Empowering Predictivity and Speed of hiPSC CM Assays by Machine Learning Approach

Haibo Liu

SORBONNE UNIVERSITÉ

INRIA

Améliorer la prédictivité et la rapidité des tests hiPSC CM par des approches d'apprentissage automatique

École doctorate Sciences Mathématiques de Paris Centre
Unité de recherche Laboratoire Jacque-Louis Lions

Par **Haibo Liu**

Sous la direction de : **Damiano Lombardi** et **Muriel Boulakia** et **Sylvain Bernasconi** et **Christophe Bleunven**

Soutenue le 02/05/2024

Membres du jury :

Molly Maleckar	Simula Research Laboratory	Rapporteure
Lisl Weynans	Université de Bordeaux	Rapporteure
Frederic Nataf	Sorbonne Université	Examinateur
Laurent Dumas	Laboratoire de Mathématiques de Versailles	Examinateur
Irene Balelli	INRIA d'Université Côte d'Azur	Examinatrice
Nejib Zemzemi	Inria Bordeaux Sud-Ouest	Invité
Damiano Lombardi	Inria, Paris	Directeur de thèse
Muriel Boulakia	Laboratoire de Mathématiques de Versailles	Co-Directrice de thèse
Sylvain Bernasconi	NOTOCORD	Co-encadrant de thèse

SORBONNE UNIVERSITÉ

INRIA

Empowering predictivity and speed of hiPSC CM assays by machine learning approaches

École doctorate Sciences Mathématiques de Paris Centre
Unité de recherche Laboratoire Jacque-Louis Lions

By **Haibo LIU**

Under the direction of: **Damiano Lombardi** and **Muriel Boulakia** and **Sylvain Bernasconi** and **Christophe Bleunven**

Date May 02 2024

Jury members:

Molly Maleckar	Simula Research Laboratory	Referee
Lisl Weynans	Université de Bordeaux	Referee
Frederic Nataf	Sorbonne Université	Examiner
Laurent Dumas	Laboratoire de Mathématiques de Versailles	Examiner
Irene Balelli	INRIA d'Université Côte d'Azur	Examiner
Nejib Zemzemi	Inria Bordeaux Sud-Ouest	Guest
Damiano Lombardi	Inria, Paris	Supervisor
Muriel Boulakia	Laboratoire de Mathématiques de Versailles	Supervisor
Sylvain Bernasconi	NOTOCORD	Supervisor

Inria, COMMEDIA
2 RUE SIMONE IFF
75012 Paris
France
and
Sorbonne Université, Laboratoire Jacques-Louis Lions
4 PLACE JUSSIEU
75005 Paris
France

Keywords: Parameter estimation, Ordinary differential equations, Autoencoder, Artificial neural network, Safety pharmacology

Mots-clés: Estimation des paramètres, Équations différentielles ordinaires, Auto-encodeur, Réseau de neurones artificiel, Pharmacologie de sécurité

Acknowledgements

This PhD project is a part of the INSPIRE European Training Network, which receives funding from the EU Horizon 2020 Research and Innovation Programme under the Marie Skłodowska-Curie GA 858070 [1]. It is an honor to be one of the early stage researchers in the INSPIRE team. I am grateful for the support provided by Pieter-Jan Guns, the scientific coordinator of the INSPIRE European Training Network, and Paz Yanez, the project manager. I am also grateful that NOTOCORD SYSTEMS and Inria gave me this opportunity to join the INSPIRE project.

First of all, I also would like to say thanks to Molly Maleckar and Lisl Weynans for agreeing to be reviewers for my PhD manuscript. Thank you for your thorough reports and insightful comments. They have been invaluable in finalising my thesis. I would also express my appreciation to Professor Frederic Nataf, Professor Laurent Dumas, and Researcher Irene Balelli for accepting to be my examiners and Nejib Zemzemi to be part of the jury for my PhD Defense.

Then, I would like to express my deepest appreciation and gratitude to my academic PhD supervisors, Damiano Lombardi and Muriel Boulakia, for their unwavering support, encouragement, and guidance throughout the journey of my PhD studies.

I would like to express my sincere appreciation to Damiano. Thank you for your patience and kindness in teaching me academic knowledge and helping me understand the material. Whenever I have any questions, you always help me understand not only the problem but also draw inferences applicable to other cases. Because of you, I could gain a lot of knowledge and skills to be creative in my research. These knowledge and skills are not only essential for my PhD project but also for my future career.

I would also like to thank you for supporting and encouraging me to attend several important conferences. I have been able to witness new research and meet other scientists. Additionally, I appreciate your encouragement to present my work at the conferences. The memories of these conferences form an important part of my PhD life, allowing me to enjoy the scientific environment.

I remember all the moments we discussed and solved problems together. You

always shared your knowledge and experience with me and taught me to think and act like a scientist. Your consistent encouragement and recognition of my work are truly appreciated. Thank you for giving me the opportunity to work with you and helping me grow in science. I am grateful for the significant amount of time and effort you dedicated to helping me complete my PhD. I feel fortunate to have had you as my supervisor, and I will carry the knowledge and skills gained throughout my academic and professional life.

I would like to extend my sincere appreciation to Muriel. Thank you for consistently providing invaluable insights, constructive feedback, and unwavering encouragement at every stage of my research. Your experience and expertise have been important in guiding me through my PhD, helping me think through solutions to challenges, and offering valuable suggestions. Your consistent support during the preparation of presentations has been particularly noteworthy, and I am grateful for the guidance that has enhanced my skills in scientific writing. Thank you for always being patient to teach and explain to me. Your instruction has made a significant impact on my academic journey, and I am truly thankful for the time and effort you have dedicated to my growth. I also would like to thank you not only for the academic support but also for the personal care and kindness that have contributed to my PhD life.

I would also like to thank my supervisor from NOTOCORD company. Thank you Sylvain for always recognising my efforts and encouraging me in my PhD. Whenever I have questions you always help and give invaluable advice. I would like to thank you for your time and efforts to support my PhD. I would also like to express my deep gratitude to my supervisor, Christophe, at NOTOCORD. Your exceptional expertise and insightful contributions have been important to my research work. Thank you for consistently providing valuable advice and generously sharing your profound knowledge about the safety pharmacology industry.

I extend my sincere appreciation to Miguel Fernández at Inria for his invaluable support during my PhD, contributing significantly to creating a comfortable working environment. I am grateful for the guidance and encouragement, which were instrumental in the success of my PhD.

I would also like to express my heartfelt thanks to all members of the COMMEDIA team, including other supervisors and my colleagues, Sara, Oscar, Fabien, Marguerite, Sebastien, Daniele, and Marea... Your kindness, care, and love have made a profound impact on my PhD experience. Each member of the team played a crucial role in a positive and collaborative atmosphere, contributing to the overall success and enjoyment of my doctoral life. I will always remember our team time together going to restaurants, Karaoke, and Inria parties, board games, escape games, and bars. Thank you for being

such a lovely team.

I would like to express my gratitude to my colleagues on the 3rd floor of Inria, including Siwar, Van-Thanh, Matthias, Daniel, Edouard, Jean Guillaume, Suraji, Yangfei, Mathieu, Nicola, Kexin... Our lunchtimes were filled with countless joyful moments. I remember our conversations, full of jokes and discussions about movies, music, and culture, as well as the multitude of stories we shared. Additionally, I extend my thanks to my friends at Sorbonne Université, Rui Li, Mingyue Zhang, Liangying Chen, Yipeng Wang, Gong Chen, Chourouk, Agustin, and .... Your companionship has brought me a lot of happiness.

I consider myself fortunate to be a member of the INSPIRE family alongside Elham, Anna, Sara, Benji, Bohdan, Brigitta, Charles, Marieke, Martina, Patriza, Tommaso, Dustin, Callan, and Chris. I express my gratitude for the countless joyful moments shared during our INSPIRE summer school and annual meetings.

Finally, I would like to say thank you to my family, my parent and my brother for always supporting, understanding, and loving me. I want to express my deepest gratitude to my Uncle Daniel for your unwavering support, encouragement, and, above all, love. All your love has been the cornerstone of my strength, providing comfort and inspiration throughout my life. Also, my friends Mandy, Wenping, Ruwen, Zoey, ..., all your love and support are very important to give me wonderful memories in my life. Thank you to all of you for coming into my life and making my life colourful.

Throughout my three years in the PhD, I have been fortunate to experience and enjoy so much. This period has been unforgettable, shaping a significant part of my life. Finally, I would like to extend my heartfelt thanks to all who have crossed paths with me during this journey.

Abstract

This thesis mainly addresses two topics : the first focuses on parameter estimation while the second explores applications relevant to safety pharmacology and disease studies.

Solving parameter estimation problems poses significant challenges, especially in dynamical systems characterised by a large number of equations and parameters. The difficulties arise from the need to solve non-linear, non-convex, and potentially high-dimensional optimisation problems. Classical optimisation methods often rely solely on the underlying dynamical systems and neglect the advantages offered by all available data. In response to this gap, we introduced a novel approach named Latent Variable Gradient Flow (LVGF), designed to leverage both data and the underlying dynamical system. This method is based on two steps. In the first step, an autoencoder is trained on the data set in order to represent it with latent variables in smaller dimension. Then, in the second step, with the help of a non-linear mapping that allows to link the latent variables and the parameter variables, we developed an algorithm that can be described as a gradient flow for the latent variables of the autoencoder. We proved convergence results for our method and the numerical tests highlighted the fact that the LVGF approach could overcome the challenges associated with parameter estimation compared with the classical gradient descent method, particularly in situations involving non-convex optimisation problems.

Regarding the applications in safety pharmacology and disease studies, we investigated several Artificial Neural Networks (ANN) methods for classifying drugs based on their effects on ion channels. Firstly, we focused on classifying whether a given data from a drug experiment alters the normal behaviour of the human ether-a-go-go-related gene (hERG) channel. The Multilayer Perception (MLP) and multivariate 1-dimensional Convolutional Neural Network (1D-CNN) demonstrated efficiency and high accuracy in drug classification, showcasing their potential to enhance drug high-throughput screening. Furthermore, we extended the application of MLP and multivariate 1D-CNN to identify healthy individuals from patients with Brugada syndrome. This testing confirmed their versatility in addressing different problems. Additionally, we explored the use of autoencoder methods in anomaly detection to automatically identify abnormal data from experimental data sets. This approach aims to enhance the quality of data during the experimental recording stage. Lastly, we presented a comparative analysis of ANN, statistical, and mathematical modelling methods employed in *in vivo* studies to examine the ageing effects on dogs' cardiovascular systems.

Résumé

Cette thèse aborde principalement deux thèmes : le premier se concentre sur l'estimation des paramètres tandis que le second explore des applications pertinentes pour la pharmacologie de la sécurité et les études sur les maladies.

La résolution de problèmes d'estimation des paramètres présente des défis significatifs, notamment dans les systèmes dynamiques caractérisés par un grand nombre d'équations et de paramètres. Les difficultés découlent de la nécessité de résoudre des problèmes d'optimisation non linéaires, non convexes et potentiellement de grande dimension. Les méthodes d'optimisation traditionnelles s'appuient souvent uniquement sur les systèmes dynamiques sous-jacents et négligent les avantages offerts par l'ensemble des données disponibles. En réponse à cette lacune, nous avons introduit une nouvelle approche appelée Latent Variable Gradient Flow (LVGF), conçue pour tirer parti à la fois des données et du système dynamique sous-jacent. Cette méthode repose sur deux étapes. Dans un premier temps, un auto-encodeur est entraîné sur l'ensemble de données afin de le représenter avec des variables latentes en plus petite dimension. Ensuite, dans la deuxième étape, à l'aide d'une application non linéaire qui relie les variables latentes et les paramètres, nous avons développé un algorithme qui peut être décrit comme une méthode de flot gradient pour les variables latentes de l'auto-encodeur. Nous avons prouvé des résultats de convergence pour cette méthode et les tests numériques ont mis en évidence le fait que l'approche LVGF pouvait surmonter les défis associés à l'estimation des paramètres par rapport à la méthode classique de descente de gradient, en particulier dans les situations impliquant des problèmes d'optimisation non convexes.

Pour ce qui est des applications en pharmacologie de la sécurité et les études sur les maladies, nous avons examiné plusieurs méthodes d'Artificial Neural Networks (ANN) pour classer les médicaments en fonction de leurs effets sur les canaux ioniques. Tout d'abord, nous nous sommes concentrés sur la classification pour déterminer si des données provenant d'une expérience sur un médicament altèrent le comportement normal du canal human ether-a-go-go-related gene (hERG). Le Multilayer Perception (MLP) et le multivariate 1-dimensional Convolutional Neural Network (1D-CNN) ont démontré une efficacité et une précision élevées dans la classification des médicaments, illustrant leur potentiel pour améliorer le criblage à haut débit des médicaments. De plus, nous avons étendu l'application du MLP et du 1D-CNN multivarié pour identifier les individus en bonne santé parmi les patients atteints du syndrome de Brugada. Ces tests ont confirmé leur polyvalence pour résoudre différents problèmes. De plus, nous avons exploré l'utilisation de méthodes d'autoencodeurs dans la détection d'anomalies pour identifier automatiquement des données anormales dans des ensembles de données expérimentales. Cette approche vise à améliorer la qualité des données lors de l'enregistrement expérimental. Enfin, nous avons présenté une analyse comparative des méthodes d'ANN, statistiques et de modélisation mathématique utilisées dans des études *in vivo* pour examiner les effets du vieillissement sur le système cardiovasculaire des chiens.

Contents

Chapter 1

Introduction

In the first part of the introduction, we formulate the problem we are interested in by presenting the general background, the basic mechanics of the electrical activity of the cardiomyocytes and the methodological questions we will focus on. Then, a brief presentation of the state-of-the-art is made on the main themes studied in this thesis: in Section 1.2 on neural network methods, in Section 1.3 on parameter estimation and in Section 1.4 on classification methods relevant in pharmaceutical studies. At last, Sections 1.5 and 1.6 present the main contributions of the thesis, first on parameter estimation and second on classification.

1.1 Motivation and statement of the problems

1.1.1 General background of the study

Drug development is a long and expensive process, marked by a high rate of failure in contrast to the relatively few successes. The aim of this process is to advance novel drug candidates that offer maximum benefits while posing minimal or manageable safety risks. The safety assessment phase of drug development focuses on identifying potential safety issues to safeguard patients from unintended harm. According to the Pharmaceutical Research and Manufacturers of America's 2016 Biopharmaceutical Research Industry Profile [2], the journey to develop a new drug spans 10 to 15 years, with an average cost of $2.6 billion from the 2000s to the early 2010s. Fewer than 12% of the drugs that enter clinical trials have finally been approved. Thus, detailed analyses related to clinical safety concerns and nonclinical toxicity issues that contribute to attrition during drug discovery and development become important.

Among those drug safety issues, unexpected cardiac adverse effects are the leading causes of drug attrition, discontinuation of clinical trials and withdrawal of drugs from the market ([3, 4, 1]). As a result, cardiotoxicity assessment in drug development has gained importance for many years.

Cardiotoxicity assessments for humans are usually done using *in vivo* animal models and *in vitro* non-human tissues. For animal models, there is a lack of cross-species translation due to different biological pathways and pharmacokinetic properties. Conventional *in vitro* testing is also limited because it is low throughput, expensive and time-consuming [4]. Similarly, disease modelling is usually done on animal models or human cells to study the pathogenesis of genetic cardiac diseases. The limitations of

cross-species translation in animal models and the limited source of human cells also create challenges in disease study [5].

Therefore, considering limitations exist in cardiotoxicity assessments, it is important to develop high-throughput screening with thousands of compounds to select the best candidates as fast as possible and reduce drug-induced safety concerns in drug development. Considering disease studying, it is essential to explore alternative approaches, moving beyond reliance on animal models and human cells. One such alternative involves conducting experiments on drug safety and disease studies using human induced pluripotent stem cell-derived cardiomyocytes (hiPSC-CMs) [6]. A human induced pluripotent stem cell (hiPSC) is a reprogrammed cell with the capacity for self-renewal and the capability to be differentiated in any cell type, in which cells close to cardiomyocytes, hiPSC-CM.

Compared with using differentiated human cells and animal experiments, the unlimited source of hiPSC-CMs is one the most promising innovations of medical research that can overcome the limitations regarding effectiveness and efficiency. HiPSC-CMs can provide new opportunities to create *in vitro* models which could be used in regenerative medicine, disease modelling, drug screening, and precision medicine.

Cardiomyocytes are excitable cells, with a well characterised electrical activity. Thus, recording and analysis of this activity allows to screen for all drugs interfering with it, which are one of the main attrition causes in cardiac safety. Patch clamp [7] is a technology used to study the electrical properties of a single cell at the level of ion channels and membrane currents. During patch clamp recording, a microelectrode is put inside the cell membrane. The microelectrode creates a tight seal with the cell membrane to measure the electrical activity of the ion channels. A typical patch clamp recording captures the intracellular action potential (AP). The AP results from rapid changes in the voltage across a membrane. Patch clamping is considered the "gold standard" for ion channel research, but it is low-throughput and has high requirements of expertise for experimenters [8]. Therefore, new technologies like Multi-Electrode Array (MEA) have been proposed for high-throughput studies which is more automated compared with patch clamping.

An MEA is a grid of closely spaced microscopic electrodes embedded at the bottom of wells in a multi-well MEA plate [9]. This technology allows for the cultivation of electrically active cells like cardiomyocytes directly over the electrodes. HiPSC-CMs based *in vitro* model, when observed with MEA can overcome the challenges presented above and allow the high-throughput screening [10]. Indeed, MEA makes it possible to record the electrical activity (the electrograms obtained are called field potential (FP)) of multiple cells (usually a monolayer, a small tissue) to study a large number of drugs in parallel. The FP of hiPSC-CMs is the extracellular electrical activity from a population of these cells (a small tissue). It is a measure of the overall electrical activity of the hiPSC-CMs. In the following section, we will give more detail of the mechanisms related to the electrical activity of cardiomyocytes.

The extracellular FP waveforms and intracellular action potential are long time series data. One cardiac beat has two parts: depolarisation and repolarisation phases. The depolarisation phase may only last 5 to 10 milliseconds (ms). To be able to represent depolarisation waveforms, the recording device must have a high sampling rate. For instance, MEA often uses a sampling rate of 20 kilohertz (kHz, a unit of frequency is equivalent to one cycle per second), which means, it records 20000 values of electrical potential per second. When we assume there is only one electrode and one well, having 30

minutes of recording would take 288 Megabytes of memory in double precision. However, MEA can have 24, 48, 96, and more wells and multiple electrodes per well. Due to the large amount of data to be analysed, the large amount of variability (caused by sensitivity to experimental conditions and biological phenomena), and the measurement noise, exploiting MEA recordings is a challenging task. The commonly used approach is to compute key biomarkers which are values that can quantify a larger electrical activity. This approach may not be reliable enough because the selected biomarkers may give indirect and limited information on specific ion channel activities and other valuable information can be ignored.

In this work, we will investigate computational methods that can efficiently and effectively analyse data from hiPSC-CMs assays to address problems in drug safety assessment and disease studies. To do so, it is necessary to understand the mechanisms of perturbation caused by a drug or disease on cardiomyocytes. Another challenge lies in managing big data recorded from hiPSC-CMs assays to perform high-throughput screening and study of cardiac diseases.

1.1.2 Electrophysiology

In this section, we present very briefly the basic mechanisms governing the electrical activities of the cardiomyocytes and the mathematical models which describe them.

The generation of electrical activity, such as the intracellular action potential [11] of cardiomyocytes, results from the selective permeability of ion channels (which are pores that allow specific charged particles to cross the membrane) distributed on the cell membranes. The cell membrane consists of a phospholipid bilayer, allowing only substances capable of diffusing directly through the hydrophobic core to pass unaided. Charged particles, being hydrophilic, cannot traverse the cell membrane without assistance. This assistance is provided by transmembrane proteins, particularly channel proteins. The involvement of various channels and specialised energy-dependent "ion pumps", for instance, the carrier protein known as the sodium-potassium pump, is responsible for moving sodium ions out of the cell and potassium ions into the cell. This process helps regulate ion concentrations on both sides of the cell membrane. Therefore, the permeability of ion channels enables a specific ion to move passively down its electrochemical gradient, thereby altering the membrane potential of the cardiomyocyte. The ion motion and the different concentrations of ion species on both sides of the cellular membrane determine a voltage difference across it, called transmembrane potential. Then, a dynamic and transient alteration of this potential occurs in response to a stimulus, called an action potential, allowing for cell communication and signalling. This provides several information about the physiological state and function of the cardiomyocyte.

Referring to [11] and Figure 1.1, one cycle of AP begins with a rapid transient influx of sodium ions into the cell, leading to a rapid depolarisation (a rapid increase in membrane potential). This phase is triggered by the opening of voltage-gated sodium channels. Briefly following the depolarisation, rapid inactivation of the sodium channels, along with the transient opening and closing of potassium channels contributes to a partial repolarisation (the membrane potential becomes slightly decreased). Then, a balance between the inward flow of calcium ions and the outward flow of potassium ions can lead to a plateau phase that prolongs the AP, contributing to the sustained contraction of the cardiac muscle. Finally, calcium channels close and the potassium channels remain

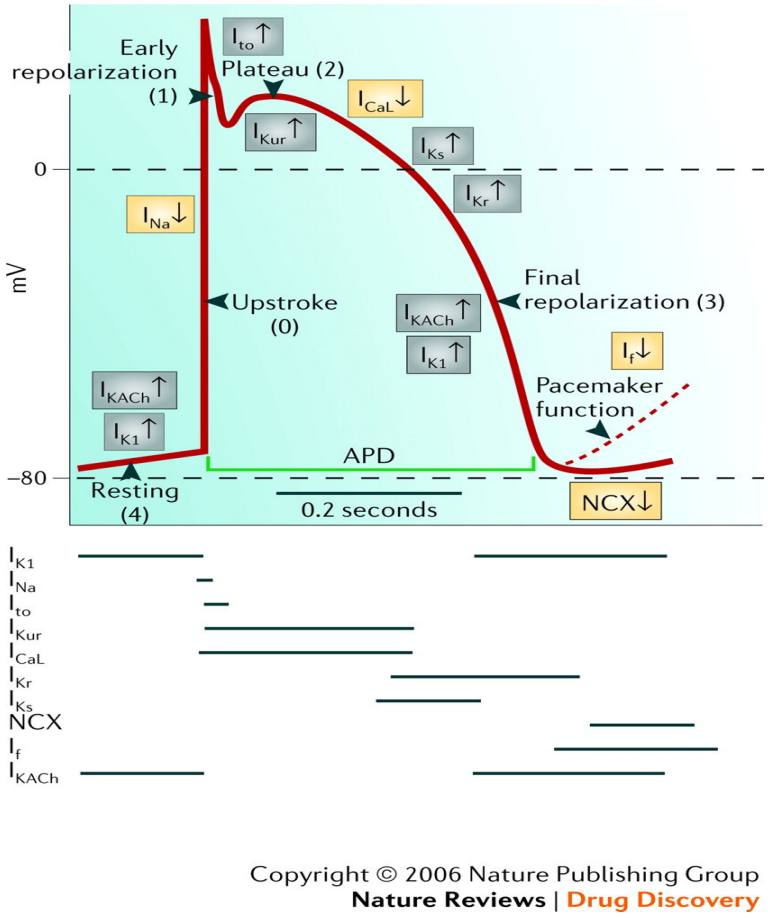


Figure 1.1: Membrane currents that generate a normal action potential [11]

open, the efflux of potassium ions ensures an outward positive current results in the repolarisation of the membrane potential, and then the cell is ready for the next cycle.

