ELC 433-L1

Lab 3 – Application of DFT/FFT to Analyze Signals, and Explore Transform Properties

Brian Worts and Chris Jenson

10/23/20



**Introduction:**

This lab used the concepts from the previous labs to gain experience in applying the DFT/FFT for analyzing deterministic signals and exploring the properties of the DFT. The background for this lab is that the DFT transforms a length N sequence of samples into a length N vector in the frequency domain. The FFT is an implementation of the DFT that requires multiplications to compute. When given the sample rate, the X[k] component can represent the continuous-time frequency. Using these concepts, the inverse DFT or the IDFT can be found. Some fundamentals of music are used. There are 12 semitones in an octave, and they are geometrically spaced. That can be used to create a relationship. The combination of knowledge about DFT/FFT and music basics are used to complete this lab and further the student’s understanding of the material.

**Procedure:**

The lab begins by acquiring the la\_la\_land.m4a file from Canvas. The file is read into MATLAB using the provided line of code. Once entered, the first column is extracted. The waveform is plotted and listened to. Now with the waveform, a sizable segment of the recording that appeared to have good quality was extracted to a new signal.

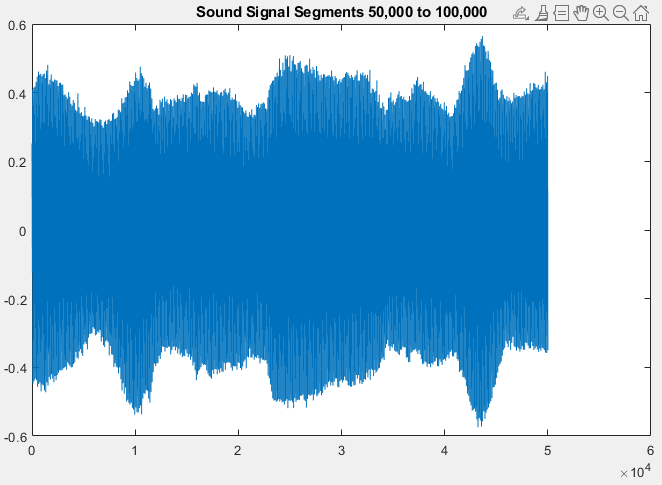
Step 5 was to create a plot with 3 subplots. The first plot was gained by taking the FFT of the extracted piece of the first column. It had to be considered that the first index in the array is a 0, not a 1. This meant that the x-axis had to be defined. The second subplot was the same plot as the first, but only the left half of the FFT. It was done by dividing the length by 2. For the third subplot, the first plot was analyzed to find the component at the fundamental frequency and then plotting that component specifically. This was done to find the array value of the peak. The array index was then converted to frequency.

Step 6 was to analyze time-varying signals using an STFT by taking the FFT’s of overlapping segments. To do this, three parameters had to be defined. The first of these was SEGMENT\_LENGTH. It should span cycles at the frequency of interest and be as large as the period of the desired resolution frequency. A base value of 1024 was used. The second parameter was OFFSET\_PER\_SEGMENT. It was about half of SEGMENT\_LENGTH. The final parameter was FFT\_LENGTH. This parameter is driven by the frequency resolution needs and so that the continuous-time frequency of one FFT index is smaller than the desired frequency resolution. The value 4096 was used as the baseline. A loop was created where each cycle through the loop computed one FFT segment and assembled an array of coefficients. It was then plotted as a 3D plot. Step 7 was to do Step 6 but in an easy way using the spectrogram function.

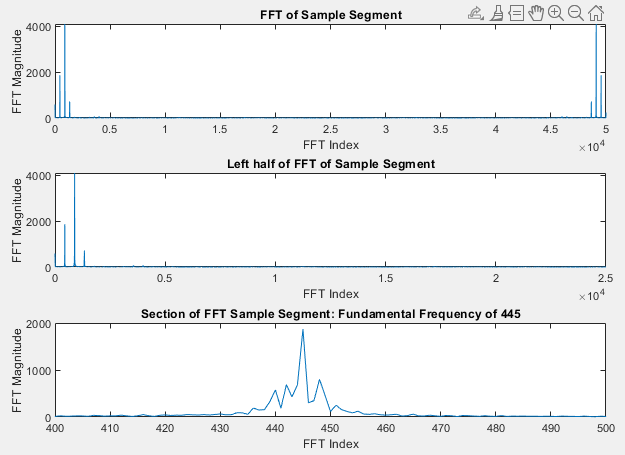
Step 8 was to select and demonstrate two properties of the DFT. The two properties chosen were the inverse transform and the relationship between time shift and linear phase shift.

