CmpE 362
Spring 2019
Project 3

Gurkan Demir
2015400177

May 26, 2019

Contents

1 Introduction

1.1 Project Description

In this homework, I am expected to implement some simple frequency domain exercises with MATLAB. We have 2 questions in this homework, named **Advanced Peak Finder, Converting a Hubble Deep Space Image Into Space Sound**, respectively.

In the first question, I am expected to plot number of peaks I found versus changing cutoff-frequencies using **PinkPanther30.wav**.

In the second question, I am expected to convert Hubble deep space image into space sound. Image that I used in this question is named **Hubble-Massive-Panorama**

2 Advanced Peak Finder

2.1 Question Description

In this part, I am expected to improve my peak detection algorithm that I developed in the previous homework. I am designing a low pass filter. For this filter, I changed the limit frequency of the low pass filter between 1000Hz, 2000Hz, 3000Hz and 4000Hz. I applied these four different low pass filters with cut off frequencies (1k,2k,3k,4k). Additionally, I am expected to plot number of peaks versus changing cut off frequencies.

For this question, in order to implement low pass filter I used built-in function for low pass. It takes 3 parameters, data file, cutoff-frequency and sampling frequency, respectively.

2.2 Implementation

First I read the input file named **PinkPanther30.wav**, and got its sampling frequency.

Then, I found the number of peaks for original data without any filter using built-in findpeaks function.

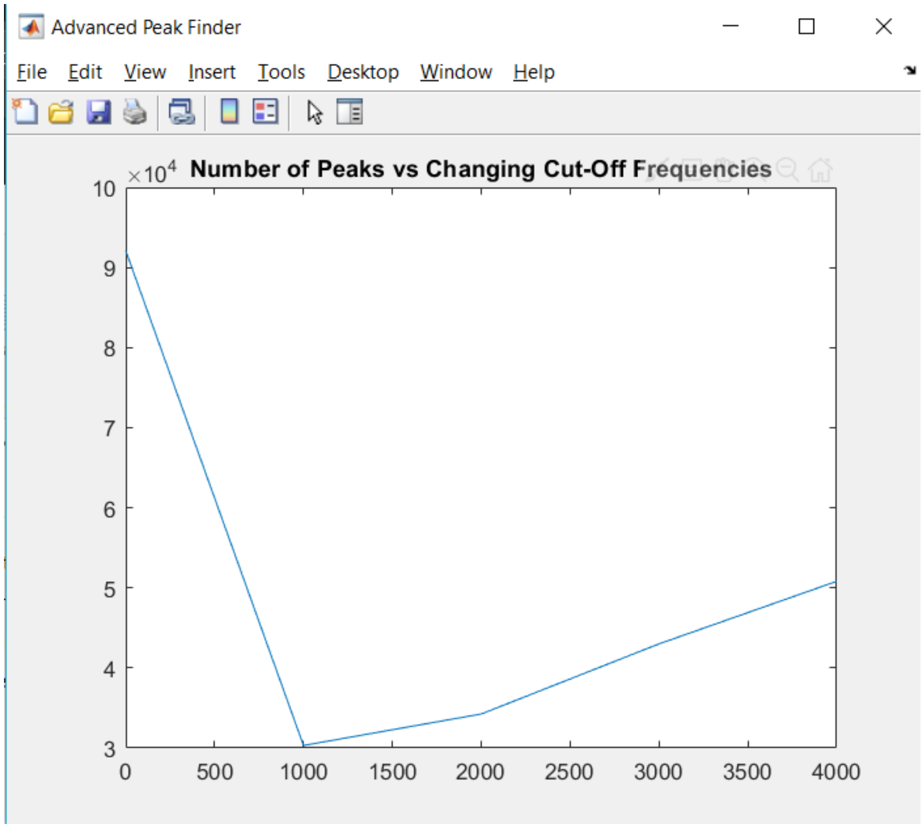
Then, for cutoff-frequencies (1000Hz,2000Hz,3000Hz,4000Hz) using built-in lowpass function, I got a filtered data, and using built-in findpeaks function I found number of peaks for that cutoff-frequency.

Finally, I plotted number of peaks versus changing cut-off frequencies.

