Project Report

Electrocardiogram (ECG) Signal Analysis Tool with Stochastic Modelling

Mohammad Fares Aljamous 1088672

Omar Mohammad Zeineh 1088546

Hadi Albanna 1088677

Supervised By: Eng. Gasm El Bary

جامعة أبوظبي
ABU DHABI UNIVERSITY

Submitted: June 8, 2024

Contents

List of Figures

List of Tables

Abstract

In this project, we were asked to create a program that would filter an ECG signal This program was created on the MATLAB platform, which can be considered a programming language with an extensive library of built-in functions. The ECG program can have many applications, and its features can be extended way beyond what we have made here, as this is just a basic program that has basic features and functionality that can be used by anyone easily. The program has very little computational cost as it uses fast algorithms to do the tasks it is supposed to do. The program can clean the ECG signal and create a synthetic heart rate

1 Introduction

Nowadays, healthcare is one of the most important fields that an engineer must work on and develop. In our project, we were given an ECG signal(Electrocardiogram) which are heart electric signal. These signals were not pure, they contained a lot of noise and baseline drift which had to be filtered, using low-pass and high-pass filters. This project may save lives. In addition, we were able to create a synthetic heart rate to stimulate the heart rate. [1]

1.1 Motivation

In modern days, heart problems are one of the common diseases a person may face.ECG signals are useful because their results can help diagnose an unusually fast heart rate or an unusually slow heart rate. Still, a signal that contains a lot of noise and baseline drift may lead doctors to make wrong assumptions as a result of a wrong diagnosis. Our project could help cardiology doctors save lives.

1.2 Problem Statement

Our project aims to filter noise in an ECG signal, using a high pass filter to allow frequencies that are more than 0.5 Hz to pass, and a low pass filter to allow frequencies less than 50 Hz to pass.

1.3 Literature Review

In reviewing current solutions for ECG signal analysis, we found several advanced systems detailed in reputable literature. These systems generally include strong pre-processing methods for removing noise and artifacts, feature extraction, and various visualization tools. Some tools are particularly good at baseline drift removal and precise R-wave peak detection, leading to accurate heart rate variability analysis. However, despite their effectiveness, these systems often have downsides like high computational demands and complicated user interfaces, which can be difficult for non-experts to navigate. Moreover, many lack comprehensive stochastic modeling features essential for simulating heart rate variations. Our project aims to bridge these gaps by developing a MATLAB-based GUI that is easier to use and

combines signal processing, analysis, and advanced stochastic models, offering a more complete tool for healthcare professionals and researchers.

2 Design

Matlab was used in this project to design and create the ECG Signal Analysis Tool. Many of Matlab's built-in functions were used to create this project, these functions include functions that can take the Fourier transform of signals, functions that find peaks (minimas and maximas), functions that can do statistical calculations such as finding the mean and the standard deviation, and many more. The graphical user interface was created using Matlab's app builder.

2.1 Requirements

- Requirements:
  1. ECG Signal Preprocessing:
     - Implement algorithms for ECG signal preprocessing, including baseline drift removal, noise filtering, and artifact removal.
     - Allow users to adjust preprocessing parameters such as filter cutoff frequencies, window sizes, and artifact detection thresholds.
  2. Feature Extraction:
     - Develop algorithms for extracting relevant features from ECG signals, such as R-wave peak detection, RR interval calculation, and heart rate variability (HRV) analysis.
     - Compute time-domain and frequency-domain HRV parameters, including mean heart rate, standard deviation of RR intervals, and spectral power in different frequency bands.
  3. Stochastic Heart Rate Modelling:
     - Incorporate stochastic models to simulate heart rate variability and dynamic changes in heart rhythm over time.

- Use stochastic processes such as autoregressive (AR) models, Gaussian processes, or fractal-based models to generate synthetic heart rate time series with realistic variability.

4. Visualization:
   - Provide interactive visualizations of ECG signals and extracted features, including time domain plots, RR interval histograms, and frequency-domain spectrograms.
5. Analysis Tools:
   - Provide options for calculating correlation coefficients, mean squared errors, and other similarity metrics.

2.2 Design Process

2.2.1 Taking The Values From Excel To Matlab

1. Step one, we first started by finding the sampling frequency at which the signal was taken at from excel file, this was done by finding how many samples were taken each second, then we found the number of points in the ECG signal, this was also done by going to the excel file and finding the number of points in the signals.

```
Fs = 128
n = 7680
```

2. Step two, then we took the values of seconds and the actual signal values from the excel file by using the function "readmatrix()", and we stored the values of seconds in T1_s and we stored the values of signal one in T1_v1 and signal two in T1_v2.

```
T1 = readmatrix('T1.xls');
T1_s = T1(1:7680, 1);
T1_v1 = T1(1:7680, 2);
T1_v2 = T1(1:7680, 3);
```

2.2.2 Signal Filtering and Preprocessing

1. Step one, we found the discrete time Fourier transform of the ECG signal by using the function "fft()" or fast Fourier transform, and found

all of its frequencies.

```
F1_v1 = fft(T1_v1);
Fd_1 = Fs/n*(-n/2:n/2-1);
```

2. Step two, we ask the user to enter the cutoff frequencies that we are going to use for the low pass and high pass filters.

```
Fcb = input('Please enter the cutoff frequency for the
    highpass filter (0.5Hz-0.6Hz): ');
Fcp = input('Please enter the cutoff frequency for the
    lowpass filter (50Hz-60Hz): ');
```

3. Step three, we check whether the cutoff frequencies entered fit the range we allowed the user to enter.

```
if (Fcb >= 0.5 && Fcb <= 0.6) && (Fcp >= 50 && Fcp <= 60)
------
------
else
disp("Please Enter The Specified Frequencies")
end
```

4. Step three, we removed the baseline drift and noise by applying a high pass filter and a low pass filter, the high pass filter takes the cutoff frequencies of 0.5Hz to 0.6Hz, that is to remove the baseline drift (which is caused by frequencies 0.5Hz to 0.6Hz and less), and the low pass filter takes the cutoff frequencies of 50Hz to 60Hz, that is to remove the noise (which is caused by frequencies 50Hz to 60Hz and higher). We apply these filters by using a loop which sets the Fourier transform of the signal for the unwanted frequencies to zero.

```
% Remove Baseline Drift and Powerline noise
FF1_v1 = F1_v1;
for i = [1:n]
    if ((Fd_1(i) <= Fcb && Fd_1(i) >= -Fcb) | (Fd_1(i) <= -Fcp
        | Fd_1(i) >= Fcp))
        FF1_v1(i) = 0;
    end
```

```
end
```

2.2.3 Feature Extraction

1. Step one: we took the inverse Fourier transform of the filtered ECG signal from the frequency domain to find it in the time domain, then we found the values of peaks and their locations using the "findpeaks()" function, we passed this function our signal, the sampling frequency, the minimum distance between peaks, and the minimum height of a peak. When we pass the sampling frequency to this function, then the minimum distance between peaks becomes in seconds rather than in indexes, and the locations also become in seconds rather than indexes. [2]

```
FT1_v1 = ifft(FF1_v1);
[pks,locs] = findpeaks(FT1_v1, Fs,'MinPeakDistance',0.29,
    'MinPeakHeight', 0.1);
```

