Mamadou Kaba

27070179

ELEC 342 CN-X

18/07/2024

25/07/2024

Nadia Abdolkhani

Objective

This lab aims to explore the Discrete Time Fourier Transform (DTFT) and introduce the basics of Simulink within MATLAB. In Part I of the lab, we will implement the DTFT of a pulse input using loops and arrays, focusing on accurately plotting the magnitude of complex arrays. This section will enhance our understanding of how to compute and visualize the DTFT of discrete signals. In Part II, we will be introduced to Simulink, a simulation environment within MATLAB. We will learn to create simple models, simulate both continuous and discrete systems, and analyze their responses to various inputs. Through these exercises, we aim to develop a deeper understanding of signal processing and system modeling, leveraging MATLAB and Simulink's powerful capabilities.

Theory

Discrete Time Fourier Transform (DTFT)

The Discrete Time Fourier Transform (DTFT) is a powerful tool used in signal processing to analyze the frequency content of discrete signals. The DTFT of a discrete signal x[n] is defined by the equation: $X(e^{jw}) = \sum_{n=0}^{\infty} x[n]e^{-jwn}$

where ω represents the frequency variable. The DTFT produces a continuous function of frequency, which is typically complex-valued. When plotting the DTFT, we often focus on the magnitude and phase of the resulting complex function. The magnitude of the DTFT can be obtained using the ‘abs’ function in MATLAB, which computes the absolute value of each complex element, effectively capturing the signal's frequency content.

