

PROJECT SUBMISSION



**Program: Electronics and
Communications Engineering**

Course Code: ECE252

***Course Name: Fundamentals of
Communication Systems***

Examination Committee

Dr. Hussein Kotb

Dr. Shimaa M. Morsy

Dr. Alaa Eldin Fathy

Ain Shams University

Faculty of Engineering

Spring Semester – 2021

Student Personal Information for Group Work

Student Names:

Mohamed Karam Fouad

Mohamed Khaled Mohamed elsayed abdelwahab

Muhammad Ahmad Youssef Muhammad Shaheen

Muhammed Hossam EL-Din Mostafa Sabry

Mustafa Ahmed Allam Attia

Student Codes:

1803669

1804115

1806123

1801799

1803488

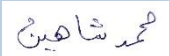
Team 24

Plagiarism Statement

I certify that this assignment / report is my own work, based on my personal study and/or research and that I have acknowledged all material and sources used in its preparation, whether they are books, articles, reports, lecture notes, and any other kind of document, electronic or personal communication. I also certify that this assignment / report has not been previously been submitted for assessment for another course. I certify that I have not copied in part or whole or otherwise plagiarized the work of other students and / or persons.

Signature / Student Name:

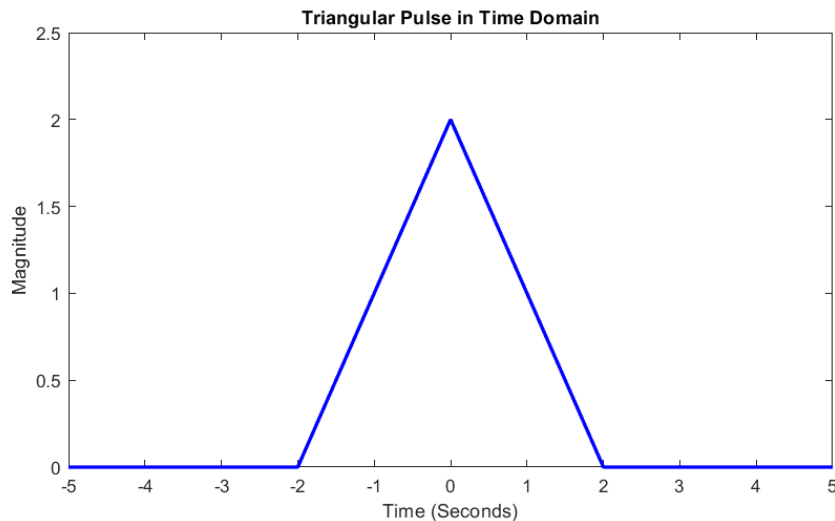
Date: 7/12/2021



Analog Communication

Part I:

1. Function's plot using MATLAB



2. Derivation for the Fourier Transform Analytically

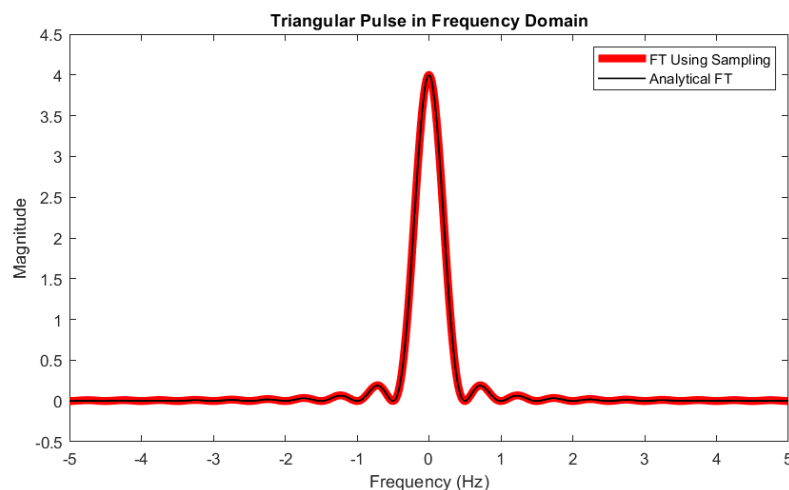
The Fourier Transform for a triangular pulse in general is given by the following expression:

$$X(f) = \text{Area of Triangle} * \text{sinc}^2\left(\frac{\text{Base Length}}{2} * f\right)$$

For the triangular pulse given, the Fourier Transform is given by:

$$X(f) = 4 \text{sinc}^2(2f) = 4 \frac{\sin^2(2\pi f)}{(2\pi f)^2}$$

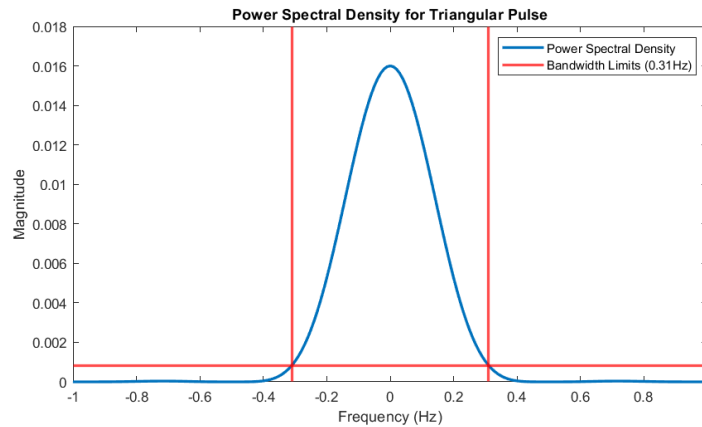
3. Calculating the Fourier Transform Using MATLAB



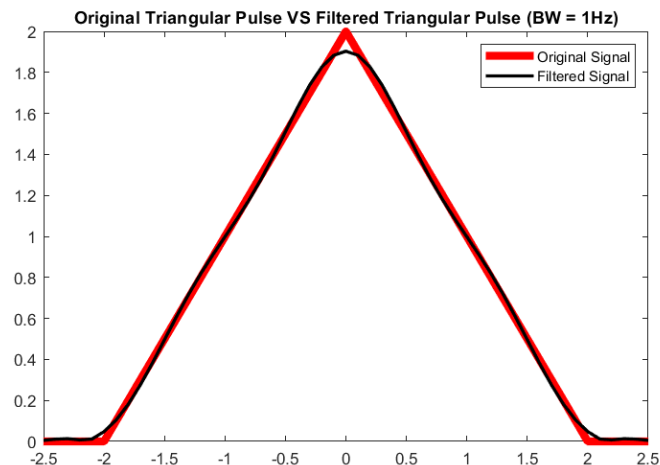
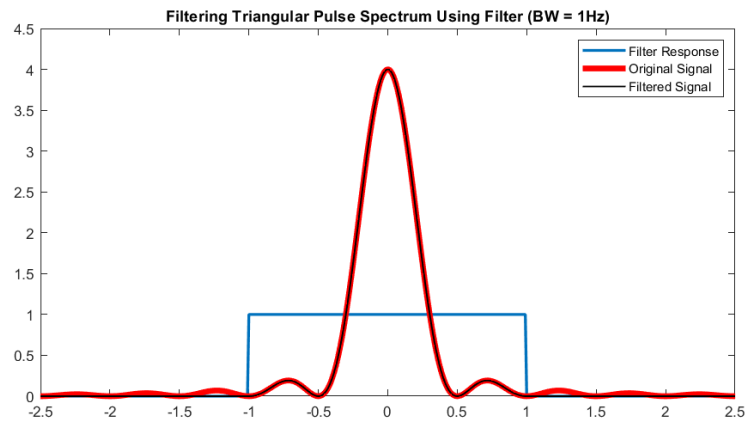
From the previous plot, we can see that the analytical results completely agree with the simulation results.

4. Signal Bandwidth

The red lines in the following figure limit the signal's bandwidth ($BW = 0.31\text{Hz}$) at which the amplitude falls to 5% of its maximum.

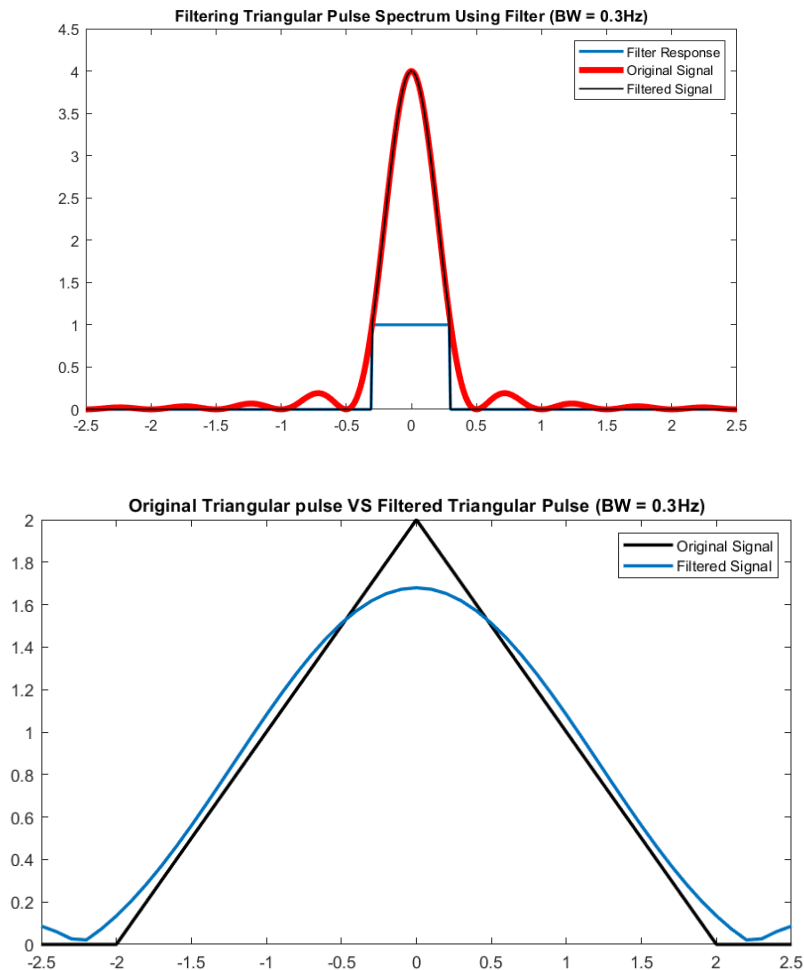


5. Passing the signal through a LPF (BW = 1 Hz)



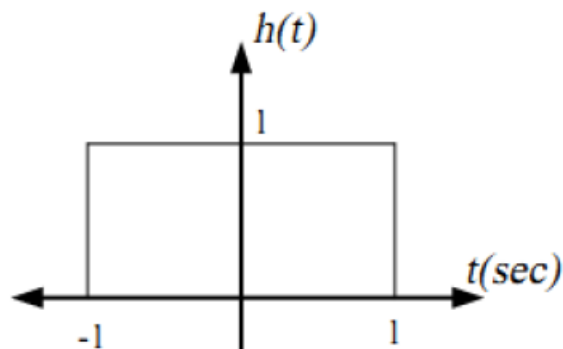
It's clear that the filtered signal is slightly distorted, due to the loss of high frequency components.

6. Passing the signal through a LPF (BW = 0.3 Hz)

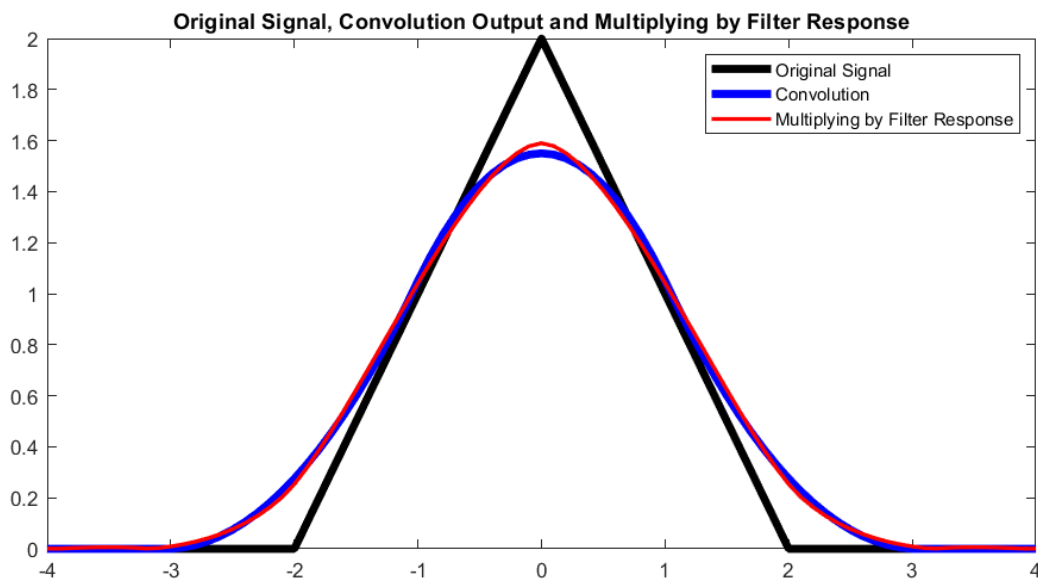


The previous plots show significant attenuation to the signal, as the used filter's bandwidth decreased significantly, increasing the amount of lost high frequency components.

7. Passing the signal through the given filter

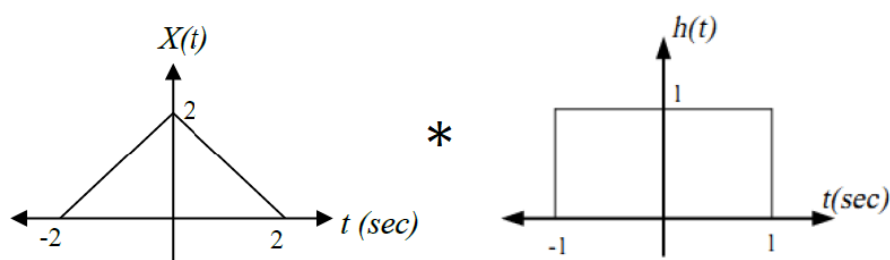


The filter's response in time domain is represented by a rectangular pulse, which means it will have a frequency response represented by a sinc function.



We can see from the above plot that the resulting filtered signal is distorted, as the frequency response of the used filter is not constant, so it multiplies different frequency components of the signal by different amplitudes, causing distortion. We can also see that the results obtained from convolution are very similar to those obtained from multiplying by the frequency response of the filter in the frequency domain and hence, verifying the convolution theorem.

