ELECENG 3TP3
LAB 4
Signal Analysis Using DFT

Authored By:

Ahmed Raja - rajaa5 - 400306467
Ahmad Choudhry - chouda27 - 400312026

Instructor:

Dr. T. Kirubarajan

McMaster University

Dec 08 2022

Title Page

Part 1

2. We hear a constant high frequency sound that plays for 10 seconds.

3. The MATLAB code used to generate a plot of the first 5 seconds of the waveform is shown in Listing 1.

Listing 1: Generating plot of waveform

```
%rajaa5 400306467 & chouda27 400312026
%Reading signal from given audio file
[signal, Fs] = audioread('tones2022.wav');
L = length(signal);
T = 1/Fs;
t = [0:L-1]*T;
%Plotting signal for first 5 msec (t_plot)
t_plot = 5;
msec_per_sec = 1000;
numSamples = t_plot*Fs/msec_per_sec;
plot(msec_per_sec*t(1:numSamples), signal(1:numSamples));
title('Input Signal Plot');
xlabel('time (ms)');
grid('minor');
exportgraphics(gcf, 'part1_q3.png');
```

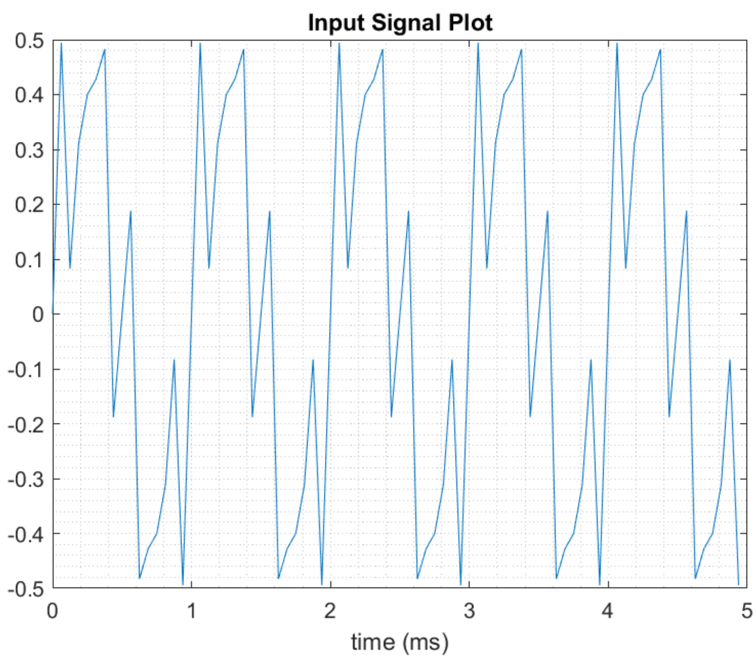

Figure 1: Plotting first 5 msec of waveform for tones2022.wav

4. In the plot of the input signal, we can see that there is a periodic signal that repeats every 1 millisecond (frequency of 1000 Hz). It is difficult to tell how many sinusoids make up the signal and their frequencies from the plot of the input signal. The number of sinusoids that

make up the signal and their frequencies can be determined by applying the Fourier transform on the input signal.

5. Magnitude vectors (|Y|(f)) of the DFT vectors using the FFT are plotted as stem plots for the entire range of Frequency (Hz) in Listing 2.

Listing 2: Generating Stem Plots of All Magnitude Vectors

```
%rajaa5 400306467 & chouda27 400312026
%Reading signal from given audio file
[signal, Fs] = audioread('tones2022.wav');
L = length(signal);
T = 1/Fs;
t = [0:L-1]*T;

%Plotting using fft of signal and shifting by |Y|(f)
Y=fft(signal);
h=fftshift(Y);
%obtaining length and using it to get magnitude of Y
len_Y_f=length(Y);
F=(-len_Y_f/2:len_Y_f/2-1)/len_Y_f*Fs;
%stem plotting
stem(F,abs(h))
xlabel 'Frequency (Hz)'
ylabel '|Y|(f)'
grid('minor')
exportgraphics(gcf, 'part1_q5.png');
```

