Faculty of Engineering and Architectural Science

Department of Computer and Electrical Engineering

Course Number	**ELE532**
Course Title	**Signal and Systems l**
Semester/Year	**Fall 2025**
Instructor	**Dr.Beheshti**

Report Title	The Fourier Transform: Properties and Applications

Section No.	12
Report Submission Date	

Name	Student ID	Signature*
Hamza Malik	501112545	H.M
Kattoji Aneesh	501233584	A.K

**By signing above, you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a "0" on the work, an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at: http://www.ryerson.ca/senate/policies/pol60.pdf. .*

Problem A.1

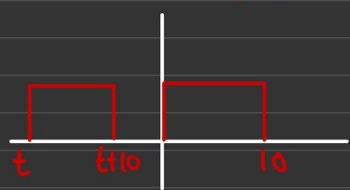
A.1

$x(t) = u(t) - u(t - t_0)$

$\hookrightarrow u(t) - u(t - 10)$

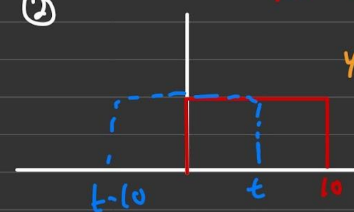
$x(t)_1 * x(t)_2 = z(t)$

① For $t < 0$



$y(t) = 0$

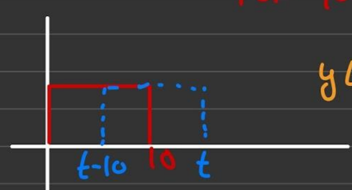
② For $0 < t < 10$



$y(t) = \int_0^t (1)(1)\, d\tau$

$y(t) = t$

③ For $10 < t < 20$



$y(t) = \int_{t-10}^{10} (1)\,(1)\, d\tau$

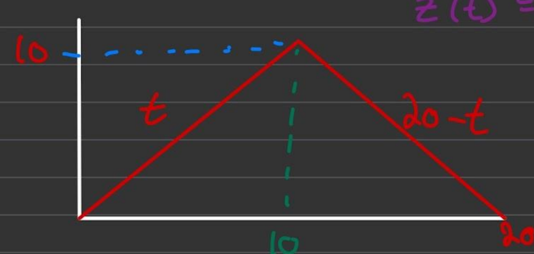
$= 10 - (t - 10)$

$= 20 - t$

④ For $t > 20$

$y(t) = 0$

$z(t) = x(t) * x(t)$



voĩe

$x(t) = x_1(t) = x_2(t) = u(t) - u(t-10)$

$X(j\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt = \int_{0}^{10} [u(t) - u(t-10)] e^{-j\omega t} dt$

$= \int_0^{10} (1) e^{-j\omega t} = -\frac{e^{-j\omega t}}{j\omega}\Big|_0^{10} = -\frac{e^{-j\omega(10)}}{j\omega} + \frac{1}{j\omega}$

$X_1(j\omega) = X_2(j\omega) = X(j\omega)$

$X_3(j\omega) = X_1(j\omega) \times X_2(j\omega) = \left(\frac{-e^{-j\omega 10}}{j\omega} + \frac{1}{j\omega}\right)^2$

$= -\frac{e^{-j\omega(20)}}{\omega^2} + \frac{2e^{-j\omega 10}}{\omega^2} - \frac{1}{\omega^2}$

$= \frac{-e^{-j\omega(20)} + 2e^{-j\omega(10)} - 1}{\omega^2} \rightarrow$

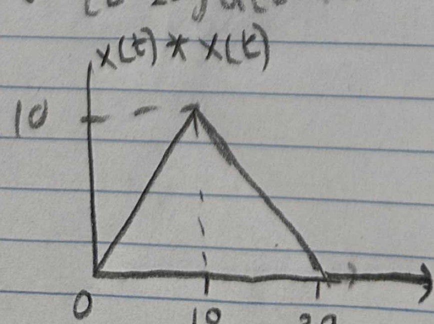
$x_3(t) = x_1(t) * x_2(t) = F^{-1}[X_3(j\omega)]$

$x_3(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X_3(j\omega) e^{j\omega t} d\omega$

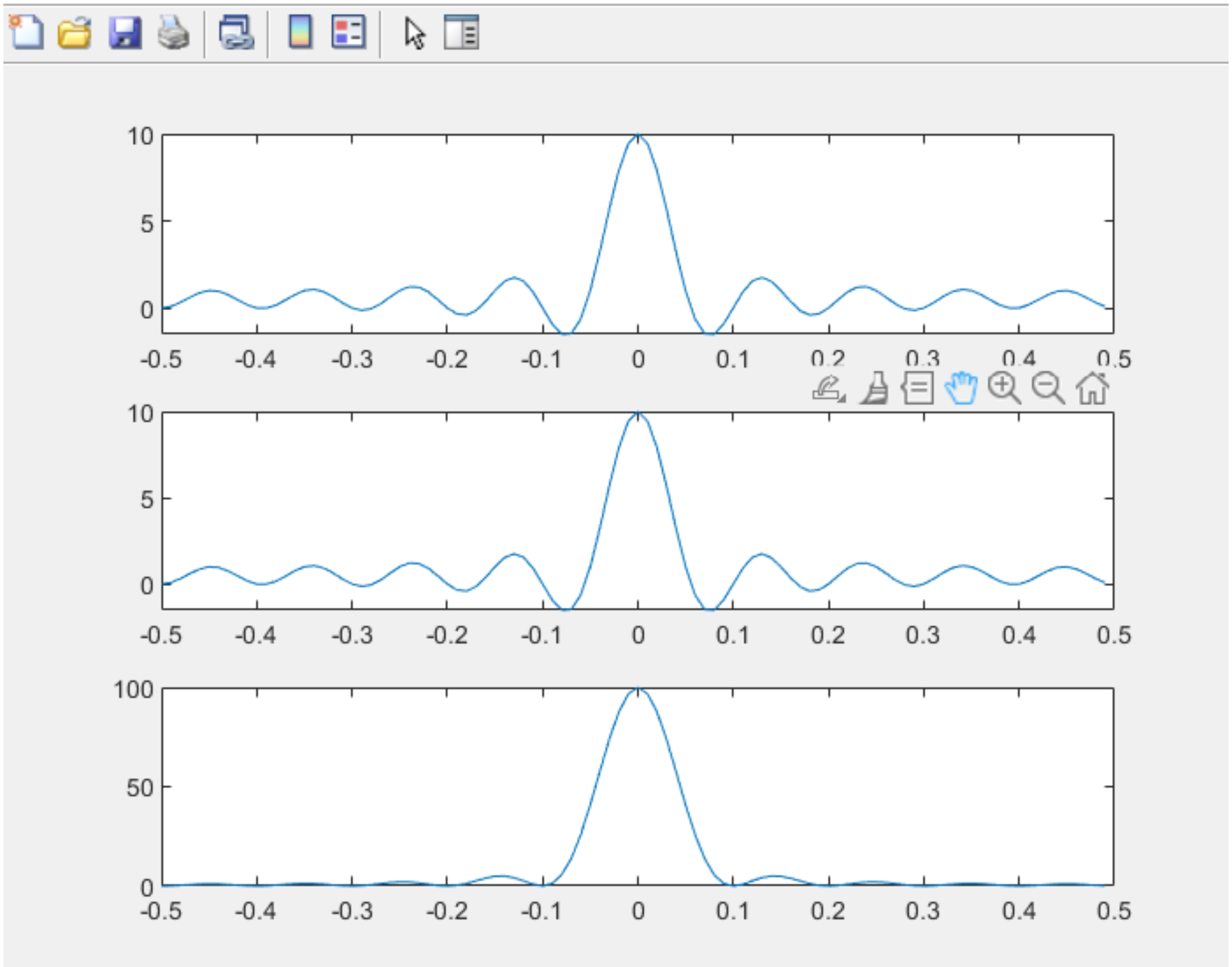
$= \frac{1}{2\pi} \int_{-\infty}^{\infty} \frac{-e^{j\omega(t-20)} + 2e^{j\omega(t-10)} - e^{j\omega t}}{\omega^2} d\omega$