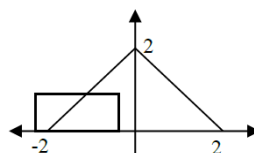
8. Analytical Expression Derivation



Region 1:

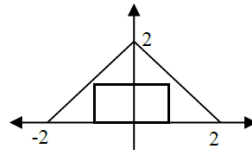
$$z(t) = 0, \quad t \leq -3$$

Region 2:



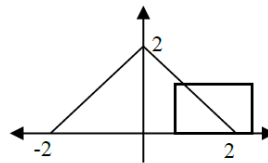
$$z(t) = \int_{-2}^{t+1} (2+t) dt = \frac{1}{2}t^2 + 3t + \frac{9}{2}, \quad -3 \leq t \leq -1$$

Region 3:



$$z(t) = \int_{t-1}^0 (2+t) dt + \int_0^{t+1} (2-t) dt = -t^2 + 3, \quad -1 \leq t \leq 1$$

Region 4:



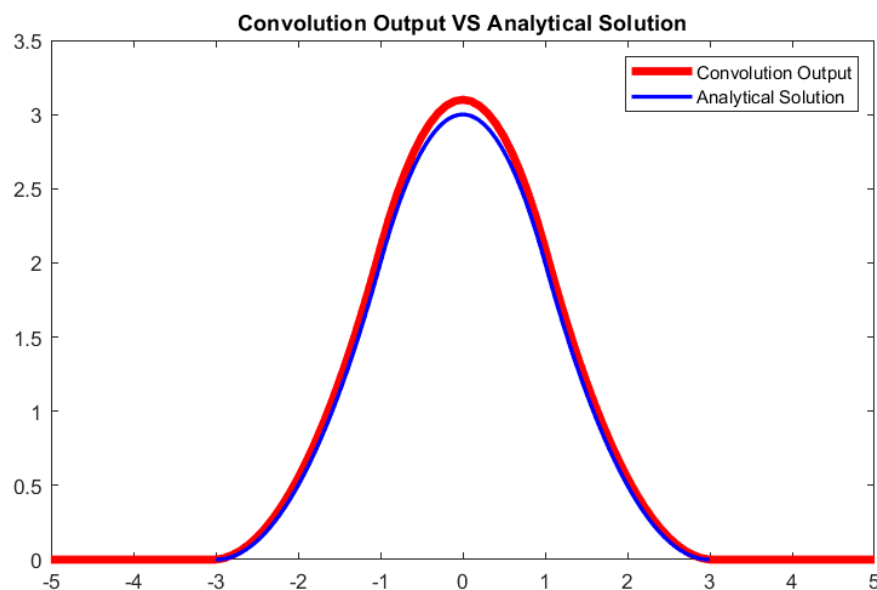
$$z(t) = \int_{t-1}^2 (2-t) dt + \int_0^{t+1} (2-t) dt = \frac{1}{2}t^2 - 3t + \frac{9}{2}, \quad 1 \leq t \leq 3$$

Region 5:

$$z(t) = 0, \quad t \geq 3$$

Finally, we have the analytical expression:

$$z(t) = \begin{cases} \frac{1}{2}t^2 + 3t + \frac{9}{2}, & -3 \leq t \leq -1 \\ -t^2 + 3, & -1 \leq t \leq 1 \\ \frac{1}{2}t^2 - 3t + \frac{9}{2}, & 1 \leq t \leq 3 \\ z(t) = 0, & \text{otherwise} \end{cases}$$





9. MATLAB Code

```
Ts = 1/10;
t = -50:Ts:50-Ts;
fs = 1/Ts;

x = 2*tripuls(t,4);
figure(1)
plot(t, x, 'b','linewidth',2);
title({'','Triangular Pulse in Time Domain'});
xlim ([-5,5]);
ylim ([0,2.5]);
xlabel('Time (Seconds)')
ylabel('Magnitude')

y = fft(x);
n = length(x);
fshift = (-n/2:n/2-1)*(fs/n);
yshift = fftshift(y);
figure(2)
plot(fshift,0.1*abs(yshift),'r','linewidth',5)
Freq_response = abs(yshift);
hold on
syms m;
fplot(4*(sin(2*pi*m)/(2*pi*m))^2,'black','linewidth',1)
ylim ([-0.5,4.5]);
xlabel('Frequency (Hz)')
ylabel('Magnitude')
title('Triangular Pulse in Frequency Domain');
legend('FT Using Sampling','Analytical FT')

figure(3)
power_spectrum = ((0.1*abs(yshift)).^2/n);

for i = 1:length(power_spectrum)
    if 0.045 * max(power_spectrum)<= power_spectrum(i) & power_spectrum(i)
<= 0.055 * max(power_spectrum) & fshift(i)>0
        BW = fshift(i);
        y = (power_spectrum(i));
    end
end
end
```




```
plot(fshift,power_spectrum,'linewidth',2)
hold on

yline(y,'r','linewidth',2);
xline(BW,'r','linewidth',2);
xline(-BW,'r','linewidth',2);
title('Power Spectral Density for Triangular Pulse');
xlabel('Frequency (Hz)')
ylabel('Magnitude')
xlim([-1 1])
legend('Power Spectral Density','Bandwidth Limits (0.31Hz)')

%% Filter

Filter1 = rectpuls(fshift,2);

figure(4)
plot(fshift,Filter1,'linewidth',2)
hold on
Filtered_signal = 0.1*Freq_response.*Filter1;
plot(fshift,0.1*Freq_response,'red','linewidth',4)
plot(fshift,Filtered_signal,'black','linewidth',1)
xlim ([-2.5,2.5]);
title('Filtering Triangular Pulse Spectrum Using Filter (BW = 1Hz)');
legend('Filter Response','Original Signal','Filtered Signal')

z = ifft(Filtered_signal);
k = length(Filtered_signal);
fshift1 = (-k/2:k/2-1)*(fs/k);
yshift1 = ifftshift(z);

figure(5)
plot(t, x, 'r','linewidth',5)
hold on
plot(t,10*abs(yshift1),'black','linewidth',2)
xlim ([-2.5,2.5]);
title('Original Triangular Pulse VS Filtered Triangular Pulse (BW = 1Hz)');
legend('Original Signal','Filtered Signal')
```



```
Filter2 = rectpuls(fshift,0.6);

figure(6)
Filtered_signal_2 = 0.1*Freq_response.*Filter2;
plot(fshift,Filter2,'linewidth',2)
hold on
plot(fshift,0.1*Freq_response,'red','linewidth',4)
plot(fshift,Filtered_signal_2,'black','linewidth',1)
xlim ([-2.5,2.5]);
title('Filtering Triangular Pulse Spectrum Using Filter (BW = 0.3Hz)');
legend('Filter Response','Original Signal','Filtered Signal')

z = ifft(Filtered_signal_2);
k = length(Filtered_signal_2);
yshift1 = ifftshift(z);

figure(7)
plot(t, x, 'black','linewidth',2)
hold on
plot(t,10*abs(yshift1),'linewidth',2)
xlim ([-2.5,2.5]);
title('Original Triangular pulse VS Filtered Triangular Pulse (BW = 0.3Hz)');
legend('Original Signal','Filtered Signal')

%% Convolution

Ts = 1/10;
t = -50:Ts:50-Ts;

x = 2*tripuls(t,4);
y = fft(x);
Freq_response = abs(yshift);

Filter3 = rectpuls(t,2);
figure(8)
title('Original Triangular pulse VS Filtered Triangular Pulse');
plot(t,x,'black','linewidth',4)
hold on
convo = conv(x,Filter3,'same');
plot(t,0.05*convo,'blue','linewidth',4)
```



```
%% Multiply by filter response

yshift = fftshift(y);
y = fft(Filter3);
n = length(Filter3);
fshift = (-n/2:n/2-1)*(fs/n);
yshift = fftshift(y);
Freq_response_2 = abs(yshift);
Filtered_signal_3 = 0.1*Freq_response.*Freq_response_2;

z = ifft(Filtered_signal_3);
k = length(Filtered_signal_3);
yshift1 = ifftshift(z);
plot(t,0.5*abs(yshift1),'r','linewidth',2)
xlim([-4,4]);
title('Original Signal, Convolution Output and Multiplying by Filter
Response');
legend('Original Signal','Convolution','Multiplying by Filter Response')

%% Analytical

figure(9)

a = -3:0.01:-1;
b = -1:0.01:1;
c = 1:0.01:3;

z = (a.^2/2 + 3.*a + 4.5);      % Analytical Expression
x = (-b.^2 + 3);               % Analytical Expression
y = (c.^2/2 - 3.*c + 4.5);      % Analytical Expression

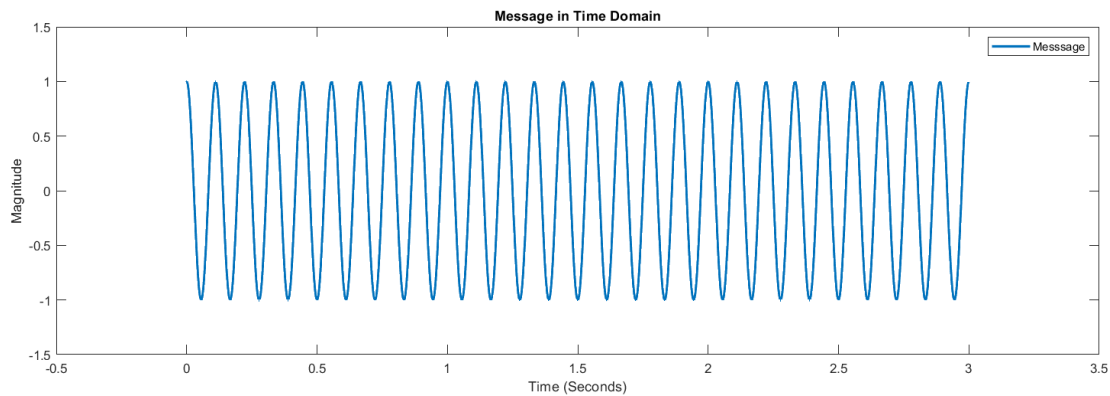
plot(t,0.1*convo,'red','linewidth',4)
hold on
plot(a,z,'b','linewidth',2)
plot(b,x,'b','linewidth',2)
plot(c,y,'b','linewidth',2)
title('Convolution Output VS Analytical Solution');
xlim([-5 5]);
legend('Convolution Output','Analytical Solution')
```

Part II:

For the message defined as:

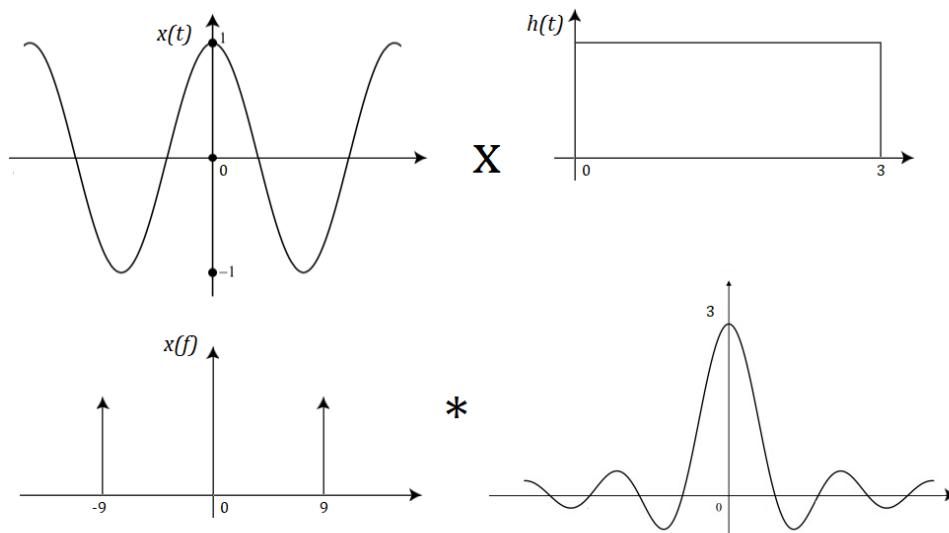
$$m(t) = \begin{cases} \cos(2\pi * 9 * t) & 0 < t < 3 \\ 0 & \text{otherwise} \end{cases}$$

1. Message Plot



2. Analytical Expression for the Fourier Transform

Method 1 (Using the convolution theory):



$$M(f) = 3(\text{sinc}(3(f - 9))e^{-3\pi(f-9)} + \text{sinc}(3(f + 9))e^{-3\pi(f+9)})$$

Method 2 (Using the Fourier Transform definition)

$$M(f) = \int_0^3 \cos(2\pi * 9 * t) e^{-j2\pi ft} dt = \frac{1}{2} \int_0^3 (e^{j2\pi * 9t} + e^{-j2\pi * 9t}) e^{-j2\pi ft} dt$$

$$M(f) = \frac{1}{2} \int_0^3 (e^{j2\pi*(9-f)t} + e^{-j2\pi*(9+f)t}) dt$$

$$M(f) = \frac{1}{2} \left[\frac{e^{j2\pi*(9-f)t}}{j2\pi*(9-f)} + \frac{e^{-j2\pi*(9+f)t}}{-j2\pi*(9+f)} \right]_0^3$$

$$M(f) = \frac{1}{2} \left[\frac{e^{j2\pi*(9-f)*3} - 1}{j2\pi*(9-f)} + \frac{e^{-j2\pi*(9+f)*3} - 1}{-j2\pi*(9+f)} \right]$$

$$M(f) = \frac{1}{j4\pi} \left[\frac{e^{j2\pi*(9-f)*3} - 1}{(9-f)} - \frac{e^{-j2\pi*(9+f)*3} - 1}{(9+f)} \right]$$

$$= \frac{1}{j4\pi} \left[\frac{(9+f)e^{j2\pi*(9-f)*3} - (9+f) - (9-f)e^{-j2\pi*(9+f)*3} + 9-f}{81-f^2} \right]$$

$$= \frac{1}{j4\pi} \left[\frac{9e^{j2\pi*27}e^{-j2\pi*3f} + fe^{j2\pi*27}e^{-j2\pi*3f} - 9e^{-j2\pi*27}e^{-j2\pi*3f} + fe^{-j2\pi*27}e^{-j2\pi*3f} - 2f}{81-f^2} \right]$$

$$= \frac{1}{j4\pi(81-f^2)} [9e^{-j2\pi*3f}(e^{j2\pi*27} - e^{-j2\pi*27}) + fe^{-j2\pi*3f}(e^{j2\pi*27} + e^{-j2\pi*27}) - 2f]$$

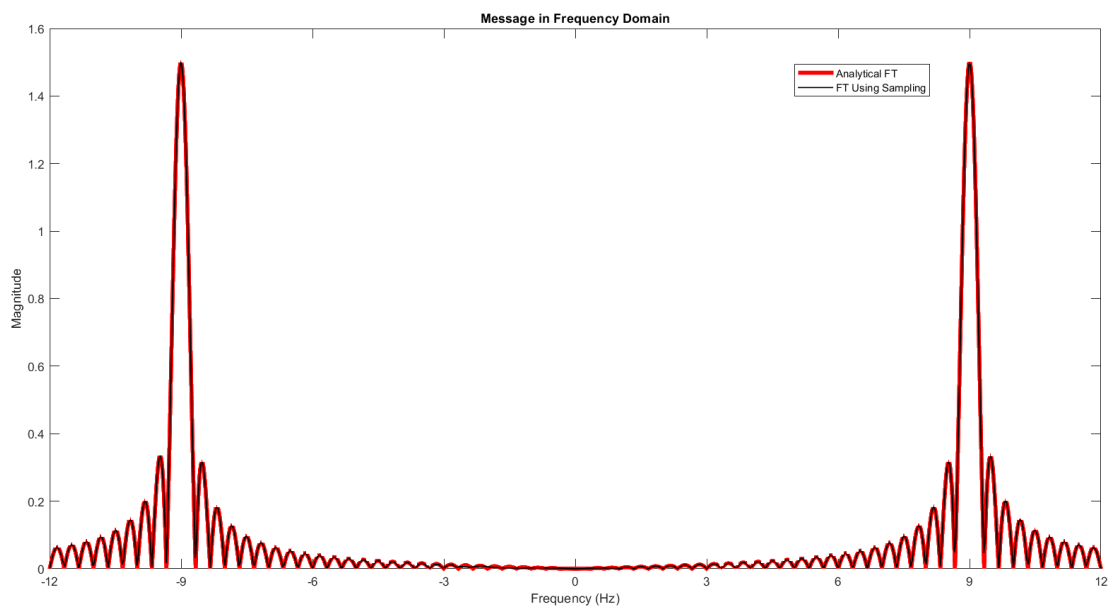
$$= \frac{1}{j4\pi(81-f^2)} [18je^{-j2\pi*3f} \sin(2\pi*27) + 2fe^{-j2\pi*3f} \cos(2\pi*27) - 2f]$$

$$= \frac{1}{j4\pi(81-f^2)} [2fe^{-j2\pi*3f} \cos(2\pi*27) - 2f]$$

$$= \frac{2f}{j4\pi(81-f^2)} [e^{-j2\pi*3f} - 1]$$

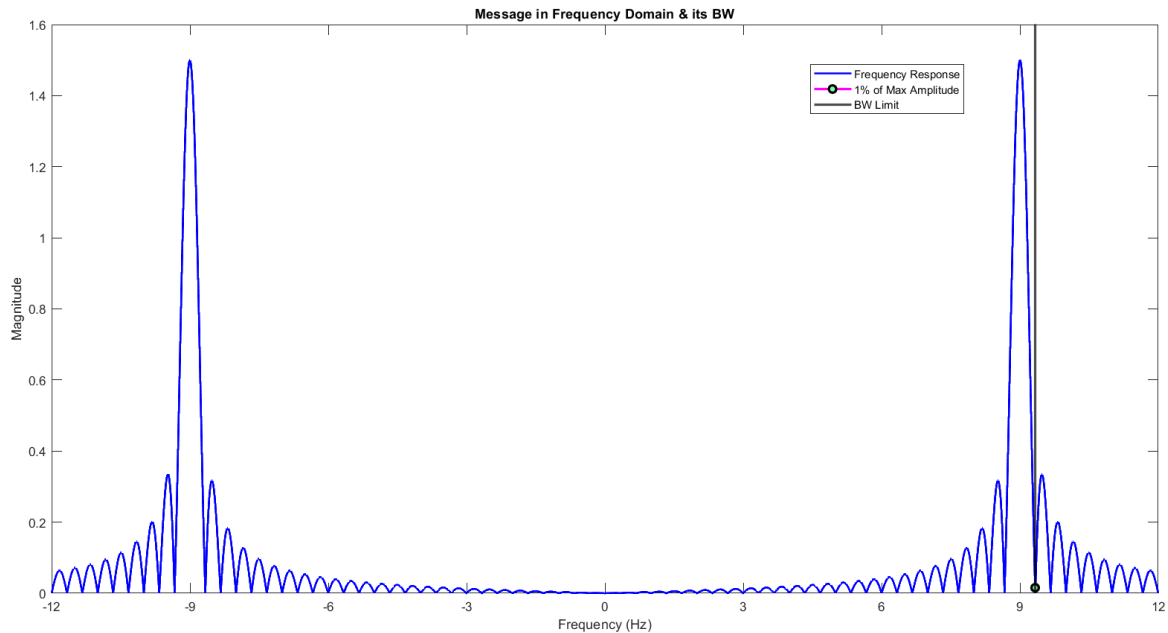
$$\therefore M(f) = \frac{jf}{2\pi(f-9)(f+9)} [e^{-j2\pi*3f} - 1]$$

3. Fourier Transform Using MATLAB



The plot shows that the analytical expression derived above perfectly matches the sampling result obtained using MATLAB.

