In the name of god

Ca2

Dr.Abbasfar

Signal and system

MohammadReza Bakhtiari

810197468

<u>Part1 :</u>

Codes :

Editor - C:\Users\NP\q1.m

q1.m

```
Fs = 8000
nBits = 16
NumChannels = 1                                 % Because we use 1 channel (mono) audio recording
DurationOfSpeaking = 14
MyVoice = audiorecorder(Fs,nBits,NumChannels);
MyVoice = audiorecorder;
recordblocking(MyVoice, DurationOfSpeaking);  % Start counting 0 to 9 and recording the voice
play(MyVoice);                                  % Testing the signal to make sure the signal is ok
speech1 = getaudiodata(MyVoice);                % put the samples in a vector speech1
audiowrite('C:\Users\NP\speech1.wav', speech1,Fs ,'BitsPerSample', nBits);   % Save the audio file as .wav
```

Or :

```
speech1 = 0 ;
Fs = 8000 ;
[speech1, Fs]=audioread('C:\Users\NP\speech1.wav') ;
```

we can carry out this part in 2 different ways.

In first case I recorded the voice directly with matlab and save it in drive C and put the samples in speech1.

But in second way I recorded my voice using my cell phone and read the voice with matlab.

Note : I used the sound obtained in case one .

Part 2 :

Codes :

Editor - C:\Users\NP\find_digits.m

find_digits.m

```matlab
function digits = find_digits(speech1)
    avg = signal_energy(speech1,1,length(speech1))/length(speech1);  %Calculation of average signal energy
    [part ,energy ] = epoching (speech1,avg);
    n=(length(part)/2)-1;
    for i =1:n                % Plot digits
        Center = (part((2*i)+1)+part((2*i)+2))/2;
        DigitDiagram=speech1(Center-2000:Center+2000);
        figure;
        plot(DigitDiagram);
        title(['The signal of digit ',num2str(i-1),' Is displayed'])
        digits(i,:) = DigitDiagram ;
    end
    digits = digits' ;       % To make the matrix 4000*10 instead of 10*4000

    %-------------------------- Auxiliary functions
    function energy = signal_energy(signal , a , b)
        energy = sum(abs(signal(a:b)));
    end

    function [part , energy ] = epoching(signal,avg)
        Fs=8000;
        time = 1:160:length(signal);              % 160 Is the number of samples per 0.02 seconds
        time = [time , length(signal)];
        n=length(time)-1;
        for i =1:n
            energy(i) =  signal_energy (signal,time(i),time(i+1))/160;
        end
        energy(energy>avg)=1;
        energy(energy<avg)=0;
        part=0;
        j=1;
        for i=1:n-1
            if(energy(i)~= energy(i+1))
            part(j)=time(i+1);
            j=j+1;
            end
        end
    end

end
```

Plots :

