Project Title:

Simulation and Analysis of Multi-Component Message Signal Processing in a Radar Channel Using MATLAB

Objective: The goal is to build a system that adds signals together and precisely determines signal frequencies using MATLAB. This entails selecting appropriate sampling techniques and simulating signal delays. Understanding the Region of Convergence (ROC) from a plot, assessing the system's output, and adding noise to the signal is also crucial. Lastly, it's critical to identify the most effective signal reconstruction techniques while examining inputs and outputs in the frequency domain.

The following scenario is provided to demonstrate the problem:

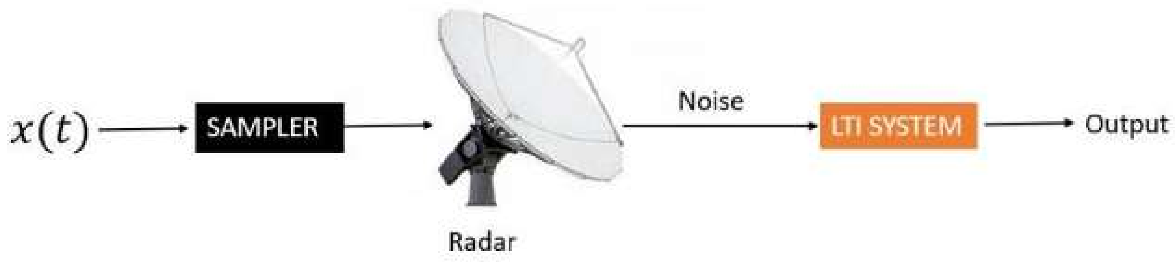


Fig 1: Given Signal Processing Model

From the above-depicted model, we have to get a message signal, x(t), which is the output of a 3-input adder. The inputs of the adder are-

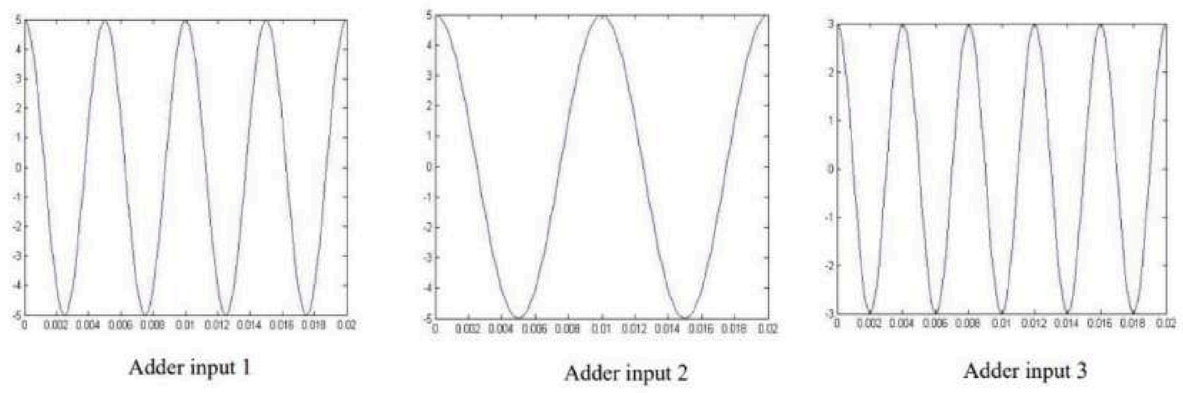


Fig 2: Adder Input Signals, which generates message signal x(t)

An input signal x(t) will be generated by adding these signals. In this project, our main task is to find the signal response after each element in the flow diagram.

Methodology:
The given scenario must be understood before choosing the design methodology. The scenario is a message signal x(t), which is essentially the result of the three additional signals.

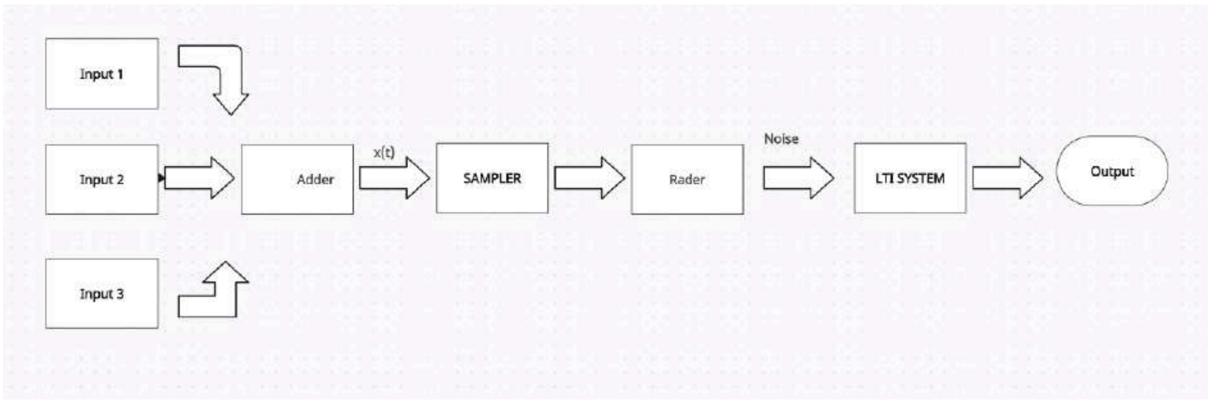


Fig 3: Workflow diagram

The message signal is initially sent to a SAMPLER, after which the Radar receives the signal output from the sampler. During the transmission of the Radar's output, noise is introduced into the signal. The signal then passes through a Linear Time-Invariant (LTI) system, ultimately resulting in the final output signal.

To model the system using the provided signals, we first determine the frequency of each signal and represent it accurately. We then select an appropriate sampling method, ensuring the sampling frequency is correctly chosen. The signal is then processed through a radar system, where it undergoes attenuation and delay. To simulate this, we adjust the signal's amplitude and introduce the appropriate time delay. Finally, we examine the system's characteristics using the given pole-zero plot.

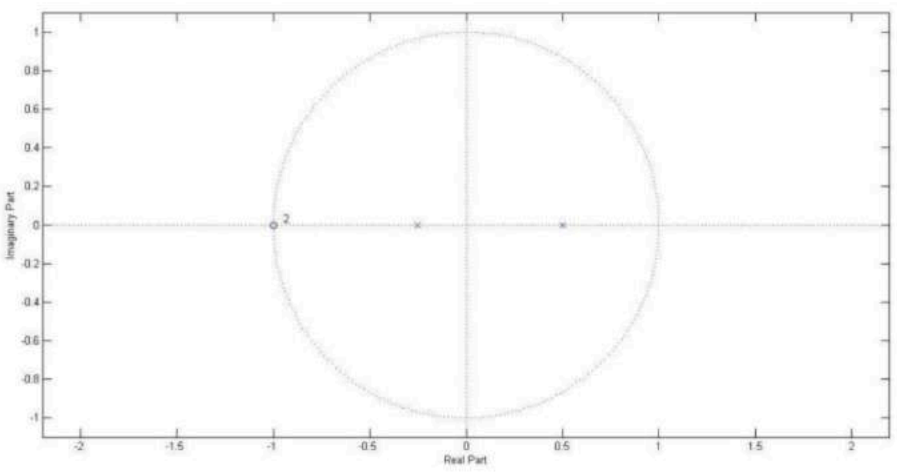


Fig 4: Provided Pole Zero Plot of the LTI System

To ensure the system's stability and predictable behavior, we need to determine its Region of Convergence (ROC) using the available information. By applying the Discrete Fourier Transform (DFT), we can analyze the frequency composition of the signals and identify the system's dominant frequency. Subsequently, we adjust the system's sampling frequency and evaluate the resulting output.

<u>Generating the Input Signals:</u>

The first step in generating the input signals is to determine their respective frequencies. The methodology section offers details about the three input signals, allowing us to derive their specific characteristics.

