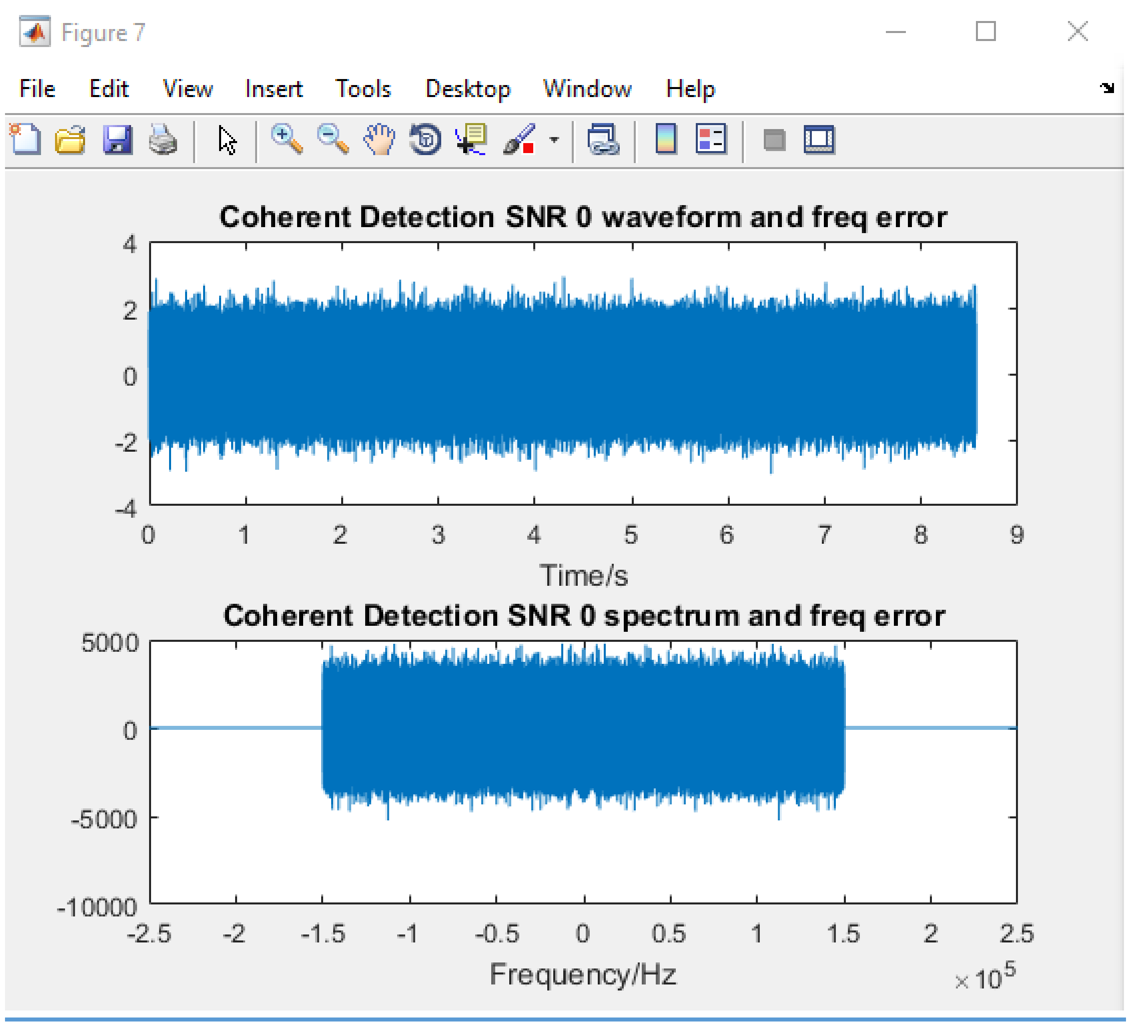


Figure 7
File Edit View Insert Tools Desktop Window Help
Coherent Detection SNR 0 waveform and freq error
4
2
0
-2
-4
0 1 2 3 4 5 6 7 8 9
Time/s
Coherent Detection SNR 0 spectrum and freq error
5000
0
-5000
-10000
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5
Frequency/Hz
$\times 10^5$

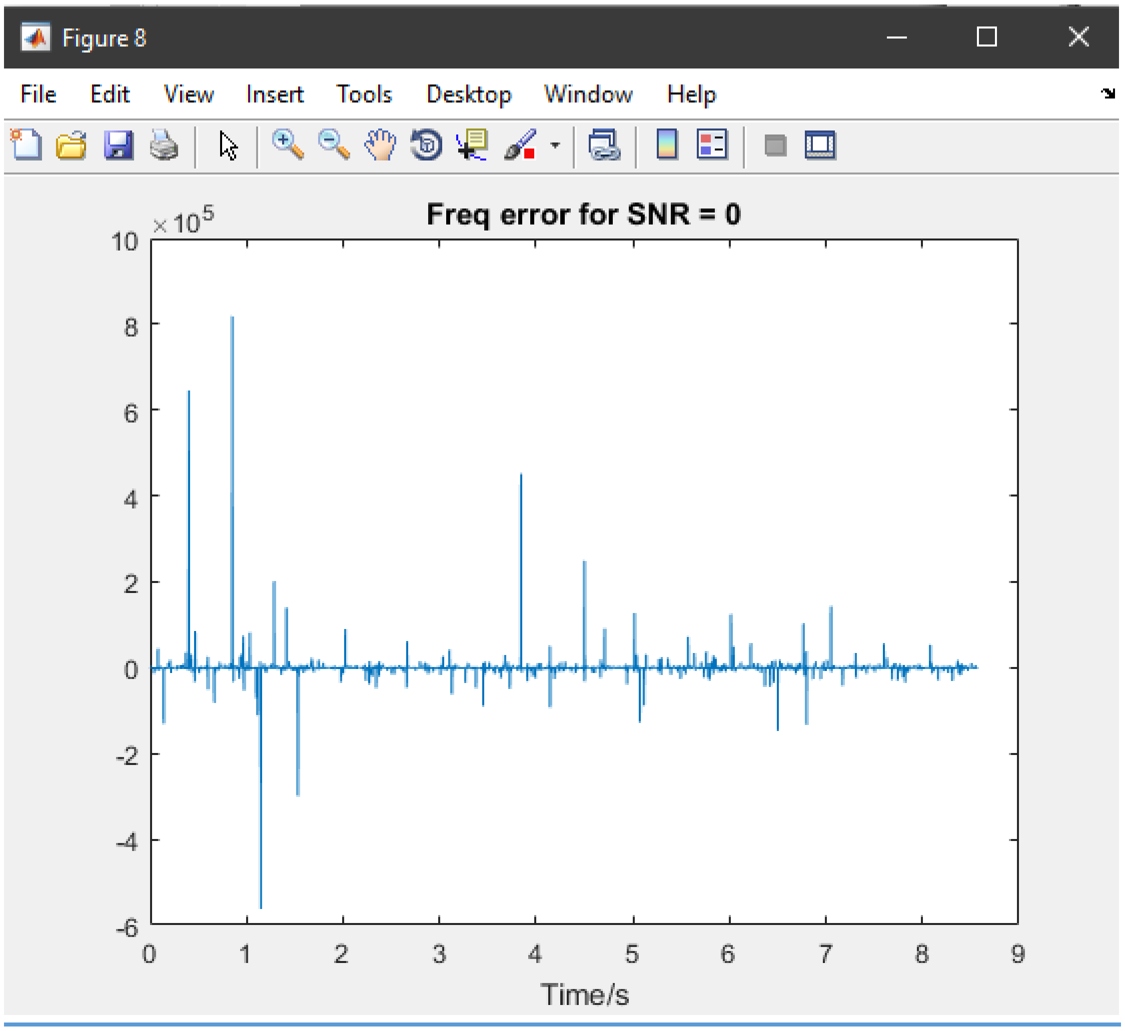


Figure 8
File Edit View Insert Tools Desktop Window Help
Freq error for SNR = 0
$\times 10^5$
10
8
6
4
2
0
-2
-4
-6
0 1 2 3 4 5 6 7 8 9
Time/s

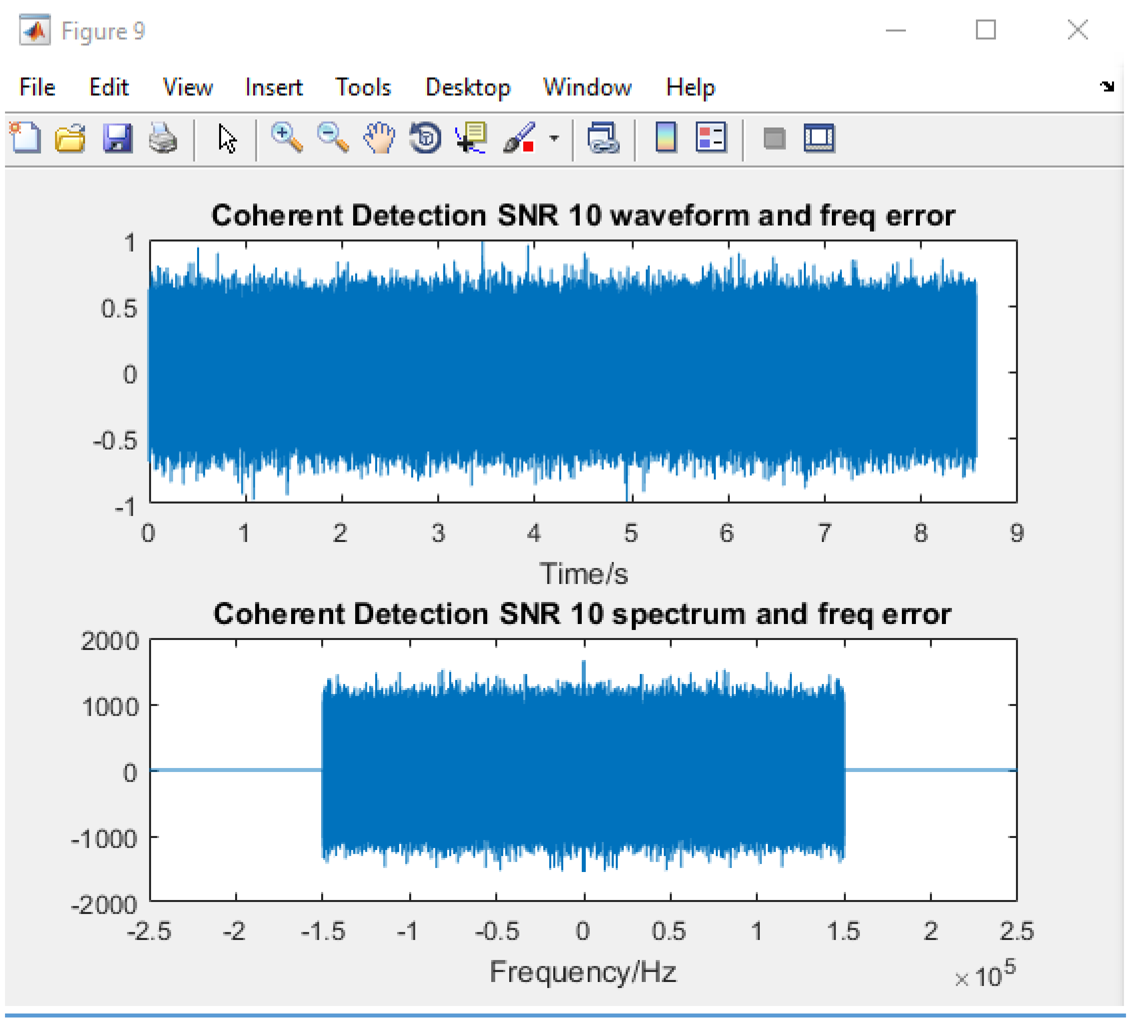


Figure 9
File Edit View Insert Tools Desktop Window Help
Coherent Detection SNR 10 waveform and freq error
Time/s
Coherent Detection SNR 10 spectrum and freq error
Frequency/Hz
×10^5

