CMPE362: Introduction to Signals for Computer Engineers

Homework-2

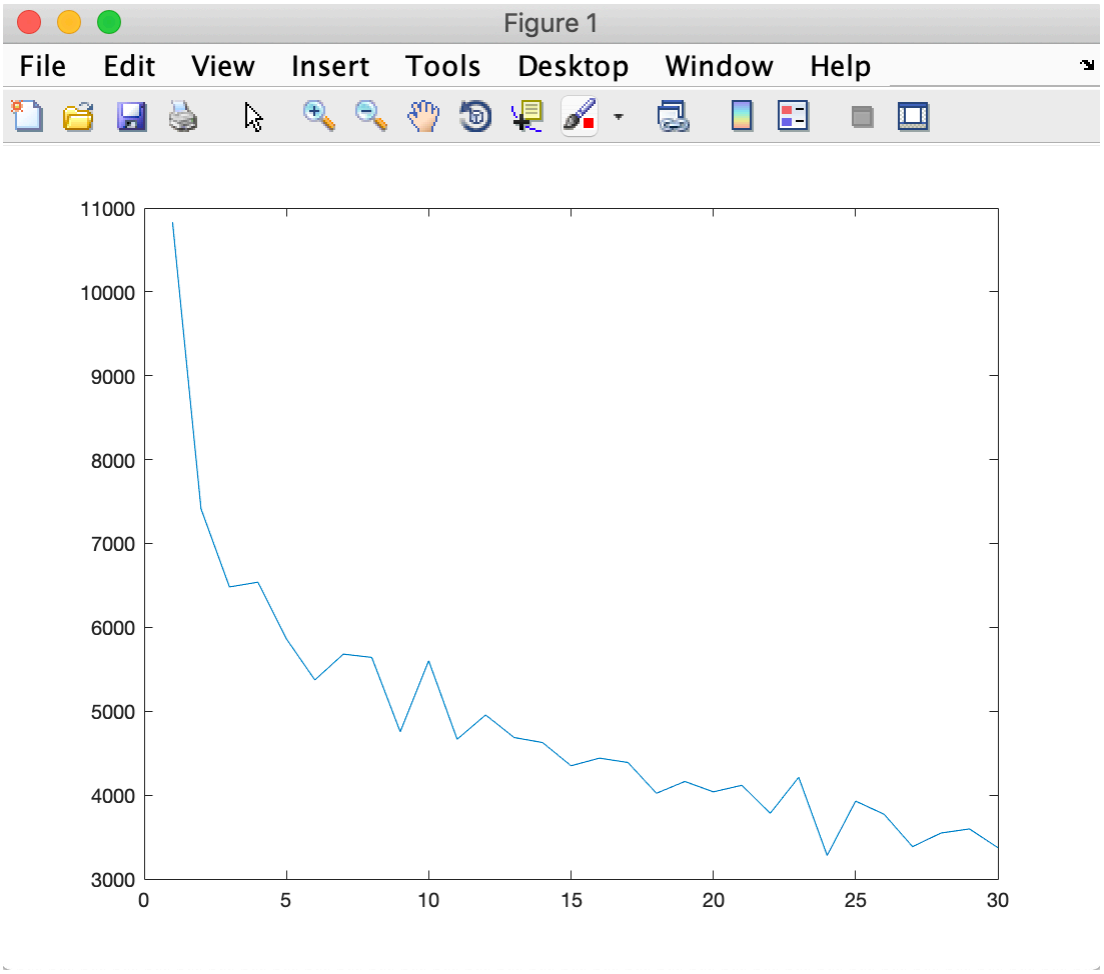
Yağmur Ceren DARDAĞAN
2014400063

Question 1

I explained the code in comments.

```
clear;
%reading data with specifying cells in csv file
data = dlmread('exampleSignal.csv',',','A4..A49502');
%peakArray will keep the peak results after the filter applied
peakArray = zeros(1,30);

%first result will be without applying the filter
peakArray(1) = numel(findpeaks(data));

%calculate the averages of windows at given number of windows
%moving_average_filter function will filter the data and return
%filtered data array

for window = 2:30
   filtered_data = moving_average_filter(data,window);
   peakArray(window) = numel(findpeaks(filtered_data));

end
%plotting the resulting peak numbers versus window size
x = 1:1:30;
plot(x,peakArray);


%filter function to take average at a given given windows size
function temp_array = moving_average_filter(data,window_size)
    array_length = size(data,1)-window_size+1;
    temp_array = zeros(1,array_length);
    for k=1:array_length
        temp = 0;
        for m=k:(k+window_size-1)
            temp = temp + data(m,1);
        end

        temp = temp./ window_size;
        temp_array(1,k) = temp;
    end

end

```

Result: Number of peaks are decreasing gradually after the window size 2-3. However it has small changes after window size 2-3 but it is smaller in average.