Adder 1:

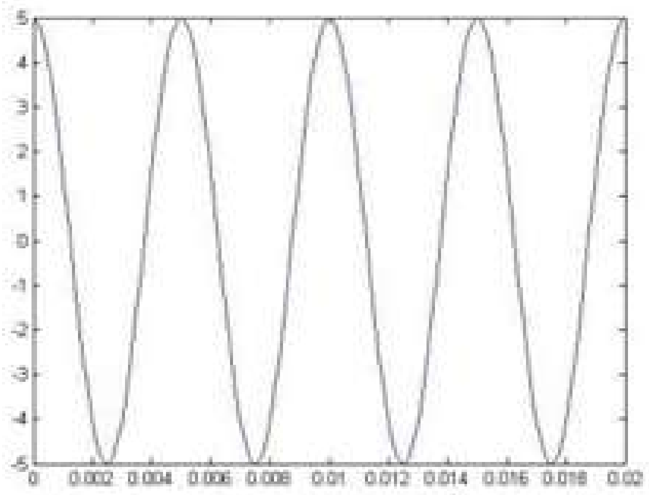


Fig 5: Adder Input 1

Here, the signal's amplitude is 5, and the period is 0.005 seconds.
Thusthefrequencyis,f = $\frac{1}{t} = \frac{1}{0.005}$ = 200Hz

Adder 2:

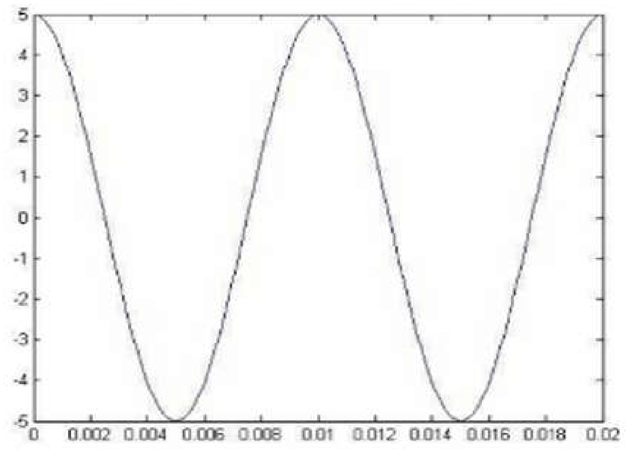


Fig 6: Adder Input 2

Here, the signal's amplitude is 5, and the period is 0.01 seconds.
Thusthefrequencyis,f = $\frac{1}{t} = \frac{1}{0.01}$ = 100Hz

Adder 3:

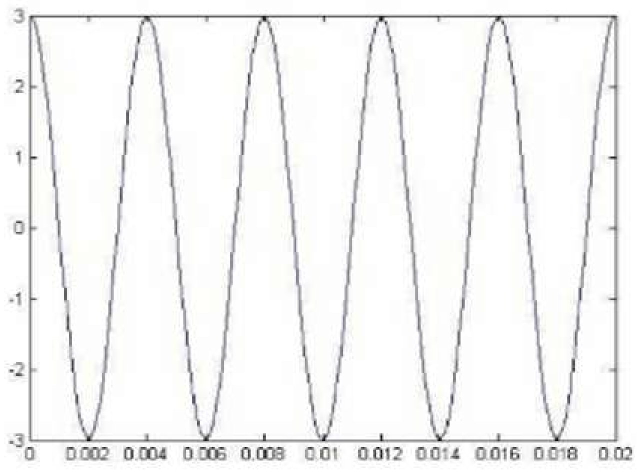


Fig 7: Adder Input 3

Here, the signal's amplitude is 5, and the period is 0.004 seconds.
Thusthefrequencyis,f = $\frac{1}{t}$ = $\frac{1}{0.01}$ = 250Hz

To summarize, the parameters for the adder inputs are as follows:

- **Adder Input 1**: Amplitude = 5, Time Period = 0.005s, Frequency = 200 Hz
- **Adder Input 2**: Amplitude = 5, Time Period = 0.01s, Frequency = 100 Hz
- **Adder Input 3**: Amplitude = 3, Time Period = 0.004s, Frequency = 250 Hz

It is important to note that all three signals are cosine waves as all the signals have started from 90 degrees.

Code:

```
clc;
clear all;
close all;
T = 0.02; % assumed
t = 0:T/100000:T;
f1 = 200;
Add1 = 5*cos(2*pi*f1*t);
f2 = 100;
Add2 = 5*cos(2*pi*f2*t);
f3 = 250;
Add3 = 3*cos(2*pi*f3*t);
xt = Add1+Add2+Add3;
max_a = max(abs(xt));
disp(['Maximum amplitude: ', num2str(max_a)]);
figure;
plot(t, xt);
xlabel("time(s)");
ylabel("Amplitude");
title("Message signal x(t)");
```

Output:

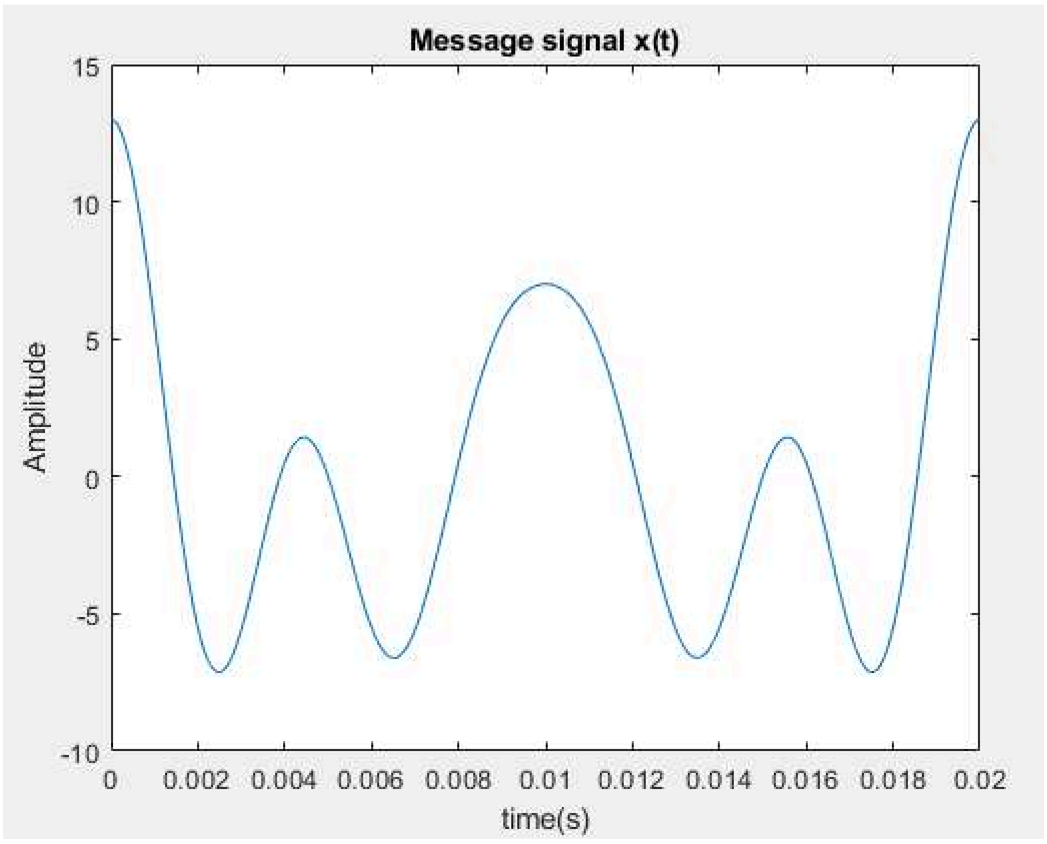
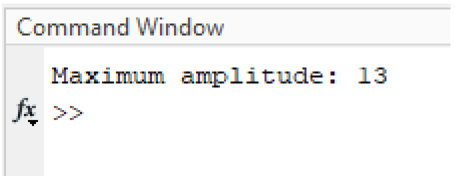


Fig 8: Message signal x(t)

From the command window, we can see the maximum amplitude:



Command Window

```
Maximum amplitude: 13
>>
```

Observation:

The methodology section of the report includes an image depicting the adder's input, which consists of three signals with different frequencies. The output graph illustrates that the highest point of the signal reaches 13, which is justified since the sum of the amplitudes of the three signals equals 13 (5 + 5 + 3). Given that the signal is a combination of different frequencies, the output frequency also varies. Since the input and the sampler share the same frequency representation, only the sampler's output is displayed.

