BIRZEIT UNIVERSITY

Faculty of Engineering & Technology
Electrical & Computer Engineering Department

Digital Signal Processing - ENCS4310

MATLAB Assignment

Prepared by : Baraa Nasar

ID Number : 1210880

Instructor : Dr. Alhareth Zyoud

Section : 1

Date : 8/28/2024

Abstract

This MATLAB assignment explores various digital signal processing concepts. In the first task, we calculate and plot the spectrum of a discrete-time signal using its Fourier transform. The second task involves computing the discrete Fourier transform (DFT) of two signals, multiplying them in the frequency domain, and converting back to the time domain using the inverse DFT. The results are then compared with the direct convolution of the two signals in the time domain to illustrate the convolution theorem. In the final task, we analyze a digital filter by plotting its pole-zero diagram and the frequency response function H(z) for $z = e^{jw}$. These exercises enhance the understanding of signal processing techniques and the relationship between time and frequency domains.

Table of Contents

Abstract ____ II
Question 1 ____ V
Question 2 ____ VIII
Question 3 ____ XI

Table of Figures

Figure 1:Magnitude Spectrum of Signal x[n] _____ VI
Figure 2: Phase Spectrum of Signal x[n] _____ VII
Figure 3: Plot of the Original Signal and Impulse Response _____ IX
Figure 4: Comparison Between Convolution Result and Inverse DFT Output _____ X
Figure 5: Pole-Zero Diagram _____ XI
Figure 6: Frequency Response H(z) for z= e^ jω _____ XII

Question 1

For the following Signal x[n] Calculate and plot the Spectrum:

$$\boldsymbol{x}[n] = \begin{Bmatrix} 1, n = 1 \ldots .10 \\ 0, Otherwise \end{Bmatrix}$$

Solution:

MATLAB Code for Spectrum Analysis of x[n]:

```
%Baraa Nasar 1210880
% Define the signal x[n]
n = 1:10;               % n ranges from 1 to 10
x = ones(1, 10);        % x[n] = 1 for n = 1 to 10

% Zero-padding to improve frequency resolution
N = 100;                % Number of points for FFT, can be increased for better
resolution
x_padded = [x, zeros(1, N - length(x))]; % Zero-pad the signal

% Calculate the DFT of x[n]
X = fft(x_padded);      % Compute the Discrete Fourier Transform (DFT) of the signal

% Shift the zero-frequency component to the center of the spectrum
X_shifted = fftshift(X); % Center the zero frequency in the spectrum plot

% Frequency vector for plotting
f = (-N/2:N/2-1)/N;     % Create a normalized frequency vector for plotting

% Plot the magnitude of the spectrum
figure;
subplot(2,1,1);          % Create a subplot for magnitude
plot(f, abs(X_shifted)); % Plot the magnitude spectrum of the signal
title('Magnitude Spectrum of x[n] - 1210880'); % Title of the magnitude plot
xlabel('Normalized Frequency (\times \pi rad/sample)'); % X-axis label for magnitude
ylabel('Magnitude'); % Y-axis label for magnitude
grid on; % Enable grid for better visualization

% Plot the phase of the spectrum
subplot(2,1,2);          % Create a subplot for phase
plot(f, angle(X_shifted)); % Plot the phase spectrum of the signal
title('Phase Spectrum of x[n] - 1210880'); % Title of the phase plot
xlabel('Normalized Frequency (\times \pi rad/sample)'); % X-axis label for phase
ylabel('Phase (radians)'); % Y-axis label for phase
grid on; % Enable grid for better visualization
```

The MATLAB code successfully computes and plots the magnitude and phase spectra of the discrete-time signal x[n]. By applying the Discrete Fourier Transform (DFT), we can observe how the signal's energy is distributed across different frequency components. The zero-padding technique used in the code improves the frequency resolution, allowing for a clearer visualization of the spectral content. This analysis helps in understanding the signal's characteristics in both time and frequency domains, which is essential for various applications in signal processing.