Among all the channels, in drug safety assessment, we will focus on detecting the impact on the human ether-a-go-go-related gene (hERG) channel. hERG encodes the pore-forming subunit of the rapid component of the delayed rectifier potassium channel (IKr) [12]. This channel is important in the repolarisation of cardiomyocytes. Inhibition of the hERG channel can prolong the action potential duration (would have the same effects on FP). This would potentially cause ventricular arrhythmia, torsades de pointes and sudden death. Therefore, many drug safety assessments focus on hERG channel studies.

To get an insight into the different mechanisms of cardiac electrophysiology, we need mathematical and computational models that can simulate the movement of sodium, potassium, calcium, and other ions across semipermeable cellular and intracellular membranes.

There are two types of models: phenomenological and physiological models. The phenomenological models aim to describe and simulate the electrical activity of the cardiomyocyte without necessarily considering the detailed biophysical processes occurring at the cellular or molecular level. In contrast, the physiological (biophysical) models describe and simulate the detailed biophysical phenomena of the cardiomyocyte. In this work, as we are focused on understanding the effects that drugs or pathologies might have on ion channels, our primary interest lies in physiological models.

The description of the electrophysiological activity by mathematical models started from the foundational model introduced by Hodgkin and Huxley in 1952 [13]. Since then, mathematical models have been important in enhancing our understanding of the physiological processes within cardiac cells. These models have evolved to cover a wide array of cell types found in the heart, including nodal, atrial, ventricular, and Purkinje cells, as documented in studies such as those by [14, 15, 16, 17, 18, 19, 20].

Moreover, specialised computational models have been developed to describe various cell types of hiPSC-CMs, including ventricular-and atrial-like AP of hiPSC-CMs in [21] and [22], and FP behaviours, as investigated in studies by [23] and [24]. These models serve as valuable tools for comprehending the electrophysiological characteristics of hiPSC-CMs and their role in cardiac function in drug safety assessments.

Those mathematical models allow scientists and researchers to conduct virtual experiments at a fraction of the cost which is much faster than performing experiments in the lab, making it an economical choice, especially when dealing with hazardous materials or complex systems. This can accelerate research and development in various fields, such as drug discovery and disease studies.

In this work, we focus on how to use computational models of cardiac electrophysiology that can help us improve the interpretation of drug effects on hiPSC-CMs and mechanisms of cardiac disease. However, since cardiac electrophysiology models often involve a large number of equations and parameters, using them in addition to experimental data can be a challenging task. Therefore, we are inspired to work on methods that can help us to perform this task in a better way.

1.1.3 Methodological questions

The work presented in this thesis mainly focuses on two methodological issues.

(I) **Enabling parameter estimation.** Mathematical models are a tool to investigate electrophysiological mechanisms, fostering the reduction of animal and human testing in drug safety assessment and toxicology studies [25]. The models are systems of parametric Ordinary (or Partial) Differential Equations (ODEs, respectively PDEs). The parameters might encode physical properties or, in the case of phenomenological closures, account for several mechanisms, not described in detail and acting at different scales. When considering hiPSC-CMs, the Paci model, presented in [21], consists of a system of ODEs well describing the AP and the main ionic currents of these cells. Several mathematical models, such as the one in [24], can describe, by means of a system of PDEs, the electrical state inside an MEA well (and, hence, well reproducing FP signals).

Numerical simulations of the solutions for different values of the parameters, sometimes referred to as *in silico experiments*, can be used to get some insight into the variability of the solutions within a population of individuals or experiments. Besides the direct problems, in a number of realistic applications, the goal consists of (or can be reached by) estimating the values of the parameters given some measurements of the system state. It is indeed the case when studying how a drug or a pathology disrupts the normal functioning of cardiomyocytes. In this case, a common situation is the following. We have a mathematical model describing the phenomena under investigation. Moreover, we usually have a database of existing experiments, sometimes with partial knowledge. The goal is to estimate the model parameters given a measurement of AP or FP, in such a way that the observation of the model solution matches the experimental data. This task is generally difficult because models that describe cardiac electrophysiology are often nonlinear and consist of a large number of equations and parameters. These difficulties motivated us to focus on the topic of parameter estimation in dynamical systems. We have investigated the idea of combining knowledge from available data and the mathematical model to enable parameter estimation.

(II) **Machine learning methods for classification in electrophysiology.** The second objective is to explore computational methods that can help to analyse data obtained from drug safety assessment and disease studies. We will investigate some Machine Learning (ML) methods, especially Artificial Neural Networks (ANN) methods to help study several questions:

- Can we automatically classify, given an FP recording, if the tested drug is an inhibitor of one or more ion channels of the cell?
- To improve the quality of the data that will be considered in the analysis, can we automatically detect some abnormal recordings from a large data set?
- Considering a specific disease, such as Brugada syndrome, and given some Patch Clamp data of healthy and patient derived hiPSC-CMs, can we classify healthy individuals from patients? Moreover, can we classify patients according to the severity of the syndrome?

All these questions are crucial in view of setting up reliable methods for high-throughput screening technologies. Question 1 and 3 can be viewed as classification problems. We will investigate the capability of different kinds of ANN to solve classification problems in which the input is a set of long time series. Question 2 is related to anomaly detection. For this problem, we will try an ANN method called autoencoder to perform the anomaly detection in a fast automatic way, requiring a minimum amount of pre-processing time.

In the three next sections, we will present some state-of-the-art of methods that will be used to answer the mentioned questions and solve parameter estimation problems.

1.2 Neural Network methods: state-of-the-art

In this section, rather than making a general presentation of the Neural Network methods, we will focus on three families of methods that we will consider in our work: Multi-layer Perception (MLP), Convolutional Neural Network (CNN) and autoencoders. The first two methods will be repeatedly used to address classification problems regarding the effects of drugs and diseases. In addition, autoencoder methods are dimension reduction methods that will play a key role for our parameter estimation method.

1.2.1 MLP

The idea of Perceptron was introduced by Frank Rosenblatt in 1958 [26]. It is a simplified neural network model that consists of an input layer, one hidden layer and an output layer to learn and perform binary classification tasks. Since the deep-learning feedforward network was introduced in 1967 [27] and the backpropagation algorithm developed in 1970 [28], MLP has been widely used in classification, regression, and pattern recognition.

MLP is a type of feedforward ANN that consists of multiple layers, including an input layer, one or more hidden layers, and an output layer. The input layer consists of neurons that represent the input data denoted as $x \in \mathbb{R}^d$ for $d \in \mathbb{N}^*$. MLP can consist of $L \in \mathbb{N}^*$ hidden layers. For each hidden layer $1 \leq l \leq L$, the l-th layer consists of $k^{(l)} \in \mathbb{N}^*$ hidden units also known as neurons. The hidden layer output is a set of $k^{(1)}$ values given by:

$$o_i = \phi\Big(\sum_{j=1}^{d} \mathfrak{w}_{ij} x_j + \mathfrak{b}_i\Big),\ 1 \leq i \leq k^{(1)}, \tag{1.2.1}$$

where $\mathfrak{w}$ represents the weights, $\mathfrak{b}$ is the bias, and $\phi(.)$ is the activation function. The output of the first hidden layer will be the input passed to the next hidden layer until the output layer. The number of neurons in the output layer will contain the result. Binary classification typically consists of a single neuron and for multi-class classification, it has as many neurons as there are classes. MLP is trained by optimising weights and biases to minimise the loss function which is, for instance, a norm of the error between the predicted output and the actual target values. The backpropagation algorithm computes gradients of the loss with respect to the weights and biases, and an optimisation method, such as the stochastic gradient is used to update them.

MLP has found extensive application in pharmacology studies, particularly in the realm of classification. Here, we will highlight a few examples. The work in [29] constructed a 3-dimensional space matrix containing key features (so called toxicophores) of biological activity and chemical structure. This input is provided to the MLP in order to predict drugs which are causing torsades de pointed arrhythmias. Another example [30] used MLP to classify the hERG channel–drug interaction potential based on values of half-maximal inhibitory concentration (IC50) of drugs which have different ion channel inhibitory. Moreover, the work presented in [31] used chemical, bioactivity and genomic data from the open source ChEMBL database to test the performance of classification of hERG-related cardiotoxicity.

1.2.2 CNN

The modern CNN was proposed by Yann LeCun in 1998 [32]. A CNN comprises an input layer, hidden layers, and an output layer. Within a CNN, the hidden layers consist of one or more layers that execute convolutions. The key operation of the convolutional layer is the convolution operation or cross-correlation operation. This operation involves the convolution kernel, a small matrix of learnable weights with a defined size, performing a dot product with the input matrix of the layer to produce a single scalar value. In the convolutional layer, the kernel will slide over the input matrix with a defined step and compute the dot product within the overlapping regions. Then, scalar biases will be added to the results and the results will be passed to an activation function to produce an output also called a feature map. This output serves as the input for the next layer in the CNN. In CNN architectures, the convolutional layer may be followed by pooling layers, fully connected layers (hidden layers mentioned in MLP), and normalisation layers [33] and [34].

Pooling layers [34] reduce the dimensions of the given layer by aggregating information. For example, max pooling is a pooling operation that selects the maximum element within a filter's region, which is a window with a predefined size, over the input. Thus, the output after the max pooling layer would be a feature map containing the most prominent features of the previous feature map. Normalisation layers are responsible for standardising and normalising the input to ensure that the results are on a similar scale.

CNNs have found many applications in various fields, including medical image classification, face recognition, handwriting analysis, and more. In the domain of safety pharmacology, we can present a couple of examples. For instance, [35] used graph CNN to classify the activity of drugs by inputting compound structures extracted from the ChEMBL database. Furthermore, a study in [36] concerned the safety issue when patients take multiple drugs. They applied deep CNNs to automatically extract information related to Drug-Drug Interactions (DDIs) from the biomedical literature. This information was then used for DDI classification, aiding researchers in gaining a deeper understanding of DDIs by analysing a vast amount of published literature.

1.2.3 Autoencoder

The autoencoder(AE) was first introduced by Rumelhart in 1986 [37]. A more detailed description of the autoencoder is given by [38]. As described in [38], the autoencoder consists of two parts: the encoder and the decoder. The encoder compresses the given

data and encodes them into a representation in the latent space. We denote the encoder function as Ψ. Given an input datum, denoted by $u \in \mathbb{R}^n$, Ψ compresses it to some latent representations $\alpha \in \mathbb{R}^p$:

$$\Psi : \begin{cases} \mathbb{R}^n \longrightarrow \mathbb{R}^p \\ u \longmapsto \alpha := \Psi(u) \end{cases}$$

When $p < n$, the encoder can be interpreted as a feature extractor, which performs a (generally non-linear) dimension reduction. The decoder function restores the data converted by the encoder. We denote the decoder function as $\Psi^\dagger$:

$$\Psi^\dagger : \begin{cases} \mathbb{R}^p \longrightarrow \mathbb{R}^n \\ \alpha \longmapsto \tilde{u} := \Psi^\dagger(\alpha) \end{cases}$$

A complete AE consists at least of three different layers, the input layer receives data $u \in \mathbb{R}^n$, the latent space contains the latent variables $\alpha \in \mathbb{R}^p$, and the output layer gives the prediction $\tilde{u} \in \mathbb{R}^n$. There is the same number of neurons in the input layer and the output layer, there are no constraints on the neuron number in the latent space (usually $p \leq n$). For the classical AE, the computation of each neuron in each layer is the same as for the MLP method, equation (1.2.1). The autoencoder is optimised by minimising a certain error norm between the original data and their reconstruction. More details of the autoencoder will be given in the later sections. Here, we will give some examples of its applications for various purposes.

After the development of the autoencoder, in recent years, there have been many different evolutions and it has been applied to many fields. In particular, it has been applied to the situations which benefit from dimension reduction.

Dealing with high-fidelity simulations of systems characterised by nonlinear PDEs can pose significant challenges and computational expenses. To address these issues, reduced-order modelling (ROM) has gained considerable attention. ROM aims to approximate the original PDE problem by utilising a reduced set of parameters or basis functions that capture the system's behaviour in a lower-dimensional space ([39, 40, 41, 42]). One approach to achieving dimension reduction and uncovering latent features, without requiring access to the full order model (FOM) operators, is through AE [43] and [44]. As demonstrated in [45], AE has been used as a nonlinear ROM method compared with proper orthogonal decomposition (POD) to investigate the capability to solve problems such as the Burgers equation and turbulent channel flow. The results revealed that AE exhibits enhanced reconstruction capabilities for the velocity field. In [46], an approach is proposed that leverages AE with parametric sparse identification of nonlinear dynamics (SINDy). This method involves using a limited number of FOM snapshots to construct a low-dimensional dynamical model aimed at approximating solutions to parametrised PDEs. As mentioned in [46], the AE+SINDy method is not only useful as a ROM but also helpful in dynamical system identification.

Another popular research direction is to use the autoencoder to be a dimension reduction and feature extraction tool in image compression and classification. For instance, the work in [47] combined a stacked autoencoder (SAE) with a 3-dimensional deep residual network (3DDRN) to classify hyperspectral images (HSIs). SAE consists of several layers of sparse autoencoders that use regularisation to enforce sparsity. It was used to reduce the dimensions of original HSIs. Then, residual network modules combined with CNNs were used to perform classification by using, as input, the reduced data [48].

Variational autoencoders (VAE) [49, 50] are a type of autoencoder, whose purpose is enabling the fast generation of new data, which belong to the same population as the ones provided. The encoder takes the input data and maps them to the mean and the standard deviation of a Gaussian distribution. Then, taking samples drawn from the Gaussian distribution as the decoder input, the decoder can generate reconstructed data belonging to the same distribution of the input. VAEs have the capability to generate meaningful new data and find applications in various domains, such as image and text generation. Notably, VAEs can also be employed for generating novel molecular structures in the context of drug development, as discussed in [51].

Denoising autoencoders (DAEs) [52, 53] are neural networks designed to learn the reconstruction of clean data from their noisy version. During the training phase, DAEs are exposed to noisy data, which are created by introducing random noise to the original data set. The primary objective is for the network to learn how to denoise the data and accurately reconstruct the original, noise-free input. DAEs find applications in various fields, including computer vision, health care, and natural language processing. For example, DAEs are employed for tasks like medical image denoising and ECG denoising [54]. They can also be used as a data preprocessing step to handle missing values in the data set by learning to reconstruct the missing information from the available data.

1.3 Parameter estimation: state-of-the-art

Parameter estimation is an inverse problem in which, given a model depending upon a certain number of parameters, we look for the values of the parameters in such a way that the model output matches given measurements. This problem can be mainly addressed in two different ways: deterministic and stochastic. In a deterministic setting, we assume that we can estimate the values of the parameters, whereas, in a stochastic setting (and especially in Bayesian parameter estimation [55]), we try to estimate the conditional probability density distribution of the parameters given the measurements.

Parameter estimation ([56], [57]) is often cast as an optimisation problem. In that case, the parameter estimation is achieved by minimising a cost function, which encodes the discrepancy between the model observations and the actual measurements of the system. In general, this formulation leads to a non-linear, non-convex, possibly high-dimensional optimisation problem. There are two major families of methods to solve these optimisation problems: local optimisation methods and global optimisation methods. We are going to describe them briefly, each time giving the main existing methods and their properties.

First, about local optimisation methods, the most common methods include the gradient-based and Newton-based methods which are iterative algorithms that update parameter estimates based on the gradient, and, for the Newton method, the Hessian, of the cost function. A very common method is the Gauss-Newton method [58], which is commonly used to solve non-linear least squares problems [59]. Another classical example is the class of Quasi-Newton methods [60]. One of the most used is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm [61]. This method approximates the Hessian in each iteration by performing two rank-one updates to the approximate Hessian from the previous iteration.

The local gradient-based optimisation methods might encounter convergence issues when the initial guess is far from the true parameter values [62]. Other methods like

Gauss-Newton or Quasi-Newton may also fail when dealing with non-convex continuous optimisation problems [63].

In the second family of methods, global optimization methods [64] are designed to avoid convergence issues related to the presence of local minimisers. There are many different kinds of stochastic methods for global optimisation, we will highlight a few commonly used classes:

1. Adaptive stochastic methods, also known as adaptive random search, were originally introduced in [65] and [66]. These methods use stochastic (random or probabilistic) processes to adaptively refine the parameter estimation process. They employ sampling optimisation strategies that use results gathered from prior samples or iterations to enhance both the exploration and convergence of the parameter space. A very commonly used example of this method is the Markov Chain Monte Carlo (MCMC) mentioned in [67] and [68]. MCMC methods generate a Markov chain of parameter samples. They adaptively adjust the next proposed sample based on the previous sample's acceptance or rejection. This adaptability allows MCMC to explore the parameter space effectively.

2. Clustering methods, as described in [69], serve the purpose of identifying groups or clusters of parameter estimations that exhibit similar characteristics. Clustering methods can group local optima which increases the efficiency by avoiding the repeated estimations of the same local optimal solutions. However, there are still some difficulties when dealing with high dimensional parameter estimation problems.

3. Evolutionary computation (EC) [70], is particularly useful when dealing with optimisation problems in which the cost function may be non-linear or non-convex continuous optimisation problems or when the search space is high-dimensional. Starting with an initial population of candidate parameter sets, EC evaluates the fitness (how well the model's predictions match the observed data) of each candidate parameter set by applying the model with the associated parameter set. The subsequent step involves selecting the best-performing parameter sets from the current population. New candidates are generated by considering the mean of the selected parameters. To maintain diversity, small random mutations are introduced, and the new population is updated with these fresh candidates. This iterative process continues until a termination criterion is met. It's essential to note that this method demands a substantial number of function evaluations, which can be computationally expensive, especially when dealing with high-dimensional parameter spaces or complex models. There are six types of evolutionary algorithms: genetic algorithms, genetic programming, evolution strategies, evolutionary programming, classifier systems, and hybrid systems. Well-known evolutionary strategy method includes the covariance matrix adaptation evolution strategy (CMAES [71]).

4. Other methods include Ant Colony optimisation [72], Simulated annealing[73], Taboo Search [74], and Particle swarm method [75].

In general, global optimisation methods can be computationally expensive because they often require evaluating a large number of functions. This becomes particularly

problematic when the parameter space is high-dimensional or when the model evaluation is time-consuming.

Considering the above mentioned issues in local and global optimisation approaches, scientists explored other methods that try to solve the problem by benefiting from some available data. Let us assume that, in certain situations, we have examples of pairs of parameters and observations, $\left\{\theta^{(i)}, u^{(i)}\right\}_{1\leq i\leq N_d}$. For example, the work in [76] used a supervised convolutional machine-learning method to learn the physical model parameters. The advantage of this kind of model is that no prior knowledge about the system is required. The main disadvantage consists in the fact that we often need a large number of training pairs parameters-observations to have a reliable estimation.

Recently, many efforts were made in order to try to exploit both the knowledge coming from the mathematical model and from sets of data, such as physics-informed neural networks(PINNs) mentioned in [77, 78, 79, 80] and deep operator network (DeepONet) find in [81] and physics-informed DeepONet (PI-DeepONet) find in [82].

PINNs represent a methodology that not only employs data-driven supervised neural networks to learn the given observation but also incorporates physics equations provided to ensure alignment with the established physics of the dynamical system. PINNs are trained to generate corresponding observations of the dynamic system, taking time as the input. The loss function, subject to minimisation, comprises two components. The first addresses discrepancies between predicted observations and provided data, while the second deals with the residuals of the differential equations, which depend on the parameters. In the study by [83], it is noted that PINNs may encounter what is known as spectral bias, indicating a tendency to prioritise learning the low-frequency component of the solution, potentially neglecting multiple frequency scales. Acknowledging this limitation, [84] introduced Fourier Features-Neural Networks (FF-NN) as a solution to learn dynamical systems and inverse problems.

DeepONet is designed to learn continuous nonlinear operators. The training of a DeepONet typically involves three components: input functions, independent variables (often time in the case of differential equations), and observed outputs. The observed outputs correspond to the results of the operators. In traditional parameter estimation with DeepONet, the neural network aims to learn a mapping from observed data to parameter estimates. This means that, in order to train DeepONet, we still need to provide pairs of parameters and observations, $\left\{\theta^{(i)}, u^{(i)}\right\}_{1\leq i\leq N_d}$.

In PI-DeepONet, the loss function includes a physics-informed term. This term ensures that the estimated parameters follow the known physical laws governing the system. This helps in cases where the differential equations governing the system are well-established. Compared with DeepONet, PI-DeepONet only requires input functions and observed outputs as input. PI-DeepONet can set the unknown parameters θ as trainable parameters in the neural network. It means the pairs of parameters and observations, $\left\{\theta^{(i)}, u^{(i)}\right\}_{1\leq i\leq N_d}$ are not required to provide for the training. Then, the unknown parameters θ can be optimised by the weights and biases of the neural network that minimise the loss function. Similar to PINNs, the loss function contains a data mismatch term and a physics-informed regularisation term. The physics-informed regularisation term is based on the residual of the differential equations. This regularisation term aids in estimating the unknown parameters θ.