Question 2

```
%% EXERCISE II
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Re-arrange the data so that   %
%   the frequency is halved and play the file  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

y2 = y;
for i=1:length(y)
    y2(2*i-1) = y(i);
    y2(2*i) = y(i);
end
sound(y2, Fs);


%% EXERCISE III
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Double Fs and play the sound  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

sound(y, 2*Fs);
%% EXERCISE IV
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Divide Fs by two and play the sound  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

sound(y, Fs/2);

```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%
%   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
pause(duration + 2)                         % Wait that much + 2 seconds

%% CHANGE THE PITCH

sound(y(1:2:end), Fs);                      % Get rid of even numbered samples and play the file


%% EXERCISE I
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Re-arrange the data so that   %
%   the frequency is quadrupled and play the file   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

y = y(1:4:end);
sound(y,Fs);

```

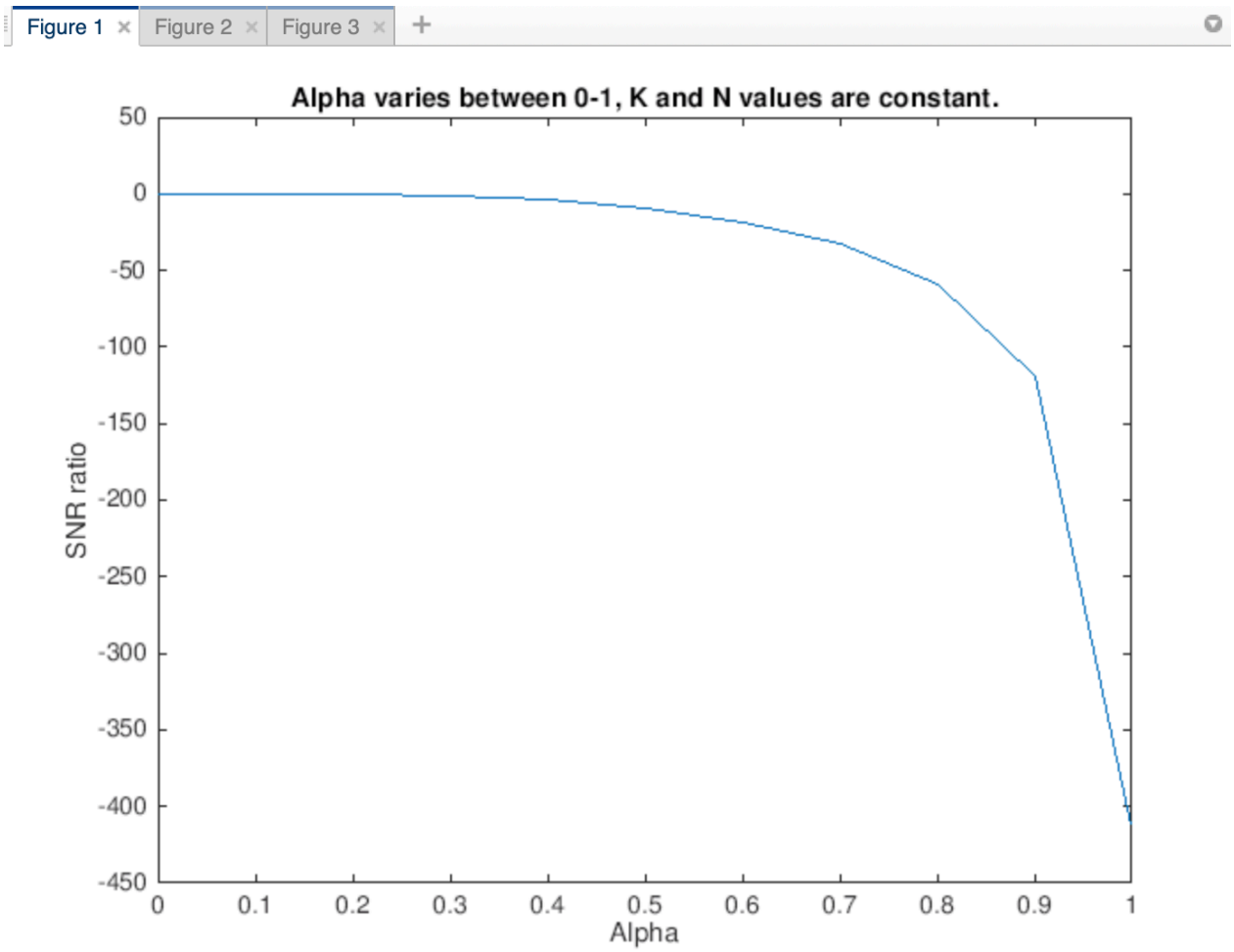
Question 3

Functions

