University of Sheffield

Models of heart rate and blood pressure variability

Demetra Camelia Neacsu

Supervisor: Richard Clayton

A report submitted in partial fulfilment of the requirements
for the degree of Bachelor's in Computer Science

in the

Department of Computer Science

January 9, 2021

Declaration

Name: Demetra Camelia Neacsu

Signature: Demetra Camelia Neacsu

Date: 03/06/2020

COVID-19 Impact Statement

The quick spread of COVID-19 has made countries all over the world go into lockdown, to save lives. This has meant that I've had to adapt to a new way of working, with university buildings being closed and without the possibility of having face-to-face meetings with my supervisor. Working from home was very difficult, as I did not have the necessary setup and I was sharing the office living-room with other family members, making it hard to concentrate. This proved to be a very challenging end of the Academic year as I felt that for the entire period of time, I had to deal with many hardships and the stress that they caused.

Acknowledgements

Firstly, I would like to thank my supervisor, Dr. Richard Clayton from the Department of Computer Science, for the guidance that I have received throughout the entire dissertation project. Secondly, I would like to thank my personal tutor, Dr. Siobhán North from the Department of Computer Science, for the help and advices that I have received throughout my University years. Finally, I would like to thank my friends and family for the enormous support and encouragement that I have received.

Abstract

Heart rate variability represents the physiological phenomenon of the variation in the time interval between successive heartbeats, measured in milliseconds. These days, heart rate variability analysis is widely used by wearable smart devices to help measure the user's overall health, sleep patterns or athletic performance. However, before smart devices, heart rate variability has long been known to be associated with morbidity and mortality after cardiovascular diseases. The dissertation project aims to implement a model of heart rate and blood pressure variability and then to use the model to investigate whether the recording of heart rate variability, from both healthy subjects and subjects with cardiovascular disease can be accurately reproduced. For this, we have implemented deBoer's model, in order to obtain values for each beat, for systolic and diastolic pressure, the arterial time constant and RR interval. These were then converted into the frequency domain, and thus, we obtained the value for baroreflex sensitivity. We simulated for three different groups, "healthy sedentary elderly", "endurance-trained elderly", "healthy sedentary young", as described in the "Baroreflex function in sedentary and endurance-trained elderly people" paper. The results have been very promising, showing that the model was able to reproduce results for the two elderly groups accurately and have small deviations in reproducing the young sedentary group.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In countries around the world, cardiovascular disease is one of the biggest causes of mortality, with 17.9 million deaths happening every year, globally, because of cardiovascular disease, more than from any other type of disease, according to *Cardiovascular Diseases - World-Health-Organization* (n.d.).

Most cardiovascular disease can be prevented through the addressing of several factors, such as unhealthy diet, physical inactivity or tobacco. Furthermore, all people with the cardiovascular disease, or who are at high cardiovascular risk, can benefit from early identification of the disease and drastically reduce the risk of death.

It is estimated that in the UK alone, right now, 7.4 million people are living with a heart or circulatory disease and on average, in a day there are 460 deaths attributed to a heart or circulatory disease. Furthermore, 280 of the hospital admission in a day are due to a heart attack, and 13 babies are diagnosed daily with a heart defect. *Heart Statistics* (2020)

Although the UK numbers have improved over the years, going down from 320 000 deaths a year in 1961 to 167 000 as of now, this number is still extremely high, and around 44 000 of the people dying are under the age of 75. To put these numbers in perspective, the population of the city of Oxford is 154 000, which shows the severity of cardiovascular diseases in the UK. Based on the articles Hillebrand et al. (2013) and Goldenberg et al. (2019), there is a direct correlation between heart rate variability and cardiovascular disease, as low heart rate variability values can be associated with a 32-45 % increased risk of a first cardiovascular event, in groups without a known **CVD** [1].

1.1 Aims and Objectives

The overall aim of this project is to implement a beat to beat model of the cardiovascular system, as presented in deBoer's paper and to evaluate the model's ability to replicate real heart rate variability, comparing against data extracted from three different subject groups, healthy sedentary elderly people, endurance-trained elderly people and healthy young people,

[1]Cardiovascular Disease

making necessary adjustments to increase the model's accuracy.

The overall aim can be broken down into several smaller objectives as follows :

- Gather and process data on the subjects to be used in the evaluation of the model
- Find real-time values and visualization of the heart rate variability interval and blood pressure in both normal subjects and the selected subject groups
- Implement deBoer's model, generating beat-to-beat values for systolic and diastolic blood pressure, arterial time constant and heart rate variability interval
- Plot initial time-domain results of the model
- Compare and test initial results against healthy normal human subjects and adjust model accordingly
- Perform frequency domain analysis on the time-domain values computed to generate the power spectrum of the signal
- Compute mean values for heart rate variability interval over a medium frequency interval(0.05-0.15 Hz) and high-frequency(0.15-0.35 Hz)
- Compute baroreflex sensitivity for each subject
- Evaluate results against real data, making adjustments as necessary to improve model's accuracy

1.2 Constraints

One of this dissertation's main constraints arises from the data used in evaluating the model's accuracy, as we are unable to conduct our own data gathering process, by recording electrocardiograms, blood pressure and chest wall movement and have instead chosen to gather data from research conducted by BOWMAN et al. (1997*b*), looking into the differences associated with age and endurance exercise training on the baroreflex function of healthy subjects.

1.3 Report Structure

The remaining chapters of this report have been structured as follows:

1. Literature review, presenting relevant work in the area, detailing the papers that were taken into consideration when working on this dissertation project, and also offering information on the programming language and packages used in the development stages
2. Requirements, presenting a breakdown of the project's objectives, separated into smaller achievable goals, with considerations given to methods of testing and analysis

3. Design analysis, describing the project's design approach based on some of the elements presented in the requirements section
4. Implementation of the model, an overall look into how de Boer's model was implemented in Python, describing the process and giving consideration to the testing methods used for verification and evaluation of the model
5. Findings and results at the end of the project, with plots to visualize the results obtained, and an analysis of the goals achieved
6. Discussion around further work that can build on this project, or ways the model implemented could be improved
7. Conclusion, bringing together all the previous chapters, in a short review of the work done for this project, from start to end.

1.4 Project relevance to degree

As a student in the Computer Science department, I can confidently say that this project has been closely related to my degree programme. Throughout the development of this project, I have made use of many of the skills developed during my years of study, putting together information from different modules, and skills gained from working on projects as part of my degree.

My degree programme has definitely improved my critical thinking and problem-solving skills, which I have used as part of this project. I have had to analyse the project description, create project requirements, and develop against the set of requirements I set out. Small problems have come up quite often throughout the project, but that was something I was expecting from the beginning, and having the experience of past projects, I was confident that I would be able to use my critical thinking and solve any issues found.

Moreover, in my degree programme, I have taken modules that involved working in Python, working with large sets of data, learning data driven computing techniques, which helped me in the development of this project. Finally, as part of this project, I performed frequency domain analysis, computing the power spectra of a time series array of values for a signal. Frequency analysis is something I have been introduced to before during my degree, but for this project, I have had to do some further research on this area to improve my understanding, which has been an enjoyable process.

Chapter 2

Literature Review

2.1 Overview

This chapter will focus on describing and presenting relevant work done by other authors in the field of heart rate variability analysis in relation to the work which will be carried out as part of this dissertation project. Papers which have implemented heart rate and blood pressure models, as this project intends to do, will be analysed, and differences in the approach taken will be presented. Furthermore, this chapter includes a review of the paper from which data was gathered for the evaluation of this project, as well as analysis and discussion around papers that present work done in the heart rate variability analysis field.

Finally, this chapter includes a review of the interactive development environment selected, the programming language used, and the packages imported for this language, that helped in the development of the project.

2.2 Heart rate variability in research papers

2.2.1 Heart rate variability analysis research

Complex systems can be defined by the characteristic patterns of variation over time they present. Although complex biological systems are composed of parts that have dynamic, interdependent and non-linear relationships between them, they actually present a very robust systemic stability. Seely & Macklem (2004)

Heart rate variability analysis is a very complex research topic, with a very high degree of importance, as it has been shown that the values of heart rate variability change in the presence of infections or disease, as well as correlate with severity, Ahmad et al. (2009), and further research on the implementation of a model capable of accurately reproducing this variability can help in early prevention and diagnosis in infected patients.

According to Whittam et al. (2000), basic research on human subjects presents difficulties, due to the high complexity of the human cardiovascular control system, as it is not possible to vary one signal, without observing changes in the other signals of the system.

This is why there is a great need for a model that could provide a framework for scientists and researchers to observe beat-to-beat heart rate variability and how alterations in values of one signal influence the other signals.

In 1987, DeBoer et al. (1987) proposed a beat-to-beat heart rate model, analyzing hemodynamic fluctuations and baroreflex sensitivity in humans. The model proposed has the objective to study the relationship between short term fluctuations in blood pressure and heart rate variability, computing values of the feature signals for each beat based on values obtained from previous beats. Before 1987, deBoer had previously published another paper De Boer et al. (1985), which presented an analysis of the relations between short term blood-pressure fluctuation and heart rate variability. This paper, published in 1985, presented a very simple beat-to-beat model of the cardiovascular system. The model presented limitations in being able to explain the shape of the phased spectrum when comparing variability in systolic pressure against heart rate interval variability. The model published later, in the 1987 paper, improved on the results of the previous paper, showing that the model was able to reproduce the power spectra of heart rate variability in a normal subject, as well as being able to simulate a responses to Phenylephrine injection, a blood-pressure increasing drug.

Following from deBoer's paper,Whittam et al. (2000), published a study in which they examined the ability of deBoer's model to reproduce short term variability in heart rate and blood pressure from normal subjects. The model from deBoer's paper was used as an initial starting point, due to its simplicity, with the eventual aim of improving the existing model, by including physiological interactions. This study has been successful, proving that deBoer's model was able to reproduce heart rate variability in healthy subjects, and improving the accuracy of its predictions by making small additions to the model.

In this dissertation project, we investigated the capacity of deBoer's model to reproduce short term heart rate and blood pressure fluctuations recorded from three different subject groups.

Following our research in this area, we have concluded that there is a gap in research papers in evaluating deBoer's model against real data gathered from people with different mean heart rate and blood pressure values, as Whittam et al. (2000) and deBoer have focused on evaluating the accuracy of the model against data gathered from normal healthy subjects.

As it can be seen from Bowman et al. (1997*a*) and BOWMAN et al. (1997*b*) There is an age-associated reduction in baroreflex sensitivity values in elderly people. Moreover, only small increases have been seen in the values for healthy elderly people undergoing aerobic training, in the high-frequency range (0.15-0.35Hz) and mid or low frequency range (0.05-0.15Hz), when compared to sedentary elderly people.

According to A.J. Bowman, the slope of pulse interval against systolic blood pressure is a measure of baroreflex sensitivity. The baroreflex sensitivity can be described by the alpha-index, which is a function of frequency. Several studies suggest that the mid-frequency range component of the alpha-index is a representative of the sympathetic activity controlling heart rate and vasomotor activity, while the high-frequency range component is representative of parasympathetic activity. Akselrod et al. (1985)

The subject groups consisted of healthy sedentary elderly people, endurance- trained elderly people and healthy young people, and the data representing these groups was not physically acquired as part of this project. We have used data according to BOWMAN et al. (1997*b*), which gathered the data from the three subject groups to determine differences associated with age and endurance exercise training on the baroreflex function on healthy subjects.

Moreover, the frequency ranges we have decided to study when analysing the accuracy of our model were different from those analysed in Richard's paper. The frequency ranges we are analysing are 0.05-0.15Hz and 0.15-0.35 Hz, whereas Whittam et al. (2000) inspected values over three frequency ranges: very low frequency(0-0.04Hz), low frequency(0.04-0.15Hz) and high frequency(0.15-0.40Hz).

In suggesting further work, we have taken into consideration research conducted by Chiew et al. (2019), presenting how machine learning models, based on heart rate variability can increase the rate of prediction in patients suspected of sepsis, by early identification of high-risk patients. According to Calvin J Chiew, a machine learning model using heart rate variability analysis can improve early prediction and detection in patients suspected of sepsis, when compared to more traditional methods.

2.3 Beat-to-Beat Model

According to Campos (2017), the heart rate variability represents a measure of the variation in time between successive heartbeats. The autonomic nervous system controls this specific variation along with the sympathetic(fight-or-flight mechanism) and parasympathetic(rest system) branches of the **ANS** [1].

