ENG EK 122 Final Project Report

ECG Data Analysis in MATLAB

Preprocessing, R-Peak Detection, Heart Rate, and HRV Analysis

Name:	Parth Kheni
School:	Boston University
Course:	ENG EK 122
Section:	C2
Instructor:	Stormy Attaway
Date:	December 13, 2024

GitHub Repository: https://github.com/parthnkheni/ECG-Data-Analysis

Abstract

Electrocardiography (ECG) is widely used to monitor cardiac activity and can reveal clinically meaningful features such as the QRS complex and R-peaks. This project implements a MATLAB-based ECG data analysis workflow that takes raw ECG input, reduces noise via filtering, detects R-peaks using a Pan–Tompkins-style approach, and computes derived metrics including heart rate and heart rate variability (HRV). The tool produces visualizations of the raw and processed signals and highlights detected peaks to support interpretation. This report describes the problem motivation, design choices, algorithmic pipeline, MATLAB implementation, and opportunities for improvement.

Contents

1 Introduction

An ECG is a time-series measurement of the heart's electrical activity. In a typical ECG waveform, the QRS complex corresponds to ventricular depolarization, and the R-peak is the most prominent feature within the QRS complex. Robust detection of R-peaks enables computation of:

- **Heart rate (HR)** in beats per minute (bpm), from the time between consecutive R-peaks (RR intervals).
- **Heart rate variability (HRV)**, which summarizes variability in RR intervals and can be used as a noninvasive indicator related to autonomic regulation.

However, raw ECG signals can be contaminated by baseline wander, muscle noise, and powerline interference. This motivates a preprocessing stage prior to peak detection and metric extraction.

2 Project Objectives

The primary objectives of this project were:

1. Build a MATLAB workflow to **import ECG data** and visualize raw signals.
2. Apply **signal preprocessing** (filtering and normalization) to improve signal quality.
3. Implement **R-peak detection** suitable for typical ECG morphologies.
4. Compute **heart rate** from RR intervals and report summary statistics.
5. Compute **HRV metrics** and generate HRV-related plots.
6. Present results clearly through figures and annotated outputs.

3 System Overview

The repository contains two key MATLAB artifacts:

- `projectfinalv3.mlx`: Live Script used as the main driver for the analysis and for producing plots/outputs.
- `ECG_Data_Analysis.m`: MATLAB script/function file supporting the processing steps.

At a high level, the workflow is:

ECG Input → Preprocessing → R-peak Detection → RR Intervals → HR & HRV Metrics → Visualizations

4 Methods

4.1 Data Input and Assumptions

The tool is designed to accept ECG data from a file or data source and operate on a single-channel ECG vector sampled at a known sampling frequency f_s (Hz). A correct sampling frequency is important because it determines filter design and converts sample indices to time.

4.2 Preprocessing

To improve reliability of R-peak detection, the preprocessing stage targets common artifacts:

- **Baseline wander**: slow drift due to respiration/movement.
- **High-frequency noise**: muscle activity and sensor noise.
- **Amplitude scaling differences**: mitigated by normalization.

A typical approach is to use a **bandpass filter** to keep the ECG frequency content that best preserves the QRS complex while rejecting drift and high-frequency noise. After filtering, the signal can be normalized to stabilize subsequent thresholds.

4.3 R-Peak Detection (Pan–Tompkins-Style)

The project uses a Pan–Tompkins-style approach (or a similar QRS detection strategy), which is a classic pipeline for emphasizing QRS complexes. Conceptually, this approach:

1. Enhances the slope of QRS complexes (e.g., differentiation).
2. Emphasizes large changes (e.g., squaring).
3. Smooths into an energy-like envelope (e.g., moving-window integration).
4. Applies thresholding and peak selection rules to identify candidate R-peaks.

In practice, additional logic is often used to prevent double-counting within a short time window (refractory period) and to reject peaks that are too small relative to recent detections.

