

*Signals Project*

**ECG SIGNALS**

|  |  |  |  |
| --- | --- | --- | --- |
| محمد شعبان محمد حسن **Name:** | **Sec:**3 | **B.N**:44 | **Code:**9230775 |
| نبيل إبراهيم عبد العاطي عبد الرازق**Name:** | **Sec:**4 | **B.N:**26 | **Code:**9230945 |

**Submitted to : Dr Michael Abdel-Sayed**

**Requirement a:**

**OUTPUTA screen shot of a graph

Description automatically generated**

**Explanation**

We extracts the ECG signal and its sampling frequency, we calculates the total number of samples in the signal and create a time vector based on the sampling rate and create the first 10 seconds of the ECG signal using the sampling frequency to determinehow many samples are in that time and creates a corresponding time vector for this segment after that we plot the amplitude of the first 10 seconds of the ECG signal in the time domain against time and enhances the plot with axis labels, a title, and a legend for clarity and proper presentation of the da

**Requirement b:**

**OUTPUT**

A white background with black text

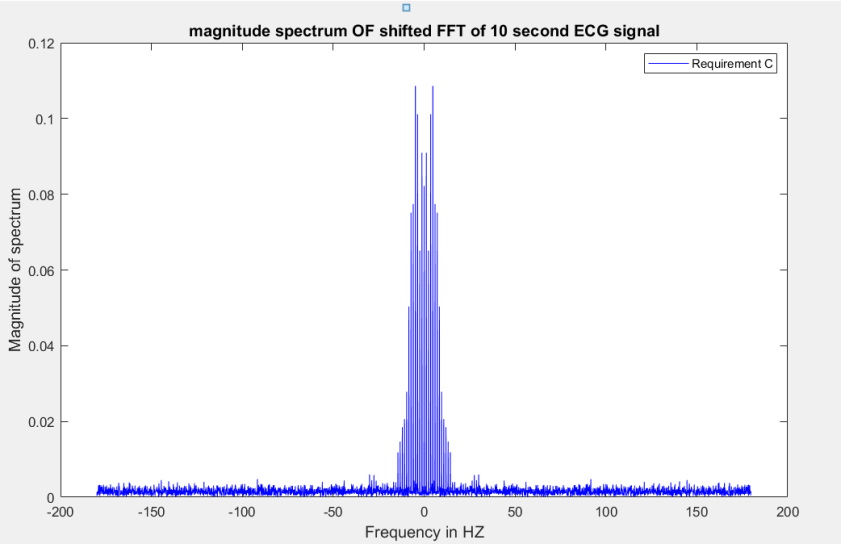
Description automatically generated

**Explanation**

In the first 10 seconds of the ECG signal and we will find the heartbeats as peaks with findpeaks and taking the minimum peak Highet to be 0.5 according to min BPM and minimum distance 0.5 times the sampling rate after that we get counts of valid peaks corresponding to heartbeats and finally computes the average heart rate in beats per minute by multiplying the number of the detected peaks by 6 due to the 10\_second time duration.

**Requirement c:**

**OUTPUT**



**Explanation**

We computes the fft of the signal and normalizes it with 10 second samples to converts it to frequency domain and shifts the frequency spectrum to center around zero using fftshift and finally plots the magnitude of the shifted fft against frequency in Hz to showing the frequency components of the ECG signal during 10 second .

**Requirement d:**

**OUTPUT**

**A screen shot of a graph

Description automatically generated**

**Explanation**

We filters the first 10 seconds of an ecg signal using a low-pass filter we will justify it below and plots the filtered signal in the time domain (not required) and computes fft and normalizes it by dividing it by the number of samples and shifts the frequency spectrum to center the zero frequency and plots the magnitude of the shifted fft against frequency showing frequency content of the filtered ecg signal.

**The filter design:**

**A screenshot of a computer

Description automatically generated**

**Justifiction**

The main content of signal ends at a frequency of 15 Hz so we Filtering an ECG with butterworth because it provides a smooth and clear signal with these frequencies because the heart’s main content are below 15 Hz, and higher frequencies often contain noise that can make the signal unclear and fpass 10 HZ it provided the best output due to non ideality response of filter

**Requirement E:**

**OUTPUT**

|  |  |
| --- | --- |
|  | * **Comparison**   **Avg\_BPM\_fd=Avg\_Bpm\_td**  **The average BPM in time domain same as frequency domain for the 10 second ecg signal** |

**Explanation**

in the frequency domain we need to get the average heart rate first we will get the valid frequency range from 0.5 to 3.33 Hz corresponding to the heart rate range of 30 to 200 BPM after that From this range we get vector of the frequencies and their amplitude responses to get the dominant frequency that have max response we will find the maximum amplitude in this range which corresponds to dominant frequency . we finally converts to frequency into beats per minute by multiply it by 60.

