BOĞAZİÇİ UNIVERSITY

CMPE 362: SIGNAL PROCESSING

Assignment 2

Huriye Özdemir - 2016400318

April 21, 2019

Contents

1 Peak Finder

1.1 Explanation

To improve the peak detection algorithm from the previous homework, I designed a Moving Average Filter. I changed the number of samples taken to average, N, from 2 to 30 for the moving average filter. I plotted the number of peaks versus the number of N. I've added the plot below.The peak numbers decrease as N rises, and it was expected because the more signal you take the more peaks you lose on average. But, as some peaks are meaningless, it gets more accurate.

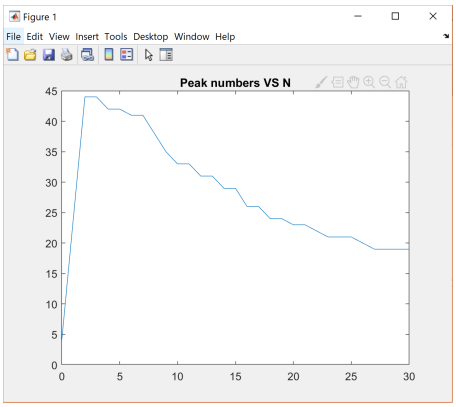

Figure 1: Peaks vs Changing Number of Samples Taken into Average

1.2 Code

You can find the code for question 1 below:
Details are explained in the comments.

```
clear; clc;

data = csvread('exampleSignal.csv');
% Remove first row of the read data
 Fs = data(1,1);

% Find nofilter peaks
nofilter = findpeaks(data, 'MinPeakProminence', 0.6, '
    MinPeakDistance', 200, 'MinPeakHeight', 3.5, 'MinPeakWidth',
    100);


% MOVING AVERAGE FILTERING
Sample_Maf = [0' 2:30];
NSample_Maf = length(Sample_Maf);
NPeaks_Maf = zeros(1,NSample_Maf);
```

```
% Add no filter result to first index
NPeaks_Maf(1) = length(nofilter);

for i = 2:NSample_Maf
    % Design moving average filter
    mf_filter = 1/i * ones(i,1);

    % Apply MAF
    mf_result = filter(mf_filter,1,data);

    % Find peaks
    peaks = findpeaks(mf_result, 'MinPeakProminence', 0.6, '
        MinPeakDistance', 200);

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

% Plot the result
figure;
plot(Sample_Maf, NPeaks_Maf); title('Peak numbers VS N');
```

2 Frequency(Pitch) of a Sound

2.1 Explanation

We changed the pitch and sound frequencies in this question. We also changed the frequency of sampling.

2.2 Similarities and Differences

Similar sounds for exercise 2 and 4. There is only a difference of clarity between them. In the second exercise, the frequency is halved and in the fourth exercise, the sampling frequency is halved. The sound is thicker than that. Exercise 3 and the first sound is similar.There is only a difference of clearness between them. We're getting rid of even numbered samples and playing the file so it's like doubling the frequency. In the third exercise, the sampling frequency is doubled. The sound is sharper. The difference is that exercise 1 is sharper than the rest of them. It is normal because the frequency is quadrupled.

2.3 Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    CMPE 362 Homework II-b    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                                        % Fs is the frequency = number of samples per second
                                        % y is the actual sound data
hfile = 'laughter.wav';                 % This is a string, corresponding to the filename
clear y Fs                              % Clear unneded variables

%% PLAYING A WAVE FILE

[y, Fs] = audioread(hfile);             % Read the data back into MATLAB, and listen to audio.
                                        % nbits is number of bits per sample
sound(y, Fs);                           % Play the sound & wait until it finishes