4.4 Heart Rate Computation

If the detected R-peak sample indices are $\{n_1, n_2, \ldots, n_K\}$, then the RR intervals (in seconds) are:

$$RR_i = \frac{n_{i+1} - n_i}{f_s}, \quad i = 1, \ldots, K-1.$$

A simple estimate of heart rate (bpm) is:

$$HR = \frac{60}{\overline{RR}},$$

where $\overline{RR}$ is the mean RR interval over the analyzed segment.

4.5 HRV Metrics

HRV can be summarized using time-domain metrics computed from RR intervals. Common examples include:

- **SDNN**: standard deviation of RR intervals.
- **RMSSD**: root mean square of successive RR differences.
- **pNN50**: percentage of successive RR differences greater than 50 ms.

This project computes HRV metrics and produces HRV plots to help interpret variability over time.

5 Implementation in MATLAB

5.1 Tools and Requirements

The project is implemented in MATLAB (R2021a or newer recommended) and relies on the Signal Processing Toolbox for filtering and signal operations.

5.2 Program Structure

The analysis is organized as a step-by-step pipeline in `projectfinalv3.mlx`:

1. Load ECG data and set sampling frequency.
2. Plot raw ECG for initial inspection.
3. Apply preprocessing (bandpass filtering and normalization).

4. Run R-peak detection to obtain peak indices/timestamps.
5. Compute RR intervals, heart rate, and HRV metrics.
6. Visualize processed ECG with detected R-peaks and generate HRV-related plots.

5.3 Pseudocode Summary

```
% Inputs: ecg (vector), fs (Hz)
% Outputs: r_locs (sample indices), HR (bpm), HRV metrics, plots

ecg_raw = ecg;

% 1) Preprocess
ecg_filt = bandpass_filter(ecg_raw, fs);
ecg_norm = normalize(ecg_filt);

% 2) R-peak detection (Pan-Tompkins style)
feature = moving_window_integrate(square(diff(ecg_norm)));
r_locs  = threshold_and_pick_peaks(feature, fs);

% 3) Metrics
RR = diff(r_locs) / fs;        % seconds
HR = 60 / mean(RR);            % bpm
HRV = compute_time_domain_hrv(RR);

% 4) Visualizations
plot_raw_and_filtered(ecg_raw, ecg_filt, fs);
plot_peaks(ecg_filt, r_locs, fs);
plot_rr_series(RR);
```

6 Results and Discussion

6.1 Expected Outputs

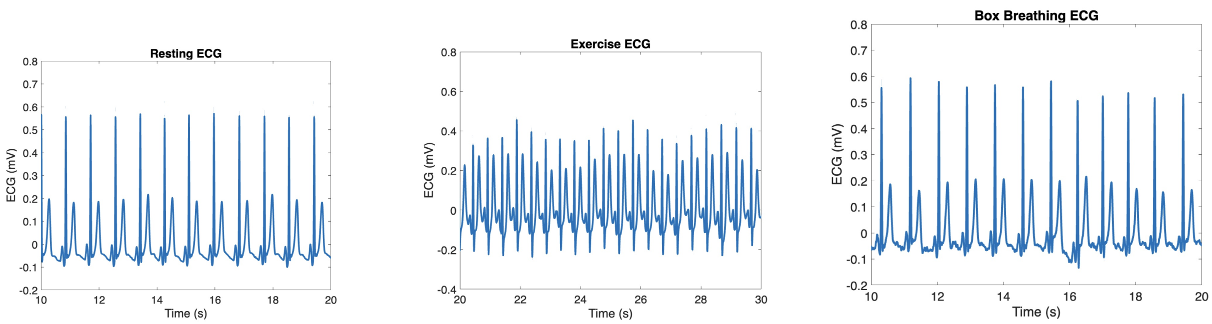
When executed with a valid ECG segment, the tool is expected to generate:

- Raw ECG plots for multiple conditions (resting, exercise, box breathing).

