Lab 3
Elec 342

Adrian PATTERSON
ID: 40048841

November 10, 2020

Date Performed:	October 27, 2020
Instructor:	Professor Lynch
Section:	UQ-X

I certify that this submission is my original work and meets the Faculty's Expectations of Originality.

Université Concordia University
GINA CODY SCHOOL OF ENGINEERING AND COMPUTER SCIENCE

1 Introduction

In the third experiment for ELEC 342, titled "*The Discrete Time Fourier Transform and Introduction to Simulink*," Discrete Time Fourier transform in MATLAB and Simulink basics were investigated. Experiment 3 consisted of two parts; part one explored the discrete Fourier transform of a pulse input, and part two introduced Simulink, a simulation environment in MATLAB. In the first part, the fundamental MATLAB concepts learned in the first two experiments were implemented in order to perform and plot discrete time Fourier transforms. On the other hand, the second part introduced an entirely new aspect of MATLAB: The Simulink simulation environment. In retrospect, experiment two provided key insights on how MATLAB can be used for simulating discrete Fourier transforms as well as how Simulink can be useful for visualizing discrete quantization of continuous signals.

2 Theory

The objective of experiment 3 was to use MATLAB to compute the Discrete Time Fourier Transform. Specifically, in part one, the Fourier transform was calculated both manually and using a MATLAB function, and the results thereof were compared. It is therefore necessary to have a base understanding of the discrete fourier transform and how it works mathematically. The following equation describes the Discrete Time Fourier Transform.

a. **Discrete Time Fourier Transform**
The Discrete Time Fourier Transform of some equation $x[n]$ is given by

$$X(e^{jw}) = \sum_{n=-\infty}^{n=+\infty} x[n] \cdot e^{-jwn} \qquad (1)$$

This equation was implemented using MATLAB to compute the Discrete Time Fourier Transform.

3 Results, Discussion, and Questions

The results, discussion, and questions for experiment 3 will be discussed separately, in sections 1 and 2. Section 1 concerns the MATLAB portion of this experiment, while section 2 concerns the Simulink portion.

a. **Part 1**

(a) Question 1

In part (a), the objective was to obtain the DTFT of the following pulse x[n], as seen below in figure 1.

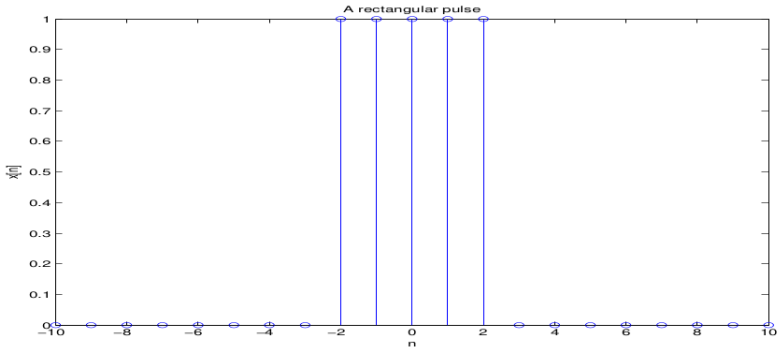

Figure 1: x[n] for Part 1, Question 1

For computing the DTFT, the step size used was user defined. Thus, the script waits and asks the user to enter a step size before. Then, using equation 1 the values of the DTFT over the range [-10,10] were computed with the user defined step size. Figure 2 below shows an output when the step size w was 0.1.