DeBoer et al. (1987) presents a beat-to-beat heart rate model, created by a set of difference equations, which are presented below. The purpose of the model was to help receive more information about **CVS** [2]. In this system, n represents the number of beats and I,S,D,T represents the RR interval, systolic, diastolic pressure and peripheral resistance with arterial compliance. Furthermore, $i(n)$, $s(n)$, $s'(n)$, $d(n)$, $p(n)$, $t(n)$ represent the beat-to-beat deviations: RR interval, systolic pressure, effective systolic pressure, diastolic pressure, pulse pressure, arterial time constant with c being arterial compliance. The **OPs** [3] were equalled with normal human values in the following way: $S = 120mmHg$, $D = 75mmHg$, $I = 800ms$ and $T = 1,425ms$.

We know that:
$i_n{=}I_n - I$,
$s_n{=}S_n - S$,
$d_n{=}D_n - D$,

[1]Autonomic Nervous System
[2]Cardiovascular System
[3]Operation Points

t_n=$T_n - T$.

In the **Effective Pressure**, the first equation in the model presents the effective systolic pressure. According to *Regulation of Blood Pressure: Short Term Regulation Baroreceptors* (2020), we know that baroreceptors increase proportionally with the aorta sinus.

$$S\prime(n) = F(S(n)) \tag{2.1}$$

where $F(S) = 120\ +18arctan[\frac{(S-120)}{18}]$

$$s\prime(n) = f(s(n)) \tag{2.2}$$

In the **Baroreflex on Heart Rate** we have the second equation which presents the RR-interval with its current length and the way we can determine its value:

$$I_n = a_0 * S'(n) + \sum_{k>0} a_k S'(n-k) + c_1; i_n = I_n - I \tag{2.3}$$

$$i_n = a_0 s'(n) + \sum_{k>0} a_k s'(n-k) \tag{2.4}$$

a_0 = vagal contribution and a_k = sympathetic contributions a_k with $k > 0$

In the **Baroreflex on Pheripheral Resistance**, the next equation from the model presents R_n [4] and T_n [5], with C being constant. The minus is a representation of the increased pressure and, therefore, decreased resistance.

$$T_n(= R_n C) = T^* - \sum_{k>0} b_k S'(n-k) \tag{2.5}$$

With $b_k = 2, 4, 6, 4, 2$ mmHg and $k = 2, 3, 4, 5, 6$. The third equation is:

$$t_n(= r_n C) = -\sum_{k>0} b_k s'(n-k) \tag{2.6}$$

In the **Properties of Myocardium**, the next equation is dependant on the previous interval and γ represents the ratio of the pulse.

$$P(n) = \gamma I(n) + c_2 \tag{2.7}$$

with $P(n) = S(n) - D(n)$

The **Windkessel equation** represents the combination of the beat-to-beat deviations, $I(n)$, $S(n)$, $T(n)$, which lead to a new value of the diastolic pressure "$D(n + 1)$".

[4]Peripheral resistance

[5]Time constant of the arterial Windkessel

$$D(n) = c_3 S(n-1) \exp \frac{-I(n-1)}{T(n-1)} \tag{2.8}$$

The **Simulation of Resting Data** is an important stage as it is important to include respiratory influence in DeBoer et al. (1987) model. The first time the respiratory can be observed is in the pulse pressure. We know that $p(n) = s(n) - d(n)$.

Therefore, from the equation $p(n) = \gamma i(n-1)$ we get this:

$$p(n) = \gamma i(n-1) + Asin(2\pi f_{resp} \sum_k I_k) \tag{2.9}$$

2.4 Programming Language

2.4.1 Python

The language used for developing this dissertation project was Python3. Python is a general-purpose high-level programming language, with a simple and easy to learn syntax that emphasizes readability and helps its users increase their productivity. Some of the fields where Python is commonly used include:

- Scientific and mathematical computing
- Big data(machine learning, data mining)
- System automation and administration
- Finance(due to its excellent data analysis capabilities) Python focuses on simplicity and pragmatism as it deliberately tries to make the code written as readable and elegant as possible. What is more, Python has built-in garbage collection, which means that its memory allocation and deallocation is automatic. It supports both an object-oriented approach and a functional programming approach, and, as it is an interpreted programming language and there is no compilation step, code can be written, tested and debugged fast. Furthermore, the language supports modules and packages, thus encouraging modularity, and allowing programmers to easily import and reuse external libraries that can be of help in the projects they are working on.

The main reason why I chose Python over other programming languages when developing this project was for its excellent data analysis and data visualization capabilities. There are several open-source libraries available online that have been specifically developed to help users perform data analysis and visualization that can be imported in your project and used straight away. (*What is Python?* (2020))

2.5 IDE used

IDE stands for integrated development environment and it is a piece of software which allows the programmer to write, run and test code. Furthermore, IDEs provide debugging features, with the ability to stop the execution of the code at 'breakpoints', points which the programmers specifically choses. This has proven to be extremely helpful, as it has allowed me to evaluate the values of variables in my code at different points in the execution of the program, and has helped me detect and fix mistakes easier and faster, whenever I would detect a bug that either stopped the code from running or produced unexpected results. The IDE I have chosen for this project is ‘Spyder’ or ‘The Scientific Python Development Environment’. As the name would suggest, ‘Spyder’ is and IDE that is perfectly suited for scientific programming. The IDE offers several windows which allow the user to quickly switch between plots produced, variable values and the Python console, making it easier to debug and trace the values in real time, as well as to visualize the results produced by the code. Furthermore, Spyder offers built-in integration with many popular external packages, such as NumPy, Scipy and Matplotlib, that I intended to use in the project, meaning that I did not have to manually install them, and instead I could simply import them and start using the functions provided by these packages in my project.(*SPYDER* (2020)) such

2.6 Libraries and Framework

2.6.1 NumPy

NumPy[6] is an open source library, created in 2005, that offers support for multidimensional arrays and matrices and basic tools to perform calculations on these arrays. NumPy provides the user the computational power of low-level languages such as C or Fortran in Python, a high-level language that is easier for scientists without a background in programming to use. The library is widely used in the scientific world, as its comprehensive numerical computing tools and powerful n-dimensional arrays are extremely helpful in Data Science and Machine Learning projects. (*NumPy* (2020))

2.6.2 Matplotlib

Matplotlib[7] is an open source Python library. It is the brainchild of John Hunter, and it has received a tremendous amount of support from the open-source development community, making it a piece of software used by scientists all over the world. Matplotlib offers tools to help with data visualizations, helping developers create static, animated and interactive plots. The visualizations created with the help of this library are highly customizable, allowing the users to set font properties, axes properties and tailor the style of a plot to their specific needs. (*Matplotlib: Visualization with Python* (2020))

2.6.3 Scipy

Scipy[8] is a community-driven library, and, as Numpy, it is part of the Scipy ecosystem of open-source software for mathematics, science and engineering. Scipy enables the developer to work with Numpy arrays, and perform different tasks on those arrays. Few examples include interpolation, integration or Fast-Fourier-Transforms. Scipy offers a signal module, which has been used in this dissertation project, allowing us to convert time-domain values into frequency domain values. Like Numpy and Matplotlib described above, Scipy is a library that is widely spread in the scientific world, as it allows users to scientific stuff with it. (*Scientific computing tools for Python* (2020))

[6] https://numpy.org/

[7] https://matplotlib.org/

[8] https://www.scipy.org/about.html

2.6.4 Unittest

Unit Test[9] is a unit testing framework for Python. Similar to other unit testing frameworks from different languages, it supports test automation, setup and shutdown code for tests, and using mocks in tests. Mocks allow the programmer to test in isolation. If one function that is being tested has other external dependencies, those dependencies can be mocked with objects that simulate their behaviour, enabling the program to verify that the function being tested does what he is expecting.(*unittest–Unit testing framework* (2020))

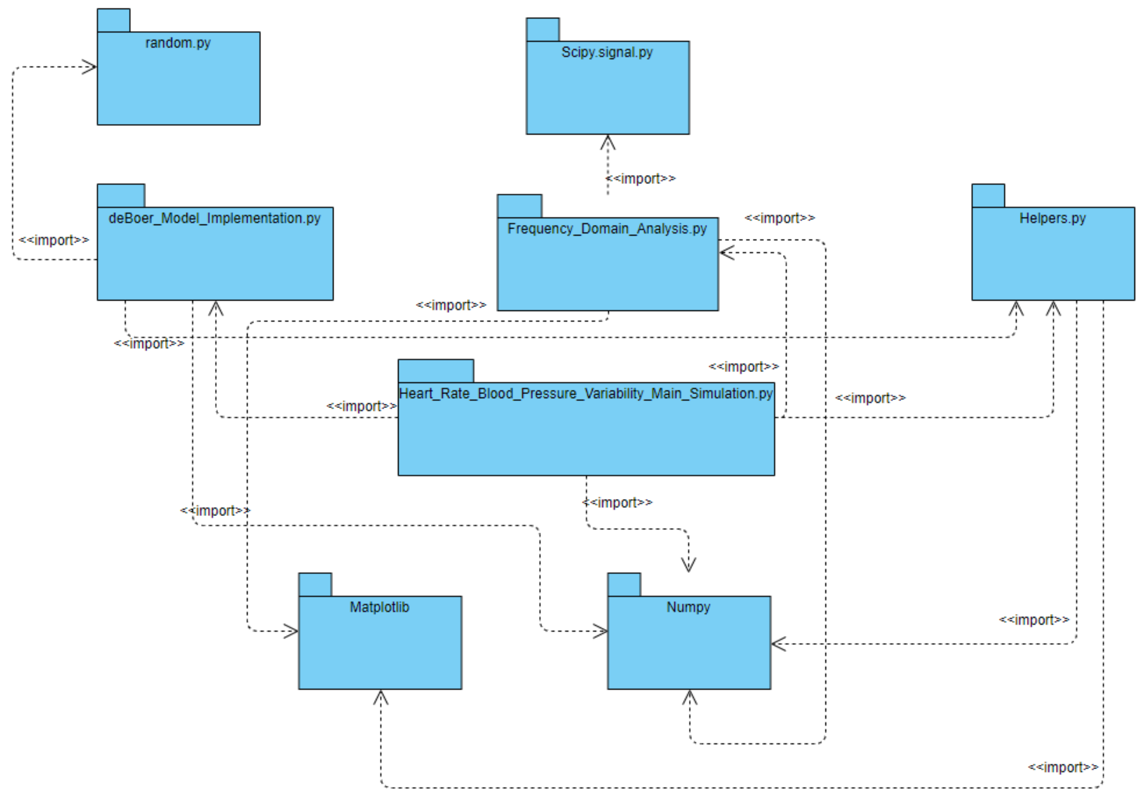

random.py
Scipy.signal.py
deBoer_Model_Implementation.py
Frequency_Domain_Analysis.py
Helpers.py
Heart_Rate_Blood_Pressure_Variability_Main_Simulation.py
Matplotlib
Numpy
<<import>>

Figure 2.1: *Package Diagram*

[9]https://docs.python.org/3/library/unittest.html

Chapter 3

Requirements and Analysis

3.1 Overview

This chapter will present a breakdown of the project's objectives, into smaller steps, with an analysis of the approach taken and the options considered. Furthermore, this chapter will present the set of data used in evaluating the accuracy of the model being developed, and how the evaluation will be performed between simulated data and real data.

3.2 Requirements

The project will implement deBoer's model to simulate heart rate and blood pressure variability in a range of healthy and endurance trained subjects. The project will also perform frequency analysis on the simulated data and compute mid-frequency and high frequency mean values for heart rate variability and systolic blood pressure from the time-domain values, as well as baroreflex sensitivity values. The final step will be comparing the simulated data against real data from the subjects tested, analysing the accuracy of the model in reproducing heart rate and blood pressure variability.

The requirements will be ranked based on their importance in one of the following three categories: LOW, MEDIUM and HIGH, as seen below in **table 3.1**.

Requirements	**Priority**
Find research papers inspecting heart rate variability and baroreflex function in different subject groups	HIGH
Decide on a set of data from one of the research papers investigated to use to evaluate the accuracy of the model to be developed	HIGH
Implement deBoer's model with mean values for systolic pressure, diastolic pressure, heart rate variability and arterial time constant as input parameters	HIGH
Generate Gaussian noise sample values to be added to the model calculations for pulse pressure and heart rate variability	HIGH
Plot time domain values computed for systolic pressure, diastolic pressure and heart rate variability	MEDIUM
Use time series measurement of values to estimate power spectral density using Welch's method for Systolic pressure and heart rate variability signals	HIGH
Compute baroreflex sensitivity and heart rate variability at mid-frequency and high-frequency	HIGH
Comparison and evaluation of the results obtained from the simulations against real data from the research paper chosen	HIGH
Modifications to the model as needed to improve accuracy of results	MEDIUM
Implementation of unit testing for the code developed	LOW

Table 3.1: *Main project requirements, labelled as LOW, MEDIUM OR HIGH priority.*

- Research papers inspecting heart rate variability and baroreflex sensitivity from different subject groups will have to be found, as they will provide the data sets for testing the model.
- From the research papers inspected, one will be selected as the focal point of comparison between the simulation and real values.
- deBoer's model will be implemented in code, with mean Systolic and Diastolic blood pressure, mean Heart Rate Variability and mean Arterial Time constant as input parameters, as well as the total number of beats for each the model will simulate time-domain values.