The local optimisation and global optimisation methods usually do not consider potential benefits from some observation data. PINN-based and PI-DeepONet methods

introduced an interesting point to leverage both available data and underlining dynamical systems to benefit parameter estimation. Those methods are heavily weighted on the training process. It often requires a large number of iterations and evaluations to converge to an accurate solution. This computational intensity can be a drawback, especially when dealing with dynamic systems consisting of a large number of equations and parameters.

1.4 Classification methods empowering hiPSC-CMs assays: state-of-the-art

The primary focus regarding applications in pharmaceutical and disease studies is to develop computational methods for the analysis of data generated during hiPSC-CMs assays. Drug safety assessments often involve the collection of a large amount of data. Extracting relevant information from them can be challenging from a computational point of view.

In the following, we are going to focus on classification problems, which are a relevant class of problems in pharmaceutical studies. As an example, the process of identifying and grouping drugs according to their specific positive or negative effects can be cast as a classification problem. We will review the contributions and applications of statistical and ML classification methods.

The study in [85] used data on the properties of molecules as input and their corresponding biological reactions as output targets. The authors compared two methods: Support Vector Machine (SVM) [86] and ANN. The results, quite similar for both methods, showed that valuable compound candidates with respect to a specific therapeutic target response can be identified. These classifiers could be used for the virtual screening of millions of molecules.

In the study conducted by [87], chemical toxicity and molecular description data sets from the Leadscope Toxicity Database were employed to compare the predictive performance of Random Forests and deep NN in estimating toxicity levels. The results from both methods demonstrated effective predictions for over 30000 compounds.

Similarly, [88] undertook a comparative analysis of ANN, SVM, and Decision Trees (DT) to classify compounds as either active or inactive within a specific target biological system. The objective was to explore the potential of metabotropic glutamate receptor 5 (mGluR5) compounds as novel treatments for schizophrenia. The study revealed that both ANN and SVM outperformed DT in terms of accuracy, with an area under the curve (AUC) of 0.77 and 0.78 compared to 0.63 for DT.

Furthermore, in the work presented by [89], Bayesian and SVM classifiers were employed to categorize compounds based on their potential to cause drug-induced liver injury, cardiotoxicity, renal toxicity, and genotoxicity. This research aimed to enhance the understanding of the toxicological profiles of various compounds.

In the context of cancer treatment development, evaluating Multidrug Resistance (MDR) is crucial to prevent cancer cells from developing resistance to different drugs. In a study by [90], molecular description data was employed to classify Multidrug Resistance Reversal (MDRR) activity, categorizing compounds into active and inactive classes. The tested naïve Bayes classifier demonstrated a commendable performance, correctly predicting MDRR activities for 82.2% of 185 compounds in a testing set.

From the above studies, ANNs have exhibited significant potential across various

classification methods. In the following, we will explore several applications of ANNs within the biomedical and pharmaceutical domains.

In this thesis, we primarily focus on ANN methods used in drug safety assessment to study cardiotoxicity. There is a similar study in [91], which proposed a multi-labelled neural network combined with an automatic feature extraction to process cardiomyocytes' mechanical beating signals obtained by an interdigital electrode biosensor. Their work classified different drug-induced cardiotoxicities and predicted drug concentrations corresponding to the degree of cardiotoxicity. The classification method provided the possibility to screen the drugs with high-throughput cardiotoxicity assessment.

One notable area of application is in drug discovery. As we mentioned, drug development is a multi-stage, time and resource consuming process due to the number of candidate molecules is large. ANNs have emerged as a valuable tool for enhancing the efficiency of molecular design, pre-designing synthetic routes, predicting protein structures, and identifying macromolecular targets [92]. For instance, the work in [51] proposed a variational autoencoder capable of transforming high-dimensional Simplified Molecular-Input Line-Entry System (SMILES) strings into compact latent space representation vectors. These vectors were subsequently employed in an MLP to estimate target properties associated with each molecule. Moreover, in [93], the authors showed that ANNs like MLP have found utility in forecasting drug release behaviour by using formulation characteristics, such as drug content, pH, or composition as inputs, and providing neurons that represent the dissolution performance of the formulation as outputs. ANNs could correctly predict the *in vitro* response of the drugs.

For instance, [94] used deep learning methods to classify and predict drug-induced liver injury with chemical structure data. In another study [95], the authors used microscopy images of fluorescently labelled nuclei from DAPI-stained cells pre-treated with a set of drugs with differing toxicity mechanisms as input to a deep CNN model. The deep CNN model could obtain abstract nucleus patterns of images and predict toxicity.

ANN has been also used in the study of *in vitro in vivo* relationship (IVIVR) [92] to describe the relationship between *in vitro* dissolution and *in vivo* bioavailability. The conventional statistical method often faces difficulties in linking the input variable with the dissolution used. In contrast, ANN-based models offer the advantage of incorporating a broader range of factors related to formulation composition, dissolution profiles, data from *in vitro* and *in vivo* studies, and manufacturing process parameters into the model. Leveraging these factors, the IVIVR neural model can comprehensively evaluate the various influences on *in vivo* responsiveness and make predictions regarding total plasma concentration-time profiles. This advancement in modelling contributes significantly to the understanding of drug behaviour and its translation from laboratory studies to real-life applications in pharmacology.

In many studies, ANN also has been mentioned for disease prediction and diagnosis. Given the current shift in the healthcare system towards disease prevention rather than treatment, there is a growing need for more accurate, rapid, and effective disease prediction and diagnosis. ANN has demonstrated its capabilities in meeting these requirements. For instance, in the field of medical diagnostics [96, 97, 98, 99], MLP and CNN have been used in arrhythmia classification using electrocardiogram (ECG) data and in sleep analysis and seizure detection based on electroencephalogram (EEG) data [100, 101]. Additional studies in [102] and [103] used echocardiograms to train and evaluate CNN models for multiple tasks, including automated identification of viewpoints and segmentation

of cardiac chambers. The output of these segmentation models has been helpful in quantifying chamber volumes, determining ejection fraction, and enabling automated assessment of longitudinal strain through speckle tracking. These models have also played a significant role in the detection of diseases such as hypertrophic cardiomyopathy, cardiac amyloidosis, and pulmonary arterial hypertension. ANN's ability to enhance disease prediction and diagnosis is making substantial contributions to the healthcare landscape, aligning with the industry's evolving focus on proactive healthcare management.

In the summarised studies from [104] by comparing different methods used for classification in pharmacology studies, the SVM and ANN generally demonstrated robust performance in classification tasks. DT faced challenges related to overfitting or underfitting issues. SVM, DT, and the naïve Bayes classifier are noted for their relatively low computational costs when compared to ANN. These methods often rely on feature data computed from data about the compounds, emphasising the importance of the quality of these features. For different data types, such as signals or images, ML methods like SVM, DT, and the naïve Bayes classifier may necessitate data pre-processing to compute relevant features. This pre-processing step can be resource-intensive, particularly when dealing with large data sets. Evaluating the feature computation poses challenges, as there is a risk of information loss during the process. Compared with different types of ANN which can directly use raw data like signals or images as input, the feature computation process may be avoided.

1.5 Contributions in parameter estimation

As mentioned in Section 1.3, local optimisation algorithms may encounter convergence issues whereas global optimisation algorithms usually require high computational costs. Moreover, those methods do not often incorporate available data. PINN-based methods benefit from both available data and the underlying dynamics of the system but need a large computational cost. In this thesis, we aim to explore an idea different from those methods which can leverage data from the target dynamic system but does not solely rely on training, as is the case with PINN-based methods.

To start with, let us set the framework of our study. We first consider a system of parametric ODEs which describes a phenomenon and we denote by θ the vector of the parameters. The parameter-to-observation map (the relationship between the parameter vector, eventually containing the initial conditions, and the measurements) is denoted by φ:

$$u = \varphi(\theta).$$

In addition to this model, we assume that we have access to a data set $U = \left\{u^{(i)}\right\}_{1\leq i\leq N_d}$ taken from a population of individuals or experiments (characterised by an unknown distribution of the parameters θ). Then we consider a specific new measurement u^* which is assumed to belong to the same population and, under these working hypotheses, our objective is to identify the parameter θ^* which satisfies:

$$u^* = \varphi(\theta^*).$$

To do so, we want to exploit the fact that in addition to the knowledge of the non-linear function φ which encodes our a priori knowledge about the system, we have access to U

which encodes a statistical knowledge about the population the datum belongs to.

The inherent challenge lies in the fact that, in the majority of cases, comprehensive information regarding the distribution of the parameters θ in the population is not available. In some cases, for instance, in models with a large number of parameters, it may happen that a significant number of these parameters are constant or are non-linearly dependent on each other in the population. Getting a knowledge about this could be precious in view of reducing the cost of the parameter estimation and increasing the robustness of the computation.

Given a data set U, we propose to setup an autoencoder in order to approximate the measurements $u \in U$. The encoder Ψ maps the values of u into a smaller dimensional latent variable $\alpha \in \mathbb{R}^p$. The decoder maps the latent variables to an approximation of the given data. In general, the value of p (the dimension of the latent variable space) is unknown, and it is a hyperparameter that needs to be pre-defined in order to use autoencoder. We can assume that, in order to get a good approximation of the data, we need the value of p to be larger or equal to the intrinsic dimension of the data. This is primarily determined by the identifiable parameters which are not constant in the population. It is henceforth crucial to have a good estimation of p. To this end, we introduce a criterion based on an approximation of the product of the Lipschitz constants of the encoder and decoder functions. This makes it possible to have an a posteriori criterion to adjust the value of p.

In order to perform the parameter estimation by exploiting the model and the result of the autoencoder, we introduce a novel method, Latent Variable Gradient Flow (LVGF), which is a gradient flow in the latent variables α. The LVGF method consists in computing a non-linear mapping between the parameter space and the latent variable space. This is done by using the model φ and the encoder Ψ. Given $u \in U$ and the encoder function Ψ, it holds:

$$\Psi \circ \varphi(\theta) = \alpha.$$

Using this relationship, we can transform an optimisation of a non-convex function in θ, into an optimisation of a convex cost function in α.

Chapter 2 will present in detail the parameter estimation problem and the LVGF method, including the rationale behind its derivation and the theoretical proofs. We will also present the theoretical results and methodology of using an autoencoder to perform the intrinsic dimension estimation of the latent variable. Furthermore, we will present the results of various tests conducted using the LVGF on simple dynamical systems, showcasing its applicability and effectiveness in estimating parameters. Considering the intrinsic dimension estimation, we will show that we can use the dimension of α to estimate the intrinsic dimension of some given data generated by the Van der Pol oscillator and FitzHugh–Nagumo model. The results obtained by considering Lipschitz constants as a criterion outperformed the classical approach, which only considers the autoencoder's reconstruction errors. Then, we will present some numerical results in which we compare the LVGF method with a classical gradient method. In these tests, we observe that LVGF method reduces the error in predicting parameter values by around 10 times compared with the gradient method.

1.6 Contribution in classification

Considering the high computational resources and time costs in drug safety assessment, there is a clear demand for automatic analysis tools in high-throughput drug screening. The goal is to improve the efficiency of drug discovery and development. Given the extracellular FP recorded by MEA from hiPSC-CMs, the first practical question is to determine whether a given chemical drug is altering the electrical activity of cardiomyocytes by disrupting the normal behaviour of certain ion channels like the hERG channel. This question can be viewed as a classification problem to distinguish drugs based on their effects on different ion channels. In this work, we will compare four different Neural Network methods to perform this classification. These differ in terms of pre-processing requirements and architecture.

In Chapter 3, we will explore a comparison of four types of ANN methods employed for the classification of hERG channel blockade. This chapter will encompass a detailed discussion of the experimental data, along with the associated data pre-processing steps tailored to each tested ANN method. Additionally, we will outline the setup of each ANN method and conclude with a comparison of their performance with respect to classification accuracy and computational costs. This chapter aims to provide a better understanding of the methodologies employed and the outcomes achieved through the ANN-based approach to hERG channel blockade classification. Among the tested methods, we can find that MLP and multivariate 1D-CNN could give an accuracy of more than 97% to predict the hERG channel blockade.

Considering the data recording step, it is important to monitor and enhance data quality during assessments. Experimental conditions, such as temperature control, cell conditions, and equipment variables, can introduce uncertainties that may result in problematic data recordings. As a consequence, we may create biases in the analysis results. Automated monitoring and identification of such problematic recordings can reduce potential biases and risks in the analysis results. To help in this task, we will investigate Replicator Neural Networks (RNNs), which are a type of autoencoder, to perform anomaly detection and remove some lower quality recordings.

Another contribution is to test an autoencoder for anomaly detection, it can be considered as a semi-supervised learning problem. Indeed, during the training phase, only normal recordings are used. The idea is the following: when reconstructing normal data the autoencoder will have a small error, whereas for abnormal data (not used in the training phase), it will have a larger (potentially much larger) error. By checking the value of the reconstruction error, we can identify anomalies. More details about the setup for the autoencoder in anomaly detection will be explained in the last section of Chapter 3. We compared a linear autoencoder with a non-linear one to test their capability to classify normal and abnormal data. The best method could provide anomaly detection with 95% accuracy.

Another topic presented in the first part of Chapter 4 concerns data generated in experiments studying genetic pathologies. We focus in particular on Brugada syndrome. Data generated using patient-derived hiPSC-CMs might have a large variability despite the fact that they belong to the same individual and they all carry the same genetic disease. This may be due to perturbations like genetic mutations [105]. The large variability in the data could raise a question about the reproducible character of the experiments and how much information do they convey about the disease mechanisms. In particular,

very different signal shapes can be observed in both healthy and non-healthy individuals, making it difficult to directly compare them.

In Chapter 4, we will explore MLP and multivariate-1DCNN to classify healthy and patient individuals. Subsequently, we will introduce a strategy for distinguishing between patients with different degrees of symptom severity by employing an autoencoder in feature extraction combined with a clustering method. Afterwards, we will discuss potential future directions regarding the use of the cardio electrophysiology model to explain the mechanisms of Brugada syndrome, especially exploring further possibilities by testing the LVGF method to address this specific question. The points mentioned above would provide valuable assistance to scientists studying Brugada syndrome.

Finally, in the second part of Chapter 4, we will also explore an investigation centred on detecting age-related effects in dogs. This exploration will encompass the application of statistical, ANN, and mathematical modelling methods, utilising *in vivo* data. The objective is to offer a comprehensive understanding of age-related impacts, leveraging diverse analytical approaches to enhance our insights into the physiological changes associated with ageing. This will help scientists to see the advantages and disadvantages of each type of method applied to dogs' cardiovascular data.

1.7 Publications and preprint

The contributions presented above correspond to the following preprint and publications:

1. Preprint: Muriel Boulakia, Haibo Liu, and Damiano Lombardi. Parameter identification through gradient flow on latent variables. 2023. *https://inria.hal.science/hal-04364114*

2. Publication: Haibo Liu, Tessa De Korte, Sylvain Bernasconi, Christophe Bleunven, Damiano Lombardi and Muriel Boulakia. Artificial Neural Network Comparison on hERG Channel Blockade Detection. International Journal of Computer Applications 184(14):1-9, May 2022.

3. Publication: Elham Ataei Alizadeh, Sara Costa Faya, Haibo Liu, Damiano Lombardi, Sylvain Bernasconi, Pieter-Jan Guns and Michael Markert. Comparison of statistical, machine learning, and mathematical modelling methods to investigate the effect of ageing on dog's cardiovascular system. ESAIM: Proceedings and Surveys, 73, 2-27, 2023.

Chapter 2

Parameter Identification Through Gradient Flow on Latent Variables

This chapter corresponds to a submitted article, achieved with Muriel Boulakia and Damiano Lombardi

Abstract

In this chapter, we consider a system of parametric ODEs which involves unknown parameters and we seek to identify the values of the parameters associated to a given measurement. To do so, we place ourselves within the fairly usual framework that this single measurement is in fact taken from a population of data and we therefore want to take advantage of the statistical knowledge about the population to regularise the classical minimisation problem associated to our identification problem. In the method that we propose and that we call the Latent Variable Gradient Flow method, the data set is represented by an autoencoder neural network which allows to associate to each element of the data set a latent variable. Then, introducing a non-linear mapping between the parameter space and the latent variable space allows to convexify the cost function and to demonstrate convergence properties. These properties are numerically illustrated with different tests on Van der Pol and FitzHugh-Nagumo models.

2.1 Introduction

In many studies of experimental sciences, such as chemistry, biology or environmental engineering, mathematical models are used to describe the behaviour of the dynamical systems. Those mathematical models consist of systems of ordinary differential equations (ODEs). For the most part, these ODEs contain parameters that are associated to physical, biological, or other properties of the system. To assess the relevance of the model, it is necessary to understand the role of these parameters and to set their values in order to generate signals that are close to reality. In addition to parameters, the system is completed by the data of the initial state whose values may be unknown or uncertain. When faced with an experimental observation, identifying the unknown parameters and the initial state to closely match the experimental results is a key step in the development and understanding of ODEs [106]. However, this identification problem which belongs

to the class of "inverse problems" is a complex task, especially when the ODEs system consists of a large number of equations and parameters.

To define the context more precisely, let us introduce some notations which will be defined in more detail later on. Let a system be described by a mathematical model, in which the output u is affected by a certain number of parameters and by the initial conditions. Let us denote by θ the vector of these parameters and initial conditions. To simplify the formulation, in what follows, we will describe in general θ as a model parameters vector, even if it also includes the initial conditions. Then, we introduce the parameter-to-output map φ which associates to the vector θ the output u: $u = \varphi(\theta)$.

The general problem reads as follows: given some measurements u^* of the output, we try to estimate the model parameters θ^* such that $u^* = \varphi(\theta^*)$.

Classically, the estimation of model parameters ([57, 56]) is cast as an optimisation problem. In particular, a cost function is defined based on the data misfit and the parameter θ is estimated by minimising the discrepancy between the model observations $\varphi(\theta)$ and the actual measurement u^* of the system. This formulation often leads to a non-linear non-convex possibly high-dimensional optimisation problem. This could be dealt with in two ways:

1. Local (often gradient based) optimisation methods. These methods include Quasi-Newton methods [107, 108], the Gauss-Newton method (incorporating explicitly derived and efficiently computed sensitivity equations [58]), as well as other gradient-based methods [109]. The local gradient-based optimisation methods might encounter convergence issues when the initial guess is far from the true parameter values [62].

2. Global optimisation approach. For example, commonly utilized stochastic search algorithms include evolutionary computation, adaptive stochastic methods, and clustering methods. Other global optimisation methods are the genetic algorithms, as proposed for instance in [110] for the determination of rate constants in heterogeneous reaction systems, and collocation methods, as proposed for instance in [111] and [112]. In many cases, stochastic search algorithms are used to select good initial guesses for starting either a Quasi-Newton method or a gradient-based type minimisation. Global optimisation methods are often very expensive from a computational point of view and might become prohibitive when the number of parameters is large [113, 114].

Another way to try to solve the problem is purely based on data. Let us assume that, in certain situations, we have examples of pairs of parameters and observations, $\left\{\theta^{(i)}, u^{(i)}\right\}_{1\leq i\leq N_d}$. For example, the work in [76] used a supervised convolutional machine-learning method to learn the physical model parameters. The advantage of this kind of model is that no a priori knowledge about the system is required. The main disadvantage consists in the fact that we often need a large number of training pairs parameters-observations.

Recently, many efforts were made in order to try to exploit both the knowledge coming from the mathematical model and from sets of data, such as physics-informed neural networks (PINNs) mentioned in [80, 77, 79, 78], deep operator network (DeepONet) find in [81] and physics-informed DeepONet (PI-DeepONet) find in [82].

PINNs operate by taking time as input and by training the network to generate corresponding observations of the dynamic system. This training process leverages both

the available data and the underlying dynamics of the system. When addressing inverse problems, the PINN's loss function integrates two components: the discrepancies between predicted observations and given data, and the residual of the differential equations, which depends on the parameters. Then, the network optimises the parameters of the network and estimates the parameters of the dynamical system. PINNs might suffer from the so-called spectral bias [83], which refers to the fact that it tends to prioritize learning the low-frequency part of the solution, not rendering properly multiple frequency scales. To overcome this issue, Fourier Features-Neural Networks (FF-NN) have been proposed to solve dynamical systems and perform inverse problems [84]. Another limitation of PINN-based methods is that they often require a large number of iterations and evaluations to converge to an accurate solution. This computational intensity is a drawback, especially when dealing with dynamic systems consisting of a large number of equations and parameters. In the continuity of PINN-based methods, the method that we propose in this paper seeks to leverage both available data and underlining dynamical systems. However, as we will see, our method provides an alternative method of PINN for which the computational demanding part corresponding to the training on the data set is done in a preliminary part.

About the data set, we focus here on a specific situation: we consider that we do not have data composed of pairs of parameters-observations but that we only have observations

$$\left\{u^{(i)}\right\}_{1\leq i\leq N_d}$$

consisting in a collection of measurements taken on a population or a set of experiments and we assume that our (novel) observation u^* comes from the same population. This situation, which is quite common in a number of realistic applications, is by far more difficult than the one in which pairs of parameters-observations are available (as we have no direct information about the parameters anymore).

In this paper, we will propose a new method of parameter estimation which relies on a regularisation of a classical parameter estimation by means of the data set. A representation of the data set by an autoencoder will allow to describe an element of the data set by a latent variable. Then, the method performs a non-linear mapping between the parameter space, where the cost function associated with the parameter estimation is non-convex, and the latent variable space, where the cost function is convex. Since our method can be described as a gradient flow for the latent variables of the autoencoder, it will be called the Latent Variables Gradient Flow (LVGF) method.

In addition to the presentation and study of the LVGF method, this work also presents a novel method to estimate the intrinsic dimension of a data set $U = \left\{u^{(i)}\right\}_{1\leq i\leq N_d}$. Identifying this intrinsic dimension is an important point in the implementation of our method since it allows to set the value of the latent dimension of the autoencoder trained on the data set. The proposed method will be based on the use of autoencoders and a criterion based on the approximation of the Lipschitz constant of the encoding and decoding maps. For an ODE system satisfying observability and identifiability properties, this intrinsic dimension is related to the dimension of the set of parameters which are varying in the population.

The structure of the work is the following: in Section 2.2, we will state the parameter identification problem, present our parameter estimation method which we call the LVGF method, and study the convergence properties of this method. Moving on to

Section 2.3, we will show some theoretical results and the methodology of using an autoencoder to perform intrinsic dimension estimation which is a part of the parameter estimation problem. Section 2.4 will be dedicated to demonstrating the performance of our proposed intrinsic dimension estimation method using two example dynamical systems. At last, in Section 2.5, we will assess the LVGF method's ability to solve the parameter estimation problem on the dynamical systems presented in the first section and compare its performance with the classical minimization method.