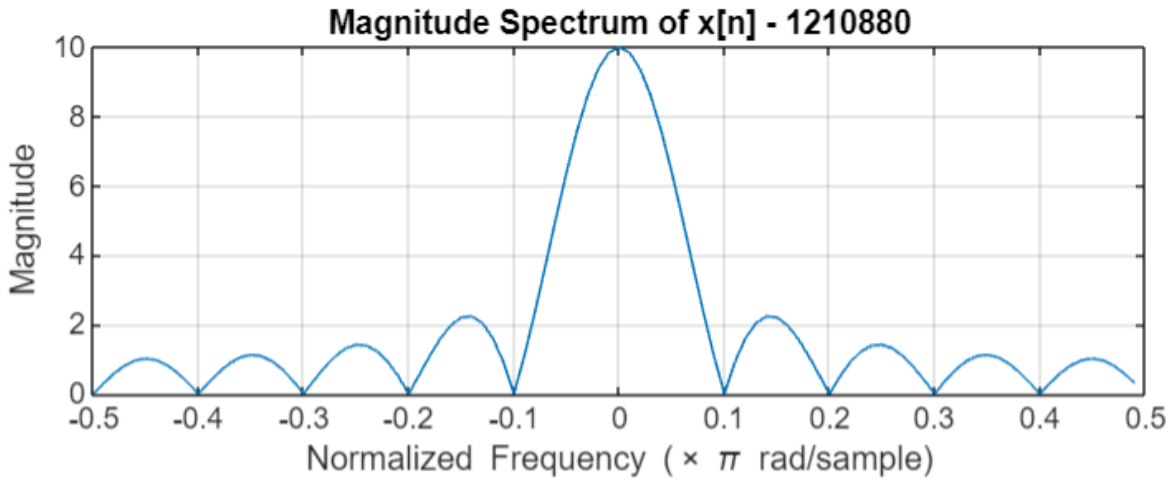

Figure 1:Magnitude Spectrum of Signal x[n]

- The magnitude spectrum shows a sinc-like pattern, which is typical for a rectangular pulse in the time domain. The main lobe of the spectrum corresponds to the central frequency component of the signal, while the side lobes represent higher frequency components with diminishing magnitudes.

- The concentration of energy at low frequencies indicates that the signal has most of its power distributed near the zero frequency, confirming that x[n] is a low-pass signal.

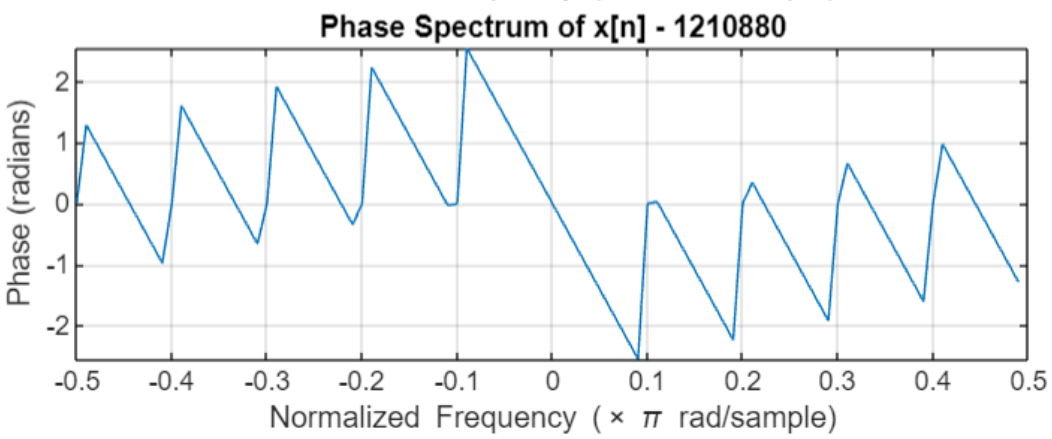

Figure 2: Phase Spectrum of Signal x[n]

- The phase spectrum reveals the phase relationship between the frequency components of the signal. For this particular signal, the phase is relatively constant, except for specific points where abrupt changes occur, indicating phase shifts at certain frequencies.
- Understanding the phase information is crucial in applications where the timing and phase relationship between signal components are important, such as in communication systems and audio processing.