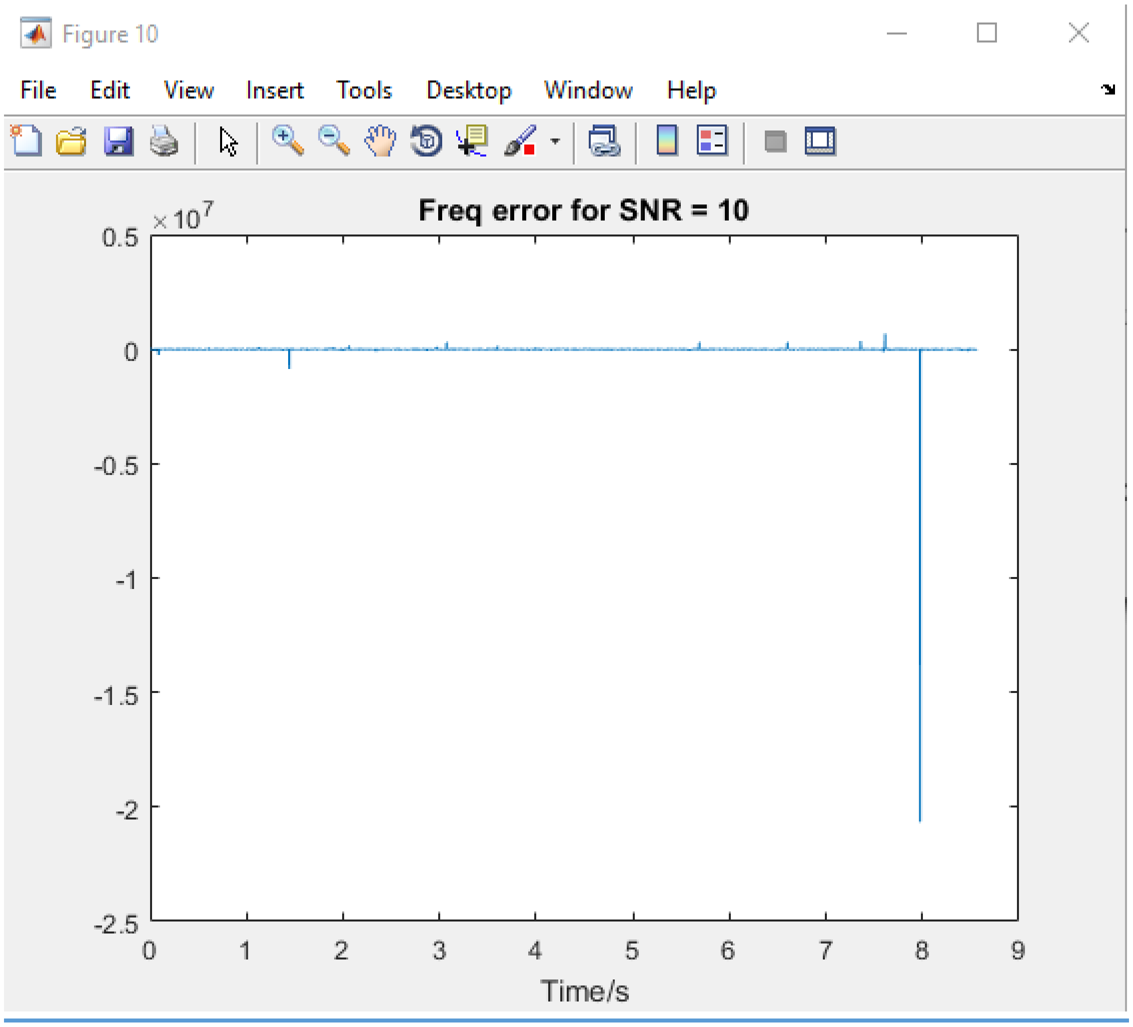


Figure 10
File
Edit
View
Insert
Tools
Desktop
Window
Help
Freq error for SNR = 10
$\times 10^7$
0.5
0
-0.5
-1
-1.5
-2
-2.5
0
1
2
3
4
5
6
7
8
9
Time/s

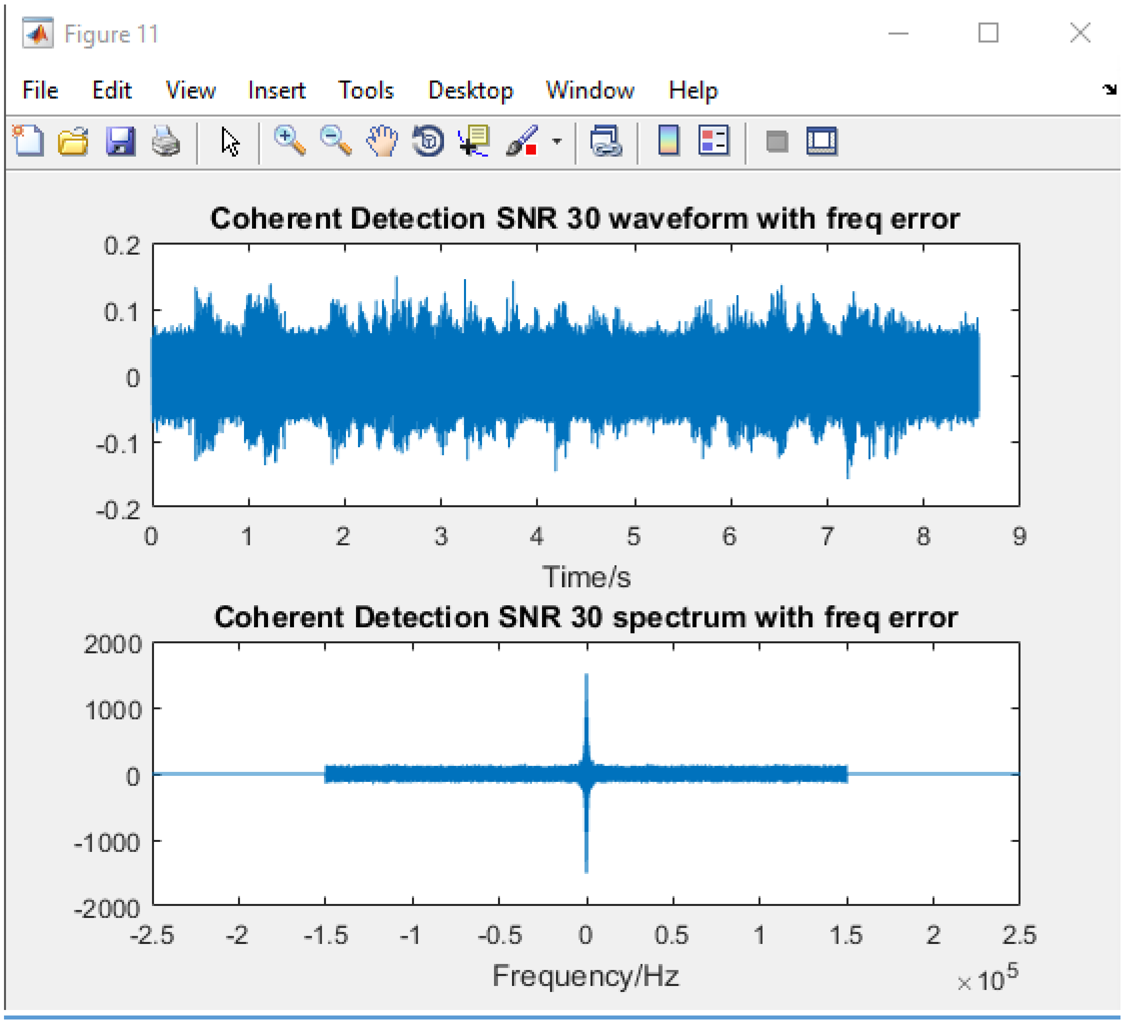


Figure 11
File Edit View Insert Tools Desktop Window Help
Coherent Detection SNR 30 waveform with freq error
0.2
0.1
0
-0.1
-0.2
0 1 2 3 4 5 6 7 8 9
Time/s
Coherent Detection SNR 30 spectrum with freq error
2000
1000
0
-1000
-2000
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5
Frequency/Hz
$\times 10^5$

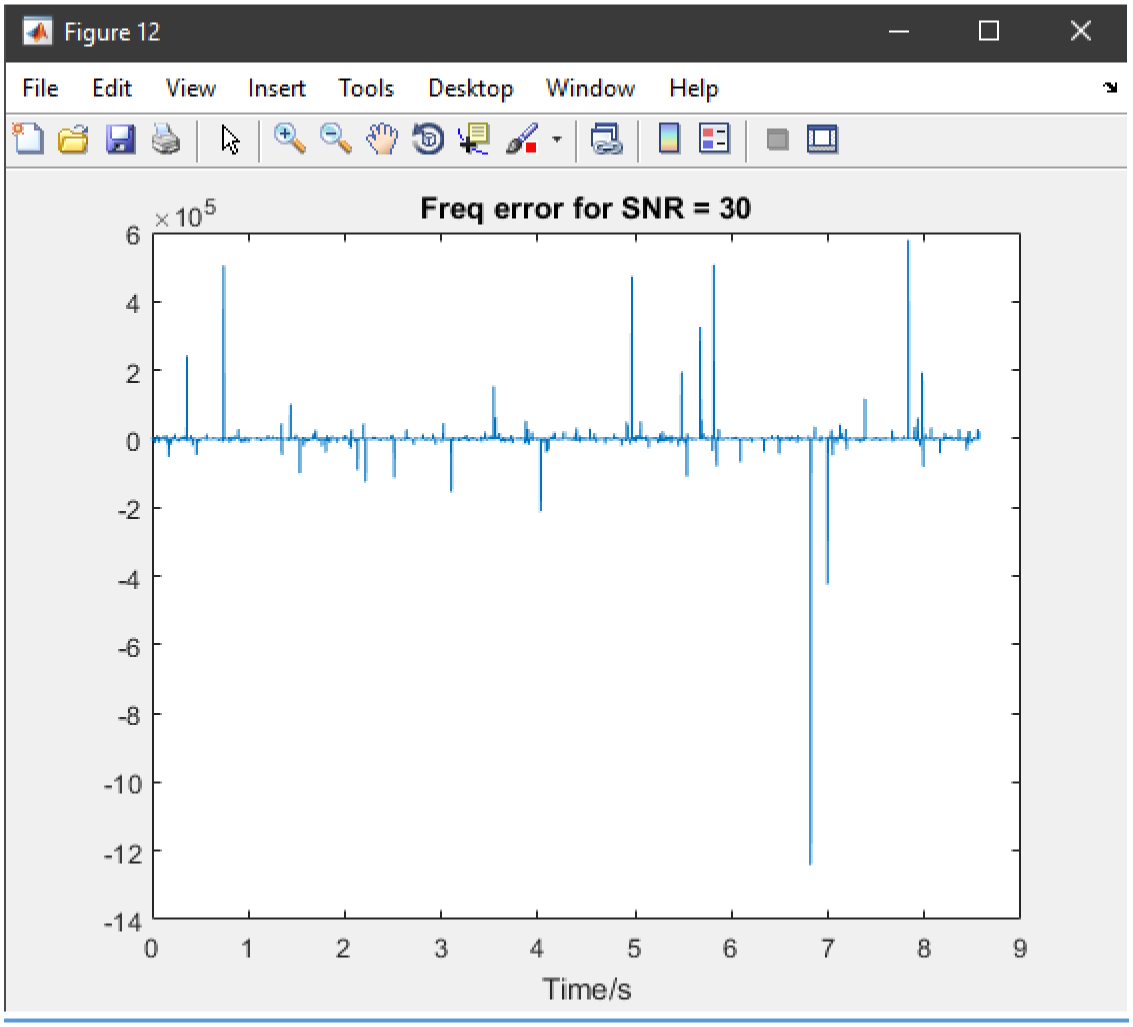


Figure 12
File
Edit
View
Insert
Tools
Desktop
Window
Help
Freq error for SNR = 30
$\times 10^5$
6
4
2
0
-2
-4
-6
-8
-10
-12
-14
0
1
2
3
4
5
6
7
8
9
Time/s