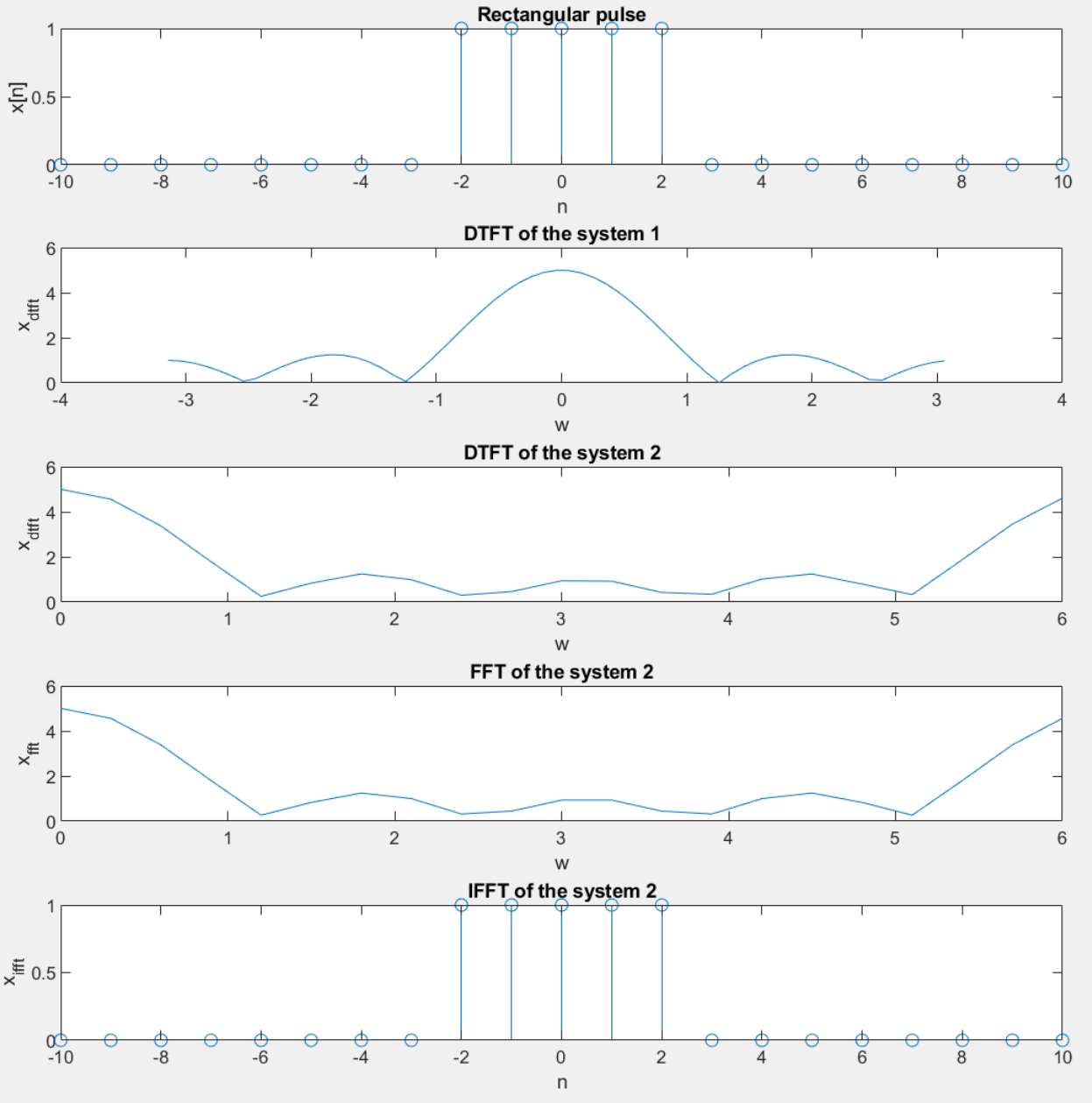
Plotting Complex Arrays in MATLAB

When dealing with complex numbers in MATLAB, direct plotting commands like plot will ignore the imaginary parts and plot only the real components. To visualize the magnitude of complex arrays, the ‘abs’ function is used to compute the magnitudes before plotting. This is crucial for correctly interpreting the frequency content of signals in the DTFT.

