Department of Electronics and
Electrical Communications Engineering
Faculty of Engineering - Cairo University

Signals Project

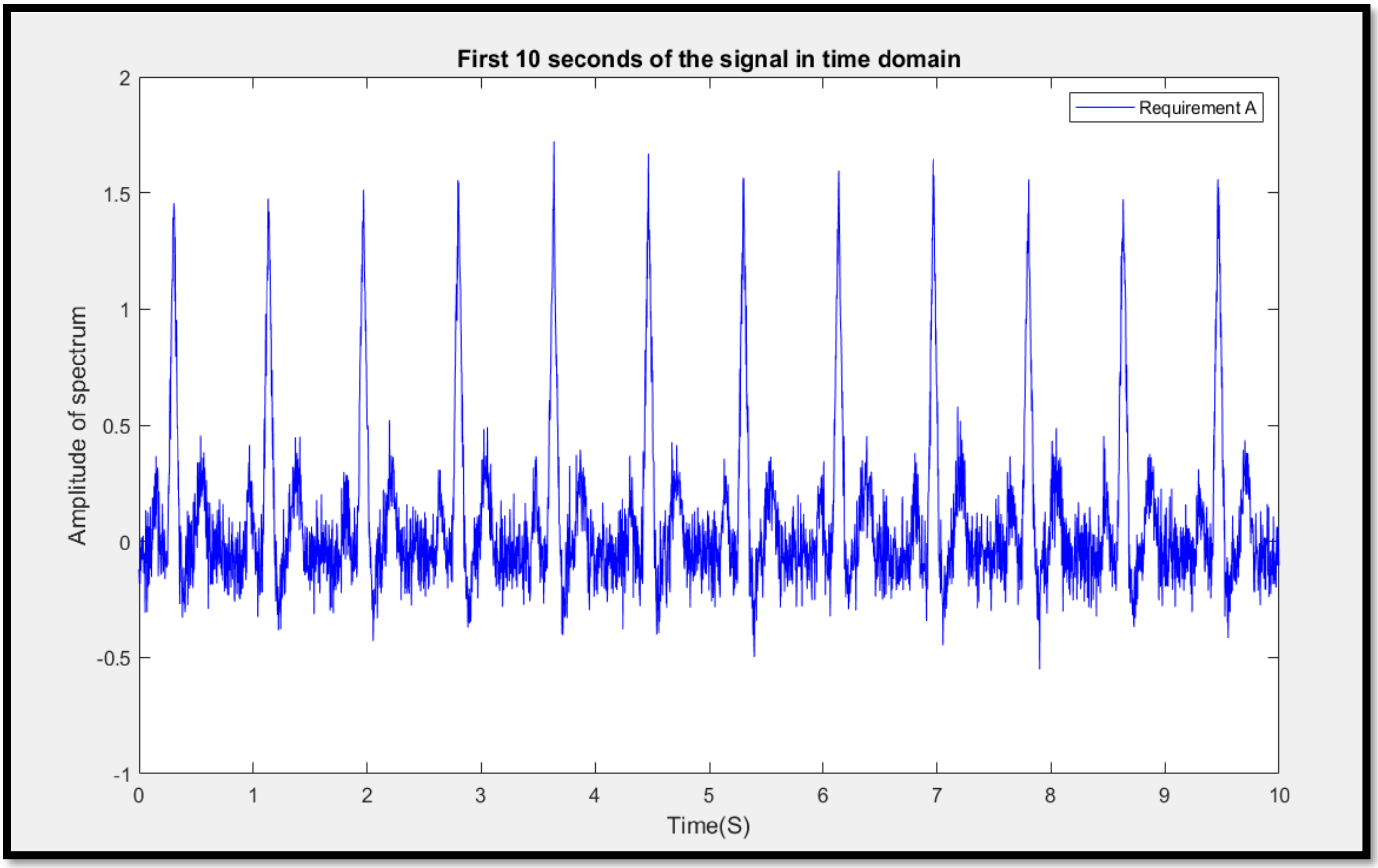
ECG SIGNALS

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

Submitted to : <u>Dr Michael Abdel-Sayed</u>

Requirement a:

OUTPUT



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 determine how 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

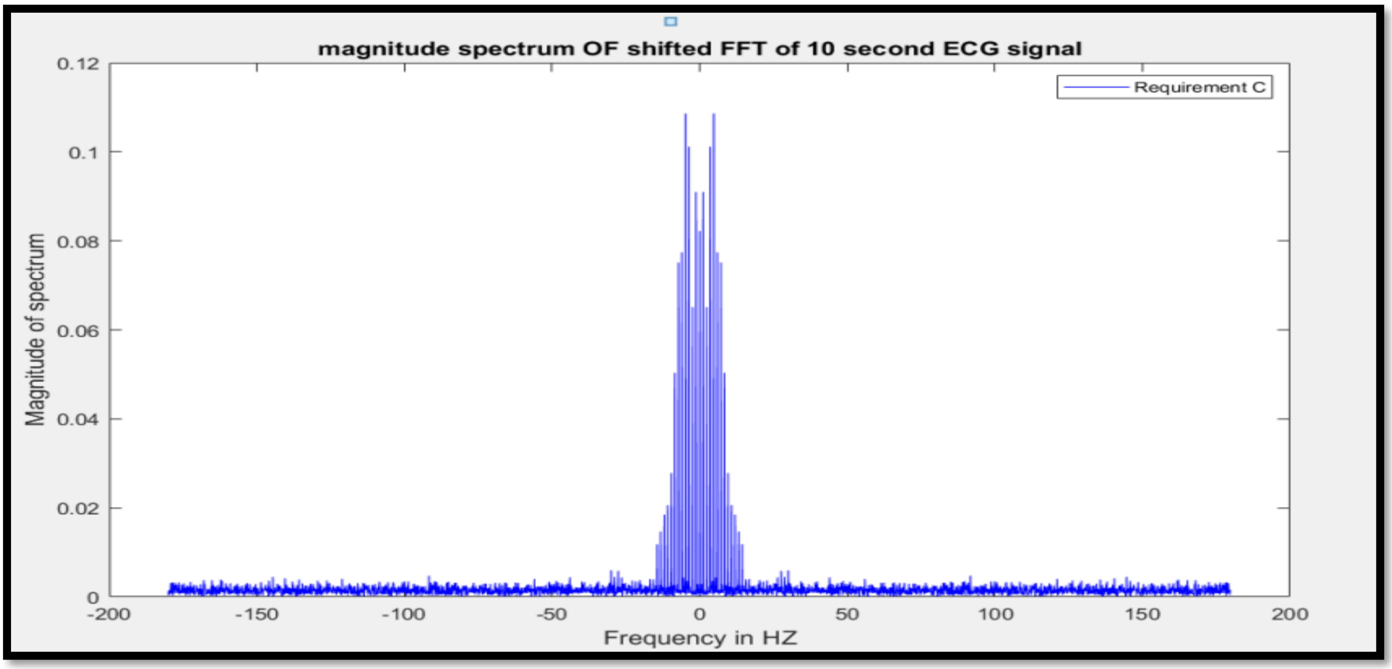
```
Requirement B

Avg_BPM_td =

     72
```

