**ELECENG 3TP3 Signals and Systems**

Lab 4: Signal Analysis Using the Discrete Fourier Transform

Section C01, Instructor: Prof. Jun Chen

Khaled Hassan

Hassak9

400203796

Part I

Note: The code used is provided as screenshots in the body of the report (as it looks cleaner), and as text in the Appendix.

2.

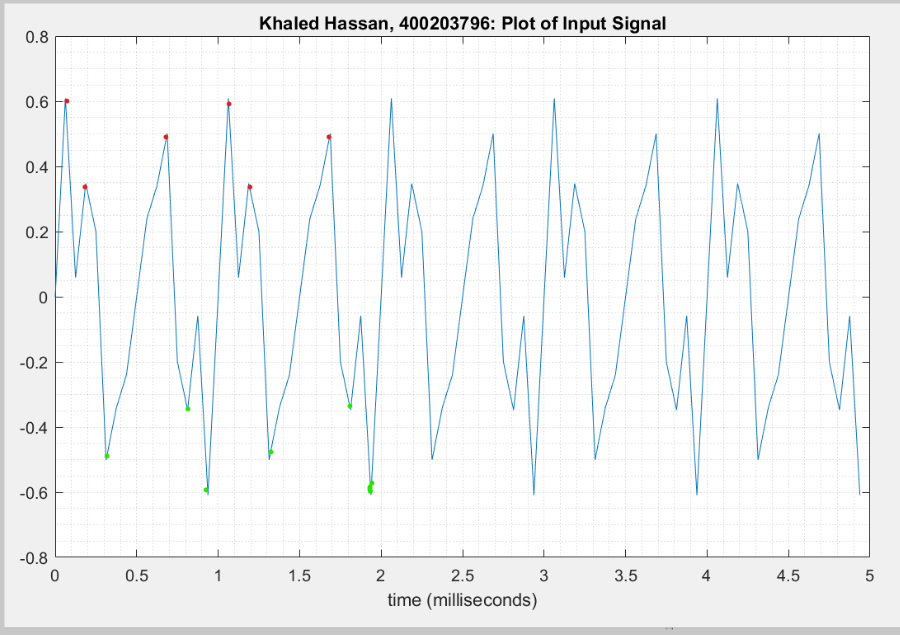
“tones2020.wav” is a 10 second audio file that seems to be comprised of a single high frequency sinusoid or superposition/combination of sinusoids. Upon listening to it, a single high-pitched beep is heard. While the pitch changes very slightly, it remains extremely high judging by the varying but sharp pitch of the sound in the audio file. That indicates that while frequency might change, it remains high in the kHz range.

3.

|  |  |
| --- | --- |
|  |  |

The code titled ‘Part1Q3.m’ in the appendix was used to generate the above plot of the first 5 msec of the audio file. This plot confirms my assumptions about the function used to generate the audio file; it is a compound periodic function made from a linear combination of sinusoids. The compound periodic function has a period of 1msec, meaning a frequency of 1KHz. The irregular shape indicates that the sinusoids that make up the compound function are either out of phase, or have differing frequencies.

4.



In any single period of 1 msec on the plot, there are 3 positive (red) peaks and 3 negative (green) corresponding peaks. As such, I estimate that the function that makes up this audio signal is comprised of 3 sinusoids. To estimate frequencies, the distance between the peaks was measured, multiplied by 10^-2 (as we are dealing with msec), then the inverse (1/ANS) is taken to determine the frequency from the difference in “period”. For example, the difference between the first 2 red peaks is (0.1875 – 0.0625 msec) => one of the frequencies is estimated to be equal to 8 KHz. Similarly between the 2nd and 3rd positive peaks, another frequency is found to be approximately 2 KHz. Finally, this was repeated for the 3rd and 4th positive peaks to find the third frequency to be approximately 5 KHz.

5.

|  |  |
| --- | --- |
|  |  |

6.