- R-peak detection plots with peaks overlaid on ECG.
- Heart rate distributions and comparisons between conditions.
- Additional diagnostic plots (beat-aligned overlays, slope/feature comparisons).

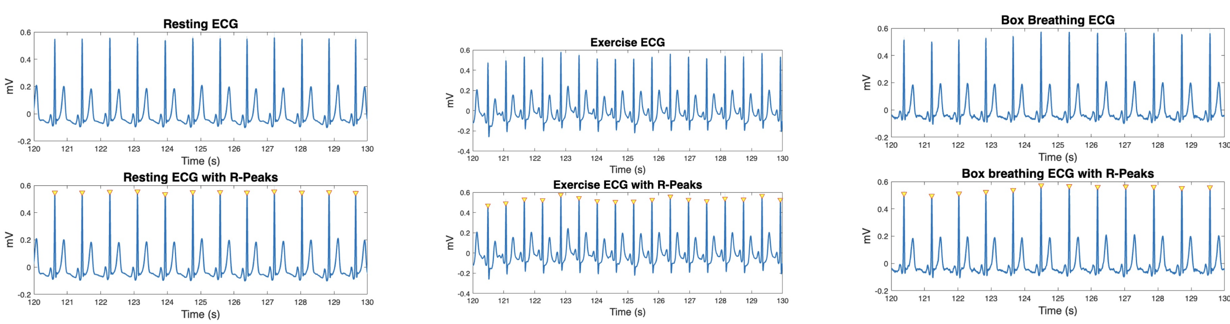
6.2 Figures

All figures are placed in an Overleaf folder named `figures/` using the filenames below.



(a) Resting (b) Exercise (c) Box breathing

Figure 1: Example raw ECG segments under three conditions.



(a) Resting (b) Exercise (c) Box breathing

Figure 2: Detected R-peaks overlaid on ECG signals for each condition.

