Lab Two: Discrete-Time Fourier Transform

Digital Signal Processing Laboratory

Matthew Bohr

Dr. Pearlstein

17 October 2022

Student	Analysis	Development	Coding	Results	Writing
Matthew Bohr	100%	100%	100%	100%	100%

Introduction

In this lab, the students gained insight into the relationships between the DTFT, DFT, and FFT. The DTFT takes a discrete-time signal and converts it to a continuous signal in the frequency domain. This mirrors the FT which take a continuous-time signal and converts it to a continuous signal in the frequency domain.

This lab is concerned with digital signal processing so it would be useful to sample the DTFT at uniformly spaced frequencies to prepare it to be stored in a finite sized digital memory block. The DFT does just that—samples the DTFT at uniformly spaced frequencies, where the spacing between angular frequencies is $2\pi/N$. The FFT is just an algorithm to compute the DFT very quickly.

Procedure

In part one of this lab, the students were given the following continuous-time signal:

$$x(t) = 0.95^t, t \geq 0.$$

The exponential signal was sampled at a rate of $f_s = 1000\ Hz$ and then plotted. The DTFT was first derived analytically using pen and paper, then it was sampled at uniform spaces of $\frac{2\pi}{1024}$ to produce the DFT, and finally the result was plotted on the interval $[0, 2\pi]$. After plotting the DFT, the FFT was also computed using the same number of samples. Since the DFT is just the DTFT sampled at uniformly spaced frequencies, and the FFT is just a quick algorithm for computing the DFT, then the graph of the sampled DTFT and the FFT should have the same shape.

In the next part of the lab, the students needed to superimpose multiple sinusoidal sequences, plot the results, and plot multiple variations of the FFT. This included FFTs with different numbers of samples as well FFTs on the real and imaginary components. Producing the graphs enabled the students to identify the unique frequency components constituting the signal.

Results

The students first needed to plot the exponential sequence for 200 samples. The code is shown in Appendix A and it is very simple. It consists of two vectors, one for storing the indexes and one for storing the value of the exponential at each index. Figure 1 shows the plot of the exponential sequence.

The shape of the plot is the expected exponential decay since the $|\alpha| = 0.95 < 1$.

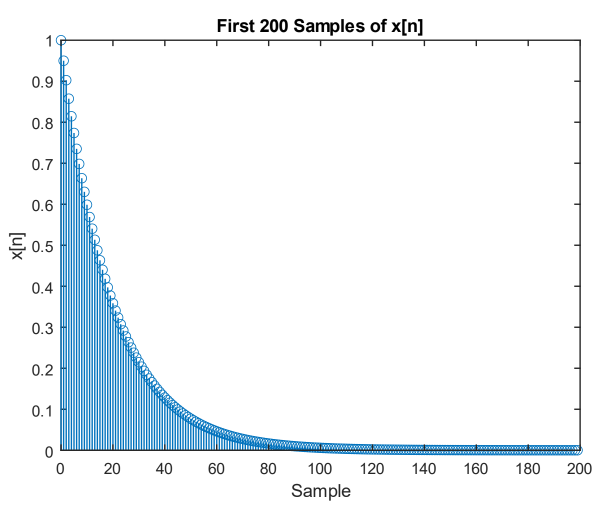

Figure 1: Plot of Exponential Sequence

The next part of the lab required the students to compute an analytical expression of the DTFT of the exponential by hand:

$$F\{x[n]\} = X\left(e^{jw}\right) = \sum_{n=-\infty}^{\infty} x[n]e^{-jwn}$$

We can write the exponential function as:

$$x[n] = \alpha^n \mu[n]$$

Plugging x[n] into the DTFT expression yields:

$$\sum_{n=-\infty}^{\infty} \alpha^n \mu[n]e^{-jwn} = \sum_{n=0}^{\infty} \alpha^n e^{-jwn} = \sum_{n=0}^{\infty} (\alpha e^{-jw})^n = \frac{1}{1-\alpha e^{-jw}}$$

$$X\left(e^{jw}\right) = \frac{1}{1-\alpha e^{-jw}} \Leftrightarrow x[n] = \alpha^n \mu[n]$$

We cannot plot continuous signals in MATLAB as that would require an infinite amount of memory to store the infinite number of points from the DTFT. Therefore, we must sample the DTFT at uniformly spaced points before plotting, which is equivalent to finding the DFT. Figure 2 shows the graphs of the DFT and FFT and, as expected, they are the same.