- The time-domain values will be computed to allow to visualize the simulated data and provide a first step in assessing the accuracy of the model
- Frequency domain analysis will be performed on the set of simulated results, converting from time-domain to frequency domain and computing baroreflex sensitivity and mid and high frequency mean values for Systolic Pressure and Heart Rate variability. This values will then be evaluated against real data from the subjects tested in the paper selected
- As a final step, some of the model's parameters such as the gaussian noise samples or the ratio of pulse pressure to preceding RR interval can be modified, in order to increase the accuracy of the model, as needed.
- During the development of the code, unit tests will be added to ensure functions perform as expected, and changes can be made with confidence, knowing that if any new additions break current functionality this will be immediately observed and fixed.

3.3 Data extraction

After researching through academic papers documenting heart rate variability in different groups of people, the paper that has been chosen to evaluate the model developed against is 'Baroreflex function in sedentary and endurance-trained elderly people'.

The initial aim of this paper is to determine the differences associated with age and training levels on heart rate variability and baroreflex function on healthy subjects. This paper has conducted its research following 3 groups of people: healthy sedentary elderly people, endurance-trained elderly people and healthy sedentary young people. The values of the baroreflex sensitivity were calculated at high frequency(0.15 - 0.35 Hz) and mid-frequency (0.05 - 0.15 Hz).

For analysis, 26 healthy elderly people, 8 endurance-trained elderly people and 8 healthy sedentary young people were available for analysis. All subjects were non-smokers and had normal past medical history, taking no medication.

Information regarding subject details can be seen in the table below. For the aim of this project, from the information provided by the paper, only resting RR interval and Resting Systolic Blood pressure were input as parameters to our model.

Subject Info	Elderly People		**Healthy Seden. young people**
	Healthy Sedentary	**Endurance trained**	
Resting heart rate(min$^-1$)	68 ± 11	58 ± 12	63 ± 9
Resting RR Interval(ms)	958 ± 122	1069 ± 160	975 ± 173
Resting systolic BP(mmHg)	132 ± 23	134 ± 13	126 ± 13

Table 3.2: *Data Extraction*

3.4 Approach analysis

The approach taken was to create the model with the mean values of systolic and diastolic blood pressure, heart rate variability and arterial time constant taken as input from the user. Furthermore, values for total number of heart beats to be modelled, and initial values for the deviations of the functions were also user inputs.

This ensured that the model is flexible enough to work for several groups of people, and allowed for testing a vast range of scenarios, increasing confidence in the results obtained. Furthermore, developing the model this way, ensures that further work can be done using the model with different subject groups, without needing modifications for each particular group.

Once the simulations are run, and time domain values are calculated by the model, frequency domain indices of heart rate and blood pressure variability are obtained, to compare the output from the model with the real results from the paper selected.

Plots of time domain values and power spectrum of the systolic and diastolic blood pressure and the heart rate variability signals are generated, allowing for clear data visualization.

Frequency values are split into mid-frequency (0.05 - 0.15 Hz) and high-frequency (0.15 - 0.35 Hz) as per the research paper taken as reference, and mean values for the signals simulated are calculated over these frequencies.

3.5 Testing and Evaluation

As the code implementing deBoer's model is developed, unit tests are created for the functions. There were two main reasons behind writing unit tests.

Firstly, although initially it might have looked as an activity that is bringing down productivity, as the time spent writing unit tests could be spent further developing the model and then testing everything at the end, past-experience have shown that unit tests are very useful in detecting changes that break existing functionality and small bugs do not go unnoticed.

Secondly, having high unit-test coverage gives confidence that the model developed will be able to be used in future work, and if there are any changes that need to be made to the model, those changes can be made knowing that any issues introduced will be quickly spotted through failing tests and will be fixed.

In order to evaluate the model's accuracy the following approach will be taken:

- Firstly, in order to validate the model, the following values will be set for the input parameters. Mean systolic pressure = 120 mmHg, mean Diastolic pressure = 75 mmHg, mean heart rate variability = 800 ms and mean arterial time constant = 1425 ms. These values represent normal human values and they were also used in deBoer's initial paper(1). The total number of beats will be set to 150. The generated time-domain plots will be compared against the ones from deBoer's paper. High-similarity will give confidence that the model has been implemented correctly.
- Secondly, the power spectrum will be computed from the time-domain values and plotted against the frequency for the Systolic Pressure and Heart Rate Variability signals. Again, the resulting plots will be evaluated against plots from deBoer's paper, looking for high-similarity.
- Finally, after the model has successfully passed the two initial evaluations, we will run simulations using the data we extracted from the selected paper. The values obtained for the average Heart Rate Variability over mid-frequency and high-frequency and the baroreflex sensitivity in the two frequencies will be evaluated against paper data, for each of the subject groups being observed.

3.6 Summary

This chapter has focused on describing the requirements of the project, ranking them as either low, medium or high based on their importance to the project.

Furthermore, in this chapter the data that will be used when running the model has been described, stating the different subject groups that will be considered and the details extracted for each subject. An overview of the approach taken has been described, but a more in-depth look at the exact steps taken, and the computations performed will be presented in the following chapter of this paper.

Finally, this chapter has presented the approach taken towards testing and evaluation. Explanations towards why unit testing has been implemented were discussed, as well as a description of the evaluation methods chosen for this project.

Chapter 4

Design

4.1 Overview

This chapter will focus on the approach chosen for the project, providing an in-depth explanation on the methods used in the development of the model, explaining how the model calculates values for each one of the signals. Furthermore, this chapter will offer detail into how the final results are computed and compared against real data as well as a discussion about the data visualization techniques used.

4.2 Model Design

The model we have implemented closely resembles deBoer's model. It is a beat-to-beat implementation of the cardiovascular system in which the current values of the features of each beat depend on feature values from previous beats. The workflow is presented in **Figure 4.1**. For the initial values of the deviations of systolic pressure, diastolic pressure, arterial time constant and heart rate variability, the model uses values input by the user, and we have chosen those values as 0 for our model simulations.

The model operates inside a loop which runs from 1 to the total number of beats input to the model. At each iteration, two random Gaussian noise sample values are calculated. One is then added when pulse pressure is computed, and the other one is added to the computation of the heart rate variability. Following the sampling of the two random noise values, the respiratory signal is evaluated as sine function of elapsed time. We have set the initial respiration value as 0.25. Initially, we considered using a value of 0.3Hz, as this represents slightly more than 3 breaths per minute, but after further evaluation and testing, the value that produced the most accurate results proved to be 0.25Hz.

Next, a value for the deviation of pulse pressure from the mean value is calculated, based on the previous values of the deviation of the heart rate variability, the Gaussian noise sample value and the respiration value.

The diastolic pressure is calculated next, and this value depends on the previous beat values of all the other features, as well as their mean values. After the diastolic pressure is obtained, its deviation from the mean is computed as the difference between the current value and the mean value. The diastolic pressure deviation is then used, along with the pulse pressure deviation, to compute the value of the systolic pressure deviation.

Once the systolic pressure deviation is obtained, it can be added to the mean systolic pressure value to generate the Systolic pressure value for the current beat. The effective systolic pressure is then calculated as an inverse trigonometric function of the tangent of systolic pressure.

The deviation of the heart rate variability is determined by the previous values of the effective systolic pressure, a weighted sum of previous systolic pressure values, plus the random Gaussian noise generated. Following deBoer's model, we have chosen to give our model a triangular weighting function for obtaining the deviation of the heart rate variability interval. The values of a_ks were taken as a_k= 1, 2, 3, 2, and 1 ms/mmHg for $k = 2, 3, 4, 5, 6$. All other values outside this range were set to 0. The next value to be calculated was the arterial time constant deviation. Again, using deBoer's original technique, we used a triangular weighting function for b_k and the values were taken as b_k= 2, 4, 6, 4, and 2 for $k = 2, 3, 4, 5, 6$ and set to 0 for all other values of k. The arterial time constant deviation was also dependent on the previous effective systolic pressure values. Finally, the arterial time constant for the current beat, T(n) is computed as the sum of the mean value and the deviation.

Inside the loop that runs from n = 1 to the total number of beats, each value computed for systolic and diastolic pressure, heart rate variability and the arterial time constant is stored, along with the current n value, inside a dictionary.

4.3 Data Visualisation

4.3.1 Time domain representation

Once the main loop has finished running after n reaches the total number of beats, the dictionaries that contain the stored values of Systolic and Diastolic Pressure and the heart rate variability are then used to generate time-domain plots. The two blood pressure values are plotted on the same plot, with different colors and a label identifying them, while heart rate variability is plotted on its own. As it can be seen from the figure below, both plots are the time-domain representations of the features, with the x-axis representing the time, and are plotted using Python's matplotlib package.

4.3.2 Frequency domain representation

In order to generate frequency domain representations, the mean value is extracted from both the heart rate variability interval and the systolic pressure time series values. Then, the power spectral density using Welch's method, Welch (1967), a method based on the concept of using periodogram spectrum estimates, which represent the result of converting a signal from time domain to frequency domain. The reason we have chosen to evaluate power spectral density using Welch's method and not the standard periodogram, is that this method reduces noise

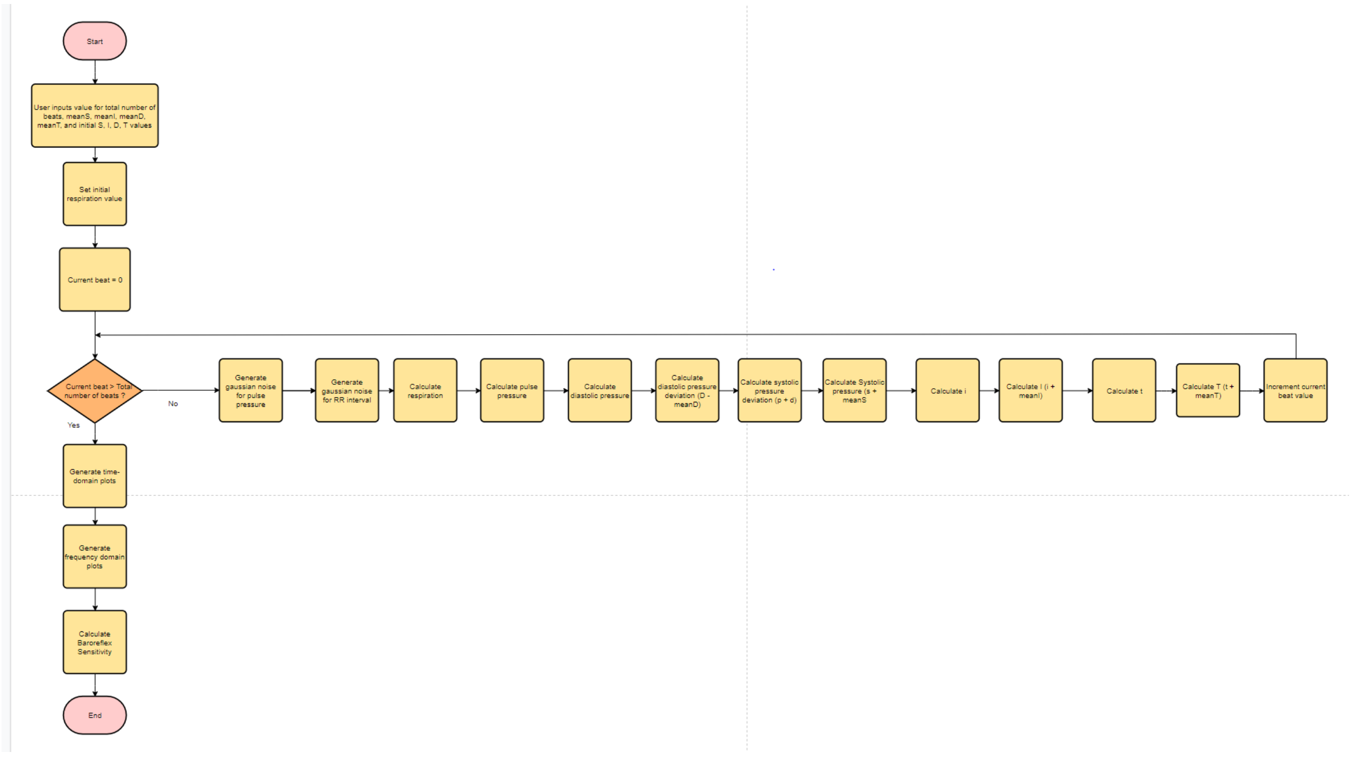

Start
User inputs value for total number of beats, meanS, meanI, meanD, meanT, and initial S, I, D, T values
Set initial respiration value
Current beat = 0
Current beat > Total number of beats ?
No
Yes
Generate gaussian noise for pulse pressure
Generate gaussian noise for RR interval
Calculate respiration
Calculate pulse pressure
Calculate diastolic pressure
Calculate diastolic pressure deviation (D - meanD)
Calculate systolic pressure deviation (p + d)
Calculate Systolic pressure (s + meanS
Calculate i
Calculate I (i + meanI)
Calculate t
Calculate T (t + meanT)
Increment current beat value
Generate time-domain plots
Generate frequency domain plots
Calculate Baroreflex Sensitivity
End

Figure 4.1: *Flowchart of the system*

in the estimated power spectra, thus offering an improvement on the periodogram method.