2.2 Parameter estimation

2.2.1 Problem statement

Let us consider a system of ODEs:

$$\begin{cases} \dot{x}(t) = f(x(t), \zeta), \, \forall \, t \in (0, T), \\ x(0) = x_0 \end{cases} \tag{2.2.1}$$

where $x : [0, T] \to \mathbb{R}^N$ is a real vector valued function, $\zeta \in \mathbb{R}^q$ represents a vector of constant parameters and $x_0 \in \mathbb{R}^N$ is the initial state. We assume that, for every $x_0 \in \mathbb{R}^N$ and $\zeta \in \mathbb{R}^q$, system (2.2.1) admits a unique solution x in $C^1([0, T])^N$.

For this system, we consider that the parameter values and the initial condition are unknown or uncertain and we are interested in the simultaneous identification of parameters and initial data. We denote by $\theta = (x_0, \zeta) \in \mathbb{R}^m$ with $m = N + q$ the vector that we want to identify and look for θ in a subset of $\mathbb{R}^m$ that we denote by Θ.

The overall goal of our study is to identify the value of $\theta \in \Theta \subset \mathbb{R}^m$, by exploiting some discrete measurements that we denote by $u \in \mathbb{R}^n$ of the solution x of system (2.2.1). We denote by H the observation operator that models the measurement procedure:

$$H : C^1([0, T])^N \to \mathbb{R}^n. \tag{2.2.2}$$

For the sake of simplicity in the presentation, we assume that H only gives one scalar information (for instance, one component of the vector valued function x) discretised in time.

In addition, we define the map which associates the measurement to the unknown parameters and initial data:

$$\begin{array}{rccc} \varphi : \mathbb{R}^m & \to & \mathbb{R}^n \\ \theta & \to & H(x(\theta, \cdot)) \end{array} \tag{2.2.3}$$

where $x(\theta, \cdot)$ is the solution of system (2.2.1) associated to $\theta = (x_0, \zeta)$.

The classical deterministic identification problem can be formulated in the following way: we assume that a datum $u^* \in \mathbb{R}^n$ is given and we would like to identify $\theta^* \in \Theta \subset \mathbb{R}^m$ such that

$$\varphi(\theta^*) = u^*. \tag{2.2.4}$$

Most of the time, this problem is numerically studied by reformulating it as an optimisation problem. In a classical way, the functional to minimise can be for instance

of the form:

$$J(\theta) = \|\varphi(\theta) - u^*\|_n^2 \tag{2.2.5}$$

where $\|\cdot\|_n$ corresponds to the Euclidean norm in $\mathbb{R}^n$.

Such a problem is known to be cumbersome in a number of situations, due to the fact that it is usually non-linear and non-convex. We will give a practical example of those situations in the end of this section.

In the present work, we consider that, in addition to the map φ and the datum u^*, we have access to a set of $N_d \in \mathbb{N}^*$ data, denoted $U = \left\{u^{(i)}\right\}_{1\leq i\leq N_d} \in \mathbb{R}^n$ which u^* is a part. We assume that these data come from a population (of individuals, or experiments), characterised by a certain (unknown) distribution of parameters θ. This setting is justified by the fact that it is indeed common to have access not to a single measurement but to a large number of data. Therefore, our objective is to study how we can take advantage of this whole set of measurements U in the identification of parameters associated to a single measurement u^* and if adding information coming from the knowledge of the set U can allow to construct a regularisation for the parameter estimation problem. This idea will be the starting point of the LVGF method presented in 2.2.3.

In what follows, we will illustrate our approach on two ODE systems.

First, we will consider Van der Pol model, a model proposed in the 1920s by Van der Pol [115] to represent the oscillations in vacuum tube circuits. It is given by the following system:

$$\begin{cases} \dot{x} = v, \\ \dot{v} = \mu(1 - x^2)v - x, \\ (x, v)(0) = (x_0, v_0). \end{cases} \tag{2.2.6}$$

In this equation, $\mu > 0$ is a fixed parameter which reflects the degree of nonlinearity of this system, x_0 is the initial position and v_0 is the initial velocity. For this model, we assume that we measure the state variable x on the whole time interval and we are interested in the identification of $\theta = (x_0, v_0, \mu)$.

Second, we will consider FitzHugh-Nagumo model which describes the dynamics of a spiking neuron. It is given by

$$\begin{cases} \dot{v} = v - \frac{v^3}{3} - w + I_{ext}, \\ \dot{w} = \frac{1}{\tau}(v + a - bw), \\ (v, w)(0) = (v_0, w_0) \end{cases} \tag{2.2.7}$$

where v corresponds to the membrane potential, w to the recovery variable and I_{ext} to a stimulus current. For this model, we assume that we only observe the state variable v and that the value of I_{ext} is known. For this problem, we are interested in the identification of $\theta = (v_0, w_0, a, b, \tau)$.

2.2.2 Identifiability and observability of the inverse problem

Before testing numerical methods for the resolution of the identification problem (2.2.4), it is essential to ensure the uniqueness of a solution to (2.2.4) or, in other words, the injectivity of φ in Θ. Not having this type of theoretical property can in fact explain

numerical convergence issues independently of the numerical method chosen.

Moreover, contrary to Tykhonov methods that can overcome a lack of uniqueness in the identification problem through the add of a prior, in our case the only information that we add to regularise our minimisation problem comes from the knowledge of a data set. Thus, the underdetermined nature of the problem would remain despite the regularisation process.

Since θ is composed both of model parameters and initial data, this identification property is related to two distinct notions in inverse problems: first, the notion of identifiability which refers to the fact that model parameters are uniquely determined by the output and, second, the notion of observability which refers to the fact that the initial conditions are uniquely determined by the output.

Let us notice that the identifiability and observability of the problem introduced above can be reduced to the observability of the following augmented system (as presented for instance in [116] and [117]):

$$\begin{cases} \dot{X}(t) = F(X(t)), \forall\, t \in (0,T), \\ X(0) = (x_0, \zeta) = \theta \end{cases} \tag{2.2.8}$$

where θ is taken in Θ, $X = (x, \zeta) : [0,T] \to \mathbb{R}^m$ is a regular function and F is given by

$$\begin{array}{rcl} F : \mathbb{R}^m & \to & \mathbb{R}^m \\ (x, \zeta) & \to & (f(x,\zeta), 0). \end{array} \tag{2.2.9}$$

Contrary to the observation operator given by (2.2.2) and adapted to the numerical framework where the solution is discretised in time, we consider in the theoretical framework that we have access to measurements at each time in $[0,T]$ and we denote by

$$h : C^1([0,T])^N \to C^1([0,T]) \tag{2.2.10}$$

the observation map which associates the time-dependent measurement to the solution X of the ODE system (2.2.8).

Numerous works address the question of identifiability and observability for ODE systems and there is a wide variety of methods for doing so. About ODEs modelling, we can quote [118], [119], [120] and [121] among many references. Following these last two references, we are interested by the structural observability property for system (2.2.8) (or by the structural identifiability and observability property for system (2.2.1)) which means that, for almost all initial conditions $X_1(0)$ and $X_2(0)$ in Θ

$$h(X_1) = h(X_2) \text{ in } [0,T] \Rightarrow X_1(0) = X_2(0).$$

For the two ODE models presented in the previous section, the we have the following results:

Proposition 2.2.1. *If we measure the variable x, Van der Pol model (2.2.6) is structurally observable and identifiable in the variables (x_0, v_0, μ), in the sense that (x_0, v_0, μ) is uniquely determined from the measurement of the function x in $[0,T]$.*

Proposition 2.2.2. *If we measure the variable v, FitzHugh-Nagumo model (2.2.7) is structurally observable and identifiable in the variables (v_0, w_0, a, b, τ), in the sense that*

(v_0, w_0, a, b, τ) is uniquely determined from the measurement of the function v in $[0, T]$.

The proofs of these results are presented in Appendix A.

2.2.3 Latent Variables Gradient Flow (LVGF) for parameter estimation

Description of the LVGF method

The first ingredient of the proposed LVGF method relies on the construction of an autoencoder approximating the data set U. It consists in an encoder map $\Psi : \mathbb{R}^n \to \mathbb{R}^p$ which compresses an input data $u \in U$ into a latent variable $\alpha = \Psi(u) \in \mathbb{R}^p$ with $p < n$ and a decoder map $\Psi^\dagger : \mathbb{R}^p \to \mathbb{R}^n$ which recreates an approximation of the input data from the latent variable. Thus, in the data set U, which we can think about as a sampling of an embedded submanifold $\mathcal{U} \subset \mathbb{R}^n$, $\Psi^\dagger \circ \Psi(u)$ is a good approximation of u. The set-up of the autoencoder will be specified in Section 2.3.1. Let us notice that the choice of the hyperparameter p is related to the notion of intrinsic dimension of $\mathcal{U}$ and, by using some recent results in approximation theory, the estimation of the intrinsic dimension will be discussed in Section 2.3.2.

The rationale behind the introduction of the autoencoder is the following: the latent variables $\alpha \in \mathbb{R}^p$ are, at the same time, observable and related to the parameters $\theta \in \Theta$. Indeed, given $u^* = \varphi(\theta^*) \in \mathcal{U}$, thanks to the encoder Ψ, we can easily compute the latent variable α^* associated with it, $\alpha^* = \Psi(u^*)$. Furthermore, we are aware of the relationship between θ^* and α^* which is defined as follows:

$$\Psi \circ \varphi(\theta^*) = \alpha^*.$$

By leveraging these two observations, we can introduce a method in which, by trying to match the latent variables values α^*, we estimate θ^*.

The way in which we try to reach this goal is by introducing a gradient flow in the latent variable α, which motivates the method name: Latent Variable Gradient Flow (LVGF).

In what follows, we use the notation $g = \Psi \circ \varphi : \mathbb{R}^m \to \mathbb{R}^p$ and assume that

$$g \in C^{1,\beta}(\Theta), \text{ for some } \beta > 0 \text{ and } \operatorname{rank} \nabla g(\theta) = p, \forall\, \theta \in \Theta. \tag{2.2.11}$$

Under these hypotheses, we can consider the generalized inverse (also called the MoorePenrose pseudoinverse [122]) of $\nabla g(\theta)$ defined for a matrix M, by $M^\dagger := M^T(MM^T)^{-1}$. Moreover, we assume that there exists a constant $\overline{C} > 0$ such that, for all $\theta \in \Theta$

$$\|\nabla g(\theta)^\dagger\| \leq \overline{C}. \tag{2.2.12}$$

The LVGF algorithm is based on the following iterative process: let a step $s > 0$ and an initial value $\theta_0 \in \Theta$ be given. We denote by α_0 the initial value associated to θ_0, that is $\alpha_0 = g(\theta_0)$. Then, we define the sequence $(\theta_k)_{k\in\mathbb{N}}$ iteratively by: for $k \in \mathbb{N}$

$$\begin{cases} \Delta\theta_k &= sM_k^\dagger(\alpha^* - \alpha_k) \text{ where } M_k = \nabla g(\theta_k) \\ \theta_{k+1} &= \theta_k + \Delta\theta_k \\ \alpha_{k+1} &= g(\theta_{k+1}). \end{cases} \tag{2.2.13}$$

By definition, $\Delta\theta_k$ is thus the unique vector of smallest norm solution in $\mathbb{R}^m$ of the system

$$M_k \Delta\theta_k = s(\alpha^* - \alpha_k).$$

Convergence of the LVGF method

In the following proposition, we prove a convergence result on the sequence (θ_k) in the specific case where the parameter space is the full space $\mathbb{R}^m$.

Proposition 2.2.3. *We assume that $\Theta = \mathbb{R}^m$ and that g satisfies the hypotheses (2.2.11)-(2.2.12). Then, for $s > 0$ small enough, the sequence $(\theta_k)_{k\in\mathbb{N}}$ defined by (2.2.13) converges to some $\tilde{\theta} \in \mathbb{R}^m$ which satisfies*

$$\alpha^* = g(\tilde{\theta}). \tag{2.2.14}$$

Moreover, there exists a constant $C > 0$ depending on $\alpha^* - \alpha_0$ and $\overline{C}$ such that

$$\|\tilde{\theta} - \theta_k\|_m \leq \frac{C}{s}\big(1 - \frac{s}{2}\big)^k$$

Proof. Let $k \in \mathbb{N}$ be given. According to (2.2.13), we have

$$\alpha^* - \alpha_{k+1} = \alpha^* - g(\theta_k + \Delta\theta_k).$$

Since $g \in C^{1,\beta}(\Theta)$, we can write

$$g(\theta_k + \Delta\theta_k) = g(\theta_k) + M_k\Delta\theta_k + \xi_k = \alpha_k + s(\alpha^* - \alpha_k) + \xi_k$$

where $\xi_k \in \mathbb{R}^p$ satisfies

$$\|\xi_k\|_p \leq \tilde{C}\|\Delta\theta_k\|_m^{1+\beta} \leq \tilde{C}\overline{C}^{1+\beta}s^{1+\beta}\|\alpha^* - \alpha_k\|_p^{1+\beta}$$

where we have used (2.2.12) and denoted by $\tilde{C}$ the Hölderian constant of ∇g. This implies that

$$\begin{aligned}\|\alpha^* - \alpha_{k+1}\|_p = \|(1-s)(\alpha^* - \alpha_k) - \xi_k\|_p \\ \leq (1-s)\|\alpha^* - \alpha_k\|_p + \tilde{C}\overline{C}^{1+\beta}s^{1+\beta}\|\alpha^* - \alpha_k\|_p^{1+\beta}\end{aligned}$$

Assume now that the step $s \in]0,1[$ is chosen such that

$$\tilde{C}\overline{C}^{1+\beta}s^{\beta}\|\alpha^* - \alpha_0\|_p^{\beta} \leq \frac{1}{2}.$$

Then, an argument by induction allows us to deduce from the previous inequality that, for all $k \in \mathbb{N}$

$$\|\alpha^* - \alpha_k\|_p \leq \|\alpha^* - \alpha_0\|_p.$$

Therefore, we deduce that, for all $k \in \mathbb{N}$

$$\|\alpha^* - \alpha_{k+1}\|_p \leq \big(1 - \frac{s}{2}\big)\|\alpha^* - \alpha_k\|_p.$$

So the sequence $(\alpha_k)_{k\in\mathbb{N}}$ linearly converges to α^* and

$$\|\alpha^* - \alpha_k\|_p \leq \big(1 - \frac{s}{2}\big)^k \|\alpha^* - \alpha_0\|_p.$$

This implies that the sequence $(\theta_k)_{k\in\mathbb{N}}$ satisfies

$$\|\theta_{k+1} - \theta_k\|_m = \|\Delta\theta_k\|_m \leq C\|\alpha^* - \alpha_k\|_p \leq C\big(1 - \frac{s}{2}\big)^k \|\alpha^* - \alpha_0\|_p.$$

In particular, since $(\theta_k)_{k\in N}$ is a Cauchy sequence, it converges to some $\tilde{\theta}$. Moreover, we have

$$\|\tilde{\theta} - \theta_k\|_m \leq \frac{2C}{s}\big(1 - \frac{s}{2}\big)^k \|\alpha^* - \alpha_0\|_p$$

At last, passing to the limit in the expression $\alpha_k = g(\theta_k)$, we deduce the formula (2.2.14). □

Corollary 2.2.4. *Under the same hypotheses as Proposition 2.2.3, we assume in addition that $\Psi : \mathcal{U} \to \mathbb{R}^p$ and $\varphi : \mathbb{R}^m \to \mathbb{R}^n$ are one-to-one and that $\tilde{\theta} \in \varphi^{-1}(\mathcal{U})$. Then, for $s > 0$ small enough, the sequence $(\theta_k)_{k\in\mathbb{N}}$ defined by (2.2.13) linearly converges to $\theta^* \in \mathbb{R}^m$.*

This corollary directly comes from the fact that, by definition of α^* and according to equation (2.2.14), the limit $\tilde{\theta}$ of the sequence $(\theta_k)_{k\in\mathbb{N}}$ satisfies:

$$\Psi \circ \varphi(\tilde{\theta}) = \Psi \circ \varphi(\theta^*).$$

Thus, under the hypotheses of the corollary, $\tilde{\theta} = \theta^*$.

Let us add a few comments on the additional assumptions of Corollary 2.2.4. First, we remark that the hypothesis that $\Psi : \mathcal{U} \to \mathbb{R}^p$ is one-to-one is naturally related to the encoding properties of this function for which a given latent variable can correspond to only one vector in $\mathcal{U}$. The hypothesis of injectivity of $\varphi : \mathbb{R}^m \to \mathbb{R}^n$ is also a natural assumption related to the identifiability and observability discussed in Section 2.2.2.

Since the algorithm given by equation (2.2.13) does not allow to ensure that the property $\tilde{\theta} \in \varphi^{-1}(\mathcal{U})$ is satisfied, in practice, we correct it by adding a projection procedure. Let us denote by $\mathcal{P} : \mathbb{R}^m \to \mathbb{R}^m$ a projection operator on $\varphi^{-1}(\mathcal{U})$. For a given $\theta_{-1} \in \mathbb{R}^m$, we compute $\theta_0 = \mathcal{P}(\theta_{-1}) \in \varphi^{-1}(\mathcal{U})$ and replace system (2.2.13) by

$$\begin{cases} \Delta\theta_k &= sM_k^{\dagger}(\alpha^* - \alpha_k) \text{ where } M_k = \nabla g(\theta_k) \\ \theta_{k+1} &= \mathcal{P}(\theta_k + \Delta\theta_k) \\ \alpha_{k+1} &= g(\theta_{k+1}). \end{cases} \tag{2.2.15}$$

Remark 2.2.5. *In most of the realistic situations, Θ is not the whole space $\mathbb{R}^m$ but a compact set in $\mathbb{R}^m$. The method can be written in a parametric domain (say, a box) containing this domain, and such that θ^* is an interior point of the domain. In the case in which the iterates are all interior points, the result presented holds true without any modification. If some iterates fall out of the domain, they can be reprojected into it. In this case, further investigations would be needed to determine the speed of convergence.*

Remark 2.2.6. *The LVGF method can be applied also in the case in which $\bar{p} = m$ and it has a remarkable interpretation. Let us assume that a parametric model is identifiable. Let us recall that $g \in C^{1,\beta}(\Theta)$. It follows that $M^{\dagger} = M^{-1}$, for all $\theta \in \Theta \subseteq \mathbb{R}^m$. Solving*

the problem with a classical cost function would amount to solve a non-linear, non-convex optimisation problem. The function $g = \Psi \circ \varphi$ can be seen as a change of variables making it possible to cast the parameter estimation problem as the optimisation of the convex function $J(\alpha) = \|\alpha^* - \alpha\|_m^2$.

Remark 2.2.7. *The LVGF method comes with a way to verify, a posteriori, whether the model solution $\varphi(\theta)$ does not belong to the population. This can be done by exploiting the decoder $\psi^\dagger$. Let us consider: $\|\Psi^\dagger \circ \Psi \circ \varphi(\theta) - \varphi(\theta)\|_n^2$. If this quantity is significantly larger than the error obtained on the validation set when training the autoencoder, it implies that the obtained solution is far from the ones belonging to the population.*

Numerical implementation

For the numerical implementation of this method, it is necessary to describe how the projection on $\varphi^{-1}(\mathcal{U})$ is defined and computed. By noticing that the elements θ which belong to $\varphi^{-1}(\mathcal{U})$ satisfy

$$\Psi^\dagger \circ \Psi \circ \varphi(\theta) - \varphi(\theta) = 0,$$

we introduce the functional

$$E(\theta) = ||\Psi^\dagger \circ \Psi \circ \varphi(\theta) - \varphi(\theta)||_n^2 \tag{2.2.16}$$

and define the projection $\mathcal{P}(\theta)$ as the solution of the minimisation of E starting from the initial data θ. The minimisation of the function E is achieved by using the gradient descent method.

In practice, to initialise the sequence $(\theta_k)_{k\in\mathbb{N}}$ with a θ_0 such that $\theta_0 \in \varphi^{-1}(\mathcal{U})$, the gradient descent method may fail if we start with a value far from the submanifold $\varphi^{-1}(\mathcal{U})$. So, we consider different initial values for θ_0, run for each initial value the gradient descent method and keep a θ_0 for which the optimisation procedure gives a value of E close to zero. In addition, we do not run the projection step at each iteration but only if θ_k begins to move away from $\varphi^{-1}(\mathcal{U})$ which is tested by comparing the value $E(\theta_k)$ to a threshold value. The details of the LVGF method are shown in Algorithm 1.

2.2.4 A numerical illustration of LVGF

In this section, a first numerical illustration of the LVGF method is proposed for the identification of a two-dimensional parameter vector. Moreover, our method is compared to a gradient descent method applied to the minimisation of the classical functional (2.2.5).

We consider Van der Pol model (2.2.6) and are interested by identifying $\theta = (v_0, \mu)$ whereas x_0 is assumed to be known and its value is given by $x_0 = 0.5$. We consider a measurement u^* defined by $u^* = \varphi(\theta^*)$ with $\theta^* = (0.5, 0.5)$ and our objective is to identify this value θ^*. In addition to u^*, we consider that we have access to a data set U (contain $N_d = 1200$ simulations) which has been generated in a preliminary step by varying the values of μ, randomly drawn from a uniform distribution in $[0.05, 2.0]$ and setting $v_0 = 0.5$. So in this case, the intrinsic dimension of the submanifold $\mathcal{U}$ is equal to 1. The latent dimension of the autoencoder trained on this data set is fixed to $p = 1$ (this dimension can also be rediscovered following the method presented in Section 2.3.2). We used AE1 (mentioned in Table 2.1) to approximate the data set U. There are 1000 samples used in the training set and 200 in the validation set. Moreover, the parameters given as input of

Algorithm 1 The LVGF algorithm

Input: N_{max} = maximum number of iterations; σ = tolerance for $||\alpha^* - \alpha||_p$; $S = [s_1, s_2, ..., s_n]_{1 \leq i \leq n}$
start with a random initial guess θ_0
if $E(\theta_0) > \epsilon$ **then**
  run the projection $\theta = \mathcal{P}(\theta_0)$
else
  $\theta = \theta_0$
end if
compute $\alpha = \Psi(\varphi(\theta))$
while $j \leq N_{max}$ and $||\alpha^* - \alpha||_p \geq \sigma$ **do**
  compute $M = [\nabla\Psi(\varphi(\theta))][\nabla\varphi(\theta)]$
  compute $\Delta\theta = [M]^\dagger(\alpha^* - \alpha)$
  select $1 \leq i \leq n$ the minimum point of $\{\|\alpha^* - \Psi(\varphi(\theta + s_i\Delta\theta))\|_p,\ 1 \leq i \leq n\}$
  update $\theta = \theta + s_i\Delta\theta$
  if $E(\theta) > \epsilon$ **then**
    run the projection $\theta = \mathcal{P}(\theta)$
  end if
  compute $\alpha = \Psi(\varphi(\theta))$
end while
return the value of the last iteration of θ

Algorithm 1 have been fixed to $N_{max} = 2000$, $\sigma = 10^{-4}$ and $S = [0.0005, 0.0025, 0.0125, 0.0625]$.