Developing Sampler :

The input signal x(t)x(t)x(t) generated above is a continuous signal that needs to be sampled. According to the Nyquist theorem, the sampling frequency must be at least twice the highest frequency component of the signal. In this case, the maximum frequency is 250 Hz, obtained from Adder Input 3. Therefore, applying the Nyquist theorem:

$$\boxed{\textbf{Nyquist Frequency } (F_N) = 2\times250 = 500\text{ Hz}}$$

Therefore, the sampling frequency selected should be at least 500 Hz or higher. For this scenario, a sampling frequency of 6000 Hz is used throughout the project to ensure an accurate representation of the continuous signal.

Code:

```
clc;
clear all;
close all;
f1=200;
f2=100;
f3=250;
Fs = 6000; % Sampling Frequency
F1 = f1/Fs;
F2 = f2/Fs;
F3 = f3/Fs;
N = 1/F3;
n = 0 : 5*N;
xt_s= 5*cos(2*pi*F1*n) + 5*cos(2*pi*F2*n) + 3*cos(2*pi*F3*n);
stem(n,xt_s);
title("Sampled Signal in Time Domain");
xlabel("Index");
ylabel("Amplitude");
```

Output:

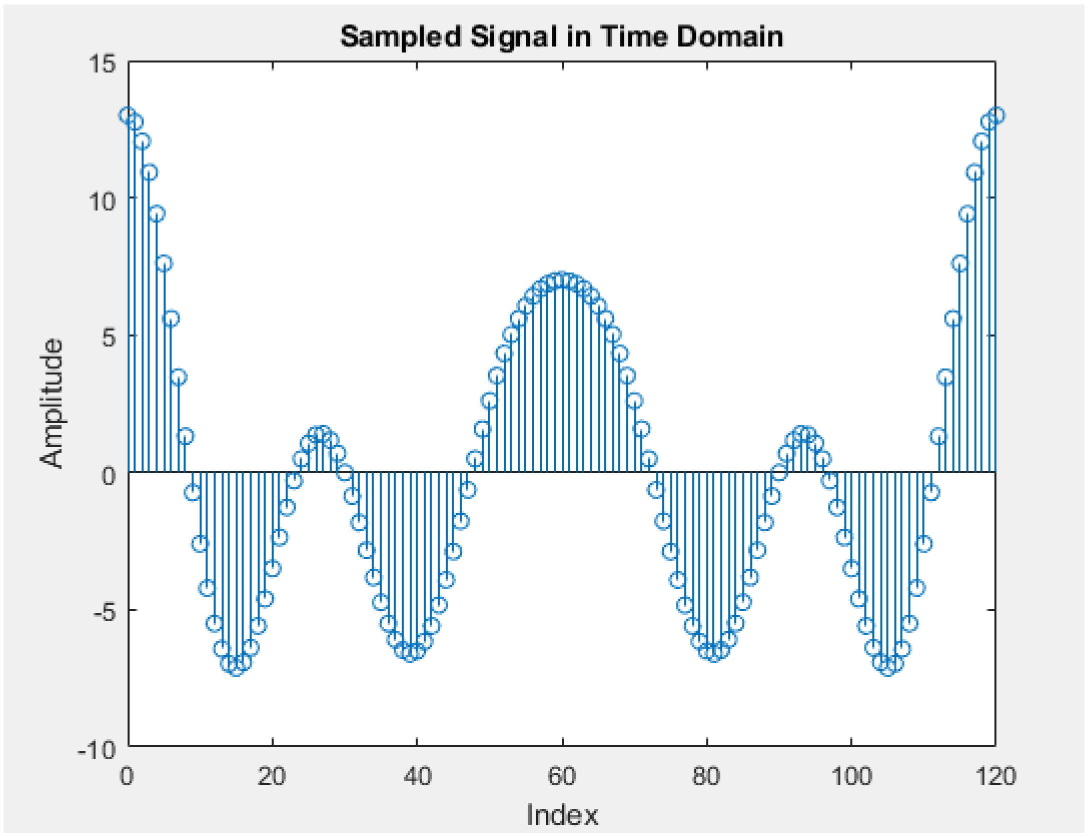


Fig 9: Sampled signal in the time domain

Observation:

By applying the sampling frequency, the continuous signal's frequencies were successfully converted into a discrete form. This confirms that the transformation was effective, as the original continuous signal x(t)x(t)x(t) has been accurately converted into a discrete-time signal.

Frequency Representation of the Sampler's Output

After observing the time domain representation of the Sampler's output, the frequency domain representation is analyzed using the fft() function. The fft() stands for Fast Fourier Transform, which is used to convert the time-domain signal into its frequency-domain representation, allowing us to examine the frequency components of the sampled signal.

Code:

```
clc;
clear all;
close all;
f1=200;
f2=100;
f3=250;
Fs= 6000; % Sampling Frequency
```

```
F1= f1/Fs;
F2= f2/Fs;
F3= f3/Fs;
N = 1/F3; %Time Period for discrete time signal
n = 0 : 5*N; % 3 full cycles
xt_s= 5*cos(2*pi*F1*n) + 5*cos(2*pi*F2*n) + 3*cos(2*pi*F3*n);
xfft = fft(xt_s,length(xt_s)); % fast Fourier transform
id = (Fs/length(xt_s)) * (0:length(xfft)/2-1);
xfft = xfft(1:length(xfft)/2);
stem(id, abs(xfft));
xlabel("Frequency");
ylabel("Amplitude");
title("Fast Fourier Transform(FFT)");
```

Output:

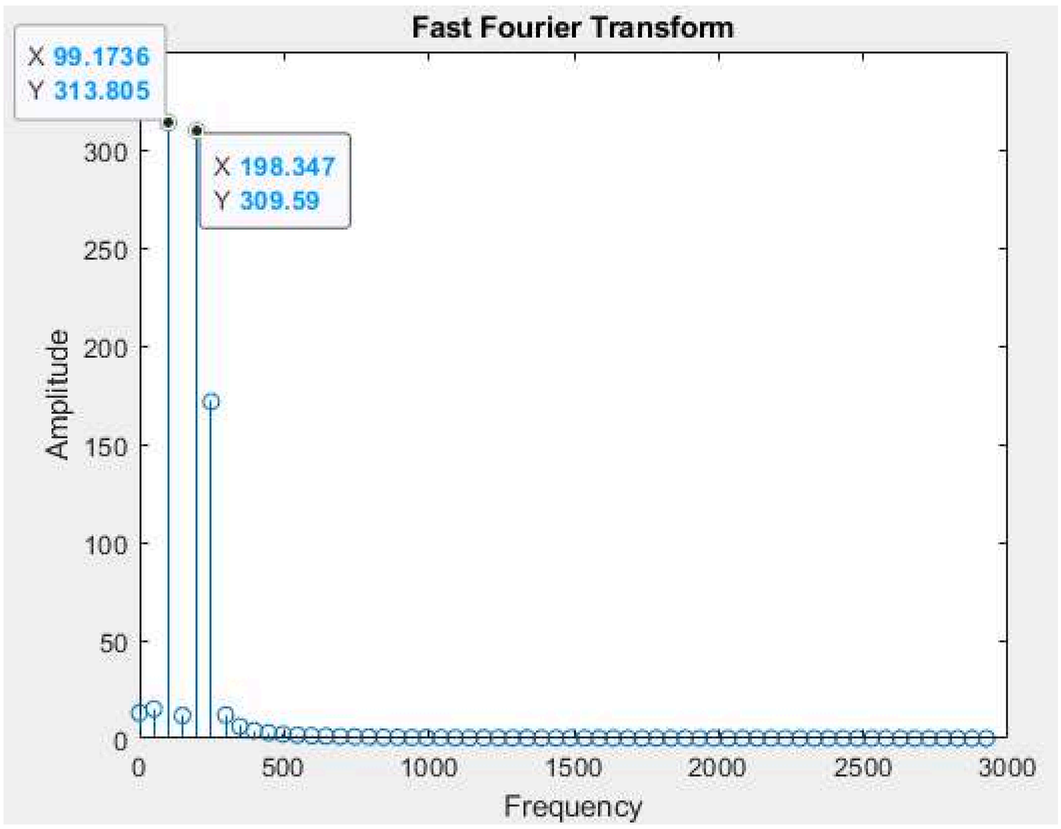


Fig 10: Fast Fourier transformation

Observation:

After generating the frequency domain representation of the Sampler's output, it is observed that the highest amplitudes occur at 99.1736 Hz and 198.347 Hz. This indicates that the majority of the signal's components are concentrated around these frequencies. Notably, there is a clear pattern, as 198.347 Hz is exactly twice 99.1736 Hz. This suggests that at 198.347 Hz, the signal contains a harmonic of 99.1736 Hz.