2. Step two, we found all RR intervals, which are the differences between every two consecutive peaks, we found them using a for loop that computes the distance between every two consecutive peaks and stores it in an array; then we found the time for each interval, by using the function "cumsum()" which gets the cumulative sum of some array, and then we shifted the cumulative sum by adding the time at which the first peak was found.

```
% Find all R-R intervals and store them in an array
RR = [];
for i = 2:length(locs)
    RR(i-1) = locs(i)-locs(i-1);
end

% Find time for each RR interval
RR_s = cumsum(RR) + locs(1);
```

3. Step three: We found the stats of the RR interval, all of the statistics were found using built-in Matlab functions, the stats found were the

mean, standard deviation, minimum value, maximum value, and the difference between the minimum and maximum values.

```
%claculations for RR interval
Avg_RR = mean(RR)
sd_RR = std(RR)
min_RR = min(RR)
max_RR = max(RR)
diff_min_max = max_RR-min_RR
```

4. Step four, we found the heart rate array and its stats, we found that by dividing 60 by each element of the RR interval, we found the average heart rate and the standard deviation of the heart rate, and then we found the root mean square of the RR interval.

```
%calculate heart rate
Heart_Rate = 60./RR; %heart rate for each RR interval
Avg_heart_rate = mean(Heart_Rate)
sd_heart_rate = std(Heart_Rate)

%Time Domain HRV analysis
SDRR = sd_RR
RMSRR = sqrt(mean(RR.^2))
```

5. Step five: we found the Fourier transform of the RR intervals and found the power spectrum from that, We found it by using:

$$P = \frac{RR(f)^2}{n^2}$$

and then we found the sampling frequency of the RR intervals, and from that, we found the frequencies that makeup RR.

```
RR_f = fft(RR);
RR_ps = (abs(RR_f).^2)/(length(RR_f).^2);
FsRR = length(RR)/RR_s(length(RR));
Fd_RR = FsRR/length(RR)*(-length(RR)/2:length(RR)/2-1);
```

2.2.4 Stochastic Modeling

1. Step one, we first started by determining which type of stochastic model we wanted to use, and we chose a second-order autoregressive model, autoregressive models work by generating points based on previous points of the generated time series, and adding white Gaussian noise to each new element of the generated time series means that the next points are affected by this noise, a second order AR model means that we multiply the point two points back from the point we want to generate by a first coefficient, and we multiply the point one point back from the point we want to generate by a second coefficient, then we add the results of the multiplications (this operation can be seen as a dot product between the coefficients vector and the two previous points vector), the noise at that point, and the mean we set for the time series to get the point we want to generate. We found the length of the original heart rate array which we would like to generate a synthetic heart rate time series that is similar to it, and we chose the coefficients 0.4 and -0.4. [3]

```
n = length(Heart_Rate);
c = [0.4, -0.4];
```

2. Step two, we created the white Gaussian noise by creating an array of random numbers between zero and one, and we multiplied that by the standard deviation, we want for our time series, in this case, we want the standard deviation of our synthetic heart rate time series to be the same as the standard deviation of the heart rate we got from our ECG signal (in the GUI we will allow the user to input the mean and standard deviation they want), then we initialized the synthetic heart rate time series by using the function "zeros()" which will create an array full of zeros of length n (the length of the heart rate array we got from the ECG signal), then we set the first two values of the synthetic heart rate time series to be the same as the heart rate signal we found from the ECG signal, as the first two points cannot be generated by the AR model.

```
noise = sd_heart_rate*rand(n, 1);
synth_heart_rate = zeros(n, 1);
synth_heart_rate(1:2) = Heart_Rate(1:2);
```

3. Step three: we implemented the AR model by using a for loop which generates each point of the synthetic heart rate by adding the average we set, the dot product of the coefficients vector and the two previous points vector, and the noise we created.

```
for i = 3:n
    synth_heart_rate(i) = Avg_heart_rate +
        c*synth_heart_rate(i-2:i-1) + noise(i);
end
```

2.2.5 Visualizing and Analysis

1. Step one, we first plotted the time domain ECG signal, and we plotted the filtered time domain ECG signal, and we also plotted the peaks of the signal.

```
figure
subplot(2, 1, 1)
plot(T1_s, T1_v1)
title('Time Domain ECG Signal')

subplot(2, 1, 2)
FT1_v1 = ifft(FF1_v1);
[pks,locs] = findpeaks(FT1_v1, Fs,'MinPeakDistance',0.29,
    'MinPeakHeight', 0.1);
plot(T1_s, FT1_v1)
hold on
stem(locs, pks)
title('Filtered Time Domain ECG Signal (removed noise and
    baseline drift)')
hold off
```

2. Step two, we plotted frequency domain ECG signal, and we plotted the filtered frequency domain ECG signal.

```
figure
subplot(2, 1, 1)
```

```
plot(Fd_1, abs(F1_v1))
title('Frequency Domain ECG of T1 first reading')

subplot(2, 1, 2)
plot(Fd_1, abs(FF1_v1))
title('Frequency Domain ECG of T1 first reading (removed
    noise and baseline drift)')
```

3. Step three, we plotted the RR intervals in the time domain, in the frequency domain, and we plotted it's power spectrum.

```
figure
subplot(3, 1, 1)
plot(RR_s, RR)
title('RR intervals')
subplot(3, 1, 2)
plot(Fd_RR, abs(RR_f))
title('RR in the frequency domain')

subplot(3, 1, 3)
plot(Fd_RR, abs(RR_ps))
title('Power spectra of RR')
```

4. Step four: we plotted the heart rate time series from the ECG signal and the synthetic heart rate time series we generated.

```
figure
subplot(2,1,1)
plot(Heart_Rate)
title('Heart Rate Time Series From The ECG Signal')

subplot(2,1,2)
plot(synth_heart_rate)
title('Heart Rate Time Series Generated Using An
    Autoregressive Model')
```

5. Step five, we found the correlation coefficients and the root mean squared error between the heart rate time series from the ECG signal and the synthetic heart rate time series.

```
co = corrcoef(Heart_Rate, synth_heart_rate)
rmse = sqrt(sum(mean((Heart_Rate - synth_heart_rate).^2)))
```