duration = numel(y) / Fs;               % Calculate the duration
```

```
17 pause(duration + 2)                                  % Wait that much +
       2 seconds
18
19 %% CHANGE THE PITCH
20
21 % sound(y(1:2:end), Fs);                              % Get rid of
       even numbered samples and play the file
22
23
24 %% EXERCISE I
25 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
26 %   Re-arrange the data so that   %
27 %   the frequency is quadrupled and play the file   %
28 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
29
30 sound(y(1:4:end), Fs);
31 pause(duration/4 + 2)
32
33 %% EXERCISE II
34 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
35 %   Re-arrange the data so that   %
36 %   the frequency is halved and play the file  %
37 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
38
39 sound(y(1:0.5:end), Fs);
40
41 pause((numel(y) / Fs) * 2 + 2)
42
43
44 %% EXERCISE III
45 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
46 %   Double Fs and play the sound  %
47 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
48 sound(y, Fs * 4);
49 pause(duration/2 + 2)
50
51
52 %% EXERCISE IV
53 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
54 %   Divide Fs by two and play the sound  %
55 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
56 sound(y, Fs / 2);
57 pause(2*duration + 2)
```

3 N-Tap Filter

3.1 Explanation

I applied a code and an N-Tap filter (explained in the figure below) to the mike.wav. signal. We'll mix the signal with the delayed version and then try to get it back with N-Tap Filter.

Constant N and K, change from 0 to 1
N and K are constant, changes from 0 to 1.
When alpha is increased the SNR value is increased too, but until some point. After 0.9 the value decreases.
Constant and K, change N from 1 to 50
a and K are constant, N changes from 1 to 50.
Increasing the N, which means adding more delayed input increases the SNR up to some point but after it doesnt do any effect.
Constant and N, change K
a and N are constant, K is 100,200,300,400.
Increasing the delay time first slowly increases the SNR but then improves the algorithm very extremely.

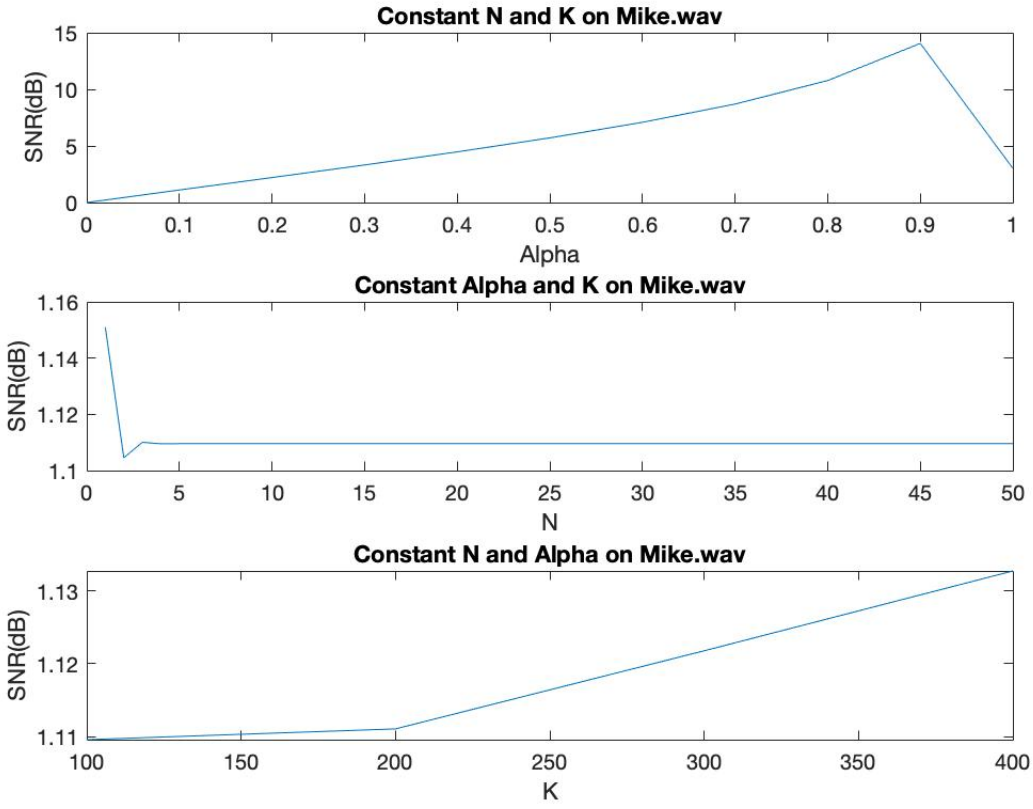

Figure 2: Constant a, N

3.2 Code

You can find the code for question 3 below:
Details are explained in the comments.

```
%Load Files
[wav,Fs] = audioread('mike.wav');

%% Problem 1

 Z= zeros(9,2);
for a=0.1:0.1:0.9
    y = ntap(wav,Fs,50,a,500);
    Z(uint8(a*10),1) = a;
    Z(uint8(a*10),2) = snr(wav,y);
end
```

```
12
13 figure('Name','Problem 1');
14 subplot(3,1,1)
15 plot(Z(:,1),Z(:,2))
16 title('Constant N and K')
17 xlabel('Alpha')
18 ylabel('SNR(dB)')
19
20
21 %% Problem 2
22
23 Z_2 = zeros(50,2);
24 for N=1:50
25     y = ntap(wav,Fs,N,0.8,100);
26     Z_2(uint8(N),1) = N;
27     Z_2(uint8(N),2) = snr(wav,y);
28 end
29
30 subplot(3,1,2)
31 plot(Z_2(:,1),Z_2(:,2))
32 title('Constant Alpha and K ')
33 xlabel('N')
34 ylabel('SNR(dB)')
35
36
37 %% Problem3
38
39 Z_3 = zeros(4,2);
40 for K=100:100:400
41     y = ntap(wav,Fs,40,0.3,K);
42     Z_3(uint8(K/100),1) = K;
43     Z_3(uint8(K/100),2) = snr(wav,y);
44 end
45
46 subplot(3,1,3)
47 plot(Z_3(:,1),Z_3(:,2))
48 title('Constant N and Alpha ')
49 xlabel('K')
50 ylabel('SNR(dB)')
51
52 %%%%%%%%%%%%%%%%%% Define Functions %%%%%%%%%%%%%%%%
53
54 function y = ntap(wav,Fs,N,a,K)
55
56 if nargin < 5
57     K = 100;
58 end
59
```

```
for i=1:N

    if mod(i,2) == 0
        a = power(a,i);
    else
        a = -(a^i);
    end


    wav = [wav;zeros(K*0.001*Fs,1)] + mul(delay(wav,K,Fs),a);

end

y = wav;

end

function y = mul(x,a)
    y = x.*a;
end

function result = snr(x,y)

    x = [x;zeros(size(y,1)-size(x,1),1)];

    x2 = x.^2;
    xy2 = (y-x).^2;
    up = sum(x2);
    down = sum(xy2);
    result = 10*log(up/down)/log(10);

end
function y = delay(x, K, Fs)

    y = [ zeros(K*0.001*Fs,1) ; x ];

end
```