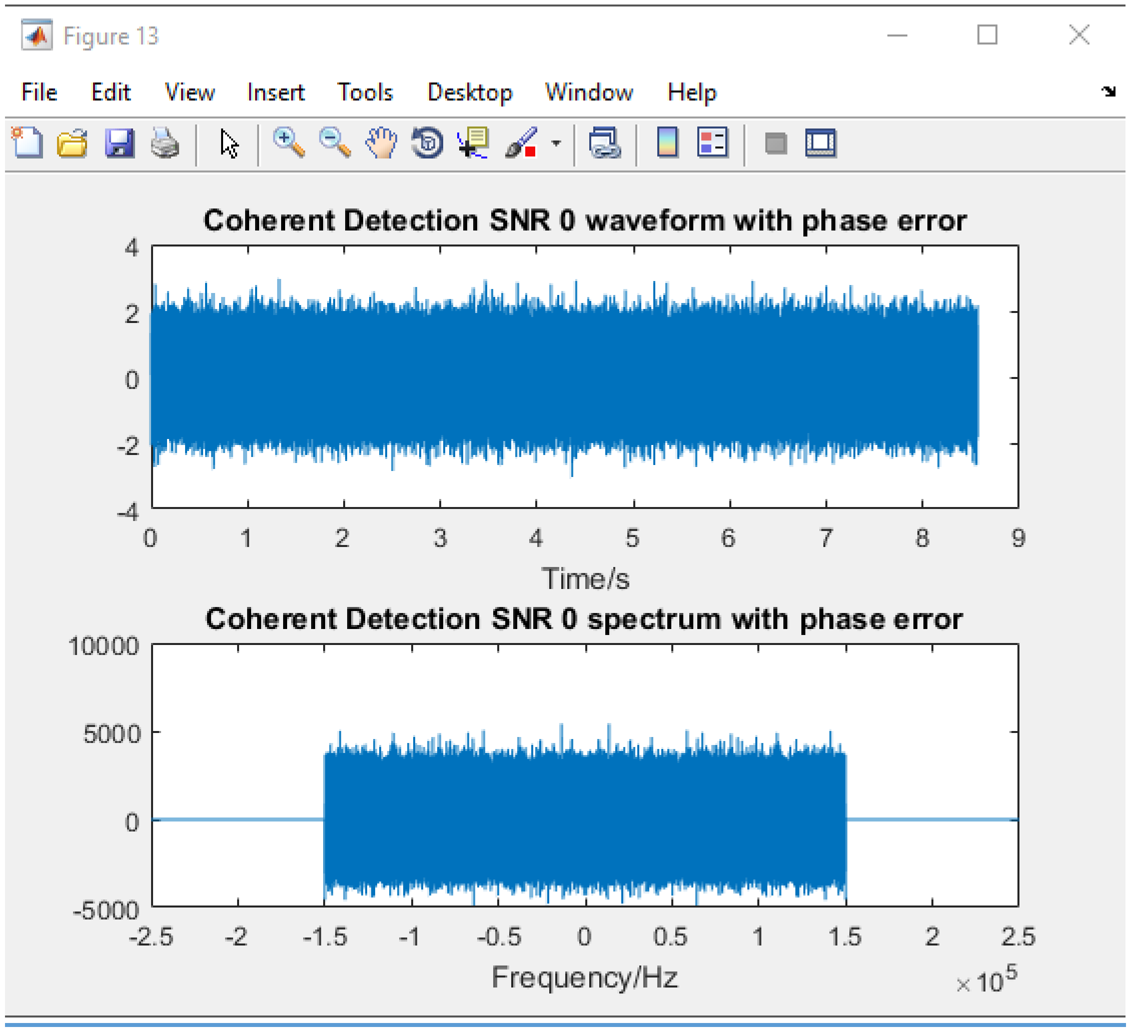


Figure 13
File Edit View Insert Tools Desktop Window Help
Coherent Detection SNR 0 waveform with phase error
4
2
0
-2
-4
0 1 2 3 4 5 6 7 8 9
Time/s
Coherent Detection SNR 0 spectrum with phase error
10000
5000
0
-5000
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5
Frequency/Hz
$\times 10^5$

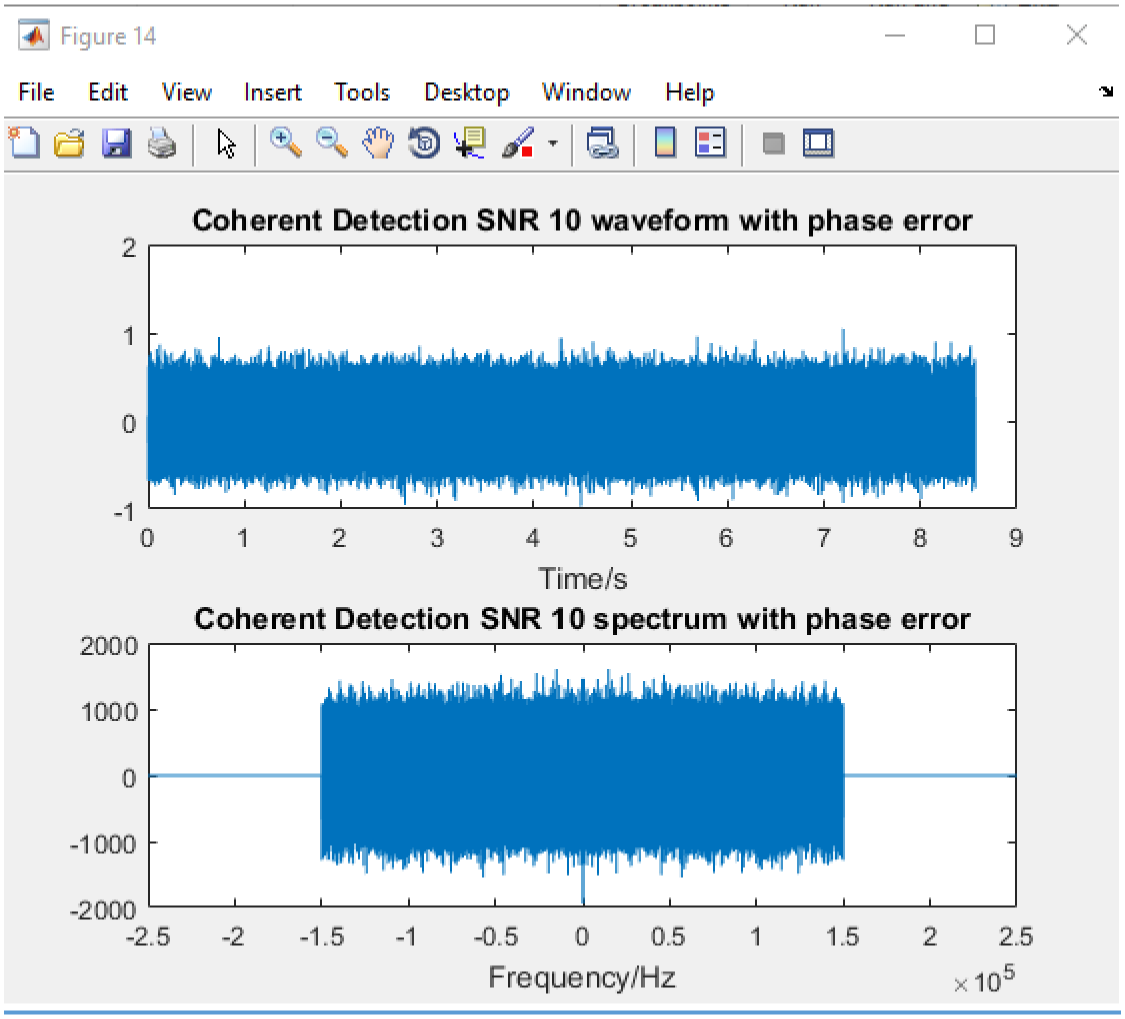
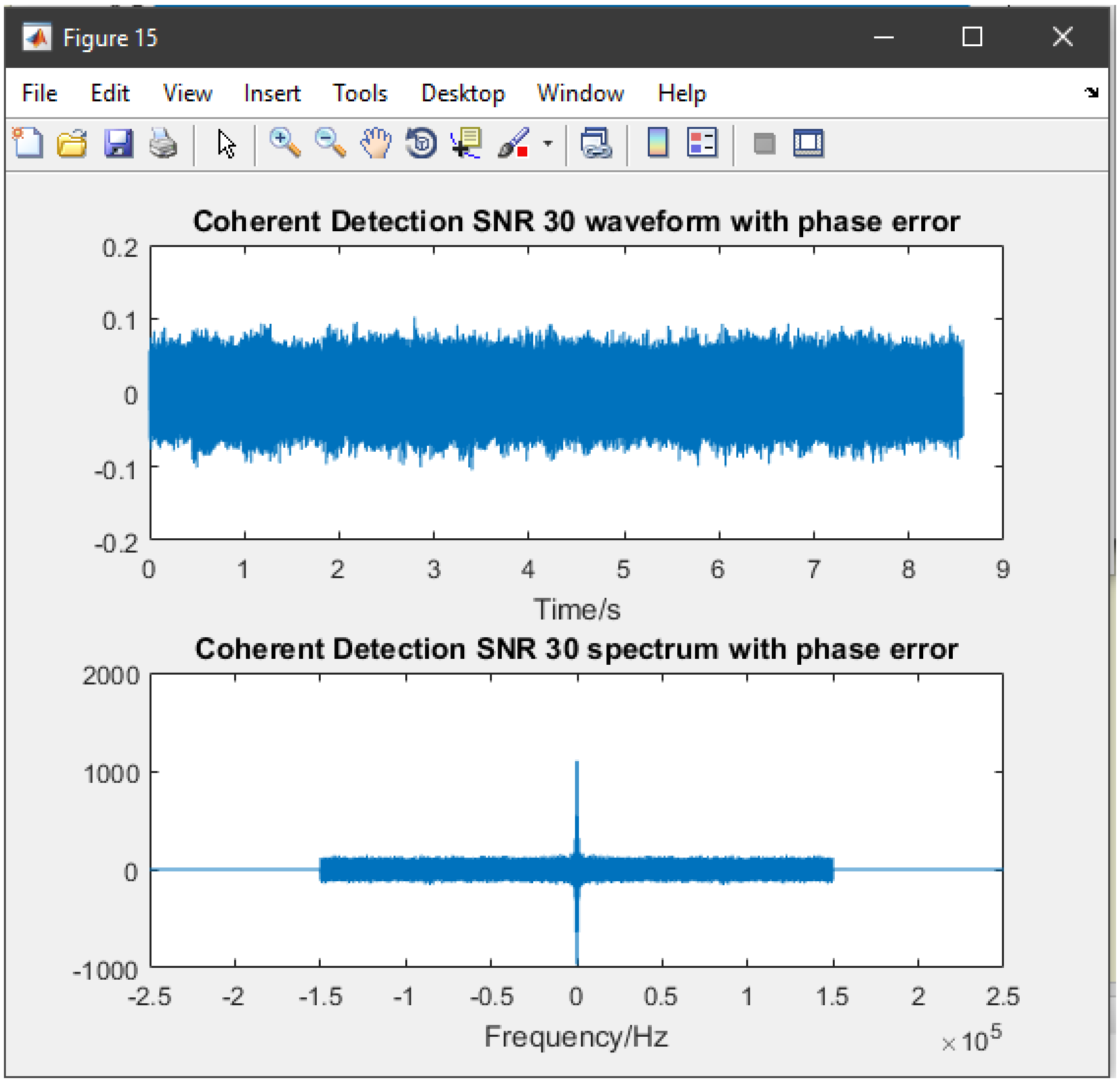


Figure 14
File Edit View Insert Tools Desktop Window Help
Coherent Detection SNR 10 waveform with phase error
Time/s
Coherent Detection SNR 10 spectrum with phase error
Frequency/Hz
$\times 10^5$



1.3 answer to the questions given:

7) Envelop detector can only work with DSB-TC and outputs noise in the DSC-SC.

9) Frequency error.