**Results:**

Part 4: Plot of the waveform



Part 5: Plot with 3 subplots

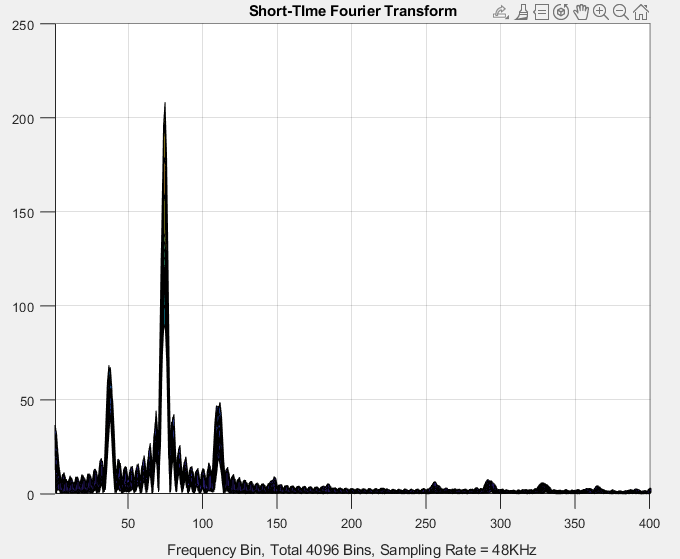


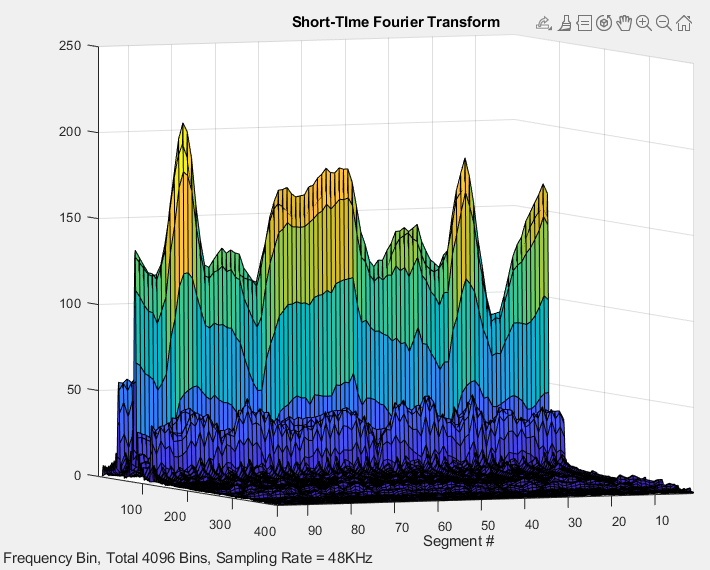
Conversion of the bin index to frequency