2.3 Code

```
clc; clear;

% Set name of figure for problem 1
f = figure('Name', 'Advanced Peak Finder', 'NumberTitle', 'off');
figure(f);

hfile = 'PinkPanther30.wav';

% Read input file
[y, Fs] = audioread(hfile);

numOfPeaks = zeros(1, 5);

% Find peaks without moving average filter
peaks = findpeaks(y);

numOfPeaks(1) = length(peaks);

cutoff_freqs = 0:1000:4000;

for i = 2:5
    % Apply low pass filter
    output = lowpass(y, cutoff_freqs(i), Fs);

    % Find peaks
    peaks = findpeaks(output);

    % Save number of peaks
    numOfPeaks(i) = length(peaks);
end

% Plot number of peaks versus changing cut-off frequencies
plot(cutoff_freqs, numOfPeaks);
title('Number of Peaks vs Changing Cut-Off Frequencies');
```

2.4 Output



x axis is cutoff frequency, y axis is number of peaks that I found. For N = 0, it shows the number of peaks without low pass filter.

2.5 Result

As we can see from the gure above, the number of peaks differs for different cut-off frequencies. When no filter applied, we can observe number of peaks around 90k. However when the cutoff-frequency changes from 0 to 1000Hz, we notice that number of peaks is decreases. It can caused from the fact that low pass filter eliminates some data points, so it decreases.

As we increase the cut-off frequencies we observed that number of peaks increases as expected, due to the fact that as cut-off frequency increases we eliminate less number of data point from the original data.

3 Converting a Hubble Deep Space Image Into Space Sound

3.1 Question Description

In this part, I am expected to convert a Hubble deep space image into space sound. The image named as Hubble Massive Panorama. I am expected to examine the image as columns. Starting with the first column I am expected to examine each pixel whether it is black or not.

There are 900 pixels in each column. If a pixel in the column is not black, it means there is a celestial body in the pixel. Divide 900 pixel in the column into 10 parts. Starting from the bottom of the picture, index parts in decreasing order from 10 to 1. If a non-black pixel exists in any part, create an amplitude in the nth pixel frequency with index of the part amplitude. If a pixel is black, create a zero amplitude in the corresponding frequency.

By applying these rules, I have different amplitudes from 0 to 900 Hz. After I create 1024 spectra that have different amplitudes from 0-900Hz, convert each of them into time domain. Duration of each spectrum must be 1 second. Concatenate each one-second wav files one after another and create a wav signal that has length of 1024 seconds.

3.2 Implementation

First I read the image file, and converted it into binary image in order to decide whether pixel is black or not.

Then, for all columns I divided it into 10 parts, there are 90 rows in each part. Part index describes the amplitude for signal, row index describes the frequency for signal.

Then, due to the fact that I converted image into binary image, using this benefit, I can understand whether the given pixel is black or not. If it not black, I filled the appropriate index with related amplitude in the spectra. For each column, there is sound signal whose length is 1 second.

After, creating 1024 spectra, I converted each of them into time domain using inverste fast fourier transform an then concatenate them and created a sound named **deepspace.wav** which has length 17 minutes 4 seconds.

3.3 Code

```
clc; clear;

% Read image
img = imread('Hubble-Massive-Panorama.png');
% Convert image to binary
blackImg = imbinarize(rgb2gray(img));

i = 1;
y = zeros(1024, 900);

% For all coloumns
while( i <= 1024 )
    j = 1;
    % For all parts
    while( j <= 10 )
        k = 1;
        % For all rows in part
        while( k <= 90 )
            rowIndex = 90 * (j-1) + k;
            isWhite = blackImg(rowIndex, i);
            % If not black fill appropriate spectra
            if( isWhite )
                y(i, 901-rowIndex) = j;
            end
            k = k + 1;
        end
        j = j + 1;
    end
    i = i + 1;
end

Z = zeros(900*1024, 1);
i = 1;
while( i<= 1024)
    temp = ifft(y(i,:), 'symmetric');
    j = 1;
    while(j <= 900)
        Z(900*(i-1) + j) = temp(j);
        j = j + 1;
    end
    i = i + 1;
end

% Construct .waw file
```

```
audiowrite('deepspace.wav', Z, 900);
% Play the sound
sound(Z, 1000);
```

4 Conclusion

In this homework, I implemented some simple time domain exercises using MATLAB. I was expected to answer 2 questions, named Advanced Peak Finder, Converting a Hubble Deep Space Image Into Space Sound respectively. I implemented each questions to separate MATLAB scripts.

Thanks to this project, I learned how to code some simple functionalities in MATLAB like if, for, while etc. Additionally, due to the this project, now I have more knowledge about lowpass filtering since I had a opportunity to have hands-on experience about the facts that I learned during lectures.

Moreover, due to the this project, now I can create a sound from any image using this script that I implemented.