2.2.6 Creating The GUI

1. Step one, we designed the layout we wanted for the GUI, and we created all the GUI elements needed for them Figure 1.

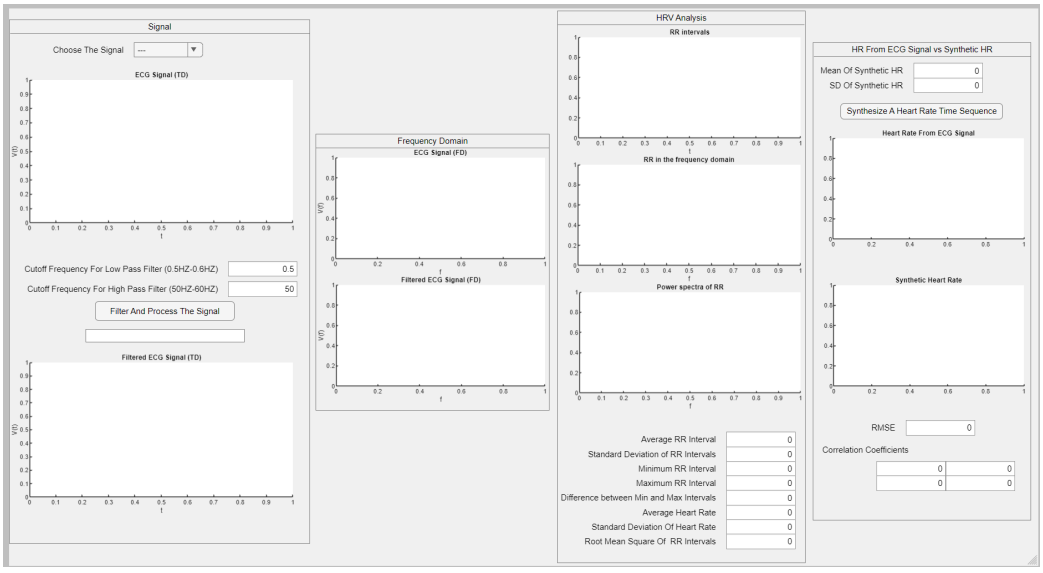

Figure 1: GUI Layout.

2. Step two, we bound each element of the GUI to our code, the code for the GUI dropdown selector can be seen in section 6, processes signal button in section 7, and synthesize heart rate in section 8.

2.3 System Overview

When all the code was written we had a simple system that takes an ECG signal, preprocesses it, extracts the features from it, generates a stochastic heart rate time series, shows the user plots and graphs, and finds similarity metrics between the heart rate we get from the ECG signal and the one generated by the program.

1. The user starts by running the GUI.
2. The user chooses the signal they want to processes Figure 2, Figure 3.
3. The user enters the cutoff frequencies for the low pass and high pass filters Figure 4, Figure 5.
4. The user presses Filter And Processes The Signal.
5. The GUI shows the filtered signal along with it's peaks, the data extracted from it, and the Heart Rate Variability analysis Figure 6.
6. Then the user can synthesize a heart rate time series to their liking by entering the mean and the standard deviation they want for their synthetic heart rate time series.
7. The GUI shows the synthesized heart rate time series and calculates the similarity metrics between the synthesized heart rate time series and the time series we get from the chosen ECG signal Figure 7.

2.4 Component Design

The system was completely made using built in MATLAB functions.

3 Experimental Testing and Results

3.1 Testing Plan and Acceptance Criteria

Testing the program is quite simple, since the program just does calculations and plotting based on the selected ECG signal, and the cutoff frequencies entered by the user. For the program to pass the testing it just needs to take the ECG signal and produce the desired outputs based on the input signal.

3.2 Results

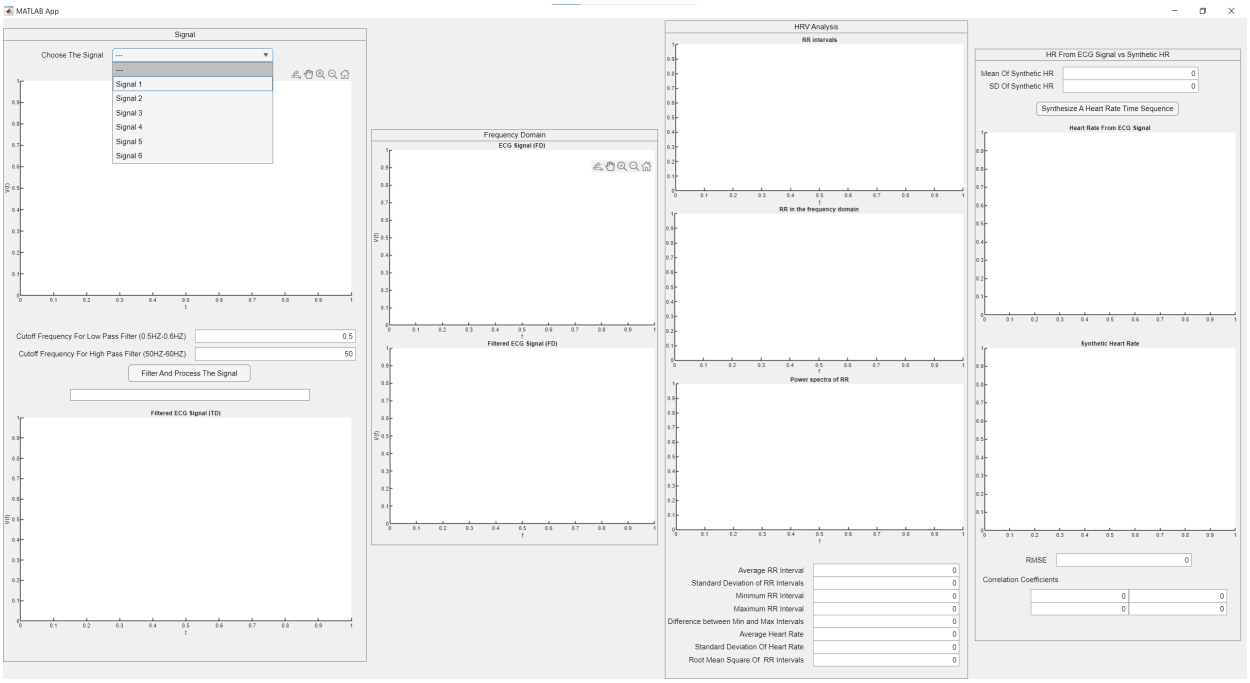

Figure 2: Dropdown Menu For Selecting ECG Signal.