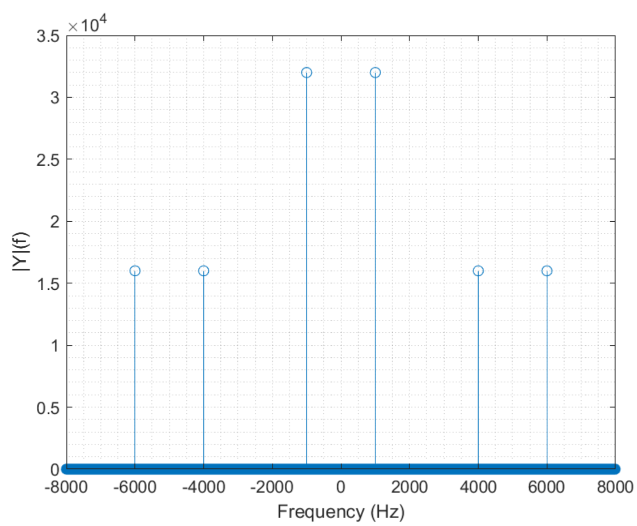

Figure 2: Plotting the vectors for all Frequency

6. The plot of the DFT of the audio signal is shown in Figure 3. The MATLAB code used to generate the graphs is shown in Listing 3.

Listing 3: DFT of the audio signal

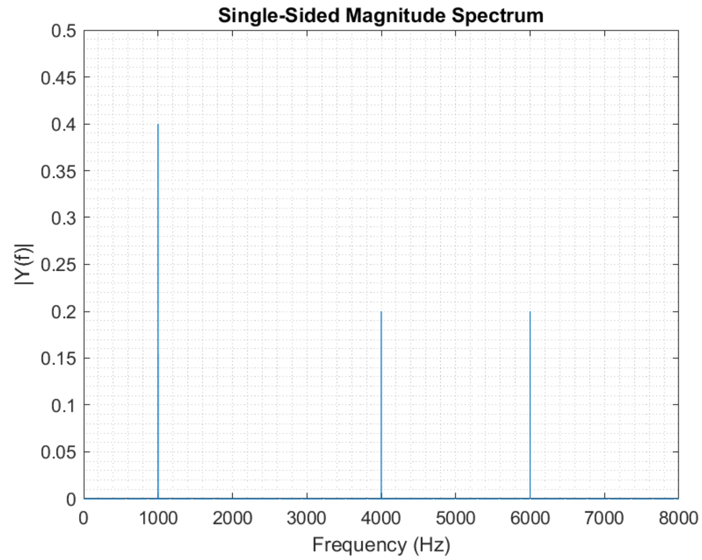

Figure 3: DFT of the audio signal

```
%rajaa5 400306467 & chouda27 400312026
part1_q3;
%Taking the Discrete Fourier Transform
Y = fft(signal)/L;
F = Fs/2*linspace(0,1,L/2+1);
%Plotting only the single-side of the magnitude spectrum as
    desired
plot(F,2*abs(Y(1:L/2+1)));
title('Single-Sided Magnitude Spectrum');
xlabel('Frequency (Hz)');
ylabel('|Y(f)|');
axis([0 Fs/2 0 .5]);
grid('minor');
exportgraphics(gcf, 'part1_q6.png');
```

7. The audio signal is made up of three sinusoids, and the frequencies and magnitudes of the sinusoids are: 1000 Hz with 0.4, 4000 Hz and 0.2, and 6000 Hz with 0.2 as shown above in 3. This means that the input signal is made by the 3 frequencies shown and the magnitudes Y(|f|) will be the amplitudes at the corresponding frequencies of the sinusoid.