Overall Conclusion:

The analysis of the magnitude and phase spectra provides valuable insights into the nature of the signal x[n]. By examining the frequency components, we can better understand the signal's behavior and its implications in various applications. The DFT and the resulting spectra are powerful tools for analyzing signals in the frequency domain, revealing details that are not readily apparent in the time domain.

Question 2

Consider the following signals:

$x[n]$ = [0, 0.3, 0.6, 0.8, 1] and $h[n]$ = [0.5, 1, 0.5]

Calculate the product of their DFTs and then come back to the time domain by inverse DFT. Compare the obtained result to the convolution product of the two signals and conclude.

Solution:

MATLAB Code for Analyzing and Comparing the Convolution of Two Signals with Their DFT Product:

```
%Baraa Nasar 1210880
% Given signals
x = [0, 0.3, 0.6, 0.8, 1];
h = [0.5, 1, 0.5];

% Length of the output signal after convolution
N = length(x) + length(h) - 1;

% Compute the DFTs of the two signals with zero-padding to match the length N
X = fft(x, N);
H = fft(h, N);

% Multiply the DFTs (element-wise multiplication)
Y = X .* H;

% Compute the inverse DFT to obtain the result in the time domain
y_ifft = ifft(Y);

% Compute the linear convolution of the two signals
y_conv = conv(x, h);

% Display the results
disp('Result from inverse DFT (ifft of product of DFTs):');
disp(y_ifft);

disp('Result from linear convolution:');
disp(y_conv);

% Compare the results
if isequal(round(y_ifft, 10), round(y_conv, 10))  % Compare with some tolerance due
to numerical precision
    disp('The results are identical. The product of the DFTs and the linear
convolution give the same result.');
else
    disp('The results are different.');
end

% Plot the original signals
figure;
subplot(4,1,1);
```

```
stem(0:length(x)-1, x, 'filled');
title('Original Signal x[n] - 1210880');
xlabel('n');
ylabel('x[n]');
grid on;

subplot(4,1,2);
stem(0:length(h)-1, h, 'filled');
title('Impulse Response h[n] - 1210880');
xlabel('n');
ylabel('h[n]');
grid on;

% Plot the convolution result
subplot(4,1,3);
stem(0:length(y_conv)-1, y_conv, 'filled');
title('Convolution Result (y_{conv}[n] = x[n] * h[n]) - 1210880');
xlabel('n');
ylabel('y_{conv}[n]');
grid on;

% Plot the result from inverse DFT
subplot(4,1,4);
stem(0:length(y_ifft)-1, real(y_ifft), 'filled');
title('Result from Inverse DFT (Real Part of y_{ifft}[n]) - 1210880');
xlabel('n');
ylabel('y_{ifft}[n]');
grid on;
```