The code in part 5 was used to generate the single-sided magnitude spectrum above. The Fast Fourier Transform algorithm was called using the MATLAB function fft() to implement the Discrete Fourier Transform of the compound audio signal, to “break it down” into the separate sinusoids that form the audio signal. From the single-sided magnitude spectrum, we can find out how many sinusoids there are by looking at how many peaks there are. The magnitude of the peaks is found by taking the y-axis value, representing magnitude, of the highest point of the peak. The x-axis value represents the frequency of each sinusoid. As can be seen in the figure generated for part 5, there are 3 sinusoids; the 1st is of frequency 2 KHz and magnitude 0.4. The 2nd is of frequency 5 KHz and magnitude 0.2. Finally, the 3rd sinusoid is of frequency 6 KHz and amplitude 0.2. These findings indicate that my earlier estimates for the frequencies were incorrect due to an incorrect method applied. I was correct, however, in my estimation of the existence of three sinusoids making up the compound periodic signal.

7.

|  |  |
| --- | --- |
|  |  |

The code used to generate the recreated plot of the audio signal can be found on the right. The compound periodic function that was used here was derived from the values found in part 6; the 3 sinusoids with their respective amplitudes and frequencies. The recreated plot is identical to the plot derived directly from the audio signal in part 3. This indicates that my analysis of the audio signal through the DFT in Q5 and Q6 was correct, along with my choice to use the sin() function instead of cos().

Part II

2.

“SecretMessage2020.wav” is a 64 second audio file that is comprised of a sequence of “beeps” with varying frequencies, that could vaguely be heard behind the constant static. Besides the static, the pitch of the beeps could be heard changing every second, which indicates a change in frequency. 64 frequencies represent 64 coded characters, which can be decoded using the file ‘CodeBook.pdf’.

3.

|  |  |
| --- | --- |
|  |  |

Similar to what was done in Part I, the FFT of the whole audio signal was performed. I found that there 64 second coded message is comprised of 7 sinusoids: The 1st of frequency 1 KHz and magnitude 0.03245, the 2nd of frequency 2 KHz and magnitude 0.02551, the 3rd of frequency 3 KHz and magnitude 0.0243, the 4th of frequency 4 KHz and magnitude 0.02785, the 5th of frequency 5 KHz and magnitude 0.03469, the 6th of frequency 6 KHz and magnitude 0.02601, and the 7th of frequency 7 KHz and magnitude 0.02781.

4.

|  |  |
| --- | --- |
| 1st Character |  |
| 2nd – 64th characters |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Character** | **Second** |  | **Character** | **Second** |  | **Character** | **Second** |  | **Character** | **Second** |
| I | 1 |  | E | 17 |  | U | 33 |  | Y | 49 |
| T | 2 |  | R | 18 |  | SPACE | 34 |  | O | 50 |
| SPACE | 3 |  | SPACE | 19 |  | G | 35 |  | U | 51 |
| D | 4 |  | H | 20 |  | O | 36 |  | SPACE | 52 |
| O | 5 |  | O | 21 |  | SPACE | 37 |  | D | 53 |
| E | 6 |  | W | 22 |  | A | 38 |  | O | 54 |
| S | 7 |  | SPACE | 23 |  | S | 39 |  | SPACE | 55 |
| SPACE | 8 |  | S | 24 |  | SPACE | 40 |  | N | 56 |
| N | 9 |  | L | 25 |  | L | 41 |  | O | 57 |
| O | 10 |  | O | 26 |  | O | 42 |  | T | 58 |
| T | 11 |  | W | 27 |  | N | 43 |  | SPACE | 59 |
| SPACE | 12 |  | L | 28 |  | G | 44 |  | S | 60 |
| M | 13 |  | Y | 29 |  | SPACE | 45 |  | T | 61 |
| A | 14 |  | SPACE | 30 |  | A | 46 |  | O | 62 |
| T | 15 |  | Y | 31 |  | S | 47 |  | P | 63 |
| T | 16 |  | O | 32 |  | SPACE | 48 |  | PERIOD | 64 |

The code titled ‘Part2Q4.m’ in the appendix was used to decode the message in the ‘SecretMessage2020.wav’ audio file. The audio file was split into 64 1-second intervals, with FFT being performed on each interval to determine the frequencies present. Every second contained a 4-frequency combination from all the possible frequencies found in Q3: 1 KHz, 2 KHz, 3 KHz, 4 KHz, 5 KHz, 6 KHz and 7 KHz, with each 4-frequency combination referring to a character from [A, B, C, D, …, Z, SPACE, PERIOD]. To find the 1st character only, a unique block of code was used since the method to decode all other characters was the same. After performing the FFT 64 times, the coded message was found to be: “It does not matter how slowly you go as long as you do not stop.”