In addition to the LVGF method, we have implemented the gradient descent method applied to the minimisation of the classical functional (2.2.5) in order to compare both methods. In Figure 2.1, we plot the contour of the classical cost function given by equation (2.2.5). We observe that the function is non convex and presents local minima in addition to the global minimum at $\theta^* = (0.5, 0.5)$ represented by a red sign "+". The different iterations for the classical gradient descent method applied to equation (2.2.5) and for the LVGF method have been also depicted starting with the same initialisation $\theta_0 = (0.8, 1.9)$.

It is interesting to observe that the gradient descent method converges as expected to a local minimiser whereas the LVGF method is able to go against the direction of the steepest slope in order to change valley and reach the global minimiser. Thus the information coming from the data set and added through the use of the latent variable made it possible to correct the convexity defect of the classical functions and to identify the right value of the unknown parameters.

Further numerical tests to better assess the performances of LVGF will be presented in Section 2.5.

2.3 Lipschitz-stable autoencoders and intrinsic dimension

As presented in Section 2.2.3, our LVGF method relies on a description of the data set $U = \{u^{(i)}\}_{1 \leq i \leq N_d}$ by an autoencoder. These neural networks are dimensionality reduction

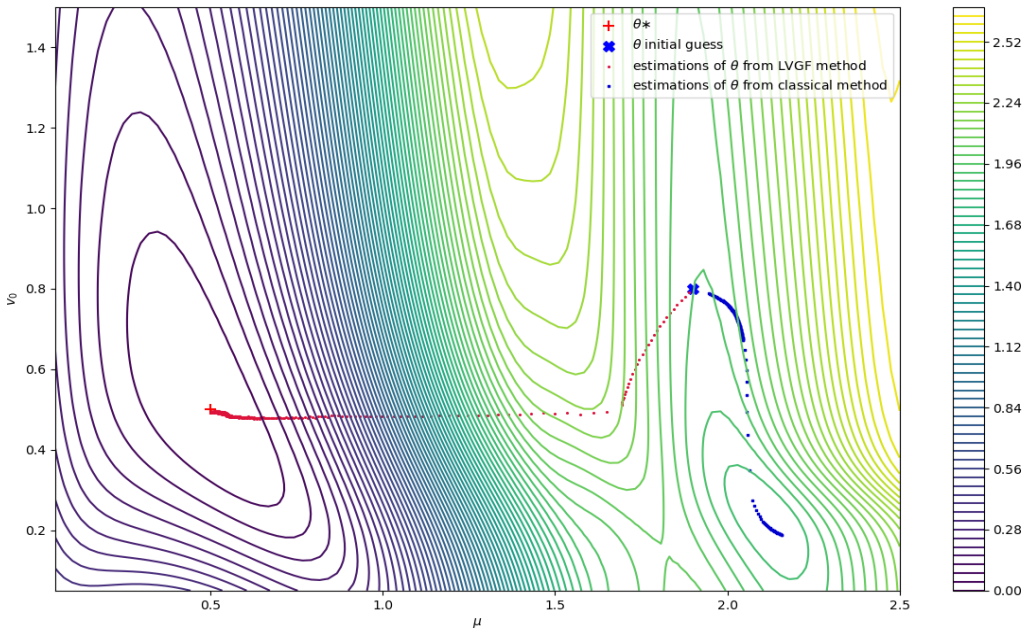


Figure 2.1: Contour plot of the classical function $J(\theta) = \|u^* - \varphi(\theta)\|_n^2$ with $\theta = (v_0, \mu)$: comparison between the iterations of the classical gradient descent method (blue dots) and the LVGF steps (red dots) with the same initial guess (blue cross). The value of the true parameter, which is the minimum of J is $\theta^* = (0.5, 0.5)$, marked with a red cross.

algorithms which learn two functions called encoder and decoder. The encoder function associates to an input vector in $\mathbb{R}^n$ a vector called a code or a latent variable of smaller dimension p.

It is important to notice that the dimension p is an hyperparameter that has to be set before training the autoencoder even though, in general, the intrinsic dimension of a data set is unknown. That is why the first question that naturally arises in the representation of a data set by an autoencoder is to understand how to settle the dimension of the latent variable. In this section, we propose a criterion based on the notion of stable manifold width to estimate the intrinsic dimension of the data set. This criterion is a consequence of the result given by Corollary 2.3.2 which highlights the fact that, if the dimension of the latent variable is smaller than the intrinsic dimension, the product of the Lipschitz constants of the encoder and decoder functions will blow up.

2.3.1 Presentation of the autoencoder

In this section, we describe the autoencoder neural network and give some details on the implementation of the autoencoder that we have considered in this work.

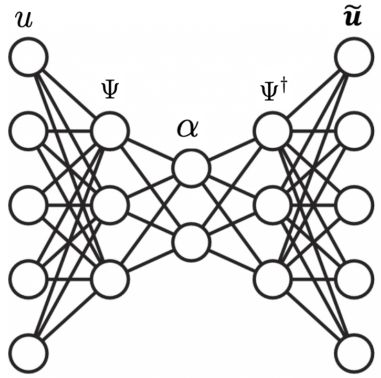


Figure 2.2: autoencoder systemic structure.

As described in [123] and [124], an autoencoder is a feed-forward neural network with a systemic structure that is composed of multiple hidden layers which include an encoding and a decoding part. The input and output layers have the same size as shown in Figure 2.2. The encoder compresses the input data ($u \in \mathbb{R}^n$) to some latent representation ($\alpha \in \mathbb{R}^p$):

$$\Psi : \begin{cases} \mathcal{U} \longrightarrow \mathbb{R}^p \\ u \longmapsto \alpha := \Psi(u) \end{cases}$$

whereas the decoder decodes the compressed latent representation α and reconstructs the input in $\tilde{u} \in \mathbb{R}^n$:

$$\Psi^\dagger : \begin{cases} \mathbb{R}^p \longrightarrow \mathbb{R}^n \\ \alpha \longmapsto \tilde{u} := \Psi^\dagger(\alpha). \end{cases}$$

Among the key aspects that have to be set in order to define an autoencoder, we need to specify the loss function to be minimised. In our paper, we have considered the

following loss function:

$$\frac{1}{N_d}\sum_{i=1}^{N_d}\|u^{(i)}-\Psi^\dagger\big(\Psi(u^{(i)})\big)\|_n^2 \tag{2.3.1}$$

where $U=\left\{u^{(i)}\right\}_{1\leq i\leq N_d}$ corresponds to the data set. In addition, we chose the exponential linear unit (ELU) [125] as the activation function:

$$ELU(x)=\begin{cases} x & \text{if } x\geq 0 \\ \rho(\exp(x)-1) & \text{otherwise}\end{cases} \tag{2.3.2}$$

where $\rho>0$ is a constant number (we have taken $\rho=1.0$ in our numerical tests). This activation function is differentiable, a property which will allow to estimate the Lipschitz constants associated to the encoder and decoder functions. The weights and biases of the hidden layers were optimised by using an Adam optimiser [126].

In the numerical tests described in Sections 2.4 and 2.5, different architectures will be considered and, for each of them, we will specify the values of the hyperparameters (numbers of hidden layers, number of neurons per layer).

2.3.2 A criterion based on stable manifold widths to determine the latent variable dimension

In this section, we consider $\mathcal{U}\subset\mathbb{R}^n$ a compact embedded submanifold of topological dimension $\overline{p}$ and we propose a criterion based on autoencoders to identify the dimension $\overline{p}$. We refer to [127] and [128] for an overview of the intrinsic dimension estimation methods, the methods aim to project the original data $\mathcal{U}\subset\mathbb{R}^n$ to a lower M-dimensional submanifold of $\mathbb{R}^n$ that $M<n$ in a way that we will not lose the information of the original data. When the minimum M-dimension is necessary to represent the observed properties in $\mathcal{U}$, M can be called intrinsic dimension.

Having in mind autoencoder applications, we are interested in nonlinear methods of approximation of $\mathcal{U}$ depending on p parameters and built on two Lipschitz mappings that correspond to the encoder map and the decoder map. Following [129], we then introduce the quantity: for $p\in\mathbb{N}$ and for given constants $\gamma>0$ and $\gamma^\dagger>0$,

$$\delta_{p,\gamma,\gamma^\dagger}(\mathcal{U})=\inf_{\Psi,\,\Psi^\dagger}\sup_{u\in\mathcal{U}}\|u-\Psi^\dagger\circ\Psi(u)\|_n$$

where the infimum is taken over the γ Lipschitz functions $\Psi:\mathcal{U}\to\mathbb{R}^p$ and the $\gamma^\dagger$ Lipschitz functions $\Psi^\dagger:\mathbb{R}^p\to\mathbb{R}^n$.

This definition is introduced in [129] under the name of *stable manifold width* with the slight nuance that the definition of [129] also involves the infimum over all the norms in $\mathbb{R}^p$. As explained in [129], the concept of stable manifold width compared to the concept of manifold width (where the functions Ψ and $\Psi^\dagger$ are only assumed to be continuous) is motivated by the fact that the Lipschitz regularity allows to explicitly control perturbations coming from noise or numerical approximation.

In the following proposition, we are interested by the case where $p<\overline{p}$. This corresponds to the case where the dimension of the latent representation space is smaller than the topological dimension of the submanifold that the autoencoder has to approximate. Under this assumption, the proposition gives a bound from below for the

stable manifold width.

Proposition 2.3.1. *Let $\mathcal{U}$ be a compact embedded submanifold of $\mathbb{R}^n$ of topological dimension $\overline{p}$. We assume that $p < \overline{p}$. Then, there exists a constant $C > 0$ which only depends on $\mathcal{U}$ such that*

$$\delta_{p,\gamma,\gamma^\dagger}(\mathcal{U}) \geq \left(\frac{C}{\gamma^\dagger\gamma}\right)^{p/(\overline{p}-p)} R \tag{2.3.3}$$

where $R > 0$ is such that $\mathcal{U} \subset B_n(0, R)$.

For $m \in \mathbb{N}^*$, we have denoted by $B_m(y, r)$ with $y \in \mathbb{R}^m$ and $r > 0$ the closed ball of center y and radius r.

Proof. The proof of this proposition will rely on a result given in [130]. This paper introduces and studies the notion of *Lipschitz width*, a concept close to *stable manifold width*.
Let ϵ be such that $\epsilon > \delta_{p,\gamma,\gamma^\dagger}(\mathcal{U})$. Then, there exist a γ Lipschitz function $\Psi : \mathcal{U} \to \mathbb{R}^p$ and a $\gamma^\dagger$ Lipschitz function $\Psi^\dagger : \mathbb{R}^p \to \mathbb{R}^n$ such that

$$\sup_{u\in\mathcal{U}} \|u - \Psi^\dagger \circ \Psi(u)\|_n \leq \epsilon.$$

Since Ψ is γ Lipschitz and $\mathcal{U} \subset B_n(0, R)$, we have $\Psi(\mathcal{U}) \subset B_p(\Psi(0), \gamma R)$. Thus, we get that

$$\inf_{\Psi_p^\dagger} \sup_{u\in\mathcal{U}} \inf_{\alpha\in B_p(\Psi(0),\gamma R)} \|u - \Psi_p^\dagger(\alpha)\|_n \leq \epsilon$$

where the infimum is taken over all the maps $\Psi_p^\dagger : (B_p(\Psi(0), \gamma R), \|\cdot\|_p) \to \mathbb{R}^n$ such that

$$\sup_{\alpha_1,\alpha_2\in B_p(\Psi(0),\gamma R)} \frac{\|\Psi_p^\dagger(\alpha_1) - \Psi_p^\dagger(\alpha_2)\|_n}{\|\alpha_1 - \alpha_2\|_p} \leq \gamma^\dagger$$

Let us now introduce the scaled norm $\|\cdot\|_{p,s}$ in $\mathbb{R}^p$ defined by $\|\alpha\|_{p,s} = \frac{1}{\gamma R}\|\alpha\|_p$. Then we have

$$\inf_{\Psi_p^\dagger} \sup_{u\in\mathcal{U}} \inf_{\alpha\in B_{p,s}(\Psi(0),1)} \|u - \Psi_p^\dagger(\alpha)\|_n \leq \epsilon$$

where the infimum is taken over all the maps $\Psi_p^\dagger : (B_{p,s}(\Psi(0), 1), \|\cdot\|_{p,s}) \to \mathbb{R}^n$ such that

$$\sup_{\alpha_1,\alpha_2\in B_{p,s}(\Psi(0),1)} \frac{\|\Psi_p^\dagger(\alpha_1) - \Psi_p^\dagger(\alpha_2)\|_n}{\|\alpha_1 - \alpha_2\|_{p,s}} \leq \gamma^\dagger\gamma R.$$

This property coincides with the fact that the fixed Lipschitz width of $\mathcal{U}$ associated to the norm $\|\cdot\|_{p,s}$ for the Lipschitz constant $\gamma^\dagger\gamma R$ is smaller than ϵ. Now, let us introduce the Lipschitz width of $\mathcal{U}$ for the Lipschitz constant $\gamma^\dagger\gamma R$ which is defined as the infimum of the fixed Lipschitz widths over all the norms in $\mathbb{R}^p$ and denote it by $d_{p,\gamma^\dagger\gamma R}(\mathcal{U})$. We thus have the following property:

$$d_{p,\gamma^\dagger\gamma R}(\mathcal{U}) < \epsilon.$$

Thus according to Proposition 3.5 in [130], this implies that we have an upper bound on

the Lipschitz constant which is given by:

$$\gamma^\dagger \gamma R \geq \frac{1}{3}\epsilon N_{2\epsilon}^{1/p}(\mathcal{U})$$

where $N_\epsilon(\mathcal{U})$ is the $\epsilon-$covering number of $\mathcal{U}$. Since $\mathcal{U} \subset B_n(0,R)$ is a compact embedded submanifold of dimension $\overline{p}$, we have that

$$N_\epsilon(\mathcal{U}) \geq C\frac{R^{\overline{p}}}{\epsilon^{\overline{p}}},$$

where C depends on the Lipschitz constants of the local maps of the finite atlas describing $\mathcal{U}$. Thus, we get that

$$\gamma^\dagger \gamma \geq C R^{\overline{p}/p-1}\epsilon^{1-\overline{p}/p}$$

which implies that

$$\epsilon \geq \left(\frac{C}{\gamma^\dagger\gamma}\right)^{p/(\overline{p}-p)} R$$

for every $\epsilon > \delta_{p,\gamma,\gamma^\dagger}(\mathcal{U})$. This property allows us to conclude the proof. □

In the following corollary, we give another formulation of the previous proposition which will be more useful for our numerical tests.

Corollary 2.3.2. *Under the same notations and hypotheses as in Proposition 2.3.1, we assume that there exist a γ Lipschitz function $\Psi : \mathcal{U} \to \mathbb{R}^p$ and a $\gamma^\dagger$ Lipschitz function $\Psi^\dagger : \mathbb{R}^p \to \mathbb{R}^n$ such that*

$$\sup_{u\in\mathcal{U}} \|u - \Psi^\dagger \circ \Psi(u)\|_n \leq \epsilon.$$

Then, there exists a constant $C > 0$ which only depends on $\mathcal{U}$ such that

$$\gamma^\dagger \gamma \geq C R^{\overline{p}/p-1}\epsilon^{1-\overline{p}/p}. \tag{2.3.4}$$

This result can be interpreted in the following way: if the dimension p is smaller than the topological dimension of $\mathcal{U}$, then getting an accurate approximation of the elements of $\mathcal{U}$ with the latent dimension p may be achieved only if the Lipschitz constants of the approximation mappings are sufficiently large. It is related to the fact that, as long as the latent dimension is smaller than the intrinsic dimension, the encoder and decoder functions have to compensate this with strong variations, in the same vein as the space-filling curves. Our criterion thus relies on the representation of the variation of the product of the Lipschitz constants of the encoder and decoder functions with respect to the latent dimension p at a given accuracy ϵ. Its relevance will be illustrated in Section 2.4.

2.3.3 Estimation of the Lipschitz constants

The criterion that we propose to estimate the intrinsic dimension relies on the inequality (2.3.4) and thus it is necessary to estimate the Lipschitz constants of the encoder Ψ and the decoder $\Psi^\dagger$. In this section, we explain how this is numerically achieved.

Since the activation function used to build these functions is the ELU, Ψ and $\Psi^\dagger$ are regular functions, in particular they are C^1 functions. Let us introduce the gradient of

the function $\Psi : \mathcal{U} \to \mathbb{R}^p$ and denote it by $\nabla\Psi$:

$$\nabla\Psi : \begin{cases} \mathcal{U} \longrightarrow \mathbb{R}^{p\times n} \\ u \longmapsto \nabla\Psi(u) \end{cases}$$

Similarly, we denote by $\nabla\Psi^\dagger$ the gradient of $\Psi^\dagger : \mathbb{R}^p \to \mathbb{R}^n$:

$$\nabla\Psi^\dagger : \begin{cases} \mathbb{R}^p \longrightarrow \mathbb{R}^{n\times p} \\ \alpha \longmapsto \nabla\Psi^\dagger(\alpha) \end{cases}$$

The estimation of the product $\gamma^\dagger\gamma$ of the Lipschitz constants relies on the following lemma whose proof uses classical arguments.

Lemma 2.3.3. *For a function $f : \mathcal{U} \subset \mathbb{R}^n \to \mathbb{R}^n$ of class C^1, we have the following estimate: for all $u^{(1)}$, $u^{(2)} \in \mathcal{U}$,*

$$\frac{\|f(u^{(1)}) - f(u^{(2)})\|_{l^{q,n}}}{\|u^{(1)} - u^{(2)}\|_{l^{q,n}}} \leq \left[\sum_{k=1}^{n}\left(\sum_{j=1}^{n}\sup_{u\in\mathcal{U}}\left|\frac{\partial f_k}{\partial u_j}(u)\right|^z\right)^{\frac{q}{z}}\right]^{\frac{1}{q}}$$

where $z, q \in]1, +\infty[$ are such that $\dfrac{1}{z} + \dfrac{1}{q} = 1$.

In particular, if we set $z = q = 2$, we get that, for all $u^{(1)}$, $u^{(2)} \in \mathcal{U}$,

$$\frac{\|f(u^{(1)}) - f(u^{(2)})\|_n}{\|u^{(1)} - u^{(2)}\|_n} \leq \left(\sum_{j,k=1}^{n}\sup_{u\in\mathcal{U}}\left|\frac{\partial f_k}{\partial u_j}(u)\right|^2\right)^{\frac{1}{2}}$$

Thus, if we apply this estimate to $f = \Psi^\dagger \circ \Psi$, we can have an approximation of a lower bound of $\gamma^\dagger\gamma$ by computing

$$\left(\sum_{j,k=1}^{n}\sum_{j=1}^{n}\max_{1\leq m\leq N}\left|\sum_{l=1}^{p}\frac{\partial\Psi_k^\dagger}{\partial\alpha_l}(\Psi(u^{(m)}))\frac{\partial\Psi_l}{\partial u_j}(u^{(m)})\right|^2\right)^{\frac{1}{2}}$$

where we have selected N samples $(u^{(m)})_{1\leq m\leq N}$ to have a discrete approximation of the supremum value of the gradient.

2.4 Numerical tests of intrinsic dimension estimations

In this section, we will present some numerical tests on the estimation of the intrinsic dimension of the data set by using an autoencoder, as described in Section 2.3. In particular, we will compare the criterion that we propose which is based on the estimation of the Lipschitz constants with a more classical criterion (we refer to [131] for a presentation of other criteria) based solely on the evolution of the accuracy with respect to the dimension of the latent dimension.

For the tests which we will present hereafter, we consider the architectures of three autoencoders, named AE1, AE2 and AE3, and described in Table 2.1, in which only half

of the architecture is written, since we consider symmetric autoencoders (the encoder and the decoder have the same architectures). The number of layers corresponds to the number of hidden layers.

Each autoencoder is trained for different choices of p, the dimension of the latent layer. Then, the product of the Lipschitz constants $\gamma^\dagger\gamma$ is evaluated following the method presented in Section 2.3.3 and the error ϵ (which corresponds to the value of the loss function given by (2.3.1)) is computed in a validation set.

Table 2.1: Architectures of the autoencoder models

	Number of layers	Number of hidden units
AE1	7	150, 80, 20, p
AE2	11	250, 150, 80, 20, 10, p
AE3	15	250, 180, 120, 80, 50, 30, 15, p

2.4.1 Tests on Van der Pol model

To build a data set from Van der Pol model (2.2.6), among the three parameters of the model, we fixed the initial velocity $x_0 = 0.5$ while the parameters v_0 and μ were randomly drawn from a uniform distribution in $[0.05, 1.5]$ and $[0.05, 2.0]$ respectively. Then, for each parameter value, we computed an approximated solution of the ODE system in $[0, T]$, with $T = 30$ thanks to the Crank-Nicolson scheme with the time step given by $\Delta t = 0.075$. A total of $N_d = 1200$ solutions were generated, of which 1000 were used in the training set and 200 in the validation set.