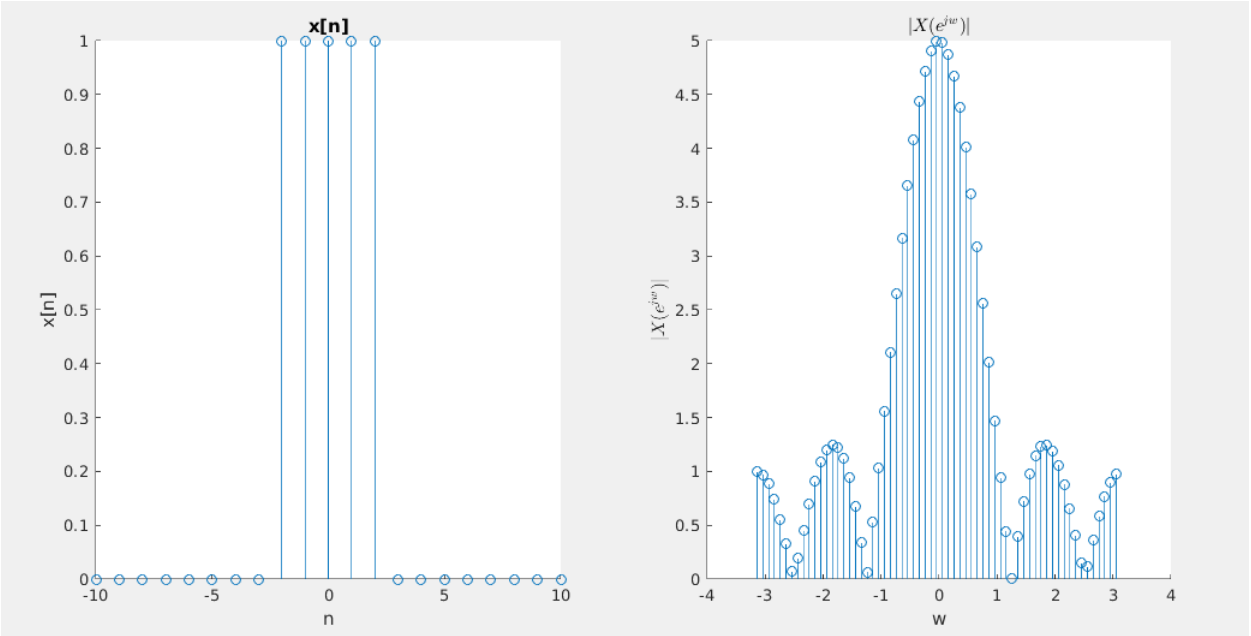

Figure 2: Part 1, Question 1 Plot

(b) Question 2

Similar to question 1, we computed the DTFT of the the input in figure 1. However, in question 2 we computed the DTFT in two different ways. First, we computed it using the manual method (i.e. using equation 1). Second, we computed it using MATLAB's built in `fft` function to take the DTFT. Finally, the results were plotted over the range $[0, 2\pi]$. Before plotting both transforms, though, it was necessary to choose a step size w that corresponded to MATLAB's built in function. In other words, it was necessary to have an array of size 21. Over a range from $[0, 2\pi]$, the step size w is thus given by

$$w = \frac{2\pi}{21} \approx 0.3$$

Therefore, using a step size of 0.3 for the manually computed DTFT, we see the comparison of the manually generated DTFT and the MATLAB function-generated DTFT in figure 3 below.