8. The MATLAB code to generate the signal, and compare the plots of the generated signal and the input signal is shown in Listing 4. The two plots are shown in Figure 4. The plot of the generated signal is identical to the plot of the input signal found in Part 3, which is expected as there is no noise in the input signal (as seen in question 6) and the input signal is made entirely of the sum of the three determined sinusoids. The satisfying equation in terms of addition of sinusoids will be

$$y = 0.4\sin(2\pi * 1000t) + 0.2\sin(2\pi * 4000t) + 0.2\sin(2\pi * 6000t)$$

Note, the $*$ here does not indicate convolution, only multiplication.

Listing 4: Generating signal from sinuosoids

```
%rajaa5 400306467 & chouda27 400312026
part1_q3;

sin_1 = 0.4*sin(2*pi*1000*t);
sin_2 = 0.2*sin(2*pi*4000*t);
sin_3 = 0.2*sin(2*pi*6000*t);

generated_signal = sin_1+sin_2+sin_3;

tiledlayout('flow');
nexttile; plot(msec_per_sec*t(1:numSamples), signal(1:numSamples));
title('Plot of Input Signal'); xlabel('time (ms)'); grid('minor');
nexttile; plot(msec_per_sec*t(1:numSamples), generated_signal(1:numSamples));
title('Plot of Generated Signal'); xlabel('time (ms)'); grid('minor');
exportgraphics(gcf, 'part1_q8.png');
```

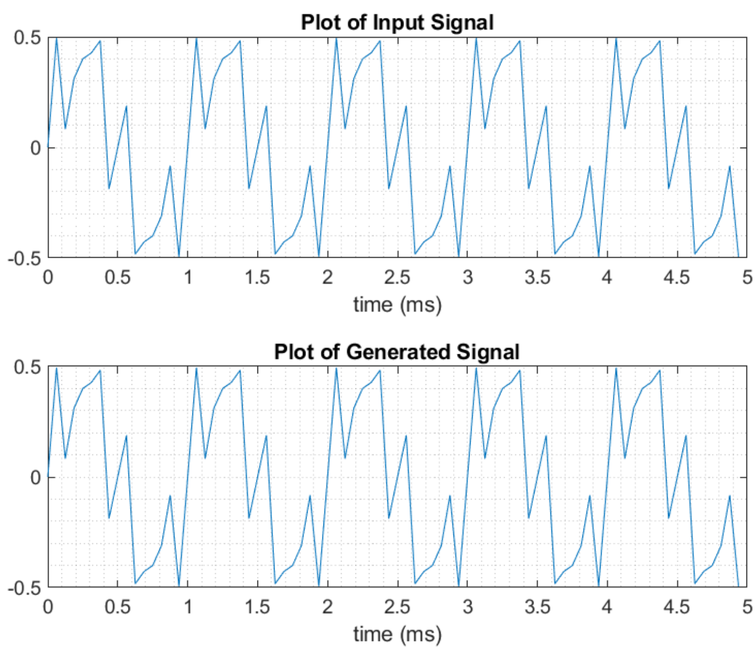

Figure 4: Comparison of two plots

Part 2

2. The signal is quite noisy, that changes to a slightly different tone every second. This behaviour persists for the remainder of the signal duration.

3. The single-sided magnitude of the DFT was plotted using the same procedure described in Part I. As described before, it is possible to observe the frequencies used in the signal by looking at the plot of the DFT shown in Figure 5. The frequencies used in the signal are:`[1000:1000:7000]` Hz with magnitudes that can be estimated by observing Figure 5. The MATLAB code used to generate the DFT and save it to a file is shown in Listing 5.