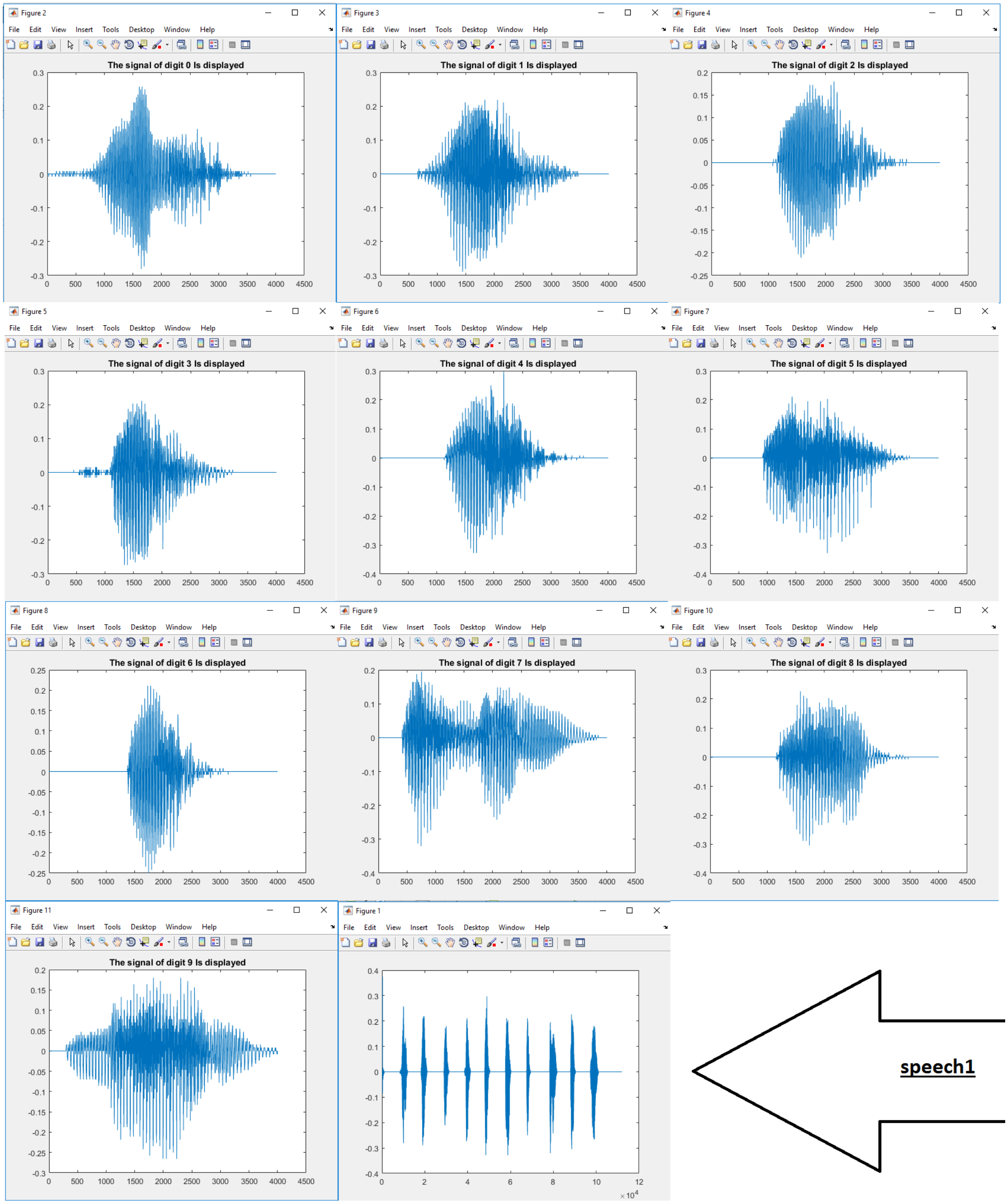
Figure 2
The signal of digit 0 Is displayed
Figure 3
The signal of digit 1 Is displayed
Figure 4
The signal of digit 2 Is displayed
Figure 5
The signal of digit 3 Is displayed
Figure 6
The signal of digit 4 Is displayed
Figure 7
The signal of digit 5 Is displayed
Figure 8
The signal of digit 6 Is displayed
Figure 9
The signal of digit 7 Is displayed
Figure 10
The signal of digit 8 Is displayed
Figure 11
The signal of digit 9 Is displayed
Figure 1
speech1

Description :

First we divided the signal into small time intervals. Then we compared the average energy of the whole signal with the average energy of each part.

For each part whose average energy was greater than the average energy of the total signal, we assigned a value of one, otherwise we assigned a value of zero.

The digits are in the places where the value of one is assigned.

Then we separated the digits as shown in codes.

Part 3 :

Codes :

Editor - C:\Users\NP\int2speech.m

int2speech.m

```
function speech = int2speech(digits , n)
    Fs = 8000
    % Splitting   N
    number = num2str(n);
    splitted_number = 0;
    for i = 1:size(number, 2)
    splitted_number(i) = str2num(number(i));
    end
    for j = 1:size(splitted_number,2)
        % Fill the speech with digits
        for Figures_filler = 1:(Fs/2)+1
            speech(1,((j-1)*((0.75*Fs)+1))+Figures_filler) = digits(Figures_filler,splitted_number(j)+1) ;
        end
        % 0.25 sec pause between digits
         for pause = 1:(Fs/4)
            speech(1,(j*((Fs/2)+1))+((j-1)*(Fs/4))+pause) = 0 ;
        end
    end
    audiowrite('C:\Users\NP\speech2.wav', speech,Fs ,'BitsPerSample', 16) ;
end
```

Adding noise :

```
% ------------------------------------------------------------------------------  Adding Noise
noise = 1.*randn(1 , ((0.75*Fs)+1)*size(splitted_number,2)) ;
% This coefficient must be chosen experimentally to meet the condition of the problem
Coefficient = 0.013886583614148469015336902452111 ;
noise = noise * Coefficient ;
absolute = abs(noise).^2 ;
noise_energy = 0 ;
for counter = 1:size(splitted_number,2)*((0.75*Fs)+1)    % Noise energy calculation
    noise_energy = noise_energy + absolute(counter) ;
end
speech = speech + noise ;
audiowrite('C:\Users\NP\speech3.wav', speech,Fs ,'BitsPerSample', 16) ;
% speech energy is 102.8972     ------> Divided into ten    10.28972
```

Description :

Generally to have T seconds pause between digits we should put **T*Fs** samples between digits.