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

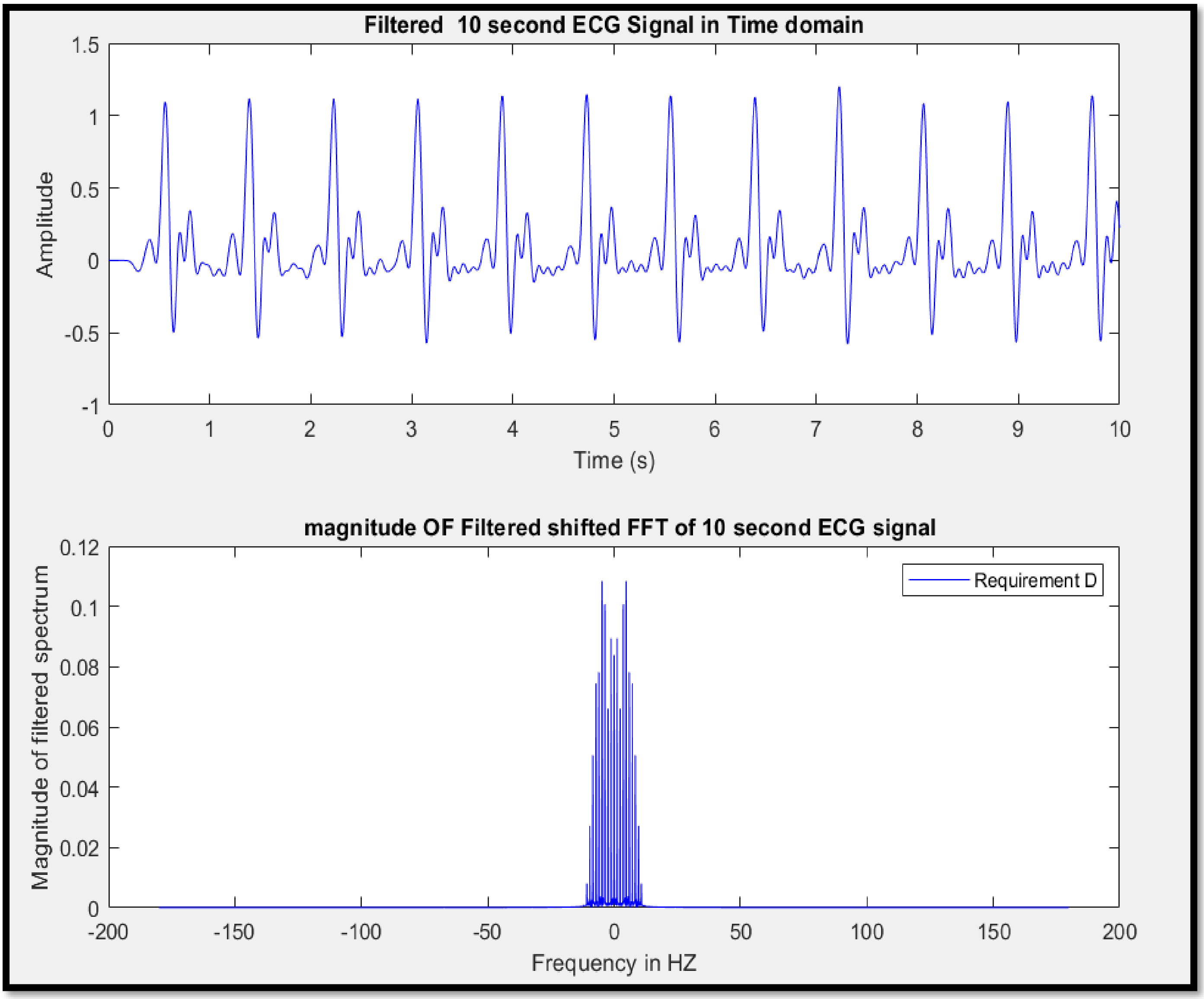


Explanation

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:

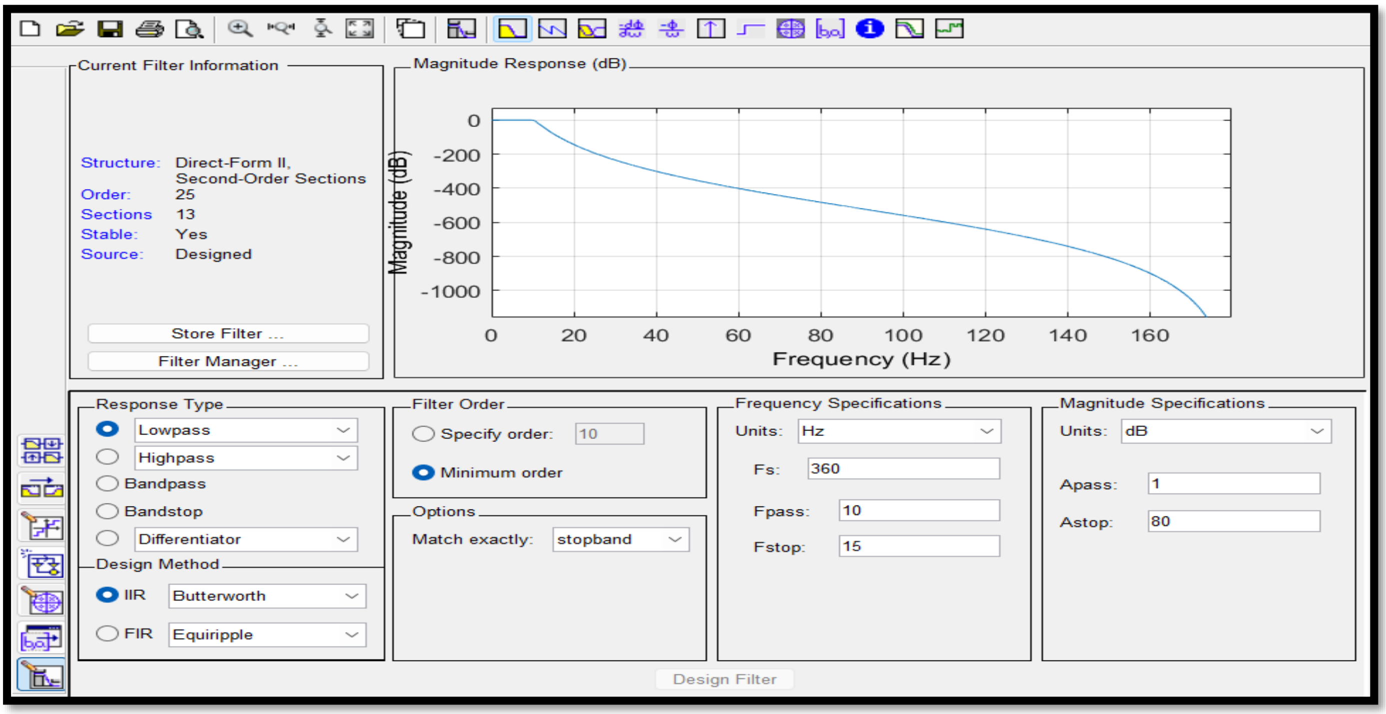
OUTPUT



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:



Justification

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

```
Requirement E

Avg_BPM_fd =

    72
```