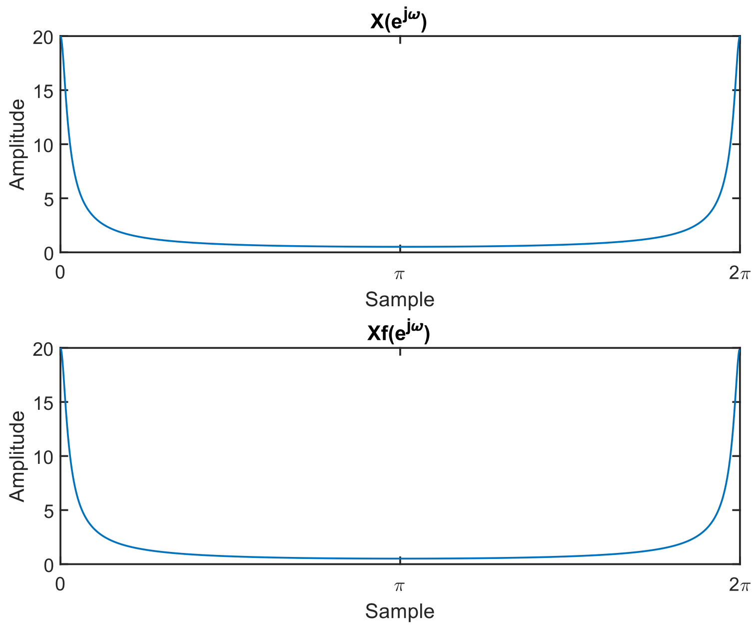

Figure 2: DFT vs FFT

To explain how this was achieved in MATLAB, let's first define the DFT:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}n}, k = 0,1, \dots, N-1.$$

We can plug in $x[n] = \alpha^n$ to produce the DFT signal that was plotted in the first graph of Figure 2.

$$X[k] = \sum_{n=0}^{N-1} \alpha^n e^{-j\frac{2\pi}{N}nk} = \sum_{n=0}^{N-1} (\alpha e^{-j\frac{2\pi}{N}k})^n = \frac{1-(\alpha e^{-j\frac{2\pi}{N}k})^N}{1-\alpha e^{-j\frac{2\pi}{N}k}} \approx \frac{1}{1-\alpha e^{-j\frac{2\pi}{N}k}}$$

The DFT is extremely similar to the DTFT except the continuous frequency w is replaced with a discrete frequency bin ($\frac{w}{2\pi} = f = \frac{k}{N}$). In the MATLAB code shown in Appendix B, this sampling mechanism is almost automatically achieved just by the nature of how we set the DTFT up. The frequency response of the DTFT is inputted into MATLAB as a function of w. However, w is a vector such that $w = 0{:}\frac{2\pi}{1024}{:}1023$, thereby absorbing $\frac{2\pi}{N}k$ from the denominator of the DFT into a single operation. So, although it appears that the DTFT equation is used in MATLAB, the discrete vector w converts it to the DFT.

In the next part of the lab, the students were to superimpose and graph the following sinusoidal sequences: $y_1[n] = \sin\left(\frac{2\pi n}{N}\right)$, $y_2[n] = \cos\left(\frac{4\pi n}{N}\right)$, $y_3[n] = \cos\left(\frac{22\pi n}{N}\right)$, and $y_4[n] = \cos\left(\frac{202\pi n}{N}\right)$. The graph is shown in Figure 3.

