***AT&T Touch Tone***

ECE 439 Digital Signal Processing

Final Project

Joseph Hilland

December 10, 2014

**Table of Contents**

Introduction 3

Basic Problems 4

Intermediate Problems 6

Advanced Problems 10

Discussion 11

Conclusion / results 12

source 12

Works Cited 16

## Introduction

1. Background

Before beginning this project, I needed to have a firm understanding of the Discrete Time Fourier Transform (DTFT). The DTFT is a form of Fourier analysis. However, the transform operates on discrete data. Fourier analysis refers to the study of the way that general functions may be represented or approximated by sums of simpler trigonometric functions.

Once I felt that I understood the general concepts of the DTFT (also from ECE 314 in the past), I then needed to ensure that I knew how to take the DTFT of a signal within Matlab, and how that would benefit me. The homework document specified that, “touch-tone system on a telephone uses signals of different frequencies to indicate which key has been pushed.” The DTFT of a sampled telephone signal could be used to identify these specific frequencies.

Luckily, these frequencies were given in a table within the assignment document. This table can also be viewed in the data section of this document. Lastly, I found that I would be utilizing the built in fft Matlab function which could be used to compute N samples of the DTFT of a finite-length signal at the frequency of

As a note, all source code is provided in the section that it applies to as well as the at the end of this report.

1. Given Data

|  |  |  |
| --- | --- | --- |
| Digit |  |  |
| 0 | .7217 | 1.0247 |
| 1 | .5346 | .9273 |
| 2 | .5346 | 1.0247 |
| 3 | .5346 | 1.1328 |
| 4 | .5906 | .9273 |
| 5 | .5906 | 1.0247 |
| 6 | .5906 | 1.1328 |
| 7 | .6535 | .9273 |
| 8 | .6535 | 1.0247 |
| 9 | .6535 | 1.1328 |

,

## Basic Problems

a.) Create Row Vectors

The first step in this assignment was to create a row vector that included digits zero through 9. I created these vectors using the given equation below.

Using this equation, row vectors were created for each digit, and I set n to equal the values from zero to 999. This gave me one thousand samples per digit.

Once this was completed, I then used the built-in Matlab function sound to listen to each signal. I also entered in a frequency of 8192, which was an assumption we made in the beginning when sampling the signal.

Below you can see the code that I created to accomplish these tasks. The sound functions are commented out so that way I would not have to hear them every time the script was run.

% ------------------------ A --------------------------------

N = 2048;

n = 0:999;

d0 = sin(0.7217\*n) + sin(1.0247\*n);

%sound(d0, 8192);

d1 = sin(0.5346\*n) + sin(0.9273\*n);

%sound(d1, 8192);

d2 = sin(0.5346\*n) + sin(1.0247\*n);

%sound(d2, 8192);

d3 = sin(0.5346\*n) + sin(1.1328\*n);

%sound(d3, 8192);

d4 = sin(0.5906\*n) + sin(0.9273\*n);

%sound(d4, 8192);

d5 = sin(0.5906\*n) + sin(1.0247\*n);

%sound(d5, 8192);

d6 = sin(0.5906\*n) + sin(1.1328\*n);

%sound(d6, 8192);

d7 = sin(0.6535\*n) + sin(0.9273\*n);

%sound(d7, 8192);

d8 = sin(0.6535\*n) + sin(1.0247\*n);

%sound(d8, 8192);

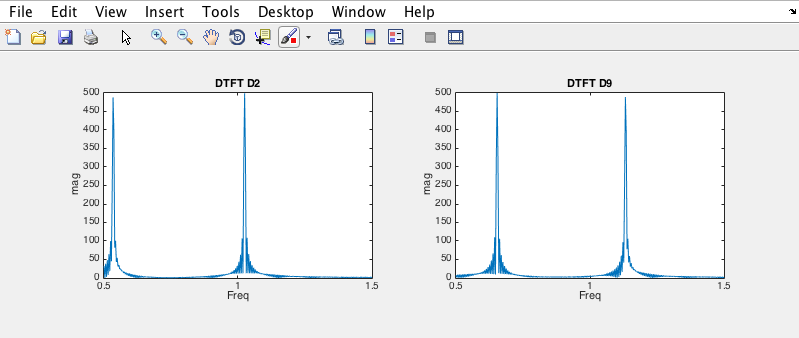
d9 = sin(0.6535\*n) + sin(1.1328\*n);

%sound(d9, 8192);

b.) DTFT of 2 and 9

For this section of the assignment, I was tasked with using the fft function for the keytones 2 and 9. Taking the fft of the signals for 2 and 9 would allow us to verify that the peaks fall at the frequencies specified in the DTFT chart. We would first need to take the DTFT of the signal, then calculate the magnitude of the DTFT and finally plot the results.