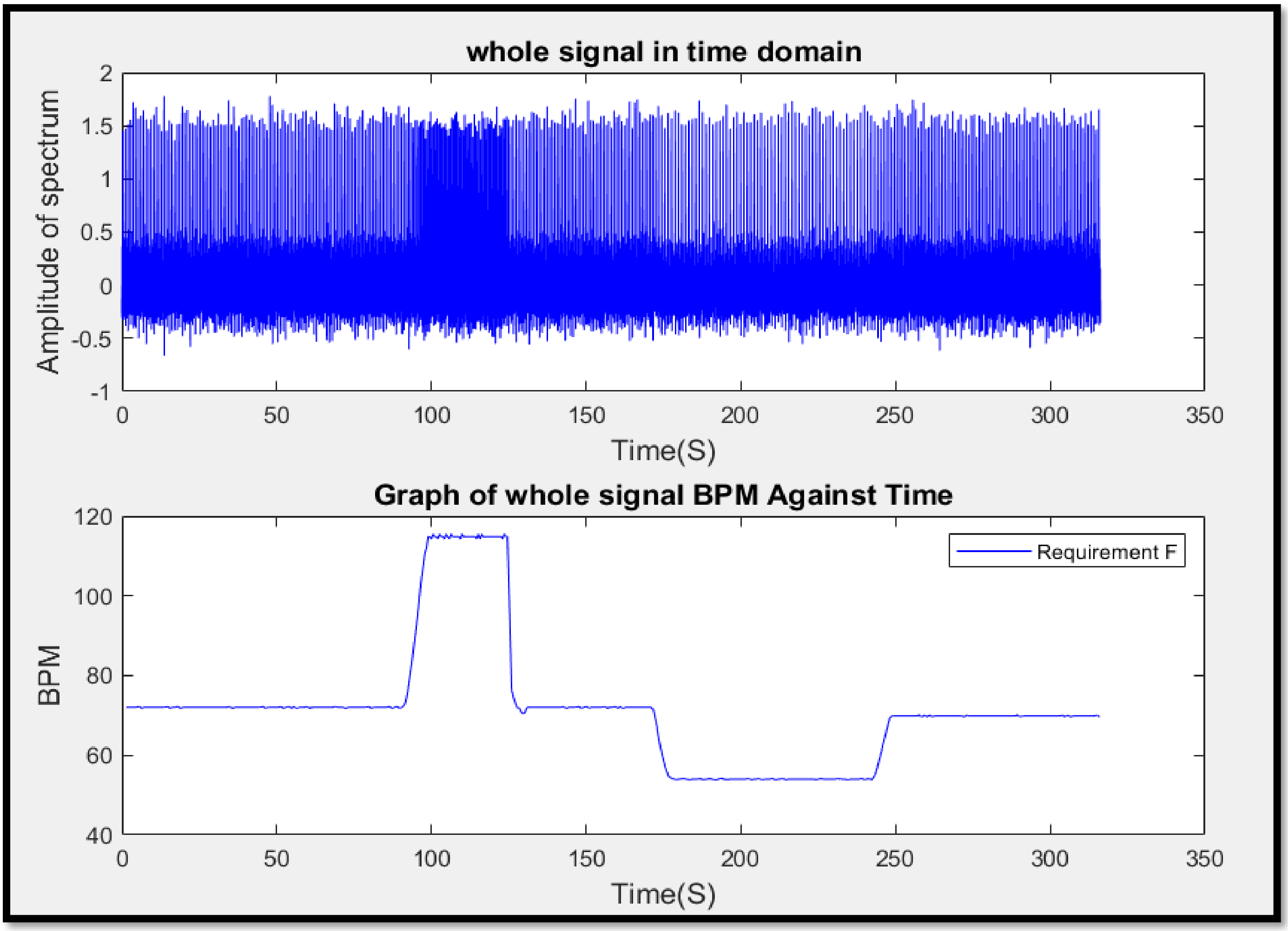
- **Comparison**
  Avg_BPM_fd=Avg_Bpm_td
  The average BPM in time domain same as frequency domain for the 10 second ecg signal