$= \frac{1}{2\pi} \int_{-\infty}^{\infty} -\frac{e^{j\omega(t-20)}}{\omega^2} d\omega + \frac{1}{2\pi} \int_{-\infty}^{\infty} \frac{2e^{j\omega(t-10)}}{\omega^2} d\omega + \int_{-\infty}^{\infty} -\frac{e^{j\omega t}}{\omega^2}$

$= (t-20)u(t-20) - 2(t-10)u(t-10) + t\,u(t)$



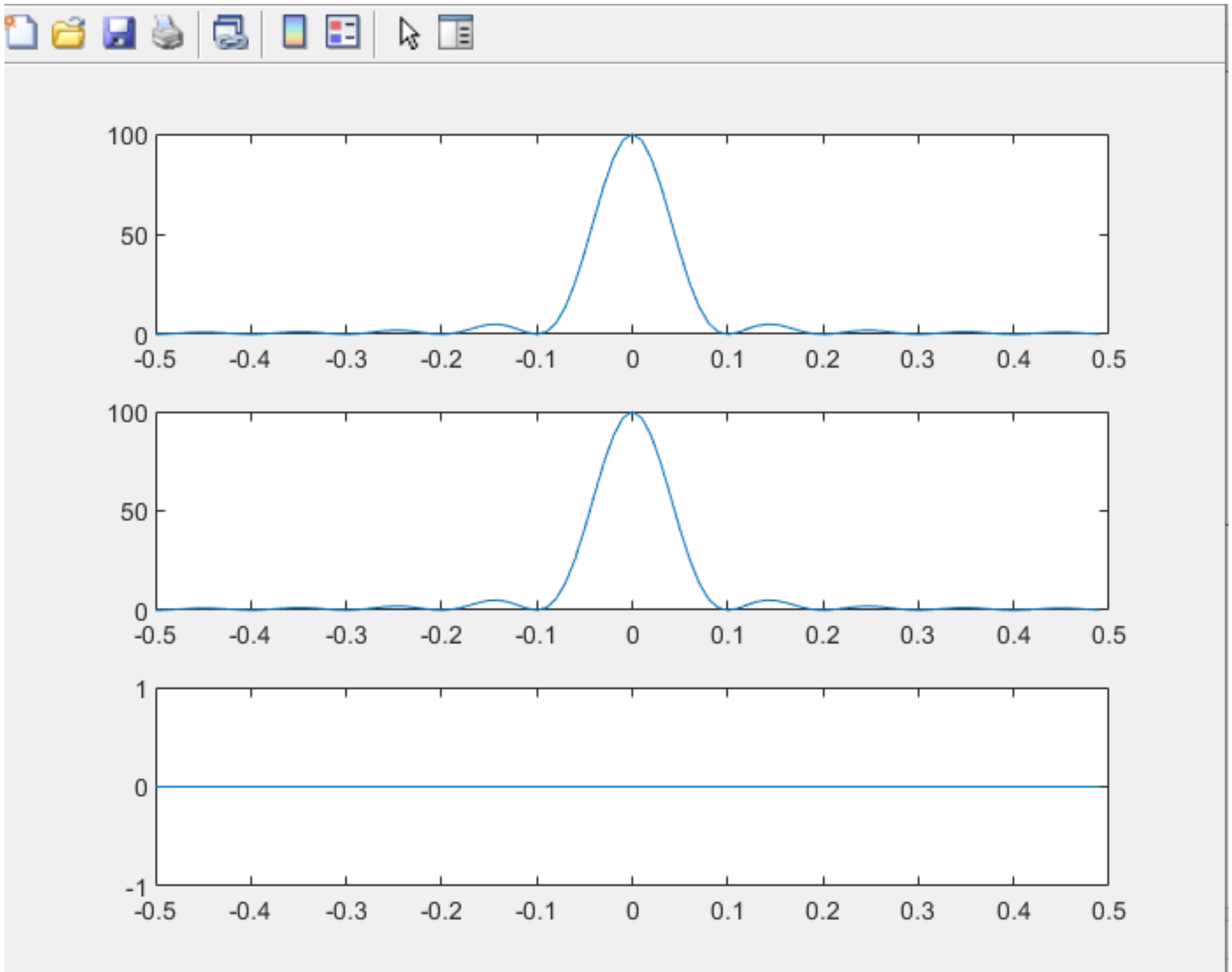
Problem A.2



Code:

```
%% A.1
N = 100;
PulseWidth = 10;
t = [0:1:(N-1)];
x = [ones(1, PulseWidth), zeros(1, N-PulseWidth)];
Xf = fft(x);
f = [-(N/2):1:(N/2)-1]*(1/N);
Zf = abs((Xf).^2);
subplot(311);plot(f,fftshift(Xf));
subplot(312);plot(f,fftshift(Xf));
subplot(313);plot(f,fftshift(Zf));
```

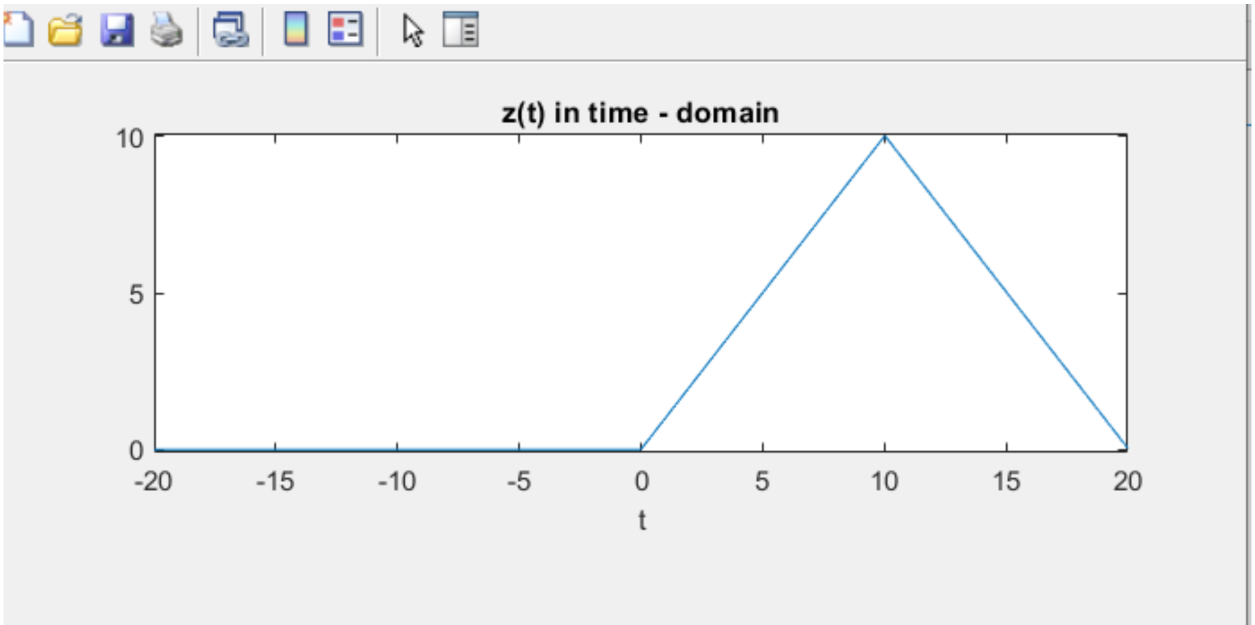
Problem A.3



Code :

```
%% A.3
N = 100;
PulseWidth = 10;
t = [0:1:(N-1)];
x = [ones(1, PulseWidth), zeros(1, N-PulseWidth)];
Xf = fft(x);
f = [-(N/2):1:(N/2)-1]*(1/N);
Zf = abs((Xf).^2);
subplot(311);plot(f,fftshift(Zf));
subplot(312);plot(f,fftshift(abs(Zf)));
subplot(313);plot(f,fftshift(angle(Zf)));
```

