Ali Mohammed
1507027647

Table of Contents

Introduction 2
Discussion...3
Results 4
Conclusion..8
References: 9
Appendix: Matlab Codes n tones...10

Table of Figures

Figure 1 DTMF Tones 2
Table 1 Result of Signals 4

Figure 3 442 5
Figure 4 823 6
Figure 5 713 6
Figure 6 x 5 x...6
Figure 7 5x3 7
Figure 8 492 7
Figure 9 x7x 7
Figure 10 38x...8

Introduction

Dual-tone multi-frequency (DTMF) signals are tones generated by sinusoidal signal. These sinusoidal signals are used by pressing a telephone keypad. Each keypad is coded to a signal that is composed of a sum of two sinusoids. . The lower frequency is called the low band, and the higher frequency is usually called the high band. By correlating the two bands, DTMF signals can be decoded to which symbol was sent. The frequencies of a DTMF system scheme are shown in the table below. DTMF signals having great applications in PBX which are private branch exchange. Telephone networks within big organizations are used to route to different people based on the extension pressed on the keypad. Based on this application, you can dial the organizations number and connect to any department based on a specific dial tone that is pressed from the keypad of the caller. My purpose for this project was to decode a three digit dial tone based on the table of frequencies below.

Table 1: DTMF Tones

	1336 Hz	1477Hz	1209 Hz
697 Hz	1	4	7
852 Hz	2	5	8
770 Hz	3	6	9

Figure 1 DTMF Tones

Discussion

I have been asked to use MATLAB which is a powerful tool for signal processing to decode a three tone DTMF signal. The input for the Matlab solution is the signals that are generated and posted on blackboard and the output should be the three keys that were pressed to generate such tones. I started my process for decoding the tone by developing the code for decoding a single keypad to get an understanding of how to isolate the frequency component for a single tone. After that I proceeded with trial and error to find ways to be able to decode the files posted on blackboard which contained three tones as an input. My solution for that problem is as follows:

Create a decoder for DTMF for n number of tones:

- First we calculate the length of the signal that was inputted and divide by the number of tones included.
- Then we find where the signal starts and ends by looping for values around the length of the signal.
- Moreover, we have to apply the FFT built in algorithm which gets us the DFT of the signal. The frequency data is important in decoding the DTMF signal.
- We take the absolute values of the FT.
- Use findpeaks to determine the peaks above 480 because of the noise (minpeakheight);
- The findpeaks function was frustrating to use because some noisy signals would have multiple signals above the peak and that changes for each of the noisy signals. So playing for the peak function for each noisy signal would give a better approximation.
- Peak frequency of low and high are shown then are mapped to the dial tone using if statements.
- Finally I displayed the printed number along with a graph of frequency & amplitude graph of each signal.

Results

I will post the result for the first five noiseless and noisy signals from blackboard in descending order 1-5 / 11-5. The noisy signal results struggled to find the proper tone because of the noisy peaks we had in the signal.Moreover a table will be included for the results..

Table 1 Result of Signals

Signal name	Dialed Tone
Noiseless Signal 1	545
Noiseless Signal 2	677
Noiseless Signal 3	442
Noiseless Signal 4	823
Noiseless Signal 5	713
Noisy Signal 11	X5X
Noisy Signal 12	5X3
Noisy Signal 13	492
Noisy Signal 14	X7X
Noisy Signal 15	38X