With the power spectral density for the signals computed, these values are then plotted against their frequencies to visualize the frequency domain representations of both systolic pressure and heart rate variability.

4.3.3 Computing heart rate variability in mid-frequency and high-frequency

To compare the model output with real data from the 'Baroreflex function in sedentary and endurance-trained elderly people' paper, mean values for the mid-frequency range (0.05 - 0.15 Hz) and high-frequency range (0.15 - 0.35 Hz) have to be calculated. This is done using the previously calculated power spectral densities and frequency values. The signals' frequencies range from 0 to 0.5 Hz therefore anything outside the MF and HF range is not taken into consideration. Values over the 2 desired ranges are stored into separate arrays and mean values are calculated for each one of the two arrays, thus producing I_{LF} and I_{HF} .

4.3.4 Computing and comparing simulated baroreflex sensitivity with real data

The systolic mean pressure values over the mid frequency and high frequency intervals are computed using the same method described above in computed values for heart rate variability over these two frequency ranges.

The **BRS** [1] for each of the simulations is estimated as:

$$\alpha = \frac{(\alpha_{LF} + \alpha_{HF})}{2} \tag{4.1}$$

where

$$\alpha_{LF} = \sqrt{\frac{I_{LF}}{S_{LF}}} \tag{4.2}$$

$$\alpha_{HF} = \sqrt{\frac{I_{HF}}{S_{HF}}} \tag{4.3}$$

Finally, in order to visualize the accuracy of our model, the computed data is plotted against real data, with different shape labels clearly differentiating between the two data sources.

4.4 Summary

This chapter has described the approach taken for the project, and how each one of the values in the model is calculated inside the code's main loop, with current beat feature values depending on previous beat values. Furthermore, this chapter has presented the approach taken in visualizing the data obtained, how the signal is split into Mid-frequency and high-frequency and how the mean values for heart rate variability and systolic pressure over the two intervals are calculated. Finally, we have detailed how the baroreflex sensitivity of the simulated data is computed and compared against real data to estimate the model's accuracy.

[1] Baroreflex Sensitivity

Chapter 5

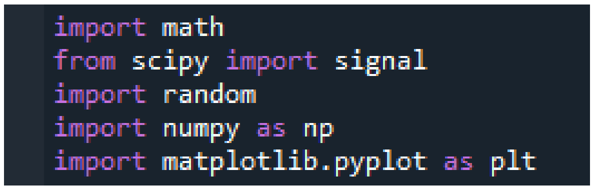
Implementation and testing

5.1 Overview

This chapter will focus on the implementation of the model, and the functions used in the development process of the code. The testing processes will be detailed with examples, both for unit-testing and functional testing, evaluating the system's performance against a standard set of results and the real-live data.

5.2 Implementation of deBoer's model

The model was implemented on a personal computer, using Python and taking advantage of several Python modules such as 'math', 'scipy', 'numpy' and matplotlib, as can be seen from the figure below.



```
import math
from scipy import signal
import random
import numpy as np
import matplotlib.pyplot as plt
```

Figure 5.1: *Python libraries used*

For running a simulation of the beat-to-beat model, *run_simulation*() was defined, which takes as input the values for mean systolic pressure(meanS), mean diastolic pressure(meanD), mean heart rate variability(meanI) and mean arterial time constant(meanT), as well as total number of beats to simulate and initial values for the deviations of the features.

The noise sequences to be added to the pulse pressure and the heart rate variability interval were both generated using python's numpy module and calling the numpy.random.normal() function which generates random samples from a Gaussian distribution. Separate functions for the generation of these random signals have been created, and they are called inside the *run_simulation*() function.

A helper function for assigning initial values to the features of the model has been created. This function takes as input the values to be assigned, creates new variables, assigns them the values passed to the function, and returns the newly created variables. The helper function is called twice inside the *run_simulation*() function, once for assigning the initial values of the deviations, s(0), d(0), i(0), t(0), and once for assigning initial values for the features, S[0], D[0], I[0], T[0].

Before running the main loop that simulates the beat-to-beat model, empty dictionaries for storing time domain values and the current beat values are created.

Inside the loop, for each iteration the feature values are calculated in order, based on their dependencies on previous feature values, by calling helper functions defined outside *run_simulation*() that implement the calculations from deBoer's model, and the current beat number increases by 1. (example: *calculate_effective_systolic_pressure*()):

When the current beat number reaches the total number of beats, the loop is exited, and the dictionaries containing the stored time-domain values are passed as inputs to functions that generate the time-domain plots.

The functions that generate the frequency domain representations convert the time-domain signal arrays into frequency values and compute the spectral density of the signals. Furthermore, they generate values for the two frequency ranges being observed. All these actions are being done by separate functions, which receive the parameters as inputs, do the computations, and return the values. This has allowed us to maintain our functions pure, making the code testable and modular.

As the research paper we have chosen as our main point of comparison for the accuracy of our model presents 3 different subject groups - healthy sedentary elderly, endurance-trained elderly and healthy young people, with different group sizes, in our implementation, we are running the simulations for as many times as there are people in the group.

As an example, when running the simulations for the first subject group, healthy elderly people, as that group included 26 subjects, we have run the simulations for 26 times, with the input values extracted from the mean values of the subject group presented in the paper. Where mean data for a feature was not given in the paper, we have decided to use normal human averages instead. The random Gaussian noise that is sampled at each beat for the total number of beats of each simulation ensures that values are always slightly different, allowing us to replicate values from 26 different subjects.

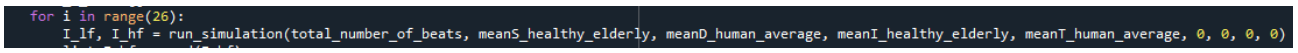

```
    for i in range(26):
        I_lf, I_hf = run_simulation(total_number_of_beats, meanS_healthy_elderly, meanD_human_average, meanI_healthy_elderly, meanT_human_average, 0, 0, 0, 0)
```

Figure 5.2: *Generating simulation that produces as output mean values for heart rate variability over mid-frequency and high-frequency ranges to replicate the 26 healthy elderly subjects from the paper.*

5.3 Coding Traps

For the development of this project's code, we have followed the principles of functional programming, **as one of the requirements we set at the beginning of the project was to develop code that is modular, maintainable and readable and following a strict set of principles ensures that this requirement can be met.**

One of the coding traps that we have tried avoiding when implementing the model was having shared mutable state. Shared mutable state is hard to track, and allowing all the functions in the code access to one variable leaves room for unexpected changes which can lead to undesired and inaccurate results. Furthermore, shared mutable state also makes following the logic of the program harder, as you may have to look outside the scope of the current function being inspected to find the declaration of a variable. Pacheco (2016)

5.4 Unit testing

We have made use of python's unit test module to write unit tests for our project. We have followed a **TDD** [1] when developing code. A TDD approach means that the test is written before writing the code for a function. The test initially fails, then the developer writes code to make the test pass. Once the test passes, the test for the next function is written and the process is repeated.

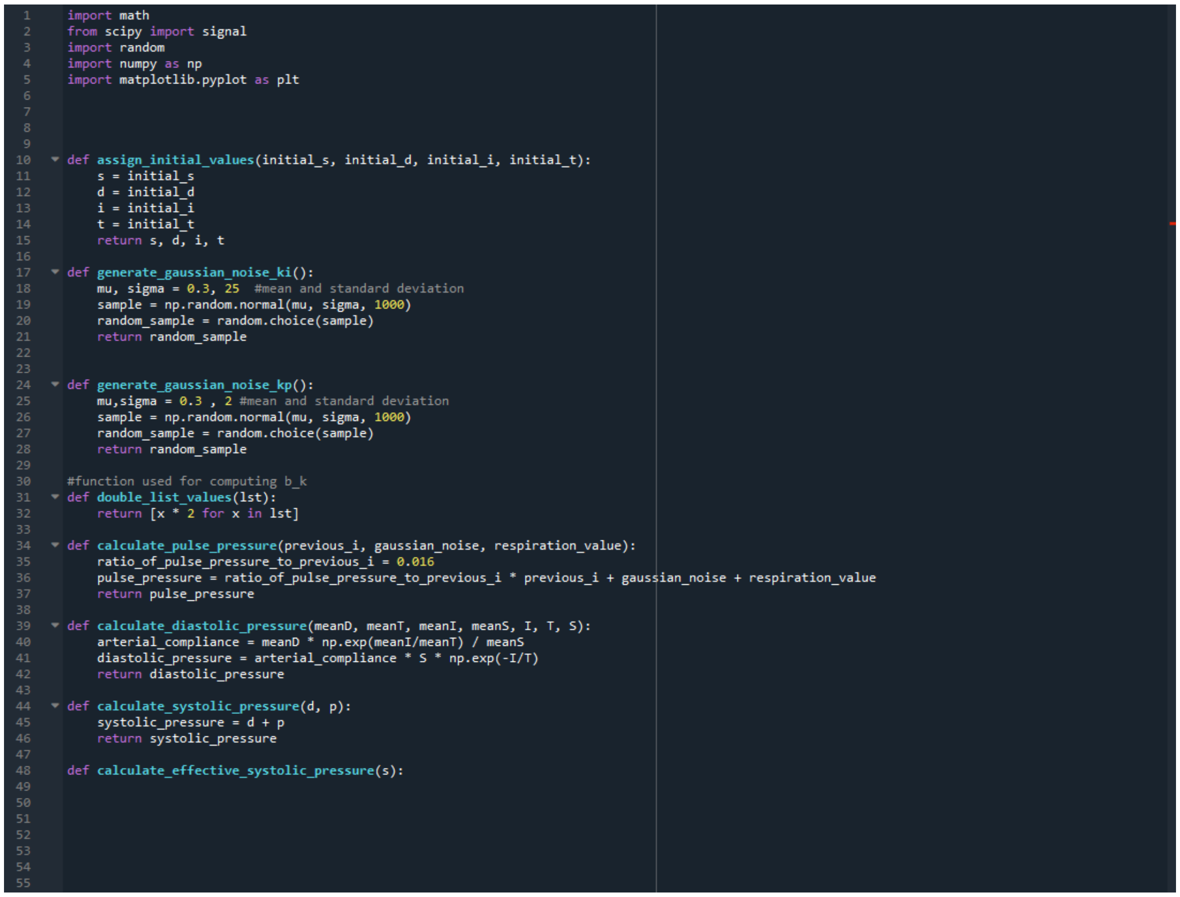
Using this approach forced us to think thoroughly about the code that we were developing, the parameters to be passed as input to a function and the parameters to be returned. Furthermore, the TDD approach has allowed us to have confidence when adding new functions, or when making changes to existing code, that if any bugs were to show up, we would quickly spot them through failing unit tests.

A separate file was created for the writing of unit tests, and the functions to be tested were imported from the Python file where the model was developed.

The TDD approach followed in this project can be seen in the figures below, for the $calculate\_effective\_systolic\_pressure()$ function. Firstly, the function with its input parameters was defined in the file where the model was developed. Then, in the test file, the test for the function was written, with the expected behaviour of the function in mind. After writing the test, the $calculate\_effective\_systolic\_pressure()$ function was implemented, and finally the tests were run.