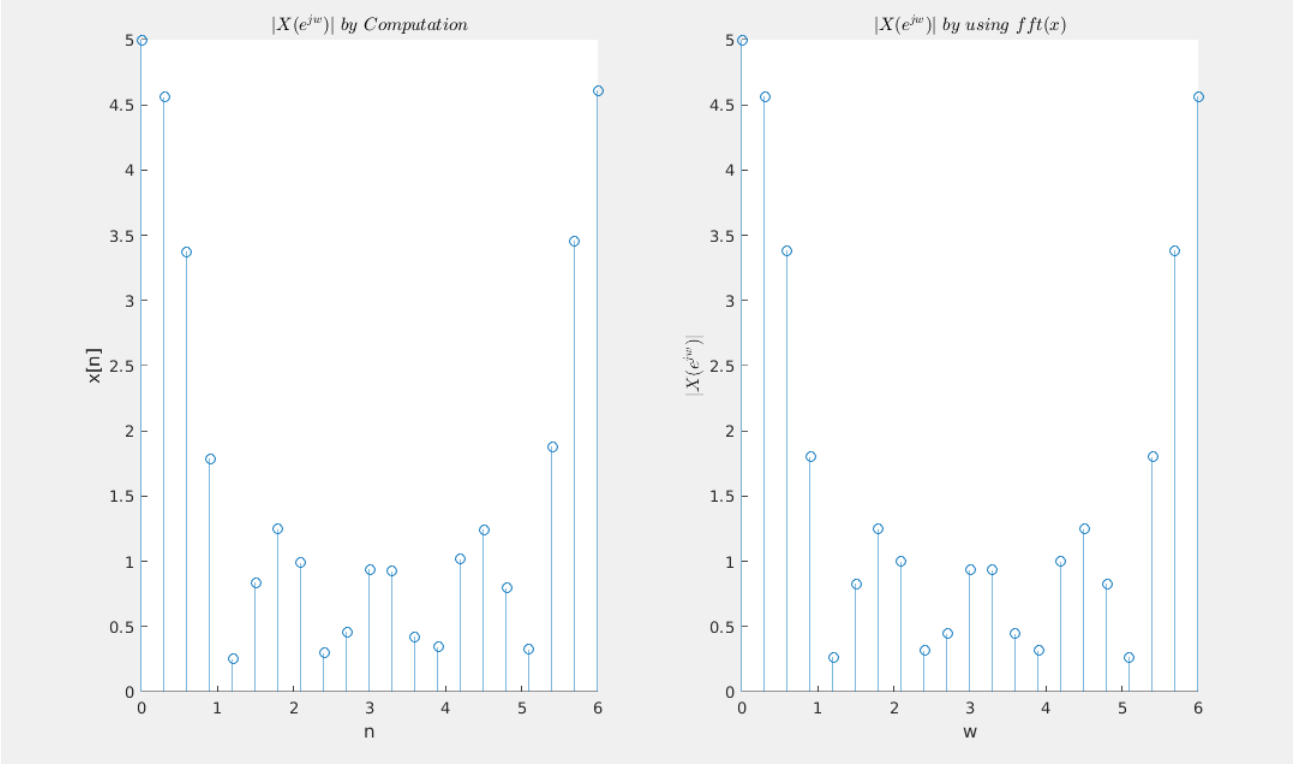

Figure 3: Part 1, Question 2 Plot

(c) Question 3

In question 3, the DTFT of x[n] was first taken using MATLAB's built-in `fft` function. Then, the inverse fourier transform of this was taken using the `ifft` function, theoretically resulting in our original equation x[n]. Figure 4 below shows the comparison of x[n] and $F^{-1}\{F\{x[n]\}\}$.