Part 2:

2.1 The code:

```
```

clear
clc
%% reading audio
% reading audio
[y,Fs] = audioread('eric.wav');
% fft application
L = length(y); % measure length of function for generation of frequency axis
Y = fftshift(fft(y)); % transform and shift the function
f = (-L/2:L/2-1)*(Fs/L); % generate frequency axis

% plotting the spectrum
plot(f,Y); % plot in frequency domain
title('Frequency Spectrum of eric.wav')
xlabel('Frequency/Hz')
figure;
%% filtering

% filtering
Yfiltered = Y;
Yfiltered(abs(f)>4000) = 0; % simulating ideal lpfilter set at 4000 Hz

% reversing fft and testing
yfiltered = ifft(ifftshift(Yfiltered));

% sounding filtered signal
sound(y,Fs);

pause(8);

%% modulation
% % resampling
yfilresampled = resample(yfiltered,125,12); % resampling to make sampling
% frequency = carrier frequency times 5

% % modulation
fc = 100000; % set carrier frequency
tcar = 0:1/(fc*5):8.567668; % generate carrier time tupple with overall
% function sampling frequency from 0 to length of audio tupple
tcar(end) = []; % remove final 'extra value'
carrier = cos(2*pi*fc*tcar)'; % generate carrier tupple
```
```

```
% % % DSB-SC

sc = yfilresampled.*carrier; % multiply carrier and filtered+resampled function tupple
Ld = length(sc); % calculate legnth of modulated wave
SC = fftshift(fft(sc));
fam = (-Ld/2:Ld/2-1)*((5*fc)/Ld);

plot(fam,SC); % plot DSB-SC spectrum
title('Frequency Spectrum of DSB-SC')
xlabel('Frequency/Hz')
figure

%% Generating LSB

% SC is the DSB-SC spectrum
LSB = SC;
LSB(abs(fam)>fc) = 0; % filter out the USB using ideal lpfilter

plot(fam,LSB);
title('Frequency Spectrum of LSB')
xlabel('Frequency/Hz')
figure

%% Coherent detection

s0 = (ifft(ifftshift(LSB))).*carrier;
S0 = fftshift(fft(s0));
L = length(s0);
f = (-L/2:L/2-1)*(500000/L);
S0(abs(f)>1.5*fc) = 0;
s0 = ifft(ifftshift(S0));

s0R = resample(s0,12,125);
sound(s0R,Fs)
pause(8);
subplot(2,1,1)
plot(tcar,s0)
title('Coherent Detection waveform')
xlabel('Time/s')
subplot(2,1,2)
plot(f,S0)
title('Coherent Detection spectrum')
xlabel('Frequency/Hz')
figure

%% Steps 5 and 6 using Butterworth filter
% step 5 (generating LSB)
[b,a] = butter(4,0.4); % generating Butterworth low pass filter of order 4,
% with Wn = fc/(fc/2) = 100,000/250,000 = 0.4
LSBf = filter(b,a,sc); % generate 'LSB' using Butterworth filter generated
% before

plot(fam,fftshift(fft(LSBf)));
title('Frequency Spectrum of LSB using Butterworth filter')
xlabel('Frequency/Hz')
figure

% step 6 (coherent detection of LSB signal)
```

```
sb0 = (ifft(ifftshift(LSBf))).*carrier;
Sb0 = fftshift(fft(sb0));
L = length(sb0);
f = (-L/2:L/2-1)*(500000/L);
Sb0(abs(f)>1.5*fc) = 0;
sb0 = ifft(ifftshift(Sb0));

s0R = resample(s0,12,125);
sound(s0R,Fs)
pause(8);
subplot(2,1,1)
plot(tcar,sb0)
title('Coherent Detection waveform (BW)')
xlabel('Time/s')
subplot(2,1,2)
plot(f,Sb0)
title('Coherent Detection spectrum (BW)')
xlabel('Frequency/Hz')
figure

%% Repeat demodulation with noise

lsbWN0 = awgn(ifft(ifftshift(LSB)),0); % adds white noise
% with SNR =  0 then  SNR = 10 then SNR = 30
lsbWN10 = awgn(ifft(ifftshift(LSB)),10);
lsbWN30 = awgn(ifft(ifftshift(LSB)),30);

% with SNR = 0

s0 = lsbWN0.*carrier;
S0 = fftshift(fft(s0));
S0(abs(f)>1.5*fc) = 0;
s0 = ifft(ifftshift(S0));

s0R = resample(s0,12,125);
sound(s0R,Fs)
pause(8);
subplot(2,1,1)
plot(tcar,s0)
title('Coherent Detection waveform SNR = 0')
xlabel('Time/s')
subplot(2,1,2)
plot(f,S0)
title('Coherent Detection spectrum SNR = 0')
xlabel('Frequency/Hz')
figure


% with SNR = 10

s10 = lsbWN10.*carrier;
S10 = fftshift(fft(s10));
S10(abs(f)>1.5*fc) = 0;
s10 = ifft(ifftshift(S10));

s10R = resample(s10,12,125);
sound(s10R,Fs)
pause(8);
subplot(2,1,1)
```

```
plot(tcar,s10)
title('Coherent Detection waveform SNR = 10')
xlabel('Time/s')
subplot(2,1,2)
plot(f,S10)
title('Coherent Detection spectrum SNR = 10')
xlabel('Frequency/Hz')
figure

% with SNR = 30

s30 = lsbWN30.*carrier;
S30 = fftshift(fft(s30));
S30(abs(f)>1.5*fc) = 0;
s30 = ifft(ifftshift(S30));

s30R = resample(s30,12,125);
sound(s30R,Fs)
pause(8);
subplot(2,1,1)
plot(tcar,s30)
title('Coherent Detection waveform SNR = 30')
xlabel('Time/s')
subplot(2,1,2)
plot(f,S30)
title('Coherent Detection spectrum SNR = 30')
xlabel('Frequency/Hz')
figure

%% SSB-TC
M = max(yfilresampled); % find amplitude of sound function
tc = (2*M + yfilresampled).*carrier; % generate 'SSB-TC' wave
TC = fftshift(fft(tc));

LSBtc = TC;
LSBtc(abs(fam)>fc) = 0;
lsbtcE = abs(hilbert(ifft(ifftshift(LSBtc))));

plot(tcar,lsbtcE);
title('Waveform of LSB-TC envelope')
xlabel('Time/s')

stcR = resample(lsbtcE,12,125);
sound(stcR,Fs)


```
```
```

2.2 the plots:

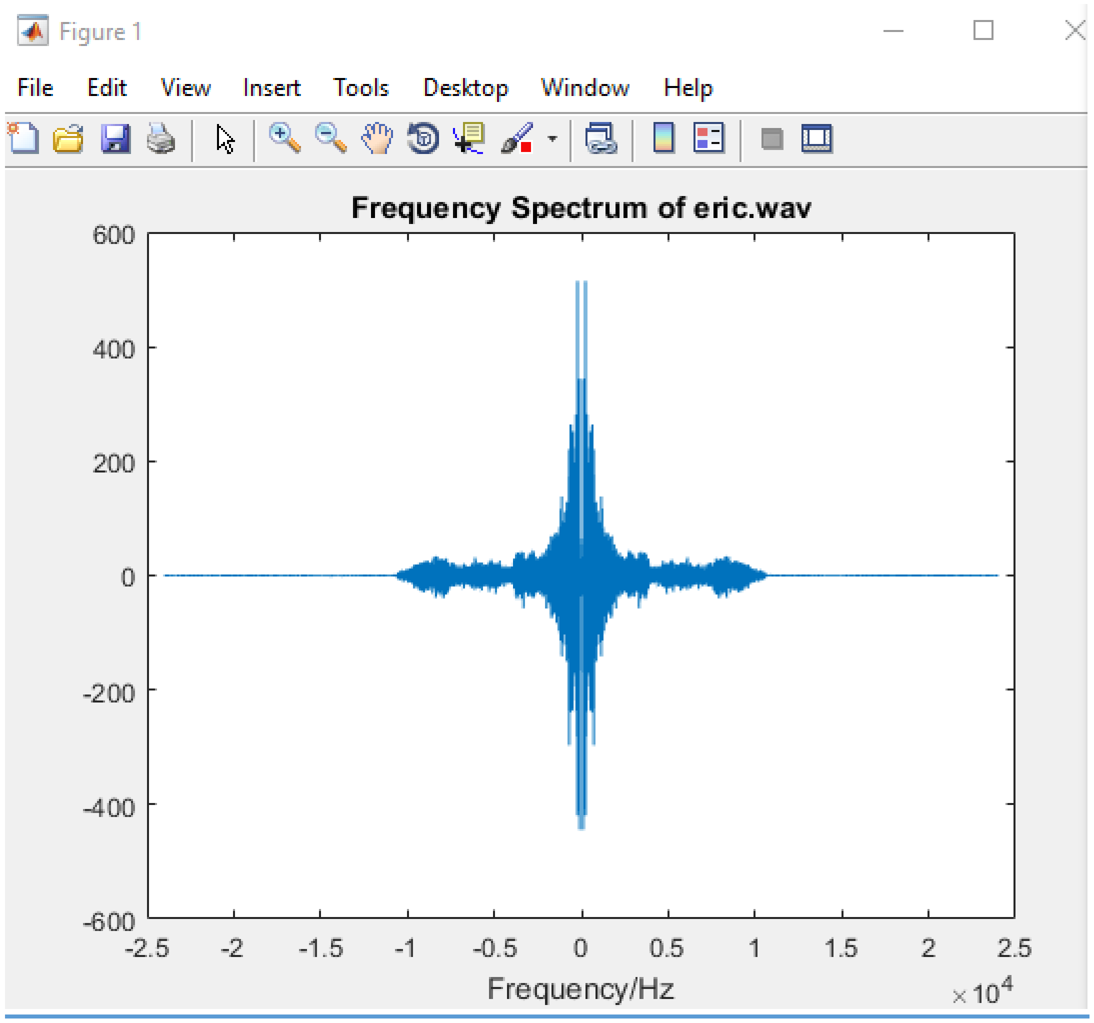


Figure 1
File Edit View Insert Tools Desktop Window Help
Frequency Spectrum of eric.wav
600
400
200
0
-200
-400
-600
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Frequency/Hz
× 10^4

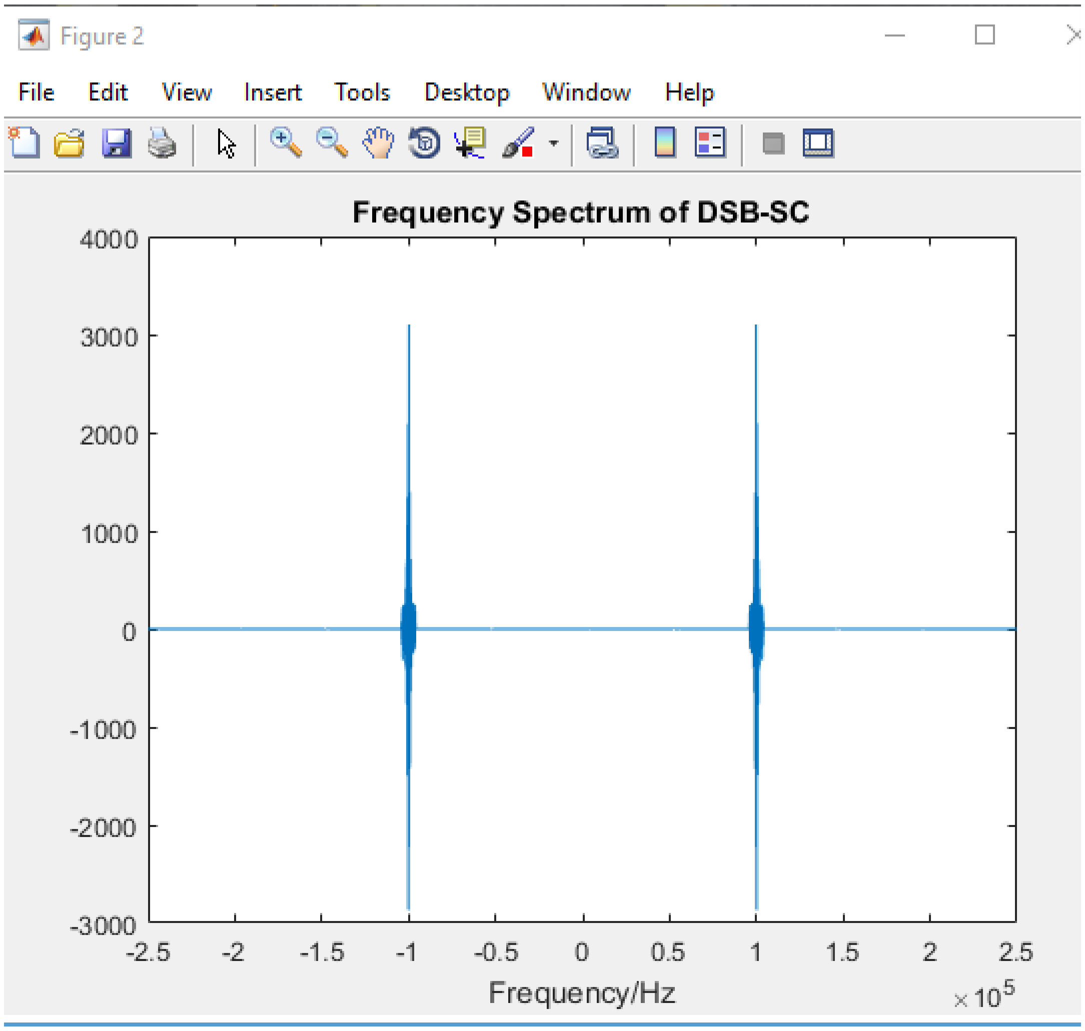


Figure 2
File Edit View Insert Tools Desktop Window Help
Frequency Spectrum of DSB-SC
4000
3000
2000
1000
0
-1000
-2000
-3000
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5
Frequency/Hz
$\times 10^5$