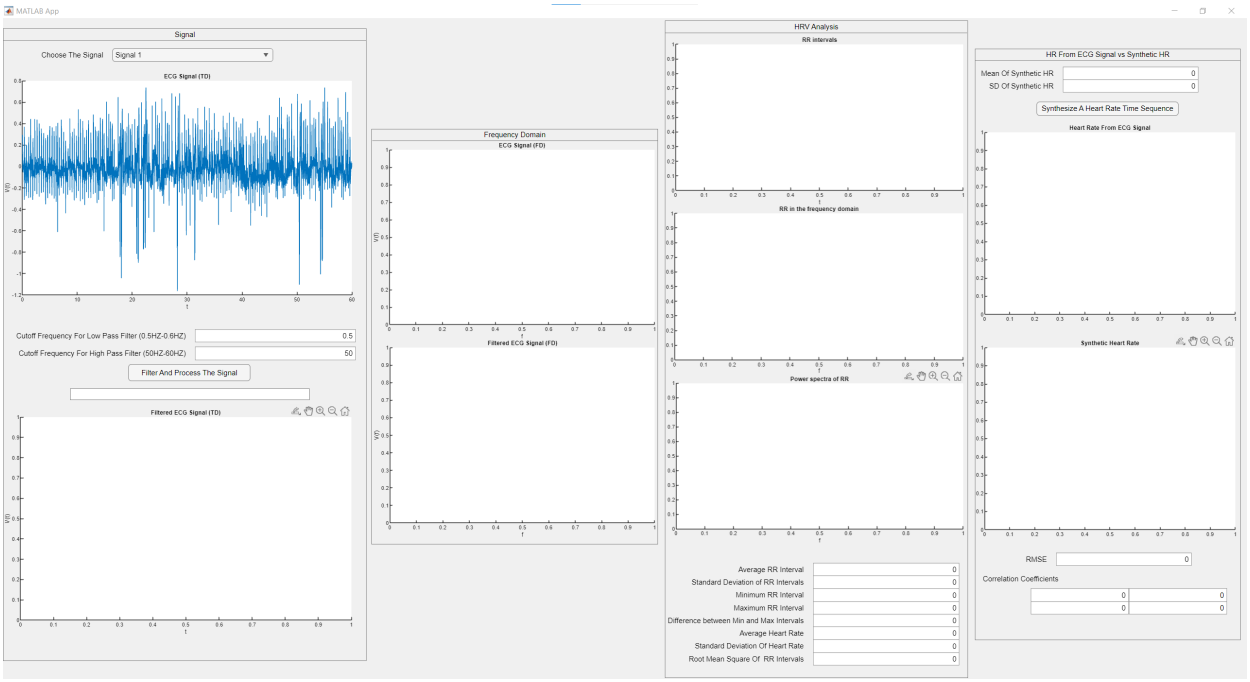

Figure 3: Selecting The ECG Signal.

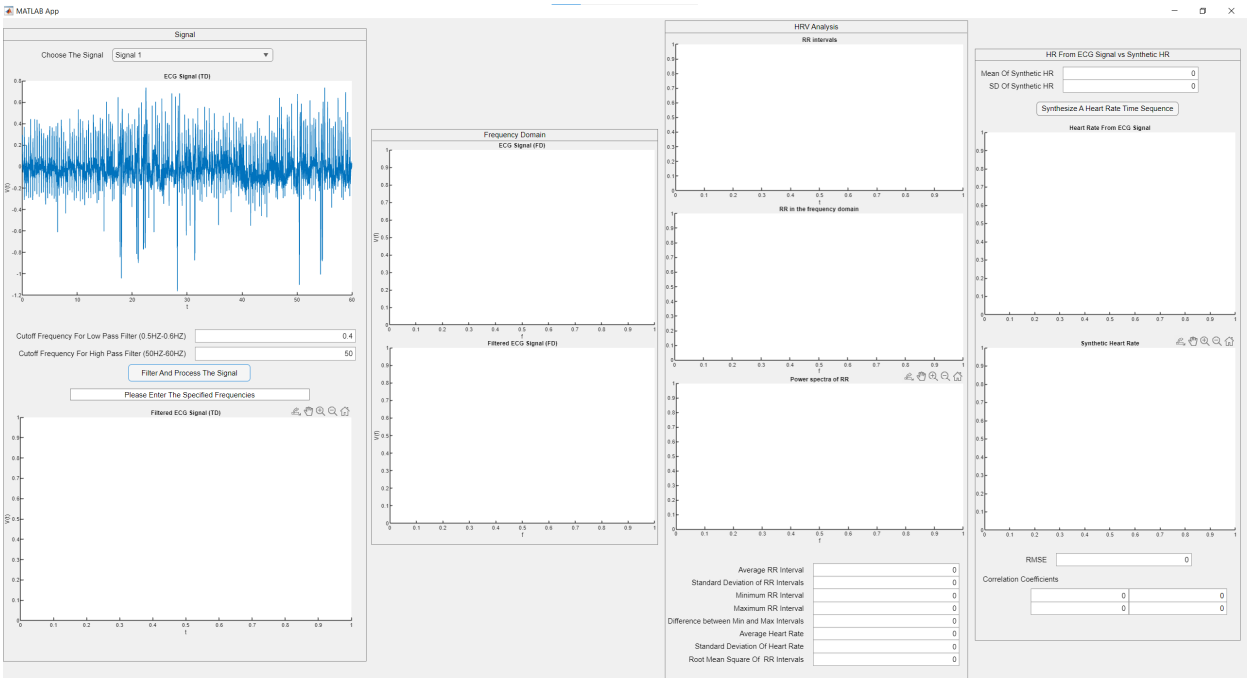

Figure 4: Trying To processes With an Invalid Cutoff Frequency Of High Pass Filter.

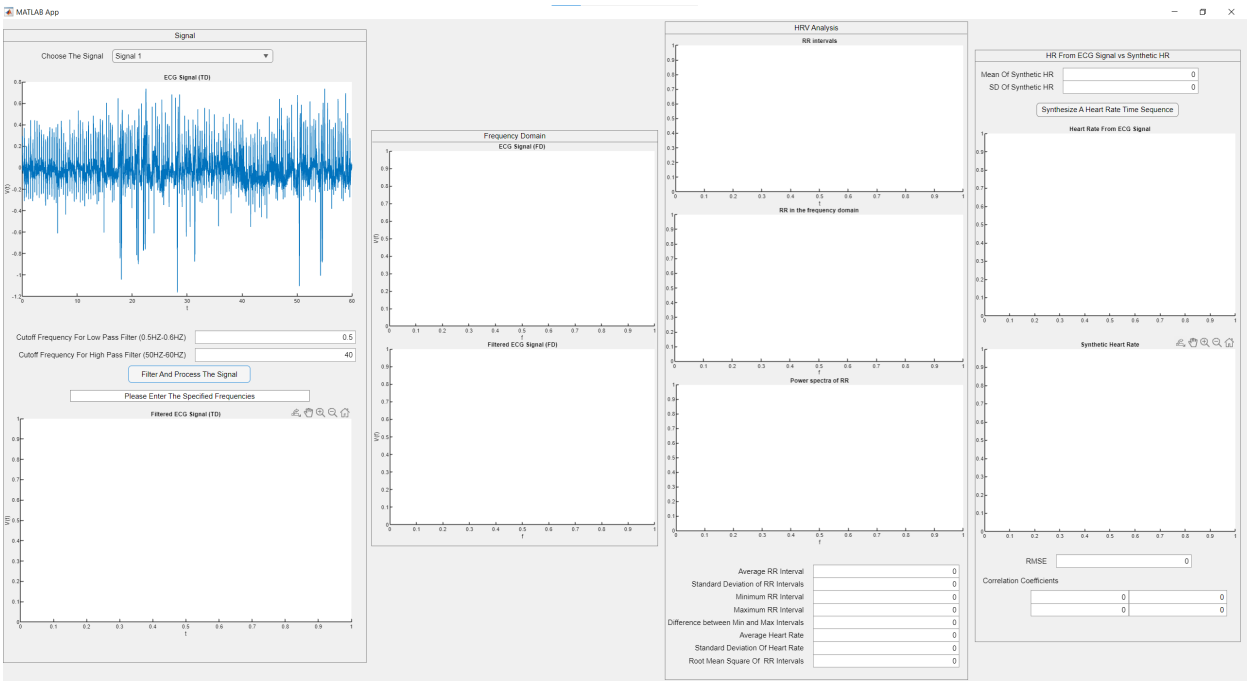

Figure 5: Trying To processes With an Invalid Cutoff Frequency Of Low Pass Filter.