Radar Output - Attenuation and Delay of the Signal:

Attenuation in a signal refers to the reduction in its magnitude, intensity, or amplitude as it propagates through a medium or encounters obstacles, indicating a weakening of the signal's strength. In this scenario, the attenuation factor is 0.A, and the signal experiences a delay of B elements added to its input. This models the signal's reduced strength and the time delay introduced during its transmission through the radar system.

Shaibal Das : 22121033

Farrdin Nowshad: 21321007

Md. Abu Anas Mridul: 22121024

Likhon Chandra Sarkar: 23321075

Summation of the Student ID: 22121033 + 21321007 + 22121024 + 23321075 = **88884139**

A = Highest number of the sum = 9

B = Lowest number of the sum = 1

Attenuation is achieved by performing element-wise multiplication of the signal with the attenuation factor, while the delay is introduced by shifting the signal accordingly. Keeping these principles in mind, the following code is implemented

Code:

```
clc;
close all;
clear all;
f1=200;
f2=100;
f3=250;
Fs = 6000; % Sampling Frequency
F1 = f1/Fs;
F2 = f2/Fs;
F3 = f3/Fs;
N = 1/F3;
n = 0 : 5*N;
xt_s=5*cos(2*pi*F1*n)+5*cos(2*pi*F2*n)+3*cos(2*pi*F3*n);
subplot(2,1,1);
stem(n,xt_s);
xlabel("Index");
ylabel("Amplitude");
title("Sampled Signal");
at=0.9;
delay=1;
radar= at.*xt_s;
R_n= n + delay; %adding delay to the original cycles
subplot(2, 1, 2);
stem(R_n, radar,'r');
title("Radar Attenuation");
xlabel("Index");
ylabel("Amplitude");
xlim([0, 95]);
```

Output:

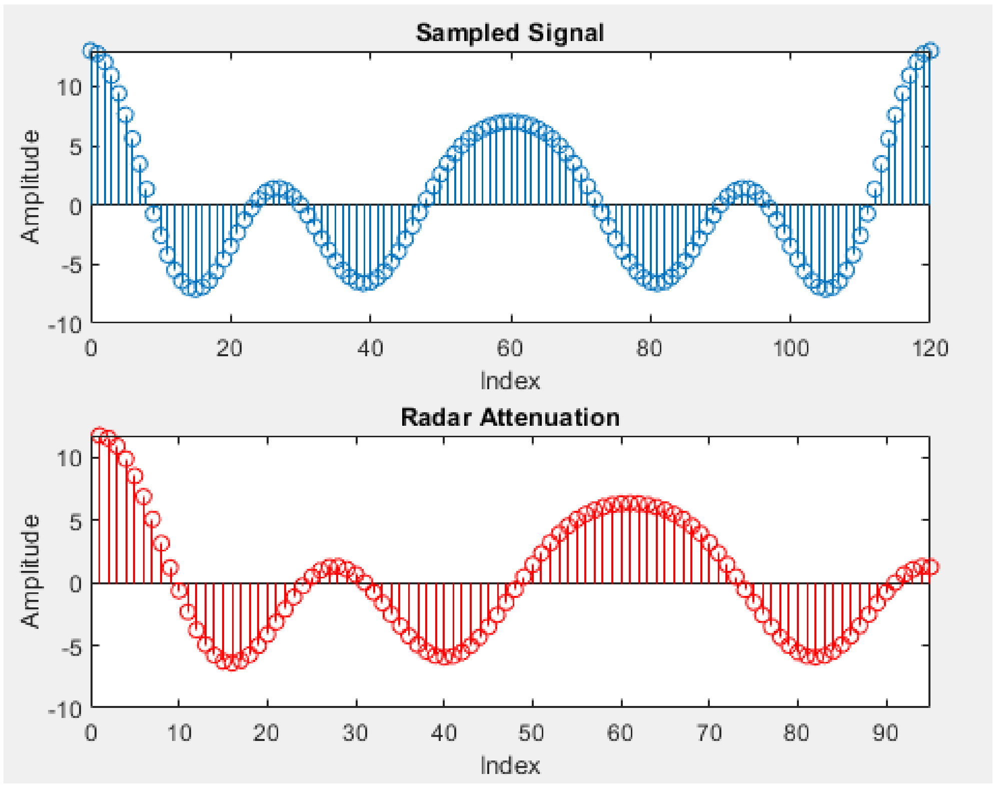


Fig 11: Sampled signal and Radar attenuation signal

Observation:

A comparison between the Sampled Signal and the Radar Attenuation Signal shows distinct differences. The Sampled Signal starts at N=0, whereas the Radar Attenuation Signal begins at N=1 due to the addition of a delay of 1. Furthermore, it is noticeable that the amplitude of the Radar Attenuation Signal is considerably lower than that of the Sampled Signal, demonstrating the impact of attenuation on the signal.

Frequency Representation of Radar Attenuation:

Code:

```
clc;
close all;
clear all;

f1=200;
f2=100;
f3=250;
Fs = 6000; %Sampling Frequency
F1 = f1/Fs;
F2 = f2/Fs;
F3 = f3/Fs;
N = 1/F3;
n = 0 : 5*N;
xt_s= 5*cos(2*pi*F1*n)+ 5*cos(2*pi*F2*n)+3*cos(2*pi*F3*n);
at=0.9; %A=9 as given above
delay=1; %B=1 as given above
radar = at.*xt_s;  %Element wise multiplication that is stored in radar
R_n= n+delay; %adding delay to the original cycles
radar_fft= fft(radar,length(radar));
id=(Fs/length(radar))*(0:length(radar_fft)/2-1);%index
radar_fft = radar_fft(1:length(radar_fft)/2);
stem(id,abs(radar_fft))
xlabel("Frequency");
ylabel("Amplitude");
title("Radar Attenuation Output");
```

Output:

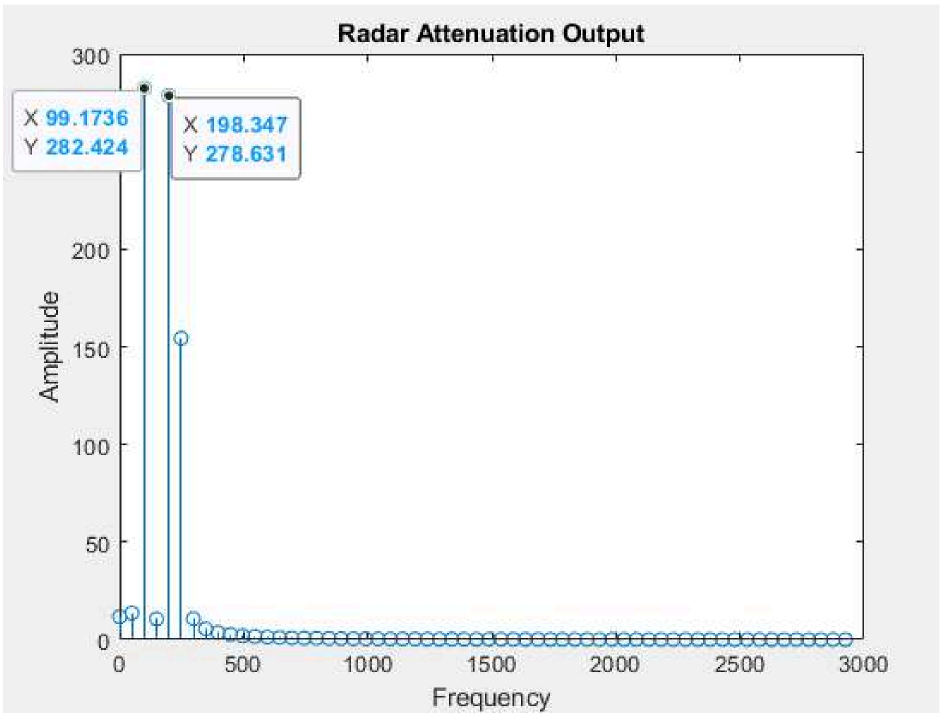


Fig 12: Radar Attunation Output

Observation:

After generating the frequency domain representation of the Radar Attenuated Signal, it is observed that the highest amplitudes are 99.1736 Hz and 198.347 Hz. This indicates that the majority of the signal's components are concentrated around these frequencies.

Noise Generation and making the input for the LTI system:

Code:

```
clc;
close all;
clear all;
f1=200;
f2=100;
f3=250;
Fs= 6000; % Sampling Frequency
F1= f1/Fs;
F2= f2/Fs;
F3= f3/Fs;
N = 1/F3;
n = 0 : 5*N; % 3 full cycles
xt_s=5*cos(2*pi*F1*n)+5*cos(2*pi*F2*n)+3*cos(2*pi*F3*n);
at=0.9; % A=9
delay=1; % B=1
radar = at.*xt_s; % Element-wise multiplication
R_n=n+delay; %adding delay
noise=(n>=7).*rand([1,length(n)]); %noise signal starting from 7
stem(n,noise);
title("Random Noise Signal");
xlabel("Index");
ylabel("Amplitude");
```

Output:

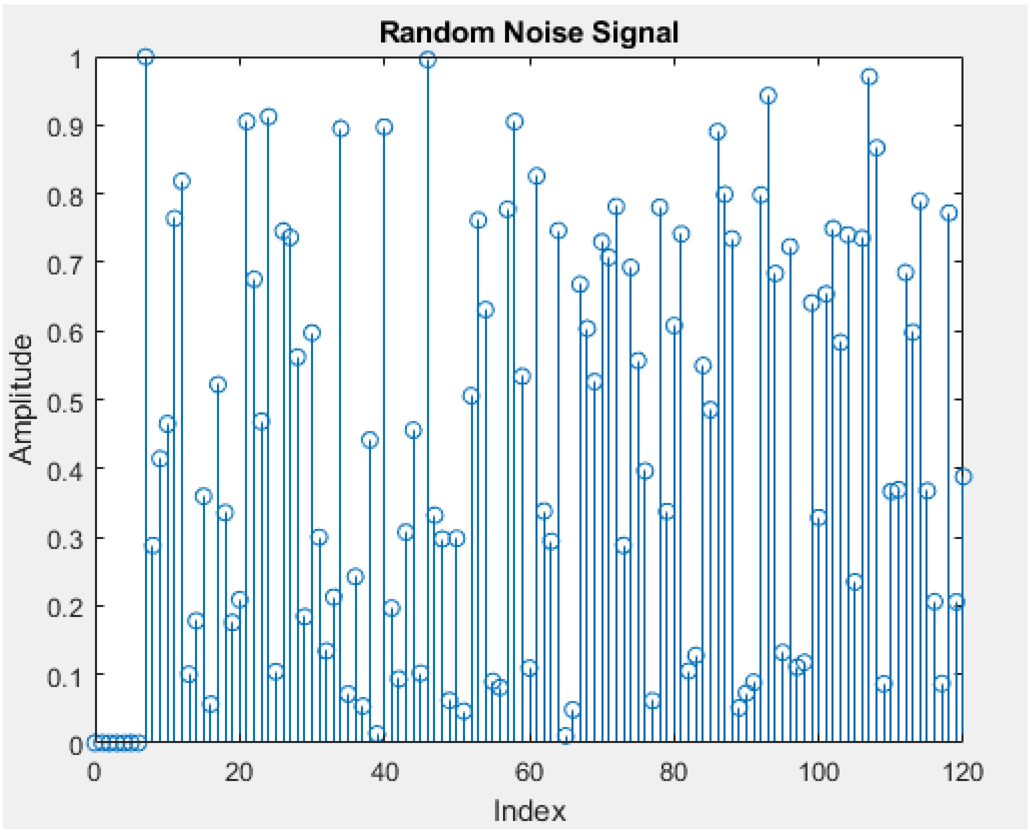


Fig 13: Generated random noise signal

Observation:

The question indicates that noise is generated when N=7 and this is demonstrated using MATLAB. The figure shows the generation of random noise.

Merging Noise with Radar in both Time Domain and Frequency Domain

Code:

```
clc;
close all;
clear all;
f1=200;
f2=100;
f3=250;
Fs= 6000; % Sampling Frequency
F1= f1/Fs;
F2= f2/Fs;
F3= f3/Fs;
N= 1/F3;
n= 0 : 5*N;
xt_s=5*cos(2*pi*F1*n)+5*cos(2*pi*F2*n)+3*cos(2*pi*F3*n);
at=0.9; % A=9
delay=1; % B=1
```

```
radar = at.*xt_s; %Element wise multiplication
R_n= n+delay; %adding delay
noise= (n>=7).*rand([1,length(n)]); %noise signal starting from 7
LTI=radar+noise; %Merging radar signal with random noise signal
LTI_n=R_n;
figure(1);
subplot (211),stem(R_n, radar,'r');
title("Radar Attenuation Output");
xlabel("Index");
ylabel("Amplitude");
xlim([0,102]);
subplot (212),stem(LTI_n,LTI);
title("Input of the LTI System in Time Domain");
xlabel("Index");
ylabel("Amplitude");
xlim([0,102]);
LTI_fft=fft(LTI,length(LTI)); % fast fourier transform of LTI input
id=(Fs/length(LTI))*(0:length(LTI_fft)/2-1);
LTI_fft=LTI_fft(1:length(LTI_fft)/2);
figure(2);
stem(id,abs(LTI_fft));
title("Input LTI System in Frequency Domain");
xlabel("Frequency (Hz)");
ylabel("Amplitude");
```

Output:

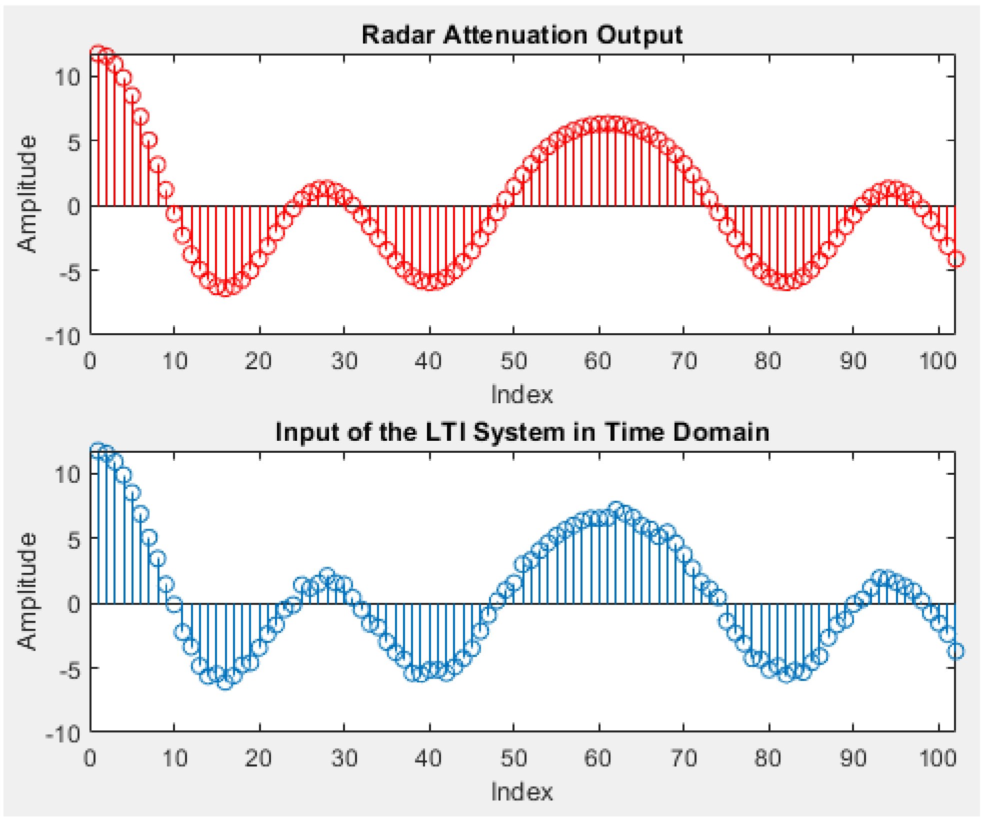


Fig 14: Radar Attenuated signal output with Input of the LTI system

Observation:

In comparing the actual graph with the one combined with noise, it is clear that irregularities appear throughout the graph, except for the first 7 points. These irregularities result from the noise added to the original signal.

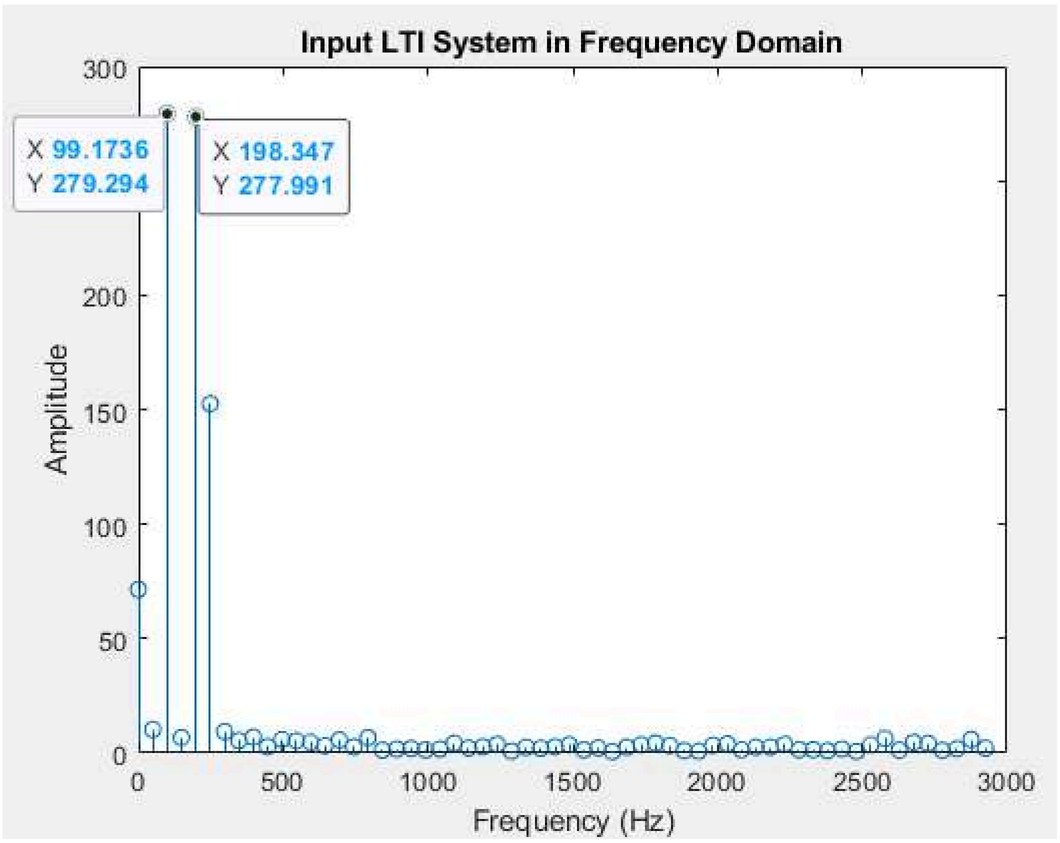


Fig 15: Input LTI system in frequency domain

Discussion:
This graph shows that the main frequency components are consistent with those of the other signals in the frequency domain.

Defining Region of Convergence (ROC) from the given pole-zero plot:

The Region of Convergence (ROC) is an essential concept in the analysis of discrete-time signals and systems, particularly when working with z-transforms. In simple terms, the ROC represents the range of values for the complex variable zzz where the z-transform of a signal converges. It defines the set of points in the complex plane where the z-transform expression is valid and the series converges absolutely, ensuring the stability and proper behavior of the system.

From the given pole-zero plot we can observe that,

$$H(z) = \frac{(z+1)(z+1)}{(z+0.25)(z-0.5)} = \frac{z^2+2z+1}{z^2-0.25z-0.125}$$

Two zeros are located at -1 and two poles are located at -0.25 and +0.5.
Thus $z_1 = 0.25 = \frac{1}{4}$ and $z_2 = 0.5 = \frac{1}{2}$
Hence, ROC for the first pole: $|z_1| > \frac{1}{4}$
ROC for the second pole: $|z_2| > \frac{1}{2}$

Thus the intersection of both the ROC of the first pole and the ROC of the second pole will be the ROC of the System is

$$\boxed{|z| > \frac{1}{2}}$$

When all of the poles of the system are located within the unit circle, it is said to be a stable system. In our case, since both the poles are located inside the unit circle, the system is stable.

If the ROC of the system is outside the outermost pole, it is referred to as a causal system. Here, the location of the outermost pole is 0.5. Therefore, the ROC for the causal system is $z > \frac{1}{2}$.

The transfer function of the LTI system is-

$$\boxed{H(z) = \frac{(z+1)(z+1)}{(z+0.25)(z-0.5)} = \frac{z^2+2z+1}{z^2-0.25z-0.125}}$$

So Based on the following equation:
The coefficient of the numerator is: [1, 2, 1]
The coefficient of the denominator is [1, -0.25, -0.125]

Generating output of the LTI system:

Code:

```
clc;
close all;
clear all;
f1=200;
f2=100;
f3=250;
Fs= 6000; % Sampling Frequency
F1= f1/Fs;
F2= f2/Fs;
F3= f3/Fs;
N= 1/F3;
n= 0 : 5*N;
xt_s=5*cos(2*pi*F1*n)+5*cos(2*pi*F2*n)+3*cos(2*pi*F3*n);
at=0.9; % A=9
delay=1; % B=1
radar = at.*xt_s; %Element wise multiplication
R_n=n+delay;
noise=(n>=7).*rand([1,length(n)]); %noise signal starting from 7
LTI=radar+noise; %Merging radar signal
LTI_n = R_n;
b = [1, 2, 1]; %Coefficients of Numerator
a = [1, -0.25, -0.125]; %Coefficients of Denominator
n = 50;
[h,t] = impz(b, a, n); %impz function to generate an impulse function
LTI_out_n = (min(LTI_n)+min(t)) : (max(LTI_n)+max(t)); % Index determine
LTI_out = conv(h, LTI);
figure(1);
stem(LTI_out_n, LTI_out);
title("Output of LTI System in Time Domain");
xlabel("Index");
ylabel("Amplitude");
xlim([0,102]);
LTI_out_fft = fft(LTI_out,length(LTI_out)); %Fast Fourier
id = (Fs/length(LTI_out)) * (0:length(LTI_out_fft)/2-1);
LTI_out_fft = LTI_out_fft(1:length(LTI_out_fft)/2);
figure(2);
stem(id,abs(LTI_out_fft));
title("Output of the LTI System in Frequency Domain");
xlabel("Frequency");
ylabel("Amplitude");
```

Output:

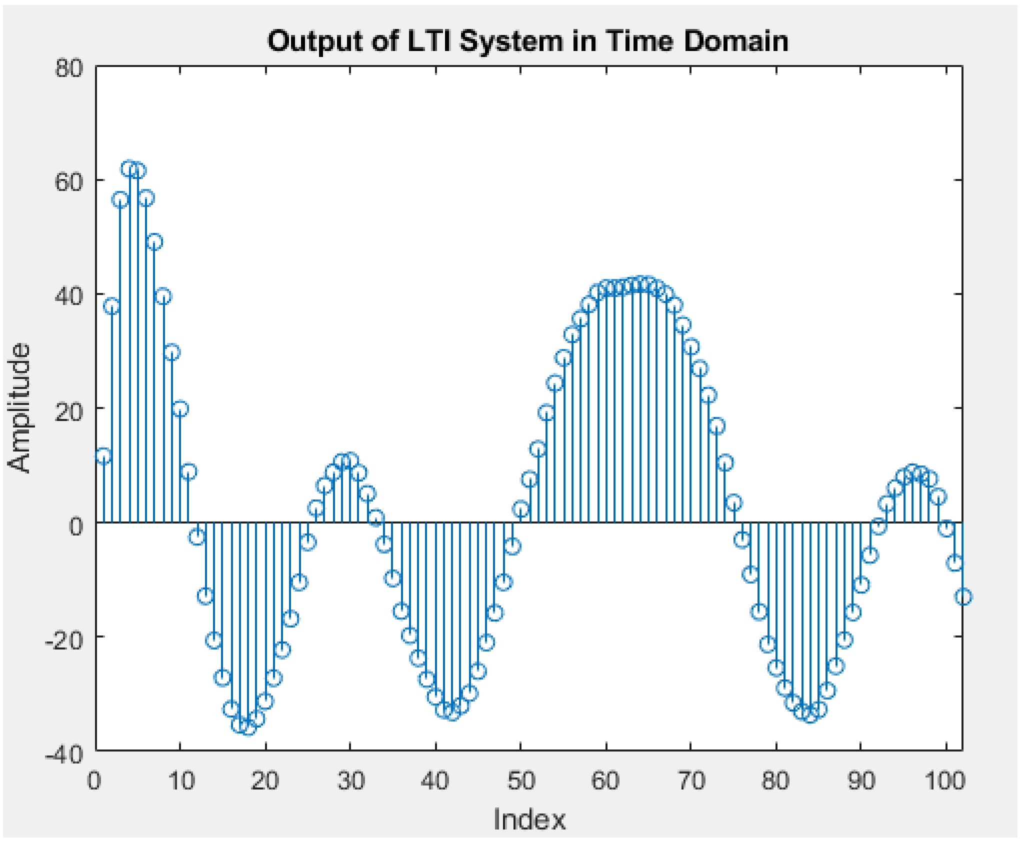


Fig 16: Output of the LTI system in Time domain

Observation:

It is apparent that, because of the delay introduced into the system with N=1, no output is observed initially.

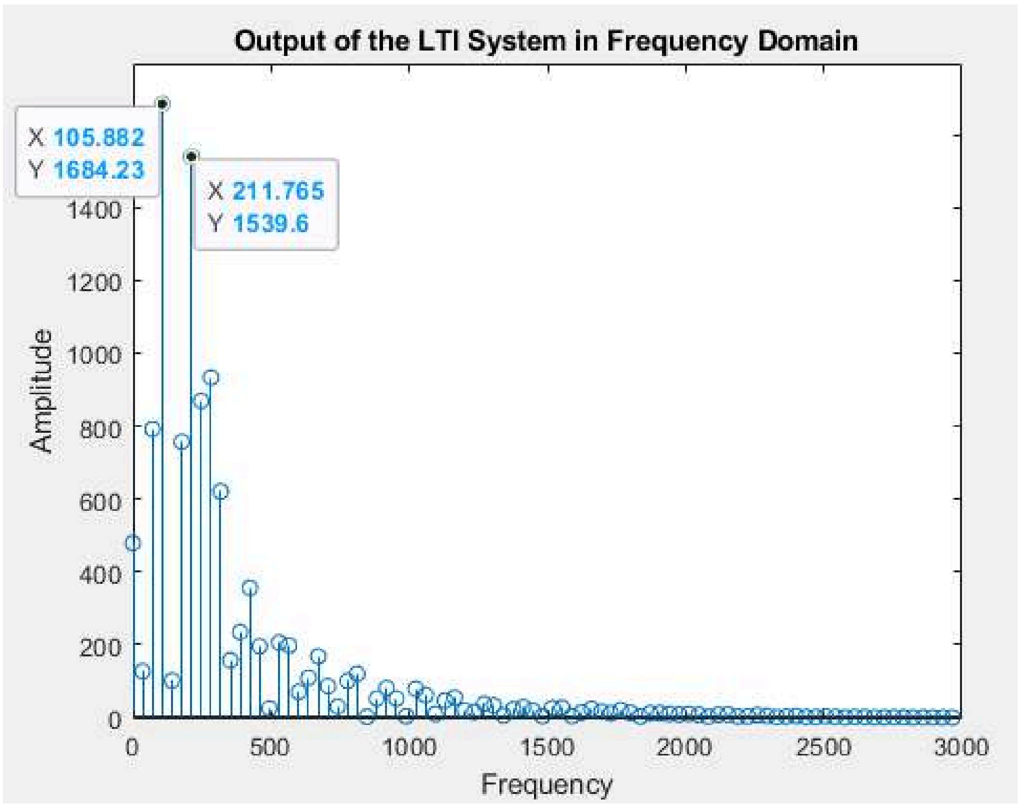


Fig 17: Output of the LTI system in the frequency domain

Observation:

In the frequency domain of the system, the output graph displays distinct peaks at 105.882 Hz and 211.765 Hz. From this, we can say that the frequency of the dominant frequency component of the output is 105.882Hz.

The frequency of the dominant frequency component of the output = 105.882 Hz

Varying the Sampling Frequency:

1. **Sampling Frequency Set to 200 Hz (Lower than the Nyquist Rate)**

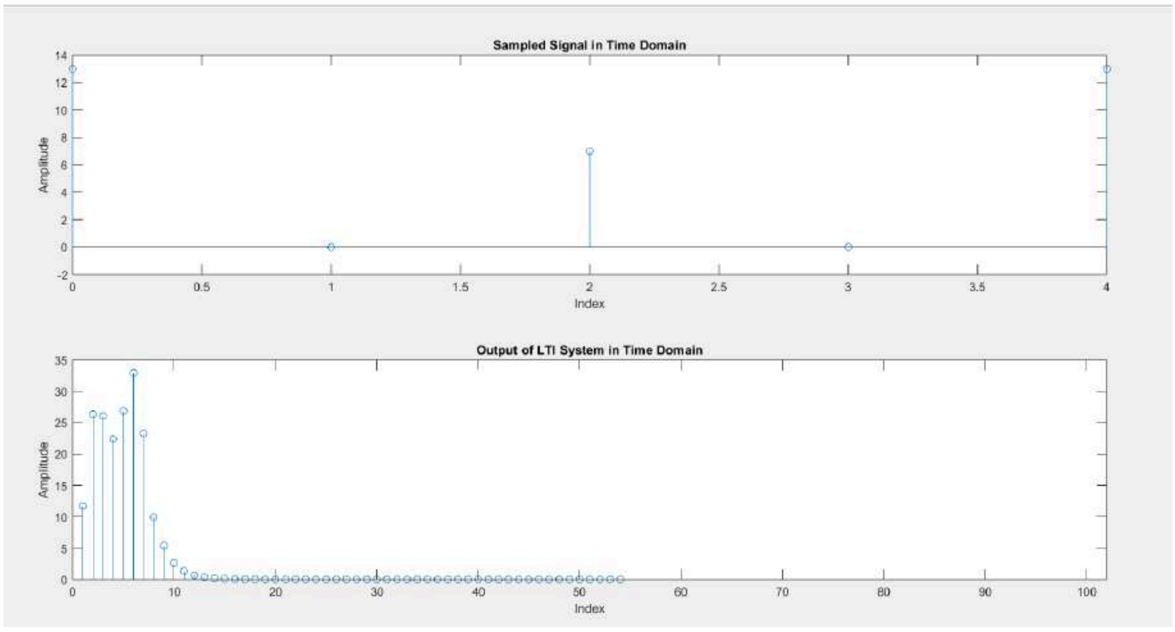


Fig 18: Sampled signal in the time domain and the output of the LTI system in the time domain for 200Hz

Observation:

It can be observed that at the sampling frequency less than the Nyquist frequency, the sampled output from the sampler does not reveal any meaningful information. The LTI system also fails to display any pattern that corresponds to the input signal at this frequency.