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



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

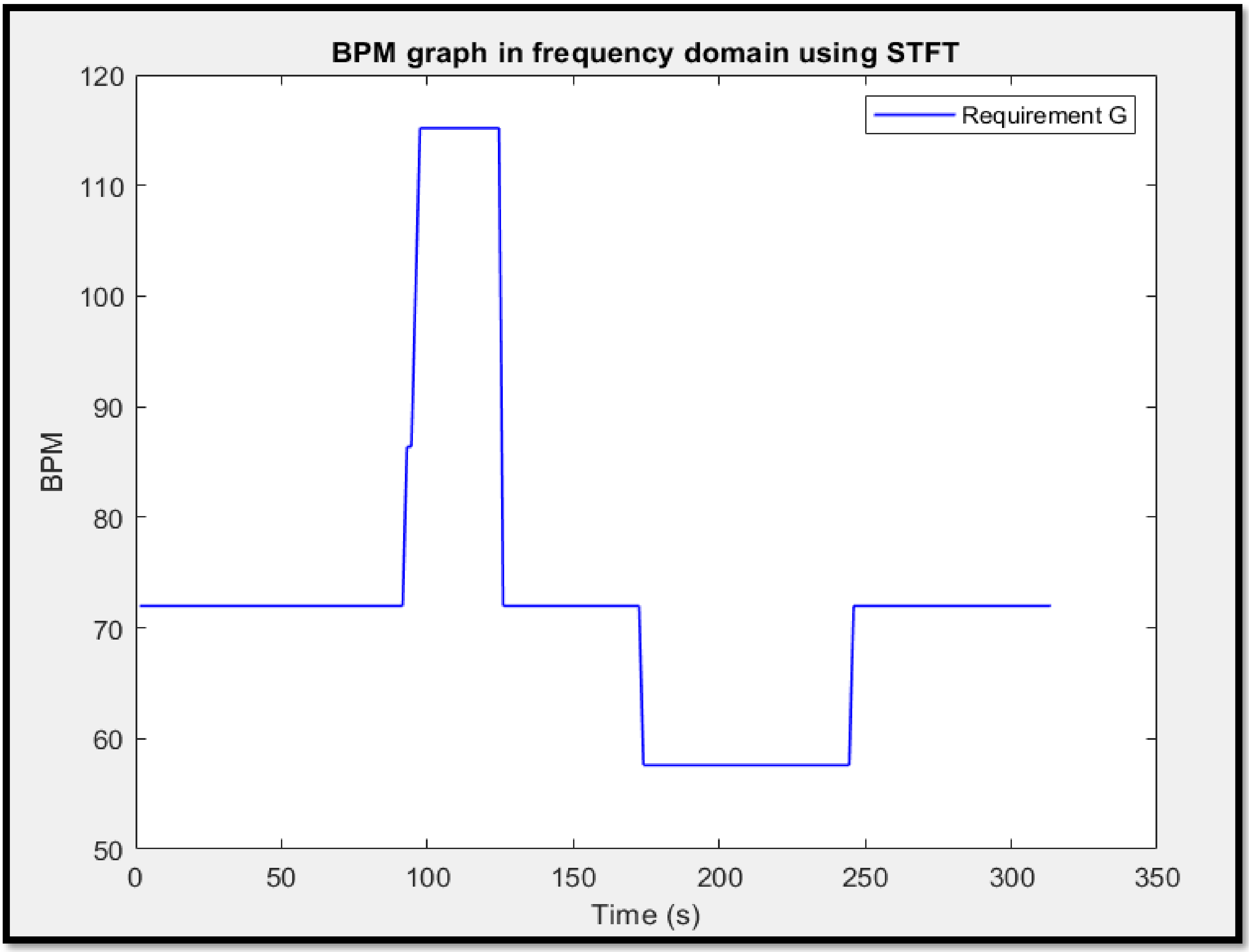
I will calculate the time interval between successive peaks using function diff and change into seconds by dividing it by sampling frequency. and get the Bpm values by divide 60 by intervals vector .we need to get time vector same size as BPM values to plot with each other