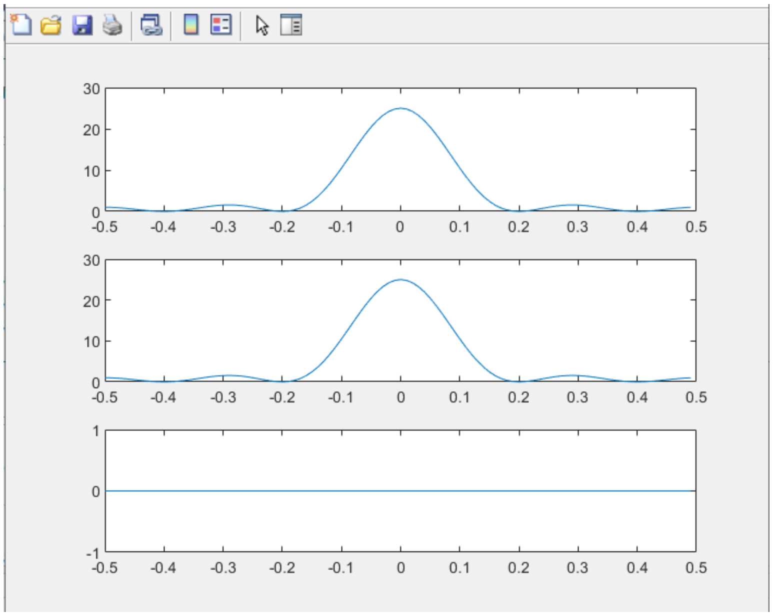
Problem A.4



Code:

```
%% A.4
t1 = -20;
t2 = 20;
N = 2000;
Delta_t = (t2 - t1)/N;
t = [t1:Delta_t:t2];
x = zeros(size(t));
x(t >= 0 & t <= 10) = 1;
x1 = x*Delta_t;
z = conv(x,x1);
subplot(2,1,1);
plot(t,z(1000:3000));
axis([t1 t2 -0.1 10.1]);
title('z(t) in time - domain');
xlabel('t');
```

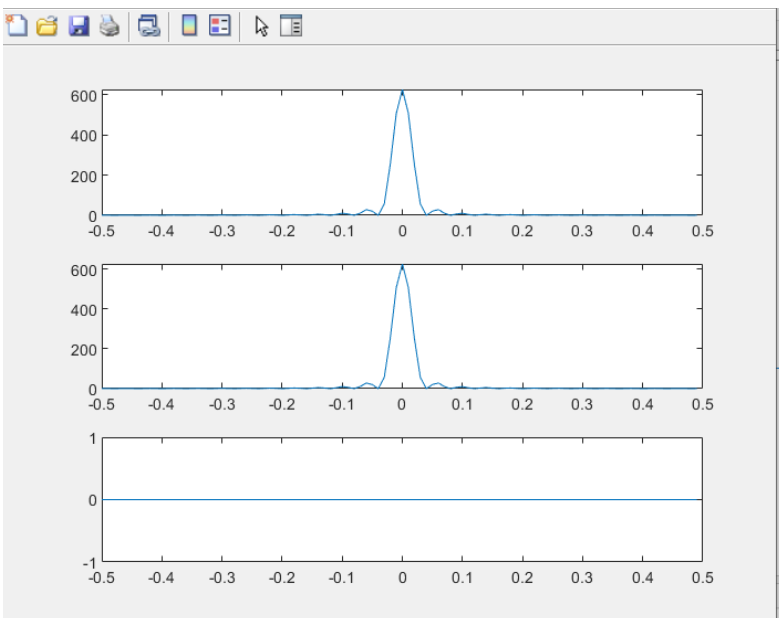
The results obtained using both time and frequency domain operations closely resemble the analytical outcomes computed in section A.1. By employing convolution in the time domain and multiplication in the frequency domain to generate z(t), we observed that both methods produced identical results. This agreement underscores a fundamental property of Fourier Transforms: the convolution of x(t) and y(t) is equivalent to the product of their respective Fourier transforms, denoted as X(w) and Y(w).



Problem A.5

Code: Part A

```
%% A.5 A
% Pulse Width of 5
N = 100;
PulseWidth = 5;
t = [0:1:(N-1)];
x = [ones(1, PulseWidth), zeros(1,
N-PulseWidth)];
Xf = fft(x);
f = [-(N/2):1:(N/2)-1]*(1/N);
Zf = abs((Xf).^2);
subplot(311);plot(f,fftshift(Zf));
subplot(312);plot(f,fftshift(abs(Zf)));
subplot(313);plot(f,fftshift(angle(Zf)));
```

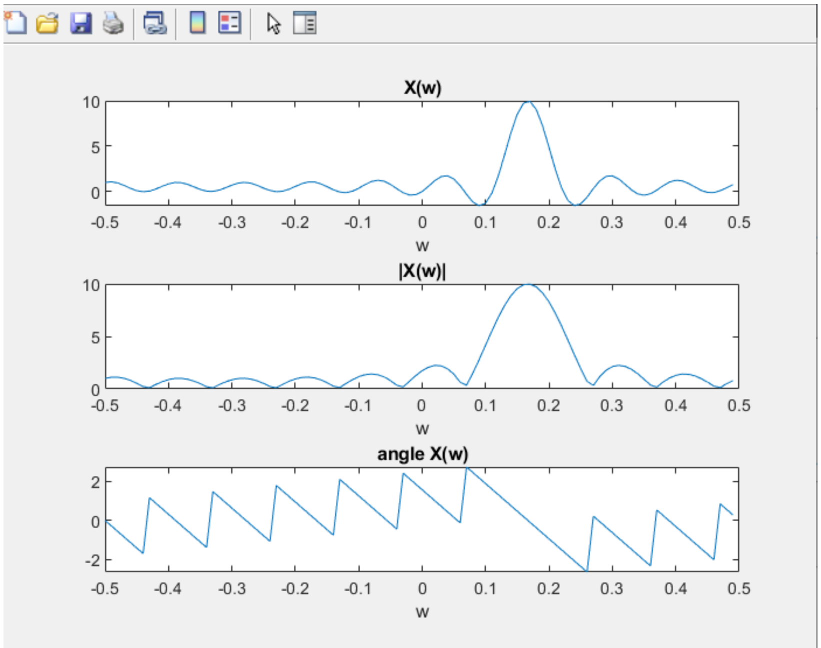


Code: Part B

```
%% A.5 B
N = 100;
PulseWidth = 25;
t = [0:1:(N-1)];
x = [ones(1, PulseWidth),
zeros(1, N-PulseWidth)];
Xf = fft(x);
f = [-(N/2):1:(N/2)-1]*(1/N);
Zf = abs((Xf).^2);
subplot(311);plot(f,fftshift(Zf));
subplot(312);plot(f,fftshift(abs(Zf)));
subplot(313);plot(f,fftshift(angle(Zf)));
```

The Fourier Transform exhibits an increase in both amplitude and frequency when the pulse width of the original function is extended. Conversely, a reduction in pulse width results in a decrease in both amplitude and frequency of the Fourier Transform. This characteristic is commonly referred to as the time-scaling property of the Fourier Transform.