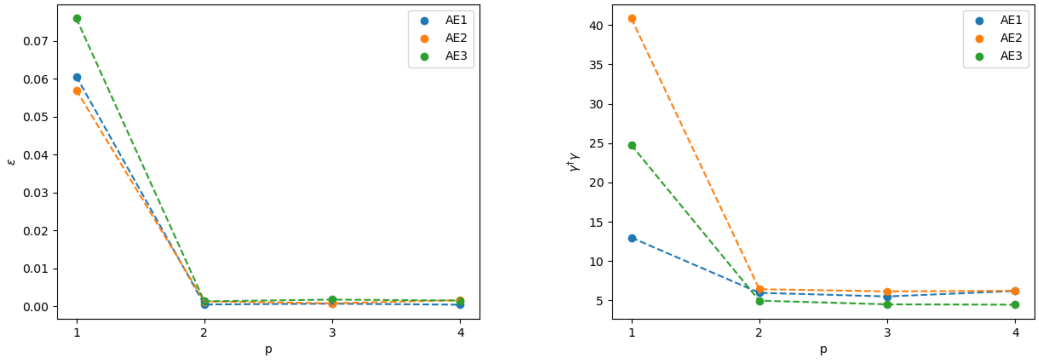
The training of the autoencoder was performed by optimising the loss function defined in equation (2.3.1) by using the Adam optimiser and by taking 1000 epochs with a batch size equal to 40. Each training for a given p was repeated 10 times and the results presented hereafter correspond to the average of these 10 tests.

In Figure 2.3(a), we have represented the variations of the error with respect to p for different architectures. As explained in [131], these curves allow to identify the intrinsic dimension which corresponds to the value of p from which the curve begins to stagnate. In Figure 2.3(b), we have represented the variations of the products of the Lipschitz constants to apply the criterion that we proposed. For both methods, we can observe a stagnation for $p = 2$. Therefore, for this simple data set, both criteria are able to identify the right intrinsic dimension $\bar{p} = 2$.

2.4.2 Tests on the FitzHugh-Nagumo model

In this section, we present the results obtained on the intrinsic dimension estimation for the solutions of the FitzHugh-Nagumo model. We consider here a scenario in which we test the dimension estimation when $\bar{p} = 5$. The numerical approximation of FitzHugh-Nagumo model is carried out by using the Crank-Nicolson scheme on a time interval $(0, T)$ with $T = 200$ and with the time step given by $\Delta t = 0.5$.

To generate the data set, we have kept I_{ext} fixed to the value $I_{ext} = 0.325$ whereas b, τ, a, v_0 and w_0 are drawn from a uniform distribution respectively in the intervals $[0.05, 0.5]$,

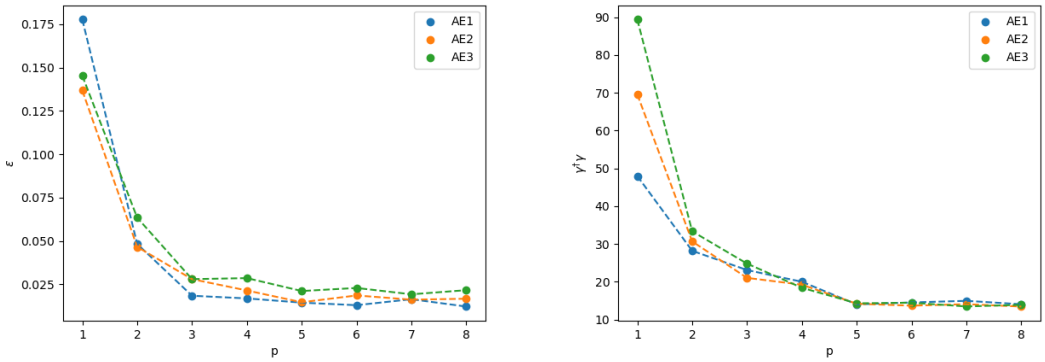


(a) ϵ against different number of p (b) $\gamma^\dagger\gamma$ against different number of p

Figure 2.3: Intrinsic dimension estimation for Van der Pol model and $\bar{p} = 2$

$[12.0, 13.0]$, $[0.05, 0.5]$, $[0.05, 1.0]$, and $[0.05, 1.0]$. So, in this case, the intrinsic dimension is $\bar{p} = 5$.

If we consider the curve of the accuracy with respect to the latent dimension p, reported in Figure 2.4(a), we observe a stagnation from $p = 3$, which would lead to a wrong conclusion. Let us mention that the difficulty to identify the latent dimension thanks to this simple and natural criterion has already been highlighted in several papers (we refer for instance to [131]). In particular, in relatively high dimension, it is often observed that the curve gradually decreases and starts to stagnate before reaching the intrinsic dimension. On the other hand, if we consider the curve of the product of the Lipschitz constants with respect to the latent dimension, reported in Figure 2.4(b), we can observe that a stagnation only occurs from $p = 5$. Thus the criterion that we propose is able to correctly identify the value of the intrinsic dimension.



(a) ϵ against different number of p (b) $\gamma^\dagger\gamma$ against different number of p

Figure 2.4: Intrinsic dimension estimation for FitzHugh-Nagumo model and $\bar{p} = 5$

2.5 Parameter identification with LVGF method

In this section, we will assess the performances of the LVGF method for the identification of parameters in Van der Pol and Fitzhugh-Nagumo models. For each numerical test, we will compare the results given by LVGF method with what we will call *the classical method*: it consists in minimising the classical misfit functional J given by (2.2.5) thanks to a gradient descent algorithm with a fixed step size.

Before describing each test, let us start with some practical details for the implementation of the LVGF and classical methods. For the training process in the preliminary step of the LVGF method, we have trained the autoencoders in 1000 epochs with 40 batch sizes. In Algorithm 1 which describes LVGF method, the maximum number of iterations has been set to $N_{max} = 2000$, the tolerance for $\|\alpha^* - \alpha_k\|_p$ has been set to $\sigma = 10^{-4}$ and the steps were given by $S = [0.0005, 0.0025, 0.0125, 0.0625]$.

To quantify the errors in the parameter estimation, we evaluated the total error by computing the root relative sum of squares (RRSSE):

$$RRSSE(\theta^*, \theta) = \frac{\|\theta^* - \theta\|_m}{\|\theta^*\|_m}$$

and, to evaluate the errors in each of the parameters, we also computed the following relative error (RE) for each parameter:

$$RE(\theta_i^*, \theta_i) = \frac{|\theta_i^* - \theta_i|}{|\theta_i^*|},\ 1 \leq i \leq m.$$

At last, the error between the observed signal u^* and the signal associated to the estimated parameters $u = \varphi(\theta)$ is defined as the root relative mean squared error (RRMSE):

$$RRMSE(u^*, u) = \frac{\|u^* - u\|_n}{\|u^*\|_n}.$$

2.5.1 Parameter identification for Van der Pol model

In this section, we consider Van der Pol model and are interested by the identification of the parameter vector $\theta = (x_0, v_0, \mu)$.

For LVGF method, we have taken the same data set of 1200 samples and the same autoencoder setup as in Section 2.4.1. In particular, in the data set, θ_1 has a valued fixed to $\theta_1 = 0.5$, whereas θ_2 and θ_3 are respectively drawn from uniform distributions in $[0.05, 1.5]$ and in $[0.05, 2.0]$. The architecture of the autoencoder corresponds to AE1 described in Table 2.1 and its latent dimension has been identified in Section 2.4.1 to be equal to $p = 2$ thanks to the criterion presented in Section 2.3.2.

In order to compare the classical method and the LVGF method, we tested them for 10 random values of θ^* taken in the same range as the data set for θ_2 and θ_3 and in the range of $[0.05, 1.5]$ for θ_1. Moreover, for each θ^*, we performed the parameter estimation methods starting from 20 different values of initial guess taken in the same range.

The results are presented in Tables 2.2, 2.3 and 2.4. Table 2.2 corresponds to the errors on the parameters and the signals averaged over the 10 different values of θ^* and the 20 different initial guesses whereas Table 2.3 details the results (still averaged over the initial guesses) obtained for each value of θ^*. To show the results more easily, we also listed

the best case and the worst case in Table 2.4. We can see that the values obtained with LVGF method are significantly closer to the right values than the classical method. With the LVGF method, both the averaged RRSSE and RRMSE are about 10 times smaller than the ones obtained by using the classical method. More specifically in Table 2.3, the LVGF method could provide us estimations with smaller standard deviation (computed for each θ over initial guesses) and RE almost for each test and each θ. Moreover, even if, for the best case, the classical method gives very accurate results, most of the time, we observe that the LVGF method could provide us estimations with smaller RE and that the classical method converges to a local minima, which prevents it from reaching the right value and leads to significant errors.

Table 2.2: Comparison between the classical method and LVGF method: averaged results on 10 tests with Van der Pol model

	RE of each estimated θ	RRSSE of the estimated θ	RRMSE of the estimated signal
classical method	(0.327, 0.66, 1.37)	1.412	0.4247
LVGF method	(0.055, 0.095, 0.01)	0.112	0.0231

Table 2.3: Comparison between the classical method and LVGF method: detailed results for 10 parameters with Van der Pol model. The results are averaged over the initial guesses.

θ^*	Methods	Average estimated θ	Standard deviation	RE of each estimated θ	RRSSE of the estimated θ	RRMSE of the estimated signal
(0.5, 0.5, 0.5)	classical	(0.37, 0.34, 1.14)	(0.18, 0.22, 0.87)	(0.26, 0.33, 1.27)	1.3378	0.6205
	LVGF	(0.52, 0.52, 0.50)	(0.01, 0.01, 0.002)	(0.05, 0.04, 0.008)	0.064	0.013
(0.5, 0.38, 0.65)	classical	(0.47, 0.01, 1.45)	(0.494, 0.987, 0.936)	(0.47, 0.97, 1.23)	1.84	0.8997
	LVGF	(0.52, 0.38, 0.64)	(0.012, 0.047, 0.045)	(0.05, 0.09, 0.03)	0.111	0.0573
(0.5, 0.6, 0.3)	classical	(0.47,0.54, 0.50)	(0.097, 0.17, 0.589)	(0.07, 0.1, 0.65)	0.6649	0.2053
	LVGF	(0.53, 0.64, 0.30)	(0.016, 0.021, 0.0009)	(0.07, 0.08, 0.006)	0.10708	0.0191
(0.5, 0.3, 0.8)	classical	(0.67, -0.05, 1.10)	(0.545, 1.12, 0.94)	(0.34, 1.18, 0.37)	1.2862	1.5
	LVGF	(0.52, 0.31, 0.80)	(0.0037, 0.0021, 0.00012)	(0.05, 0.05, 0.003)	0.0677	0.0134
(0.5, 0.8, 0.2)	classical	(0.49,0.77, 0.30)	(0.06, 0.155, 0.45)	(0.03, 0.04, 0.51)	0.5157	0.1137
	LVGF	(0.53, 0.85, 0.20)	(0.008, 0.014, 0.0003)	(0.06, 0.07, 0.005)	0.0926	0.0198
(0.5, 0.45, 0.7)	classical	(0.59, 0.17, 0.97)	(0.35, 1.10, 1.03)	(0.18, 0.63, 0.38)	0.7596	0.8561
	LVGF	(0.51, 0.46, 0.70)	(0.0088, 0.0077, 0.0017)	(0.03, 0.03, 0.005)	0.0464	0.011
(0.5, 0.25, 0.6)	classical	(0.50, -0.23, 5.63)	(1.48, 1.51, 8.94)	(1.29, 1.92, 8.38)	8.928	0.7703
	LVGF	(0.54, 0.27, 0.60)	(0.0009, 0.01, 0.006)	(0.07, 0.07, 0.005)	0.1025	0.0125
(0.5, 0.35, 0.9)	classical	(0.71, -0.08, 1.26)	(0.60, 1.24, 1.0)	(0.42, 1.23, 0.39)	1.3626	0.463
	LVGF	(0.52, 0.36, 0.90)	(0.0036, 0.003, 0.0005)	(0.03, 0.03, 0.001)	0.045	0.009
(0.5, 0.25, 0.7)	classical	(0.396, 0.194, 1.06)	(0.21, 0.11, 0.72)	(0.21, 0.22, 0.51)	0.5999	0.3387
	LVGF	(0.53, 0.35, 0.69)	(0.023, 0.35, 0.06)	(0.08, 0.42, 0.03)	0.4564	0.0694
(0.5, 0.6, 0.2)	classical	(0.5, 0.6, 0.2)	(0.0001, 0.0001, 0.0001)	(0.0003, 0.0003, 0.0001)	0.0004	0.0000832
	LVGF	(0.52, 0.63, 0.20)	(0.002, 0.02, 0.03)	(0.06, 0.07, 0.009)	0.091	0.0196

In order to better assess the performances of the methods and illustrate them, we are going to describe the results in more detail for a case which is quite representative of what is observed in general. We consider the case where the parameter to recover is given by $\theta^* = (0.5, 0.5, 0.5)$ and where we start from the initial guess $\theta = (0.6615, 1.23, 1.6994)$. The results obtained in this specific case are reported in Table 2.5. The iterations obtained with the classical method are represented in Figure 2.5 (a) where, to better visualize them, we have chosen to represent only θ_2 and θ_3. Figure 2.5 (b) represents the values taken by the misfit function J given by (2.2.5) at the successive iterations. We note that the

Table 2.4: Comparison between the classical method and LVGF method for Van der Pol model: detailed results for two specific cases (the best case and the worst case) and averaged results over 10 different parameter values. At each line, the results are averaged over 20 initial guesses.

	Methods	RE of each estimated θ	RRSSE of the estimated θ	RRMSE of the estimated signal
Best case	Classical	(0.0003, 0.0003, 0.0001)	0.0004	0.00008
	LVGF	(0.06, 0.07, 0.009)	0.091	0.0196
Worst case	Classical	(1.29, 1.92, 8.38)	8.928	0.77
	LVGF	(0.08, 0.42, 0.03)	0.46	0.07

gradient method experiences convergence issues and that the iterations stagnate to a local minimum which is far away from θ^*.

Table 2.5: Comparison between the classical method and LVGF method in a specific case for Van der Pol model: parameter to identify $\theta^* = (0.5, 0.5, 0.5)$ and initial guess $\theta = (0.6615, 1.23, 1.6994)$

	estimated θ	RRSSE of the estimated θ	RRMSE of the estimated signal
classical method	$(0.44, -0.16, 2.32)$	3.8666	1.81
LVGF method	$(0.52, 0.52, 0.502)$	0.0541	0.011

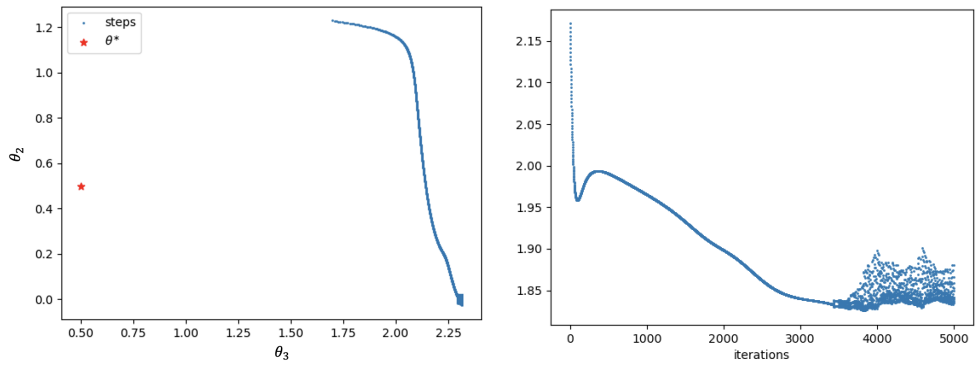
On the other hand, with LVGF method, the iterations converge to the correct values of the parameters, as illustrated in Figure 2.6. In addition, Figure 2.7 also depicts the iterations of the latent variable $\alpha = (\alpha_1, \alpha_2)$ and we observe that the descent direction for this variable allows to go in a relatively direct way towards the value α^*.

Still for this specific case, let us finally illustrate the errors on the signal presented in the last column of Table 2.5. For both methods, Figure 2.8 presents a comparison of the signal associated to the retrieved parameter with the true signal u^* associated to θ^*, the value to be identified. The signal corresponding to the initial guess (which is the same for both methods) corresponds to the green curve. We can see that the curve associated to the LVGF method perfectly fits the measured signal whereas the signal reconstructed thanks to the classical method is quite far from the measured signal.

2.5.2 Parameter identification for FitzHugh-Nagumo model

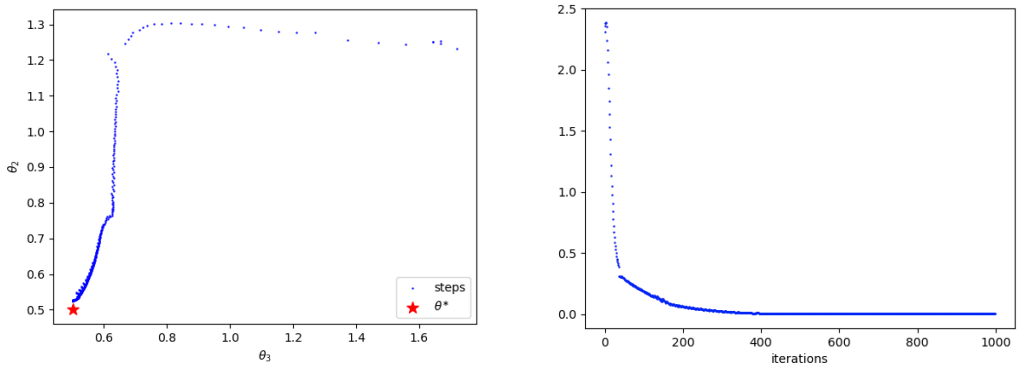
This section is devoted to a presentation of the numerical results obtained for the identification of parameters in FitzHugh-Nagumo model (2.2.7). In these tests, the values of I_{ext} and of the initial condition of v are fixed to $I_{ext} = 0.325$ and $v_0 = 1$ and we want to identify the four remaining parameters $\theta = (w_0, a, b, \tau)$.

For LVGF method, we have considered a data set of 1200 samples corresponding to a fixed value of τ given by $\tau = 12.5$ whereas the values of w_0, a and b are respectively drawn from a uniform distribution in $[0.05, 1.0]$, $[0.05, 0.8]$, and $[0.05, 0.8]$. With regard to the setup and training process of the autoencoder, we made the same choices as in the



(a) Values of θ_2 and θ_3 at each iteration (b) Values of J at each iteration

Figure 2.5: Illustration of the test presented in Table 2.5: representations of the iterations with the classical method



(a) Values of θ_2 and θ_3 at each iteration (b) Values of J at each iteration

Figure 2.6: Illustration of the test presented in Table 2.5: representations of the iterations with LVGF method

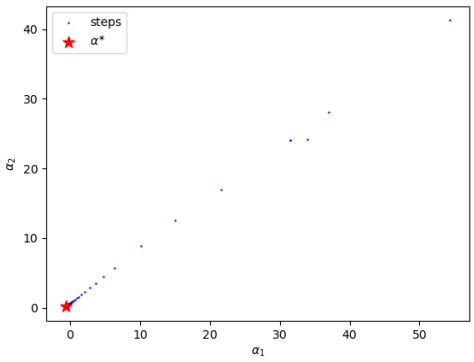
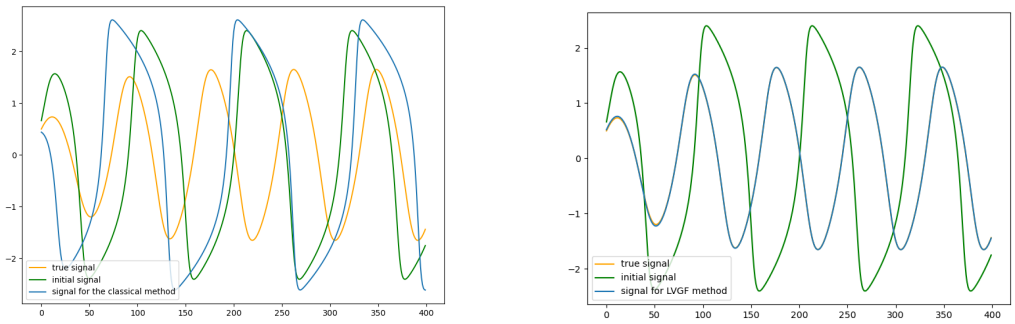


Figure 2.7: Illustration of the test presented in Table 2.5: values of α at each iteration



(a) Signal generated with the parameters obtained with the classical method

(b) Signal generated with the parameters obtained with LVGF method

Figure 2.8: Illustration of the test presented in Table 2.5: comparison of the signals with the classical method and with LVGF method (Van der Pol model)

previous section for Van der Pol model, except that the dimension of the latent layer is given by $p = 3$.

To test the performances of the LVGF method and compare them with the classical method, we repeated the same statistical test as for Van der Pol model by considering 10 random values of θ^* and 20 initial guesses for each value of θ^*. We took the parameters in the same range as the data set for w_0, a, b and in the range of $[12, 13]$ for τ.

As in the previous section, the results are given by: Table 2.6 gives the averaged results over 10 initial guesses whereas Table 2.7 details averaged results obtained from every θ^* and 2.8 corresponds to the best case and the worst case.

As for Van der Pol model, we observe that the relative error made on the parameters is smaller with LVGF method than with the classical method. More precisely, we see that the error is reduced by a factor of 2 or more and that the error on the signal is reduced by a factor of about 7. If we look at the relative error parameter by parameter, we also observe that the error on the reconstruction of the parameter b (third parameter in θ) is relatively large with both methods still with a clear improvement with LVGF method (around 54% with the classical method and 27% for LVGF method). Since this lack of accuracy has a rather low impact on the reconstruction of the signal, this difficulty to identify the parameter b compared to w_0, a and τ is related to the differences of sensitivities of the signal with respect to the parameters: its sensitivity with respect to b is smaller than its sensitivity with respect to the other parameters and so a relatively rough identification still allows to accurately reconstruct the signal.

At last, the results for a specific example corresponding to $\theta^* = (0.5, 0.5, 0.5, 12.5)$ with the initial guess $\theta = (0.4815, 0.223, 0.689, 12.52)$ are detailed in Table 2.9. In that case, the reconstruction of the signal associated to the parameters is presented in Figure 2.9. We can see that the signal generated by the parameters obtained with LVGF method perfectly fits the measured signal.

Table 2.6: Comparison between the classical method and LVGF method: averaged results on 10 tests (FitzHugh-Nagumo model)

Method	RE of each estimated θ	RRSSE of the estimated θ	RRMSE of the estimated signal
classical method	(0.174, 0.322, 0.544, 0.036)	0.703	0.3662
LVGF method	(0.074, 0.08, 0.27, 0.014)	0.3026	0.0529

Table 2.7: Comparison between the classical method and LVGF method for FitzHugh-Nagumo model: detailed results for 10 parameters. The results are averaged over the initial guesses.