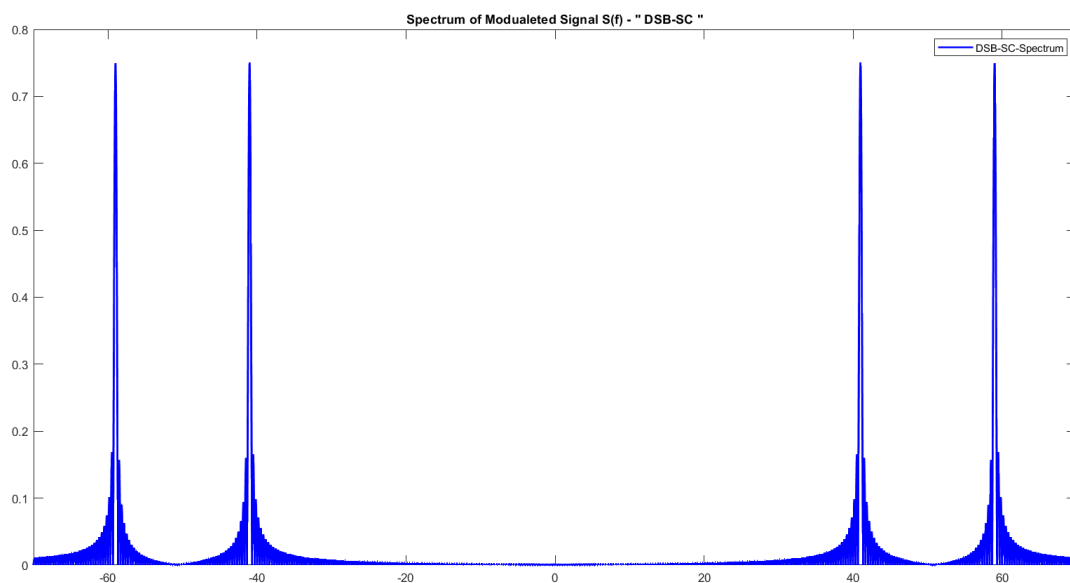
4. Bandwidth Calculation



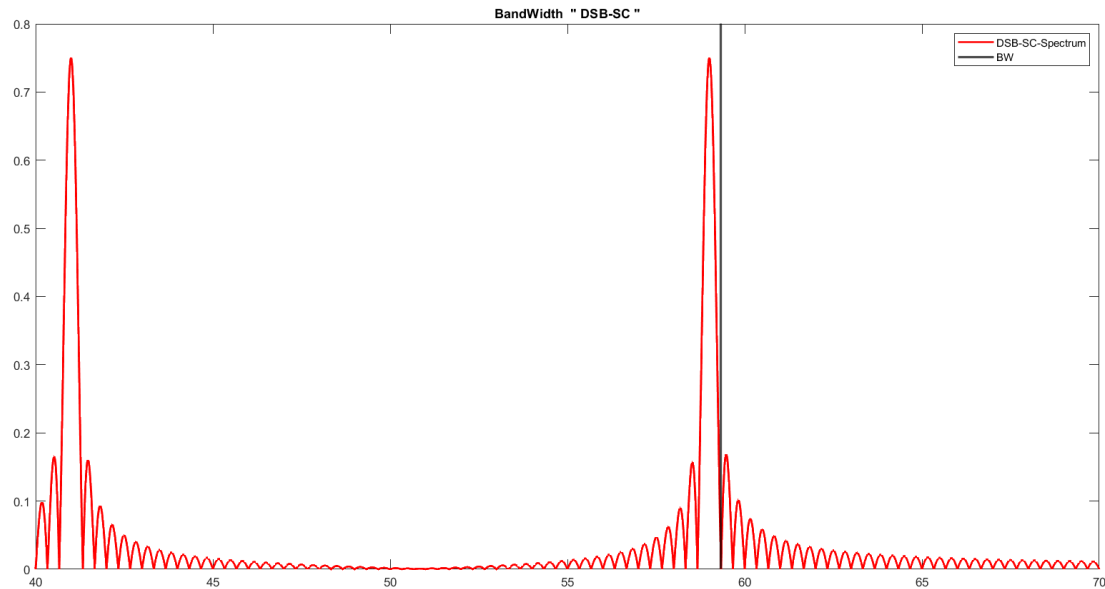
The vertical black line shows the bandwidth limit at 9.33Hz, at which the amplitude falls to 1% of its maximum.

5. DSB-SC Modulation Scheme

a) Frequency response and BW calculation

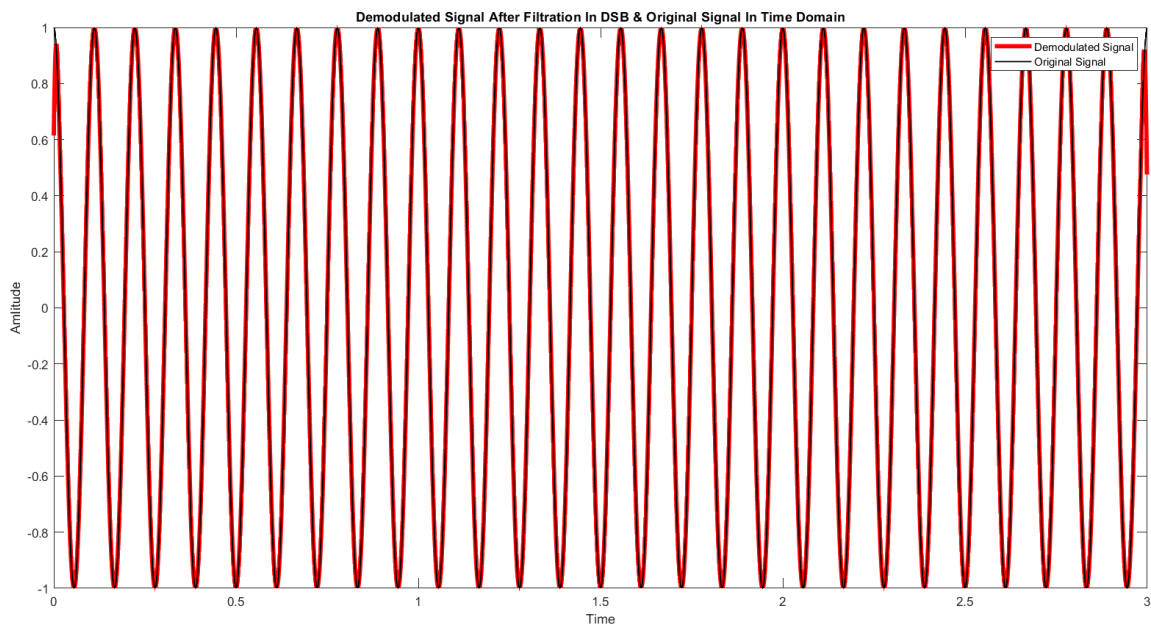


The plot shows the frequency response of the DSB-SC signal.



This plot is zoomed on the positive side of the frequency spectrum, to have a better view of the frequency components of the signal. The vertical black line shows the bandwidth limit at 59.33Hz, at which the amplitude falls to 1% of its maximum.

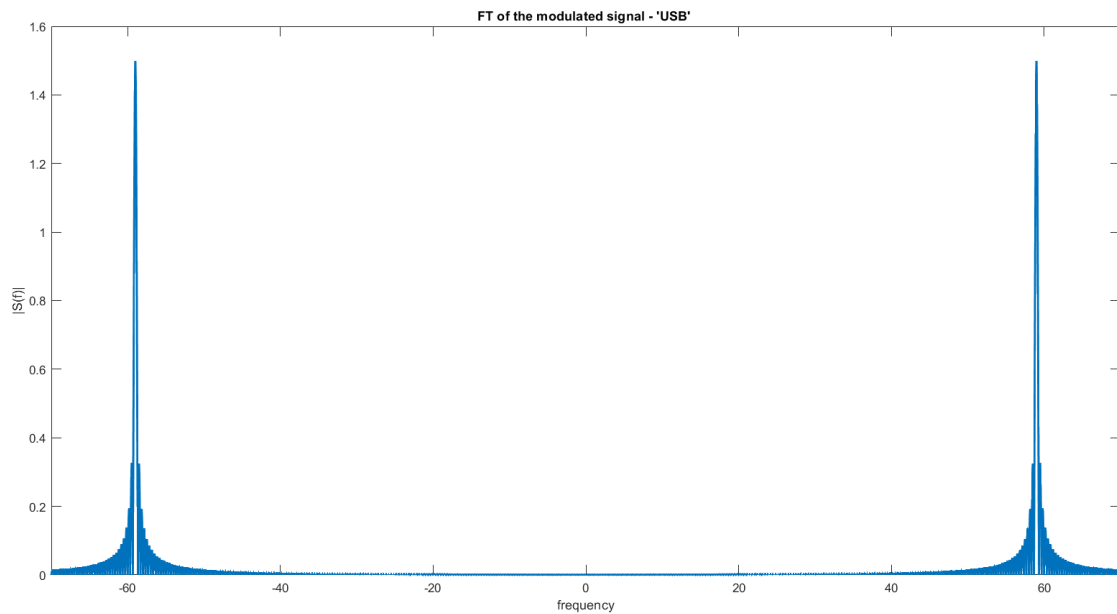
b) Demodulator output VS original signal



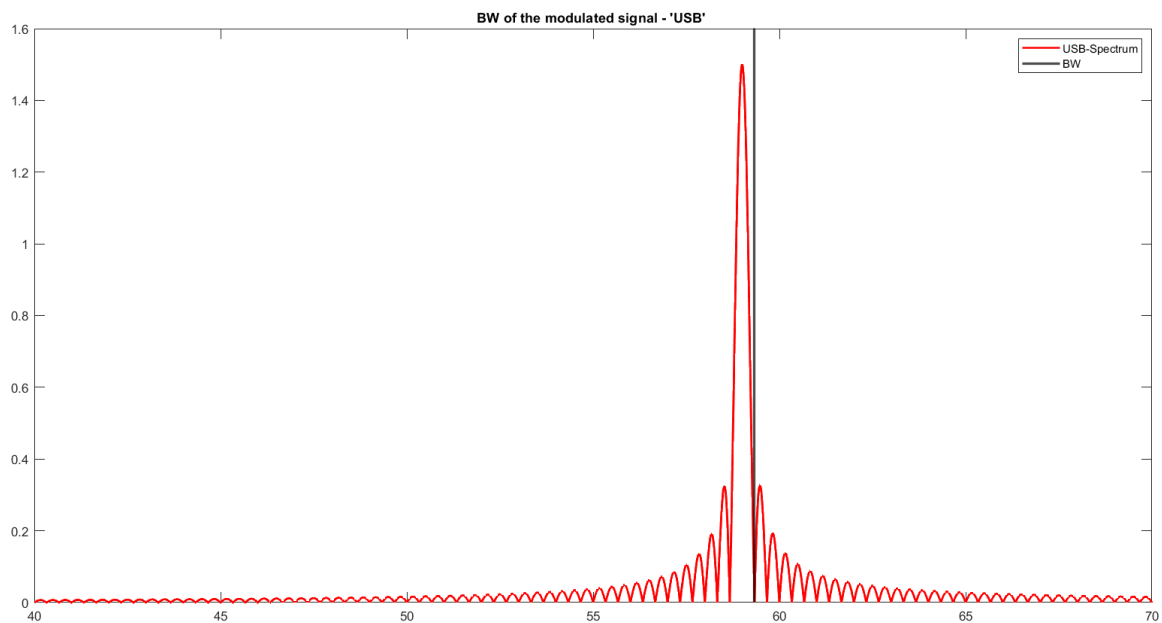
We can see that the demodulator output is almost identical to the original signal. Please note that we multiplied the amplitude of the detector's output by 2 to make up for the attenuation resulting from the multiplication by the cosine function of the detector.

6. SSB-SC Modulation Scheme

a) Frequency response and BW calculation

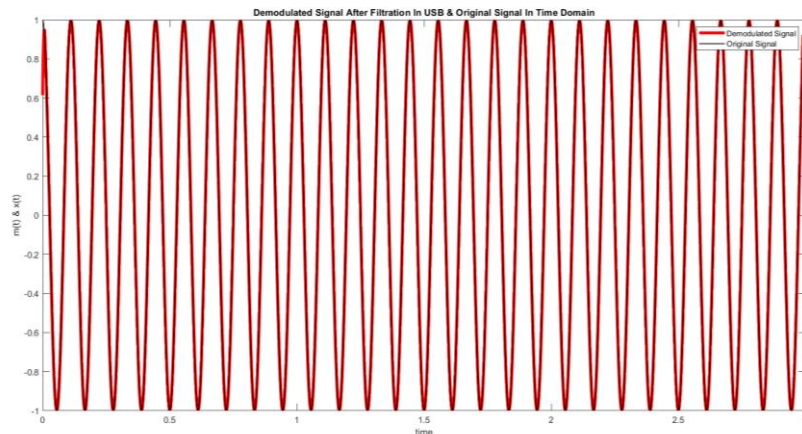


The plot shows the frequency response of the SSB-SC signal.



This plot is zoomed on the positive side of the frequency spectrum, to have a better view of the frequency components of the signal. The vertical black line shows the bandwidth limit at 59.33Hz, at which the amplitude falls to 1% of its maximum.

b) Demodulator output VS original signal



We can see that the demodulator output is almost identical to the original signal. Please note that we multiplied the amplitude of the detector's output by 2 to make up for the attenuation resulting from the multiplication by the cosine function of the detector.

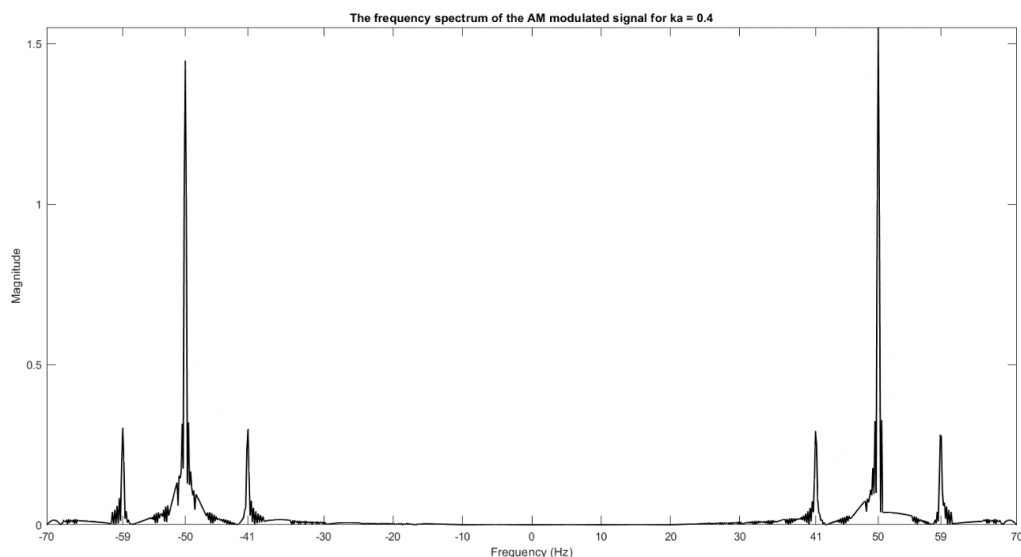
7. Conventional AM modulation:

a) Frequency response and BW calculation

Having $c(t) = \cos(2\pi * 50 * t)$ and $k_a = 0.4$, the modulated signal $s(t)$ is represented as follows.

$$s(t) = [1 + 0.4 \cos(2\pi * 9 * t)] \cos(2\pi * 50 * t)$$

The following figure shows the frequency spectrum of the AM modulated signal.



b) Calculation of Time Constants and Plotting of Signals in Time Domain

To calculate the discharging time constant τ_d for the envelope detector, we have two criteria that have to be taken into account. The first is that the discharging must be much slower than

the period of the carrier wave, so that the detector's output doesn't drop way below the desired value. This can be represented by the following equation.

$$\tau_d = R_L C \gg \frac{1}{f_c}$$

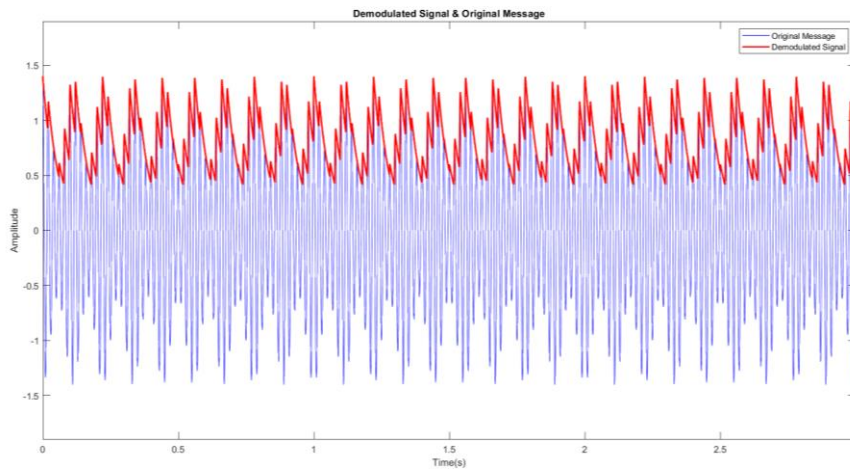
The second is that the discharging on the other hand must be much faster than the message itself, in order to keep tracking the desired message value instead of being higher than it. The following equation represents this condition.

$$\tau_d = R_L C \ll \frac{1}{f_m}$$

By experimenting with different values for τ_d in the code designed to plot the envelope detector's output, the value that showed the best tracking of the desired signal was skewed toward the lower limit and was calculated as follows.