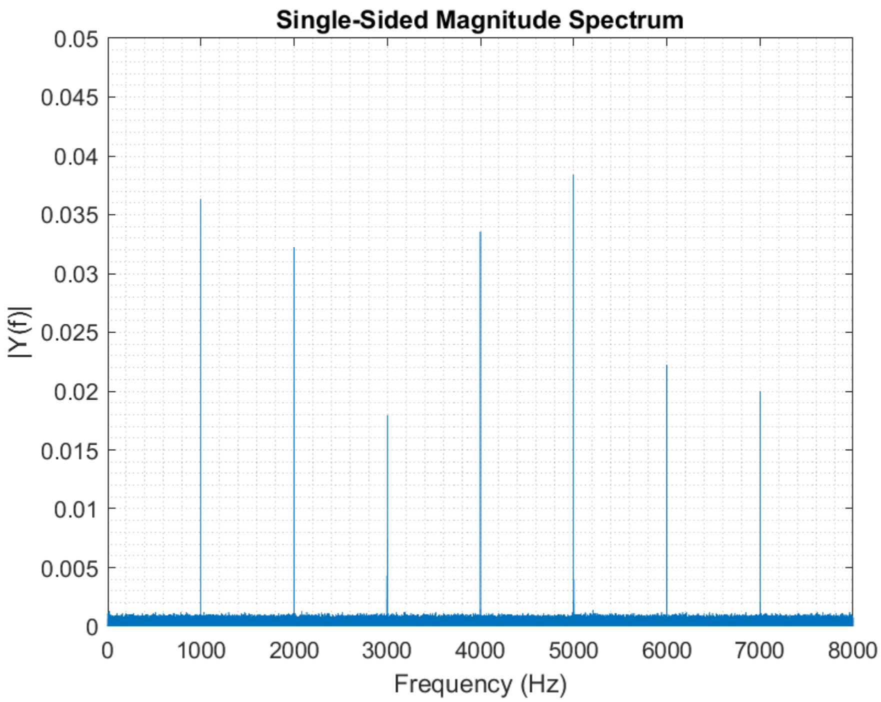

Figure 5: The DFT of the 'SecretMessage2022.wav' file

Listing 5: Generating DFT of 'SecretMessage2022.wav' file

```
%rajaa5 400306467 & chouda27 400312026
[signal, Fs] = audioread('SecretMessage2022.wav');
L = length(signal);
T = 1/Fs;
t = [0:L-1]*T;
%Taking the DFT
Y = fft(signal)/L;
f = Fs/2*linspace(0,1,L/2+1);
%Plotting desired spectrum
plot(f,2*abs(Y(1:L/2+1)));
title('Single-Sided Magnitude Spectrum');
xlabel('Frequency (Hz)');
ylabel('|Y(f)|');
axis([0 Fs/2 0 .05]);
```

```
grid('minor');
exportgraphics(gcf, 'part2_q3.png');
```

4. As described in the question, the most appropriate method here was to split the 1 Second "symbol periods" into a loop through which the FFT of each observed symbol could be taken. The `maxk` gave the 4 corresponding maximum frequencies (magnitudes) present in each full cycle (period). This was multiplied by the number of required amount of characters in other words the maximum amount of information in the array from the given signal to form a 77 x 4 matrix. This is implemented in the MATLAB code shown in Listing 6.

Listing 6: Determing frequencies contained in each symbol period

```
%rajaa5 400306467 & chouda27 400312026
%code described in report above
[signal, Fs] = audioread('SecretMessage2022.wav');
L = length(signal);
T = 1/Fs;
t = [0:L-1]*T;

freqs = zeros(77,4);

for i = 0:75
    Y = fft(signal(1+i*Fs:(i+1)*Fs))/(Fs);
    [B, I] = maxk(abs(Y(1:Fs/2+1)),4);
    freqs(i+1,:) = I-1;
end

freqs = sort(freqs,2);
```

Then, according to the codebook file give, the message was decoded using conditonal statements looped across all possible character spaces. This was done by matching the frequencies of the secret message to that which was given in the codebook A snippet of the MATLAB to perform this is shown in Listing 7. The decoded message is: "NEVER LET THE FEAR OF STRIKING OUT KEEP YOU FROM PLAYING THE GAME.BABE RUTH."