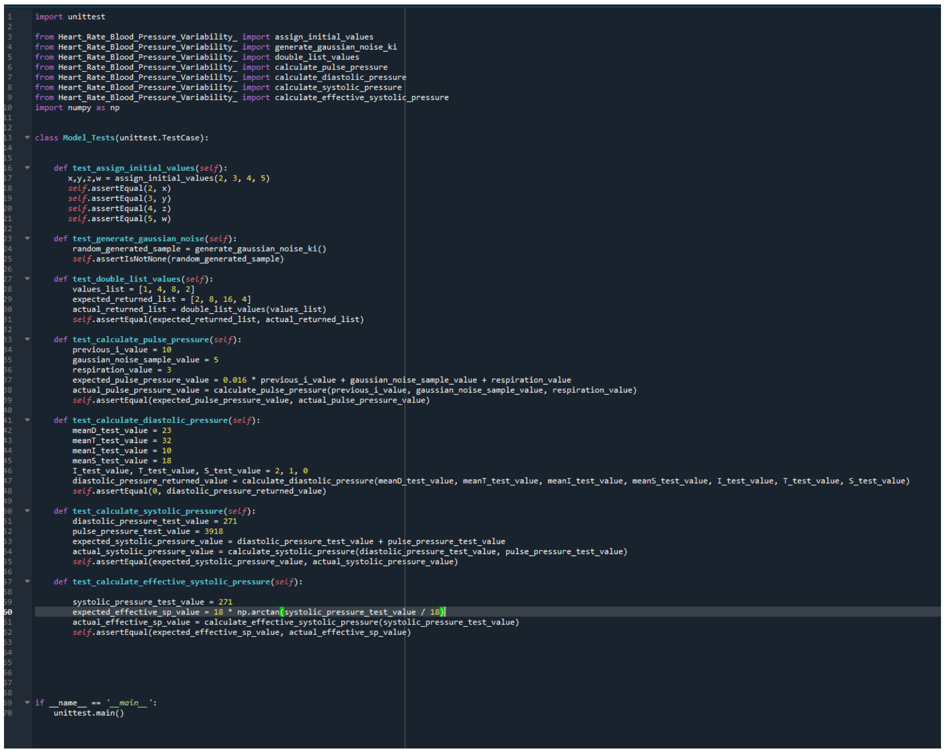
All previous tests and the newly created test passed, which allowed us to continue with the development process for the rest of the code.

[1] Test-driven-development approach



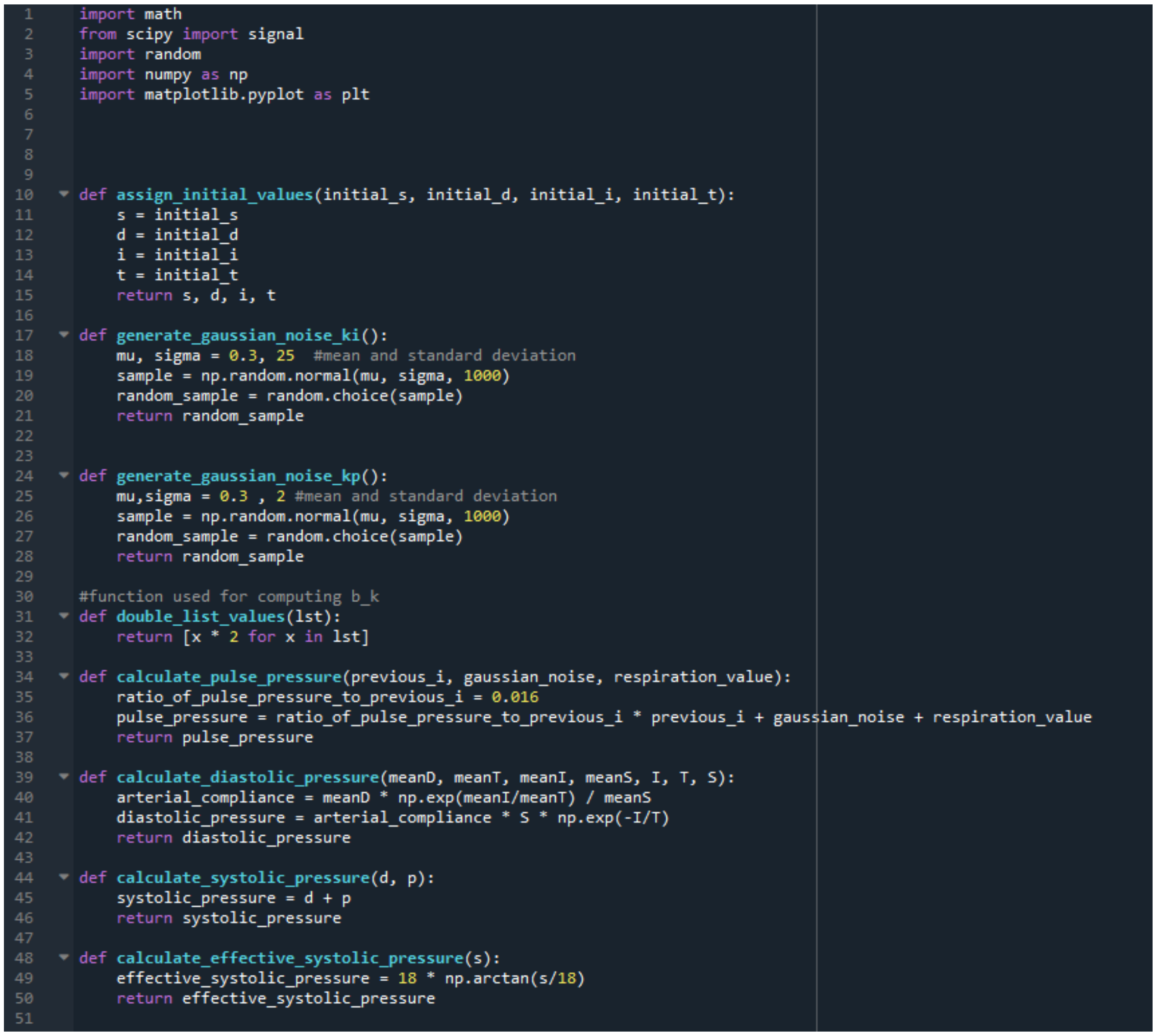
```python
import math
from scipy import signal
import random
import numpy as np
import matplotlib.pyplot as plt




def assign_initial_values(initial_s, initial_d, initial_i, initial_t):
    s = initial_s
    d = initial_d
    i = initial_i
    t = initial_t
    return s, d, i, t

def generate_gaussian_noise_ki():
    mu, sigma = 0.3, 25  #mean and standard deviation
    sample = np.random.normal(mu, sigma, 1000)
    random_sample = random.choice(sample)
    return random_sample


def generate_gaussian_noise_kp():
    mu,sigma = 0.3 , 2 #mean and standard deviation
    sample = np.random.normal(mu, sigma, 1000)
    random_sample = random.choice(sample)
    return random_sample

#function used for computing b_k
def double_list_values(lst):
    return [x * 2 for x in lst]

def calculate_pulse_pressure(previous_i, gaussian_noise, respiration_value):
    ratio_of_pulse_pressure_to_previous_i = 0.016
    pulse_pressure = ratio_of_pulse_pressure_to_previous_i * previous_i + gaussian_noise + respiration_value
    return pulse_pressure

def calculate_diastolic_pressure(meanD, meanT, meanI, meanS, I, T, S):
    arterial_compliance = meanD * np.exp(meanI/meanT) / meanS
    diastolic_pressure = arterial_compliance * S * np.exp(-I/T)
    return diastolic_pressure

def calculate_systolic_pressure(d, p):
    systolic_pressure = d + p
    return systolic_pressure

def calculate_effective_systolic_pressure(s):
```

Figure 5.3: *Defining the function with its input parameter before writing the implementation*



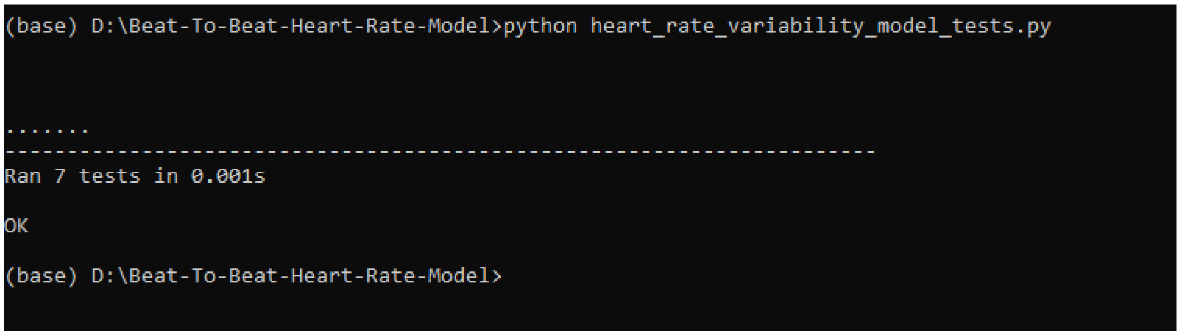
```python
import unittest

from Heart_Rate_Blood_Pressure_Variability_ import assign_initial_values
from Heart_Rate_Blood_Pressure_Variability_ import generate_gaussian_noise_ki
from Heart_Rate_Blood_Pressure_Variability_ import double_list_values
from Heart_Rate_Blood_Pressure_Variability_ import calculate_pulse_pressure
from Heart_Rate_Blood_Pressure_Variability_ import calculate_diastolic_pressure
from Heart_Rate_Blood_Pressure_Variability_ import calculate_systolic_pressure
from Heart_Rate_Blood_Pressure_Variability_ import calculate_effective_systolic_pressure
import numpy as np


class Model_Tests(unittest.TestCase):


    def test_assign_initial_values(self):
       x,y,z,w = assign_initial_values(2, 3, 4, 5)
       self.assertEqual(2, x)
       self.assertEqual(3, y)
       self.assertEqual(4, z)
       self.assertEqual(5, w)

    def test_generate_gaussian_noise(self):
        random_generated_sample = generate_gaussian_noise_ki()
        self.assertIsNotNone(random_generated_sample)

    def test_double_list_values(self):
        values_list = [1, 4, 8, 2]
        expected_returned_list = [2, 8, 16, 4]
        actual_returned_list = double_list_values(values_list)
        self.assertEqual(expected_returned_list, actual_returned_list)

    def test_calculate_pulse_pressure(self):
        previous_i_value = 10
        gaussian_noise_sample_value = 5
        respiration_value = 3
        expected_pulse_pressure_value = 0.016 * previous_i_value + gaussian_noise_sample_value + respiration_value
        actual_pulse_pressure_value = calculate_pulse_pressure(previous_i_value, gaussian_noise_sample_value, respiration_value)
        self.assertEqual(expected_pulse_pressure_value, actual_pulse_pressure_value)

    def test_calculate_diastolic_pressure(self):
        meanD_test_value = 23
        meanT_test_value = 32
        meanI_test_value = 10
        meanS_test_value = 18
        I_test_value, T_test_value, S_test_value = 2, 1, 0
        diastolic_pressure_returned_value = calculate_diastolic_pressure(meanD_test_value, meanT_test_value, meanI_test_value, meanS_test_value, I_test_value, T_test_value, S_test_value)
        self.assertEqual(0, diastolic_pressure_returned_value)

    def test_calculate_systolic_pressure(self):
        diastolic_pressure_test_value = 271
        pulse_pressure_test_value = 3918
        expected_systolic_pressure_value = diastolic_pressure_test_value + pulse_pressure_test_value
        actual_systolic_pressure_value = calculate_systolic_pressure(diastolic_pressure_test_value, pulse_pressure_test_value)
        self.assertEqual(expected_systolic_pressure_value, actual_systolic_pressure_value)

    def test_calculate_effective_systolic_pressure(self):

        systolic_pressure_test_value = 271
        expected_effective_sp_value = 18 * np.arctan(systolic_pressure_test_value / 18)
        actual_effective_sp_value = calculate_effective_systolic_pressure(systolic_pressure_test_value)
        self.assertEqual(expected_effective_sp_value, actual_effective_sp_value)






if __name__ == '__main__':
    unittest.main()
```

Figure 5.4: *Writing the test for the function that would pass after implementation*



```python
import math
from scipy import signal
import random
import numpy as np
import matplotlib.pyplot as plt




def assign_initial_values(initial_s, initial_d, initial_i, initial_t):
    s = initial_s
    d = initial_d
    i = initial_i
    t = initial_t
    return s, d, i, t

def generate_gaussian_noise_ki():
    mu, sigma = 0.3, 25  #mean and standard deviation
    sample = np.random.normal(mu, sigma, 1000)
    random_sample = random.choice(sample)
    return random_sample


def generate_gaussian_noise_kp():
    mu,sigma = 0.3 , 2 #mean and standard deviation
    sample = np.random.normal(mu, sigma, 1000)
    random_sample = random.choice(sample)
    return random_sample

#function used for computing b_k
def double_list_values(lst):
    return [x * 2 for x in lst]

def calculate_pulse_pressure(previous_i, gaussian_noise, respiration_value):
    ratio_of_pulse_pressure_to_previous_i = 0.016
    pulse_pressure = ratio_of_pulse_pressure_to_previous_i * previous_i + gaussian_noise + respiration_value
    return pulse_pressure

def calculate_diastolic_pressure(meanD, meanT, meanI, meanS, I, T, S):
    arterial_compliance = meanD * np.exp(meanI/meanT) / meanS
    diastolic_pressure = arterial_compliance * S * np.exp(-I/T)
    return diastolic_pressure

def calculate_systolic_pressure(d, p):
    systolic_pressure = d + p
    return systolic_pressure

def calculate_effective_systolic_pressure(s):
    effective_systolic_pressure = 18 * np.arctan(s/18)
    return effective_systolic_pressure

```

Figure 5.5: *Writing the function's implementation*



```
(base) D:\Beat-To-Beat-Heart-Rate-Model>python heart_rate_variability_model_tests.py


.......
----------------------------------------------------------------------
Ran 7 tests in 0.001s

OK

(base) D:\Beat-To-Beat-Heart-Rate-Model>
```

Figure 5.6: *Test run results*

5.5 Functional testing

Functional testing represents a type of software testing where the system is tested against the functional requirements. When doing functional testing, the black box technique was used, which is a method where the design and implementation of the system is not known to the tester.