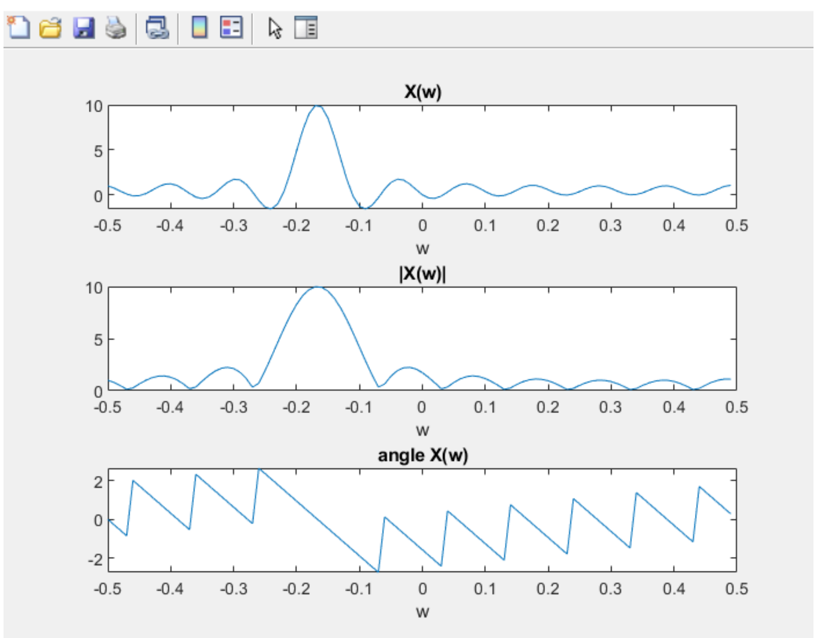
Problem A.6



w+(t):

Code:

```
N = 100;
PulseWidth = 10;
t = [0:1:(N-1)];
x = [ones(1,PulseWidth),
zeros(1,N-PulseWidth)];
wplus =
x.*exp(1j.*(pi/3).*t)
Xf = fft(wplus);
f =
[-(N/2):1:(N/2)-1]*(1/N);
figure();
subplot(311);
plot(f,fftshift(Xf));
title('X(w)');
xlabel('w');
subplot(312);
plot(f,fftshift(abs(Xf)));
title('|X(w)|');
xlabel('w');
subplot(313);
plot(f,fftshift(angle(Xf)));
title('angle X(w)');
xlabel('w');
```

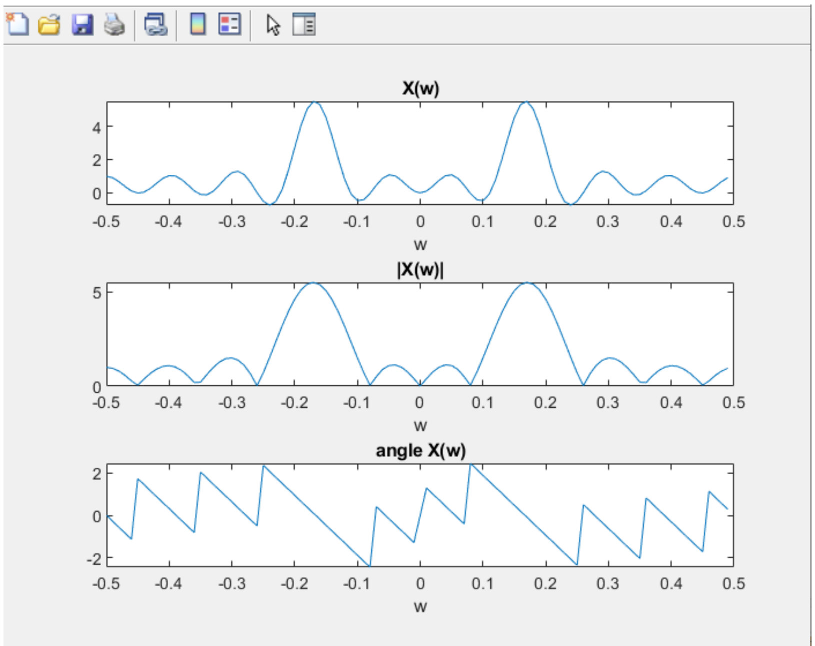


W-(t):

Code:

```
N = 100;
PulseWidth = 10;
t = [0:1:(N-1)];
x = [ones(1,PulseWidth),
zeros(1,N-PulseWidth)];
wminus =
x.*exp(-1j.*(pi/3).*t);
Xf = fft(wminus);
f =
[-(N/2):1:(N/2)-1]*(1/N);
figure();
subplot(311);
plot(f,fftshift(Xf));
title('X(w)');
xlabel('w');
subplot(312);
plot(f,fftshift(abs(Xf)));
```

```
title('|X(w)|');
xlabel('w');
subplot(313);
plot(f,fftshift(angle(Xf)));
title('angle X(w)');
xlabel('w');
```



wc(t):

Code:

```
N = 100;
PulseWidth = 10;
t = [0:1:(N-1)];
x = [ones(1,PulseWidth),
zeros(1,N-PulseWidth)];
wc = x.*cos((pi/3).*t);
Xf = fft(wc);
f =
[-(N/2):1:(N/2)-1]*(1/N);
figure();
subplot(311);
plot(f,fftshift(Xf));
title('X(w)');
xlabel('w');
subplot(312);
plot(f,fftshift(abs(Xf)));
title('|X(w)|');
xlabel('w');
subplot(313);
plot(f,fftshift(angle(Xf)));
title('angle X(w)');
xlabel('w');
```

When we multiplied the original function by a complex exponential, it caused a shift in frequency. This interesting effect is known as the frequency shift property of the Fourier Transform.

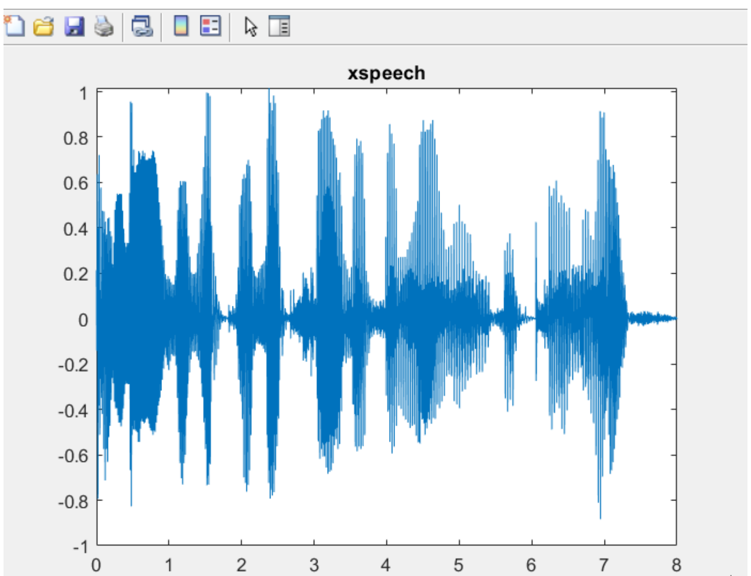
Essentially, if $x_1(t)$ **has a Fourier Transform** $X_1(jw)$**), and we multiply** $x_1(t)$ **by** e^{jw_0t}**, the resulting function** $x_2(t)$ **will have a Fourier Transform** $X_2(jw)$ **equal to** $X_1(j(w - w_0))$**. This means that the entire frequency content of the signal** $x_1(t)$ **is shifted by** w_0 **in the frequency domain.**