```
%next_signal_array arranges the delay of the original signal by multiple of
%100 ms delay.
function delay = next_signal_array(original_signal,k_value,n_value)
        temp=zeros(1,n_value*k_value);
        delay = [temp original_signal'];

end

%rearrange_column function arranges the column of the signal to be combined
%to make calculations with delayed array
function signal_to_be_combined_new = rearrange_column(signal_to_be_combined,next_signal)
        signal_to_be_combined_new = signal_to_be_combined ;
        signal_to_be_combined_new(numel(next_signal))=0;

end

%makeTap function adds original signals and the delayed versions of the
%signal
function tapped_Array = makeTap(signal_to_be_combined,next_signal,n_value,alpha,size)
        tapped_Array = signal_to_be_combined+((-1*alpha)^n_value)*next_signal;
        tapped_Array = tapped_Array(1:size);

end

```

First Part

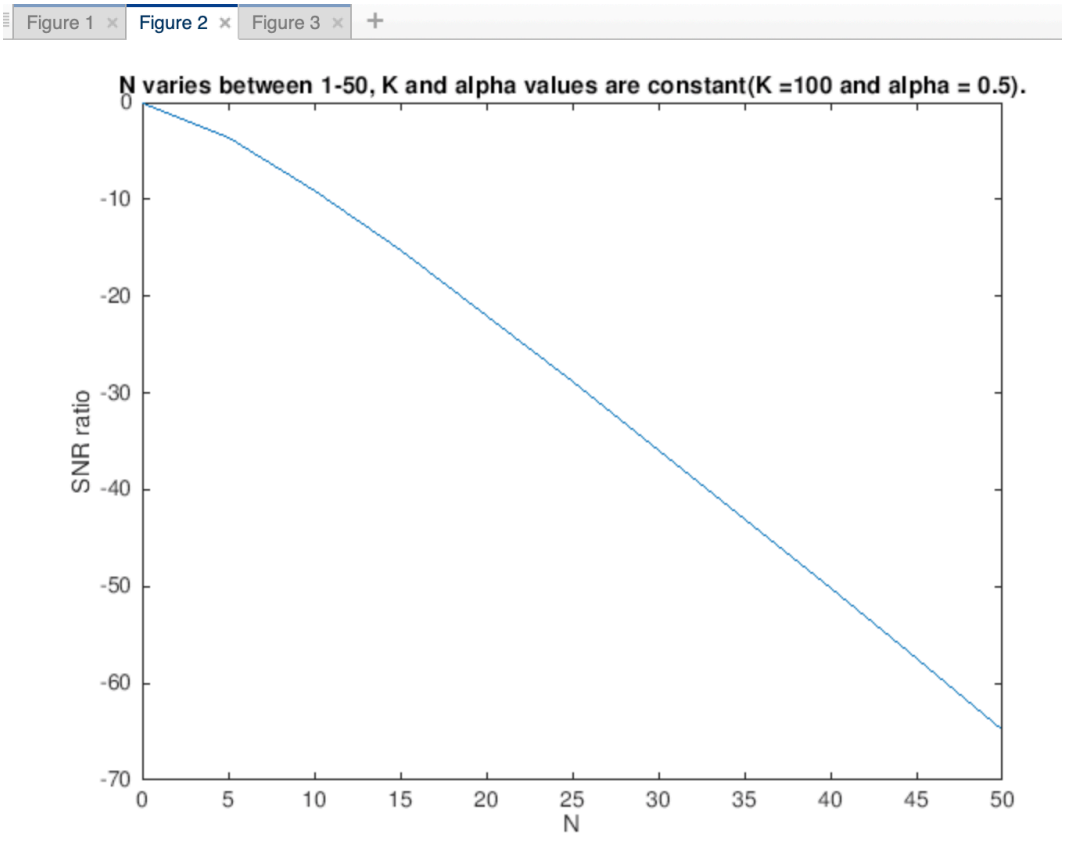
```
clear;
hfile = 'mike.wav';