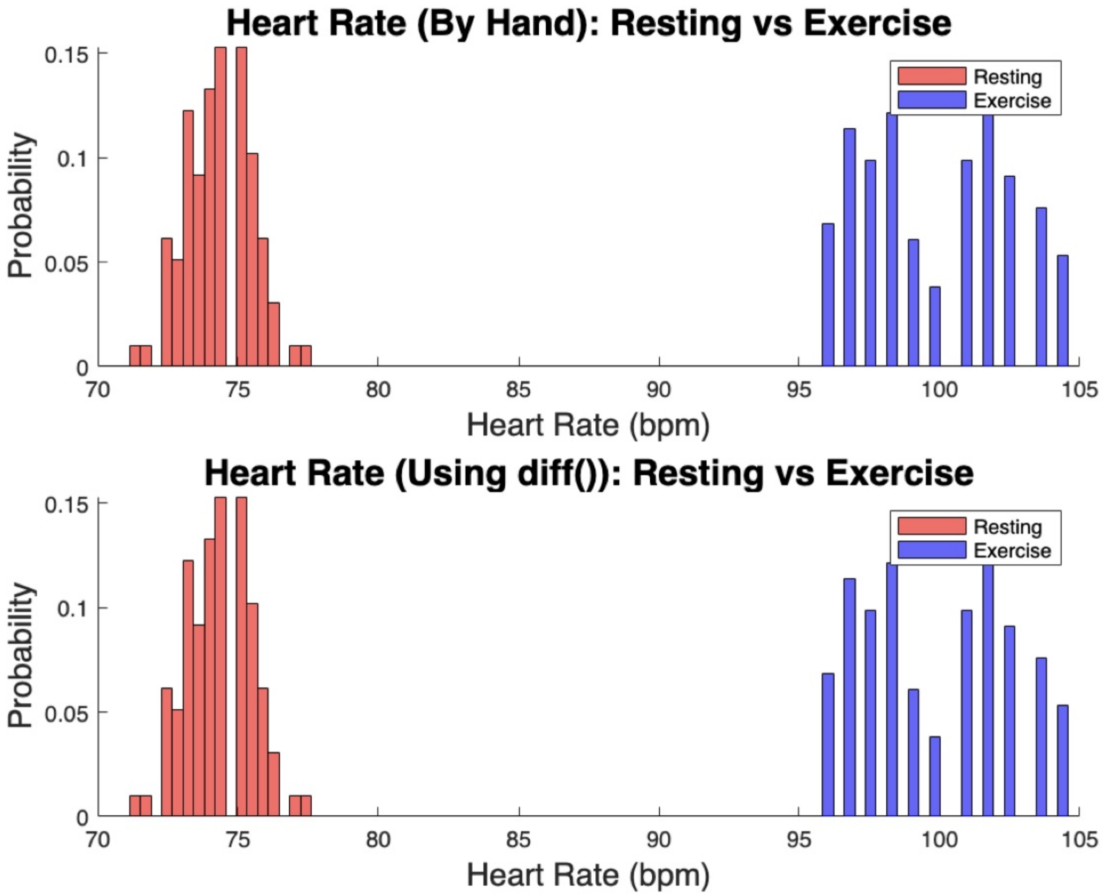


Figure 3: Heart rate distributions for resting vs. exercise (two computation methods).

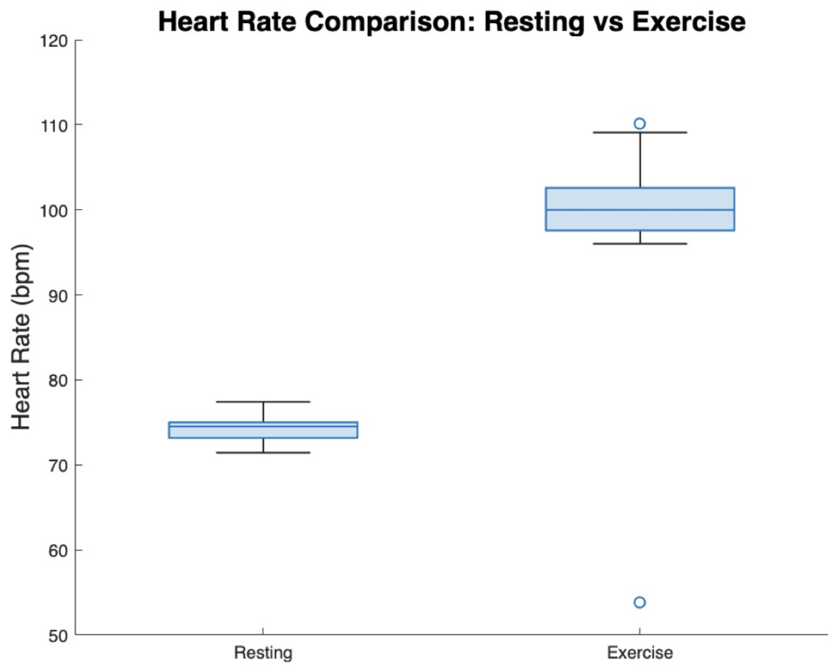


Figure 4: Heart rate comparison between resting and exercise using a boxplot.