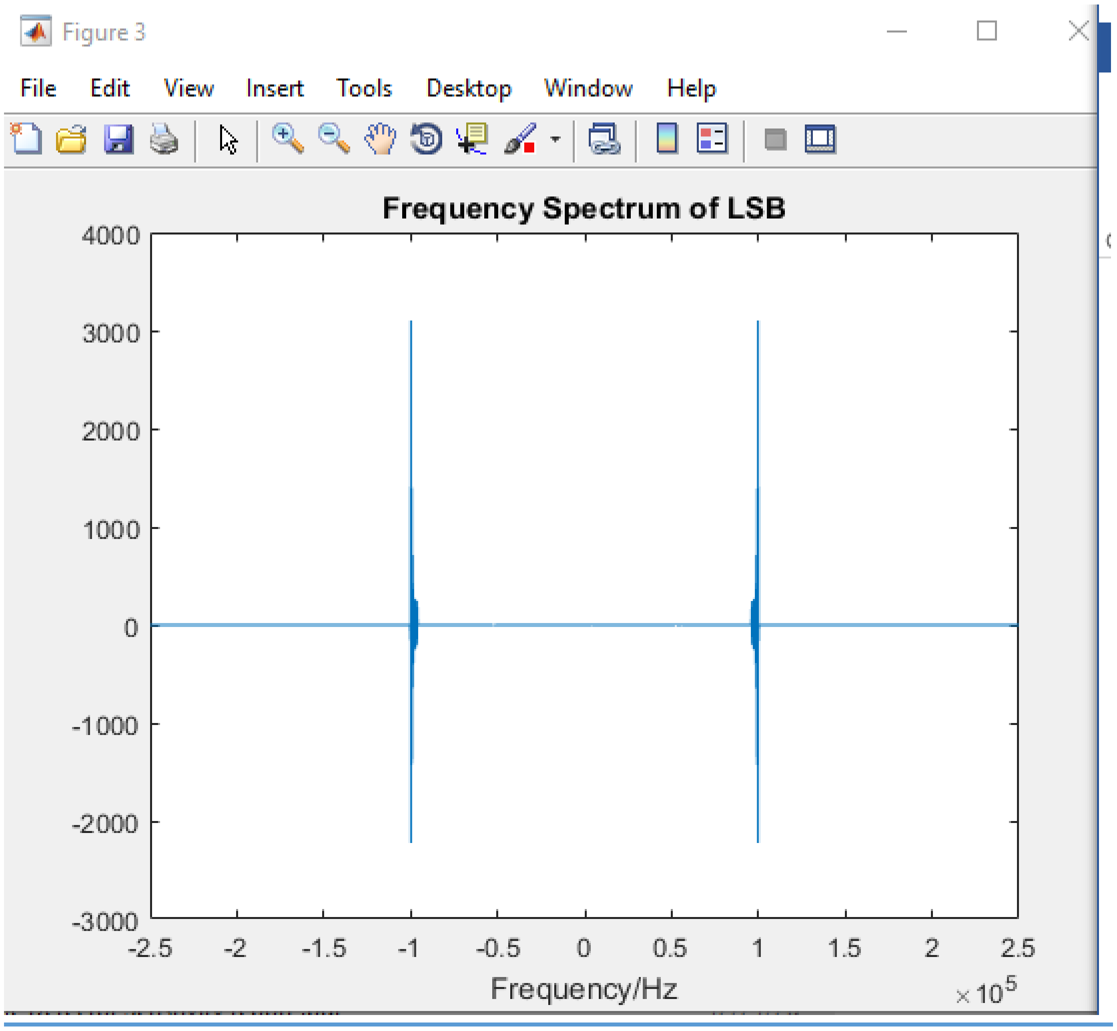


Figure 3
File Edit View Insert Tools Desktop Window Help
Frequency Spectrum of LSB
4000
3000
2000
1000
0
-1000
-2000
-3000
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Frequency/Hz
$\times 10^5$

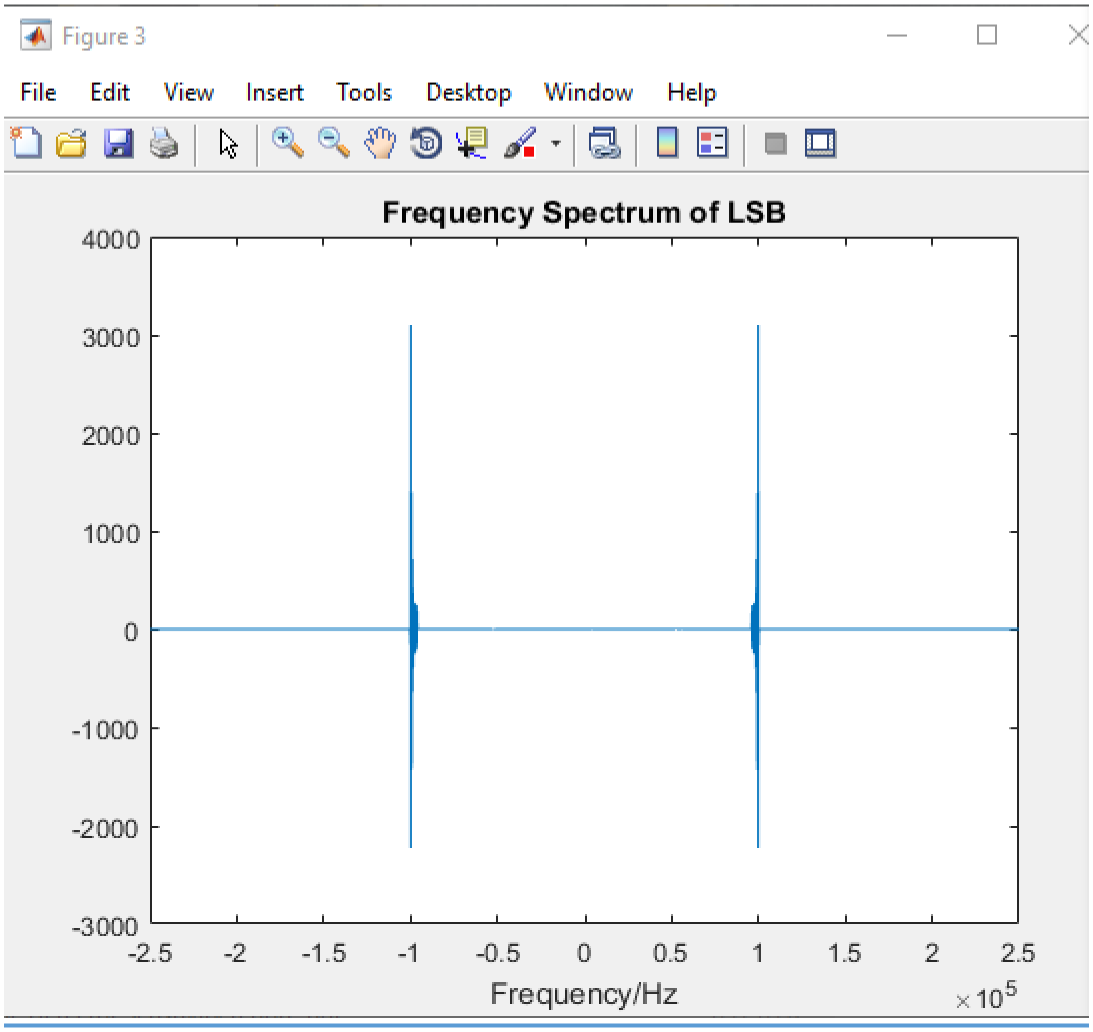


Figure 3
File Edit View Insert Tools Desktop Window Help
Frequency Spectrum of LSB
4000
3000
2000
1000
0
-1000
-2000
-3000
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Frequency/Hz
$\times 10^5$

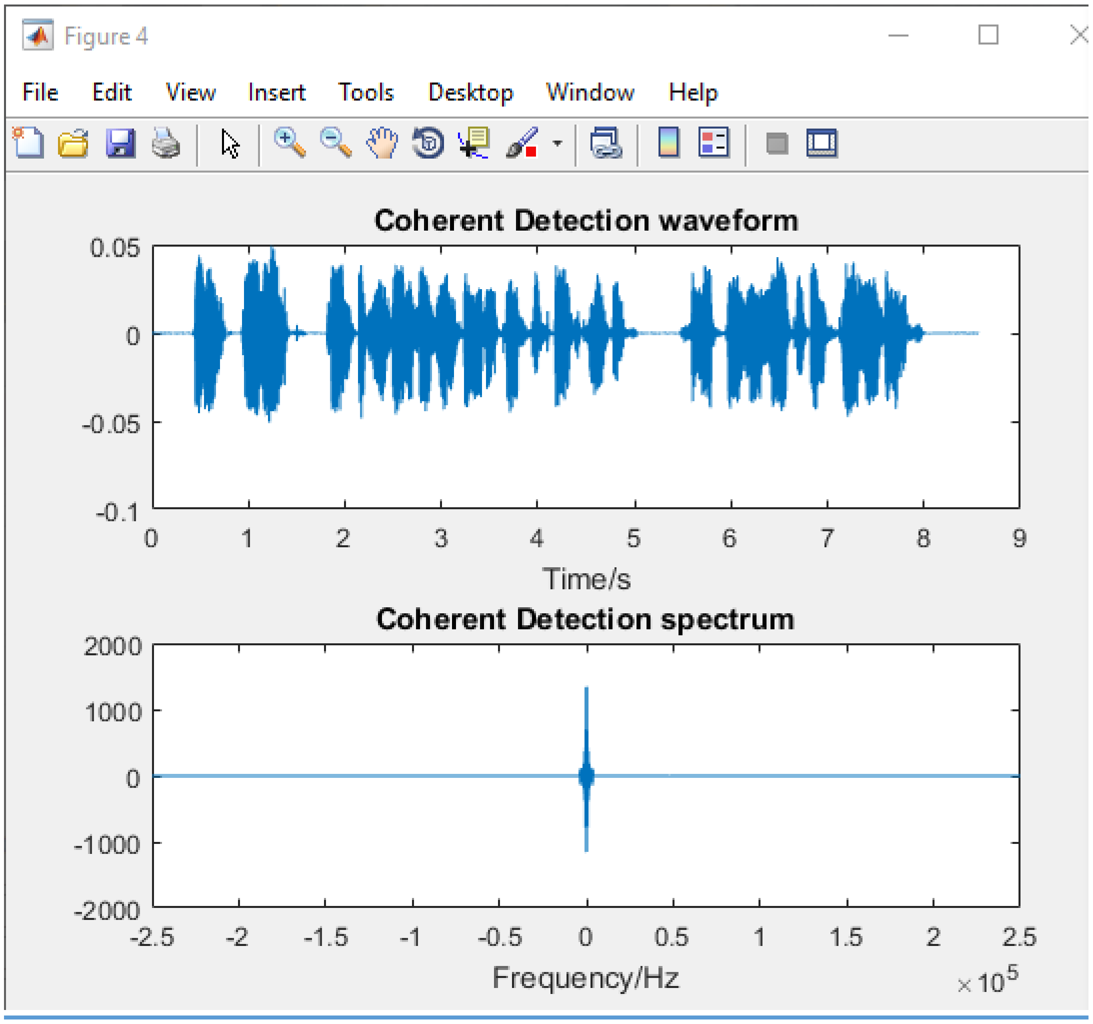


Figure 4
File Edit View Insert Tools Desktop Window Help
Coherent Detection waveform
0.05
0
-0.05
-0.1
0 1 2 3 4 5 6 7 8 9
Time/s
Coherent Detection spectrum
2000
1000
0
-1000
-2000
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5
Frequency/Hz
$\times 10^5$