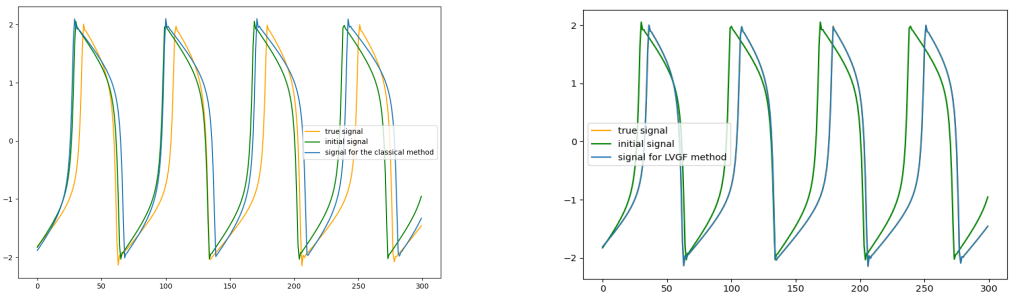
θ^*	Methods	Average estimated θ	RE each estimated θ	RRSSE of the estimated θ	RRMSE of the estimated signal
(0.5,0.5,0.5,12.5)	classical	(0.60, 0.26, 0.60, 12.97)	(0.19, 0.54, 0.20, 0.038)	0.6348	0.6045
	LVGF	(0.47, 0.48, 0.42, 12.67)	(0.052, 0.035, 0.16, 0.014)	0.1683	0.0142
(0.5,0.5,0.2,12.5)	classical	(0.66, 0.26, 0.55, 13.0)	(0.33, 0.48, 1.74, 0.04)	1.8667	0.5788
	LVGF	(0.48, 0.46, 0.199, 12.51)	(0.097, 0.12, 0.44, 0.012)	0.4905	0.096
(0.3,0.5,0.3,12.5)	classical	(0.39, 0.33, 0.45, 13.0)	(0.45, 0.35, 0.66, 0.04)	0.8804	0.4426
	LVGF	(0.28, 0.48, 0.24, 12.56)	(0.097, 0.046, 0.3, 0.0092)	0.3145	0.0161
(0.2,0.4,0.5,12.5)	classical	(0.18, 0.33, 0.39, 12.84)	(0.13, 0.18, 0.23, 0.03)	0.3352	0.188
	LVGF	(0.19, 0.38, 0.44, 12.56)	(0.13, 0.096, 0.22, 0.016)	0.2772	0.0541
(0.55,0.55,0.55,12.5)	classical	(0.68, 0.36, 0.74, 13.0)	(0.24, 0.34, 0.34, 0.04)	0.5512	0.4858
	LVGF	(0.52, 0.51, 0.44, 12.76)	(0.07, 0.08, 0.21, 0.02)	0.2467	0.0978
(0.45,0.45,0.45,12.5)	classical	(0.44, 0.33, 0.37, 13.0)	(0.034, 0.26, 0.19, 0.04)	0.3307	0.2586
	LVGF	(0.43, 0.43, 0.39, 12.60)	(0.055, 0.05, 0.2, 0.01)	0.216	0.0168
(0.7,0.55,0.35,12.5)	classical	(0.77, 0.36, 0.63, 13.0)	(0.17, 0.35, 0.88, 0.04)	0.9807	0.5015
	LVGF	(0.64, 0.50, 0.21, 12.67)	(0.1, 0.11 , 0.47, 0.019)	0.5061	0.0756
(0.75,0.4,0.4,12.5)	classical	(0.69, 0.26, 0.18, 12.99)	(0.07, 0.34, 0.56, 0.04)	0.666	0.2614
	LVGF	(0.70, 0.37, 0.35, 12.59)	(0.07, 0.1 , 0.31, 0.01)	0.3458	0.1028
(0.75, 0.35, 0.45,12.5)	classical	(0.70, 0.26, 0.28, 12.84)	(0.06, 0.24, 0.39, 0.03)	0.4757	0.1889
	LVGF	(0.74, 0.33, 0.39, 12.56)	(0.03, 0.07, 0.22, 0.01)	0.24	0.0156
(0.8,0.35,0.6,12.5)	classical	(0.76, 0.30, 0.46, 12.78)	(0.06, 0.14, 0.25, 0.02)	0.3039	0.1515
	LVGF	(0.78, 0.32, 0.51, 12.65)	(0.04, 0.09, 0.19, 0.019)	0.2205	0.04

Table 2.8: Comparison between the classical method and LVGF method for FitzHugh-Nagumo model: detailed results for two specific cases (the best case and the worst case) and averaged results over 10 different parameter values. At each line, the results are averaged over the initial guesses.

	Methods	RE of each estimated θ	RRSSE of the estimated θ	RRMSE of the estimated signal
Best case	Classical	(0.06, 0.14, 0.25, 0.02)	0.304	0.152
	LVGF	(0.052, 0.035, 0.16, 0.014)	0.168	0.0142
Worst case	Classical	(0.33, 0.48, 1.74, 0.04)	1.867	0.579
	LVGF	(0.1, 0.11 , 0.47, 0.019)	0.506	0.076

Table 2.9: Comparison between the classical method and LVGF method in a specific case for FitzHugh-Nagumo model: parameter to identify $\theta^* = (0.5, 0.5, 0.5, 12.5)$ and initial guess $\theta = (0.4815, 0.223, 0.689, 12.52)$

	Estimated θ	RRSSE of estimated θ	RRMSE of estimated signal
classical method	(0.59, 0.05, 0.59, 12.99)	0.9344	0.6695
LVGF method	(0.503, 0.502, 0.508, 12.48)	0.0203	0.00624



(a) Signal generated with the parameters obtained with the classical method

(b) Signal generated with the parameters obtained with LVGF method

Figure 2.9: Illustration of the tests presented in Table 2.9: comparison of the signals with the classical method and with LVGF method (FitzHugh-Nagumo model)

2.6 Conclusion and discussion

In this study, we have presented an innovative method called Latent Variables Gradient Flow (LVGF) which leverages the available data and the underlying dynamics of the system in order to identify parameters of the ODE model. This approach entails a dual-phase process. In the first phase, an autoencoder is trained on the data set in order to represent it in a compressed way. The description of the data set by a latent variable is then exploited in the second phase which corresponds to an optimisation method that can be described as a gradient flow for the latent variables.

For this new method, we presented numerical tests and a comparison with the classical method corresponding to a gradient descent method applied to the minimisation of the classical data misfit functional (2.2.5). A noticeable property of LVGF method is that it is able to avoid convergence towards local minimum points. In this way, LVGF method generally gives much better results than the classical method for the parameters identification problem.

In addition, we have shown that the sequence defined by the LVGF method satisfies convergence properties. These results still need to be extended to the algorithm actually implemented where there is an additional projection step. Nevertheless, our study, even if it is incomplete, allows to understand how the use of the non-linear mapping between the parameter space and the latent variable space latent variable allows to convexify our initial minimisation problem.

In a complementary manner to this parameter identification method, we also proposed a new criterion to identify the intrinsic dimension of a sub-manifold thanks

to autoencoders. This criterion is justified by a theoretical result based on the notion of stable manifold width. It describes the behaviour of the product of the Lipschitz constants of the encoder and decoder when the latent variable is too small. We have compared our criterion to the classical criterion based on the evolution of the error with respect to the latent dimension and observed that it leads to better results. Indeed, in the case of the classical criterion, the error gradually decreases when the latent dimension increases, making it difficult to pick a single value for the dimension. On the contrary, our criterion clearly highlights two phases in the evolution of the product of the Lipschitz constants and the intrinsic dimension corresponds to the elbow of the curve.

In addition to the theoretical study which must be in-depth, this work opens a certain number of perspectives. First of all, LVGF method has been tested on simple dynamical systems to start assessing its basic properties. We intend to continue exploring its use in more complex situations corresponding to larger dynamical systems that involve a large number of parameters.

At last, the autoencoder used to approximate the available population data is based on neural networks. This is not a necessary choice, and other manifold learning methods could be used ([132]). For instance, Principal Component Analysis (PCA), kernel-PCA, Locally Linear Embedding, Isomap, Laplacian Eigenmaps, Semidefinite Embedding are all methods which could be used in order to build an encoder-decoder pair.

Acknowledgements

We would like to thank the support from INSPIRE European Training Network which receives funding from the EU Horizon 2020 Research and Innovation programme, under the Marie Skłodowska-Curie GA 858070 [1]. We also appreciated all the help from Sylvain Bernasconi and Christophe Bleunven at NOTOCORD, an Instem company.

Appendix A

Proof of Proposition 2.2.1 (identifiability and observability for Van der Pol model)

We consider two solutions of system (2.2.6) $(x^{(1)}, v^{(1)})$ and $(x^{(2)}, v^{(2)})$ respectively associated to the set of initial conditions and parameters $X^{(1)} = (x_0^{(1)}, v_0^{(1)}, \mu^{(1)})$ and $X^{(2)} = (x_0^{(2)}, v_0^{(2)}\mu^{(2)})$. We assume that the measurements on these two solutions coincide, that is $x^{(1)} = x^{(2)}$ in $[0, T]$ and we want to prove that $X^{(1)} = X^{(2)}$.

First, by assumption, we immediately have that $x_0^{(1)} = x_0^{(2)}$. Next, according to the first equation of (2.2.6), we get that $v^{(1)} = v^{(2)}$ in $[0, T]$, so in particular $v_0^{(1)} = v_0^{(2)}$. At last, since $\dot{v}^{(1)} = \dot{v}^{(2)}$ in $[0, T]$, the second equation of (2.2.6) taken at $t = 0$ gives that

$$\mu^{(1)}(1 - (x_0^{(1)})^2)v_0^{(1)} = \mu^{(2)}(1 - (x_0^{(2)})^2)v_0^{(2)}.$$

Thus, for almost all $X^{(1)}$ and $X^{(2)}$, we deduce that $\mu^{(1)} = \mu^{(2)}$ and we conclude that $X^{(1)} = X^{(2)}$.

Proof of Proposition 2.2.2 (identifiability and observability for FitzHugh-Nagumo model)

We consider two solutions of system (2.2.7) $(v^{(1)}, w^{(1)})$ and $(v^{(2)}, w^{(2)})$ respectively associated to the set of initial conditions and parameters $X^{(1)} = (v_0^{(1)}, w_0^{(1)}, a^{(1)}, b^{(1)}, \tau^{(1)})$ and $X^{(2)} = (v_0^{(2)}, w_0^{(2)}, a^{(2)}, b^{(2)}, \tau^{(2)})$. We assume that the measurements on these two solutions coincide, that is $v^{(1)} = v^{(2)}$ in $[0, T]$ and we want to prove that $X^{(1)} = X^{(2)}$.

First, by assumption, we immediately have that $v_0^{(1)} = v_0^{(2)}$. Next, according to the first equation of (2.2.7), we get that $w^{(1)} = w^{(2)}$ in $[0, T]$, so in particular $w_0^{(1)} = w_0^{(2)}$. Using the second equation of (2.2.7), we have in $[0, T]$

$$\frac{1}{\tau^{(1)}}(v^{(1)} + a^{(1)} - b^{(1)}w^{(1)}) = \frac{1}{\tau^{(2)}}(v^{(2)} + a^{(2)} - b^{(2)}w^{(2)}) \tag{A.0.1}$$

and

$$\frac{1}{\tau^{(1)}}(\dot{v}^{(1)} - b^{(1)}\dot{w}^{(1)}) = \frac{1}{\tau^{(2)}}(\dot{v}^{(2)} - b^{(2)}\dot{w}^{(2)}).$$

This implies that

$$\left(\frac{1}{\tau^{(1)}}-\frac{1}{\tau^{(2)}}\right)\dot{v}^{(1)}-\left(\frac{b^{(1)}}{\tau^{(1)}}-\frac{b^{(2)}}{\tau^{(2)}}\right)\dot{w}^{(1)}=0. \tag{A.0.2}$$

Let us prove that $\dot{v}^{(1)}$ and $\dot{w}^{(1)}$ are linearly independent. If it does not hold, there exists $\lambda\in\mathbb{R}^*$ such that $\dot{w}^{(1)}=\lambda\dot{v}^{(1)}$. Using system (2.2.7), this implies that, for all $t\in[0,T]$

$$\frac{1}{\tau^{(1)}}(v^{(1)}+a^{(1)}-b^{(1)}w^{(1)})=\lambda\left(v^{(1)}-\frac{(v^{(1)})^3}{3}-w^{(1)}+I_{ext}\right)$$

Differentiating this identity, we get that, for all $t\in[0,T]$

$$\frac{1}{\tau^{(1)}}(\dot{v}^{(1)}-b^{(1)}\dot{w}^{(1)})=\lambda\left(\dot{v}^{(1)}-(v^{(1)})^2\dot{v}^{(1)}-\dot{w}^{(1)}\right)$$

Replacing $\dot{w}^{(1)}$ by $\lambda\dot{v}^{(1)}$, we get

$$\dot{v}^{(1)}\left[\lambda\big(1-(v^{(1)})^2-\lambda\big)-\frac{1}{\tau^{(1)}}(1-b^{(1)}\lambda)\right]=0.$$

So this implies that, for all $t\in[0,T]$, either $\dot{v}^{(1)}(t)=0$ or $(v^{(1)})^2(t)$ is given by a constant. Using that $v^{(1)}$ is a continuous function, this implies that it is a constant function in $[0,T]$ and since $\dot{w}^{(1)}=\lambda\dot{v}^{(1)}$, $w^{(1)}$ is also a constant function in $[0,T]$.

So, if $v^{(1)}$ and $w^{(1)}$ are not constant functions, we get that $\dot{v}^{(1)}$ and $\dot{w}^{(1)}$ are linearly independent and we deduce from (A.0.2) that

$$\left(\frac{1}{\tau^{(1)}}-\frac{1}{\tau^{(2)}}\right)=0 \text{ and } \left(\frac{b^{(1)}}{\tau^{(1)}}-\frac{b^{(2)}}{\tau^{(2)}}\right)=0.$$

By this way, we get that $b^{(1)}=b^{(2)}$ and $\tau^{(1)}=\tau^{(2)}$. At last, using (A.0.1), we deduce that $a^{(1)}=a^{(2)}$. So, if $v^{(1)}$ and $w^{(1)}$ are not constant functions (which holds for almost all $X^{(1)}$), we have obtained that $X^{(1)}=X^{(2)}$.

Proof for Lemma 3.3: estimation of the Lipschitz constants

Let $f:u\in\mathbb{R}^n\to\Psi^\dagger\circ\Psi(u)\in\mathbb{R}^n$ be a continuously differentiable function $f\in C^1(\mathbb{R}^n)$. The Lipschitz constant can be bounded by:

$$|f_k(u^{(1)})-f_k(u^{(2)})|\leq\sum_{j=1}^{n}\sup_{u\in\mathcal{U}}|\frac{\partial f_k}{\partial u_j}||u_j^{(1)}-u_j^{(2)}|,\quad 1\leq k\leq n.$$

Let us introduce a vector $v\in\mathbb{R}^n$ whose components are defined as:

$$v_k=\sum_{j=1}^{n}\sup_{u\in\mathcal{U}}|\frac{\partial f_k}{\partial u_j}||u_j^{(1)}-u_j^{(2)}|,\quad 1\leq k\leq n.$$

We can estimate the ℓ^q norm of the vector v, we get:

$$\sum_{k=1}^{n}\left(\sum_{j=1}^{n}\sup_{u\in\mathcal{U}}|\frac{\partial f_k}{\partial u_j}||u_j^{(1)}-u_j^{(2)}|\right)^q$$

Considering Hölder's inequality where $z,q\in[1,\infty]$ with $\frac{1}{z}+\frac{1}{q}=1$,

$$\sum_{k=1}^{n}\left(\sum_{j=1}^{n}\sup_{u\in\mathcal{U}}|\frac{\partial f_k}{\partial u_j}||u_j^{(1)}-u_j^{(2)}|\right)^q\leq\sum_{k=1}^{n}\left(\left(\sum_{j=1}^{n}\sup_{u\in\mathcal{U}}|\frac{\partial f_k}{\partial u_j}|^p\right)^{1/z}\left(\sum_{j=1}^{n}|u_j^{(1)}-u_j^{(2)}|^q\right)^{1/q}\right)^q$$

If we divide on both sides by $\|u_j^{(1)}-u_j^{(2)}\|_{l^{q,n}}^q$. We get

$$\frac{\|f(u^{(1)})-f(u^{(2)})\|_{l^{q,n}}^q}{\|u_j^{(1)}-u_j^{(2)}\|_{l^{q,n}}^q}\leq\frac{\sum_{k=1}^{n}\left(\left(\sum_{j=1}^{n}\sup_{u\in\mathcal{U}}|\frac{\partial f_k}{\partial u_j}|^z\right)^{1/z}\left(\sum_{j=1}^{n}|u_j^{(1)}-u_j^{(2)}|^q\right)^{\frac{1}{q}}\right)^q}{\sum_{k=1}^{n}|u_j^{(1)}-u_j^{(2)}|^q}$$

To rewrite it, we get:

$$\frac{\|f(u^{(1)})-f(u^{(2)})\|_{l^{q,n}}}{\|u_j^{(1)}-u_j^{(2)}\|_{l^{q,n}}}\leq\left[\sum_{k=1}^{n}\left(\sum_{j=1}^{n}\sup_{u\in\mathcal{U}}|\frac{\partial f_k}{\partial u_j}|^z\right)^{\frac{q}{z}}\right]^{\frac{1}{q}}$$

If we set $z=q=2$, we will get

$$\frac{\|f(u^{(1)})-f(u^{(2)})\|_{l^{q,n}}}{\|u_j^{(1)}-u_j^{(2)}\|_{l^{q,n}}}\leq\left[\sum_{k=1}^{n}\left(\sum_{j=1}^{n}\sup_{u\in\mathcal{U}}|\frac{\partial f_k}{\partial u_j}|^2\right)\right]^{\frac{1}{2}}$$

In this way, we get a method to estimate $\gamma^\dagger\gamma$. Now, we consider the function of autoencoder $\Psi^\dagger$ and Ψ to this computation, we can compute $\frac{\partial f_k}{\partial u_j}$ by computing $\sum_{l=1}^{p}\frac{\partial\Psi_k^\dagger}{\partial\alpha_l}\frac{\partial\Psi_l}{\partial u_j}$. Then, we will estimate the lower bounds of $\gamma^\dagger\gamma$ by computing

$$\left[\sum_{k=1}^{n}\left(\sum_{j=1}^{n}\max_{1\leq m\leq N}|\sum_{l=1}^{p}\frac{\partial\Psi_k^\dagger}{\partial\alpha_l}(\Psi(u^m))\frac{\partial\Psi_l}{\partial u_j}(u^m)|^2\right)\right]^{\frac{1}{2}}$$

In this case, we use the max among N samples the discrete evaluation point of autoencoder to estimate the sup on the manifold $\mathcal{U}$.

Chapter 3

Artificial Neural Network Comparison on hERG Channel Blockade Detection

The first five sections of this chapter correspond to an article that has been published in International Journal of Computer Applications (Vol.184, No.14). The article is entitled: *Artificial Neural Network Comparison on hERG Channel Blockade Detection* and is a collaboration with Tessa De Korte, Sylvain Bernasconi, Christophe Bleunven, Damiano Lombardi and Muriel Boulakia.

Abstract

This chapter presents a comparison of several Artificial Neural Network methods for classification problems related to cardiac safety assessment. Given the extracellular field potential recorded by means of micro-electrode arrays, the first aim is to determine whether a chemical drug is altering the electrical activity of cardiomyocytes by disrupting the normal behaviour of the human ether-a-go-go-related gene (hERG) channels. In this chapter, we considered four different Neural Network methods and compared them in terms of accuracy and computational costs. The results indicate that, among the tested architectures, the MLP and multivariate 1-dimensional Convolutional Neural Network (1D-CNN) give the most promising results. The second aim is to classify different drug concentrations and different channel blockades. Then, we further tested multivariate 1D-CNN to perform multiclass classification tasks to classify drug concentrations and different channel blockades. Finally, we investigated RNN methods used in anomaly detection as a perspective task.

3.1 Introduction

We recall some general information about our research purpose in this chapter. According to [133], cardiotoxicity has become one of the major causes of drug discontinuation in preclinical and clinical drug development. More specifically, as presented in the studies [134, 135], several non-cardiovascular drugs were withdrawn from clinical use from 1990 to 2001 because they were associated with QT interval prolongation by blocking ion channels. Therefore, it is crucial for the pharmaceutical industry to develop effective methods to study and identify the cardiotoxicity risk in

drug development at an early stage. To do so, measuring the electrophysiology of human-induced pluripotent stem cell-derived cardiomyocytes (hiPSC-CMs) using multi-electrodes arrays (MEA) is a very promising technology that sets up high-throughput drug screening methods. Specifically, assessing drug effects on the hERG channel [12] activity is an important part of the cardiac safety risk assessment as the main cause of QT interval prolongation. hERG channel is primarily responsible for conducting the delayed rectifier potassium current (IKr), which is important for cardiac repolarisation. Inhibiting the hERG channel can potentially induce ventricular arrhythmia, torsades de pointes, and sudden death, causing dangerous side effects for patients. Moreover, regulatory agencies, including the U.S. Food and Drug Administration (FDA) and the European Medicines Agency (EMA), emphasise the evaluation of potential hERG channel blockade during drug development. Drugs with a substantial risk of hERG channel blockade may encounter regulatory restrictions or necessitate additional monitoring [136].

In this chapter, our first aim is to evaluate the impact of a drug on the hERG potassium channel by using Artificial Neural Network (ANN) methods on the field potential (FP) recordings of hiPSC-CMs obtained with MEA technology. In particular, given the electrograms recorded by MEA after adding a drug to a mono-layer of hiPSC-CMs, we explored if the drug can be classified as a hERG potassium blocker. By investigating different ANNs, this work expects to find the most suitable and effective methods to automatically identify which drugs cause hERG potassium channel blockade.

This chapter is a continuation of the work performed by [137] who used a greedy classifier optimisation strategy applied to combining *in vitro* and *in silico* experiment data to predict pro-arrhythmic risk caused by drugs. Our objective is to test Neural Networks that are known to be particularly effective and do not generally require a pre-processing step to extract features.