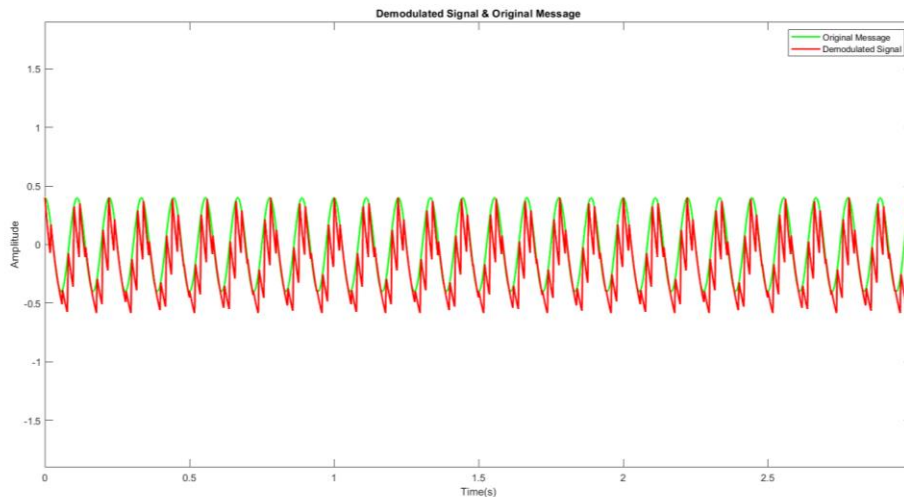
$$\tau_d = \frac{\frac{1}{f_c} + \frac{1}{f_m}}{30} = 4.37 \text{ ms}$$

The following figure shows the plot of the AM signal along with the envelope detector's output in the time domain for $k_a = 0.4$.

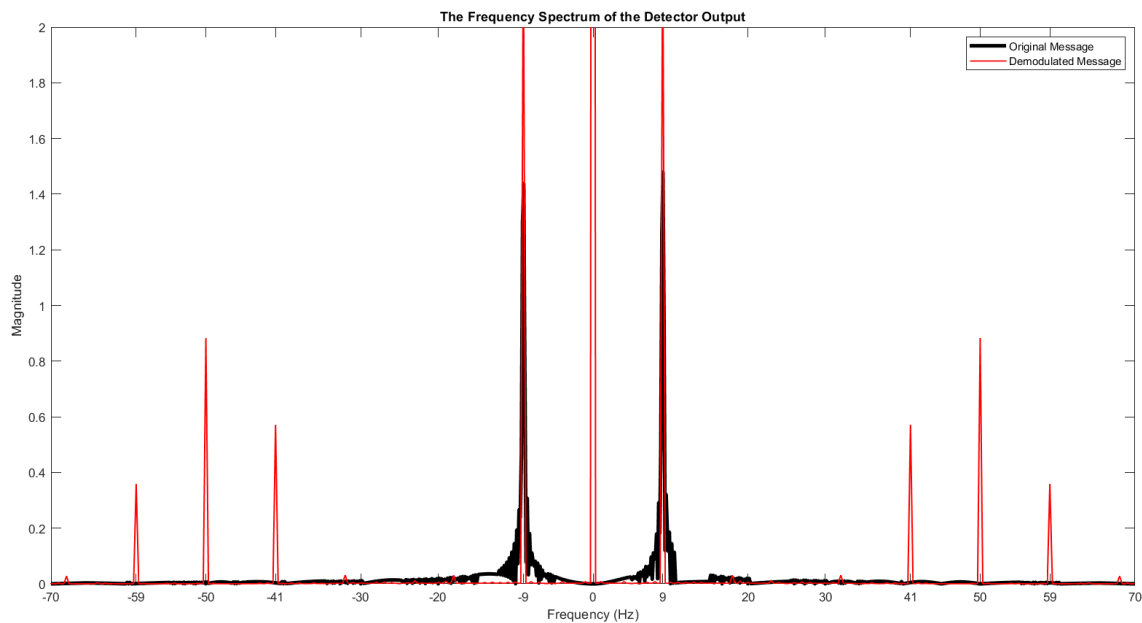


c) Demodulated Signal and the Original Message in Time Domain and in Frequency Domain

The following figure shows the demodulated signal and the original message plotted together in the time domain.



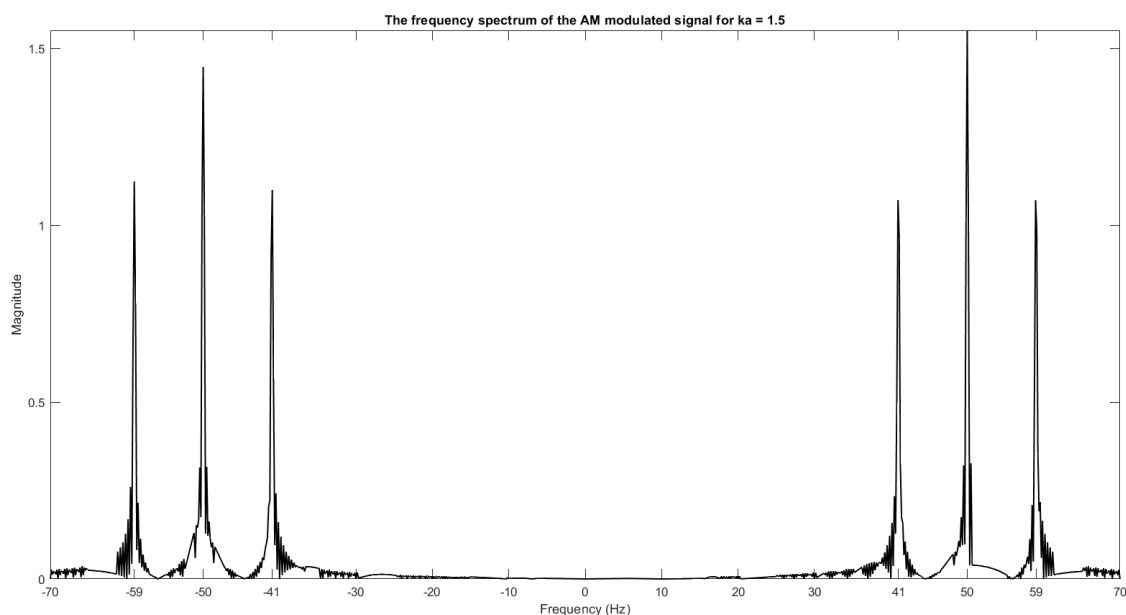
The following figure shows the demodulated signal and the original message plotted together in the frequency domain. Green represents the original message and blue represents the envelope detector's output.



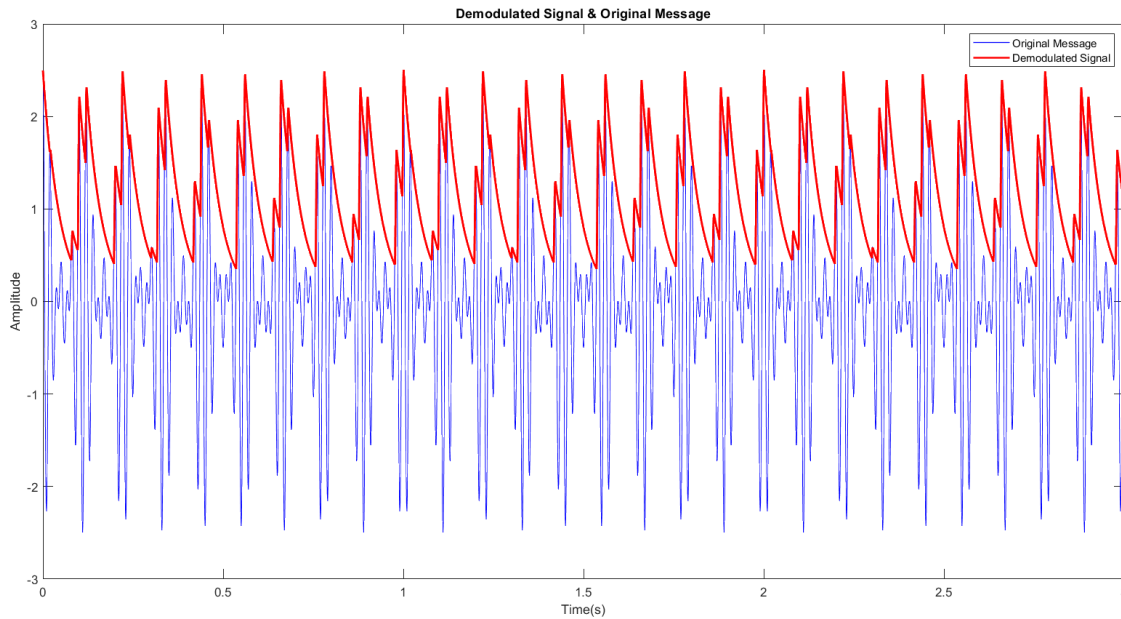
Based on the previous plot, we can see that using a bandpass filter to only pass the frequency components around 9 Hz will enhance the envelope detector's performance and make its output resemble the original message a lot better.

d) Using $k_a = 1.5$

The following figure shows the frequency spectrum of the AM modulated signal given $k_a = 1.5$. It's clear that the plot is completely similar to that obtained for $k_a = 0.4$, this is because the frequency components of the modulated signal aren't affected by the amplitude sensitivity k_a .

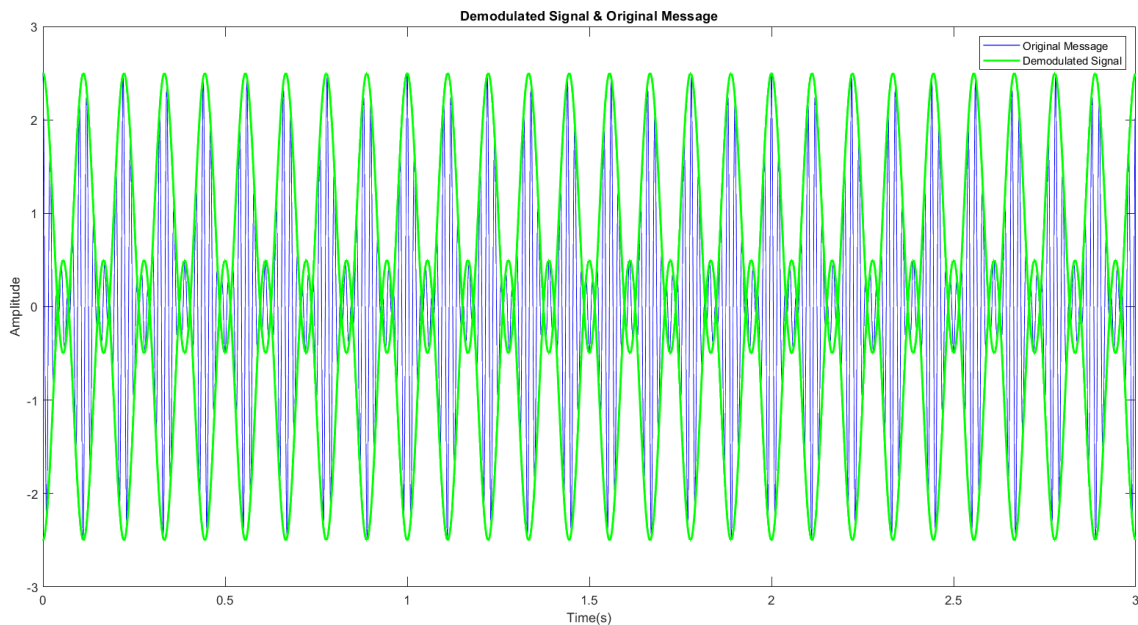


The following figure shows the plot of the AM signal along with the envelope detector's output in the time domain for $k_a = 0.4$.



The previous figure shows that the envelope detector can't keep up with the aggressive changes in the amplitude of the AM modulated signal as it has a high amplitude sensitivity.

The following figure also shows that for $k_a = 1.5$, the ideally expected envelope shows that phase reversal happens, due to the excessively high amplitude sensitivity.



We then conclude that the amplitude sensitivity should have a fairly low value that can be shown by the following equation.