The graph below depicts the magnitude of the DTFT from 2 and 9. As you can see from the graph, the peaks fall at the appropriate frequencies specified by the given table.

Lastly you can see my source code under the graph. 

% ------------------------ B --------------------------------

% calculate the DTFT of digit 2 and digit 9

% then plot the magnitude and verify peak fall at correct frequency

D2 = fft(d2, N);

D9 = fft(d9, N);

k = 0:2047;

wk = (2 \* pi \* k) / N;

% plot for digit 2 DTFT

subplot(2,2,1), plot(wk, abs(D2));

title('DTFT D2');

axis([0.5, 1.5, 0, 500]);

xlabel('Freq'), ylabel('mag')

%subplot(2,1,1), plot(wk,angle(D2));

%xlabel('freq'), ylabel('phase')

% plot for digit 9 DTFT

subplot(2,2,2), plot(wk, abs(D9));

title('DTFT D9');

axis([0.5, 1.5, 0, 500]);

xlabel('Freq'), ylabel('mag')

%subplot(2,1,1), plot(wk,angle(D9));

%xlabel('freq'), ylabel('phase')

c.) touch-tone test

For this section I needed to complete three tasks. The first task involved created a row vector of one hundred zeros. This row vector would be used to simulate silence between each signal of a touch-tone. The next step was to define a variable “phone,” as my own phone number that contained the zero row vector between each touch-tone. Lastly I needed to listen to the variable phone and compare it to a touch-tone phone’s sound when dialing my number. After completing these tasks, I found that it sounded correct.

My source code can be seen below.

% ------------------------ C --------------------------------

space = zeros(1, 100);

phone = [d8 space d3 space d5 space d9 space d0 space d3 space d2 space];

%sound(phone, 8192);

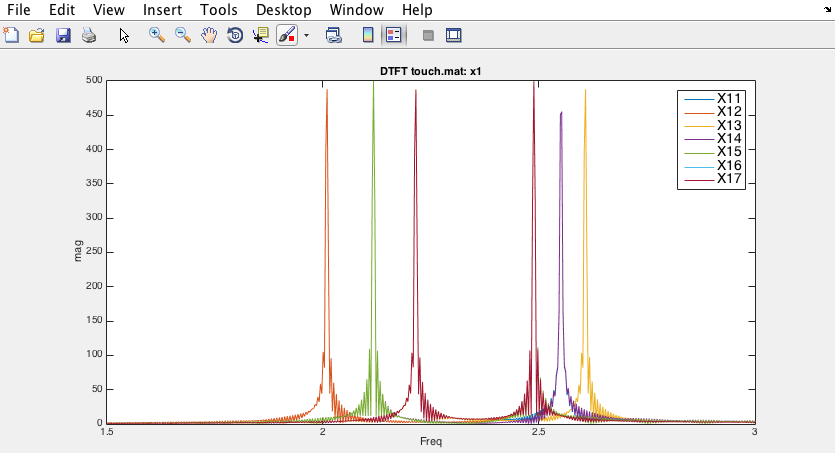
## Intermediate Problems

d.) touch.mat: x1

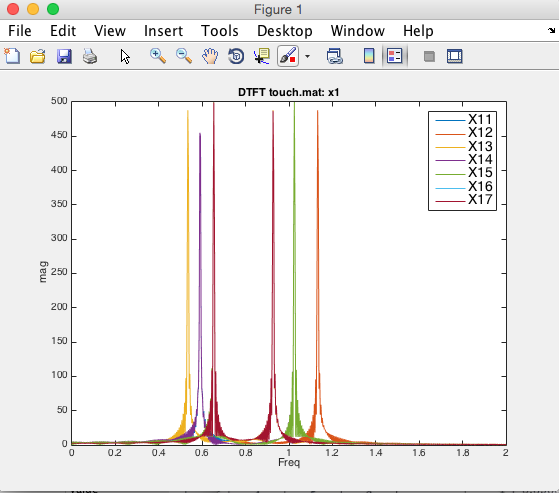
In this section of the assignment, I first needed to locate the touch.mat document that contained the variables hardx1, hardx2, x1, and x2. Once I located the file and loaded it into my script I was able to verify that these variables were there.