Listing 7: Matching frequency vectors with message characters to decode message

```
%rajaa5 400306467 & chouda27 400312026
part2_q4;

secret_message = "";

for i = 1:76
    if freqs(i,:) == [1000 2000 3000 4000]
        secret_message = secret_message + "A";
    elseif freqs(i,:) == [1000 2000 3000 5000]
        secret_message = secret_message + "B";
    elseif freqs(i,:) == [1000 2000 3000 6000]
        secret_message = secret_message + "C";
    elseif freqs(i,:) == [1000 2000 3000 7000]
```

```
            secret_message = secret_message + "D";
        elseif freqs(i,:) == [1000 2000 4000 5000]
            secret_message = secret_message + "E";
        elseif freqs(i,:) == [1000 2000 4000 6000]
            secret_message = secret_message + "F";
        elseif freqs(i,:) == [1000 2000 4000 7000]
            secret_message = secret_message + "G";
        elseif freqs(i,:) == [1000 2000 5000 6000]
            secret_message = secret_message + "H";
        elseif freqs(i,:) == [1000 2000 5000 7000]
            secret_message = secret_message + "I";
        elseif freqs(i,:) == [1000 2000 6000 7000]
            secret_message = secret_message + "J";
        elseif freqs(i,:) == [1000 3000 4000 5000]
            secret_message = secret_message + "K";
        elseif freqs(i,:) == [1000 3000 4000 6000]
            secret_message = secret_message + "L";
        elseif freqs(i,:) == [1000 3000 4000 7000]
            secret_message = secret_message + "M";
        elseif freqs(i,:) == [1000 3000 5000 6000]
            secret_message = secret_message + "N";
        elseif freqs(i,:) == [1000 3000 5000 7000]
            secret_message = secret_message + "O";
        elseif freqs(i,:) == [1000 3000 6000 7000]
            secret_message = secret_message + "P";
        elseif freqs(i,:) == [1000 4000 5000 6000]
            secret_message = secret_message + "Q";
        elseif freqs(i,:) == [1000 4000 5000 7000]
            secret_message = secret_message + "R";
        elseif freqs(i,:) == [1000 4000 6000 7000]
            secret_message = secret_message + "S";
        elseif freqs(i,:) == [1000 5000 6000 7000]
            secret_message = secret_message + "T";
        elseif freqs(i,:) == [2000 3000 4000 5000]
            secret_message = secret_message + "U";
        elseif freqs(i,:) == [2000 3000 4000 6000]
            secret_message = secret_message + "V";
        elseif freqs(i,:) == [2000 3000 4000 7000]
            secret_message = secret_message + "W";
        elseif freqs(i,:) == [2000 3000 5000 6000]
            secret_message = secret_message + "X";
        elseif freqs(i,:) == [2000 3000 5000 7000]
            secret_message = secret_message + "Y";
        elseif freqs(i,:) == [2000 3000 6000 7000]
            secret_message = secret_message + "Z";
        elseif freqs(i,:) == [2000 4000 5000 6000]
            secret_message = secret_message + " ";
        elseif freqs(i,:) == [2000 4000 5000 7000]
```

```
            secret_message = secret_message + ".";
        end
end

disp(secret_message);
```

The output message is shown below for confirmation of matlab simulation as well as all 76 plots of spectrum. You may zoom in for a clear view.

```
>> part2_q4decode
NEVER LET THE FEAR OF STRIKING OUT KEEP YOU FROM PLAYING THE GAME.BABE RUTH.
>>
```

The code that was appended to Listing 6 to create the 76 graphs is also shown below as a reference:

```
%rajaa5 400306467 & chouda27 400312026


sample_second = L/76;
f = linspace(0,1,sample_second/2)*Fs/2;
% plot
for i = sample_second:sample_second:L


    %plot
    nexttile;
    stem(f,1000*abs(Y(1:sample_second/2)));
    title("Plot");
    xlabel("Hz");
    ylabel("|Y(F)|");
    axis([0 Fs/2 0 .5]);
    grid('minor');
end


```