$$1 + k_a m(t) > 0$$



8. MATLAB Code

```
%% Functions

Ts = 1/1000;
t = 0:Ts:3-Ts;
fs = 1/Ts;

x = cos(2*pi*9.*t);

plot(t,x,'linewidth',2)
xlim([-0.5 3.5])
ylim([-1.5 1.5])
title('Message in Time Domain');
xlabel('Time (Seconds)')
ylabel('Magnitude')
legend('Messsage')
f = -12:0.001:12;
FT = abs(3/2*((sinc(3.*(f-9)))+(sinc(3.*(f+9)))));

syms t omega;
x = cos(2*pi*9*t);
y = rectangularPulse(0,3,t);
x = x*y;

z = fourier(x,omega);

syms freq
freq = omega/(2*pi);

%% BW
for i = 1:length(FT)
    if 0.0095 * max(FT) <= FT(i) & FT(i) <= 0.0105 * max(FT) & f(i) > 9 &
f(i) < 9.5
        BW = f(i);
        y = FT(i);
        x = i;
    end
end
```



```
figure(2)
plot(f,FT,'r','linewidth',3.5)
hold on
fplot(freq,abs(z),[-2*pi*12 2*pi*12],'black','linewidth',1)
ylim([0 1.6])
xlim([-12 12])
xticks (-12:3:12)
title('Message in Frequency Domain');
xlabel('Frequency (Hz)')
ylabel('Magnitude')
legend('Analytical FT','FT Using Sampling')

figure(3)
plot(f,FT,'b','linewidth',1.5)
hold on
plot(f(x),FT(x),'-
mo','LineWidth',2,'MarkerEdgeColor','k','MarkerFaceColor',[.49 1
.63],'MarkerSize',6)
xline (BW,'black','linewidth',2);
ylim([0 1.6])
xlim([-12 12])
xticks (-12:3:12)
legend('Frequency Response','1% of Max Amplitude','BW Limit')

title('Message in Frequency Domain & its BW');
xlabel('Frequency (Hz)')
ylabel('Magnitude')

%% Part 7
fm = 9;
fc = 50;
Min_Tao = 1/fm; % Lower limit for tao
Max_Tao = 1/fc; % Higher limit for tao
ka = 0.4;

Tao = (Min_Tao + Max_Tao)/30; % By experimentation, best tao is skewed
towards the lower limit

Tm = 1/10000;
t = 0:Ts:3-Ts;
fs = 1/Ts;
```



```
Message = cos(2*pi*fm*t);
Envelope = 1 + ka * Message;
ct = cos(2*pi*fc*t);
Modulated_Signal = Envelope .* ct;

Detector_Output(1,1)=1+ka;
number_of_points = length(t);

for n = 1 : number_of_points - 1
    if Detector_Output(1,n) < Modulated_Signal(1,n)
        Detector_Output(1,n+1) = Modulated_Signal(1,n);
    else
        Detector_Output(1,n+1) = Detector_Output(1,n)*exp(-Tm/Tao);
    end
end

% The previous code starts by giving the envelope output the same value
% of
% the modulating signal, then it starts looping for the number of points
% that was specified earlier. At each loop, the code checks whether the
% detector output signal is smaller than the modulated signal. If so, the
% detector output charges to the value of the modulated signal, and if
% not,
% it decays exponentially (RC-circuit behaviour) till it falls slightly
% below the modulated signal and the cycle repeats.

%% DSB
v = length(Message)*100;
DSB_SC = Message .* ct;

%%%% DSB_SC Spectrum %%%%

Sf_DSB_SC = (fftshift(fft(DSB_SC,v)));
mag_Sf_DSB_SC = abs(Sf_DSB_SC) / fs ;
f = -fs/2 : fs/v : fs/2-fs/v ;

figure (4);
plot( f , mag_Sf_DSB_SC, 'b', 'Linewidth',1.5 );
title( ' Spectrum of Modulated Signal S(f) - " DSB-SC " ' );
xlim([-70 70]);
```



```
legend('DSB-SC-Spectrum');

st = DSB_SC;

%%%% DSB_SC Spectrum %%%%
Sf_DSB_SC = (fftshift(fft(DSB_SC,v)));
mag_Sf_DSB_SC = abs(Sf_DSB_SC) / fs ;
f = -fs/2 : fs/v : fs/2-fs/v ;
y =0.01*max(mag_Sf_DSB_SC); %%%%line to estimate the band width
figure (5);
plot( f , mag_Sf_DSB_SC, 'r', 'Linewidth',1.5 );
hold on;

for i = 1:length(mag_Sf_DSB_SC)
    if 0.0095 * max(mag_Sf_DSB_SC)<= mag_Sf_DSB_SC(i) & mag_Sf_DSB_SC(i)
<= 0.0105 * max(mag_Sf_DSB_SC) & f(i)>59 & f(i)<59.335
        BW = f(i);
        y = mag_Sf_DSB_SC(i);
        x = i;
    end
end

xline(BW, 'black', 'linewidth',2);
hold off;
title( ' BandWidth  " DSB-SC  "  ');
xlim([40 70]);
legend('DSB-SC-Spectrum','BW');

st_dem = DSB_SC .* ct; % Demodulated Signal
FilteredSignal=lowpass(st_dem,20,fs);
%Time Domain Of Filtered Signal Compared To Original Signal
figure(6);
plot(t,2*FilteredSignal, 'r', 'Linewidth',3.5);
hold on;
plot(t,Message, 'black', 'Linewidth',1);
hold off;
xlabel('Time');
ylabel('Amlitude');
title('Demodulated Signal After Filtration In DSB & Original Signal In
Time Domain');
legend('Demodulated Signal','Original Signal');
```




```
%% SSB
c1 = cos(2*pi*50*t); % carrier
c2 = sin(2*pi*50*t);
v = length(Message)*100;
a = Message.*c1;
b = imag(hilbert(Message)).*c2;
st = a-b; %modulated signal of the
transmitter

%FT of the modulated signal
Sf_USB=fftshift(fft(st,v));
mag_Sf_USB = abs(Sf_USB)/fs ;
f = -fs/2 : fs/v : fs/2-fs/v ;
figure(7);
plot(f,mag_Sf_USB, 'Linewidth',1.5);
xlabel("frequency")
ylabel("|S(f)|")
title("FT of the modulated signal - 'USB' ")
xlim([-70 70]);

for i = 1:length(mag_Sf_USB)
    if 0.0095 * max(mag_Sf_USB)<= mag_Sf_USB(i) & mag_Sf_USB(i) <= 0.0105
        * max(mag_Sf_USB) & f(i)>59 & f(i)<59.335
            BW = f(i);
            y = mag_Sf_DSB_SC(i);
            x = i;
        end
    end

figure(8);
plot(f,mag_Sf_USB,'r', 'Linewidth',1.5);
hold on;
xline(BW,'black','linewidth',2);
title("BW of the modulated signal - 'USB' ")
xlim([40 70]);
legend('USB-Spectrum','BW');

c1 = 2*cos(50*2*pi*t);
z = st.*c1;
%demodulation of s(t)
x = lowpass(z,20,fs);
```



```
figure(9);
plot(t,x,'r','Linewidth',3.5);
hold on ;
plot(t,Message,'black','Linewidth',1);
hold off;

xlabel("time");
ylabel("m(t) & x(t)");
title('Demodulated Signal After Filtration In USB & Original Signal In
Time Domain');
legend('Demodulated Signal','Original Signal');

%% PLOTS
% Modulated Signal
figure (10)
plot(t,Modulated_Signal,'b','linewidth',0.8);
hold on

% Ideally Expected Envelope
plot(t, Envelope,'g','linewidth',1.5)
hold on
plot(t, -Envelope,'g','linewidth',1.5)

% Envelope Detector Output
% plot(t,Detector_Output-1,'r','linewidth',1.5);
% Graph Appearance

ylim([-1/2-(1+ka) 1/2+(1+ka)])
title('Demodulated Signal & Original Message')
legend('Original Message','Demodulated Signal')
xlabel('Time(s)');ylabel('Amplitude');

%% Fourier Transform
syms t freq
freq = omega/(2*pi);

x = cos(2*pi*fm*t);
y = rectangularPulse(0,3,t);
x = x*y;

Envelope = (1 + ka * x);
```



```
ct = cos(2*pi*fc*t);
ct = ct * y;
Modulated_Signal = Envelope * ct;

z = fourier(Modulated_Signal ,omega);

figure(11)
fplot(freq,abs(z),[-2*pi*70 2*pi*70],'black','linewidth',1)
xticks([-70 -59 -50 -41 -30 -20 -10 0 10 20 30 41 50 59 70])
title('The frequency spectrum of the AM modulated signal for ka = 0.4')
xlabel('Frequency (Hz)')
ylabel('Magnitude')

syms t omega;
x = cos(2*pi*9*t);
y = rectangularPulse(0,3,t);
x = x*y;

z = fourier(x,omega);

syms freq
freq = omega/(2*pi);

figure(12)
fplot(freq,abs(z),[-2*pi*70 2*pi*70],'black','linewidth',3)
hold on

z = fft(Detector_Output);
n = length(Detector_Output);
fshift = (-n/2:n/2-1)*(fs/n);
yshift = fftshift(z);
Freq_response = 0.005*abs(yshift);
plot(fshift,Freq_response,'r','linewidth',1)
xlim([-70 70])
ylim([0 2])
xticks([-70 -59 -50 -41 -30 -20 -9 0 9 20 30 41 50 59 70])
title('The Frequency Spectrum of the Detector Output')
xlabel('Frequency (Hz)')
ylabel('Magnitude')
legend('Original Message','Demodulated Message')
```



Digital Communication

Part I:

1. The advantage and disadvantage for each line code

A. Manchester line code:

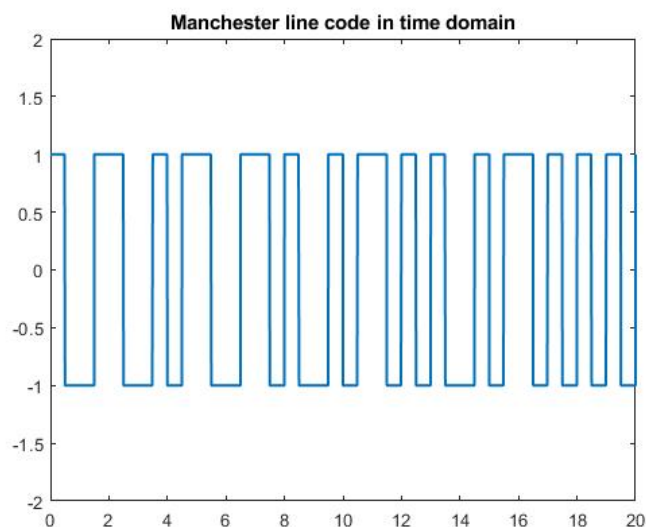
Advantages	<ol style="list-style-type: none">1. Has zero power at zero frequency in its (PSD).2. Transparency of data because the data stream carry information about clock as ones and zeros return to zero volt.
Disadvantages	<ol style="list-style-type: none">1. Has large bandwidth (twice bitrate).2. No error correction.

B. Polar non return to zero-line code:

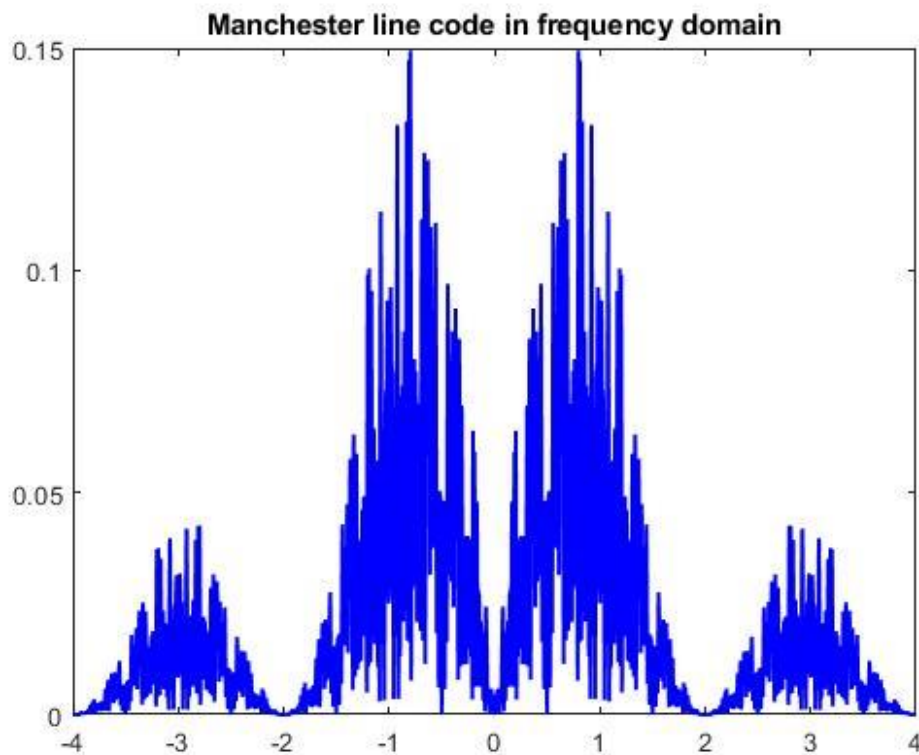
Advantages	Smaller bandwidth as the bandwidth is equal to bitrate.
Disadvantages	<ol style="list-style-type: none">1. Data is not transparent especially in long stream of ones or zeros.2. Has large power at zero frequency.3. No error correction.

2. Manchester line code plots:

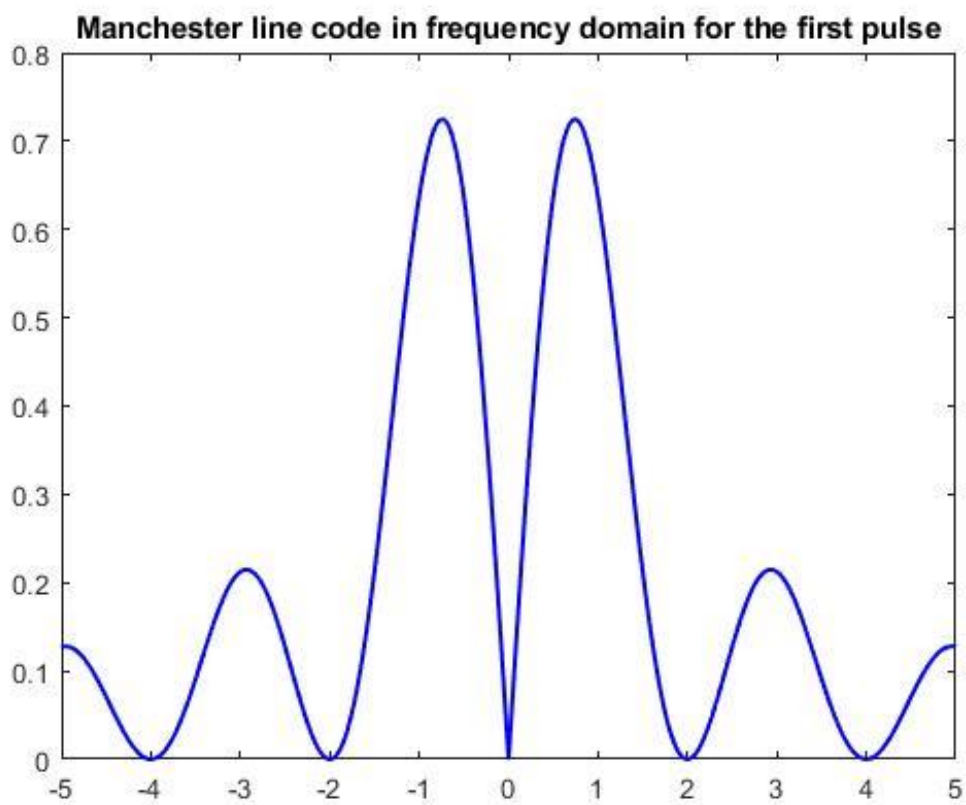
A. Manchester line code in time domain:



B. Manchester line code in frequency domain:

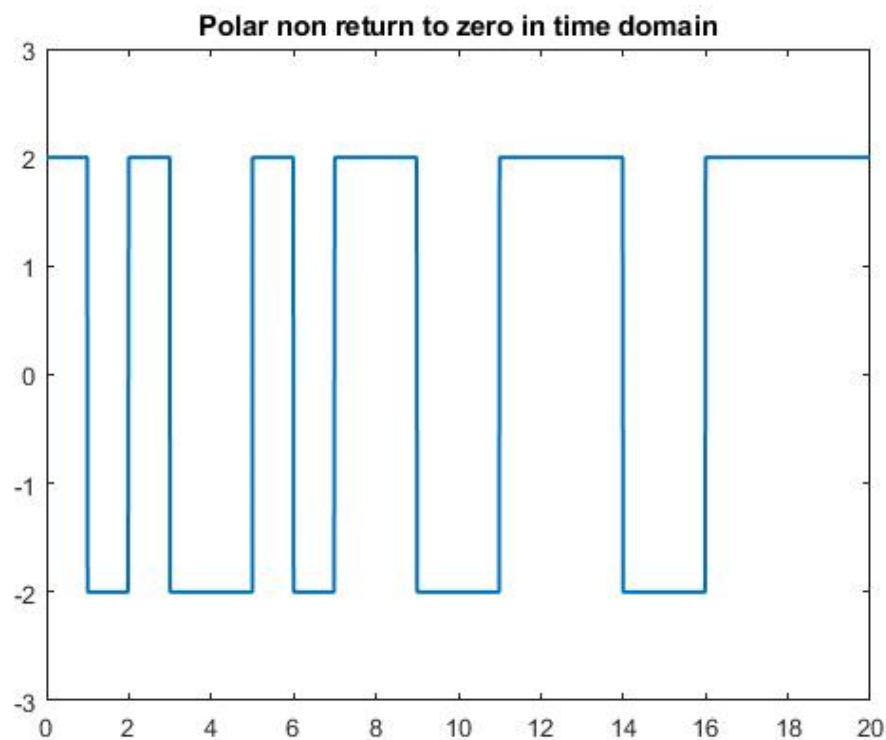


C. Manchester line code in frequency domain for the first pulse only:

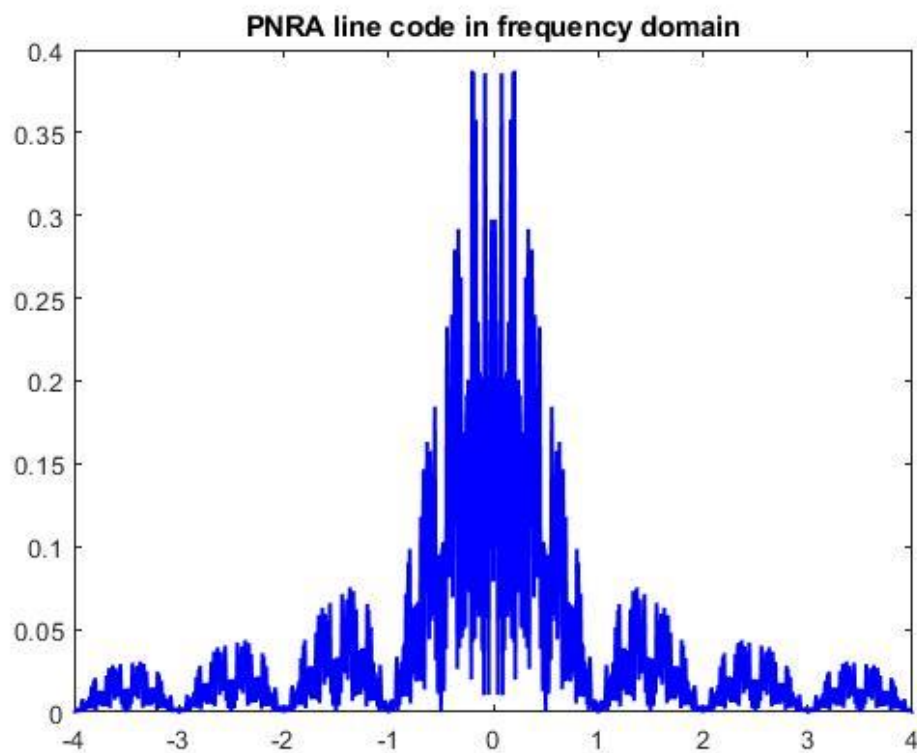


3. Polar non return to zero-line code plots:

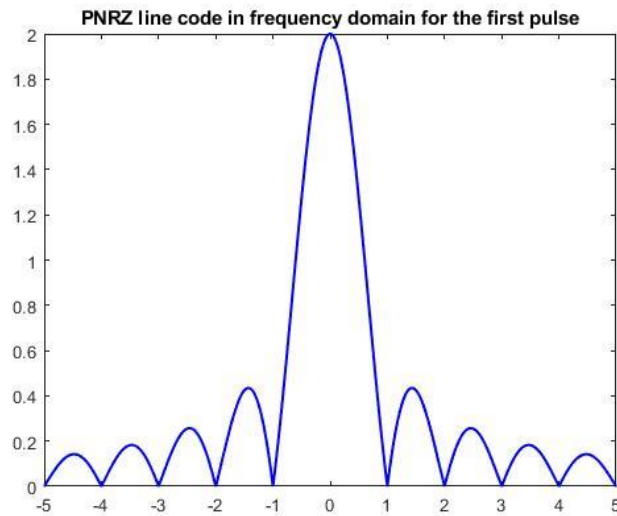
A. PNRZ line code in time domain:



B. PNRZ line code in frequency domain:



C. PNRZ line code in frequency domain for the first pulse only:



4. MATLAB Code:

A. Script:

```
bits = randi([0 1],1,100);
bitrate = 1;
n = 1000;
T = length(bits)/bitrate;
N = n*length(bits);
dt = T/N;
t = 0:dt:T;
ManTime = Manchestertimedomain(bits,t,n);
ManchesterPulsesSpectrum(ManTime,n);
ManchesterFirstPulseSpectrum(ManTime,n);
PNRZ_time = PNRZ_TimeDomain(bits,t,n);
PNRZ_PulsesSpectrum(PNRZ_time,n);
PNRZ_FirstPulseSpectrum(PNRZ_time,n);
```

B. Manchester line code in time domain function:

```
function x = Manchestertimedomain( bits , t , n )
x = zeros(1,length(t));
for i = 0:length(bits)-1
    if bits(i+1) == 1 %amplitude of function x in time domain = 2
        x(i*n+1:(i+0.5)*n) = 1;
        x((i+0.5)*n+1:(i+1)*n) = -1;
    else
        x(i*n+1:(i+0.5)*n) = -1;
        x((i+0.5)*n+1:(i+1)*n) = 1;
    end
end
figure (1);
plot(t, x , 'Linewidth', 1.5);
title('Manchester line code in time domain');
xlim([0 20]);
ylim([-2 2]);

end
```



C. Manchester line code in frequency domain function:

```
function ManchesterPulsesSpectrum(x,n)
v = length(x)*100 ;
xf = (fftshift(fft(x,v)));
mag_xf = abs(xf) / length(x) ;
f = -n/2 : n/v : n/2-n/v ;

figure(2);
plot( f , mag_xf , 'b', 'Linewidth',1.5 );
title('Manchester line code in frequency domain');
xlim([-4 4]);
end
```

D. Manchester line code in frequency domain for the first pulse only:

```
function ManchesterFirstPulseSpectrum(x,n)
k = x(1:1000);
v = length(k)*100;
kf = (fftshift(fft(k,v)));
mag_kf = abs(kf) / length(k) ;
f = -n/2 : n/v : n/2-n/v ;
figure(3);
plot( f , mag_kf , 'b', 'Linewidth',1.5 );
title('Manchester line code in frequency domain for the first pulse');
xlim([-5 5]);
end
```

E. PNRZ line code in time domain:

```
function x = PNRZ_TimeDomain( bits , t , n)
x = zeros(1,length(t));
for i = 0:length(bits)-1
    if bits(i+1) == 1
        x(i*n+1:(i+1)*n) = 2;
    else
        x(i*n+1:(i+1)*n) = -2;
    end
end
figure (4);
plot(t, x, 'Linewidth', 1.5);
title('Polar non return to zero in time domain');
xlim([0 20]);
ylim([-3 3]);
end
```

F. PNRZ line code in frequency domain:

```
function PNRZ_PulsesSpectrum(x,n)
v = length(x)*100 ;
xf = (fftshift(fft(x,v)));
mag_xf = abs(xf) / length(x) ;
f = -n/2 : n/v : n/2-n/v ;
figure(5);
plot( f , mag_xf , 'b', 'Linewidth',1.5 );
title('PNRA line code in frequency domain');
xlim([-4 4]);
end
```




G. PNRZ line code in frequency domain for the first pulse only:

```
function PNRZ_FirstPulseSpectrum(x,n)
k = x(1:1000);
v = length(k)*100;
xf = (fftshift(fft(k,v)));
mag_xf = abs(xf) / length(k) ;
f = -n/2 : n/v : n/2-n/v ;

figure(6);
plot( f , mag_xf , 'b', 'Linewidth',1.5 );
title('PNRZ line code in frequency domain for the first pulse');
xlim([-5 5]);

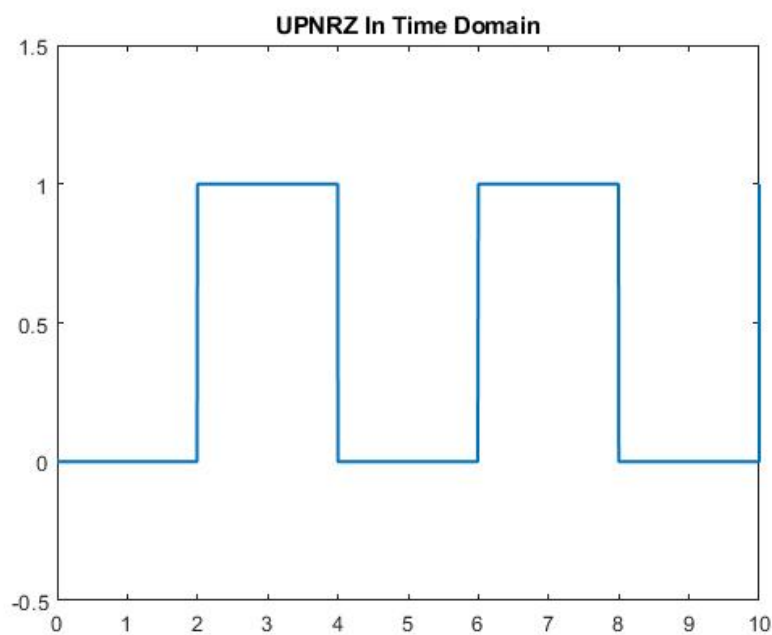
end
```

(BONUS) Part 2: ASK and PSK

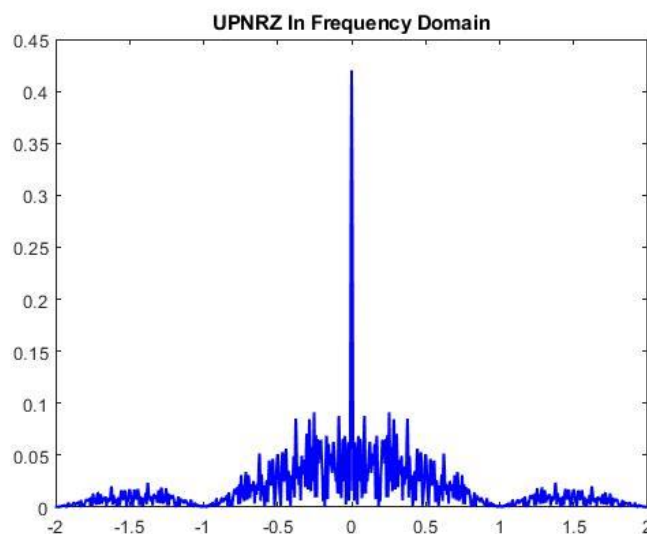
Transmitter and coherent receiver of ASK system

1. ASK MATLAB plots.

A. UPNRZ in time domain:

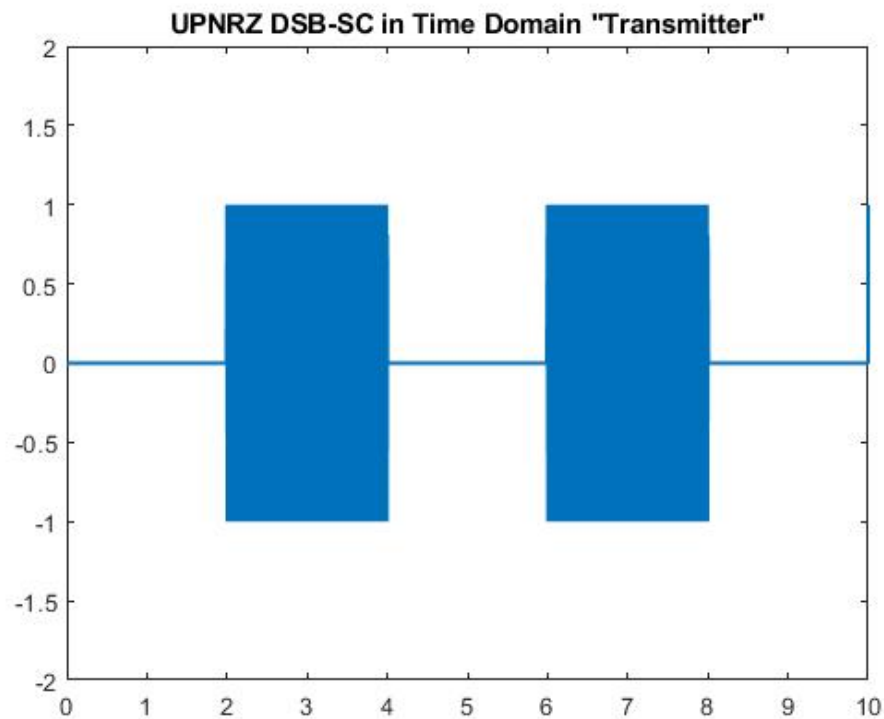


B. UPNRZ in frequency domain:

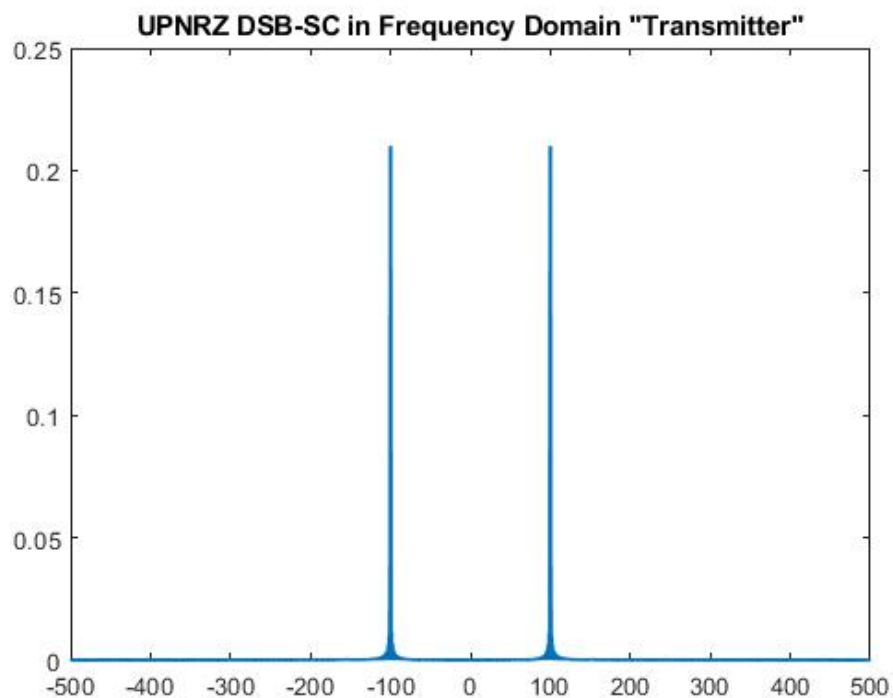


C. UPNRZ in time domain after modulation:

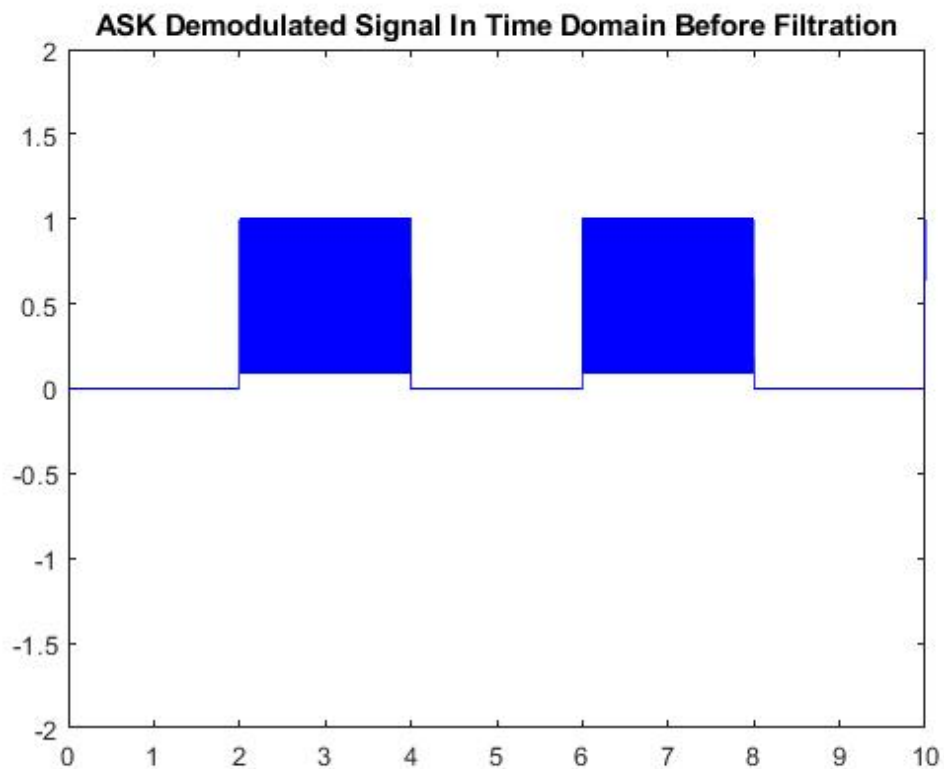
Please note that the used carrier frequency is 100Hz, that's why the sinusoids aren't visible.



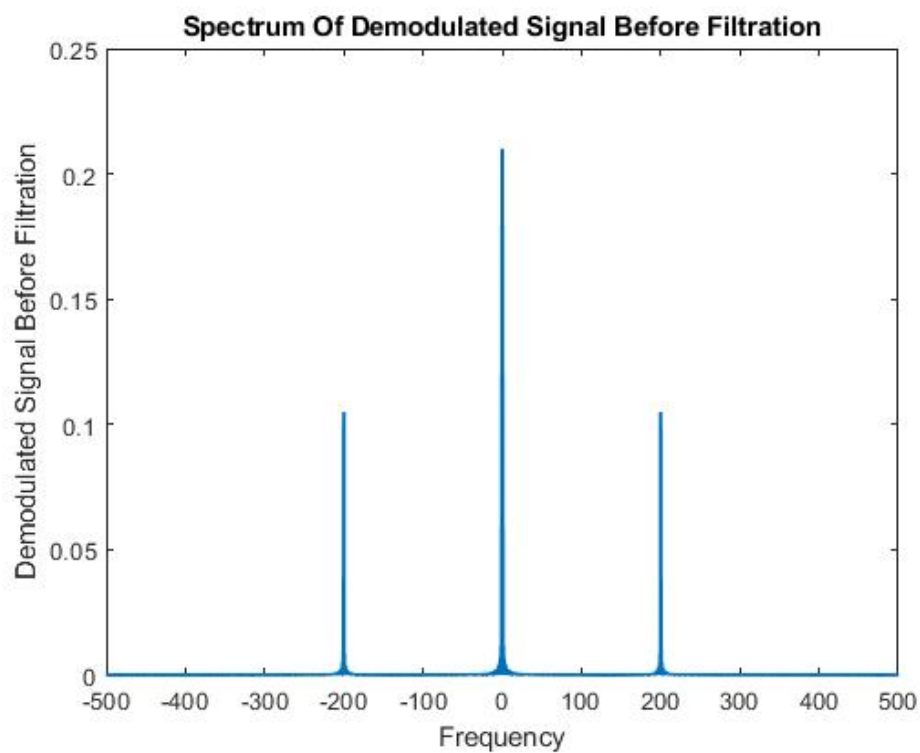
D. UPNRZ in frequency domain after modulation:



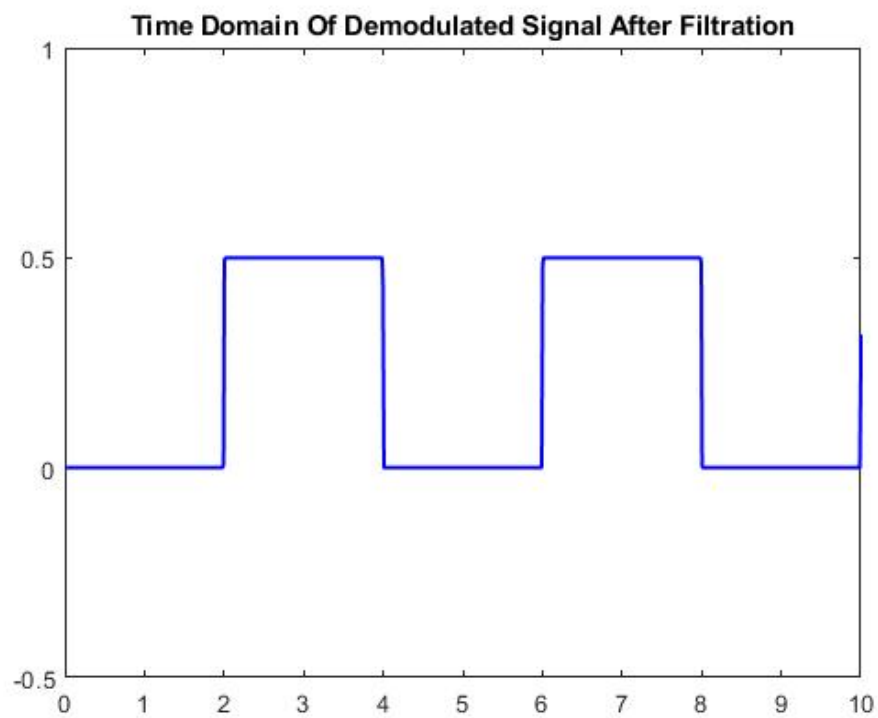
E. UPNRZ in time domain before filtration in the receiver:



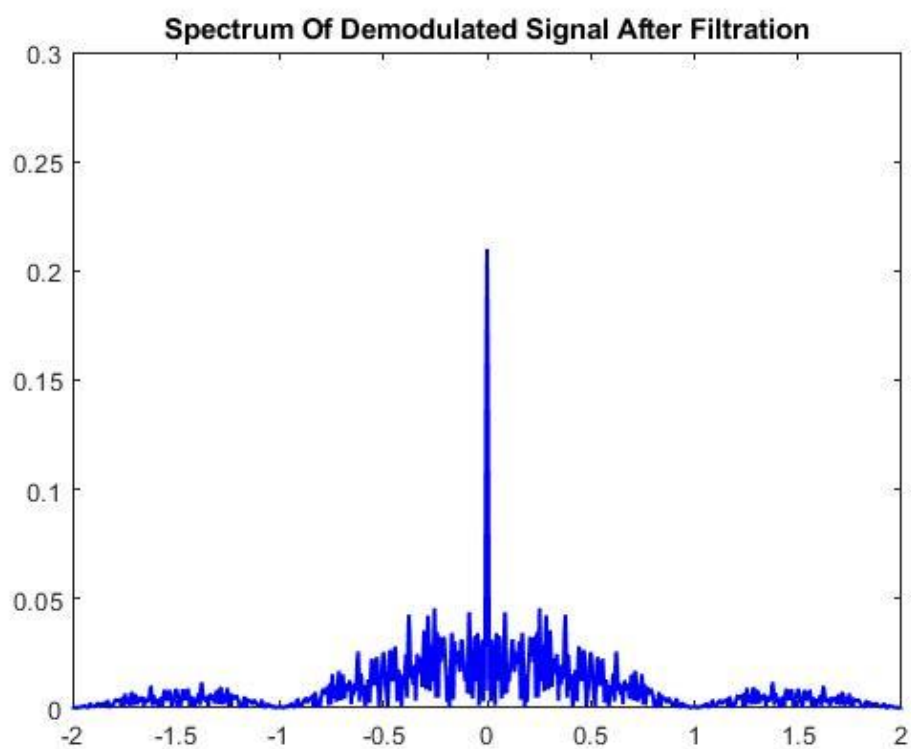
F. UPNRZ in Frequency domain before filtration in the receiver:



G. UPRZ in Time domain after filtration in the receiver:

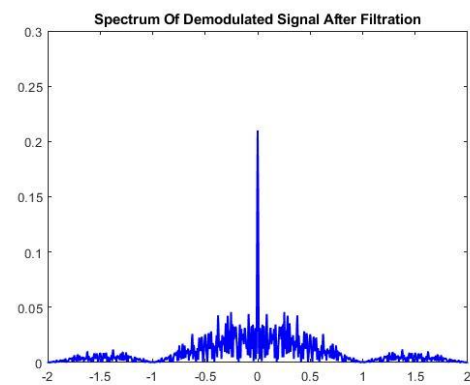
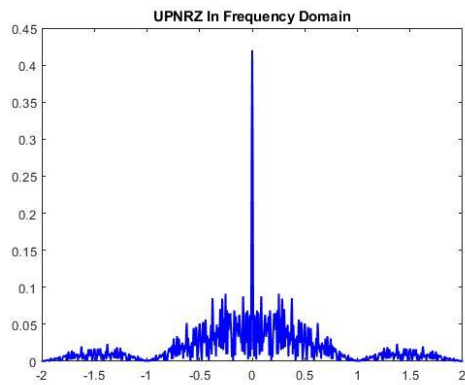
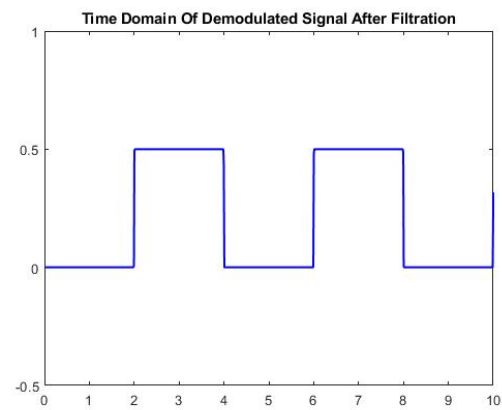
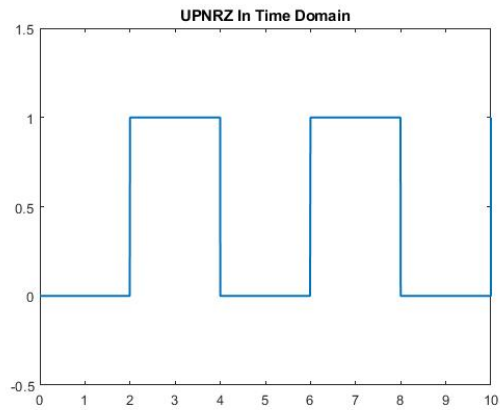


H. UPRZ in Time domain after filtration in the receiver:



2. Comparison between the received signal and the signal at different phases for the coherent detector

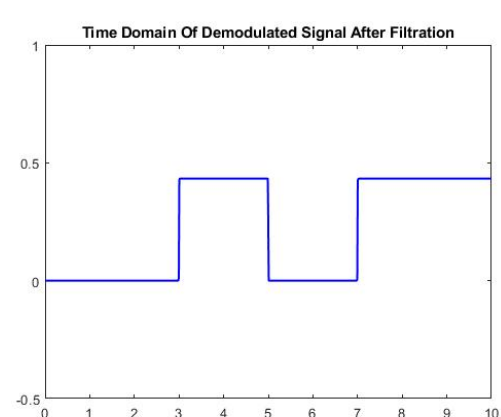
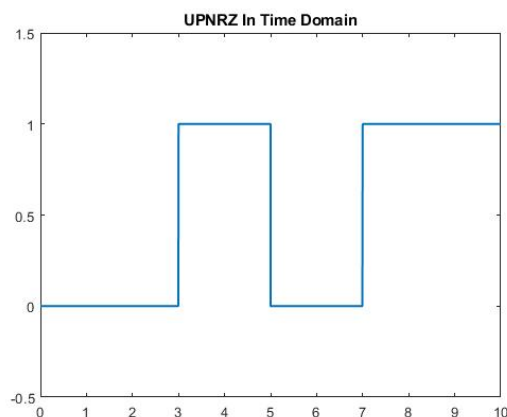
A. At $\phi = 0$

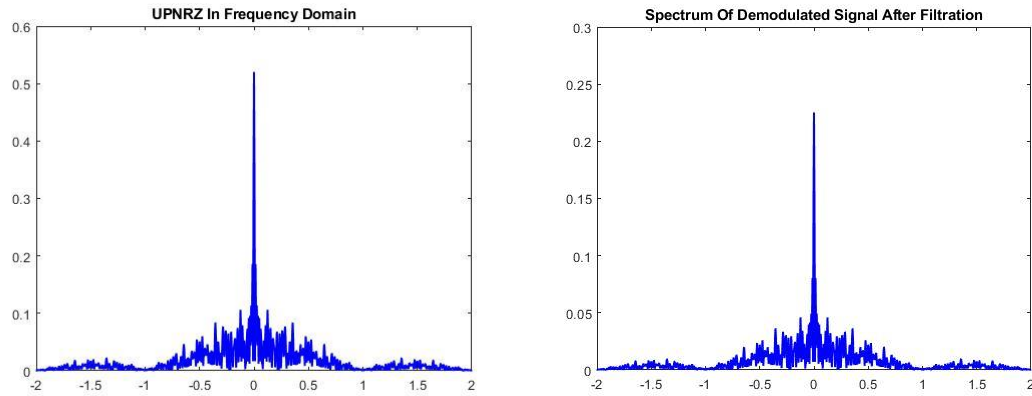


Comment:

The signal will be received successfully but the amplitude will decrease to half the amplitude of the original signal due to the multiplication of the original by two cosines.

B. At $\phi = \frac{\pi}{6}$

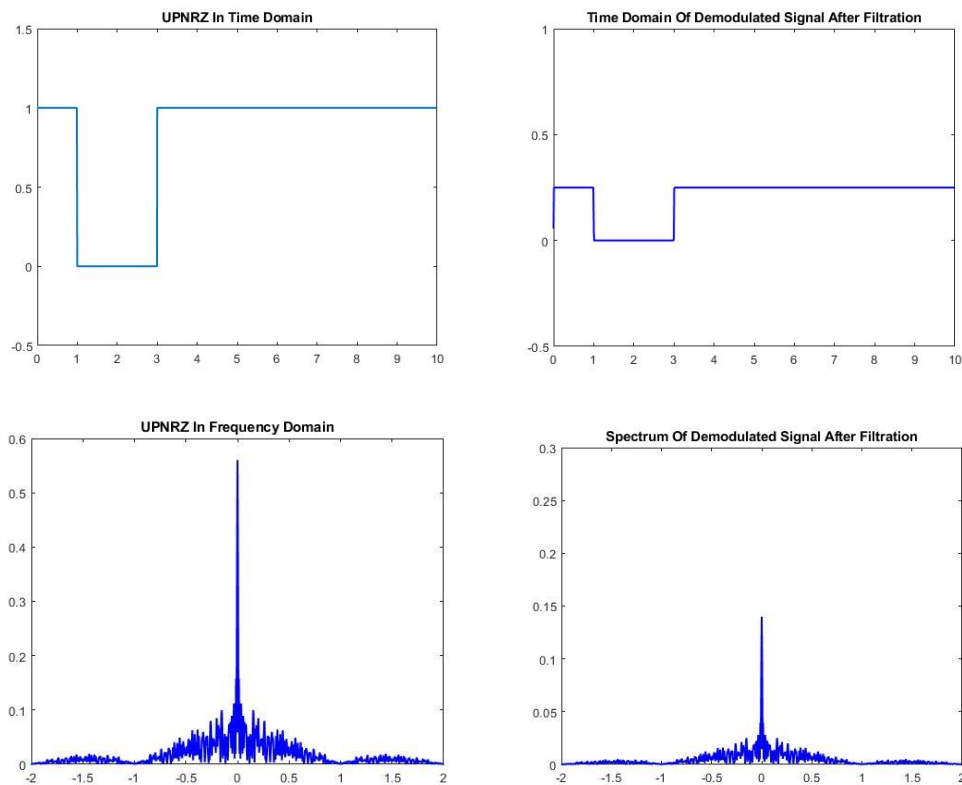




Comment:

In the equation of the received signal there is $\cos(\phi)$, so by changing the phase the amplitude will be attenuated by this factor. In this case the amplitude will decrease by factor $\frac{\sqrt{3}}{2}$ from the signal received at $\phi = 0$.

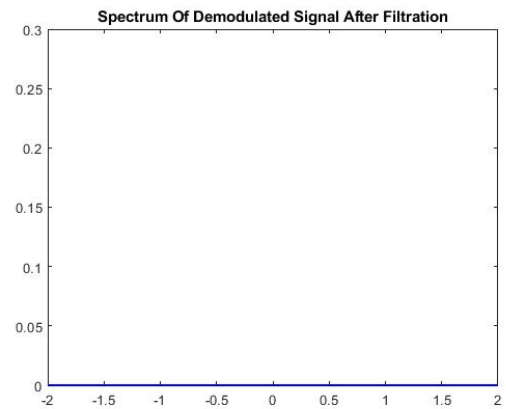
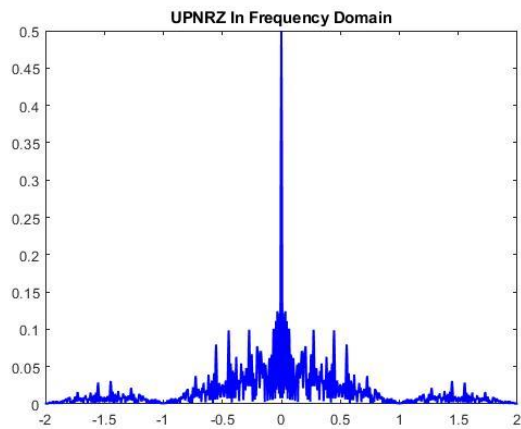
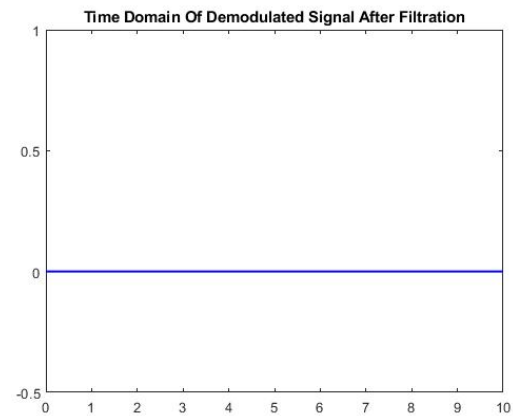
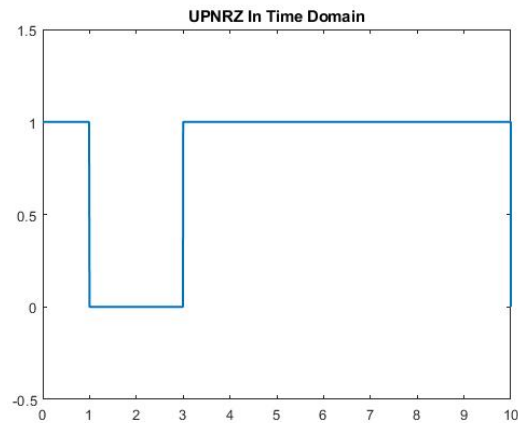
C. At $\phi = \frac{\pi}{3}$



Comment:

In the equation of the received signal there is $\cos(\phi)$, so by changing the phase the amplitude will be attenuated by this factor. In this case the amplitude will decrease by factor $\frac{1}{2}$ from the signal received at $\phi = 0$.

D. At $\phi = \frac{\pi}{2}$



Comment:

In the equation of the received signal there is $\cos(\phi)$, so at $\phi = 90$ the received signal will be equal zero.

Note:

The UPBRZ signal in time domain is different in each case because we use “randi” function in our MATLAB code which generate random binary data in each case.