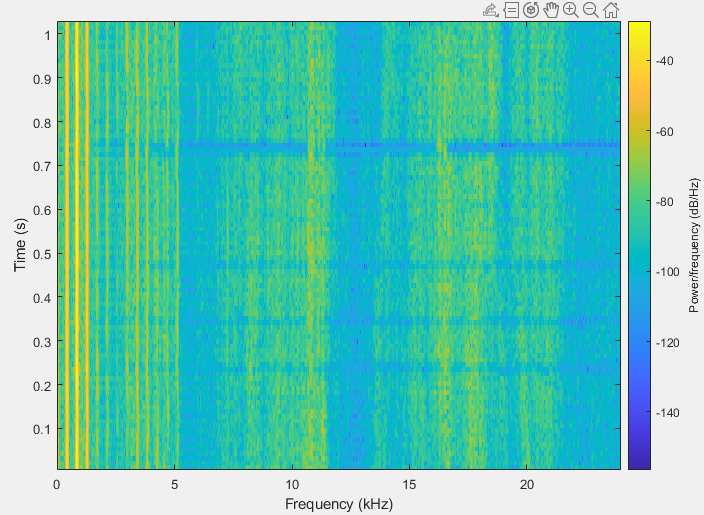


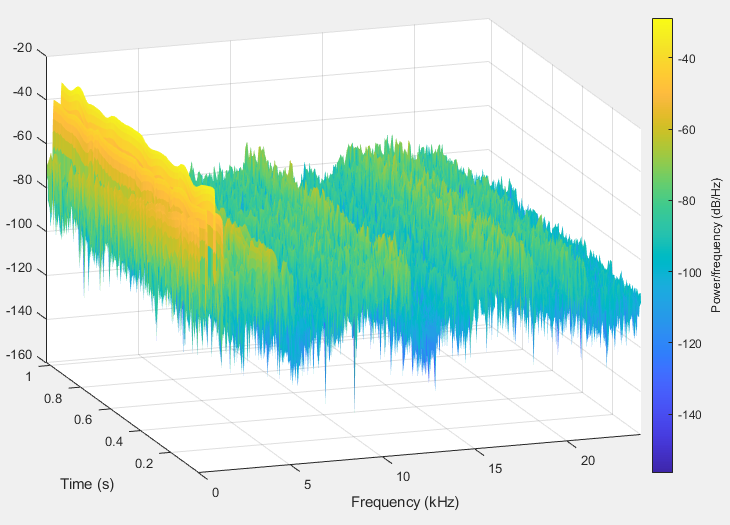
Part 6: Plot of the 2-dimensional Array





Part 7: Spectrogram Visualization

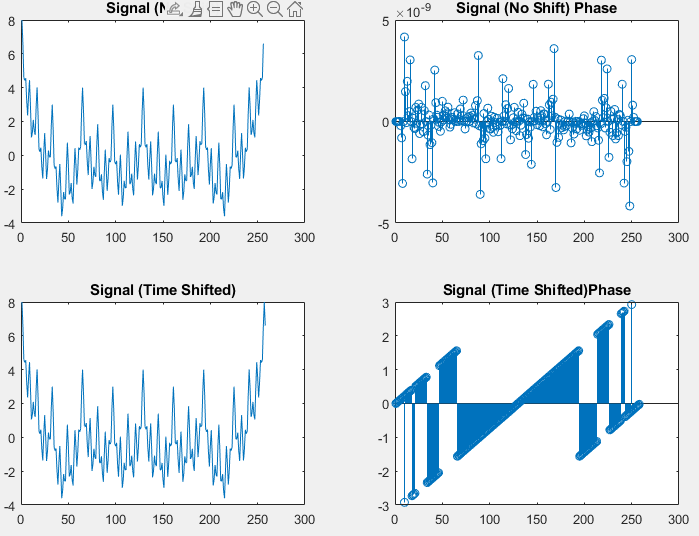




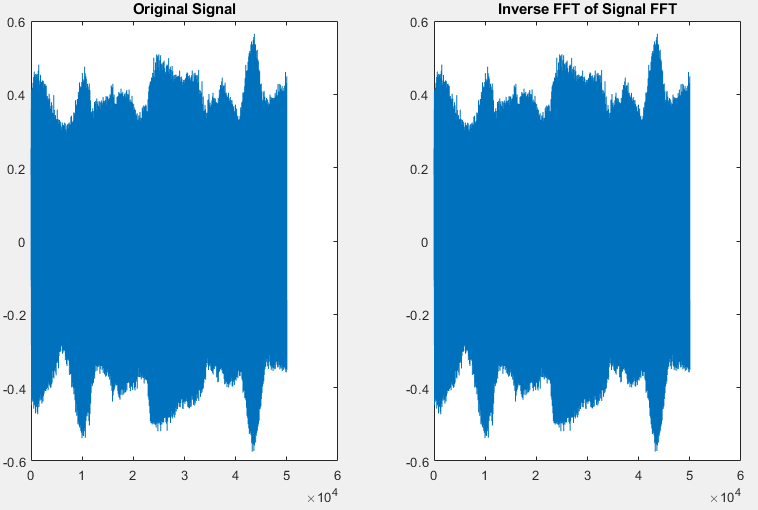
Part 8:

Property 1: Time Shift ⬄ Linear Phase Shift

OK – how much was the time shift? What slope in phase would you expect?