Next I needed to use what I had learned from previous sections and divide the given variable x1 into the appropriate sized signals. Each touch-tone would have 1000 sampled elements followed by 100 zeros that simulated silence.

I then parsed the x1 variable and separated all elements accordingly. Once this had been completed, I needed to run the fft and fftshift functions on each touch-tone digit. Then I plotted the final output so that I could guess the numbers being used within the x1 signal. The graph and source code can be seen below.



FFTSHIFT(above) FFT(below)



Adding these digits up we can see that it sums to 41 as required. The legend identifies which touch-tone belongs to which digit. The touch-tone order is 491 5877.

% ------------------------ D --------------------------------

% methods for x1

load('touch.mat');

x1\_1 = x1([1:1000]);

x1\_2 = x1([1100:2100]);

x1\_3 = x1([2200:3200]);

x1\_4 = x1([3300:4300]);

x1\_5 = x1([4400:5400]);

x1\_6 = x1([5500:6500]);

x1\_7 = x1([6600:7600]);

X11 = fftshift(fft(x1\_1,N));

X12 = fftshift(fft(x1\_2,N));

X13 = fftshift(fft(x1\_3,N));

X14 = fftshift(fft(x1\_4,N));

X15 = fftshift(fft(x1\_5,N));

X16 = fftshift(fft(x1\_6,N));

X17 = fftshift(fft(x1\_7,N));

plot(wk, abs(X11),wk, abs(X12), wk, abs(X13), wk, abs(X14), wk, abs(X15), wk, abs(X16), wk, abs(X17));

LEG = legend('X11', 'X12', 'X13', 'X14', 'X15', 'X16', 'X17');

set(LEG, 'FontSize', 14);

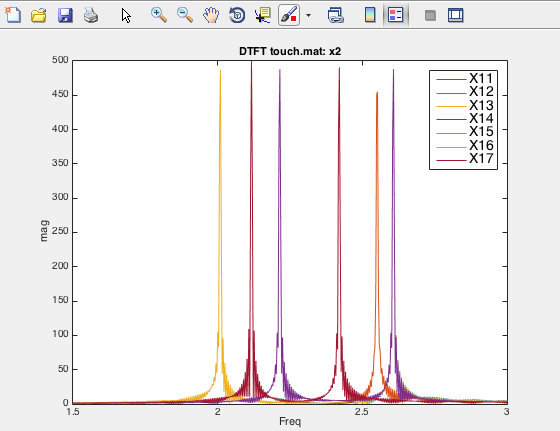
title('DTFT touch.mat: x1');

axis([1.5, 3, 0, 500]);

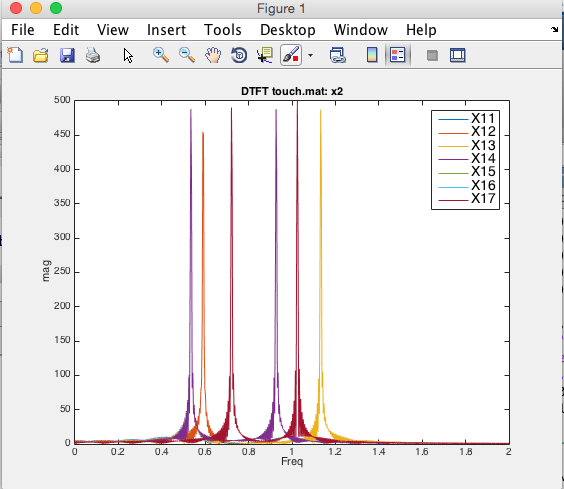
xlabel('Freq'), ylabel('mag')

e.) touch.mat: x2

This section involved the same steps as d.) however we used variable x2 rather than x1. Again the graph and code can be seen below.