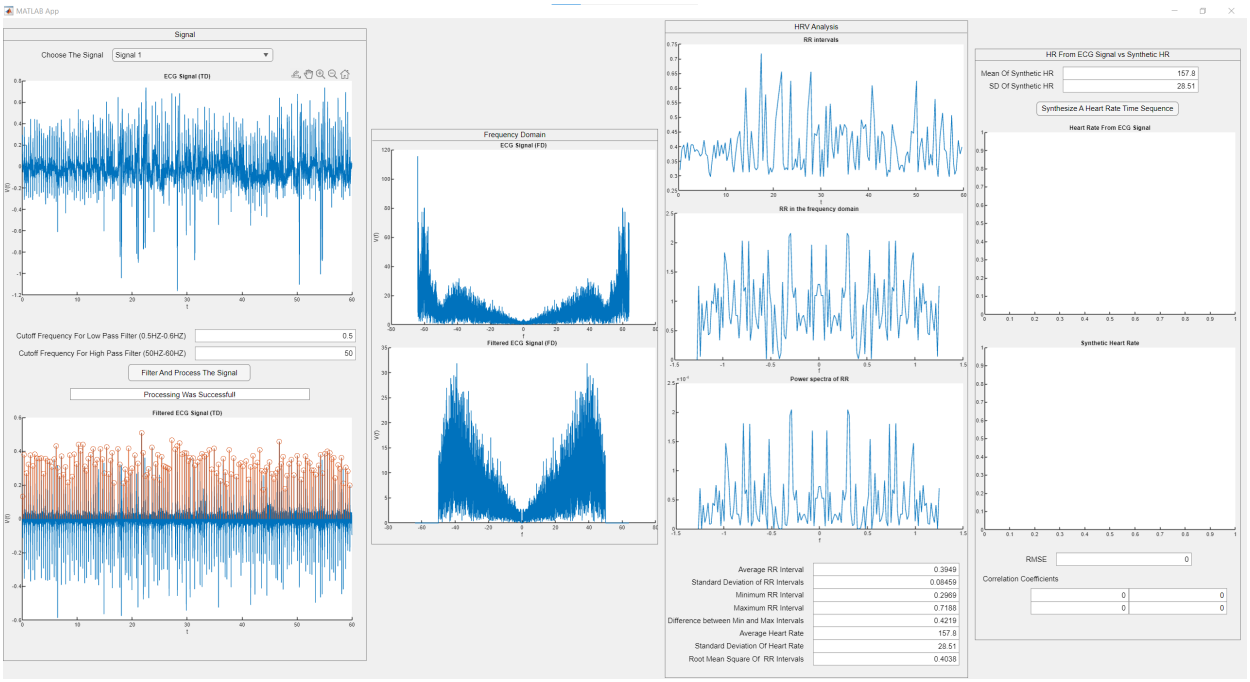

Figure 6: Successful Processing.

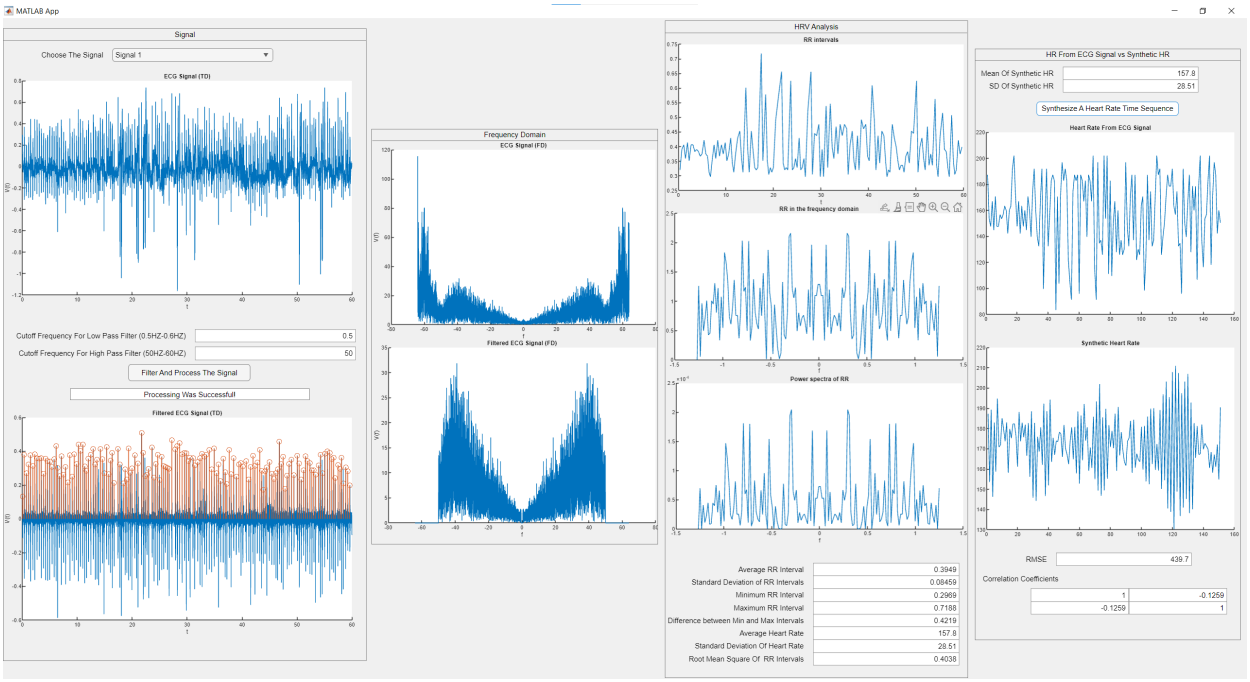

Figure 7: Synthesising Heart Rate.

3.3 Analysis and Interpretation of Data

We can see from the testing that the program works as intended, where it is able to take an ECG signal from the excel file, then preprocess it, filter it, extract the features from it, generate a stochastic heart rate time series, visualize it, and analyze it. And this means that the goal of this program has been achieved.

4 Conclusion

4.1 Summary

This report covers the creation of an ECG Signal Analysis Tool with Stochastic Modeling using MATLAB. The project was done in three parts: cleaning and analyzing ECG signals, developing models to simulate heart rate changes, and building a user-friendly interface. Our results show that we successfully removed noise from ECG signals and identified key features like R-wave peaks and heart rate variability. The models we created accurately simulate heart rate changes and provide useful visualizations. These achievements meet the project goals and have important implications for the medical field, offering a valuable tool for analyzing ECG data. This project highlights how technology can improve the understanding and diagnosis of heart conditions.