Results/Discussion

Part 1

Question 1, 2 and 3

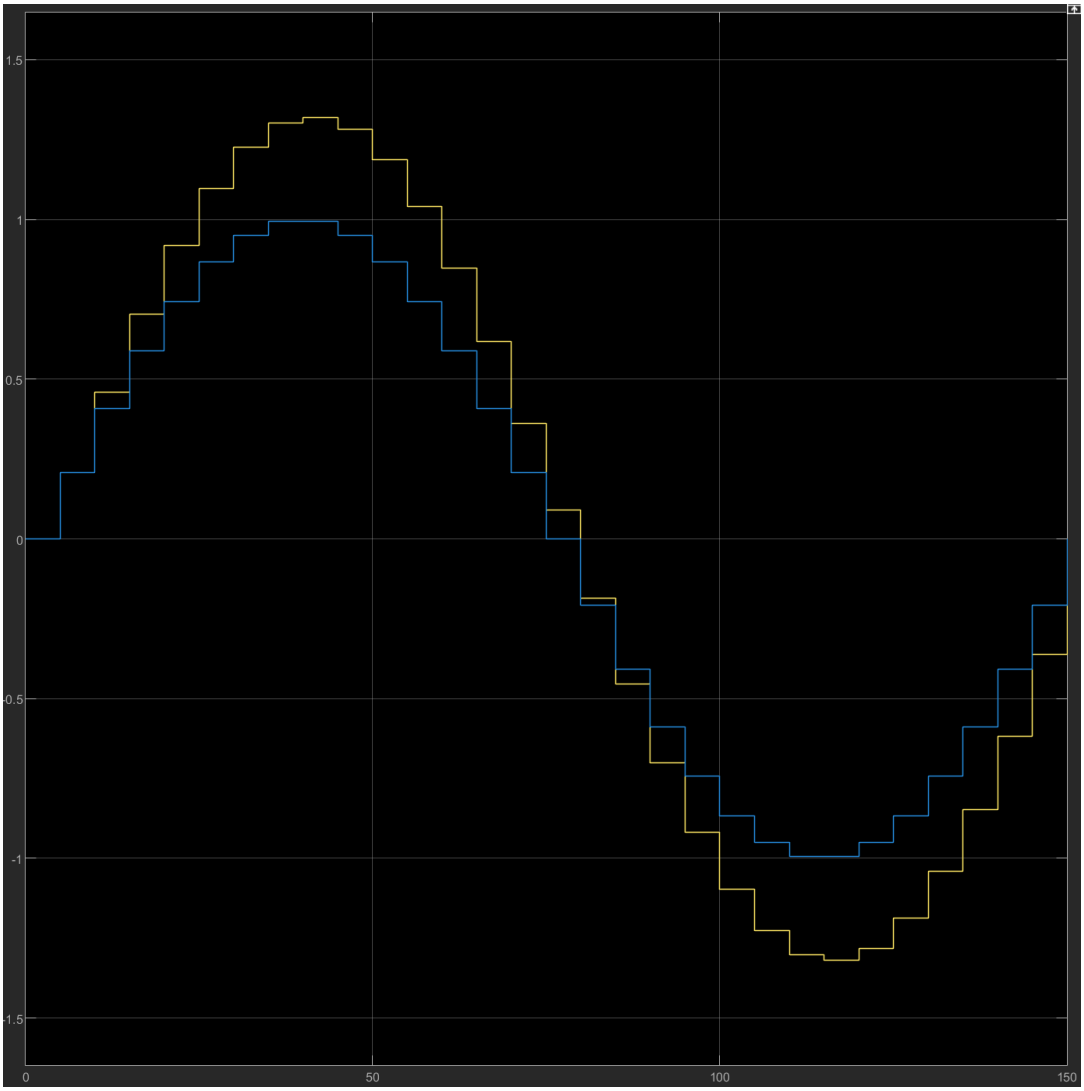


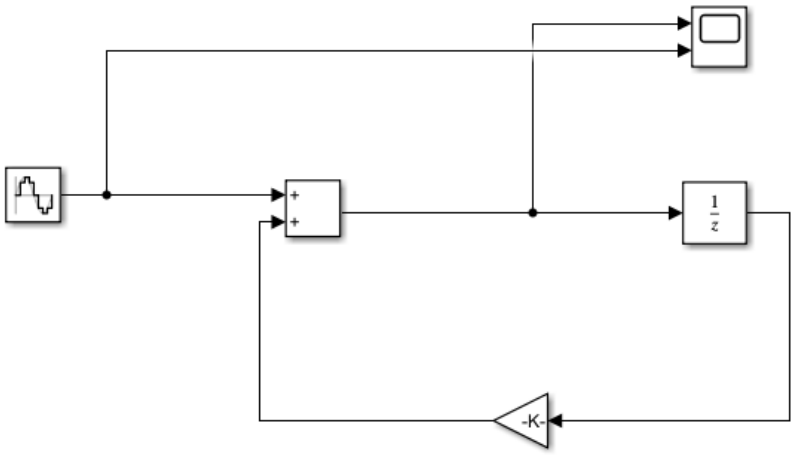
The provided plots illustrate the analysis of a rectangular pulse signal using various Fourier transform techniques. The first subplot displays the original rectangular pulse x[n], which is

defined as 1 for n between -2 and 2 and 0 elsewhere. The second and third subplots show the magnitude of the discrete-time Fourier transform (DTFT) over two different frequency ranges: $-\pi \leq w \leq \pi$ and $0 \leq w \leq 2\pi$, respectively. Both DTFT plots capture the frequency content of the signal, highlighting its periodic nature. The fourth subplot presents the fast Fourier transform (FFT) of the signal, which approximates the DTFT and shows similar frequency characteristics over $0 \leq w \leq 2\pi$. Finally, the fifth subplot illustrates the inverse fast Fourier transform (IFFT) of the FFT result, successfully reconstructing the original rectangular pulse, thereby validating the accuracy of the FFT and IFFT operations.

Part 2

Question 1





The Simulink simulation modeled the difference equation $y[n]=x[n]+1/4y[n-1]$ using the specified blocks: an Integer Delay block, a Gain block, an Add block, a Sine Wave Source block, and a Scope Sink block. The simulation results, depicted in the provided plot, show two waveforms: the input sine wave x[n] and the output y[n]. The blue waveform represents the input sine wave, while the yellow waveform corresponds to the output of the system. The output waveform y[n] demonstrates the effect of the recursive relationship defined by the difference equation. It shows a smoothing effect, where the influence of the previous output sample $y[n-1]$ is scaled by a factor of 1/4 and added to the current input sample x[n]. This results in a phase shift and amplitude modulation of the output sine wave compared to the input.

Conclusion

This lab successfully demonstrated the implementation and analysis of the Discrete Time Fourier Transform (DTFT) and introduced the basics of Simulink for system modeling. Through MATLAB, we computed and visualized the DTFT, FFT, and IFFT of a rectangular pulse, highlighting the importance of accurately handling complex arrays. The Simulink simulation provided valuable insights into the behavior of a discrete-time system defined by a difference equation, illustrating the recursive influence on the output signal. These exercises enhanced our understanding of signal processing and system dynamics, showcasing the powerful capabilities of MATLAB and Simulink in analyzing and modeling discrete-time systems.

Appendix

Lab2_part1

```
% Mamadou Diao Kaba 27070179
% Question 1,2,3

clc;clear;close all;

n=-10:10;
x = zeros(1, length(n));
x(n >=-2 & n <= 2) = 1 ;

w = -pi:0.1:pi;
w2 = 0:0.3:2*pi;

for i=1:length(w)
    sum=0;
    for k=1:length(n)
        sum = sum + (x(k)*exp(-1j*w(i)*n(k)));
    end
    x_dtft1(i) = sum;
end

for i=1:length(w2)
    sum=0;
    for k=1:length(n)
        sum = sum + (x(k)*exp(-1j*w2(i)*n(k)));
    end
    x_dtft2(i) = sum;
end

x_fft = fft(x);

x_ifft = ifft(x_fft);

subplot(5,1,1)
stem (n,x)
xlabel('n');
ylabel('x[n]');
title('Rectangular pulse');

subplot(5,1,2)
plot(w,abs(x_dtft1))
xlabel('w')
ylabel('x_{dtft}')
title('DTFT of the system 1')

subplot(5,1,3)
plot(w2,abs(x_dtft2))
xlabel('w')
ylabel('x_{dtft}')
title('DTFT of the system 2')
```

```
subplot(5,1,4)
plot(w2,abs(x_fft))
xlabel('w')
ylabel('x_{fft}')
title('FFT of the system 2')

subplot(5,1,5)
stem(n,x_ifft)
xlabel('n')
ylabel('x_{ifft}')
title('IFFT of the system 2')
```