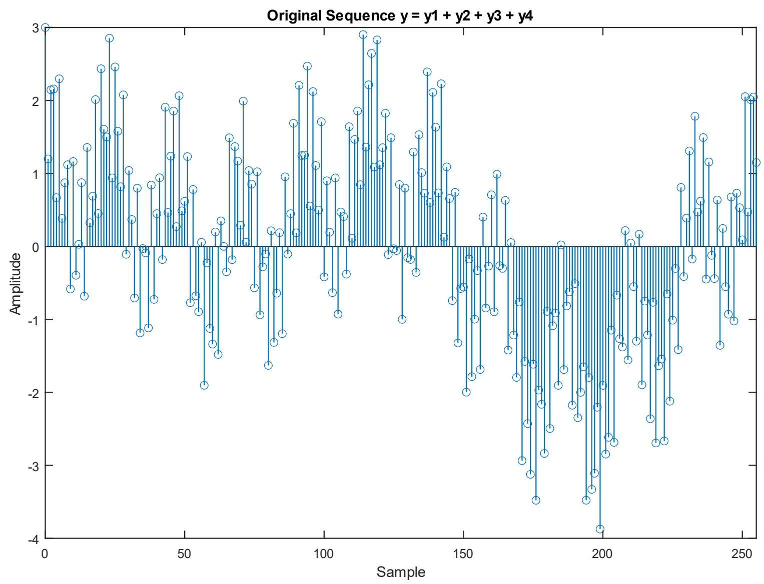

Figure 3: Superimposing Multiple Sinusoidal Sequences

The resulting sequence looks like what one would expect from overlapping sinusoidal sequences. A 255 point FFT, 128 point FFT, an FFT of the real components, and an FFT of the imaginary components are shown in Figure 4.

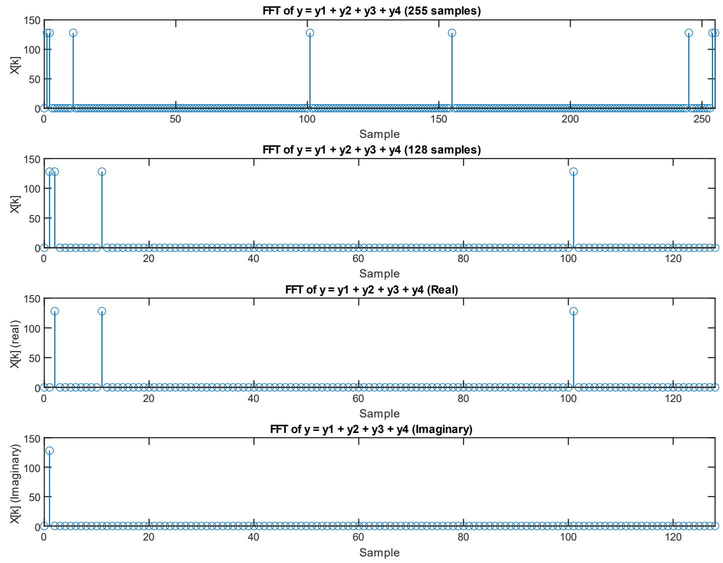

Figure 4: Multiple Variations of FFT Applied to Signal

Figure 4 produces a few interesting results. One thing to recall is that a discrete sinusoid's frequency can be written as follows:

$$2\pi k$$

Where k is the frequency bin. The frequency of each sinusoid is easily found: $k_1 = 1, k_2 = 2, k_3 = 11$, and $k_4 = 101$. In the first graph of Figure 4, there are spikes at $k = 1, k = 2, k = 11$, and $k = 101$ which makes sense since these are the frequencies of $y_1[n], y_2[n], y_3[n]$, and $y_4[n]$ respectively. However, we can also see spikes at $k = 155, k = 245, k = 254$, and $k = 255$. The first graph of Figure 4 is symmetrical about the 127$^{\text{th}}$-128$^{\text{th}}$ index of n. This is unusual since it appears as if additional frequency components that do not exist overlap with our signal.

It seems as if only the first half of the FFT contains any unique or new information, so we graphed the FFT using only 128 samples as well so that each sinusoid is represented exactly once.

The FFT of the real part shown in the third graph of Figure 4 just wipes out the spike at k = 1 since this frequency corresponds to a sine wave which is an imaginary signal. On the other hand, the FFT of the imaginary part shown in the fourth graph of Figure 4 wipes out the spikes at k = 2, 11, and 101 since these correspond to cosine waves which are real signals. The code for this section is in Appendix B.

The students were then required to plot the DFT and FFT of an additional sinusoid: $y_5[n] = \sin\left(\frac{23\pi n}{N}\right)$. Recall that we can write the following expression for $y_5[n] = \sin\left(\frac{23\pi n}{N}\right)$:

$$2\pi k_5 = 23\pi$$

$$k_5 = 11.5$$

This indicates that we should expect a spike at around k = 11.5. Appendix C shows the code used to compute the DFT and FFT. The results are shown below in Figure 5.