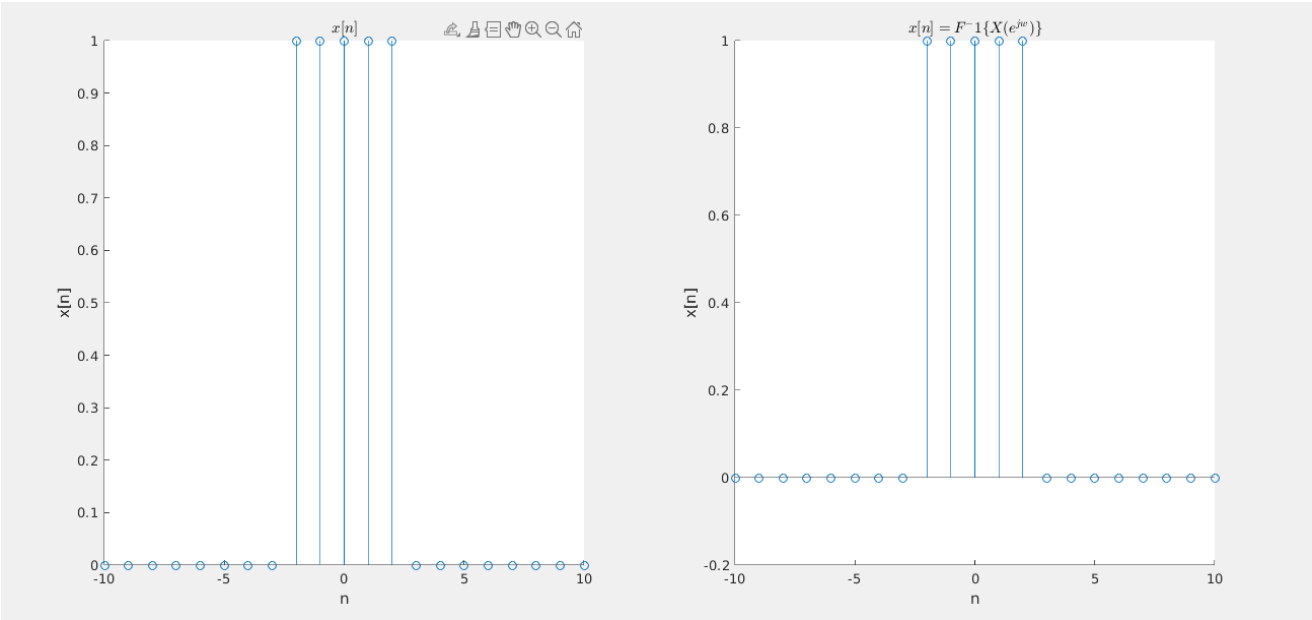

Figure 4: Part 1, Question 3 Plot

b. Part 2

In part 2, we used Simulink to simulate a sinusoidal singal. First, a Simulink project was instantiated. Second, a sine wave sink and an oscilloscope were added to the model. Finally, sine wave was connected to the oscilloscope so that the output could be viewed and recorded.
First, the output was observed for the normal sinusoidal wave as seen in figure 5.

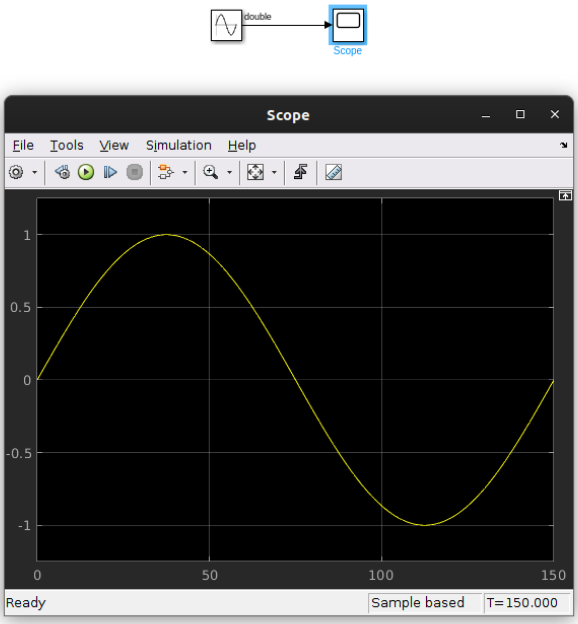

Figure 5: Part 2, Normal Sine Output

Then, the sample time was set to 5 seconds, increasing the quantization error. In this plot, the quantization error is evident as seen below in figure 6.

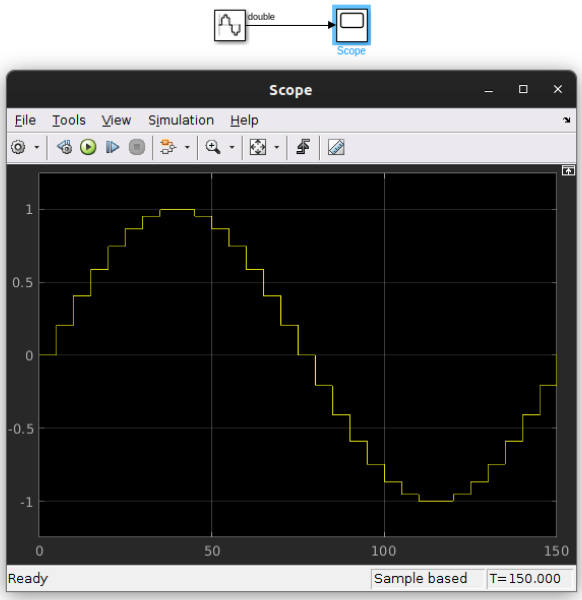

Figure 6: Part 2, Normal Sine Output

Finally, the following model was configured to obtain a system characterized by

$$y[n] = x[n] + \frac{1}{4}y[n-1]$$

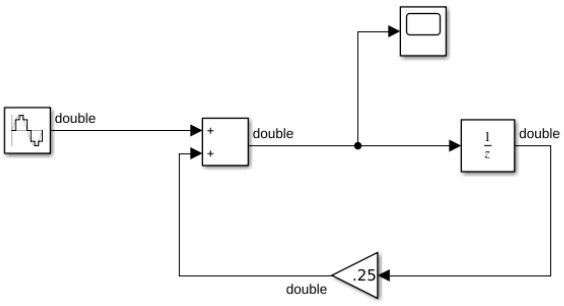

Figure 7: Part 2, Difference Equation Model

The output was tested under two conditions. First, figure 8 below shows the output with the same parameters as before (i.e. frequency of $\frac{2\pi}{150}$ and sample time of 5).