Ali Mohammed
1507027647

Noiseless signal 1

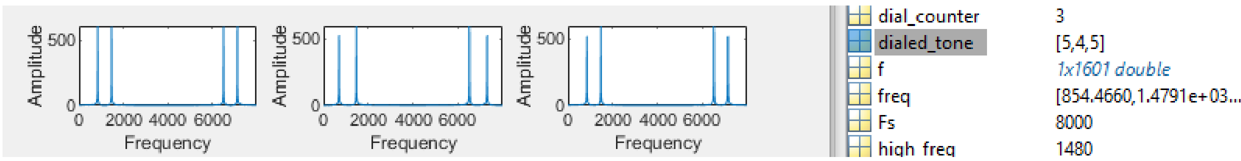

Figure 1 545

Noiseless signal 2

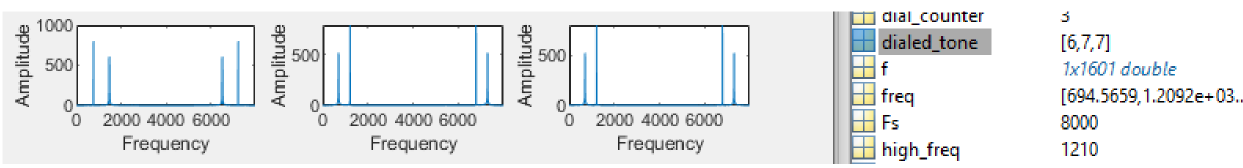

Figure 677

Noiseless signal 3

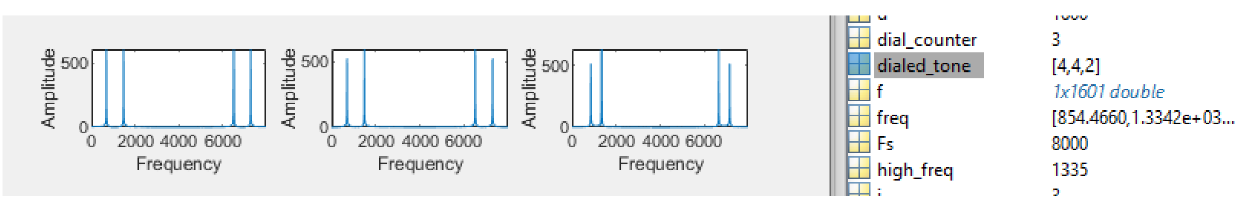

Figure 2 442

Ali Mohammed
1507027647

Noiseless signal 4

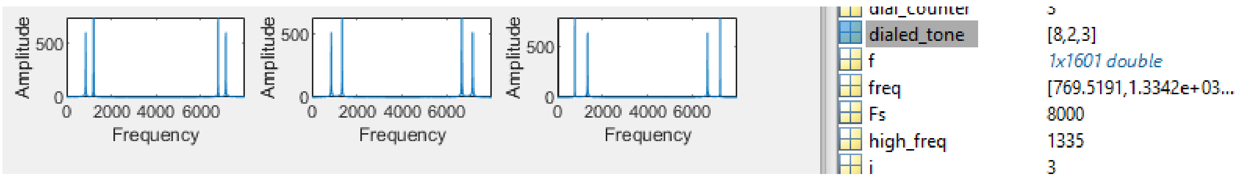

Figure 3 823

Noiseless signal 5

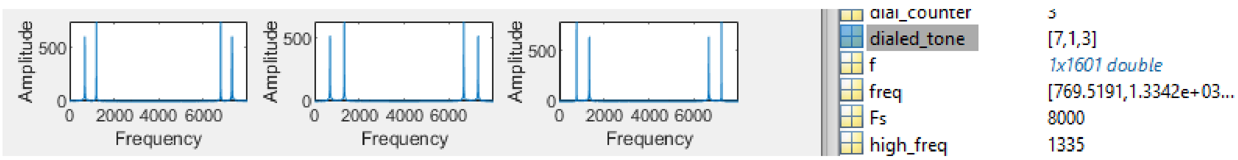

Figure 4 713

Noisy signal 11

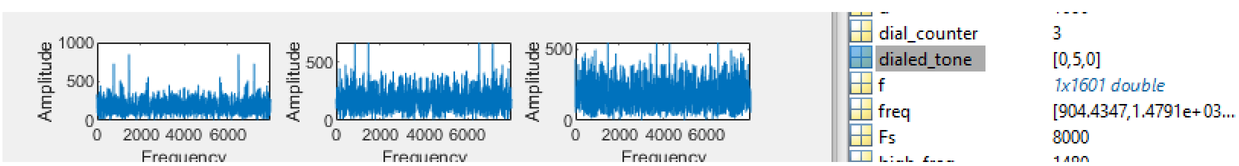

Figure 5 x 5 x

Noisy signal 12

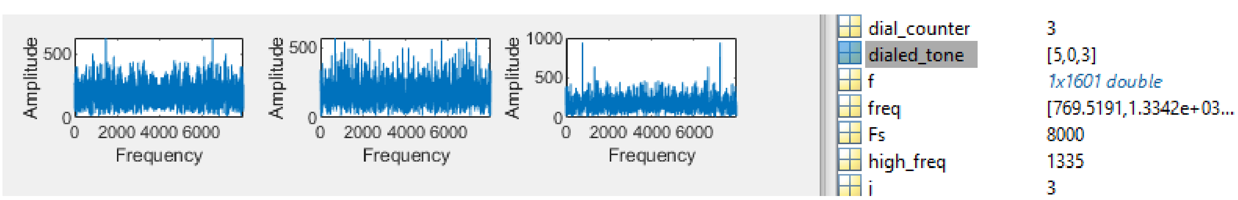

Figure 6 5x3

Noisy signal 13

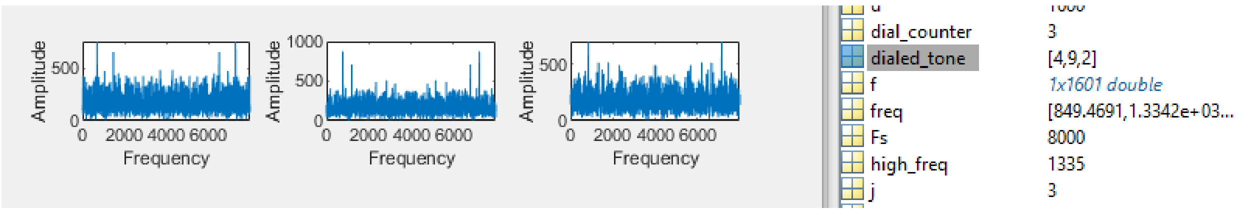

Figure 7 492