The aim of functional testing is to test the actual output of the system when given a set of input data, by comparing it to the expected output.

For this project, we have performed functional testing in two stages. The first stage involved comparing our model's output to the output from deBoer's initial paper, when the values given as input represented the normal human average values for systolic and diastolic pressure, heart rate variability and arterial time constant. Second stage involved testing the system's output against data obtained from the paper chosen as our main point of comparison, seeing if the values obtained for heart rate variability resemble the values extracted from the paper.Software-Testing-Fundamentals (2020)

5.5.1 Functional testing against a standard set of results

The system has performed well when testing against the standard set of results extracted from deBoer's initial paper. We chose as input the same values that deBoer used for his model, setting the mean average values of the heart-rate to normal human averages, and setting the ratio of pulse pressure to preceding heart rate variability interval to 0.016. The outputs we have looked at were the time-domain plots of systolic and diastolic pressure and of heart rate variability, over a range of 150 beats. As can be seen from the figure below, the values obtained closely resemble the behaviour from deBoer's paper. The values of diastolic pressure and systolic pressure for the simulated number of beats are similar to the real values extracted from real data and the plots follow similar patterns. Similarly, the heart rate variability interval output by the system over the duration of 150 beats is similar to that extracted from real average human subjects. The small differences in values which can be seen for both the blood pressure and heart rate variability interval plots come from the fact that our models were run with mean values, whereas human values will have been very close to those values, but not exactly the same, thus resulting in slight variations of the outputs.

Furthermore, despite the time-domain values computed closely resembling the values we expected to see, we decided to further evaluate our model by computing the power spectra of the signals, as this would allow for a more critical comparison. The power spectra of each signal was computed using Welch's method, Stanford.edu (2011), which is a slight improvement on the standard periodogram function, as it reduces noise in the estimated power spectra.

As can be seen from the images, when comparing our simulated power spectra for the two signals with the actual data, a good resemblance can be seen. The power spectra in the real data only takes into consideration frequencies above 0.05 Hz, but we have chosen to plot and inspect frequencies from 0 Hz. Similar to the real data, our systolic pressure representation shows an initial peak around the 0.1Hz frequency mark and a second, slightly smaller peak around 0.2Hz. The amplitude of the signal goes down to almost 0 towards the 0.5Hz frequency mark. The power spectra of the heart rate variability interval show two peaks, around the 0.1Hz and 0.25Hz frequency marks. Slight differences between the real data and simulated data can be seen in the range 0.3Hz-0.5Hz, where the simulated data signals present spikes in amplitude, while the actual data's signal amplitude goes towards 0. This can be justified from the randomness inserted into the model through the Gaussian noise values sampled with each beat.

From our results, which are presented in Figure 5.7, Figure 5.8 and Figure 5.9, Figure 5.10 we can compare them with the **graph** [2] from DeBoer et al. (1987) that presents the RR-interval and BP(mmHg).