FFTSHIFT(above) FFT(below)



Touch-tone is 253 1000

% ------------------------ E --------------------------------

% methods for x2

x2\_1 = x2([1:1000]);

x2\_2 = x2([1100:2100]);

x2\_3 = x2([2200:3200]);

x2\_4 = x2([3300:4300]);

x2\_5 = x2([4400:5400]);

x2\_6 = x2([5500:6500]);

x2\_7 = x2([6600:7600]);

X11 = fftshift(fft(x2\_1,N));

X12 = fftshift(fft(x2\_2,N));

X13 = fftshift(fft(x2\_3,N));

X14 = fftshift(fft(x2\_4,N));

X15 = fftshift(fft(x2\_5,N));

X16 = fftshift(fft(x2\_6,N));

X17 = fftshift(fft(x2\_7,N));

plot(wk, abs(X11),wk, abs(X12), wk, abs(X13), wk, abs(X14), wk, abs(X15), wk, abs(X16), wk, abs(X17));

LEG = legend('X11', 'X12', 'X13', 'X14', 'X15', 'X16', 'X17');

set(LEG, 'FontSize', 14);

title('DTFT touch.mat: x2');

axis([1.5, 3, 0, 500]);

xlabel('Freq'), ylabel('mag')

## Advanced Problems

f & g.) and corresponding k

This entire section involved completing steps that would eventually lead to the creation of a touch-tone decoder. First we needed to figure out which values and corresponding k values were closest to each of the touch-tone frequencies.

% ------------------------ F --------------------------------

w0 = 2\*pi/2048;

d0\_k = round(1 / w0 \* [0.7217 1.0247]);

d1\_k = round(1 / w0 \* [0.5346 0.9273]);

d2\_k = round(1 / w0 \* [0.5346 1.0247]);

d3\_k = round(1 / w0 \* [0.5346 1.1328]);

d4\_k = round(1 / w0 \* [0.5906 0.9273]);

d5\_k = round(1 / w0 \* [0.5906 1.0247]);

d6\_k = round(1 / w0 \* [0.5906 1.1328]);

d7\_k = round(1 / w0 \* [0.6535 0.9273]);

d8\_k = round(1 / w0 \* [0.6535 1.0247]);

d9\_k = round(1 / w0 \* [0.6535 1.1328]);

d\_k = [d0\_k; d1\_k; d2\_k; d3\_k; d4\_k; d5\_k; d6\_k; d7\_k; d8\_k; d9\_k];

g.)

% ------------------------ g --------------------------------

D8 = fft(d8, 2048);

plot(w0\*[0:2047], abs(D8).^2);

% loop to search for highest peaks

k1 = 0; max\_1 = 0;

k2 = 0; max\_2 = 0;

for k = 1:length(D8)/2

if(abs(D8(k)) > max\_1)

k2 = k1; max\_2 = max\_1;

k1 = k; max\_1 = abs(D8(k));

elseif(abs(D8(k)) > max\_2)

k2 = k; max\_2 = abs(D8(k));

end

end

h.)Ver 1.0: ttdecode.m

>> testout = ttdecode(x1)

testout =

4 9 1 5 8 7 7

>> testout = ttdecode(x2)

testout =

2 5 3 1 0 0 0

* The code can be found at the end of this report.

## Discussion

I was not able to create the second version of the code that permitted me to use the hardx1 and hardx2 variables. The output above shows that the code worked fine for the given lengths of the touch-tones and silence vectors.

The DTFT is a great method for analyzing signals especially involving touch-tone telephone technology. The ability to decode the touch-tones pressed by a user is a fundamental breakthrough in telephone technology (occurred in past of course).

## Conclusion / results

I obtained the expected results except for the final section of the assignment (i). I was able to successfully use the DTFT to sample the given signals and view the peak frequencies for each touch-tone. By doing so I was able to analyze and verify that the touch-tone signals were working correctly.

I verified this both audibly and visually using graphs and built-in Matlab functions. Matlab is a great tool for designing filters for any type of situation. The built-in functionality and helpful resources are great tools for students to start to get their hands dirty as well as understand simple concepts in signal processing.

I really enjoyed this project and felt that it was not beyond my ability. It was informative and I felt that I learned a lot more about the subject matter after completing the project.