Noisy signal 14

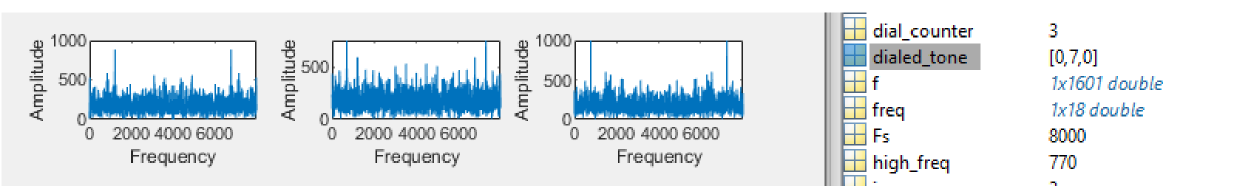

Figure 8 x7x

Noisy signal 15

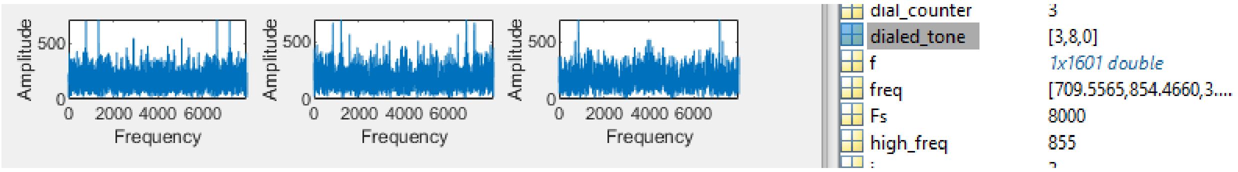

Figure 9 38x

Conclusion

In conclusion, DTMF tones are important to understand as they are still in use for different applications mainly in PBX in governments and other organizations. Being able to decode these tones into the keypad pressed allows for the routing of the call into the correct person or department. This was a challenging fun project that gave me further understanding of DSP and DFT of signals. Different methods were explored till I finally reached a decoder that can work for N tones.