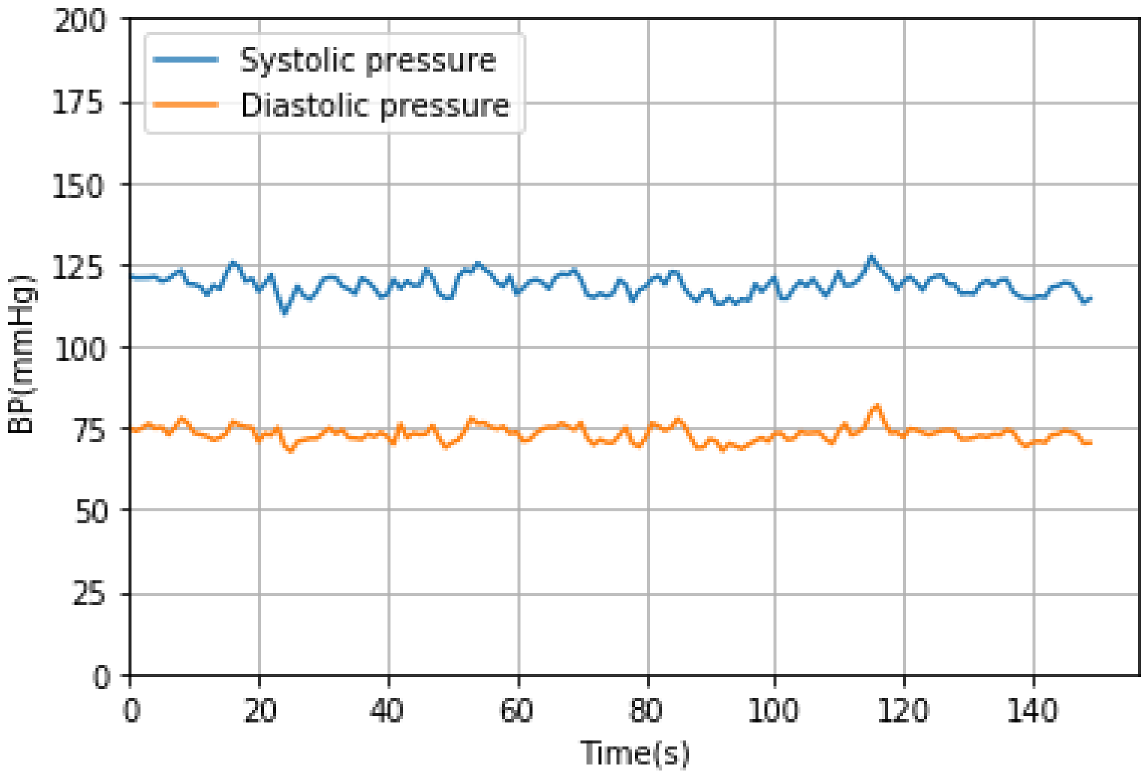


Figure 5.7: *Systolic/Diastolic pressure*

[2]https://journals.physiology.org/doi/pdf/10.1152/ajpheart.1987.253.3.H680

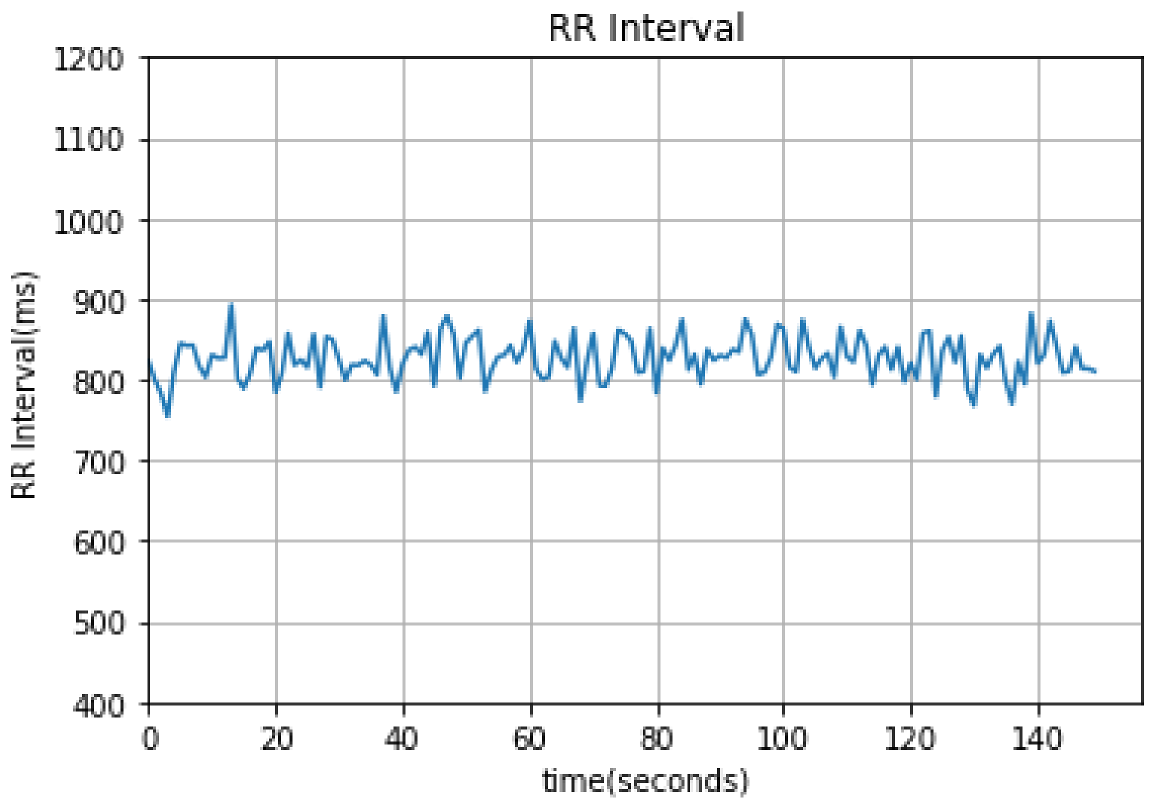


Figure 5.8: *RR interval*

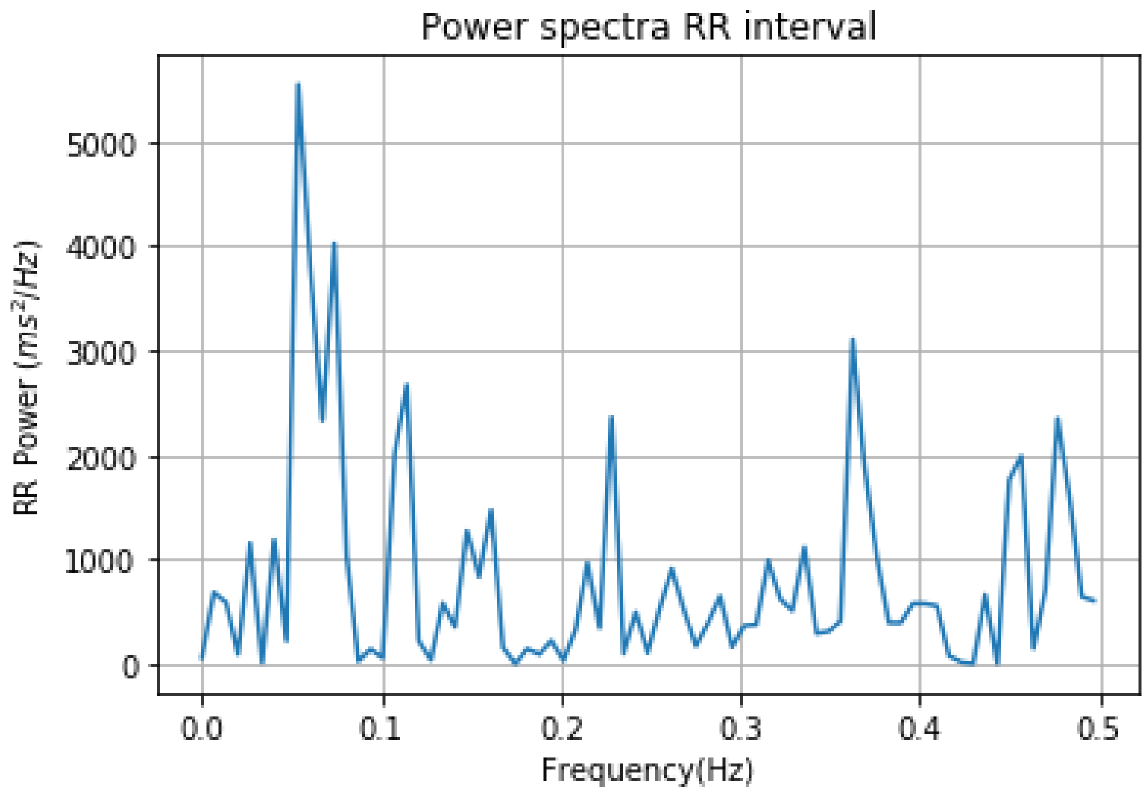


Figure 5.9: *Power Spectra RR*

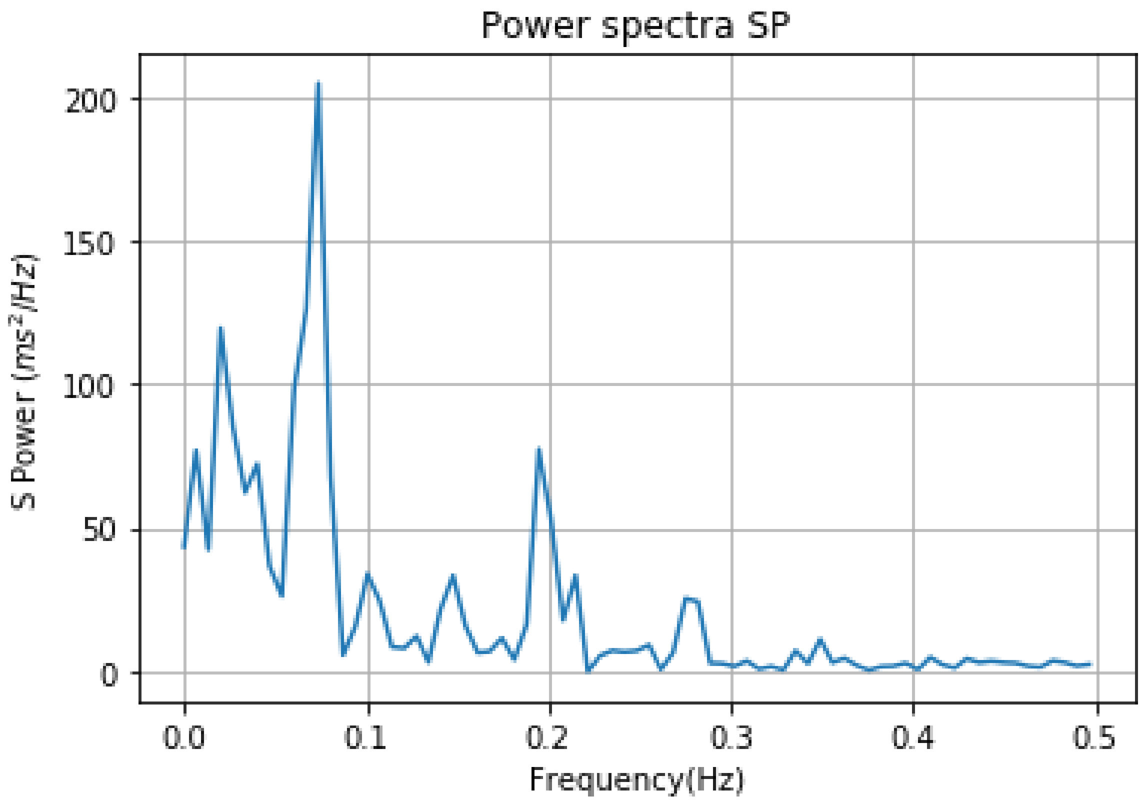


Figure 5.10: *Power Spectra SP*

5.5.2 Functional testing against point of comparison data

The second stage of functional testing of this project was carried out against real-data extracted from the 'Baroreflex function in sedentary and endurance-trained elderly people' paper.

Data provided by the paper is extracted from 3 different subject groups, where the first group represents healthy sedentary elderly people(data collected from 26 individuals), the second group represents healthy endurance trained elderly people(data collected from 8 individuals), and the last one represents healthy young people(data collected from 8 individuals).

As can be seen from the table below, our system behaved well, generating outputs similar to the data gathered from the real paper for the two groups of elderly people. However, in the real data, it can be seen that heart rate variability for both mid and high frequencies presents a clear increase in value from over both frequency ranges. This increase has not been replicated by our model, which has indeed shown an increase in value for healthy sedentary young people, but not as significant as real data.

Subject Type	Simulated Data		Real Data	
	I_{LF}	I_{HF}	I_{LF}	I_{HF}
Healthy Elderly	1642.91	1350.04	2146	1533
Endurance trained elderly	1816.91	1598.89	2227	1701
Healthy young people	2225.17	2014.35	3918	3336

Table 5.1: *Real Data vs Simulated Data*

Another method of comparing and evaluating our model's performance, was to compare the baroreflex sensitivity produced for the subjects against the data gathered from the subject groups.

Baroreflex sensitivity was calculated as:

$$\alpha = \frac{(\alpha_{lf} + \alpha_{hf})}{2} \tag{5.1}$$

where

$$\alpha_{lf} = \sqrt{\frac{i_{lf}}{s_{lf}}} \tag{5.2}$$

$$\alpha_{hf} = \sqrt{\frac{i_{hf}}{s_{hf}}} \tag{5.3}$$

A similar trend to that presented in the analysis of the heart rate variability interval can be spotted here, where our values closely resemble the real data for sedentary elderly people and endurance trained elderly athletes, but not for the healthy young people subject group.

Increases in the value of baroreflex sensitivity can be seen, with the sedentary elderly group showing the smallest mean, and healthy young people showing the highest mean, however the values are smaller than those presented in real data. Furthermore, real data in healthy young people shows a much higher standard deviation then the simulated results.

The much higher standard deviation in the healthy young people subject group can be attributed to human complexity, which our model may not be able to mimic perfectly, considering we are only inputting average values for two of the features specific to the subject group, and the other two feature values are being used as the normal human averages. In addition, we believe that the number of individuals in the subject group is quite small, and allows for higher standard deviation. A larger group of subjects may have had a much smaller standard deviation, and values closer to those we obtained, using mean feature values as input.

5.6 Summary

This chapter has focused on presenting the implementation process of our model. Coding techniques and principles followed have been described, as well as coding traps we had to avoid in order to keep our code clean. Decisions taken in order to closely replicate the real data values in our simulations have been explained, such as the way we simulate each individual in a subject group. Our testing process has been presented, starting with unit-testing and reasons for choosing to follow a test driven approach. Finally, we have presented our functional testing, comparing our system's outputs against both a standard set of results for average human values, and the different subject groups we have chosen as our point of comparison.

// Chapter 6

Results and discussion

6.1 Overview

This chapter will present the results and findings of the project, discussing observations made throughout the development of the project. Furthermore, we will have a look at the original objectives laid out for the project, seeing how we have managed to reach them, or if they were not reached, what blocked us from achieving them. Finally, possible future work will be discussed, both ideas that can build on what has been implemented during this project, and improvements to the current project that could not be implemented this time due to time constraints.

6.2 Findings

Using the model, we have run simulations for each of the three subject groups that we had extracted data for. Considering that the group consisting of healthy elderly people included 26 subjects, while the endurance trained elderly and healthy young people groups included only 8 subjects, we have decided to run 26 simulations when using the input data representative of the healthy elderly group, and only 8 simulations for the other two groups, to replicate the real data as closely as possible.

The results of our simulations are represented in Table 6.1, Table 6.2 and Table 6.3 :

Subject no.	I_{LF}	I_{HF}
1	713.81	1324.69
2	1531.12	1467.24
3	1232.30	1604.74
4	1395.90	1024.60
5	1489.01	638.67
6	1606.55	1574.41
7	1765.07	1232.36
8	1071.24	1048.14
9	1602.51	2051.11
10	1819.58	1228.22
11	1420.12	753.38
12	1472.64	1310.85
13	1676.41	1183.72
14	1695.34	891.23
15	1512.40	1029.007
16	1232.89	2356.25
17	1555.57	1303.86
18	1235.62	1729.009
19	1603.83	1096.09
20	1742.22	1658.87
21	3264.53	1705.48
22	1316.70	1143.29
23	1691.37	1018.19
24	1862.59	1048.03
25	1577.87	1008.01
26	1628.50	1671.40

Table 6.1: *Healthy sedentary elderly people*

Subject no.	I_{LF}	I_{HF}
1	1173.76	1682.73
2	1802.81	1310.37
3	1914.98	1359.05
4	1846.07	2225.17
5	1840.12	1737.29
6	2318.12	1822.96
7	1887.15	1257.07
8	1752.25	1396.46

Table 6.2: *Endurance-trained elderly people*

Subject no.	I_{LF}	I_{HF}
1	1904.30	2288.59
2	1876.98	2979.65
3	1581.83	1746.84
4	2065.45	1579.73
5	1613.01	1873.36
6	1725.62	1265.36
7	1169.92	1995.74
8	2364.24	2385.51

Table 6.3: *Healthy sedentary young people*

Our model simulations show that heart rate variability in healthy sedentary elderly people is, on average, smaller than that of endurance trained elderly people, which is in turn smaller than that of healthy sedentary young people. We can also observe that the values of heart rate variability are higher across the mid-frequency range (0.05 - 0.15 Hz) than the values observed over the high frequency range (0.15 - 0.35 Hz) for all subject groups.

The variations in values which can be seen between the different individuals across all three subject groups can be attributed to the random gaussian noise which is sampled with each beat in the simulation.

We can conclude that heart rate variability values are higher in healthy sedentary young people than in the other two subject groups, and endurance trained elderly people present slightly higher heart-rate variability values, across both frequency ranges, when compared to healthy sedentary elderly people.

Figure 6.1 presents the computed baroreflex sensitivity values for the subject groups modelled in our simulations. The healthy sedentary elderly group presents values for 26 individuals, whereas the other two groups present values for 8 individuals, as per the data we are evaluating our model against. It can be seen from the results obtained that the difference in baroreflex sensitivity between the two groups of elderly people is relatively small, with only a very slight increase, which could also be attributed to the random noise input to the model. Furthermore, from the figure we can spot an increase in baroreflex sensitivity values in healthy sedentary young people, as well as a higher standard deviation, which shows an increase in heart rate variability in young people.

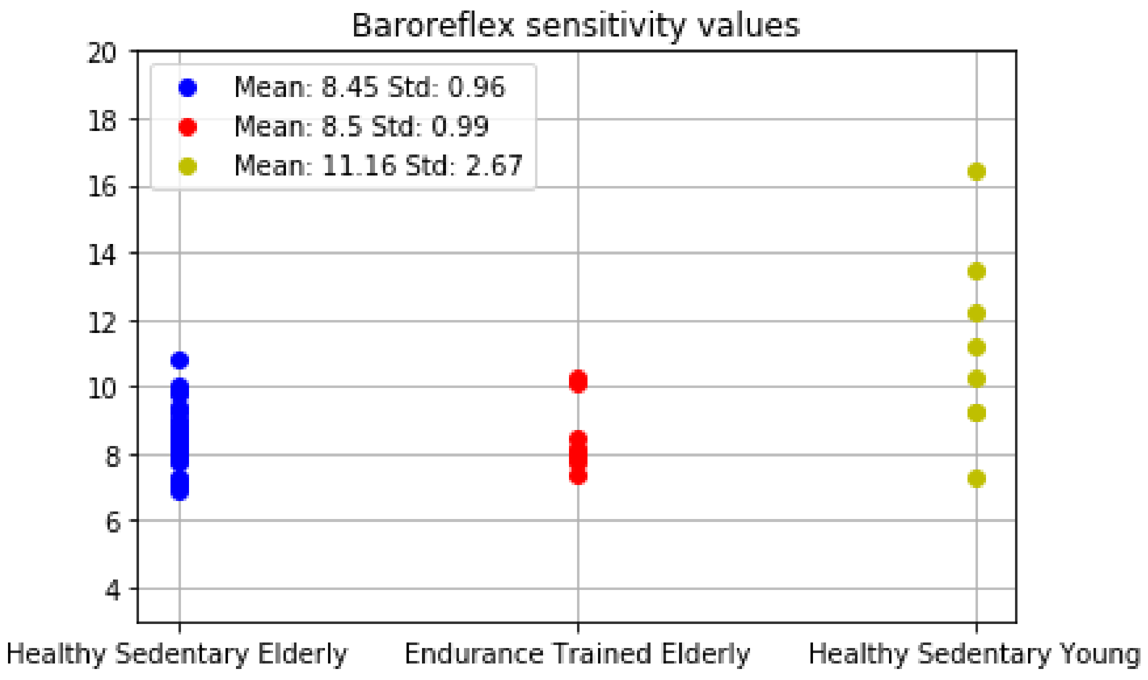


Figure 6.1: *Baroreflex sensitivity values in the subject groups examined*

6.3 Discussion

In the simulations run with our model, we set as input for each of the different subject groups, values extracted from the paper we chose as our reference point. However, the paper only provided mean values for two of the four features we use as input, specifically systolic pressure and heart rate variability. In order to be able to run the simulations, we have made the decision of using normal human average values for the two features that could not be gathered from the papers. This has definitely had an influence on our output results. As expected, the two elderly people groups, where the average values for diastolic pressure and heart rate variability are not expected to be too far from the human average values, performed well, generating good results, very close to real data.

However, in healthy young people, where the heart rate variability is usually significantly higher than in elderly people, we did not see this replicated in our output, and, despite showing an increase from the other two subject groups, it is clear that the difference is not as significant as expected.

Furthermore, another point of discussion would be the extraction of data. We believe that with access to resources, we could have conducted our own study into the three different subject groups, gathering exactly the data needed, and increasing the accuracy of our model significantly.