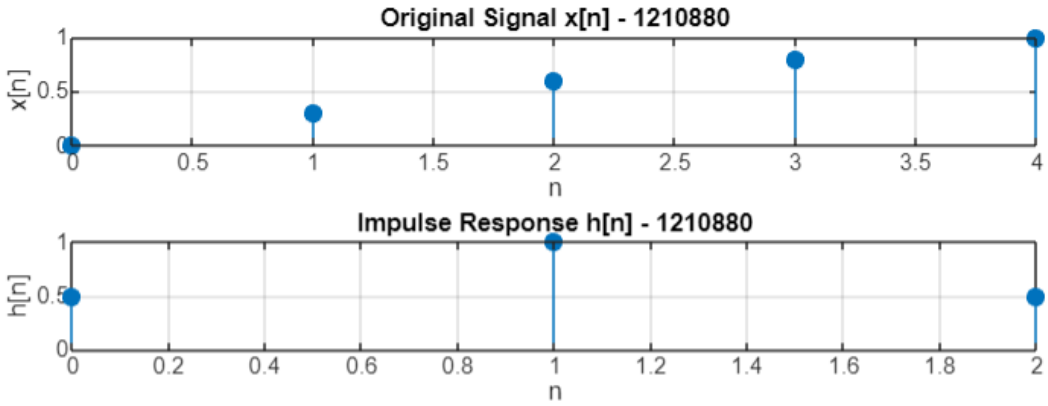

Figure 3 : Plot of the Original Signal and Impulse Response

This plot shows the original signal x[n] and the impulse response h[n]. The signal x[n] represents the sequence of samples that describe the input signal, while h[n] is the system's response to a unit impulse. These signals are the primary inputs used in the analysis of convolution and their corresponding Fourier transforms.

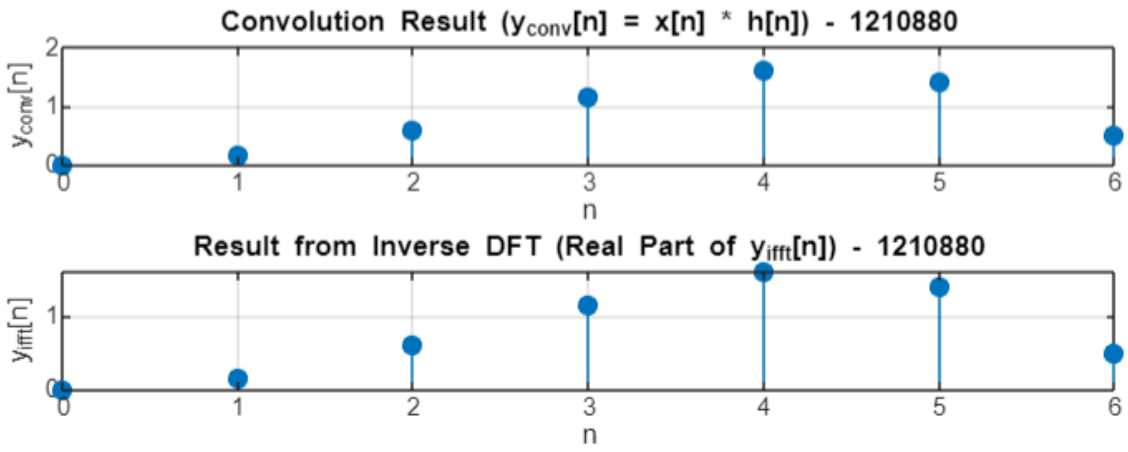

Figure 4: Comparison Between Convolution Result and Inverse DFT Output

This plot compares the result of the convolution of the two signals y_{conv}[n] with the output obtained from the inverse Discrete Fourier Transform y_{ifft}[n]. This demonstrates the equivalence between the two operations, where the convolution of the signals in the time domain matches the product of their Fourier Transforms in the frequency domain, followed by an inverse Fourier Transform to return to the time domain.

The Result:

```
>> untitled
Result from inverse DFT (ifft of product of DFTs):
         0    0.1500    0.6000    1.1500    1.6000    1.4000    0.5000

Result from linear convolution:
         0    0.1500    0.6000    1.1500    1.6000    1.4000    0.5000

The results are identical. The product of the DFTs and the linear convolution give the same result.
```

Analysis of Results

The analysis shows that the results obtained from the inverse Discrete Fourier Transform (DFT) of the product of the two signal DFTs and the linear convolution of the original signals are identical. Specifically, both approaches yielded the output:

[0,0.1500,0.6000,1.1500,1.6000,1.4000,0.5000}

This outcome confirms a fundamental property of the Fourier Transform: the convolution of two signals in the time domain is equivalent to the multiplication of their Fourier Transforms in the frequency domain. Consequently, when we multiply the DFTs of the two signals and then apply the inverse DFT, we reconstruct the exact same sequence as the one produced by direct convolution in the time domain.

This result validates the theoretical understanding that the DFT and inverse DFT processes are consistent with the linear convolution operation, making them powerful tools for signal processing, particularly in contexts where efficient computation in the frequency domain is required.