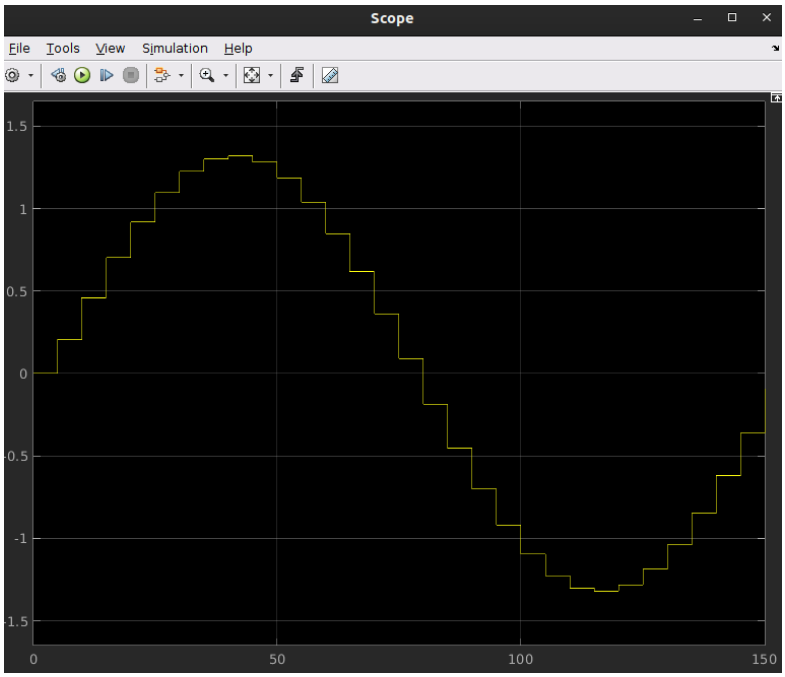

Figure 8: Part 2, Difference Equation Output 1

Finally, figure 9 below shows the output when the sample time was changed to 1.

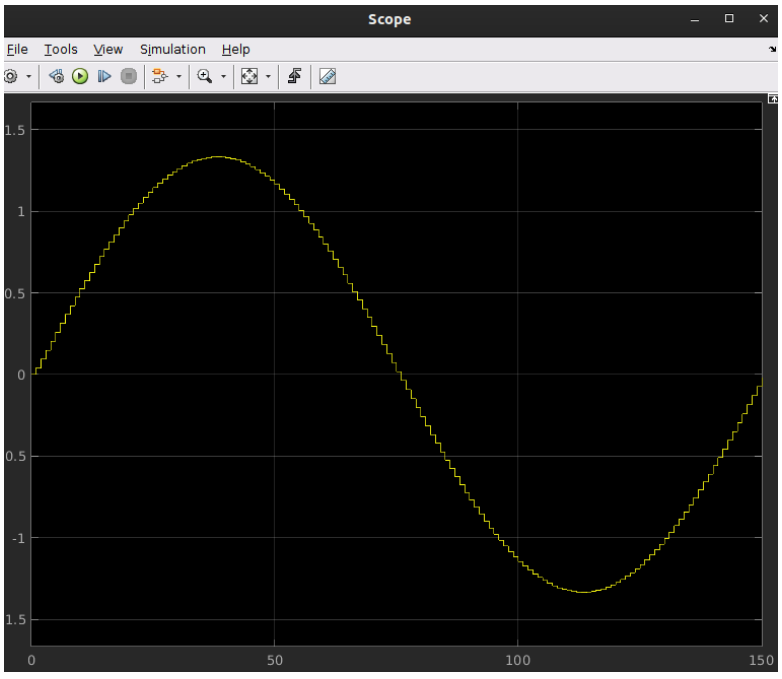

Figure 9: Part 2, Difference Equation Output 2

4 Conclusion

Experiment 3 provided us with further MATLAB tools to help analyze and experiment with discrete time functions. Specifically, the computation of the Discrete Time Fourier Transform as well as the use of Simulink were explored. First, the use of the MATLAB fft function was compared with the manual computation of the discrete time Fourier transform. Then, Simulink was used to experiment with a discrete time model and observe the output on a simulated oscilloscope. All in all, experiment three helped to further solidify MATLAB knowledge while introducing new concepts all the same.