## source

%--------------------------------------------------

% Joseph Hilland

% ECE 439 DSP

% Fall 2014

%touchTonefft.m

%--------------------------------------------------

% ------------------------ A --------------------------------

N = 2048;

n = 0:999;

d0 = sin(0.7217\*n) + sin(1.0247\*n);

%sound(d0, 8192);

d1 = sin(0.5346\*n) + sin(0.9273\*n);

%sound(d1, 8192);

d2 = sin(0.5346\*n) + sin(1.0247\*n);

%sound(d2, 8192);

d3 = sin(0.5346\*n) + sin(1.1328\*n);

%sound(d3, 8192);

d4 = sin(0.5906\*n) + sin(0.9273\*n);

%sound(d4, 8192);

d5 = sin(0.5906\*n) + sin(1.0247\*n);

%sound(d5, 8192);

d6 = sin(0.5906\*n) + sin(1.1328\*n);

%sound(d6, 8192);

d7 = sin(0.6535\*n) + sin(0.9273\*n);

%sound(d7, 8192);

d8 = sin(0.6535\*n) + sin(1.0247\*n);

%sound(d8, 8192);

d9 = sin(0.6535\*n) + sin(1.1328\*n);

%sound(d9, 8192);

% ------------------------ B --------------------------------

% calculate the DTFT of digit 2 and digit 9

% then plot the magnitude and verify peak fall at correct frequency

D2 = fft(d2, N);

D9 = fft(d9, N);

k = 0:2047;

wk = (2 \* pi \* k) / N;

% plot for digit 2 DTFT

%subplot(2,2,1), plot(wk, abs(D2));

%title('DTFT D2');

%axis([0.5, 1.5, 0, 500]);

%xlabel('Freq'), ylabel('mag')

%subplot(2,1,1), plot(wk,angle(D2));

%xlabel('freq'), ylabel('phase')

% plot for digit 9 DTFT

%subplot(2,2,2), plot(wk, abs(D9));

%title('DTFT D9');

%axis([0.5, 1.5, 0, 500]);

%xlabel('Freq'), ylabel('mag')

%subplot(2,1,1), plot(wk,angle(D9));

%xlabel('freq'), ylabel('phase')

% ------------------------ C --------------------------------

space = zeros(1, 100);

phone = [d8 space d3 space d5 space d9 space d0 space d3 space d2 space];

%sound(phone, 8192);

% ------------------------ D --------------------------------

% methods for x1

load('touch.mat');

x1\_1 = x1([1:1000]);

x1\_2 = x1([1100:2100]);

x1\_3 = x1([2200:3200]);

x1\_4 = x1([3300:4300]);

x1\_5 = x1([4400:5400]);

x1\_6 = x1([5500:6500]);

x1\_7 = x1([6600:7600]);

X11 = (fft(x1\_1,N));

X12 = (fft(x1\_2,N));

X13 = (fft(x1\_3,N));

X14 = (fft(x1\_4,N));

X15 = (fft(x1\_5,N));

X16 = (fft(x1\_6,N));

X17 = (fft(x1\_7,N));

%plot(wk, abs(X11),wk, abs(X12), wk, abs(X13), wk, abs(X14), wk, abs(X15), wk, abs(X16), wk, abs(X17));

%LEG = legend('X11', 'X12', 'X13', 'X14', 'X15', 'X16', 'X17');

%set(LEG, 'FontSize', 14);

%title('DTFT touch.mat: x1');

%axis([0, 2, 0, 500]);

%xlabel('Freq'), ylabel('mag')

% 2, 3, 4, 7, 8, 8, 9

% ------------------------ E --------------------------------

% methods for x2

x2\_1 = x2([1:1000]);

x2\_2 = x2([1100:2100]);

x2\_3 = x2([2200:3200]);

x2\_4 = x2([3300:4300]);

x2\_5 = x2([4400:5400]);

x2\_6 = x2([5500:6500]);

x2\_7 = x2([6600:7600]);

X11 = (fft(x2\_1,N));

X12 = (fft(x2\_2,N));

X13 = (fft(x2\_3,N));

X14 = (fft(x2\_4,N));

X15 = (fft(x2\_5,N));

X16 = (fft(x2\_6,N));

X17 = (fft(x2\_7,N));