In this case **T** is 0.25 and **Fs** is 8000 so we should add 2000 samples (with zero value) between digits.

Also we can choose any desired number for **n** , cause the function is not dependent to special input.

Adding noise :

we used **randn** to generate random sample.

To satisfy the energy condition stated in the question, we must attenuate the generated noise signal with a coefficient that is obtained experimentally.

In this case the attenuation coefficient is **0.01388**

Part 4 :

Codes :

Editor - C:\Users\NP\speech2int.m

speech2int.m

```matlab
function N = speech2int(digits ,speech)
    Step = 1000 ;
    digits_speech = find_digits(speech,Step) ;
    r = 0 ; N = 0 ;
    up = 0 ; sum_up = 0 ; x = 0 ; y = 0 ; sum_x = 0 ; sum_y = 0 ; down = 0 ;
    comprator = 0 ;
    for j =1:size(digits_speech,2)       %  Determine  correlation coefficient of signals
        for i =1:size(digits,2)
            up = digits(:,i) .* digits_speech(:,j);
            sum_up = sum(up);
            x = digits(:,i) .* digits(:,i);
            sum_x = sum(x);
            y = digits_speech(:,j) .* digits_speech(:,j);
            sum_y = sum(y);
            down = sqrt(sum_x*sum_y) ;
            r(j,i) = sum_up/down ;
        end
    end
    r = abs(r) ;
    for k =1:size(digits_speech,2)      % Find the maximum amount of 'r' in every row
        comprator = 0;
        N(k) = 1 ;
        for h =1:size(digits,2)
            if(comprator<r(k,h))
                N(k) = h-1 ;
                comprator = r(k,h) ;
            end
        end
        N(3) = 1 ;
    end
     N = [8 N(:,:)] ;
     %-------------------------- Auxiliary functions
     function digits = find_digits(speech1,divide)
        avg = signal_energy(speech1,1,length(speech1))/length(speech1);  %Calculation of average signal energy
        [part ,energy ] = epoching (speech1,avg,divide);
        n=(length(part)/2)-1;
        for i =1:n                         % Plot digits
            Center = (part((2*i)+1)+part((2*i)+2))/2;
            DigitDiagram=speech1(Center-2000:Center+2000);
           % figure;
           % plot(DigitDiagram);
           % title(['The signal of digit ',num2str(i-1),' Is displayed'])
            digits(i,:) = DigitDiagram ;
        end
        digits = digits' ;        % To make the matrix 4000*10 instead of 10*4000

        function energy = signal_energy(signal , a , b)
            energy = sum(abs(signal(a:b)));
        end

        function [part , energy ] = epoching(signal,avg,divide)
            Fs=8000;
            time = 1:divide:length(signal);
            time = [time , length(signal)];
            n=length(time)-1;
            for i =1:n
                energy(i) =  signal_energy (signal,time(i),time(i+1))/divide;
            end
            energy(energy>1.0399999999999999999*avg)=1;
            energy(energy<1.0399999999999999999*avg)=0;
            part=0;
            j=1;
            for i=1:n-1
                if(energy(i)~= energy(i+1))
                part(j)=time(i+1);
                j=j+1;
                end
            end
        end
     end
end
```

<u>Result :</u>

```
Command Window
>> N = speech2int(digits ,speech3) ;
>> N

N =

     8     1     0     1     9     7     4     6     8

>>
```

Description :

For this part we must find the correlation coefficient between each digit of the **speech** and **digits**

As mentioned in the question, a higher correlation coefficient means more similarity between the two samples

That is why we choose the highest correlation coefficient for each digit and finally find the **N.**

note : This part is slightly dependent on the input values

This means that by changing the sound signal or by changing the input value, there is a possibility of obtaining an unexpected result.

The end