4.2 Future Improvements and Takeaways

We gained a deep understanding of ECG signal processing, feature extraction, and stochastic models to simulate heart rate changes. We learned how to effectively remove noise, detect important ECG features, and create useful visualizations. This project provided a valuable tool for analyzing and understanding heart functions, which is beneficial for medical professionals and researchers. Overall, the experience was very rewarding, showing how technology can be applied to improve healthcare and solve real-world problems.

4.3 Lessons Learned

We acquired and applied new knowledge in ECG signal processing and stochastic modeling. We used MATLAB, enhancing our skills in this powerful com-

putational tool, and delved into tutorials to better understand noise removal and feature extraction techniques. Our research on stochastic processes, including auto-regressive models and Gaussian processes, allowed us to simulate realistic heart rate variability. We learned a great deal through experimentation, particularly in fine-tuning our algorithms for optimal performance. Additionally, our failed attempts taught us valuable lessons about the complexities of ECG signal analysis and the importance of precise parameter adjustments.

4.4 Team Dynamics

- Who was the team leader? What evidence of good leadership can you provide? Omar Mohammad was the team leader for Group 5. Evidence of his good leadership includes successfully coordinating the project phases, ensuring all deadlines were met, and facilitating effective communication among team members. Omar's leadership was instrumental in keeping the team organized and focused on their goals.

- How did you create a collaborative and inclusive environment? Were all members engaged? How did the team communicate/collaborate? Did you create a WhatsApp group? Met in the library? Met in the lab? Communicated by email? Were all members in attendance? Did you take the meeting minutes? We created a collaborative and inclusive environment by ensuring that all team members Omar, Fares, and Hadi were actively engaged. We set up a WhatsApp group for quick communication and frequent updates, met regularly in the library and the lab for in-depth discussions and hands-on work, and used email for formal communication and sharing documents. All members attended the meetings, and we kept detailed meeting minutes to track our progress and responsibilities.

- What goals did you set for your team? Our goals were to develop a comprehensive ECG Signal Analysis Tool with MATLAB, implement effective pre-processing and feature extraction algorithms, integrate stochastic modeling for heart rate variability, and create an intuitive GUI. Additionally, we aimed to meet all project deadlines and ensure that our final submission was thorough and well-documented.

- How did you plan the tasks? Did you use a Gantt Chart? Did you track

your progress and update your tasks? We planned our tasks using a Gantt Chart to clearly outline each phase of the project and the associated deadlines. This helped us allocate tasks efficiently among Omar, Fares, and Hadi. We regularly tracked our progress and updated the Gantt Chart to reflect completed tasks and any necessary adjustments, ensuring we stayed on schedule.

- Did you meet your objectives? Yes, we met our objectives. We successfully developed the ECG Signal Analysis Tool with all required features, including signal preprocessing, feature extraction, stochastic modeling, and visualization. Our final GUI was user-friendly and met the project's requirements. We submitted our reports and code on time and presented our project as scheduled.

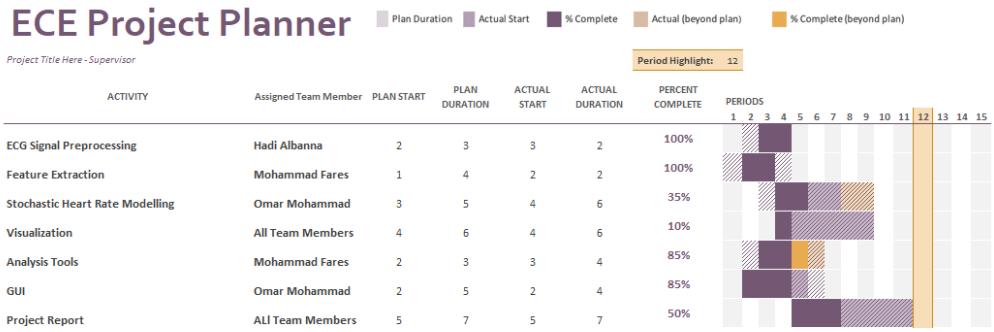

Figure 8: Gantt Chart.

4.5 Impact Statement

Environmental Impact Analysis								
Impact of your project	**Nature**	**Extent**	**Timing**	**Severity**	**Duration**	**Reversibility**	**Uncertainty**	**Significance**
The climate Example: *Does the project affect the emission of greenhouse gases into the atmosphere?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,it doesn't affect the climate because our project aim to solve ECG signal noise							
Use of Energy Example: *Does your project affect the energy consumption of the economy? How?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,it doesn't affect the Energy because our project aim to solve ECG signal noise							
Air quality Example: *Does the project have an effect on emissions of harmful air pollutants that might affect human health, damage crops or buildings or lead to deterioration in the environment (soil or rivers)?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,it doesn't affect the air quality because our project aim to solve ECG signal noise							
Biodiversity, flora, fauna and landscapes Example: *Does it affect endangered species, their habitats or ecologically-sensitive areas?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,it doesn't affect the biodiversity because our project aim to solve ECG signal noise							

Figure 9: Impact Statement.

Water quality and resources Example: *Does the project decrease or increase the quality or quantity of freshwater and groundwater?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,it doesn't affect the water quality because our project aim to solve ECG signal noise							
Renewable or non-renewable resources Example: *Does the project reduce or increase use of non-renewable resources?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: no ,it doesn't affect it because our project aim to solve ECG signal noise							
Sustainability Example: Does the option lead to more sustainable production and consumption? How?	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: no,it doesn't affect because our project aim to solve ECG signal noise							
Waste production/generation/recycling Example: *Does the project affect waste production (solid, urban, agricultural, industrial, mining, radioactive or toxic waste) or how waste is treated, disposed of or recycled?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,because our project aim to solve ECG signal noise							

Figure 10: Impact Statement.

Economic Impact Analysis								
Impact of your project	**Nature**	**Extent**	**Timing**	**Severity**	**Duration**	**Reversibility**	**Uncertainty**	**Significance**
Economic Prosperity Example: *Does the project affect the GDP/capita, employment rate, household savings?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: Yes ,because our project aim to solve ECG signal noise ,so it may increase the country economy if more nations started buying this project							
Investment Flows Example: *Does your project affect the flow of investment from outside the country? Does it encourage local investment in it?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: yes, it may help hospitals become better especially the cardiology department							
Public Budgets or Services Example: *Does the project affect the budgets of hospitals, community services, older people services, transport services, service quality, schools, policing, municipality services..etc?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: Yes,it may increase the budget of the hospitals because the have to implement this project.							
Market Mechanisms Example: *Does it affect the private sector business opportunities? Help companies reach more costumers? Change how business is done?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: Yes, it may help some engineers that are specalized in this department to find jobs							

Figure 11: Impact Statement.

