CMPE362 Homework 3

İrem Uğuz
2015400165

May 26, 2019

1 Advanced Peak Finder

In this question, we were asked to improve our peak finder algorithm to find the number of peaks in the filtered signal. The signal must be filtered by a low-pass filter that has cutoff frequencies 1k,2k,3k,4k Hz.

To achieve that, I used Matlab's built-in function lowpass(). I actually found another function that filters a signal with a cutoff frequency but the results were different and I decided to use the lowpass() function as that seemed more accurate. Name of the other function was dsp.VariableBandwidthFIRFilter('CutoffFrequency',1e3). But that function was designed for creating a filter with non-stable cutoff frequency.

In the resulting plot, it can be observed that the number of peaks that are found reduces highly after filtering the signal. Also, it can be observed that the number of peaks that are observed increases as the cutoff frequency increases from 1k Hz to 4k Hz. Of course, this is an expected trend as the higher frequencies tends to have more peaks.

The resulting plot of the number of peaks can be found below. The results are for the PinkPanther30.wav signal that was given with homework materials.

```
%AdvancedPeakFreqFilter.m
[x,Fs]=audioread("PinkPanther30.wav");
peaknums = zeros(5,1);
cutoffFreqs = [0; 1000; 2000; 3000; 4000];
c = findpeaks(x);
peaknums(1)= length(c);
for i = 2:5
    c = findpeaks(lowpass(x,cutoffFreqs(i),Fs));
    peaknums(i) = length(c);
end
plot(cutoffFreqs,peaknums);
```

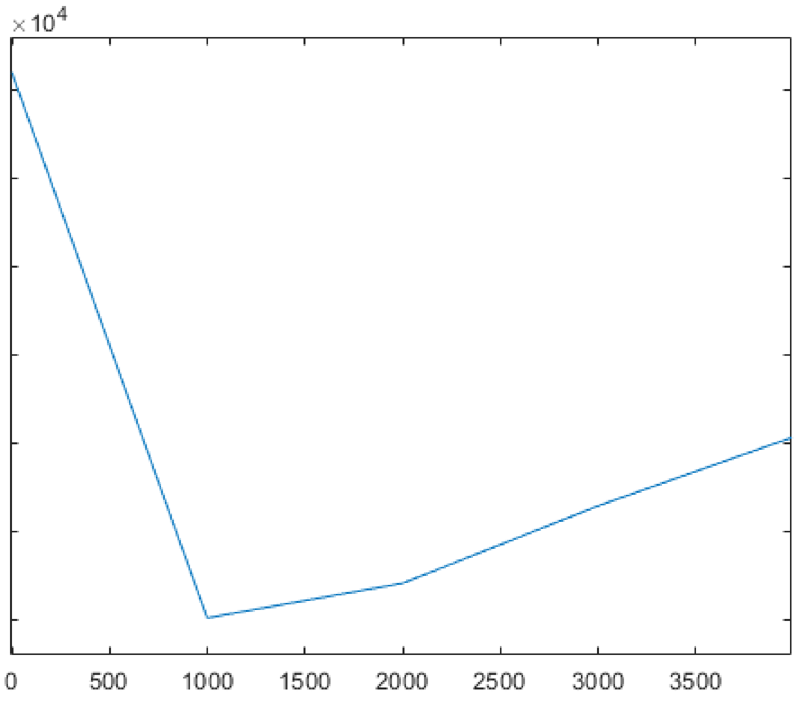

Figure 1: Graph of cutoff frequency versus number of peaks

2 Converting a Hubble Deep Space Image Into Space Sound

In this question, we were asked to convert a Hubble deep space image to a sound. To do that we were given a picture. I read it in my Matlab script by using imread function. Then to detect the non black pixels, I turned it to a gray picture by rgb2gray function and then binarize it with imbinarize function. In the imbinarized array, the pixels that are white are represented with true.

In the project we need to index the pixels, and give amplitudes to the pixel according to those indices. To do that I traversed the image and checked each pixel if the pixel is black or write. On white pixels I wrote the amplitude to the amplitude array.

After creating with an amplitudes, I created sin waves by those amplitudes and the frequencies for each column of the image and concatenated those waves into one. Then I play that concatenated wave and then put it into a wav file called result.wav so the re-play can be done.

```
%SonifiedDeepSpace.m
picture = imread('Hubble-Massive-Panorama.png');
graypic = rgb2gray(picture);
binarypic = imbinarize(graypic);
amplitudes = zeros(900,1024);
for column = 1:1024
```

```
    for index = 1:10
        for pixel = 1 : 90
            if(binarypic((index-1)*90+pixel,column)==true)
                amplitudes((index-1)*90+pixel,column)=index;
            end
        end
    end
end
Fs = 2000;
merged_audios = ones(1024*2000,1);
t = 0:1/Fs:1;
t = t.';
t(end)=[];
for i = 1:1024
    wave = zeros(2000,1);
    for j = 1:900
        sample = amplitudes(j,i).*sin(2*pi*j*t);
        wave = wave + sample;
    end
    merged_audios((i-1)*2000+1:i*2000) = wave;
end
sound(merged_audios,Fs);
audiowrite('result.wav',merged_audios,Fs);
```