%plot(wk, abs(X11),wk, abs(X12), wk, abs(X13), wk, abs(X14), wk, abs(X15), wk, abs(X16), wk, abs(X17));

%LEG = legend('X11', 'X12', 'X13', 'X14', 'X15', 'X16', 'X17');

%set(LEG, 'FontSize', 14);

%title('DTFT touch.mat: x2');

%axis([0, 2, 0, 500]);

%xlabel('Freq'), ylabel('mag')

% ------------------------ F --------------------------------

w0 = 2\*pi/2048;

d0\_k = round(1 / w0 \* [0.7217 1.0247]);

d1\_k = round(1 / w0 \* [0.5346 0.9273]);

d2\_k = round(1 / w0 \* [0.5346 1.0247]);

d3\_k = round(1 / w0 \* [0.5346 1.1328]);

d4\_k = round(1 / w0 \* [0.5906 0.9273]);

d5\_k = round(1 / w0 \* [0.5906 1.0247]);

d6\_k = round(1 / w0 \* [0.5906 1.1328]);

d7\_k = round(1 / w0 \* [0.6535 0.9273]);

d8\_k = round(1 / w0 \* [0.6535 1.0247]);

d9\_k = round(1 / w0 \* [0.6535 1.1328]);

d\_k = [d0\_k; d1\_k; d2\_k; d3\_k; d4\_k; d5\_k; d6\_k; d7\_k; d8\_k; d9\_k];

% ------------------------ g --------------------------------

D8 = fft(d8, 2048);

plot(w0\*[0:2047], abs(D8).^2);

% loop to search for highest peaks

k1 = 0; max\_1 = 0;

k2 = 0; max\_2 = 0;

for k = 1:length(D8)/2

if(abs(D8(k)) > max\_1)

k2 = k1; max\_2 = max\_1;

k1 = k; max\_1 = abs(D8(k));

elseif(abs(D8(k)) > max\_2)

k2 = k; max\_2 = abs(D8(k));

end

end

%--------------------------------------------------

% Joseph Hilland

% ECE 439 DSP

% Fall 2014

%ttdecode.m

%--------------------------------------------------

function digits = ttdecode(input)

digits = zeros(1,7);

x = zeros(7,1000);

for k = 1:7

start = 1 + 1100\*(k-1);

finish = start + 999;

x(k,:) = input([start:finish]);

end

% part f

w0 = 2\*pi/2048;

d0\_k = round(1/w0\*[0.7217 1.0247]);

d1\_k = round(1/w0\*[0.5346 0.9273]);

d2\_k = round(1/w0\*[0.5346 1.0247]);

d3\_k = round(1/w0\*[0.5346 1.1328]);

d4\_k = round(1/w0\*[0.5906 0.9273]);

d5\_k = round(1/w0\*[0.5906 1.0247]);

d6\_k = round(1/w0\*[0.5906 1.1328]);

d7\_k = round(1/w0\*[0.6535 0.9273]);

d8\_k = round(1/w0\*[0.6535 1.0247]);

d9\_k = round(1/w0\*[0.6535 1.1328]);

d\_k = [d0\_k; d1\_k; d2\_k; d3\_k; d4\_k; d5\_k; d6\_k; d7\_k; d8\_k; d9\_k];

for k = 1:7

% fft of kth digit

D = fft(x(k,:),2048);

p1 = 0, p2 = 0;

max = 0;

for p = 1:length(D)/2

if(abs(D(p)) > max)

p1 = p;

max = abs(D(p));

end

end

% first peak

D([p1-50:p1+50]) = zeros(1,101);

max = 0;

for p = 1:length(D)/2

if(abs(D(p)) > max)

p2 = p;

max = abs(D(p));

end

end

% matlab indexing fix

p1 = p1 - 1;

p2 = p2 - 1;

P = sort([p1 p2]);

tol = 10;

for j = 1:10

if((abs(d\_k(j,1)-P(1))<tol) && (abs(d\_k(j,2)-P(2))<tol))

digits(k) = j-1;

end

end

end

# Works Cited

Santhanam, B. (unknown, unknown unknown). *ECE 439 DSP & ECE 495 Matlab DSP*. Retrieved December 5, 2014, from Digital Signal Processing: http://www.ece.unm.edu/faculty/bsanthan/ece495/HW\_Exam/touch\_tone.pdf