**Requirement f:**

**OUTPUT**

A graph of a signal

Description automatically generated

**Explanation**

first will plot the whole ECG signal against time(not required). After that I will use same low-pass filter we use before to process the signal and we will find peaks locations using findpeak function and set constraints on the height and distance of peaks to get accurate peak locations detection After finding the peaks locations

I will calculate the time interval between successive peaks using function diff and change into seconds by dividing it by sampling frequency. and get the Bpm values by divide 60 by intervals vector .we need to get time vector same size as BPM values to plot with each other

Note diff remove one place of locations vector

We will get time vector same length of locations of peaks subtracted by 1 due to diff function after that we plot

**Requirement g:**

**OUTPUT**

A graph with blue lines

Description automatically generated

**Explanation**

We applying STFT with a Hamming window to segment the signal and we  calculating the magnitude spectrum for each segment after that we get range of frequencies within the heart rate range of 0.5-2.5 Hz and we determine the  dominant frequency corresponding to the maximum amplitude in each segment and we  convert the dominant frequencies to beats per minute BPM to plot it with output time axes from stft

Note: we choose hamming after trying the all functions and hamming get the best response (best output )

**Comparison of BPM between time and frequency domain**

A graph of a graph

Description automatically generated

The same graph but analysis in frequency domain more accurately as shown in graph

**Justifying**

Time Domain Analysis:   
This approach provides precise, beat-to-beat measurements that reflect instantaneous variations in heart rate.

Frequency Domain Analysis:  
This approach smooths out short-term variations, hence providing an average over the analyzed window.

**Requirement h :**

**OUTPUT**

**A screenshot of a graph

Description automatically generated**

A white background with black text

Description automatically generated

**Explanation**

We will get the BPM values that are outside of normal range  greater than 100 or less than 60 in both time and frequency domains we extract the corresponding time points of those out-of-range values and we plots the original BPM values with the highlights on high and low values using different shapes after that we calculate the total time intervals during which the BPM remains high or low in both domains and, finally and displays the intervals

**All codes**

------------------AAAAAA-------------------

%reading the heart rete in fs for the signal ECG

data=load("ecg\_data.mat");

% get signal as a vector

ecg\_signal=data.ecg\_signal;

% get sampling rate from structure

fs=data.fs;

%calc the number of elements of signal

N=length(ecg\_signal);

%creating time vector

t=(0:N-1)/fs;

%creating vectors for first 10 seconds

N\_10s=10\*fs;

% time vector

t\_10s=t(1:N\_10s);

% get the 10 second part of all signal

ecg\_signal\_10s=ecg\_signal(1:N\_10s);

% plot the first 10 seconds of signl in time domain

plot(t\_10s,ecg\_signal\_10s);

%creating axes name and graph title

xlabel("Time(S)","FontSize",12);

ylabel("Amplitude of spectrum","FontSize",12);

title("First 10 seconds of the signal in time domain","FontSize",12);

legend("Requirement A");

------------------BBBBBBBB-------------------

% we want to know number of heart beats for the first 10 seconds

% each beat represnted by peak which is resposne to heart in 10 seconds

% so we will use function that find peaks and creating peaks vector

% so we will count peaks which is seprated by certain distance

% and count peaks which is more than cretain value according to MIN BPM 30

% this value(factor of the max peak value of 10 second signal )

% another peaks represent noise (we dont need it)

peaks\_10s=findpeaks(ecg\_signal\_10s,'MinPeakHeight',0.5,'MinPeakDistance', 0.5\*fs);

% we will get the length of vector peaks\_fts that mean number of beats

num\_of\_beats=length(peaks\_10s);

% after that multiply the number of beats by 6 to get the value per minutes

disp("Requirement B");

Avg\_BPM\_td=(num\_of\_beats)\*6

------------------CCCCCCCC-------------------

%creating the 10s freq vector that shifting center to zero

freq\_10s=(-N\_10s/2:N\_10s/2-1)\*fs/N\_10s;

%so we will get fast fourier transform

% after that we will normalize the result with N\_10s=3600;

ecg\_signal\_fft=fft(ecg\_signal\_10s)/N\_10s;

%and shift it to make the center is y axes with fftshift function

shifted\_10s\_signal=fftshift(ecg\_signal\_fft);

% ploting the absolute of first 10 seconds shifted ecg against frequency in HZ

plot(freq\_10s,abs(shifted\_10s\_signal))