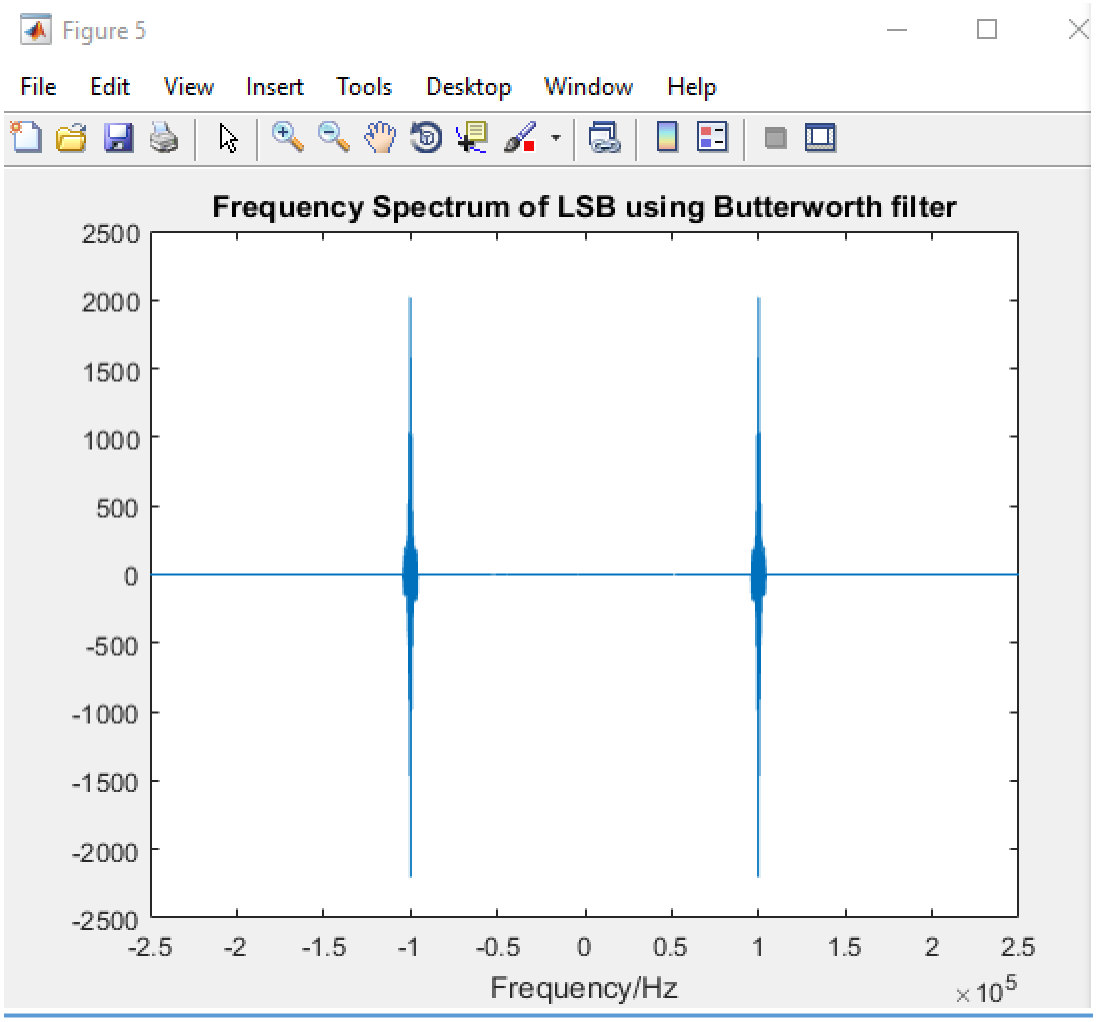


Figure 5
File Edit View Insert Tools Desktop Window Help
Frequency Spectrum of LSB using Butterworth filter
2500
2000
1500
1000
500
0
-500
-1000
-1500
-2000
-2500
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Frequency/Hz
×10^5

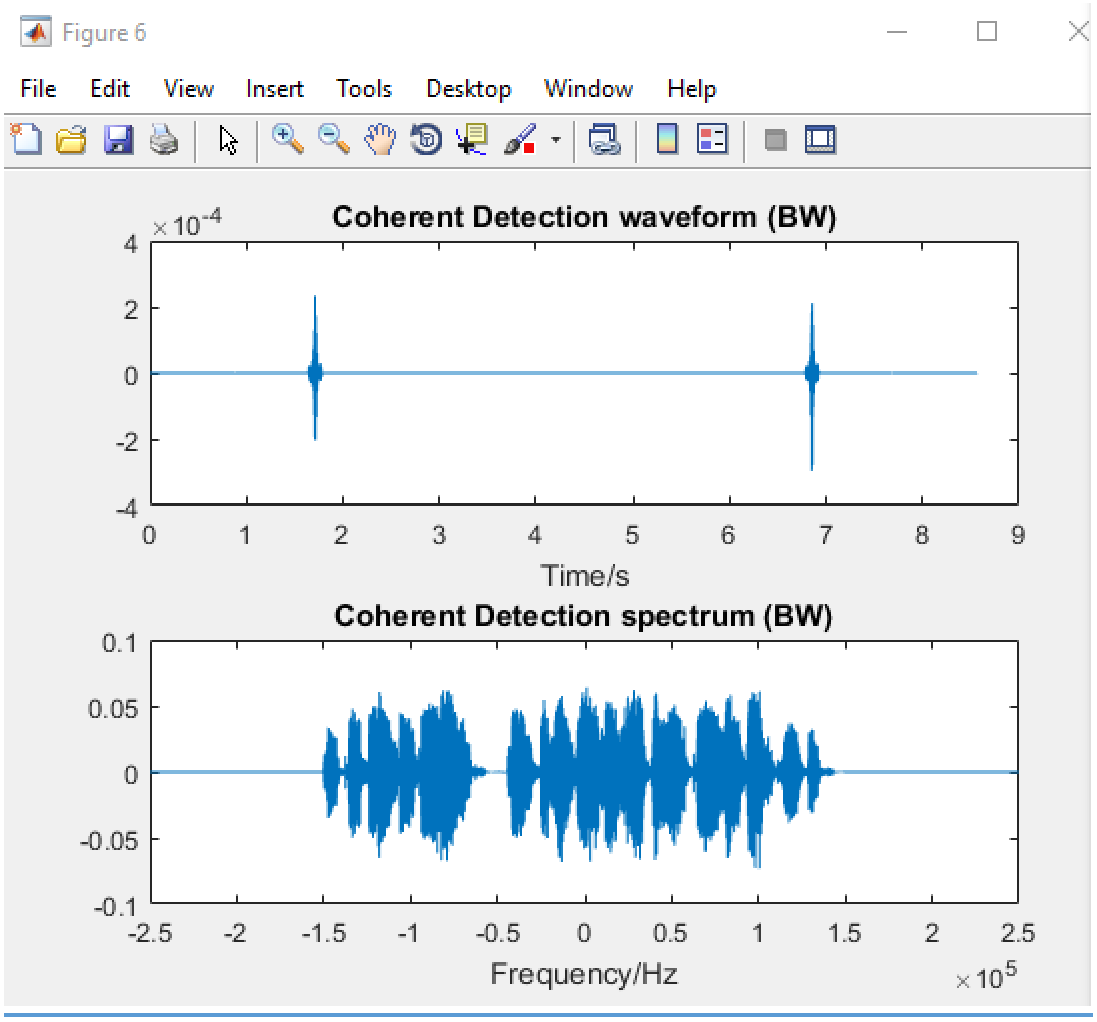


Figure 6
File Edit View Insert Tools Desktop Window Help
Coherent Detection waveform (BW)
×10^-4
Time/s
Coherent Detection spectrum (BW)
Frequency/Hz
×10^5

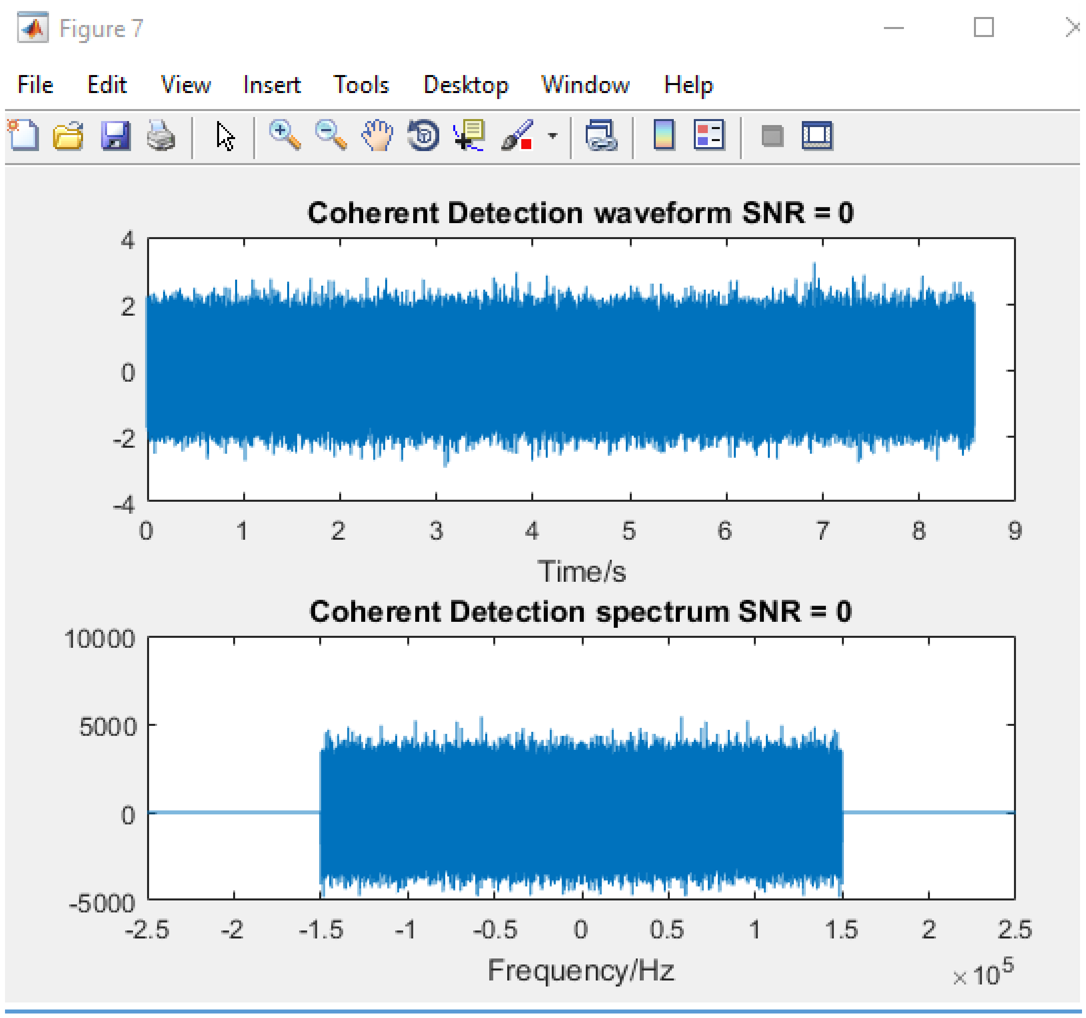


Figure 7
File Edit View Insert Tools Desktop Window Help
Coherent Detection waveform SNR = 0
4
2
0
-2
-4
0 1 2 3 4 5 6 7 8 9
Time/s
Coherent Detection spectrum SNR = 0
10000
5000
0
-5000
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5
Frequency/Hz
×10^5