Innovation, Research and Development Example: *Does the project have commercialization potential, lead to a potential patent? Does it allow others to innovate/research through it?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: Yes, people can develop this project and make it more accurate to get better diagnosis.							
Sustainable Consumption and Production Example: Does the project produce a sustainably consumed product or service? Can it be produced sustainably?	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,because our project aim to solve ECG signal noise							

Figure 12: Impact Statement.

Social Impact Analysis								
Impact of your project	**Nature**	**Extent**	**Timing**	**Severity**	**Duration**	**Reversibility**	**Uncertainty**	**Significance**
Health and Longevity Example: *Does the project impact health and longevity? Does it affect physical activity, nutrition, chronic diseases, accidental injuries, independent living, mental wellbeing?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: Yes,because our project aim to solve ECG signal noise, it can help doctors make a better diagnosis about the patient condition							
Safety Example: *Does your project affect safety of social environment, protection of older people against abuse, protection against risks, response to emergency cases, feelings of safety, physical safety?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: Yes,it may help save lives by being able to read the patient condition							
Productive and Valued Activities Example: *Does the project increase leisure time, reduce stress, lead to positive behavior, increase productivity?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,because our project aim to solve ECG signal noise							

Figure 13: Impact Statement.

Standard of Living Example: *Does it affect the quality of life? Make lives easier? Reduce poverty and deprivation? Increase life choices and opportunities?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,because our project aim to solve ECG signal noise							
Education/Life-long Learning Example: *Does the project affect literacy, use of ICT, chances of higher education, quality of education, life-long learning? Improve attainment of learning outcomes?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,because our project aim to solve ECG signal noise							
Quality of Social Interaction Example: *Does the project affect social connectedness, social participation, volunteering?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,because our project aim to solve ECG signal noise							

Figure 14: Impact Statement.

Privacy and Personal Data Example: *Does the project reveal the user identities? Create potential private data leaks or identity theft?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,because our project aim to solve ECG signal noise							
Social Reasonability Example: *Does the project affect access to products and services for people of determination? Does it affect their integration into society? Does it affect their participation in the economy? Does it address their needs?*	Direct Positive	Local	Immediate	High	Temporary	Reversible	Low Likelihood	Unimportant
	Justification/Explanation: No,because our project aim to solve ECG signal noise							

Figure 15: Impact Statement.

5 Code For Program

```
clear; clc;
format compact
close all

Fs = 128
n = 7680
Fcb = input('Please enter the cutoff frequency for the highpass
    filter (0.5Hz-0.6Hz): ');
Fcp = input('Please enter the cutoff frequency for the lowpass
    filter (50Hz-60Hz): ');

T1 = readmatrix('T1.xls');
T1_s = T1(1:7680, 1);
T1_v1 = T1(1:7680, 2);
T1_v2 = T1(1:7680, 3);

if (Fcb >= 0.5 && Fcb <= 0.6) && (Fcp >= 50 && Fcp <= 60)

F1_v1 = fft(T1_v1);
Fd_1 = Fs/n*(-n/2:n/2-1);

% Remove Baseline Drift and noise
FF1_v1 = F1_v1;
for i = [1:n]
```

```
    if ((Fd_1(i) <= Fcb && Fd_1(i) >= -Fcb) | (Fd_1(i) <= -Fcp |
        Fd_1(i) >= Fcp))
        FF1_v1(i) = 0;
    end
end

figure
subplot(2, 1, 1)
plot(T1_s, T1_v1)
title('Time Domain ECG of T1 first reading')

subplot(2, 1, 2)
FT1_v1 = ifft(FF1_v1);
[pks,locs] = findpeaks(FT1_v1, Fs,'MinPeakDistance',0.29,
    'MinPeakHeight', 0.1);
plot(T1_s, FT1_v1)
hold on
stem(locs, pks)
title('Time Domain ECG of T1 first reading (removed noise and
    baseline drift)')
hold off


figure
subplot(2, 1, 1)
plot(Fd_1, abs(F1_v1))
title('Frequency Domain ECG of T1 first reading')

subplot(2, 1, 2)
plot(Fd_1, abs(FF1_v1))
title('Frequency Domain ECG of T1 first reading (removed noise and
    baseline drift)')


% Find all R-R intervals and store them in an array
RR = [];
for i = 2:length(locs)
    RR(i-1) = locs(i)-locs(i-1);
end

```

```
% Find time for each RR interval
RR_s = cumsum(RR) + locs(1);

%claculations for RR interval
Avg_RR = mean(RR)
sd_RR = std(RR)
min_RR = min(RR)
max_RR = max(RR)
diff_min_max = max_RR-min_RR

%calculate heart rate
Heart_Rate = 60./RR; %heart rate for each RR interval
Avg_heart_rate = mean(Heart_Rate)
sd_heart_rate = std(Heart_Rate)

%Time Domain HRV analysis
SDRR = sd_RR
RMSRR = sqrt(mean(RR.^2))

%Frequency Domain HRV analysis
RR_f = fft(RR);
RR_f(1) = 0;
RR_ps = (abs(RR_f).^2)/(length(RR_f).^2);
FsRR = length(RR)/RR_s(length(RR));
Fd_RR = FsRR/length(RR)*(-length(RR)/2:length(RR)/2-1);

figure
subplot(3, 1, 1)
plot(RR_s, RR)
title('RR intervals')
subplot(3, 1, 2)
plot(Fd_RR, abs(RR_f))
title('RR in the frequency domain')

subplot(3, 1, 3)
plot(Fd_RR, abs(RR_ps))
title('Power spectra of RR')


n = length(Heart_Rate);
```

```
c = [0.4, -0.4];
noise = sd_heart_rate*rand(n, 1);
synth_heart_rate = zeros(n, 1);
synth_heart_rate(1:2) = Heart_Rate(1:2);
for i = 3:n
    synth_heart_rate(i) = Avg_heart_rate +
        c*synth_heart_rate(i-2:i-1) + noise(i);
end

figure
subplot(2,1,1)
plot(Heart_Rate)
title('Heart Rate Time Series From The ECG Signal')

subplot(2,1,2)
plot(synth_heart_rate)
title('Heart Rate Time Series Generated Using An Autoregressive
    Model')

co = corrcoef(Heart_Rate, synth_heart_rate)
rmse = sqrt(sum(mean((Heart_Rate - synth_heart_rate).^2)))

else

disp("Please Enter The Specified Frequencies")
end
```