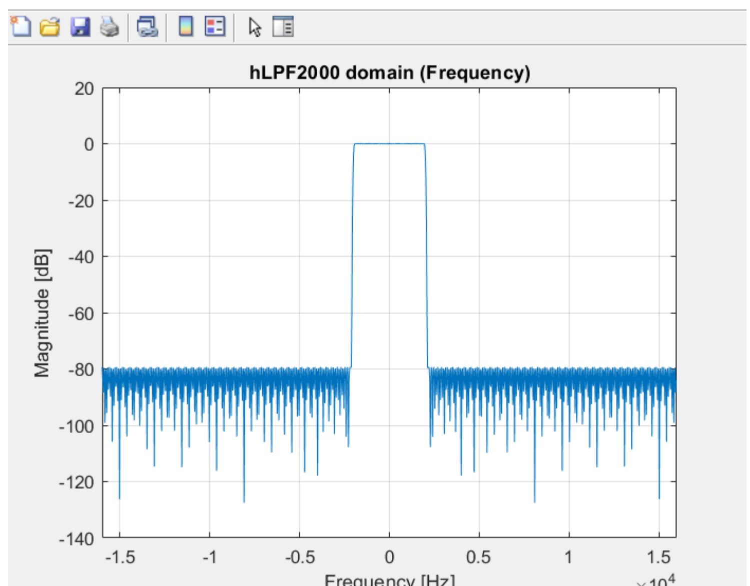
Equation (includng properity): $x_2(t) \ = e^{jw_0t} \cdot x_1(t) \ \text{-->} \ X_2(jw) \ = X_1(j(w - w_0))$

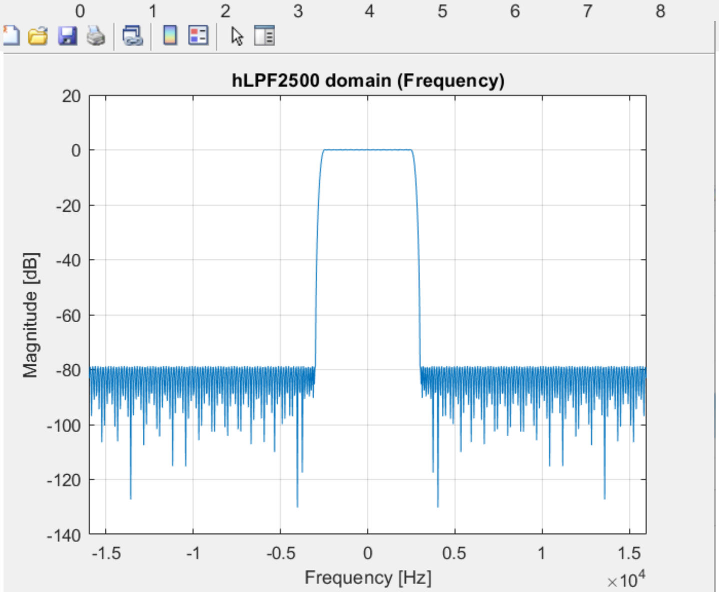
Problem B

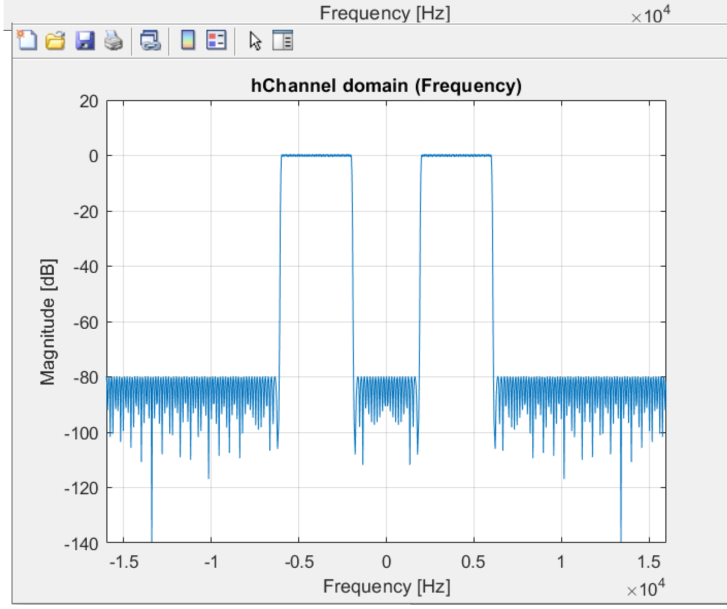
Codes:

```
%% Part B.1
load('Lab4_Data.mat');
% plotting all the values from the data given on a graph for a visual
% repersentation as well as a baseline for the encoder and decoder
figure (1)
plot (xspeech)
title('xspeech');
figure (2)
MagSpect(hChannel)
title('hChannel domain (Frequency)');
figure(3)
MagSpect(hLPF2000)
title('hLPF2000 domain (Frequency)');
figure(4)
MagSpect(hLPF2500)
title('hLPF2500 domain (Frequency)');
```