[mike_original, Fs] = audioread(hfile);
signal_to_be_combined=mike_original;
size=length(mike_original);

%K and N values are constant. I examine the change of alpha values in every
%value of N between 1-50.
%next_signal_array arranges the delay of the original signal by multiple of
%100 ms delay.
%rearrange_column function arranges the column of the signal to be combined
%to make calculations with delayed array
%makeTap function adds original signals and the delayed versions of the
%signal
%I used built-in snr ratio function to make calculation of the snr ratio
K=100;
SNR_ratio_1=[];
for alpha=0:0.1:1
    for N = 1:50
       next_signal = next_signal_array(signal_to_be_combined,K,N);
       signal_to_be_combined = rearrange_column(signal_to_be_combined,next_signal);
       signal_to_be_combined = makeTap(signal_to_be_combined,next_signal,N,alpha,size);


    end
     snrVal=snr(mike_original,signal_to_be_combined);
     SNR_ratio_1=[SNR_ratio_1 snrVal];

end
%plotting the snr ratio versus alpha
figure
alpha=(0:0.1:1);
plot (alpha,SNR_ratio_1)
xlabel('Alpha')
ylabel('SNR ratio')
title('Alpha varies between 0-1, K and N values are constant.')

```



As we see, after alpha = 0.6 approximately, filter start to fail reconstruct signal.

Second Part

```
%I fixed the alpha value at 0.5 value and K is 100 ms and constant and N
%value is varies between 1-50 and two for loops builts filter and
%calculates the SNR raios at each N value.
%next_signal_array arranges the delay of the original signal by multiple of
%100 ms delay.
%rearrange_column function arranges the column of the signal to be combined
%to make calculations with delayed array
%makeTap function adds original signals and the delayed versions of the
%signal
%I used built-in snr ratio function to make calculation of the snr ratio
alpha=0.5;   %make alpha constant at some value
SNR_ratio_2=[];
signal_to_be_combined=mike_original;
%N varies between 1-50
%temp value is temporary tap value
for N=1:5:51
    for temp=1:N
       next_signal = next_signal_array(signal_to_be_combined,K,temp);
       signal_to_be_combined = rearrange_column(signal_to_be_combined,next_signal);
       signal_to_be_combined = makeTap(signal_to_be_combined,next_signal,temp,alpha,size);

    end

     snrVal=snr(mike_original,signal_to_be_combined);
     SNR_ratio_2=[SNR_ratio_2 snrVal];


end
%plotting SNR ratio versus N values
figure
disp(SNR_ratio_2)
N=(0:5:50);
Enter a value for txt tio_2)
ylabel('SNR ratio')
title(txt,options)
title('N varies between 1-50, K and alpha values are constant(K =100 and alpha = 0.5).')
```



Third Part

```
%I fixed the alpha value at 0.5 value and N is 50 and constant and k
%value is varies between 100-400 and I examine in each 100 ms and two for loops builts filter and
%calculates the SNR raios at each K delay value.
%next_signal_array arranges the delay of the original signal by multiple of
%100 ms delay.
%rearrange_column function arranges the column of the signal to be combined
%to make calculations with delayed array
%makeTap function adds original signals and the delayed versions of the
%signal
SNR_ratio_3=[];
N=10;
alpha=0.5;
signal_to_be_combined=mike_original;
for K=100:100:400
    for N = 1:50
       next_signal = next_signal_array(signal_to_be_combined,K,N);
       signal_to_be_combined = rearrange_column(signal_to_be_combined,next_signal);
       signal_to_be_combined = makeTap(signal_to_be_combined,next_signal,N,alpha,size);


    end

     snrVal=snr(mike_original,signal_to_be_combined);
     SNR_ratio_3=[SNR_ratio_3 snrVal];
end
figure
K=[100 200 300 400]
plot (K,SNR_ratio_3)
xlabel('K Values')
ylabel('SNR Values')
title('Alpha and N are constant. K is between 100 and 400')

```