6 Code For GUI Drop Down Menu

```
value = app.ChooseTheSignalDropDown.Value;
           switch value
               case "Signal 1"
                   T1 = readmatrix('T1.xls')
                   T1_s = T1(1:7680, 1)
                   T1_v1 = T1(1:7680, 2)
                   plot(app.UIAxes, T1_s, T1_v1)
               case "Signal 2"
                   T1 = readmatrix('T1.xls');
```

```
                    T1_s = T1(1:7680, 1)
                    T1_v1 = T1(1:7680, 3)
                    plot(app.UIAxes, T1_s, T1_v1)
                case "Signal 3"
                    T1 = readmatrix('T4.xls');
                    T1_s = T1(1:7680, 1)
                    T1_v1 = T1(1:7680, 2)
                    plot(app.UIAxes, T1_s, T1_v1)
                case "Signal 4"
                    T1 = readmatrix('T4.xls');
                    T1_s = T1(1:7680, 1)
                    T1_v1 = T1(1:7680, 3)
                    plot(app.UIAxes, T1_s, T1_v1)
                case "Signal 5"
                    T1 = readmatrix('T7.xls');
                    T1_s = T1(1:7680, 1)
                    T1_v1 = T1(1:7680, 2)
                    plot(app.UIAxes, T1_s, T1_v1)
                case "Signal 6"
                    T1 = readmatrix('T7.xls');
                    T1_s = T1(1:7680, 1)
                    T1_v1 = T1(1:7680, 3)
                    plot(app.UIAxes, T1_s, T1_v1)
                case "---"
                    plot(app.UIAxes, [1], [1])

            end
```

7 Code For GUI Processes Button

```
clear; clc;
format compact
close all

Fs = 128
n = 7680
Fcb = input('Please enter the cutoff frequency for the highpass
    filter (0.5Hz-0.6Hz): ');
```

```
Fcp = input('Please enter the cutoff frequency for the lowpass
    filter (50Hz-60Hz): ');

T1 = readmatrix('T1.xls');
T1_s = T1(1:7680, 1);
T1_v1 = T1(1:7680, 2);
T1_v2 = T1(1:7680, 3);

if (Fcb >= 0.5 && Fcb <= 0.6) && (Fcp >= 50 && Fcp <= 60)

F1_v1 = fft(T1_v1);
Fd_1 = Fs/n*(-n/2:n/2-1);

% Remove Baseline Drift and noise
FF1_v1 = F1_v1;
for i = [1:n]
    if ((Fd_1(i) <= Fcb && Fd_1(i) >= -Fcb) | (Fd_1(i) <= -Fcp |
        Fd_1(i) >= Fcp))
        FF1_v1(i) = 0;
    end
end

figure
subplot(2, 1, 1)
plot(T1_s, T1_v1)
title('Time Domain ECG of T1 first reading')

subplot(2, 1, 2)
FT1_v1 = ifft(FF1_v1);
[pks,locs] = findpeaks(FT1_v1, Fs,'MinPeakDistance',0.29,
    'MinPeakHeight', 0.1);
plot(T1_s, FT1_v1)
hold on
stem(locs, pks)
title('Time Domain ECG of T1 first reading (removed noise and
    baseline drift)')
hold off


figure
```

```
subplot(2, 1, 1)
plot(Fd_1, abs(F1_v1))
title('Frequency Domain ECG of T1 first reading')

subplot(2, 1, 2)
plot(Fd_1, abs(FF1_v1))
title('Frequency Domain ECG of T1 first reading (removed noise and
    baseline drift)')


% Find all R-R intervals and store them in an array
RR = [];
for i = 2:length(locs)
    RR(i-1) = locs(i)-locs(i-1);
end

% Find time for each RR interval
RR_s = cumsum(RR) + locs(1);

%claculations for RR interval
Avg_RR = mean(RR)
sd_RR = std(RR)
min_RR = min(RR)
max_RR = max(RR)
diff_min_max = max_RR-min_RR

%calculate heart rate
Heart_Rate = 60./RR; %heart rate for each RR interval
Avg_heart_rate = mean(Heart_Rate)
sd_heart_rate = std(Heart_Rate)

%Time Domain HRV analysis
SDRR = sd_RR
RMSRR = sqrt(mean(RR.^2))

%Frequency Domain HRV analysis
RR_f = fft(RR);
RR_f(1) = 0;
RR_ps = (abs(RR_f).^2)/(length(RR_f).^2);
FsRR = length(RR)/RR_s(length(RR));
```

```
Fd_RR = FsRR/length(RR)*(-length(RR)/2:length(RR)/2-1);

figure
subplot(3, 1, 1)
plot(RR_s, RR)
title('RR intervals')
subplot(3, 1, 2)
plot(Fd_RR, abs(RR_f))
title('RR in the frequency domain')

subplot(3, 1, 3)
plot(Fd_RR, abs(RR_ps))
title('Power spectra of RR')


n = length(Heart_Rate);
c = [0.4, -0.4];
noise = sd_heart_rate*rand(n, 1);
synth_heart_rate = zeros(n, 1);
synth_heart_rate(1:2) = Heart_Rate(1:2);
for i = 3:n
    synth_heart_rate(i) = Avg_heart_rate +
        c*synth_heart_rate(i-2:i-1) + noise(i);
end

figure
subplot(2,1,1)
plot(Heart_Rate)
title('Heart Rate Time Series From The ECG Signal')

subplot(2,1,2)
plot(synth_heart_rate)
title('Heart Rate Time Series Generated Using An Autoregressive
    Model')

co = corrcoef(Heart_Rate, synth_heart_rate)
rmse = sqrt(sum(mean((Heart_Rate - synth_heart_rate).^2)))

else

```

```
disp("Please Enter The Specified Frequencies")
end
```

8 Code For GUI Synthesize Heart Rate Button

```
global Heart_Rate
n = length(Heart_Rate);
c = [0.4, -0.4];
noise = app.SDOfSyntheticHREditField.Value*rand(n, 1);
synth_heart_rate = zeros(n, 1);
synth_heart_rate(1:2) = Heart_Rate(1:2);
for i = 3:n
    synth_heart_rate(i) = app.MeanOfSyntheticHREditField.Value +
        c*synth_heart_rate(i-2:i-1) + noise(i);
end

co = corrcoef(Heart_Rate, synth_heart_rate)
rmse = sqrt(sum(mean((Heart_Rate - synth_heart_rate).^2)))

app.RMSEEditField.Value = rmse
app.CorrelationCoefficientsEditField.Value = co(1,1)
app.CorrelationCoefficientsEditField_2.Value = co(1,2)
app.EditField_2.Value = co(2,1)
app.EditField_3.Value = co(2,2)

plot(app.UIAxes2_2,Heart_Rate)

plot(app.UIAxes3_2,synth_heart_rate)
```

References

[1] M. A. Rahman, M. M. H. Milu, A. Anjum, A. B. Siddik, M. M. H. Sifat, M. R. Chowdhury, F. Khanam, and M. Ahmad, “A statistical designing

approach to matlab based functions for the ecg signal preprocessing," *Iran Journal of Computer Science*, vol. 2, pp. 167–178, 2019.

[2] L. Rajani Kumari, Y. Padma Sai, and N. Balaji, "Ecg signal preprocessing based on empirical mode decomposition," in *Microelectronics, Electromagnetics and Telecommunications: Proceedings of ICMEET 2015*. Springer, 2016, pp. 673–679.

[3] T. Kuusela, "Stochastic heart-rate model can reveal pathologic cardiac dynamics," *Physical Review E*, vol. 69, no. 3, p. 031916, 2004.