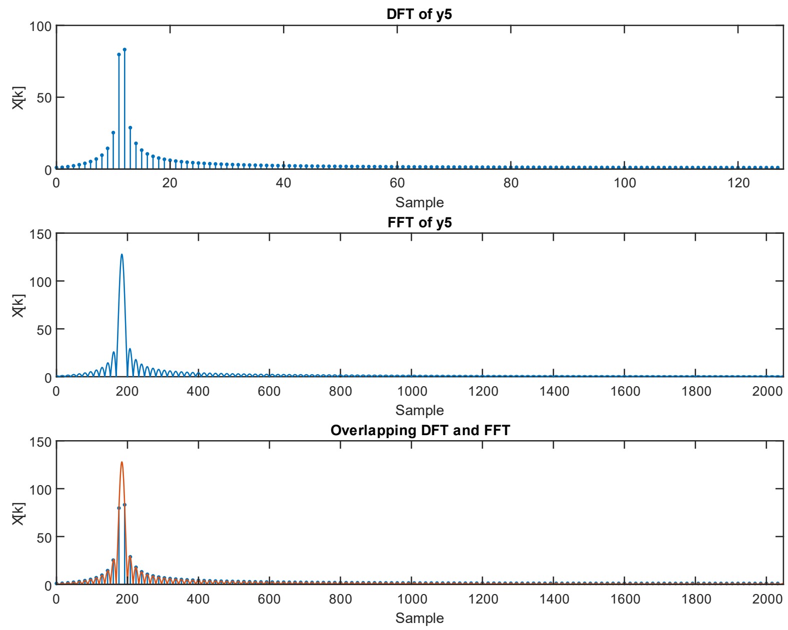

Figure 5: DFT and FFT of Signal with Non-Integer Number of Cycles

Plotting the DFT of this signal yields spikes hovering around the k = 11.5 frequency bin as shown in the first graph of Figure 5. What and why are there unwanted spikes at unwanted frequencies in what can be described as a smearing effect? The reason for this is because the signal has a frequency of 11.5 Hz meaning it will oscillate 11.5 times in N samples, so the DFT will not be on a purely sinusoidal sequence since we have half a period of samples included in the DFT. This contrasts with the DFT from part 7 in which all the signals oscillate an integer number of times in N samples, thereby producing clean spikes in the graphs.

The FFT was also computed using 16 as many points. Plotting the FFT yields a graph that appears continuous and resembles the shape of the DFT computed in the previous step. Since we are using 16 as many points to compute the FFT, the spike hovers around the k = 11.5*16 = 184 frequency bin.

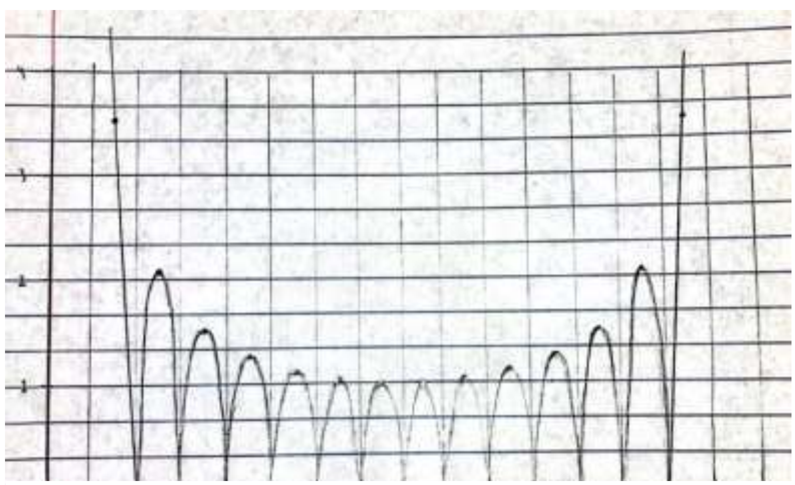

Figure 5: Window Frequnecy Domain

The second graph in figure 5 corresponds to this. Lastly, the DTF was overlaid with the FFT, now plotting the DTF every 16 iterations and increasing the total number of indexes by a factor of 16 to match the factor introduced with the FFT. Lastly, a hand drawing of the frequency window function is shown in Figure 5.