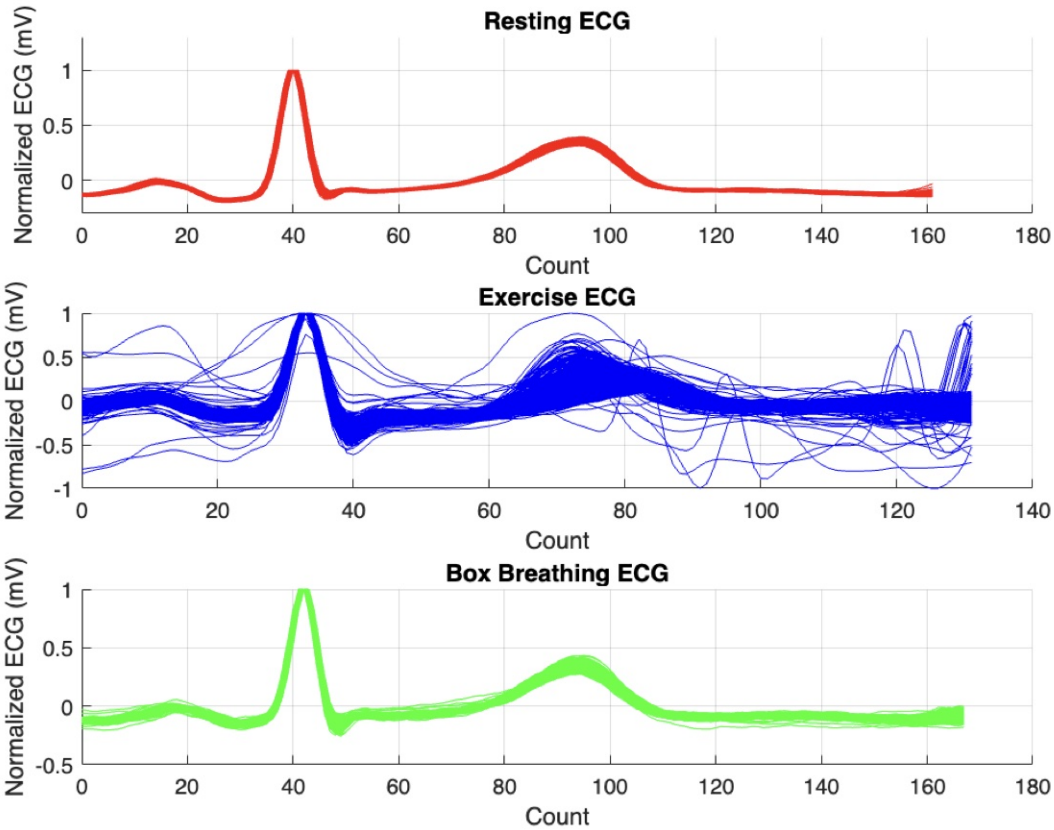


Figure 5: Beat-aligned waveform overlays used to compare typical beat morphology across conditions.