Appendix: Code Used

Part I

3. ‘Part1Q3.m’

|  |
| --- |
| % ELECENG 3TP3 Lab 4  % Khaled Hassan - 400203796  clc;  clear;  % Read in the signal from the audio file  % audioread used instead of wavread  [signal, Fs] = audioread('tones2020.wav');  L = length(signal); % signal length  T = 1/Fs; % period  t = [0:L-1]\*T;  % Plot the signal for t\_plot msec  t\_plot = 5; % plot time is 5 msec  msec\_per\_sec = 1000;  numSamples = t\_plot\*Fs/msec\_per\_sec;  plot(msec\_per\_sec\*t(1:numSamples), signal(1:numSamples));  title('Khaled Hassan, 400203796: Plot of Input Signal');  xlabel('time (milliseconds)');  grid('minor'); |

5. ‘Part1Q5.m’

|  |
| --- |
| % ELECENG 3TP3 Lab 4  % Khaled Hassan - 400203796  % Take the DFT  Y = fft(signal)/L;  f = Fs/2\*linspace(0,1,L/2+1);  % Plot the single-sided magnitude spectrum.  plot(f,2\*abs(Y(1:L/2+1)));  title('Khaled Hassan, 400203796: Single-Sided Magnitude Spectrum');  xlabel('Frequency (Hz)');  ylabel('|Y(f)|');  axis([0 Fs/2 0 .5]);  grid('minor'); |

7. ‘Part1Q7.m’

|  |
| --- |
| % ELECENG 3TP3 Lab 4  % Khaled Hassan - 400203796  [signal, Fs] = audioread('tones2020.wav');  % Fs is equal to 16000, the sampling frequency  L = length(signal); % signal length  T = 1/Fs; % period  t = [0:L-1]\*T;  % Plot the signal for t\_plot msec  t\_plot = 5; % plot time is 5 msec  msec\_per\_sec = 1000;  numSamples = t\_plot\*Fs/msec\_per\_sec;  % form of sinusoids: A \* sin(2pift)  x1 = 0.4 \* sin(2 \* pi \* 2000 \* t);  x2 = 0.2 \* sin(2 \* pi \* 5000 \* t);  x3 = 0.2 \* sin(2 \* pi \* 6000 \* t);  xT = x1 + x2 + x3; % form the compound sinusoid  plot(msec\_per\_sec\*t(1:numSamples), xT(1:numSamples));  title('Khaled Hassan, 400203796: Recreated Plot of Audio Signal');  xlabel('time (milliseconds)');  grid('minor'); |

Part II

3. ‘Part2Q3.m’

|  |
| --- |
| % ELECENG 3TP3 Lab 4  % Khaled Hassan - 400203796  clc;  clear;  [signal, Fs] = audioread('SecretMessage2020.wav');  L = length(signal); % signal length  T = 1/Fs; % period  t = [0:L-1]\*T;  Y = fft(signal) / L;  f = Fs/2\*linspace(0,1,L/2+1);  plot(f,2\*abs(Y(1:L/2+1)));  title('Khaled Hassan, 400203796: Single-Sided Magnitude Spectrum');  xlabel('Frequency (Hz)');  ylabel('|Y(f)|');  axis([0 Fs/2 0 .05]);  grid('minor'); |

4. ‘Part2Q4.m’

|  |
| --- |
| % ELECENG 3TP3 Lab 4  % Khaled Hassan - 400203796  [signal, Fs] = audioread('SecretMessage2020.wav');  L = length(signal) / 64; % signal interval length = 1024000 / 64  T = 1/Fs;  t = (0:L-1)\*T;  % for the 1st character only  %Y = fft(signal(1: Fs)) / L;  %f = Fs/2\*linspace(0,1,L/2+1);  % this works for all characters beyond the 1st one  % s is the second for which the dft is being applied  % from s to s+1. This value was increased every time from 1 until 63  s = 61; %each s finds the (s+1)'th character  Y = fft(signal((s) \* Fs: (s+1)\*Fs))/L;  f = Fs/2\*linspace(0,1,L/2+1);  plot(f,2\*abs(Y(1:L/2+1)));  title(sprintf('Khaled Hassan, 400203796: Character %d', s + 1));  %title('Khaled Hassan, 400203796: Character 1');  xlabel('Frequency (Hz)');  ylabel('|Y(f)|');  axis([0 Fs/2 0 .1]);  grid('minor'); |