Conclusion

In this lab the students gained knowledge using the DTFT, DFT, and FFT. We learned about the impact on the DFT of not having an integer number of cycles within the samples and how this occurrence produces a smearing effect making it seem as if multiple frequencies exist around a given frequency.

Appendix

Appendix A: Code for Parts 1-2

```
close all
clear

%defining sampling frequency and constant for exponential signal
fs = 1000;
Ts = 1/fs;
alpha = 0.95;

%defining samples and output sequence
n = 0:1:199;
x = alpha.^n;

%plotting
stem(n, x)
title('First 200 Samples of x[n]')
```

Appendix B: Code for Parts 3-4

```
close all
clear

%defining sampling frequency and constant for exponential signal
fs = 1000;
Ts = 1/fs;
alpha = 0.95;

%defining samples and sequence
n = 0:1:199;
x = alpha.^n;

%computing the DFT using the FFT
Xf = fft(x, 1024);

%sampling the DTFT of the exponential signal
w = 0:2*pi/1024:2*pi;
h = 1 ./ sqrt( (1-alpha*cos(w)).^2 + (alpha*sin(w)).^2 );

%plotting DTFT when sampled every 1/1024 of its period
subplot(2,1,1)
plot(w , h)
title('X(e^{j\omega})')
xticks([-3*pi -2*pi -pi 0 pi 2*pi 3*pi])        % set ticks to be
pi-ish
```

```
xticklabels({'-3\pi','-2\pi','-\pi','0','\pi','2\pi','3\pi'})   %
pi-ish labels
ylim([0 20])                                              % set vertical
axis limits
xlim([0 (2*pi)])                                        % set horizontal
axis limits

%plotting the DFT using the FFT algorithm
subplot(2,1,2)
plot((0:1:1023)*2*pi/1024, abs(Xf))
xticks([-3*pi -2*pi -pi 0 pi 2*pi 3*pi])         % set ticks to be
pi-ish
xticklabels({'-3\pi','-2\pi','-\pi','0','\pi','2\pi','3\pi'})   %
pi-ish labels
title('Xf(e^{j\omega})')                                % set title
ylim([0 20])                                              % set vertical
axis limits
xlim([0 (2*pi)])                                        % set horizontal
axis limits
```

<u>Appendix C: Code for Parts 5-9</u>

```
clear
close all

%defining number of samples
N = 256;
n = 0:1:255;

%defining each sinusoidal sequence
y1 = sin((2*pi*n)./N);
y2 = cos((4*pi*n)./N);
y3 = cos((22*pi*n)./N);
y4 = cos((202*pi*n)./N);
y5 = cos((23*pi*n)./N);

%superimposing sinusoids y1, y2, y3, and y4
y = y1 + y2 + y3 + y4;

%computing the FFT of the signal , as well as the FFT of the real and
%imaginary componenets
yf = fft(y, 256);
yf_real = fft(y2+y3+y4, 256);
yf_imag = fft(y1, 256)

%plotting FFT og the single sinusoid.
Fy = fft(y5);
subplot(3,1,1);
stem(0: (N/2-1), abs(Fy(1:N/2)), 'filled', 'MarkerSize', 2)
title('DFT of y5')
xlim([0 128])
```

```
xlabel('Sample')
ylabel('X[k]')

%increasing the sampling rate of FFT applied to y5 by 16
Fy_16 = fft(y5, N*16);
subplot(3,1,2);
plot(0: (16*N/2-1), abs(Fy_16(1:16*N/2)))
title('FFT of y5')
xlim([0 N*16/2])
xlabel('Sample')
ylabel('X[k]')

%overlaying previous two graphs
subplot(3,1,3);
stem(0:16:(16*N/2-1), abs(Fy(1:N/2)), 'filled', 'MarkerSize', 2)
hold on
plot(0: (16*N/2-1), abs(Fy_16(1:16*N/2)))
hold off
title('Overlapping DFT and FFT')
xlim([0 N*16/2])
xlabel('Sample')
ylabel('X[k]')
```