5 Appendix

a. Part 1: Q1 Script

```
% Author:   Adrian Patterson
% ID:       40048841
% ELEC 342, Lab Section UQ
% Lab 3, Part 1 Question 1
clc;
clear;
close all;

n = (-10:1:10);     % defining the interval
x = zeros(1,length(n)); % defining x
    x(9:13) = 1;

step = input("Enter desired step size for w: ");% allows user to input step
w = (-pi:step:pi);          % consequently creates w interval from input
FT = zeros(1,length(w));    % initializing fourier transform array

for h = 1:length(w)         % for loop using length function
   sum = 0;
   for k = 1:length(n)
       sum = sum + (x(k)*exp(-1i*w(h)*n(k)));  % fourier transform is computed
   end
   FT(h) = sum;
end
FT_mag = abs(FT);           % here we get the magnitude using the abs() function

subplot(1,2,1)              % plots
title("x[n]");
    hold on;
    stem(n,x);
    xlabel("n");
    ylabel("x[n]");

subplot(1,2,2)
title('$$|X(e^{jw})|$$','interpreter','latex');
    hold on;
    stem(w,FT_mag);
    xlabel("w");
    ylabel('$$|X(e^{jw})|$$','interpreter','latex');
```

b. Part 1: Q2 Script

```
% Author:   Adrian Patterson
% ID:       40048841
% ELEC 342, Lab Section UQ
% Lab 3, Part 1 Question 2
clc;
clear;
close all;

n = (-10:1:10);     % defining the interval
x = zeros(1,length(n)); % defining the input x
```

```
    x(9:13) = 1;

w = (0:0.3:2*pi);           % we need 2*pi/21 = 0.3 step size
FT = zeros(1,length(w));    % initializing fourier transform array

for h = 1:length(w)
   sum = 0;
   for k = 1:length(n)
       sum = sum + (x(k)*exp(-1i*w(h)*n(k)));   % computing the fourier transfor
   end
   FT(h) = sum;
end
FT_mag = abs(FT);       % getting the magnitude of FT

FFT = fft(x);           % computing fourier transform using fft() function
FFT_mag = abs(FFT);     % getting the magnitude of FFT

subplot(1,2,1)          % plots
title('$$|X(e^{jw})| \ by \ Computation$$','interpreter','latex');
    hold on;
    stem(w,FT_mag);
    xlabel("n");
    ylabel("x[n]");

subplot(1,2,2)
title('$$|X(e^{jw})| \ by \  using \ fft(x)$$','interpreter','latex');
    hold on;
    stem(w,FFT_mag);
    xlabel("w");
    ylabel('$$|X(e^{jw})|$$','interpreter','latex');
```

c. Part 1: Q3 Script

```
% Author:   Adrian Patterson
% ID:       40048841
% ELEC 342, Lab Section UQ
% Lab 3, Part 1 Question 3
clc;
clear;
close all;

n = (-10:1:10);     % defining the interval
x = zeros(1,length(n)); % defining the input x
    x(9:13) = 1;

FFT = fft(x);       % taking fourier transform of x
IFFT = ifft(FFT);   % taking inverse fourier transform of FFT

subplot(1,2,1)      % plots
title('$$x[n]$$','interpreter','latex');
    hold on;
    stem(n,x);
    xlabel("n");
    ylabel("x[n]");

```

```
23  subplot(1,2,2)
24  title('$$x[n] = F^-1\{X(e^{jw})\}$$','interpreter','latex');
25      hold on;
26      stem(n,IFFT);
27      xlabel("n");
28      ylabel("x[n]");
```