2. **Sampling Frequency is set to 3000 Hz (Half of the 6000 Hz that was initially chosen)**

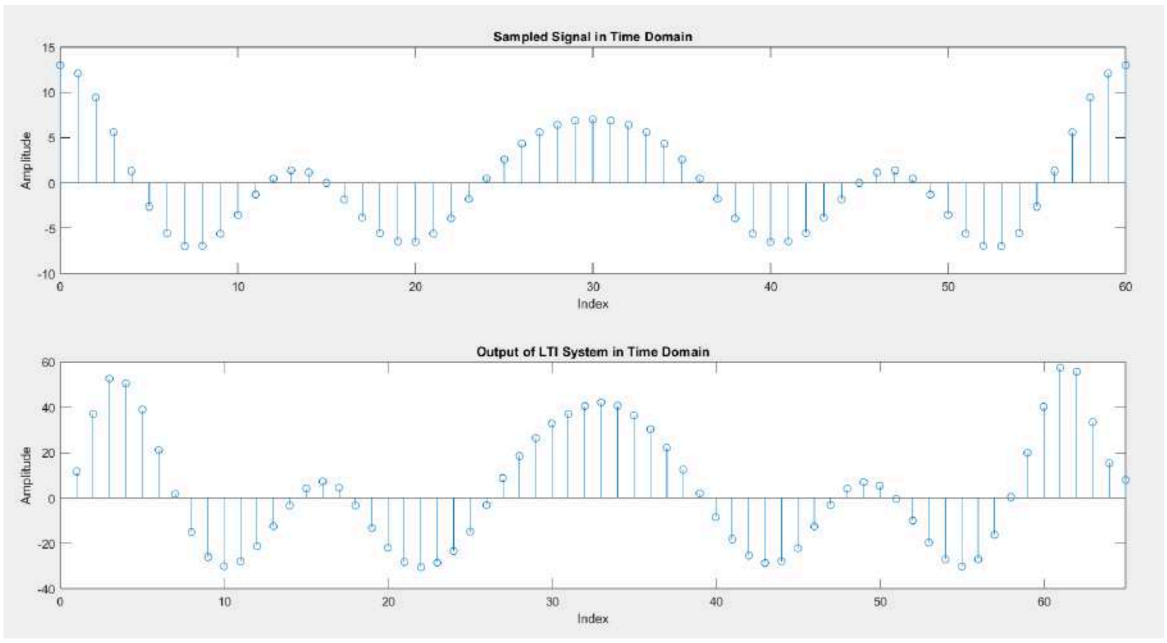


Fig 19: Sampled signal in the time domain and the output of the LTI system in the time domain in 3000Hz

Observation:

From Figure 19, it can be clearly concluded that when the sampling frequency exceeds the Nyquist frequency, the resulting outputs are more precise and closely match the input.

3. **Sampling Frequency is set to 20000 Hz**

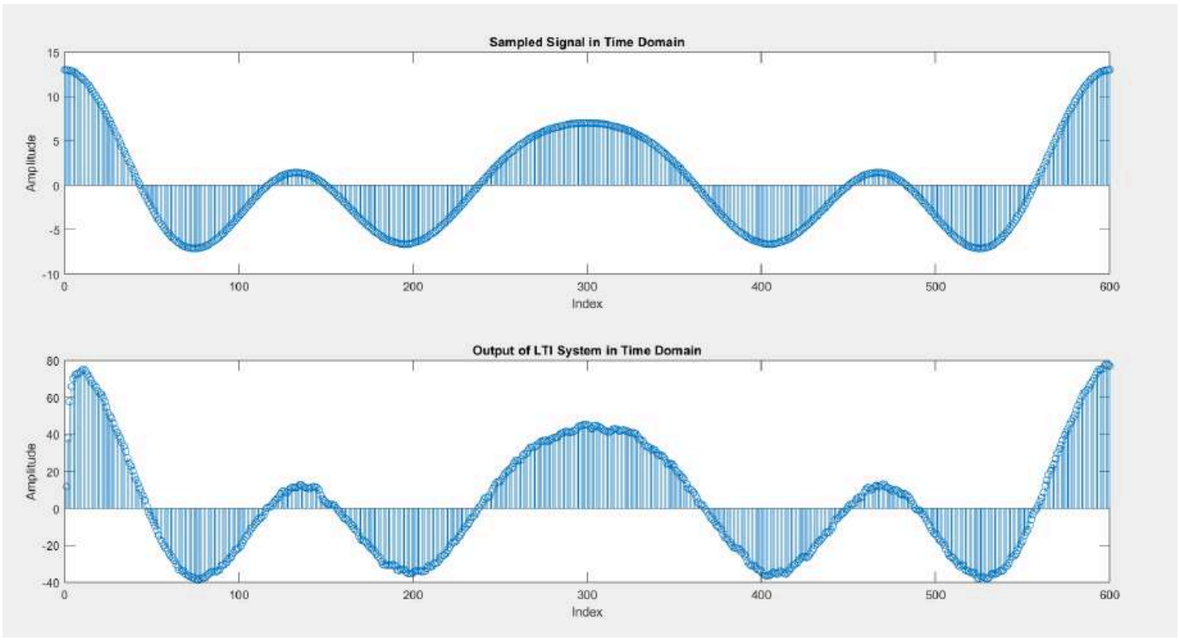


Fig 20: Sampled signal in the time domain and the output of the LTI system in the time domain in 20000Hz

Observation:

From Fig: 20, it is clear that as the sampling frequency increases, the generated outputs become more precise and closely match the input graphs.

Suggestions on Signal Reconstruction:

To reconstruct the signal, it is essential to reverse the processes applied earlier. The output signal, distorted by noise, requires careful handling to restore its original form. A band-pass filter is necessary to eliminate noise present in both higher and lower frequencies, effectively isolating the desired frequency components and reducing unwanted noise. Convolution can also be employed for this purpose. Additionally, to counteract the effects of the radar system, which introduced attenuation and delay, the signal needs to be amplified and shifted accordingly. For reconstructing the sampled signals into an analog form, interpolation is a key method. This involves generating a continuous impulse train from the discrete samples, applying a low-pass filter with an appropriate impulse response, and then convolving the filtered impulse train to achieve the reconstructed signal. Another effective approach involves using cardinal basis splines, which provide piecewise polynomial reconstructions with continuous derivatives, offering smooth signal restoration. The Whittaker-Shannon reconstruction formula is considered the ideal method for this task, utilizing an ideal low-pass filter and the sinc function impulse response to reconstruct bandlimited signals sampled at a sufficiently high rate perfectly.

Discussion:

In this project, we focused on signal addition and sampling, which are crucial for converting analog signals into discrete ones. By applying the Fast Fourier Transform (FFT), we gained valuable insights into the frequency representation of signals, improving our understanding of system dynamics. Through the analysis of pole-zero plots and the use of inverse z-transforms for LTI systems, we explored the system's characteristics and examined its output. We found that lower sampling frequencies led to inaccuracies, while higher sampling frequencies greatly improved the accuracy of the system. During the project, we encountered some notable challenges. One significant issue involved stem plots, where a mismatch in the lengths of the x and y parameters caused the error message, 'X must be the same length as Y.' We resolved this by checking and adjusting the lengths of the x and y arrays using the 'length()' function. While it was possible to consolidate all the source code into a single .m file, doing so would have prevented us from displaying all the graphs included in the report. To address this, we created separate source code files for each task.