3. MATLAB Code:

A. Script:

```
bits = randi([0 1],1,100);  
phi = input('please enter the phase shift in the detector ');  
bitrate = 1;  
n = 1000;  
T = length(bits)/bitrate;  
N = n*length(bits);  
dt = T/N;  
t = 0:dt:T;  
UP_Time = UPNRZ_TimeDomain(bits,t,n);  
UPNRZ_Spectrum(UP_Time,n);  
UP_Time_DSB_SC = UPNRZ_ASK_TransmitterModulation(UP_Time,t);  
UPNRZ_ASK_TransmitterSpectrum(UP_Time_DSB_SC,n);  
UP_Time_Dem_NoFilter = ASK_Dem_NoFilter(UP_Time_DSB_SC,t,phi);  
ASK_Dem_NoFilter_Spectrum(UP_Time_Dem_NoFilter,n);  
UP_Time_Dem_AfterFilter =  
ASK_Received_WithFilter(UP_Time_Dem_NoFilter,t,n);  
ASK_Received_WithFilter_Spectrum(UP_Time_Dem_AfterFilter,n);
```

B. UPNRZ in time domain function:

```
function x = UPNRZ_TimeDomain(bits,t,n)  
x = zeros(1,length(t));  
  
for i = 0:length(bits)-1  
    if bits(i+1) == 1  
        x(i*n+1:(i+1)*n) = 1;  
    else  
        x(i*n+1:(i+1)*n) = 0;  
    end  
end  
figure (1);  
plot(t, x , 'Linewidth', 1.5);  
title('UPNRZ In Time Domain');  
xlim([0 10]);  
ylim([-0.5 1.5]);  
end
```

C. UPNRZ in frequency domain function:

```
function UPNRZ_Spectrum(x,n)  
v = length(x)*100 ;  
Mf = (fftshift(fft(x,v)));  
mag_Mf = abs(Mf) / length(x) ;  
f = -n/2 : n/v : n/2-n/v ;  
  
figure(2);  
plot( f , mag_Mf , 'b', 'Linewidth',1.5 );  
title('UPNRZ In Frequency Domain');  
xlim([-100 100]);  
  
end
```



D. UPNRZ in time domain after modulation function:

```
function DSB_SC = UPNRZ_ASK_TransmitterModulation(x,t)

ct = cos(2 * pi * 100 * t );
DSB_SC = x .* ct;
figure (3);
plot( t , DSB_SC, 'Linewidth',1.5);
xlim([0 10]);
ylim([-2 2]);
title('UPNRZ DSB-SC in Time Domain "Transmitter"');

end
```

E. UPNRZ in frequency domain after modulation:

```
function UPNRZ_ASK_TransmitterSpectrum(DSB_SC,n)

v = length(DSB_SC)*100 ;
Sf_DSB_SC = (fftshift(fft(DSB_SC,v)));
mag_Sf_DSB_SC = abs(Sf_DSB_SC) / length(DSB_SC) ;
f = -n/2 : n/v : n/2-n/v ;
figure (4);
plot( f , mag_Sf_DSB_SC, 'Linewidth',1.5 );
title('UPNRZ DSB-SC in Frequency Domain "Transmitter"');

end
```

F. UPNRZ in time domain before filtration in the receiver function:

```
function s_dem = ASK_Dem_NoFilter(x,t,phi)
at = cos(2 * pi * 100 * t + phi );
s_dem= x .* at; % Demodulated Signal
figure(5);
plot(t,s_dem,'b');
xlim([0 10]);
title('ASK Demodulated Signal In Time Domain Before Filtration');
ylim([-2 2]);
end
```

G. UPNRZ in Frequency domain before filtration in the receiver:

```
function ASK_Dem_NoFilter_Spectrum(s_dem,n)
v = length(s_dem)*100 ;
L= length(s_dem);
f= -n/2:n/v:(n/2)-(n/v);
X = fftshift(fft(s_dem,v));
mag = abs(X);
figure(6);
plot(f,mag/L, 'Linewidth',1.5);
xlabel('Frequency');
ylabel('Demodulated Signal Before Filtration');
title('Spectrum Of Demodulated Signal Before Filtration');
end
```



H. UPNRZ in Time domain after filtration in the receiver function:

```
function median = ASK_Received_WithFilter(s_dem,t,n)

FilteredSignal=lowpass(s_dem,50,n);
median = medfilt1(FILTEREDSignal,100);

figure(7);
plot(t,median,'b', 'Linewidth',1.5);
xlim([0 10]);
ylim([-0.5 1]);

title('Time Domain Of Demodulated Signal After Filtration');
end
```

I. UPNRZ in Time domain after filtration in the receiver function:

```
function ASK_Received_WithFilter_Spectrum(FILTEREDSignal,n)

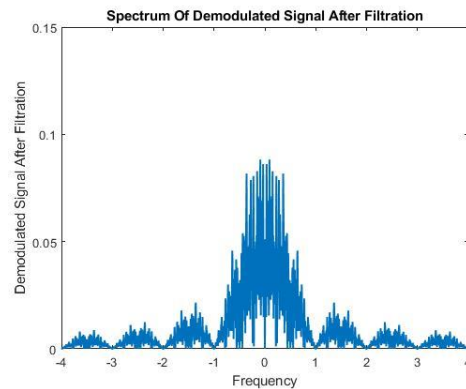
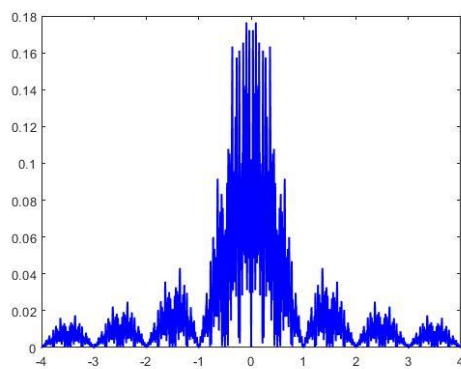
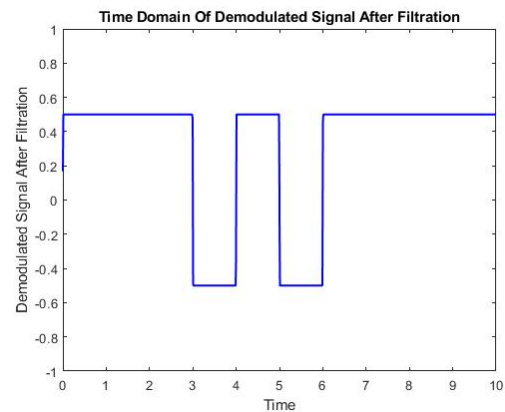
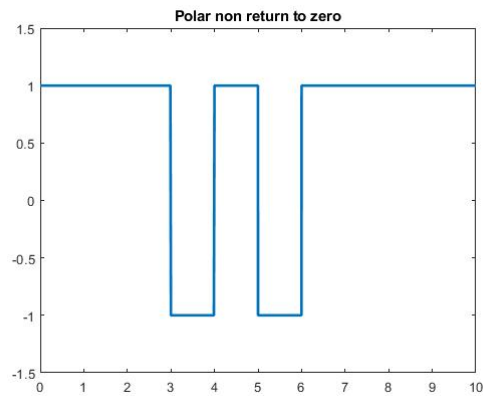
v = length(FILTEREDSignal)*100 ;
L= length(FILTEREDSignal);
f= -n/2:n/v:(n/2)-(n/v);
X = fftshift(fft(FILTEREDSignal,v));
mag = abs(X);
figure(8);
plot(f,mag/L,'b', 'Linewidth',1.5);
title('Spectrum Of Demodulated Signal After Filtration');
xlim([-100 100]);
ylim([0 0.3]);

end
```

Transmitter and coherent receiver of PSK system.

1. Comparison between the received signal and the original signal at different phases for the coherent detector

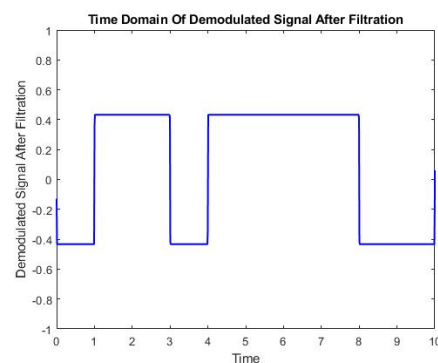
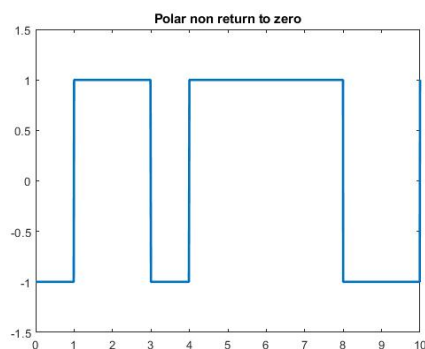
A. At $\phi = 0$

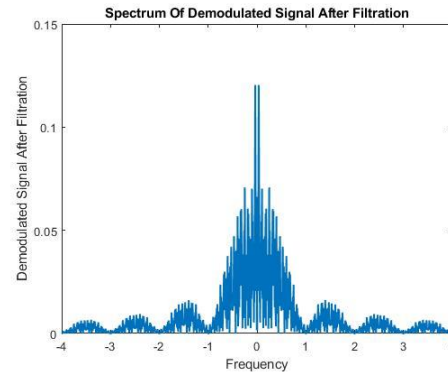
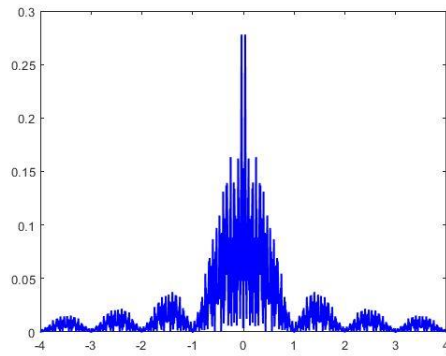


Comment:

The signal will be received successfully but the amplitude will decrease to half the amplitude of the original signal due to the multiplication of the original by two cosines.

B. At $\phi = \frac{\pi}{6}$

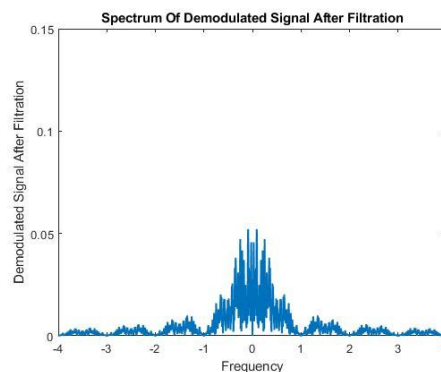
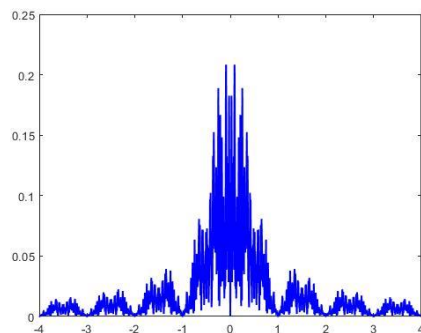
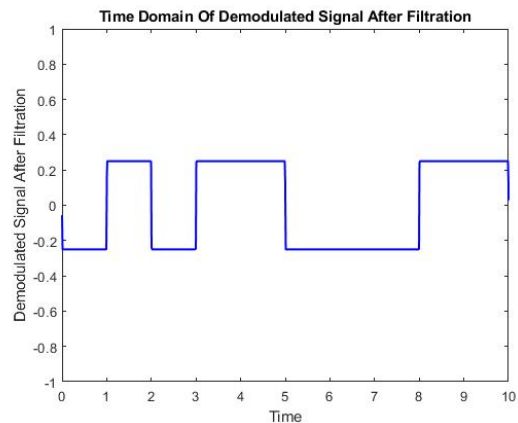
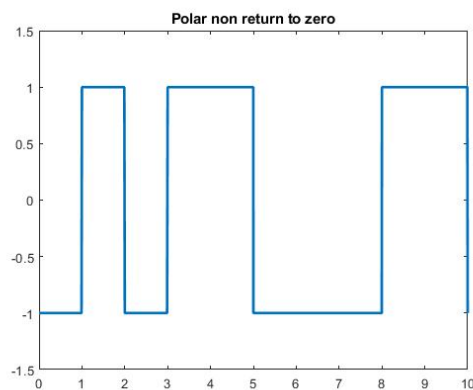




Comment:

In the equation of the received signal there is $\cos(\phi)$, so by changing the phase the amplitude will be attenuated by this factor. In this case the amplitude will decrease by factor $\frac{\sqrt{3}}{2}$ from the signal received at $\phi = 0$.

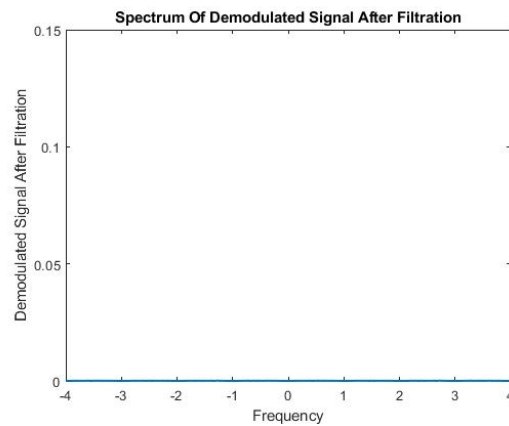
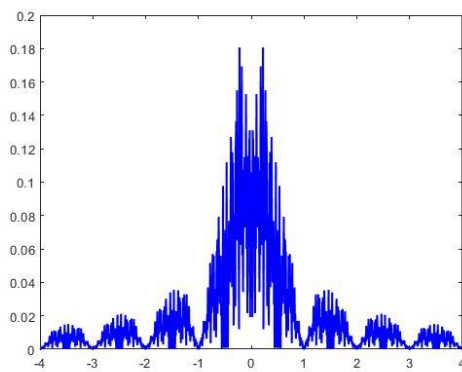
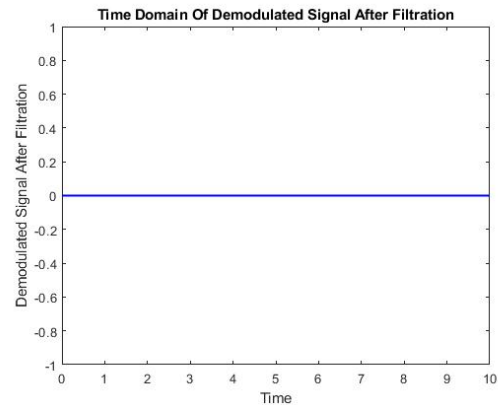
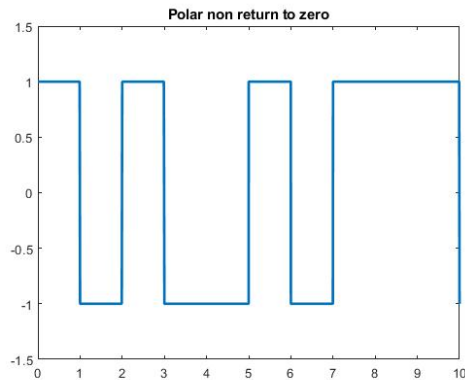
C. At $\phi = \frac{\pi}{3}$



Comment:

In the equation of the received signal there is $\cos(\phi)$, so by changing the phase the amplitude will be attenuated by this factor. In this case the amplitude will decrease by factor $\frac{1}{2}$ from the signal received at $\phi = 0$.

D. At $\phi = \frac{\pi}{2}$



Comment:

In the equation of the received signal there is $\cos(\phi)$, so at $\phi = 90$ the received signal will be equal zero.

Note:

The UPBRZ signal in time domain is different in each case because we use "randi" function in our MATLAB code which generate random binary data in each case.



2. MATLAB Code:

A. Script:

```
bits = randi([0 1],1,100);  
phi = input('please enter the phase shift in the detector ');  
  
bitrate = 1;  
n = 1000;  
T = length(bits)/bitrate;  
N = n*length(bits);  
dt = T/N;  
t = 0:dt:T;  
P_Time= PNRZ_Time(bits,t,n);  
PNRZ_Spectrum(P_Time,n);  
FilteredSignal = PSK_Dem_Time(P_Time,t,n,phi);  
PSK_Dem_Spectrum(P_Time,FilteredSignal,n);
```

B. PNRZ in time domain function:

```
function y= PNRZ_Time(bits,t,n)  
  
y = zeros(1,length(t));  
  
for i = 0:length(bits)-1  
    if bits(i+1) == 1  
        y(i*n+1:(i+1)*n) = 1;  
    else  
        y(i*n+1:(i+1)*n) = -1;  
    end  
end  
  
figure (1);  
plot(t, y, 'Linewidth', 2);  
title('Polar non return to zero');  
xlim([0 10]);  
ylim([-1.5 1.5]);  
end
```

C. PNRZ in frequency domain function:

```
function PNRZ_Spectrum(y,n)  
  
v = length(y)*100 ;  
Mf = (fftshift(fft(y,v)));  
mag_Mf = abs(Mf) / length(y) ;  
f = -n/2 : n/v : n/2-n/v ;  
  
figure(2);  
plot( f , mag_Mf , 'b', 'Linewidth',1.5 );  
xlim([-4 4]);  
  
end
```



D. PNRZ in time domain after PSK modulation and demodulation function:

```
function Filteredsignal = PSK_Dem_Time(y,t,n,phi)
ct = cos(2 * pi * 100 * t );
DSB_SC = y .* ct;
at = cos(2 * pi * 100 * t + phi );
s_dem= DSB_SC .* at; % Demodulated Signal
Filteredsignal=lowpass(s_dem,50,n);
median = medfilt1(Filteredsignal,100);
figure(3);
plot(t,median,'b-', 'Linewidth',1.5);
xlim([0 10]);
ylim([-1 1]);

xlabel('Time');
ylabel('Demodulated Signal After Filtration');
title('Time Domain Of Demodulated Signal After Filtration');

end
```

E. PNRZ in frequency domain after PSK modulation and demodulation function:

```
function PSK_Dem_Spectrum(y,Filteredsignal,n)
v = length(y)*100 ;
L= length(Filteredsignal);
f= -n/2:n/v:(n/2)-(n/v);
X = fftshift(fft(Filteredsignal,v));
mag = abs(X);
figure(4);
plot(f,mag/L,'Linewidth',1.5);
xlabel('Frequency');
ylabel('Demodulated Signal After Filtration');
title('Spectrum Of Demodulated Signal After Filtration');
xlim([-4 4]);
ylim([0 0.15]);

end
```