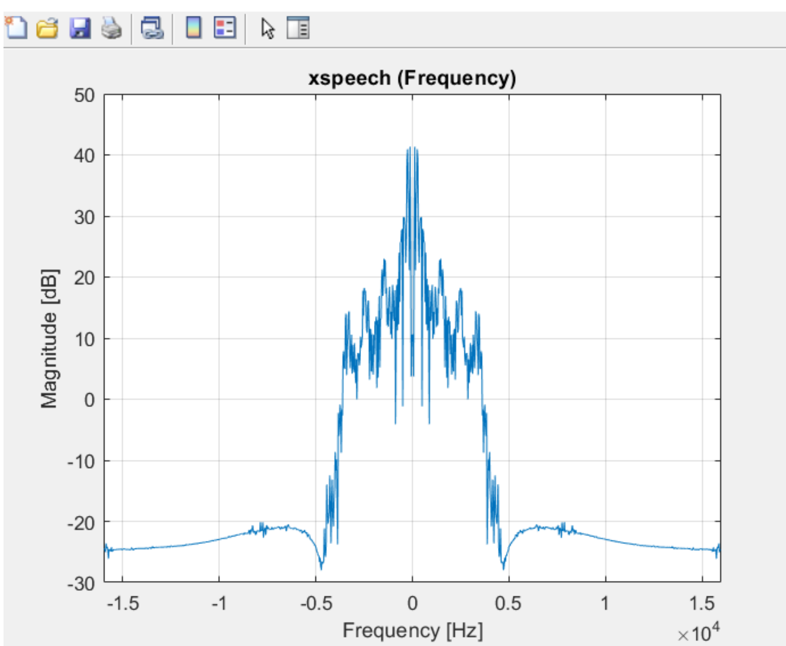


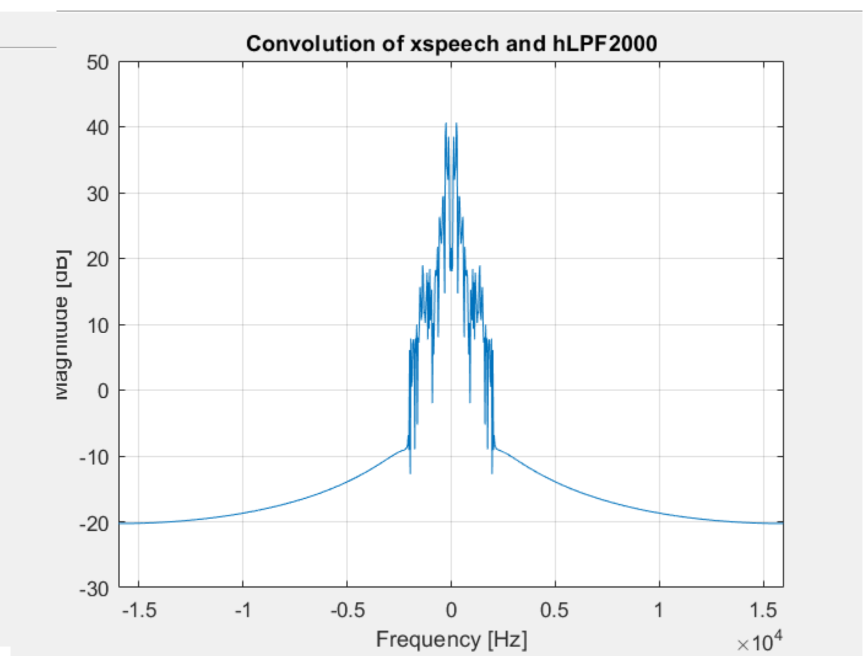


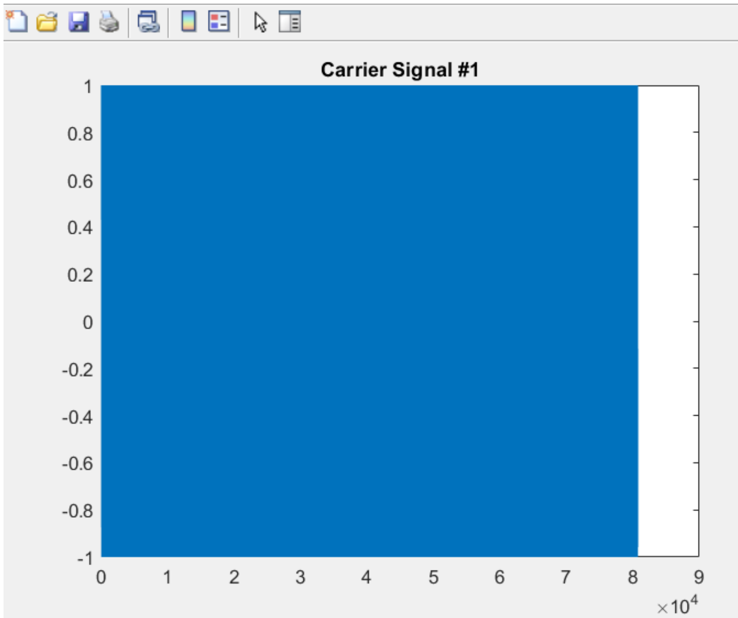


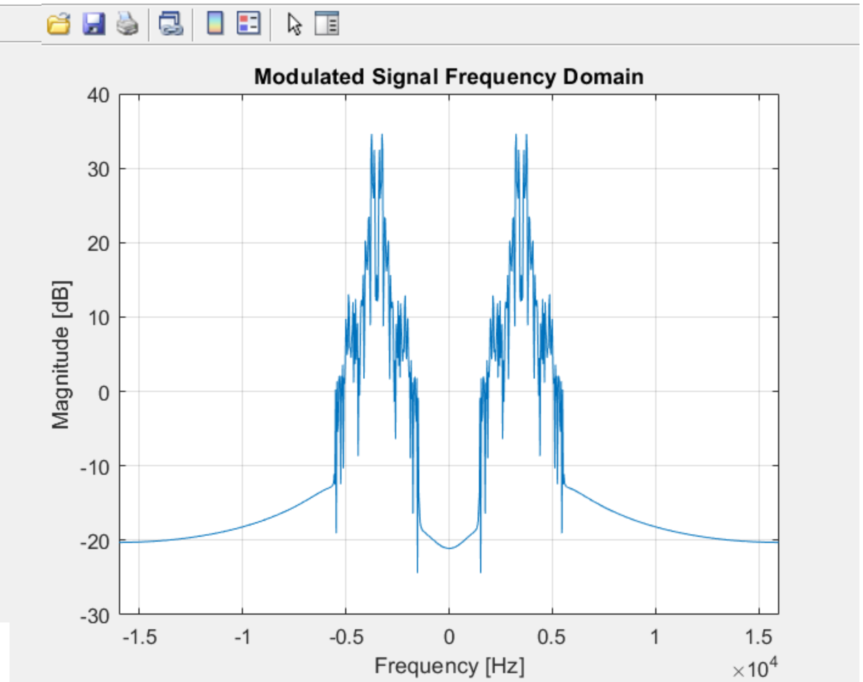


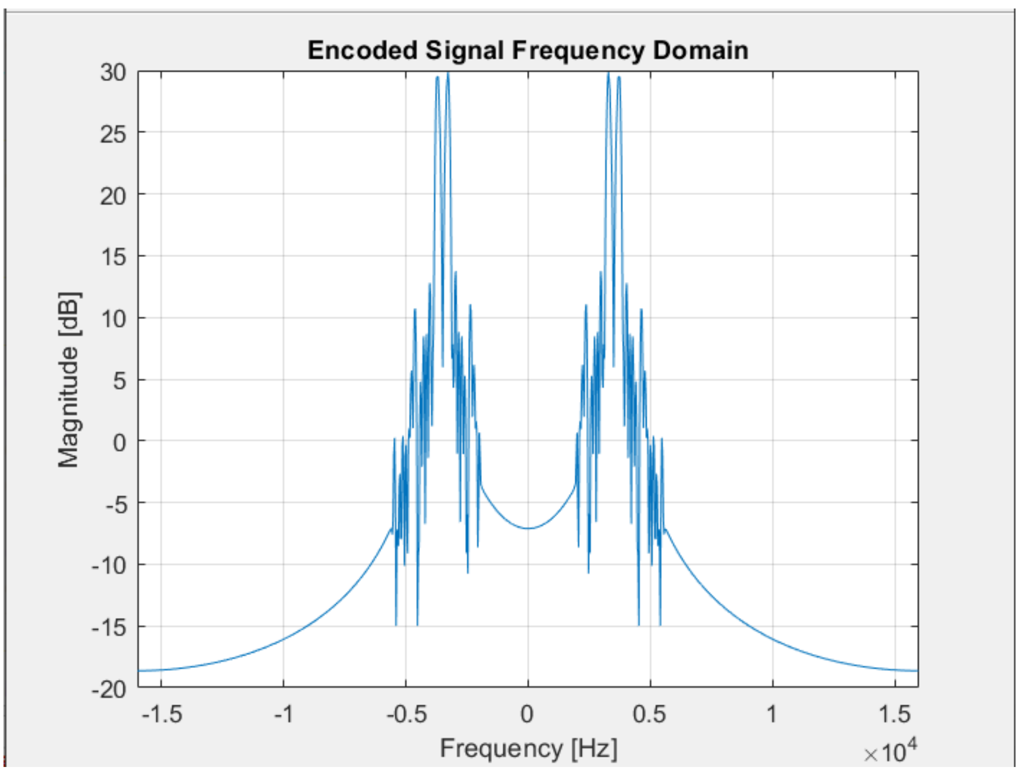
```
%% Encoder:
% The audio file will be encoded with the following sets of codes below. To
% encode the file, the first step will be to convolute the audio file with
% a low pass filter to ensure the higher frequency are removed. This is
% because it is not needed. Secondly, the convoluted signal are multiplied
% with the Carrier signal function (from osc.m) to ensure efficient
% transmission. At the very end, he product is convoluted with the
% hChannel to get the final encoded signal.
MagSpect(xspeech)
title('xspeech (Frequency)');
Convolution_1 = conv(xspeech,hLPF2000);
figure(5)
MagSpect(Convolution_1)
title('Convolution of xspeech and hLPF2000');
Carrier = osc(3500,80710,32000);
figure(6)
plot(Carrier)
title('Carrier Signal #1');
Mod = Convolution_1.*Carrier;
figure(7)
MagSpect(Mod)
title('Modulated Signal Frequency Domain');
Output = conv(Mod, hChannel);
figure(8)
MagSpect(Output)
title('Encoded Signal Frequency Domain');
```