% creating axes names

xlabel("Frequency in HZ",FontSize=12);

ylabel("Magnitude of spectrum",FontSize=12);

title("magnitude spectrum OF shifted FFT of 10 second ECG signal","FontSize",12);

legend("Requirement C");

%end

------------------DDDDDDDDD-------------------

% first we will design a low pass filter

load('filtermoussa.mat')

filtered\_10s\_signal=filter(moussa,ecg\_signal\_10s);

% we will plot the filtered signal in time domain

subplot(2,1,1);

plot(t\_10s, filtered\_10s\_signal);

% axes names

xlabel("Time (s)");

ylabel("Amplitude");

title("Filtered 10 second ECG Signal in Time domain");

% we will compute fft of 10s filtered signal and normalize it with N\_10s

filtered\_10s\_signalfft=fft(filtered\_10s\_signal)/N\_10s;

%we will shift the signal with fftshift

sh\_filtered\_10s\_signalfft=fftshift(filtered\_10s\_signalfft);

%we will plot the abs of shifted signal aganist freq\_10s

subplot(2,1,2);

plot(freq\_10s,abs(sh\_filtered\_10s\_signalfft))

%create axes names

xlabel("Frequency in HZ");

ylabel("Magnitude of filtered spectrum");

title(" magnitude OF Filtered shifted FFT of 10 second ECG signal");

legend("Requirement D")

------------------EEEEEEEEE---------------

% in frequency domain BPM =frequency \*60

% we know Maximum BPM is 200 BPM and let say the minumum BPM is 30 BPM

% so we need to search in postive frequencies the elements from 0.5-->3.33

% we will create the vector of range frequencies from 10\_s frequency vector

fundmental\_frequency\_range = freq\_10s(freq\_10s >=0.5 & freq\_10s <=3.33);

% we need to get the abs response value according to domainant frequency range

% note we use the 10\_s filtered signal frequency domain

spectrum\_of\_freq= abs(sh\_filtered\_10s\_signalfft(freq\_10s >=0.5 & freq\_10s <=3.33));

% after that we need to get the frequency component which is have max amp

% the maximum response at domainant freq in valid range

% first get the place of max response from spectrum\_of\_freq vector

% we will use function find to find the index at maximum spectrum

place\_of\_max = find(spectrum\_of\_freq == max(spectrum\_of\_freq));

% after that we will put the index of max spectrum to get the corresponding frequency

frequency\_of\_max\_spectrum = fundmental\_frequency\_range(place\_of\_max);

% why we do that becuase the Avg\_BPM in frequency domain

% Avg\_BPM =frequency\_of\_valid\_max\_spectrum\*60 to became rate per minute

disp("Requirement E ")

Avg\_BPM\_fd = frequency\_of\_max\_spectrum \* 60

-------------FFFFFFFF---------------

% take the whole signal

% plot of the whole signal against time vector we created before

subplot(2,1,1);

plot(t,ecg\_signal);

% create axes names

xlabel("Time(S)","FontSize",12);

ylabel("Amplitude of spectrum","FontSize",12);

title("whole signal in time domain","FontSize",12);

% now we will apply same filter we used before to whole signal

filtered\_all\_signal=filter(moussa,ecg\_signal);

% we need to get the locations peaks in filtered signal

% and control the the choosed peaks under cretain sizes

[~,loactions]=findpeaks(filtered\_all\_signal,"MinPeakHeight",0.5,"MinPeakDistance",0.5\*fs);

% we need to get the diffrenece between the time components

% we get diffrence using function diff

% dividing by the sampling frequency converts the difference from samples to seconds.

time\_intervals\_per\_second=diff(loactions)/fs;

% to calc the BPM vector we will divide 60 by time intervals we get before

BPM\_values=60./time\_intervals\_per\_second;

% we need to get time vector coressponding to intervals\_per second

% we start from index 2 due to diff start subtract from second element

time\_Bpm=t(loactions(2:end));

% plot the BPM values Against time

subplot(2,1,2);

plot(time\_Bpm,BPM\_values);

% create axes names

xlabel("Time(S)","FontSize",12);

ylabel("BPM","FontSize",12);

title("Graph of whole signal BPM Against Time","FontSize",12);

legend("Requirement F")

% end

-------------GGGGGGGGG---------------

% create window size and overlab 50%

%we need to cap 3 complete periods 3

wind\_L = 3 \* fs;

overlap\_L = wind\_L / 2;

% and now we need to make window function to length of it

% after trying we get hamming is best

win\_function = hamming(wind\_L);