Property 2: Inverse Transform



**Engineering Work:**

%BRIAN WORTS AND CHRIS JENSON

%ELC 433-L1

%LAB 3

clear

close all

%Part 3

[y, Fs] = audioread('E:/College Work/CurSem/SignalsLab/lab3/la\_la\_land.m4a');

%Part 4

y = y(:,1); % comments

lenSig = length(y); % comments

%sound(y(50000:100000),Fs) %Signal, sample freq

signalSegment = y(50000:100000);

figure(1)

plot(signalSegment);

title('Sound Signal Segments 50,000 to 100,000');

%Part 5a

figure(2)

subplot(3,1,1);

sigFFT = abs(fft(signalSegment));

%Starts graphing at 0, not 1

plot(0:length(sigFFT)-1, sigFFT) % comments

title('FFT of Sample Segment')

xlabel('FFT Index')

ylabel('FFT Magnitude')

%Part 5b

subplot(3,1,2);

plot(0:length(sigFFT)-1, sigFFT) %Plot whole signal

xlim([0 length(sigFFT)/2])

title('Left half of FFT of Sample Segment')

xlabel('FFT Index')

ylabel('FFT Magnitude')

%Part 5c

fundFreq = sigFFT(1);

subplot(3,1,3);

plot(0:length(sigFFT)-1, sigFFT) %Plot whole signal

xlim([400 500])

title('Section of FFT Sample Segment: Fundamental Frequency of 445')

xlabel('FFT Index')

ylabel('FFT Magnitude')

Fk = 445/length(sigFFT)\*Fs %Equation 3. = 427.1915

%From inspection, fundamanetal freq is 445 since peaks are at multiples of 445.

%Part 6

%the lowest frequency of interest is about 200 Hz,

%and the desired frequency resolution is about 5 Hz

SEGMENT\_LENGTH = 1024;

OFFSET\_PER\_SEGMENT = 512;

FFT\_LENGTH = 4096;

idx = 1; % comments

for i=1:OFFSET\_PER\_SEGMENT:length(signalSegment)+1-SEGMENT\_LENGTH % comments

data = signalSegment(i:i+SEGMENT\_LENGTH-1); %DEBUG should this be fft?

data\_fft = abs(fft(data, FFT\_LENGTH)); % comments

coeffs(:,idx) = data\_fft;

idx = idx + 1;

end

figure(3)

surf(coeffs)

xlim([1 idx])

ylim([1 400])

xlabel('Segment #')

ylabel(sprintf('Frequency Bin, Total %d Bins, Sampling Rate = %dKHz',FFT\_LENGTH, Fs/1000))

title('Short-TIme Fourier Transform');

%Part 7

figure (4)

nooverlap = OFFSET\_PER\_SEGMENT; % comments

spectrogram(signalSegment,SEGMENT\_LENGTH,nooverlap,FFT\_LENGTH,Fs); % comments

%Part 8

%a)

figure (5)

N = 256;

for n = 0:(N-1) % comments

y1(n+1) = cos(2\*pi\*n/N);

y2(n+1) = cos(4\*pi\*n/N);

y3(n+1) = cos(8\*pi\*n/N);

y4(n+1) = cos(16\*pi\*n/N);

y5(n+1) = cos(32\*pi\*n/N);

y6(n+1) = cos(64\*pi\*n/N);

y7(n+1) = cos(128\*pi\*n/N);

y8(n+1) = cos(6\*pi\*n/N);

end

y = y1+y2+y3+y4+y5+y6+y7+y8; % comments

for n = 3:(N+1) %Time shifted

y1(n+1) = cos(2\*pi\*n/N);

y2(n+1) = cos(4\*pi\*n/N);

y3(n+1) = cos(8\*pi\*n/N);

y4(n+1) = cos(16\*pi\*n/N);

y5(n+1) = cos(32\*pi\*n/N);

y6(n+1) = cos(64\*pi\*n/N);

y7(n+1) = cos(128\*pi\*n/N);

y8(n+1) = cos(6\*pi\*n/N);

end

timeShifted = y1+y2+y3+y4+y5+y6+y7+y8; % comments

Well, unfortunately, this isn’t quite right – you step from 3 to 257, and assign indices 4 to 258. I don’t think that’s what you intended to do.