Finally, through running our simulations and changing parameters of the model, we have been able to observe that changes in the value of ratio of pulse pressure to preceding heart rate variability interval did not have an impact on the output data, unless the changes made were of significant magnitude.

6.4 Goals Achieved

In order to evaluate whether our findings and the project results support the original objectives laid out for the project we have decided to inspect the requirements defined in Chapter 3 ”Requirements and Analysis” and see if they were all met.

Firstly, we have *achieved* our goal of conducting research among existing papers and finding one with data that could be extracted for our model, and results to be used in evaluating our model’s accuracy.

Secondly, we have managed to *successfully implement* a beat-to-beat heart rate variability model, with mean values for systolic pressure, diastolic pressure, heart rate variability interval and arterial time constant as inputs. The model created is flexible and can accommodate any values for these inputs.

Time domain values for heart rate features *have been generated* and the data *has been visualized* with the help of plots for systolic and diastolic pressure and heart rate variability interval.

Furthermore, our goal of computing baroreflex sensitivity and heart rate variability at mid-frequency and high frequency for each of the individuals in the different subject groups has been *achieved.* We have then compared and evaluated our results against those from the paper chosen as point of comparison, concluding that our model can closely replicate values for healthy sedentary elderly people and healthy endurance trained elderly people, but presents some inaccuracies when reporting values for healthy sedentary young people.

By following a test driven development approach, and having the intention of keeping our functions pure, with many abstractions, we believe that the goal of writing code that is modular, maintainable and reusable has been *achieved.*

Finally, we have *not managed to achieve* our goal of altering model parameters to improve accuracy of results. This was due to time-constraints, and it represents an area of further work that could be done to improve the heart rate variability model developed for this dissertation project.

6.5 Further work

There is a lot of potential for building on the work done for this dissertation project. As stated in the beginning of this paper, currently heart rate variability analysis is being used by consumers all over the world through the widespread use of mobile apps tracking an individual's sleep, athletic performance or general well-being. Smart wearable devices have not been around for long, and the future is exciting for this technology. Smart glasses already exist, and we believe that these could incorporate heart rate variability analysis to vital details to the user, that would enhance their experience when doing physical related activities, offering feedback on their performance during the workouts.(Wareable (2015))

Furthermore, the heart rate variability model developed for this project, can provide groundwork for a heart rate variability based machine learning model that could help in the health industry. Spotting similarities between heart rate variability in patients with different diseases through the use of this model may provide valuable data to train a machine-learning model for early predictions in patients. Early identification of diseases is extremely valuable, as it allows for more time of treatment, as well as not allowing the disease to reach severe stages. Work in this area has already been done, as can be seen from the ' Heart Rate Variability based machine learning models for risk prediction of suspected sepsis patients in the emergency department' paper. The conclusion of this research paper is that a model using heart rate variability analysis can be used to improve prediction and help with early identification of suspected patients.

In addition to future work that could be done building on the model we have developed, further work can also be done to improve the accuracy of the model.

We believe that further research can be done on the generation of the random gaussian noise. We have generated random gaussian noise samples with standard deviations of 25ms for the heart rate variability interval and 2 mm Hg for pulse pressure and these values were taken from deBoer's initial model, which had proven quite successful, but we believe there is room for improvement in this area. Moreover, we believe that more insights into heart rate variability could be obtained if we would look at three different frequency ranges, rather than the two we have used for this model. For our results, we have not taken into consideration the frequency range from 0Hz to 0.05Hz, but the computed power spectra for the signals has shown that the signal has smaller peaks in that area that could provide further insight. Chiew et al. (2019)

6.6 Summary

This chapter has presented the findings of the work carried out during this dissertation project in a table format along with explanations of what the values inside the tables represented. We have also presented a discussion around off-topic findings, and justification for the differences in values of the simulated data against the real data for one of the subject groups.

The degree to which the initial requirements were met was also presented, stating requirements that were fully achieved and also the one requirement that was not. For the unmet requirement, we have given reasoning as to why we were not able to meet it.

Finally, this chapter presented further work that can be done on this project. There was a discussion around future work that could be done building on the model developed, using it as a tool in other projects, referencing an academic paper that used machine learning and heart rate variability analysis, as well as a discussion around further work that could improve the current model, increasing its accuracy.

Chapter 7

Conclusion

Developing this project has been a challenging but rewarding experience, made significantly harder by the unprecedented times the world is facing right now.

The aim of this project was to implement a model of heart rate and blood pressure variability, as described in deBoer's paper, where current values of heart rate features depend on feature values from previous beats and use the model to investigate whether recording of heart rate variability in normal subjects, and subjects from different age groups and with different athletic performance levels could be reproduced.

Despite the challenges faced, the project's goal has been achieved, and, as described in earlier chapters, the model implemented can reproduce heart rate variability in normal subjects, as well as subjects from three different groups: healthy sedentary elderly, endurance trained elderly and healthy young people.

The project has been developed in Python, making use of some of its many open-source libraries, such as numpy, scipy and matplotlib. Furthermore, during the development process, a Test Driven Development approach was taken, which means that the code written for the project is modular, maintainable and readable, allowing for future work to be done integrating this model, or even making improvements to it.

Some of the proposed ideas for future work on the project represented using the model as groundwork for a machine learning model that could provide early detection of patients with different types of diseases. Patterns can be spotted from heart rate variability analysis in different patients through the use of our model, and then this information could be used in developing a machine learning model capable of early identification of infected patients, helping stop a disease from its early stages.

An initial step in validating our model during the development process was to compare time domain values for systolic pressure, diastolic pressure and heart rate variability, with the values observed in deBoer's paper. This comparison was done by generating the time domain plots for these features, and has proved to be successful, with our model closely replicating the data from the paper and allowing us to continue with the model's implementation.

In order to assess the accuracy of our model, we have looked at the power spectrum of the heart rate variability interval in the subjects being modelled, analysing two frequency ranges, mid-frequency(0.05 - 0.15 Hz) and high-frequency (0.15 - 0.35 Hz) and computing mean values over these two frequencies. Furthermore, we have compared values obtained from the computation of baroreflex sensitivity for the subjects, to values gathered from real data. This comparison has shown us that, although the model is successful in reproducing trends in the data, and also reproduces values for the elderly group of people, both endurance trained and sedentary, there is still room for improvement in reproducing heart rate variability in healthy young people. One of the findings from this comparison was that, in order to increase the accuracy, data for the mean diastolic pressure and mean arterial time constant would be needed, specific to each subject group being modelled. Currently, as we were not able to gather mean values for these features, we are using in our model normal human averages, which in turn, is affecting our output results, making the model unable to accurately reproduce all the differences between different subject groups.

To conclude, the heart rate and blood pressure variability model implemented for this project can accurately reproduce recordings from normal subjects, as well as subjects from different groups, varying in age and athletic performance levels.

There is room for further work that could be done in order to increase accuracy in the outputs for younger subjects, which may include gathering of more data to be input to the model, or further changing of model parameters, such as the initial respiration value or the ratio of pulse pressure to preceding heart rate variability interval, however the results are very promising and this model, with slight improvements, could be used as groundwork for a machine learning model trained in early detection of patients infected with diseases, in order to prevent more severe cases.

Bibliography

Ahmad, S., Tejuja, A., Newman, K. D., Zarychanski, R. & Seely, A. J. (2009), 'Clinical review: a review and analysis of heart rate variability and the diagnosis and prognosis of infection', *Critical Care* **13**(6), 232.
URL: *https://ccforum.biomedcentral.com/articles/10.1186/cc8132*

Akselrod, S., Gordon, D., Madwed, J. B., Snidman, N. C., Shannon, D. C. & Cohen, R. J. (1985), 'Hemodynamic regulation: investigation by spectral analysis', *American Journal of Physiology-Heart and Circulatory Physiology* **249**(4), H867–H875.

Bowman, A., Clayton, R., Murray, A., Reed, J., Subhan, M. & Ford, G. (1997*a*), 'Effects of aerobic exercise training and yoga on the baroreflex in healthy elderly persons', *European journal of clinical investigation* **27**(5), 443–449.
URL: *https://onlinelibrary.wiley.com/doi/abs/10.1046/j.1365-2362.1997.1340681.x*

BOWMAN, A. J., CLAYTON, R. H., MURRAY, A., REED, J. W., SUBHAN, M. F. & FORD, G. A. (1997*b*), 'Baroreflex function in sedentary and endurance-trained elderly people', *Age and ageing* **26**(4), 289–294.
URL: *https://academic.oup.com/ageing/article/26/4/289/35983*

Campos, M. (2017), 'Heart rate variability: A new way to track well-being', *Cambridge: Harvard Health Blog, Harvard Medical School* .

Cardiovascular Diseases - World-Health-Organization (n.d.).

Chiew, C. J., Liu, N., Tagami, T., Wong, T. H., Koh, Z. X. & Ong, M. E. (2019), 'Heart rate variability based machine learning models for risk prediction of suspected sepsis patients in the emergency department', *Medicine* **98**(6).
URL: *https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6380871/*

De Boer, R., Karemaker, J. & Strackee, J. (1985), 'Relationships between short-term blood-pressure fluctuations and heart-rate variability in resting subjects i: a spectral analysis approach', *Medical and biological engineering and computing* **23**(4), 352–358.
URL: *https://link.springer.com/article/10.1007/BF02441589*

DeBoer, R., Karemaker, J. M. & Strackee, J. (1987), 'Hemodynamic fluctuations and baroreflex sensitivity in humans: a beat-to-beat model', *American Journal of Physiology-Heart and Circulatory Physiology* **253**(3), H680–H689.
URL: *https://journals.physiology.org/doi/abs/10.1152/ajpheart.1987.253.3.H680*

Goldenberg, I., Goldkorn, R., Shlomo, N., Einhorn, M., Levitan, J., Kuperstein, R., Klempfner, R. & Johnson, B. (2019), 'Heart rate variability for risk assessment of myocardial ischemia in patients without known coronary artery disease: The hrv-detect (heart rate variability for the detection of myocardial ischemia) study', *Journal of the American Heart Association* **8**(24), e014540.
URL: *https://www.ahajournals.org/doi/10.1161/JAHA.119.014540*

Heart Statistics (2020).
URL: *https://www.bhf.org.uk/what-we-do/our-research/heart-statistics*

Hillebrand, S., Gast, K. B., de Mutsert, R., Swenne, C. A., Jukema, J. W., Middeldorp, S., Rosendaal, F. R. & Dekkers, O. M. (2013), 'Heart rate variability and first cardiovascular event in populations without known cardiovascular disease: meta-analysis and dose–response meta-regression', *Europace* **15**(5), 742–749.
URL: *https://academic.oup.com/europace/article/15/5/742/673395*

Matplotlib: Visualization with Python (2020).
URL: *https://matplotlib.org/*

NumPy (2020).
URL: *https://numpy.org/*

Pacheco, D. (2016), *Building Applications with Scala*, Packt Publishing Ltd.
URL: *http://sd.blackball.lv/library/Building$_A$pplications$_w$ith$_S$cala$_($2016).pdf*

Regulation of Blood Pressure: Short Term Regulation Baroreceptors (2020).
URL: *https://study.com/academy/lesson/regulation-of-blood-pressure-short-term-regulation-baroreceptors.html*

Scientific computing tools for Python (2020).
URL: *https://www.scipy.org/about.html*

Seely, A. J. & Macklem, P. T. (2004), 'Complex systems and the technology of variability analysis', *Critical care* **8**(6), R367.
URL: *https://pubmed.ncbi.nlm.nih.gov/15566580/*

Software-Testing-Fundamentals (2020), 'Functional testing'.
URL: *http://softwaretestingfundamentals.com/functional-testing/*

SPYDER (2020).
URL: *https://www.spyder-ide.org/?fbclid=IwAR03jD8TmEQrpnCrjeBmPAHLb2dnjC4N7iCuChZ4AdnNOK*

Stanford.edu (2011), 'Welch's method'.
URL: *https://ccrma.stanford.edu/ jos/sasp//Welch$_{sM}$ethod.html*

unittest–Unit testing framework (2020).
URL: *https://docs.python.org/3/library/unittest.html*

Wareable (2015), 'A brief history of wearable tech'.
URL: *https://www.wareable.com/wearable-tech/a-brief-history-of-wearables*

Welch, P. (1967), 'The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms', *IEEE Transactions on audio and electroacoustics* **15**(2), 70–73.
URL: *https://ieeexplore.ieee.org/document/1161901*

What is Python? (2020).
URL: *https://www.python.org/doc/essays/blurb/*

Whittam, A. M., Clayton, R. H., Lord, S. W., McComb, J. M. & Murray, A. (2000), 'Heart rate and blood pressure variability in normal subjects compared with data from beat-to-beat models developed from de boer's model of the cardiovascular system', *Physiological measurement* **21**(2), 305.