% now we will divide the filtered all signal and get fourier tranasform to each part

% and we will get the time and frequency axes corresponding stft

[stft\_signal, freq\_stft, t\_stft ]= stft(filtered\_all\_signal, fs,'Window', win\_function, 'OverlapLength', overlap\_L, 'FFTLength', 1500);

% we will get the transform of transformed signal

mag\_stft = abs(stft\_signal);

% as we say before max and min heart rate is 150BPM\_\_30BPM

% we need to get range of freq 0.5--2.5 and corresponding amp

freq\_range = freq\_stft(freq\_stft >= 0.5 & freq\_stft <=2.6);

mags\_range = mag\_stft(freq\_stft >= 0.5 & freq\_stft <= 2.6, :);

% we need to take each time segment and search on it for max amp

% but we dont need values we need freq corresponding to maximum values

% so we will create domainant freq vector

[~, max\_locations] = max(mags\_range, [], 1);

domainant\_freq\_stft = freq\_range(max\_locations);

% bpm = 60 \* freq so we will get the BPM vlaues vector

BPM\_values\_stft= domainant\_freq\_stft \* 60;

% we will plot

figure;

plot(t\_stft, BPM\_values\_stft);

xlabel("Time (s)");

ylabel("BPM");

title("BPM graph in frequency domain using STFT")

legend("Requirement G");

% compared with time domain method

figure;

hold on

plot(time\_Bpm,BPM\_values,'r');

plot(t\_stft, BPM\_values\_stft,'b--');

title("compare of time domain method with freq domain method");

legend time-domain frequency-domain

-----------HHHHHHHHHHH------------

% in time domain BPM graph

% first we will get the vectors of out of range values for 2 domains

Bpm\_high=BPM\_values(BPM\_values>100);

Bpm\_low=BPM\_values(BPM\_values<60);

BPM\_high\_stft=BPM\_values\_stft(BPM\_values\_stft>100);

BPM\_low\_stft=BPM\_values\_stft(BPM\_values\_stft<60);

% and get time vector corresponding to out of range values to 2 domins

t\_high=time\_Bpm(BPM\_values>100);

t\_low=time\_Bpm(BPM\_values<60);

t\_high\_stft=t\_stft(BPM\_values\_stft>100);

t\_low\_stft=t\_stft(BPM\_values\_stft<60);

% first we plot the original BPM graph

% and we will hold the two graphs in 1 figure for 2 domains

subplot(2,1,1);

hold on;

% for time domain

plot(time\_Bpm,BPM\_values);

% we will plot the high and low values with different shapes to highlight it

plot(t\_high,Bpm\_high,"ro");

plot(t\_low,Bpm\_low,"go");

% axes names

xlabel("Time(S)","FontSize",12);

ylabel("BPM","FontSize",12);

title("highlight in time domain","FontSize",12);

legend("Requirement H",'Bpm high','Bpm low');

% for freq domain

subplot(2,1,2);

hold on

plot(t\_stft,BPM\_values\_stft);

% we will plot the high and low values with different shapes to highlight it

plot(t\_high\_stft,BPM\_high\_stft,"ro");

plot(t\_low\_stft,BPM\_low\_stft,"go");

% axes names

xlabel("Time(S)","FontSize",12);

ylabel("BPM","FontSize",12);

title("highlight in frequency domain","FontSize",12);

legend("Requirement H",'BPM high','BPM Low');

% after that we need to display the time intervals of low and high intervals

% we will get for 2 domains

fprintf("time domain\n");

fprintf(" start High BPM interval: %.4f\tto end High BPM interval: %.4f seconds\n", t\_high(1),t\_high(end));

fprintf(" start Low BPM interval: %.4f\tto end Low BPM interval: %.4f seconds\n", t\_low(1),t\_low(end));

fprintf(" time domain High BPM interval: %.4f seconds\n", t\_high(end)-t\_high(1));

fprintf(" time domain Low BPM interval: %.4f seconds\n", t\_low(end)-t\_low(1));

fprintf("frequency domain\n");

fprintf(" start High BPM interval: %.4f\tto end High BPM interval: %.4f seconds\n", t\_high\_stft(1),t\_high\_stft(end));

fprintf(" start Low BPM interval: %.4f\tto end Low BPM interval: %.4f seconds\n", t\_low\_stft(1),t\_low\_stft(end));

fprintf(" freuency domain High BPM interval: %.4f seconds\n",t\_high\_stft(end)-t\_high\_stft(1));

fprintf(" frequency domain Low BPM interval: %.4f seconds\n", t\_low\_stft(end)-t\_low\_stft(1));