signal = y; %cos((-2\*pi:0.01:2\*pi));%cos((-2\*pi:0.01:2\*pi));

subplot(2,2,1)

plot(signal);

title('Signal (No Shift)')

subplot(2,2,2)

stem(angle(fft(signal)+0.0001)); %Need to add very small number to phase since essentially 0

title('Signal (No Shift) Phase')

subplot(2,2,3)

plot(timeShifted);

title('Signal (Time Shifted)')

subplot(2,2,4)

stem(angle(fft(timeShifted)+0.0001));

title('Signal (Time Shifted)Phase')

What do we expect?

Expect a slope of , in other words 2 full cycles over 256 samples.

But we only see half that. Because of a bug in your code.

%b)

figure (7)

subplot(1,2,1)

plot(signalSegment);

title('Original Signal')

subplot(1,2,2)

plot(ifft(fft(signalSegment)));

%Take difference

title('Inverse FFT of Signal FFT')

**It’s a nice plot, but to verify that there is really a good match you’d want to be a little more thorough. Like measuring the maximum difference, overall elements in the vectors.**

**Knowledge Gained:**

The students both gained experience in applying the DFT and the FFT when analyzing deterministic signals. It was visualized what the effects were of the FFT on a sample piece of audio. When trying to find the fundamental frequency, the highest peak was originally thought to be the answer. Further inspection revealed that it was the common sub-multiple of the peaks observed. The initial attempt at Step 6 had a mistake where the code was not actually iterating but instead doing the same calculation repeatedly, causing it to take an extremely long time. This was fixed by changing the loop parameters, removing a useless outer loop, and properly offsetting the waveform. The spectrogram function was used which showed how a quick visual inspection can also be utilized to gain a better understanding of the components in a signal. Another issue the team faced was in part 8a when trying to demonstrate the time shift ⬄ linear phase shift property of the DFT. The FFT of the signal the team created was resulting in values too close to 0 causing MATLAB to not get the phase. To solve this the team added a value of 0.0001 to the FFT before getting the angle, thus showing the linear time shift property. The value and utility of the DFT was also reinforced by demonstrating multiple of its properties. In addition to this, the students' existing understanding of various MATLAB functionalities was increased.

**Who Did What:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Student** | **Analysis** | **Development** | **Coding** | **Results** | **Writing** |
| Brian | 50 | 50 | 75 | 50 | 25 |
| Chris | 50 | 50 | 25 | 50 | 75 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Expectation** | **Max Pts.** | **Pts. Deducted** |
| Introduction | Brief overview | 1 |  |
| Procedure | Brief description of procedures | 1 |  |
| Design/Engineering Work |  |  |  |
|  | Mathematical analysis and code are correct. Illustrate two properties of FFT. | 2 | 0.3 |
|  | Paste text of code, not images | 0.5 |  |
|  | Use SPACE not TAB characters. Strict indentation. Consistent placement of IF, ELSE, FOR, END | 0.5 |  |
|  | Full commenting | 1 | 0.2 |
| Results |  |  |  |
|  | Explain all conversions between FFT result to continuous time frequencies | 0.5 |  |
|  | Plot with 3 subplots. First bin is Bin 0. Full FFT, left half, zoom in. | 0.5 |  |
|  | Accurately identify the bin index of the peak. Remember to use ‘plot( x, y )’ syntax, with the x-axis appropriately set, and then xlim will show the result with correct numbers for the x-axis. Record your value for the bin index of the fundamental frequency. Convert that bin index to frequency. | 0.5 |  |
|  | STFT - the hard way, and spectrogram | 0.5 |  |
| Knowledge Gained |  | 1 |  |
| Who Did What |  | 1 |  |
| **Total** |  | 9.5 |  |