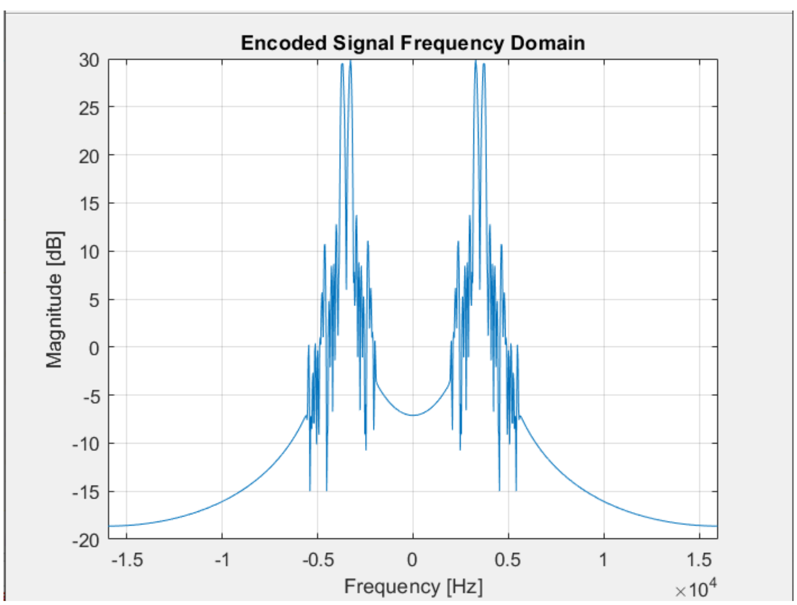


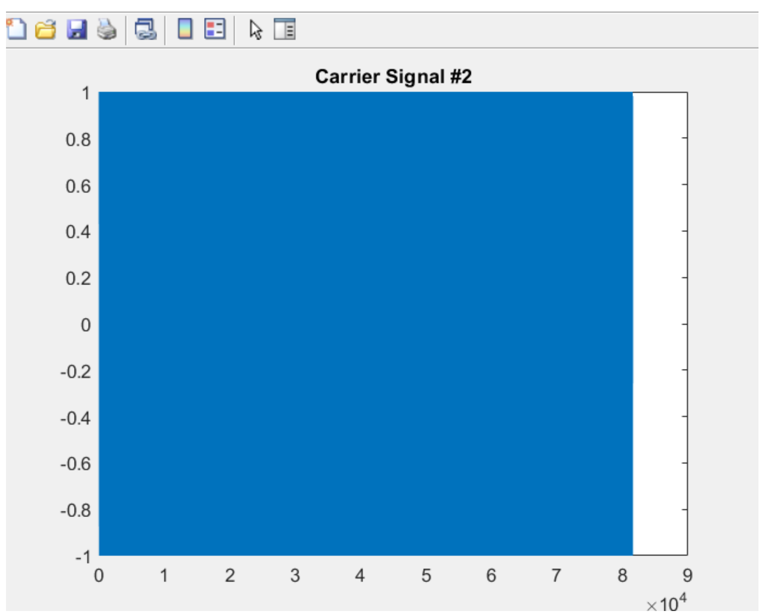

Carrier Signal #1
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0 1 2 3 4 5 6 7 8 9
$\times 10^4$

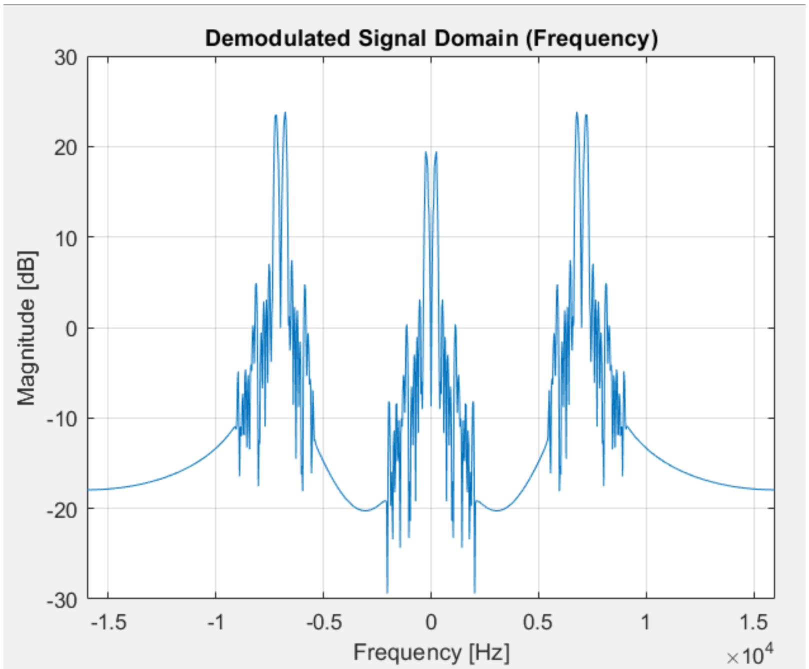

Modulated Signal Frequency Domain
Magnitude [dB]
40
30
20
10
0
-10
-20
-30
-1.5 -1 -0.5 0 0.5 1 1.5
Frequency [Hz]
$\times 10^4$

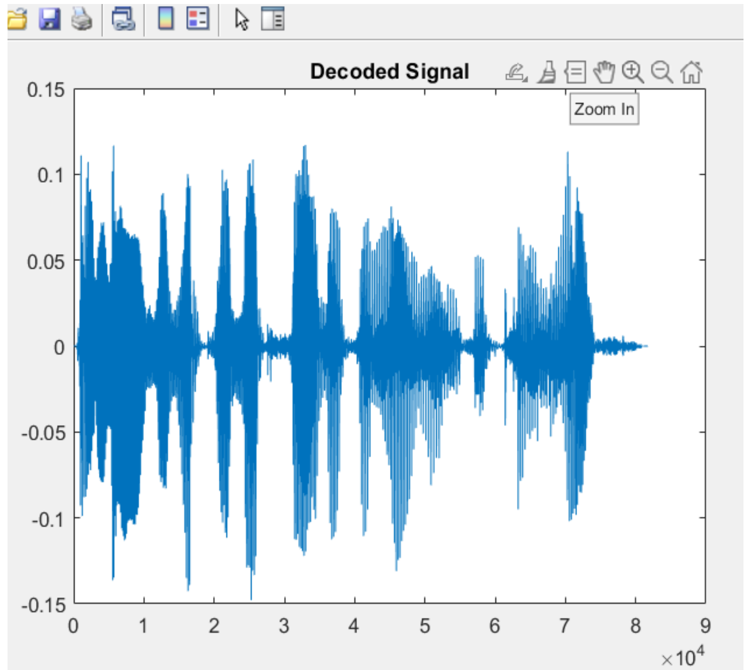

Encoded Signal Frequency Domain
Magnitude [dB]
30
25
20
15
10
5
0
-5
-10
-15
-20
-1.5 -1 -0.5 0 0.5 1 1.5
Frequency [Hz]
$\times 10^4$