Question 3

Consider the following impulse response of a digital filter $h[n] = \{1, 1.5, 1\}$

a) Plot the Pole zero diagram

b) Plot function $H(z)$ for $z = e^{jw}$

Solution:

a)

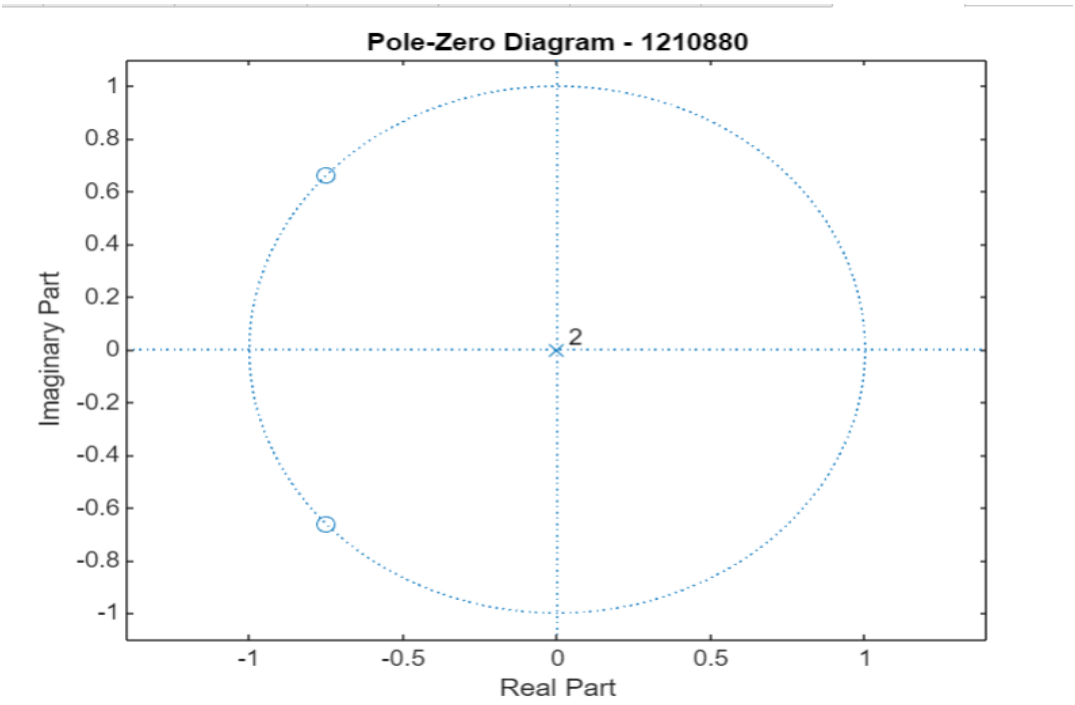

Figure 5: Pole-Zero Diagram

In this section, we analyze and illustrate the pole-zero diagram of the given digital filter. The pole-zero diagram visually represents the locations of the zeros and poles of the system in the Z-plane. Zeros are the values of z that make the transfer function H(z) equal to zero, and poles are the values of z that make the transfer function undefined (i.e., where the denominator equals zero).

MATLAB Code:

```
%Baraa Nasar 1210880
% Define the coefficients of the transfer function
b = [1 1.5 1];  % Coefficients of the numerator
a = [1];        % Coefficients of the denominator (no poles at the origin)

% Plot the pole-zero diagram
figure;
zplane(b, a);
title('Pole-Zero Diagram - 1210880');
```

b)