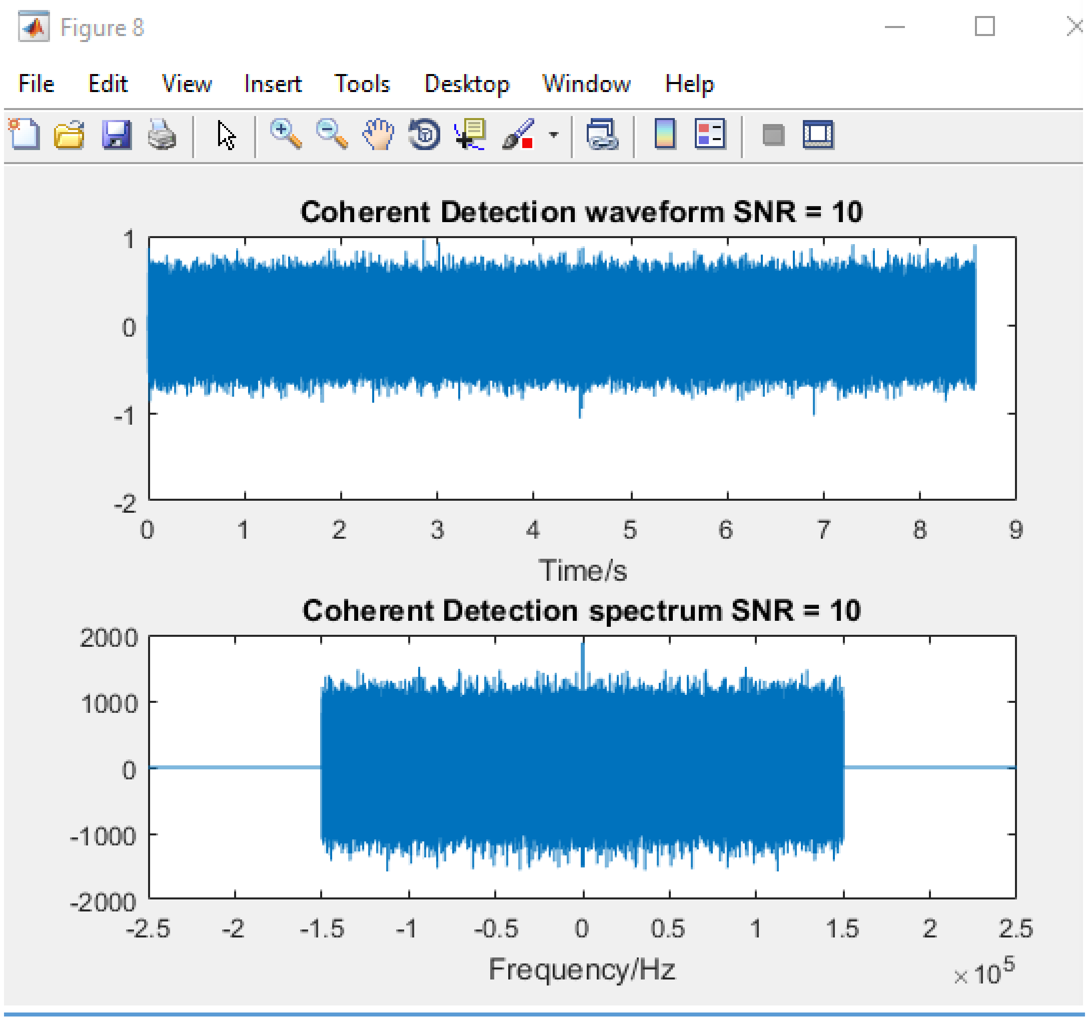


Figure 8
File Edit View Insert Tools Desktop Window Help
Coherent Detection waveform SNR = 10
Time/s
Coherent Detection spectrum SNR = 10
Frequency/Hz
$\times 10^5$

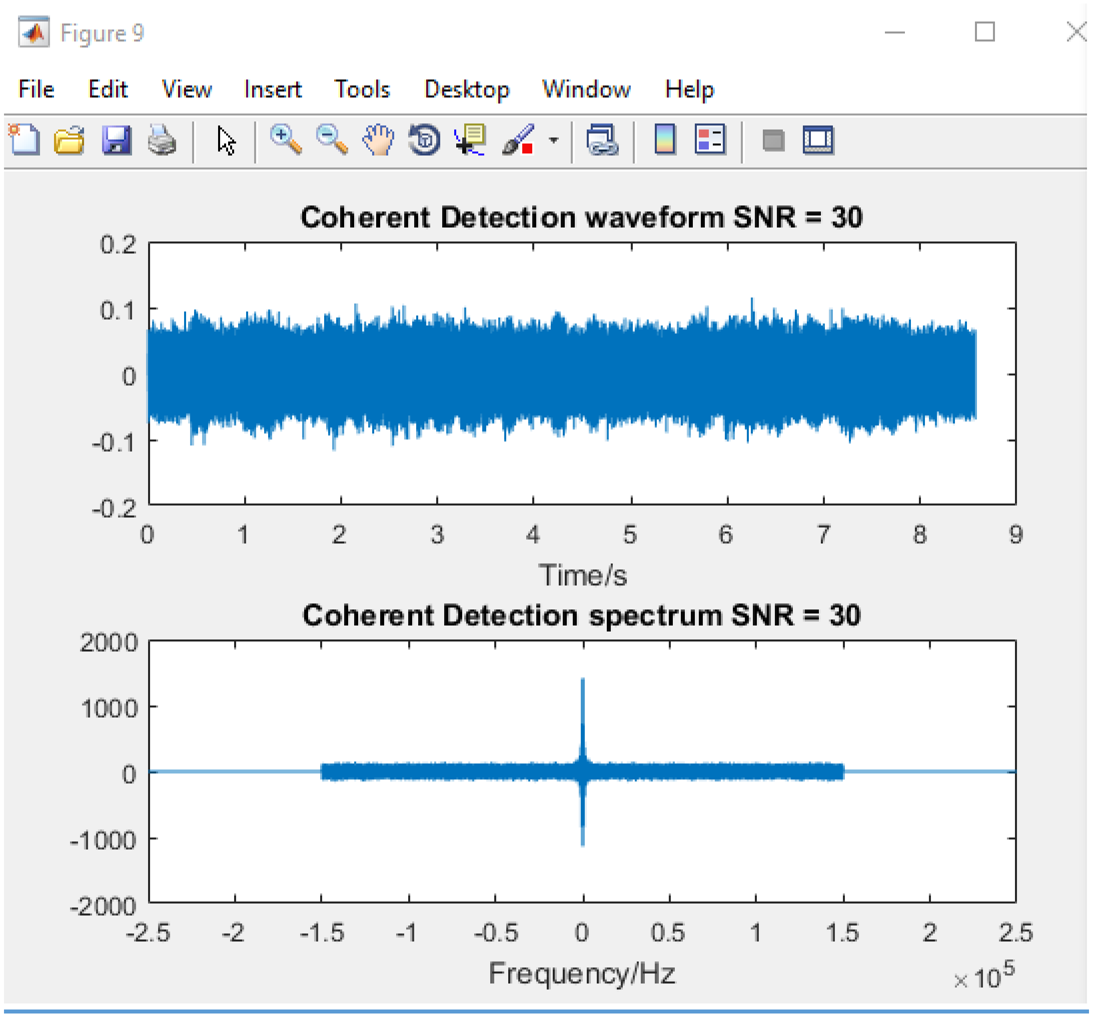


Figure 9
File Edit View Insert Tools Desktop Window Help
Coherent Detection waveform SNR = 30
0.2
0.1
0
-0.1
-0.2
0 1 2 3 4 5 6 7 8 9
Time/s
Coherent Detection spectrum SNR = 30
2000
1000
0
-1000
-2000
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5
Frequency/Hz
× 10^5

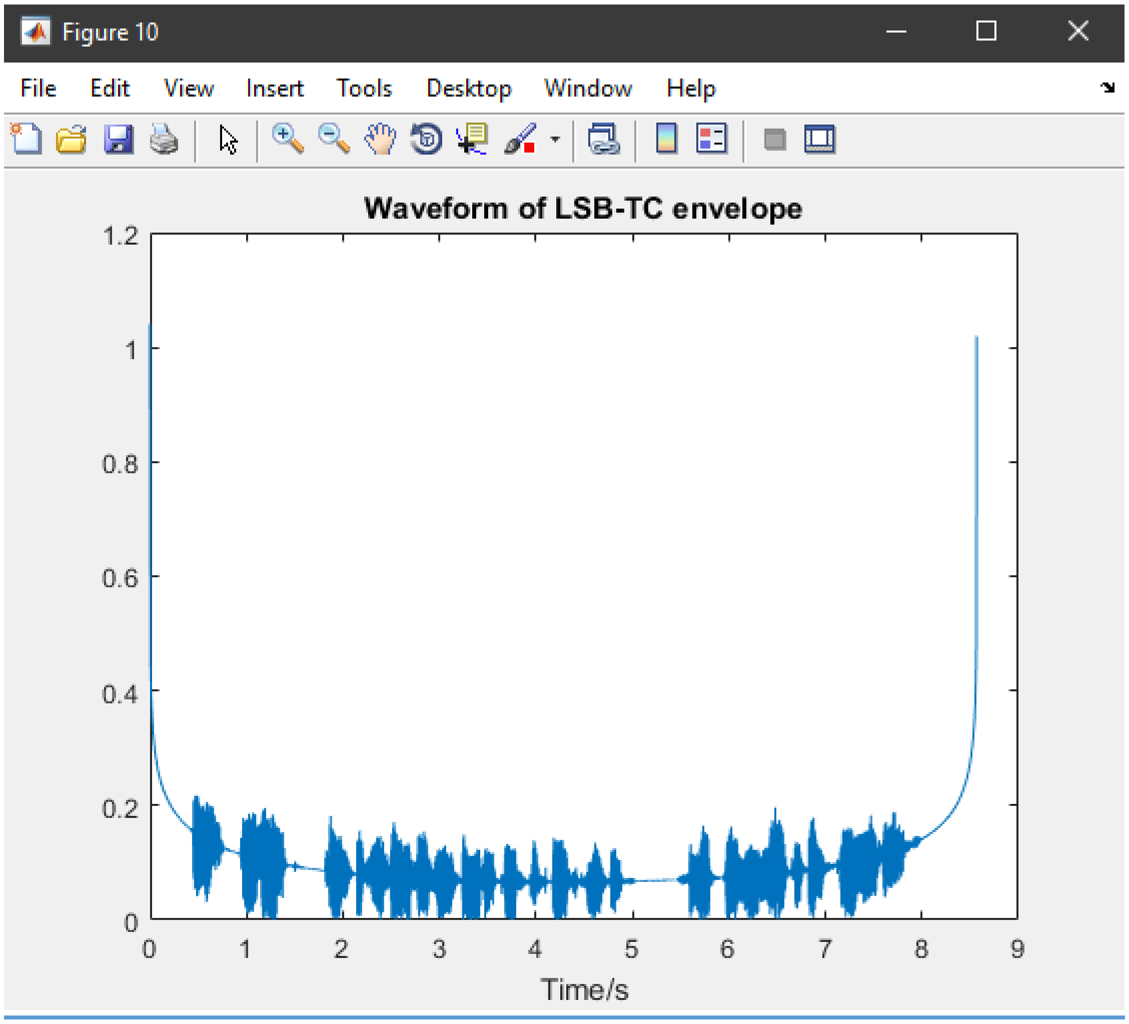


Figure 10
File
Edit
View
Insert
Tools
Desktop
Window
Help
Waveform of LSB-TC envelope
1.2
1
0.8
0.6
0.4
0.2
0
0
1
2
3
4
5
6
7
8
9
Time/s

Part 3:

3.1 The code:

```
clear
clc
%% reading audio
% reading audio
[y,Fs] = audioread('eric.wav');
% fft application
L = length(y); % measure length of function for generation of frequency axis
Y = fftshift(fft(y)); % transform and shift the function
f = (-L/2:L/2-1)*(Fs/L); % generate frequency axis
% plotting the spectrum
plot(f,Y); % plot in frequency domain
title('Frequency Spectrum of eric.wav')
xlabel('Frequency/Hz')
figure;
%% filtering
% filtering
Yfiltered = Y;
Yfiltered(abs(f)>4000) = 0; % simulating ideal lpfilter set at 4000 Hz
% reversing fft and testing
yfiltered = ifft(ifftshift(Yfiltered));
% sounding filtered signal
% % sound(y,Fs);
% % pause(8);
%% modulation
% by Narrow-band FM
% % resampling
yfilresampled = resample(yfiltered,125,12); % resampling to make sampling
% frequency = carrier frequency times 5


% % modulation
fc = 100000; % set carrier frequency
tcar = 0:1/(fc*5):8.567668; % generate carrier time tupple with overall
% function sampling frequency from 0 to length of audio tupple
tcar(end) = []; % remove final 'extra value'
Ld = length(yfilresampled); % calculate length of resampled wave
ffm = (-Ld/2:Ld/2-1)*((5*fc)/Ld);

kf = 0.1; % set frequency variation constant to low enough value
x = (cumsum(yfilresampled))'; % calculate cumulative summation of audio
% function from -inf to 't'
s = cos(2*pi*fc*tcar + 2*pi*kf*x); % generate NBFM signal
S = fftshift(fft(s)); % fourier transform and shift

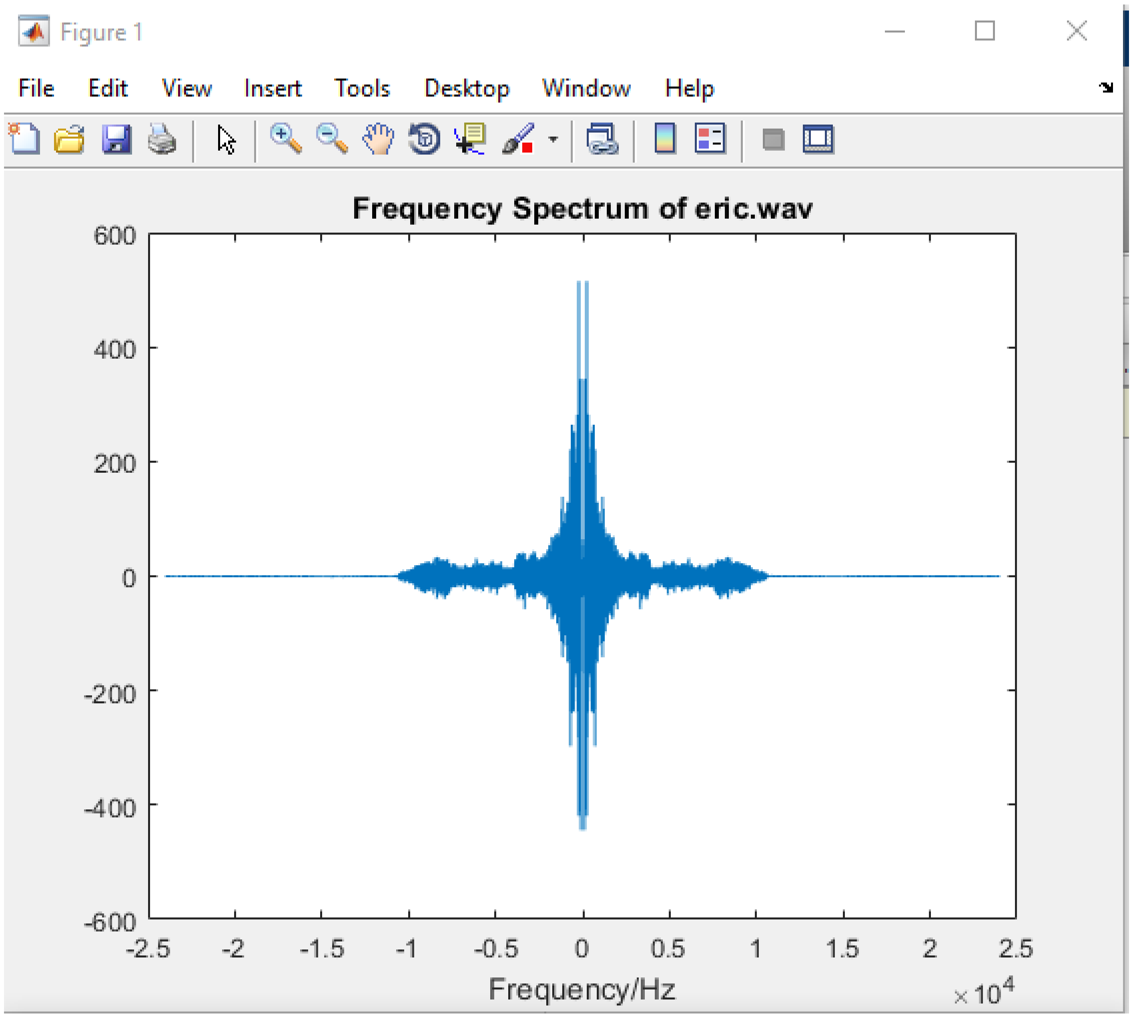

plot(ffm,S); % plot FM modulated signal in frequency domain
title('Frequency Spectrum of FM modulated signal')
xlabel('Frequency/Hz')
figure
```

```
%% demodulation using differentiator and envelope detector

sdiff = diff(s);
senv = abs(hilbert(sdiff)); % envelope detector function
tcar(end) = [];
plot(tcar,senv)
title('Waveform of Envelope')
xlabel('Time/s')


```
```

3.2 The plots:


```

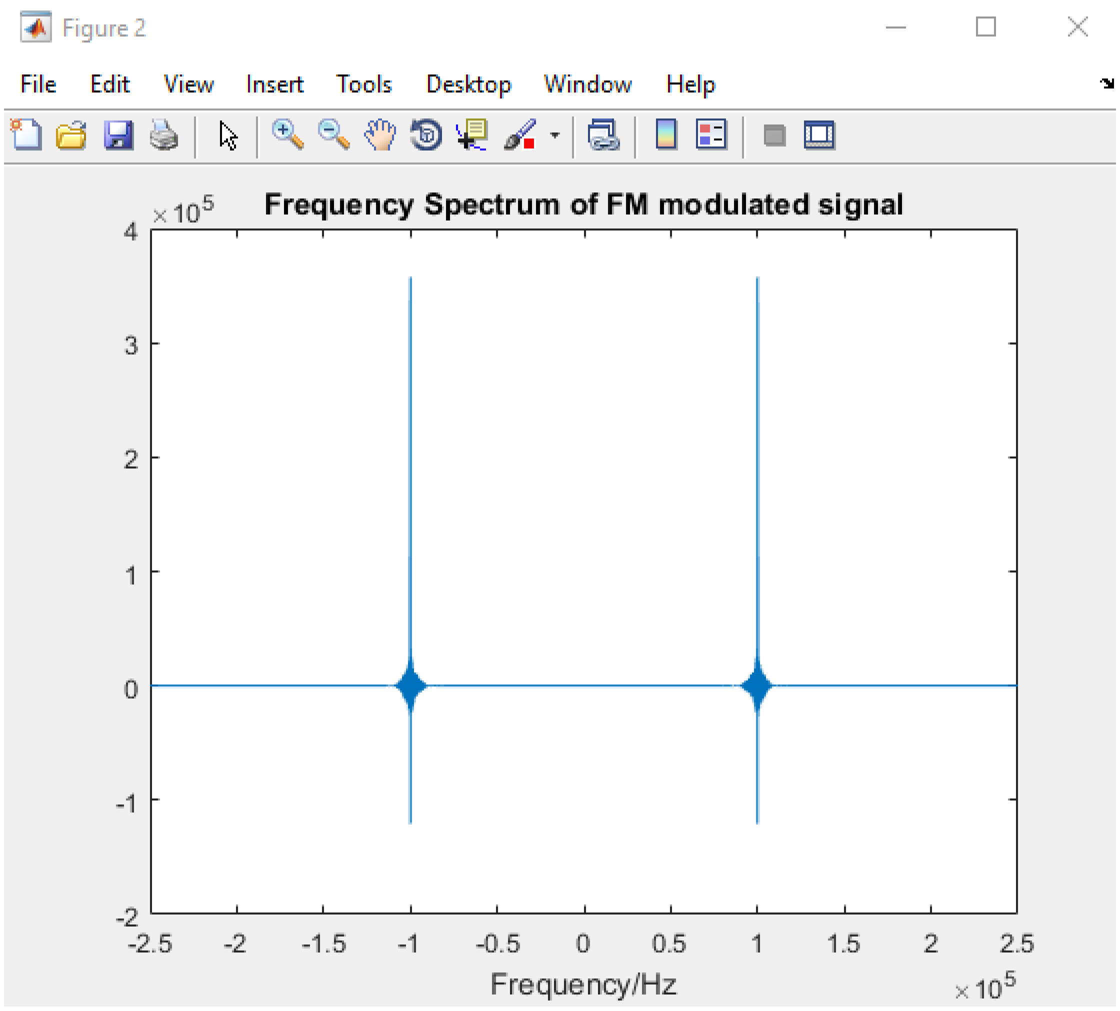
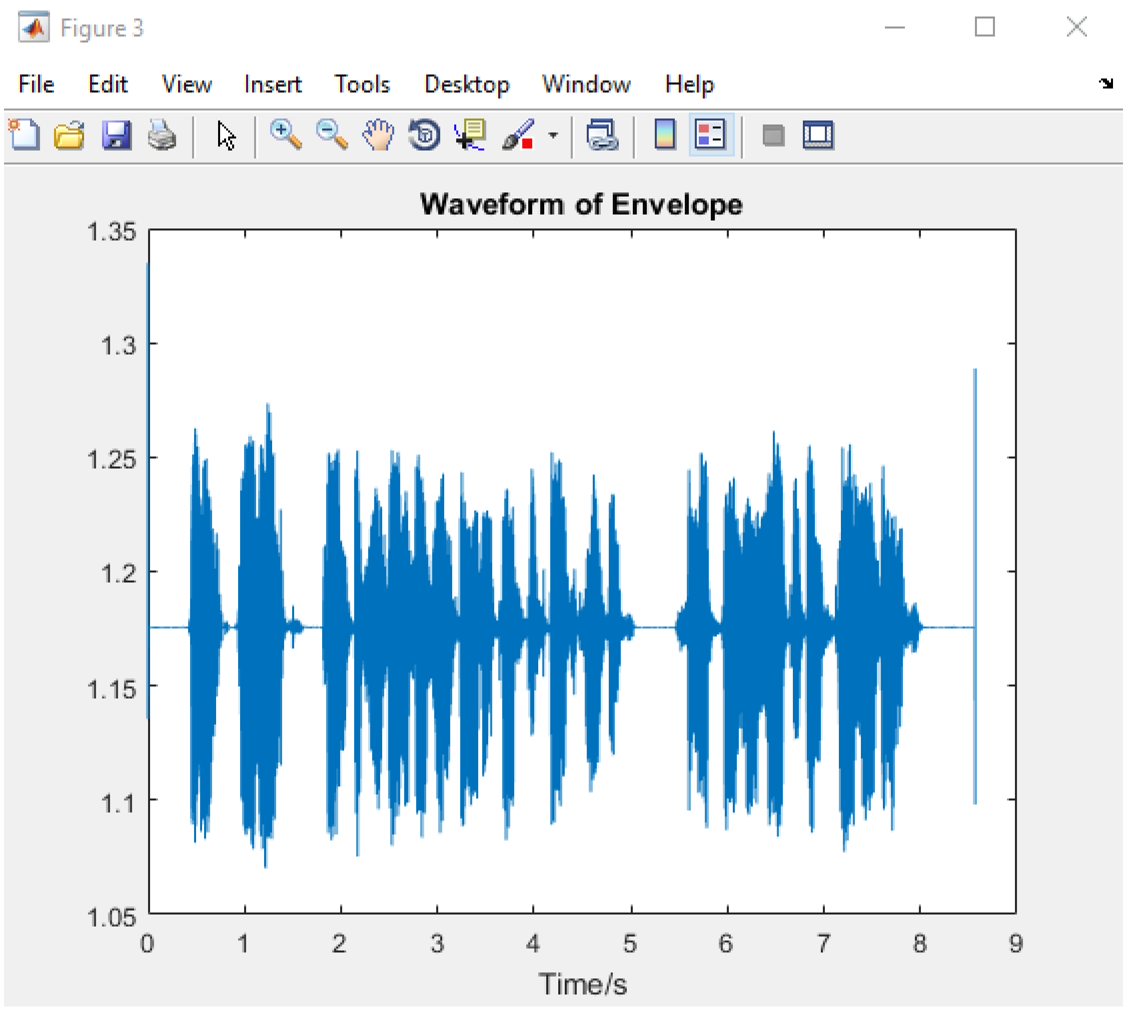


Figure 2
File Edit View Insert Tools Desktop Window Help
Frequency Spectrum of FM modulated signal
×10^5
4
3
2
1
0
-1
-2
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5
Frequency/Hz
×10^5



## 3.3 Answer to the questions presented:

2) This plot looks similar to the DSB-TC plot.

3)-The most important condition is that the input signal to noise ratio exceeds a threshold value. Below this threshold, the signal to noise output deteriorates rapidly, and for low inputs the signal to noise is worse than SSB.

-The value of beta must be lower than 0.**5 (to satisfy the condition of the Narrow Band Signal).**

## 4.1 The conclusion:

-DSB-TC is the easiest modulation method as it can be demodulated using envelop detector, however it is inefficient as we send both LSB and USB resulting in double baseband bandwidth. It is also more susceptible to noise. Also most of the power is wasted as carrier power which is non-useful power to us.

-SSB is a better alternative from the point of efficiency as we only send either USB or LSB resulting in bandwidth equal to that of the baseband signal. It is more resistant to noise than DSB. Since only one side is sent, the sent signal is more susceptible to corruption and loss of data.

-For envelop detector to work there must be a carrier with the sent signal.

-Coherent detector can work with any type of amplitude modulation, but it is expensive.

-Signals with 30 db SNR is the best quality, however 0 db and 10 db SNR are too noisy to understand.

-signals with 30 db SNR and phase error was distorted, but understandable still.

-in case of phase error as long as it’s not Π/2 the signal will suffer from attenuation. And in case of Π/2 the signal is completely lost.

-in case of frequency error, we must introduce a pilot carrier to help detect the signal in ED (Also, any other detectors are acceptable such as: Synchronous detector (Coherent), Envelope detector (Non-Synchronous), and Superheterodyne receivers).

-Phase modulation is more resistant to noise as it does not depend on the amplitude of the modulated signal for its detection rather we send the signal as changes in frequency.