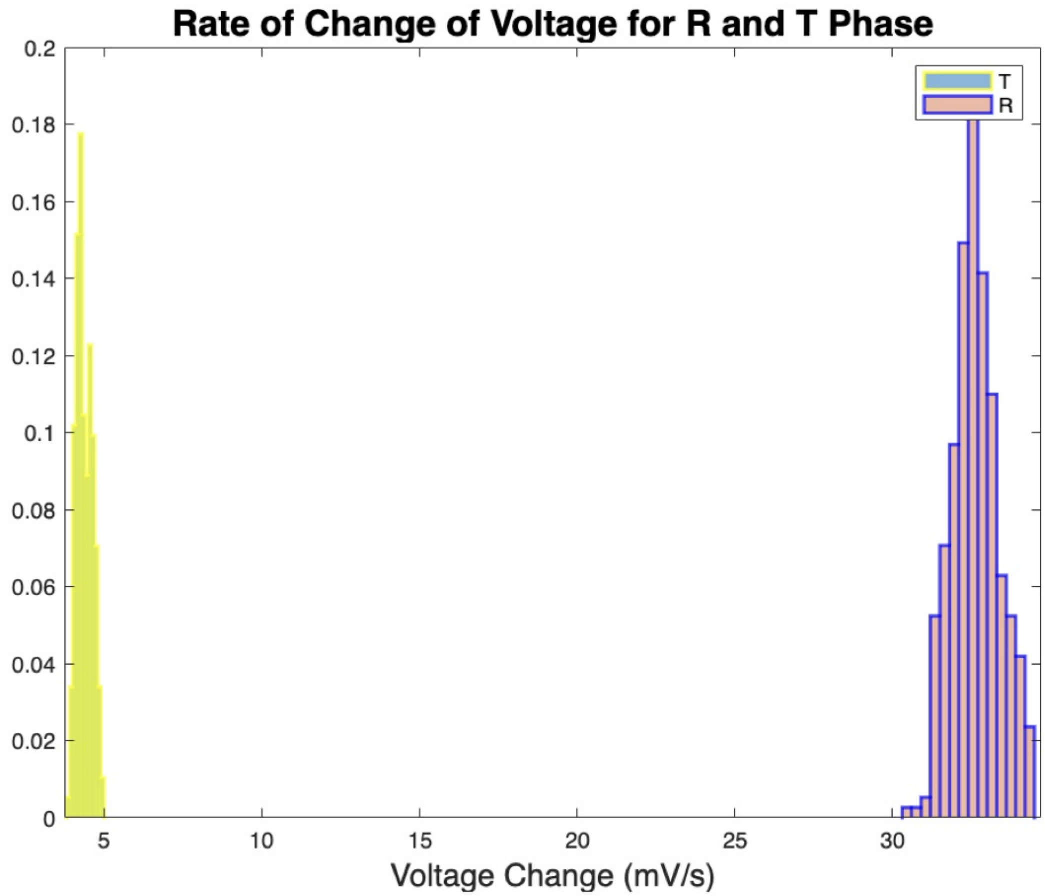


Figure 6: Slope/feature distribution comparison between the R-peak phase and T-wave phase.

6.3 Summary Table (Fill With Your Run Output)

Table 1: Example summary metrics reported by the tool (populate using MATLAB output).

Metric	Value
Sampling frequency f_s (Hz)	____________
Number of detected R-peaks	____________
Mean RR interval (s)	____________
Estimated heart rate (bpm)	____________
SDNN (s)	____________
RMSSD (s)	____________

6.4 Discussion

Across the three conditions, preprocessing improves the visibility of QRS complexes and reduces false detections caused by baseline drift and high-frequency noise. The R-peak detection plots verify that peaks occur at the expected locations in the waveform. Heart rate distributions and boxplots provide a clear comparison between resting and exercise. Additional diagnostic plots (beat-aligned overlays and slope/feature distributions) support the correctness of the detected features and help explain why R-peaks are separable from other waveform components such as the T-wave.

7 Limitations

Key limitations of the current approach include:

- **Sensitivity to noise and motion artifacts**: extremely noisy signals can still produce false peaks.
- **Parameter dependence**: filter cutoffs and detection thresholds may require tuning for different datasets.
- **Single-lead focus**: results may vary across leads; multi-lead fusion is not implemented.

8 Future Work

Potential improvements include:

1. Extend the pipeline to **classify arrhythmias** using machine learning.
2. Improve robustness for **low-quality recordings** with adaptive filtering and artifact detection.
3. Add **real-time analysis** support for streaming ECG from wearable devices.

9 Conclusion

This project delivers a MATLAB workflow for ECG data analysis, including preprocessing, R-peak detection, heart rate computation, HRV-related analysis, and visualization. The figures demonstrate signal behavior under multiple conditions and validate the detection and analysis pipeline.

References

[1] J. Pan and W. J. Tompkins, *A Real-Time QRS Detection Algorithm*, IEEE Transactions on Biomedical Engineering, vol. 32, no. 3, pp. 230–236, March 1985.

[2] Task Force of the European Society of Cardiology and the North American Society of Pacing and Electrophysiology, *Heart rate variability: standards of measurement, physiological interpretation and clinical use*, Circulation, vol. 93, no. 5, pp. 1043–1065, 1996.

[3] PhysioNet, *MIT-BIH Arrhythmia Database*, https://physionet.org/content/mitdb/1.0.0/.

A How to Run the Project

1. Clone the repository:

```
git clone https://github.com/parthnkheni/ECG-Data-Analysis.git
```

2. Open `projectfinalv3.mlx` in MATLAB.

3. Ensure the ECG data file path and sampling frequency are set correctly.

4. Run the Live Script to generate processed signals, detected peaks, heart rate metrics, and plots.

B Repository Link

https://github.com/parthnkheni/ECG-Data-Analysis