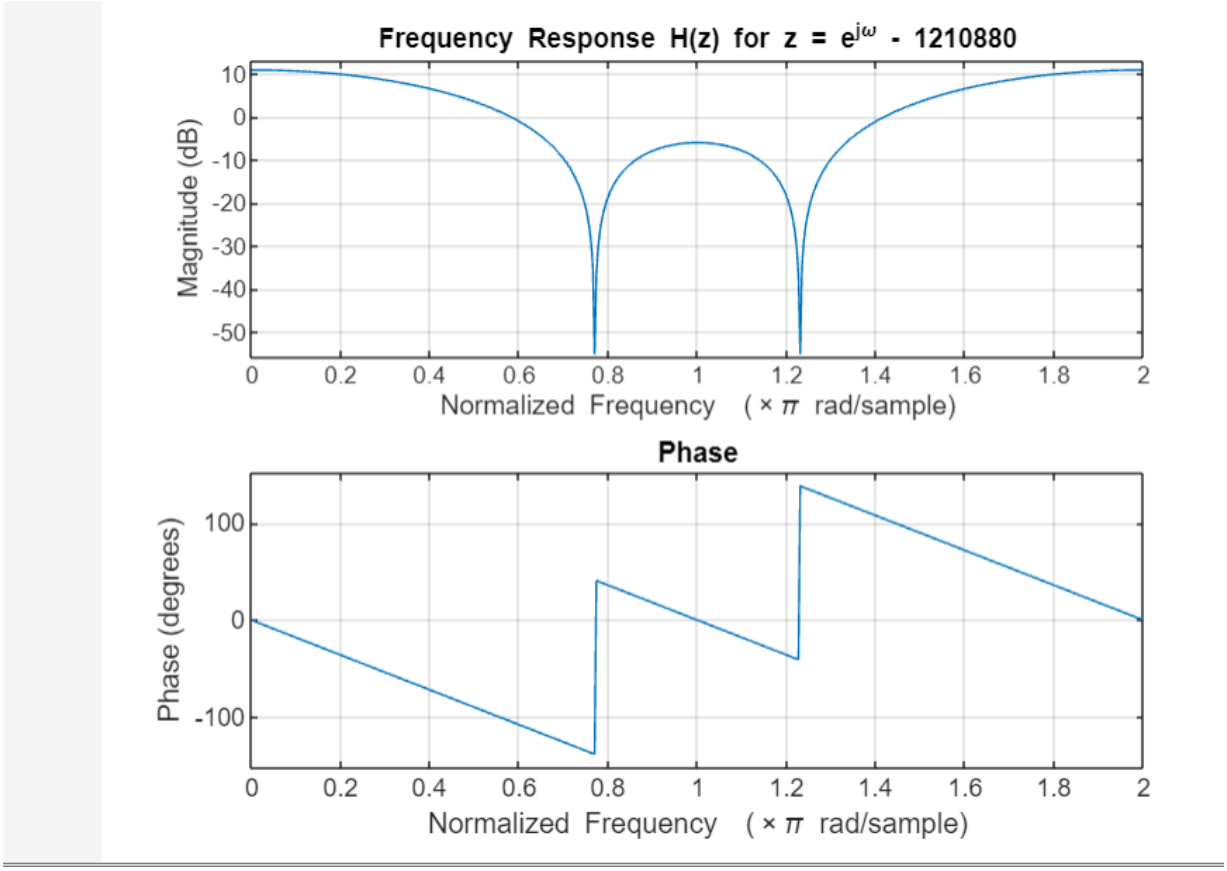

Figure 6: Frequency Response H(z) for z= e^ jω

In this part, we compute and plot the frequency response of the transfer function H(z) over the entire frequency range. The frequency response H(e^jω) shows how the system responds to different frequencies, providing insight into the behavior of the filter across the spectrum from ω=0 to ω=2π.

MATLAB Code:

```
%Baraa Nasar 121088
% Frequency response plot
figure;
freqz(b, a, 'whole');  % 'whole' covers the full range of ω from 0 to 2π
title('Frequency Response H(z) for z = e^{j\omega} - 1210880');
```

Conclusion:

we successfully generated and analyzed the pole-zero diagram and frequency response of the given digital filter with an impulse response of h[n]={1,1.5,1}.The pole-zero diagram helps us understand the stability and behavior of the filter, while the frequency response shows how the filter behaves across different frequencies.