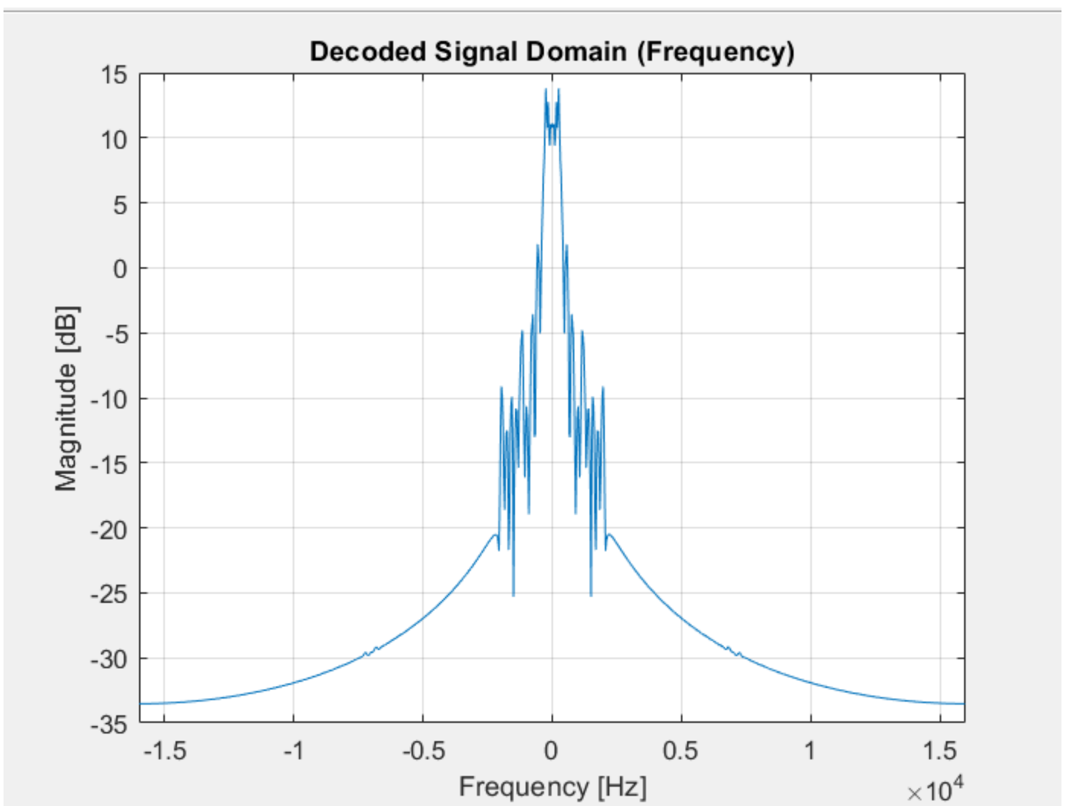
```
%% Decoder:
% To start off, the signal is multiplied with the carrier signal. Secondly,
% the function is convoluted with the second low-pass filter to remove the
% higher frequencies which will return the final signal.
% %centeral Freq change.
Carrier_2 = osc(3500,81520,32000);
figure(9)
plot(Carrier_2)
title('Carrier Signal #2');
Demod = Output.*Carrier_2;
figure(10)
MagSpect(Demod)
title('Demodulated Signal Domain (Frequency)');
new_xspeech = conv(Demod, hLPF2500);
figure(11)
plot(new_xspeech)
title('Decoded Signal');
figure(12)
MagSpect(new_xspeech)
sound(new_xspeech,32000)
title('Decoded Signal Domain (Frequency)');
% 1. Encoded the audio file using the hLPF2000 low pass filter
% 2. Carrier signal generated by the osc.m file and the hChannel signal
% 3. Encoded signal is decoded using another carrier signal (from)the osc.m.
% and hLPF25000 low pass filter.
```





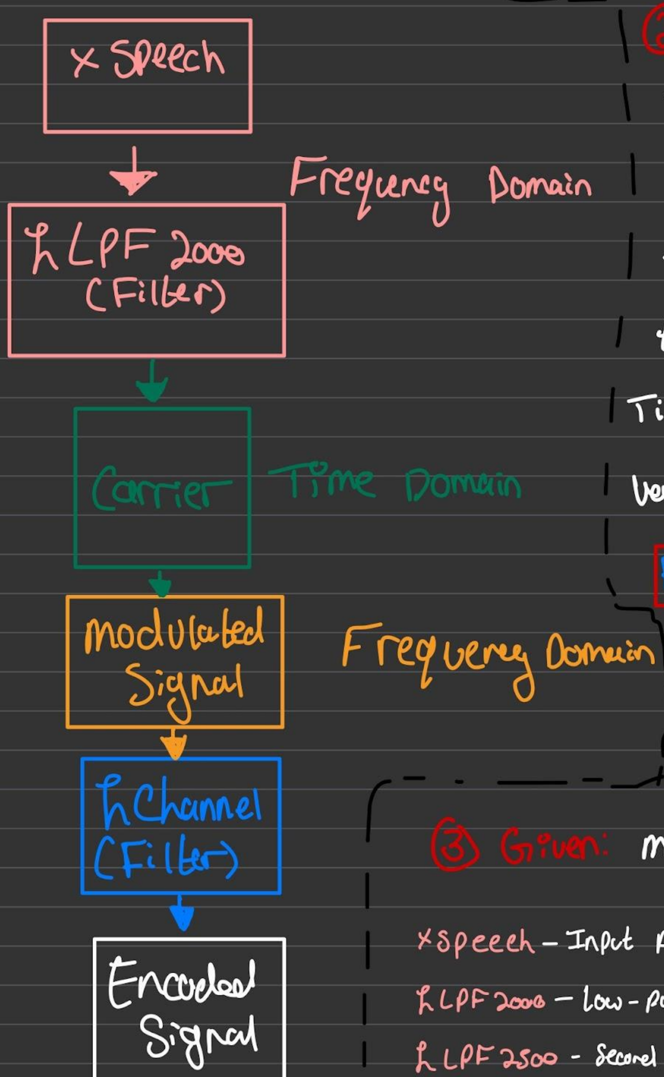

Demodulated Signal Domain (Frequency)
Magnitude [dB]
Frequency [Hz]
×10^4


Decoded Signal
Zoom In
×10^4


Decoded Signal Domain (Frequency)
Magnitude [dB]
Frequency [Hz]
×10^4

① Purpose:

Implement an audio transmission using an encoder and decoder. Convolution of xSpeech with low pass filter (hLPF 2000), modulating the result with a carrier signal, then convoluting again with the 'h Channel' filter. For decoder, demodulating the signal, filter with the low-pass filter ('hLPF 2500').



② Domain

-> Analysis in both Frequency & Time. Frequency is used to visualize the signal & Filters in terms of their frequency components. The Time domain is used to show variation of signal over time.

Frequency Analysis

↳ Time-Frequency Domain

↳ Frequency Analysis

③ Given:

Magnitude spectra is plotted using the 'MagSpec' function

xSpeech – Input Audio Signal

hLPF 2000 – Low-pass filter

hLPF 2500 – Second low pass filter

Carrier – First Carry Signal.

Carrier_2 – Second Carry Signal

hChannel – Channel Filter

④ Coder:

Convolution		modulation		
x Speech →	(hLPF2000) →	(carrier) →	hChannel →	Encoded Signal
Frequency Domain →	Time Domain →	Frequency Domain →	Frequency Domain	↑

→ Magnitude spectra is plotted after each step for a visual representation of the frequency domain

⑤ Decoder

	Demod.	Convolution	
Encoded Signal →	(carrier_2) →	(hLPF 2500) →	Decoded Signal
Frequency Domain →	Frequency Domain →	Frequency Domain	↑

→ Magnitude spectra is plotted after each step for a visual representation of the frequency domain

After all the steps completed above, an audio sound will play