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

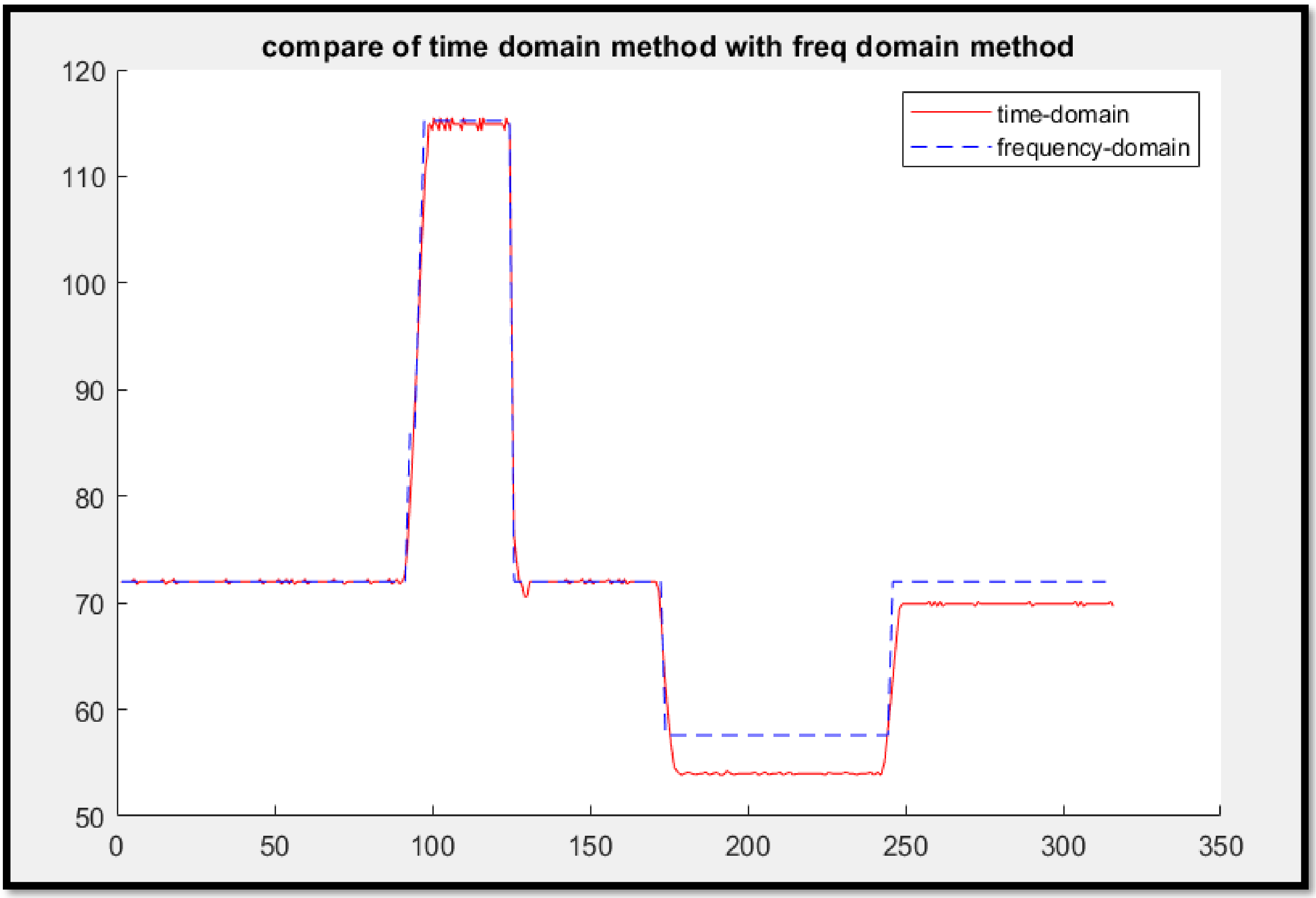


Explanation

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



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

Justifying

Time Domain Analysis:

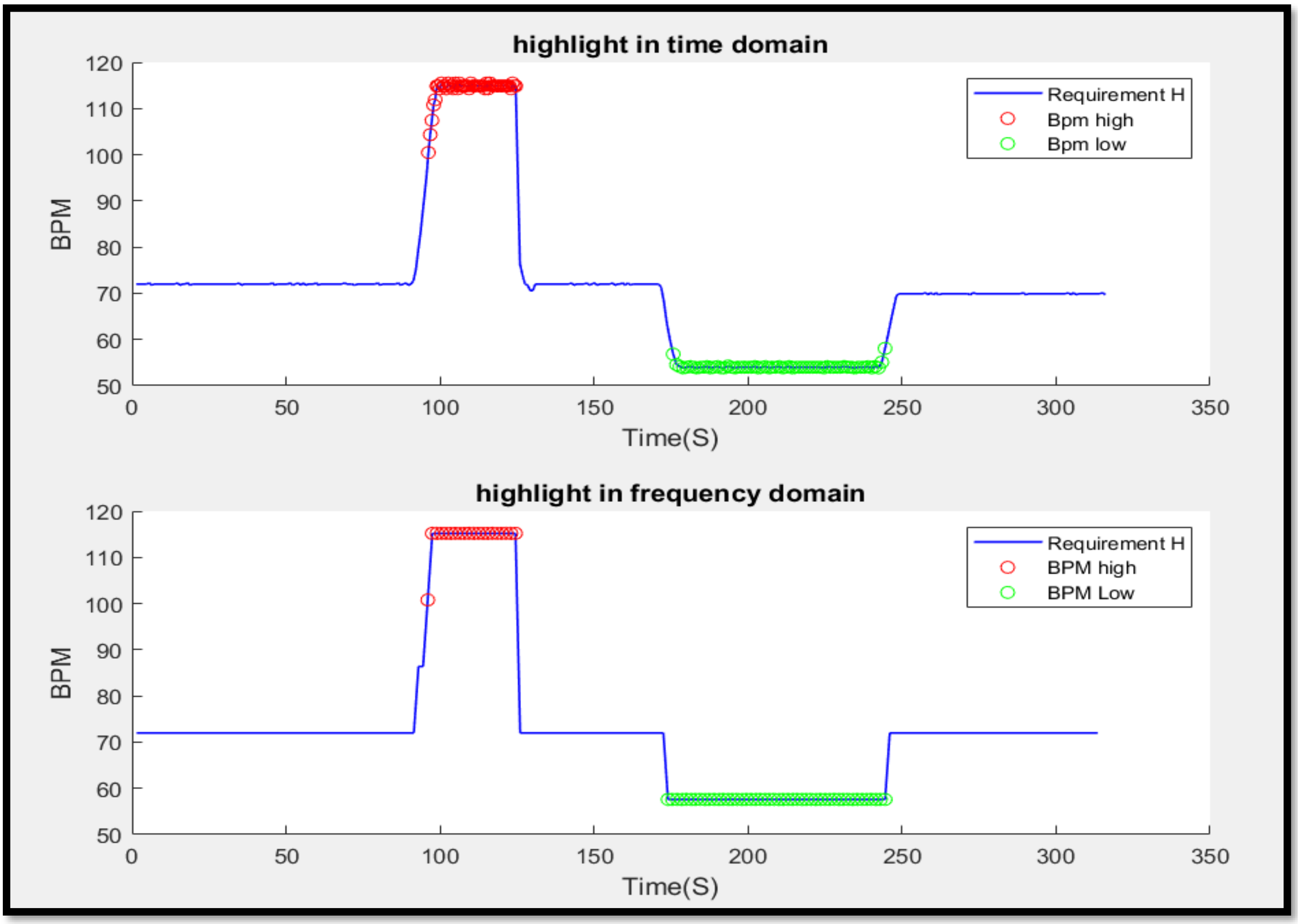
This approach provides precise, beat-to-beat measurements that reflect instantaneous variations in heart rate.

Frequency Domain Analysis:

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

Requirement h :

OUTPUT




```
time domain
 start High BPM interval: 96.2278   to end High BPM interval: 124.5500 seconds
 start Low BPM interval: 175.6833   to end Low BPM interval: 244.4611 seconds
 time domain High BPM interval: 28.3222 seconds
 time domain Low BPM interval: 68.7778 seconds
frequency domain
 start High BPM interval: 96.0000   to end High BPM interval: 124.5000 seconds
 start Low BPM interval: 174.0000   to end Low BPM interval: 244.5000 seconds
 freuency domain High BPM interval: 28.5000 seconds
 frequency domain Low BPM interval: 70.5000 seconds
```

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");
        ------------------BBBBBBB------------------
% 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,'MinPeakDist
ance', 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,"Mi
nPeakDistance",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));
```

References

1. https://www.mathworks.com/help/matlab/ref/double.max.html
2. https://www.mathworks.com/help/signal/ref/findpeaks.html
3. https://www.mathworks.com/help/signal/ref/stft.html
4. https://www.mathworks.com/help/matlab/ref/fft.html
5. https://www.mathworks.com/help/matlab/ref/fprintf.html
6. https://www.mathworks.com/help/signal/ug/windows.html
7. https://www.mathworks.com/matlabcentral/answers/183642-finding-the-dominant-frequency-of-a-time-series-data-using-fft-matlab