References:

- Project_DSP_F22.pdf
- *Matlab Help Documentation - MATLAB & Simulink*. Available at: https://www.mathworks.com/help/index.html (Accessed: November 19, 2022).

Ali Mohammed
1507027647

Appendix: Matlab Codes n tones

```
%Ali Mohammed
%1507027647
%DTMF DEOCDER USING FFT By finding the peaks and then applying frequency of
% low & high peak to the correct keypad.
%if the signal is noisy it has to be passed through a gaussian filter first
or
%through a clustering ML algorithm to identify the frequencies of the
%signal
%playSignal sound
dialed_tone = [ 0 0 0 ];
sound(Signal,Fs);

lengthh = 3;%set the amount of tones u want to decode
d = floor(length(Signal)/lengthh); %round down the length of the signal/n
number of tones
for j = 1:lengthh
    if j-1 == 0
        start = 1;
    else
        start = (j-1)*d;
    end

%preform FFT provides frequency information about the signal (get the dft)
p = abs(fft(Signal(start:j*d)));
n = length(Signal(start:j*d));

%get the proper frequencies of the signal and plot them
f = (0:n-1)*(Fs/n);
subplot(4,3,j)
plot(f,p) %freq amp graph for each signal
 xlabel('Frequency')
 ylabel('Amplitude')

%use findpeaks to determine the peaks above 300 (minpeakheight);
[a, location] = findpeaks(p,'Minpeakheight',480);

%change location into frequency
freq = f(location);
```

Ali Mohammed
1507027647

```
%limit to our DTMF to less then 2000
peak_freq = freq(freq< 2000 );

%peak freq shows the frequencies in the dialtone where two signals are high
%adn the other is low.


%round the frequencies min&&max
low_freq = ceil(peak_freq(1));
high_freq = ceil(peak_freq(2));

%see outputs of high and low
sprintf('%f', low_freq)
sprintf('%f', high_freq)

%determine which number it is w/ high and low frequencies with their
%mangintude

for dial_counter = j:j
if (low_freq >= 685 && low_freq <= 710) && (high_freq >= 1330 && high_freq <=
1380)
    dialed_tone(dial_counter)=1;
    elseif (low_freq >= 840 && low_freq <= 890)  && (high_freq >= 1330 &&
high_freq <= 1380)
    dialed_tone(dial_counter)=2;
    elseif (low_freq >= 740 && low_freq <= 800 ) && (high_freq >= 1330 &&
high_freq <= 1380)
    dialed_tone(dial_counter)=3;
    elseif (low_freq >= 685 && low_freq <= 710) && (high_freq >= 1450 &&
high_freq <= 1500)
   dialed_tone(dial_counter)=4;
    elseif (low_freq >= 840 && low_freq <= 890) && (high_freq >= 1450 &&
high_freq <= 1500)
    dialed_tone(dial_counter)=5;
    elseif (low_freq >= 740 && low_freq <= 800) && (high_freq >= 1450 &&
high_freq <= 1500)
    dialed_tone(dial_counter)=6;
    elseif (low_freq >= 685 && low_freq <= 710) && (high_freq >= 1199 &&
high_freq <= 1240)
    dialed_tone(dial_counter)=7;
    elseif (low_freq >= 840 && low_freq <= 890) && (high_freq >= 1199 &&
high_freq <= 1240)
    dialed_tone(dial_counter)=8;
    elseif (low_freq >= 740 && low_freq <= 800) && (high_freq >= 1199 &&
high_freq <= 1240)
    dialed_tone(dial_counter)=9;
end
end
end
```