The second aim is to help improve the cardiac safety assessments by determining 1) if a given FP recording can be associated with a specific drug concentration or classified as a specific channel blocker, 2) if abnormal/low quality recordings can be identified in a large amount of FP recordings. These two questions and our proposed methods will be covered in more details in the last two sections.

3.1.1 State-of-the-art/Literature reviews

The articles [96]-[99] presented several Neural Network methods that are able to improve the accuracy of different learning tasks for electrocardiogram (ECG, a test that records electrical signals in the heart) analysis. Among the proposed methods, MLP and CNN are the most popular Neural Network methods that have been widely used for classification and prediction purposes in different domains, including ECG arrhythmia classification. Neural Networks have also been widely tested in electroencephalography (EEG, a test that measures electrical activity in the brain) research like brain computer interfaces, sleep analysis and seizure detection. Especially, CNN methods are used to detect and diagnose seizures based on EEG signals ([100, 101]).

In their studies, [103] and [102] employed CNN to classify echocardiograms (a test that uses high frequency sound waves to generate pictures of your heart which can check the structure and function of the heart), aiming to develop an automated diagnostic tool for clinical use. Furthermore, the work of [91] introduced a multi-label neural network that employs feature extraction techniques on mechanical beating signals of cardiomyocytes.

This network is designed to classify various drugs and drug concentrations by leveraging the distinctive features found in individual cardiomyocyte beating patterns.

3.1.2 Structure of this chapter

In this chapter, Section 3.2 presents the experimental data set that we have considered and the data pre-processing needed for MLP, one of the tested ANN methods. Then, Section 3.3 details the type of input and the architecture for the tested ANN methods: MLP, Univariate 1D-CNN, Multivariate 1D-CNN and 2D-CNN. In Sections 3.4 and 3.5 we present the design and results of the classification tasks given by these four ANN methods and we compare them in terms of performances, data processing costs and network training costs. At the end of the chapter, we will also present some further studies regarding multiclass classification and semi-supervised learning methods to detect anomalies in Section 3.6 and 3.7.

3.2 Presentation of the data set and pre-processing

3.2.1 Experimental data set

This study has considered the same experimental data and setup as in the work of [137]. The experimental data are FP recordings of hiPSC-CMs obtained with MEA technology. MEA includes a two-dimensional arrangement of micro-electrodes that can monitor the extracellular electrical activity of the cultured cells (we refer to [138] for an overview on the background of MEA measurements of hiPSC-CM). More specifically, the experimental data correspond to FP signals before and after addition of 12 drugs, in 96-well MEA plates, each well containing 8 recording electrodes. Each drug has been added at four concentrations to four different wells using five replicates per concentration.

As mentioned in [137], when a drug is added to the hiPSC-CM, the FP can be altered. For instance, the drug can have some impact on certain ion channels of the cells: it can reduce the original amplitude or prolong the duration of each cardiac beat. However, there are big variabilities in the whole recording and each drug has a specific effect. Using basic statistical tools to analyse those signals often faces difficulties and gives biased results. In this work, ANNs have been tested to have a more comprehensive and effective analysis on FP data set.

Table 3.1 lists the drugs included in the data set and indicates the hERG potassium (K), Calcium (Ca) and Sodium (Na) IC50 values and concentrations that have been tested for each drug. The last column of Table 3.1 summarizes the known effects of the drugs by classifying them as K, Ca or Na blockers or mixed blockers. A tested drug with a predominant impact on K channel is considered as a K blocker. Based on this information, 11 drugs can be considered as pure or mixed K blockers. Only Diltiazem has a K IC50 value that is much higher than the Ca IC50 and higher than the top tested concentration and has been classified as a Ca blocker.

Each cardiac beat has been extracted from the recording sequences coming from each electrode. Since each beat may have a different length depending on the variability of the hiPSC-CMs and the recording conditions, all cardiac beats have been resampled using 0.08 ms time step to normalize each beat duration to 885.6 ms (11070 samples). For more

Table 3.1: Experimental data information

Drug	IC50 (μM)			Concentration (μM)				Type of blocker
	K	**Ca**	**Na**	**#1**	**#2**	**#3**	**#4**	
Loratadine	6.1	11.4	28.9	0.001	0.003	0.001	0.03	K, Ca
Ibutilide	0.018	62.5	42.5	0.0001	0.001	0.01	0.1	K
Droperidol	0.06	7.6	22.7	0.03	0.1	0.32	1.0	K
Mexiletine	62.2	125	38	0.1	1.0	10	100	Na , K
Dofetilide	0.03	26.7	162.1	0.0003	0.001	0.003	0.01	K
Diltazem	13.2	0.76	22.4	0.01	0.1	1.0	10	Ca
Chlorpromazine	1.5	3.4	3	0.1	0.3	0.95	3	K , Ca , Na
Clozapine	2.3	3.6	3	0.1	0.3	0.95	3	K , Ca
Clarithromycine	32.9	>30	NA	0.1	1	10	100	K
Cisapride	0.02	11.8	337	0.003	0.01	0.03	0.1	K
Bepridil	0.16	1.0	2.3	0.01	0.1	1	10	K , Ca , Na
Azimilide	<1	17.8	19	0.01	0.1	1	10	K , Ca , Na

The information contained in this table comes from [137] and the references therein.

details on the experimental data and setup, we refer to the work [137] which considered the same data set.

For each beat, an acceptable signal must contain both depolarisation and repolarisation phases. Among all the collected cardiac beats, some signals may have an altered depolarisation or repolarisation phase compared to the majority of the signals and these abnormal signals have been removed from the data set. The abnormal signals will be used in later studies to investigate a method to detect anomalies.

In the experiments, the drugs have been added after a few minutes of baseline recording so that each electrode provides two recordings: the time period corresponding to the baseline recording (prior to drug addition) is denoted by $\mathfrak{P}_1$, and the time period corresponding to the post-addition recording (after drug addition) is denoted by $\mathfrak{P}_2$. To construct the training set, all the beats in the period $\mathfrak{P}_1$ and the beats obtained from the experiments of Diltiazem in the period $\mathfrak{P}_2$ will be labelled as non-K blocker (this class is denoted by NO-K-blocker). On the other hand, the beats obtained when one of the 11 K blockers was added to the experiments in the period $\mathfrak{P}_2$ will be labelled as K blocker (this class is denoted by K-blocker).

3.2.2 Pre-processing for MLP method

MLP method, in contrast to CNN methods that can directly use the raw signal, requires a pre-processing step to extract some markers or features from the signal that will be used as input. This data pre-processing step represents an additional computational cost but it can significantly reduce the input size compared to the raw signals and, by this way, it can substantially speed up the training phase. To facilitate the feature extraction process, we have separated the signal into two phases: each beat of the signal has been split into a depolarisation phase with a duration of 25.6 ms, and a repolarisation phase with a duration of 860 ms.

A Gaussian filtering, following [139] has been used to mitigate the impact of signal noise and make the computation of the features more accurate. Since the signal to noise ratio for the repolarisation phase of the signal is larger, we chose a Gaussian kernel with a larger standard deviation for the repolarisation phase (its value is equal to 40) than for

the depolarisation phase (its value is equal to 2.5).

To summarize, the pre-processing step has extracted 64 features from each beat, distributed into 23 features for the depolarisation phase, 37 features for the repolarisation phase, and 4 features for the entire signal. The features are listed in Tables 3.2 and 3.3 and some of them are displayed in Figure 3.1 and 3.2. The following notations are used in the tables:

- the covariance matrix used to compute Maximum Eigenvalue for depolarisation phase (DEV) and Maximum Eigenvalue for repolarisation phase (REV) is the matrix given by: for all $1 \leq i, j \leq 8$

$$C_{ij} = \frac{1}{N_t} \sum_{k=1}^{N_t} (e_i^{(k)} - \bar{e}_i)(e_j^{(k)} - \bar{e}_j)^T$$

  where N_t corresponds to the number of time steps, $e_i^{(k)} \in \mathbb{R}$ is the FP signal recorded by the i-th electrode and $\bar{e}_i$ is the mean value of e_i

- Dw is the filtered signal restricted to the depolarisation phase (that corresponds to the time interval delimited by DD in Figure 3.1) whereas Rw is the part of the filtered signal restricted to the repolarisation wave. Dw is defined on $[0, t_D]$ and Rw is defined on $[{t_R}^1, {t_R}^2]$.

- A_D (resp. A_R) is the area under the curve of Dw (resp. of Rw):

$$A_D = \Big| \int_0^{t_D} Dw(t)dt \Big| \text{ and } A_R = \Big| \int_{{t_R}^1}^{{t_R}^2} Rw(t)dt \Big|$$

The third and sixth columns of Table 3.2 and the third column of Table 3.3 give the feature indexes. The DEV and REV correspond to one scalar feature. Regarding the other features listed in the tables, they have been extracted from each signal and, in a given well and at a fixed beat. Then, we computed the average, maximum, minimum, and standard deviation of the features over the different electrodes of the well. Then, these four values were stored in the entries. In addition, Duration DD and Arrival time at the centre DCT minimum values have been removed considering they have too extreme values when they are computed on a large number of beats.

Due to the fact that the orders of magnitude of the features widely vary, the features have been rescaled to ensure that the statistical distribution of the input data is roughly in the same range. Since our goal is to detect the impact of drugs on the signal, the idea is to rescale the features extracted from beats corresponding to period $\mathfrak{P}_1$ or $\mathfrak{P}_2$ by features corresponding to period $\mathfrak{P}_1$.

More precisely, a feature coming from one beat taken in period $\mathfrak{P}_1$ is rescaled by dividing the similar feature computed from another beat in period $\mathfrak{P}_1$ in the same well. The arrays containing these rescaled features are labelled as NO-K-blocker. On the other hand, a feature coming from one beat taken in period $\mathfrak{P}_2$ is rescaled by dividing the similar feature computed from another beat in period $\mathfrak{P}_1$ in the same well. An array containing these features computed from beats extracted from Diltiazem experiments is labeled as NO-K-blocker whereas an array containing these features computed from beats extracted from K-blocker experiments will be labeled as K-blocker.

Table 3.2: Features for the depolarisation and repolarisation signals

depolarisation Phase			repolarisation Phase		
Features Name	**Methodology**	**Index of features**	**Features Name**	**Methodology**	**Index of features**
Maximum Eigenvalue (DEV)	Maximum eigenvalue of the covariance matrix C	1	Maximum Eigenvalue (REV)	Maximum eigenvalue of the covariance matrix C	24
Amplitude (DA)	$DA = \max(Dw) - \min(Dw)$	2, 3, 4, 5	Amplitude (RA)	$RA = \max(Rw) - \min(Rw)$	25, 26, 27, 28
Duration (DD)	Duration of Dw	6, 7, 8			
Amplitude in the Center (DC)	Value of Dw at the time where the area under the curve reaches $0.5A_D$	17, 18, 19, 20	Amplitude in the Center (RC)	Value of Rw at the time where the area under the curve reaches $0.5A_R$	49, 50, 51, 52
Maximum Slope ($Dsmax$)	$Dsmax = \max_{[0,t_D]} Dw'$	9, 10, 11, 12	Maximum Slope ($Rsmax$)	$Rsmax = \max_{[0,t_R]} Rw'$	33, 34, 35, 36
Minimum Slope ($Dsmin$)	$Dsmin = \min_{[0,t_D]}$Dw'	13, 14, 15, 16	Minimum Slope ($Rsmin$)	$Rsmin = \min_{[0,t_R]} Rw'$	37, 38, 39, 40
Arrival Time at the Center (DCT)	Time where the area under the curve reaches $0.5 \times A_D$	21, 22, 23	Arrival Time for Maximum Amplitude (RCT)	Time when Rw reaches its maximum value	29, 30, 31, 32
			25% of the area under the curve (RCT0.25)	Time where the area under the curve reaches $0.25A_R$	41, 42, 43, 44
			50% of the area under the curve (RCT0.5)	Time where the area under the curve reaches $0.5A_R$	45, 46, 47, 48
			75% of the area under the curve (RCT0.75)	Time where the area under the curve reaches $0.75A_R$	53, 54, 55, 56
			90% of the area under the curve (RCT0.9)	Time where the area under the curve reaches $0.9A_R$	57, 58, 59, 60

Table 3.3: Features for the whole signal

Features Name	**Methodology**	**Index of features**
Field potential duration (FPD)	The duration from beginning of depolarisation wave to the end of the repolarisation wave	61, 62, 63, 64

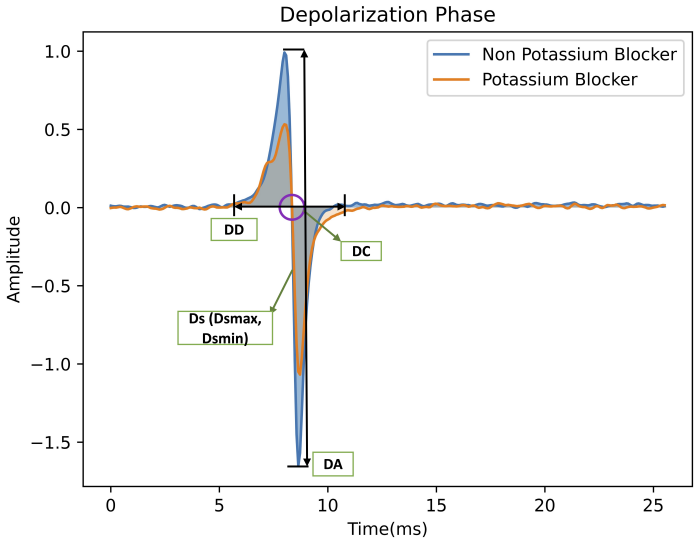


Figure 3.1: A selection of depolarisation phase features

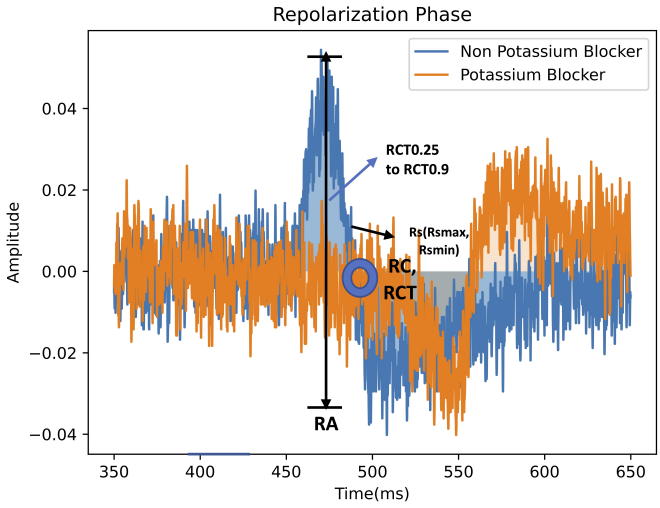


Figure 3.2: A selection of repolarisation phase features

3.3 Artificial Neural Networks methods

This part of the chapter focuses on the classification between K or non-K blockers of the drugs listed in Table 3.1. Four types of ANNs have been tested: MLP, Univariate 1D-CNN, Multivariate 1D-CNN and 2D-CNN. In this section, the architecture of these ANN methods are explained.

3.3.1 Techniques used in ANN

Before presenting the different ANNs, the important techniques that have been used are listed. We start by detailing the activation functions:

1. Since Rectifier Linear Unit (ReLU) has been used to improve Boltzmann Machines in [140], ReLU becomes a commonly used activation function [141]:

$$ReLU(x) = max(x, 0). \tag{3.3.1}$$

2. The leaky ReLU function has been introduced by [142] and it is a variant of the ReLU function whose expression is given by:

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ ax & \text{otherwise} \end{cases} \tag{3.3.2}$$

where a is a small constant number (we have taken $a = 0.3$ in our tests).

3. The sigmoid activation function [141, 143] transforms the input into an output that lies in the interval $(0, 1)$ as follows:

$$\chi(x) = \frac{1}{1 + \exp(-x)} \tag{3.3.3}$$

The Batch Normalisation operation is used to normalise and stabilise the distributions of the input layers, considering the study of [144]. The formula is the following ([145]):

$$\bar{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{3.3.4}$$

This equation normalises the input elements x_i (which are the outputs from the previous activation layer) by calculating the mean μ_B and variance σ_B^2 over all the samples. Here, the positive constant ϵ prevents the calculation from being invalid when the variance is very small or equal to zero.

3.3.2 MLP

The first tested neural network is MLP. Since MLP was first proposed in [146], it has become a commonly used method that consists of sets of fully connected layers. The work of [147] proposed a method to detect arrhythmia in ECG using MLP.

In MLP method, if there are $L \in \mathbb{N}^*$ hidden layers and that, for each $1 \leq l \leq L$, the l-th layer has $k^{(l)} \in \mathbb{N}^*$ units. The input is denoted by $x \in \mathbb{R}^d$ (where $d \in \mathbb{N}^*$ corresponds to the number of features), the first hidden layer output is a set of $k^{(1)}$ values given by:

$$o_i = \phi\Big(\sum_{j=1}^{d} \mathfrak{w}_{ij} x_j + \mathfrak{b}_i\Big),\ 1 \leq i \leq k^{(1)}, \tag{3.3.5}$$

where $\phi(.)$ is the activation function. Then, the output of the first hidden layer will be the input of the second hidden layer, the output of the second hidden layer will be the input of the third hidden layer, until the last hidden layer (see [126, Chapter 4, section 1.1], [148]).

The architecture of the MLP network is shown in Figure 3.3 (we refer to [149, Chapter 6], for the definitions of the technical terms that follow). In the first layer which corresponds to the input layer, 64 features are fed into the network. A batch-load [126, Chapter 11, section 5] has been used and each load will have 40 sets of 64 features propagated through the network. The weights for the first hidden layer are initialized with random normal distributed numbers. The bias for the first layer is initialized to zero. The output from the first hidden layer is then rendered to the next 6 sets of fully connected layers. In the 6 sets of fully connected layers, they have 320, 320, 192, 64, 32, and 10 hidden units in the layers and the neurons are activated by utilizing the ReLU activation function (3.3.1) in each hidden layer considering the features were rescaled in Section 3.2. The output from the last hidden layer will be passed to the fully connected output layer of 1 neuron with the sigmoid activation function (3.3.3) to provide a prediction for the binary classification.

During the training phase of the network, the predictions from the MLP method will be compared with the actual labels in order to compute the loss for each training. The Binary Cross Entropy has been chosen as the loss function. The weights of hidden layers were computed by using an Adam optimiser [126, Chapter 11, section 7].

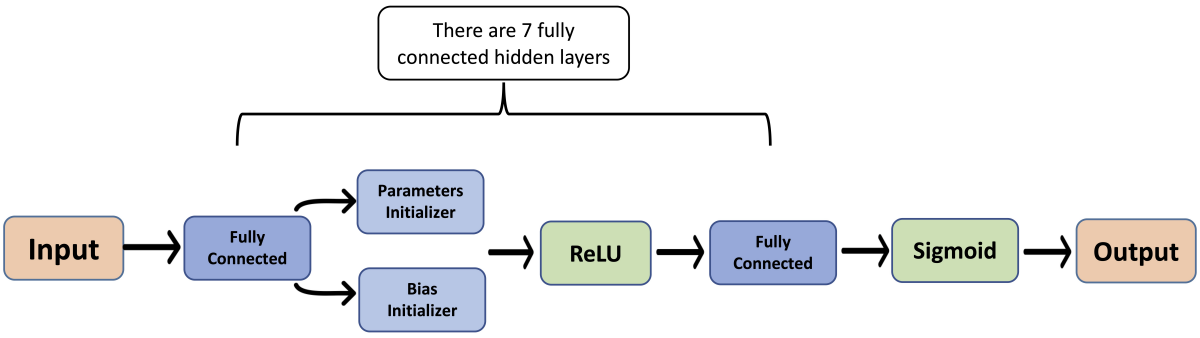


Figure 3.3: Architecture of the MLP model

3.3.3 1D-CNN

When CNN was first proposed in [150], it was named a self-organised neural network. After many improvements and extensions [151], CNN has become a neural network commonly applied to image analysis. While 1D-CNN is very often used for the classification of time series data, 2D-CNN is a leading machine learning approach for image classification. Compared to MLP, the CNN method consists of convolutional layers and works as a feature extractor, so it does not require heavy data pre-processing.

Pre-processing for 1D-CNN

Compared to the MLP method, the data pre-processing for 1D-CNN is much simpler. In this work, the univariate 1D-CNN and the multivariate 1D-CNN have been tested. The univariate 1D-CNN method takes a single signal as input. We label the individual signals in period $\mathfrak{P}_1$ and the signals obtained from the experiments of Diltiazem as NO-K-blocker and those signals are denoted by $\mathfrak{s}_{\mathfrak{p}}(t)$. We label the individual signals obtained from experiments of K-blocker drugs as K-blocker and those signals are denoted by $\mathfrak{s}_{n\mathfrak{p}}(t)$.

On the other hand, the multivariate 1D-CNN takes a pair of signals as input. One signal from period $\mathfrak{P}_1$ is paired with another signal in the same well and beat but recorded by a different electrode (in particular, it also belongs to period frP_1). $\mathfrak{s}_{c1}(t)$ and $\mathfrak{s}_{c2}(t)$ are the notations of these paired signals and N_t is the notation of their length. Then, $\mathfrak{S}$ is defined as the array in $\mathbb{R}^{2\times N_t}$ given by :

$$\mathfrak{S}_{ij} = \begin{cases} \mathfrak{s}_{c1}(t_j), & i = 1 \\ \mathfrak{s}_{c2}(t_j), & i = 2 \end{cases} \tag{3.3.6}$$

where $(t_j)_{1\leq j\leq N_t}$ corresponds to the set of the time steps. All these pairs of signals will be labelled as NO-K-blocker.

In a comparable way, one signal from period $\mathfrak{P}_1$ (that is denoted by $\mathfrak{s}_c(t)$) is paired with one signal from period $\mathfrak{P}_2$ (that is denoted by $\mathfrak{s}_d(t)$) in the same electrode and well by introducing the array $\mathfrak{S}$ of size $2 \times N_t$ given by:

$$\mathfrak{S}_{ij} = \begin{cases} \mathfrak{s}_c(t_j), & i = 1 \\ \mathfrak{s}_d(t_j), & i = 2 \end{cases} \tag{3.3.7}$$

For these pairs of signals, the ones coming from experiments of Diltiazem will be labeled

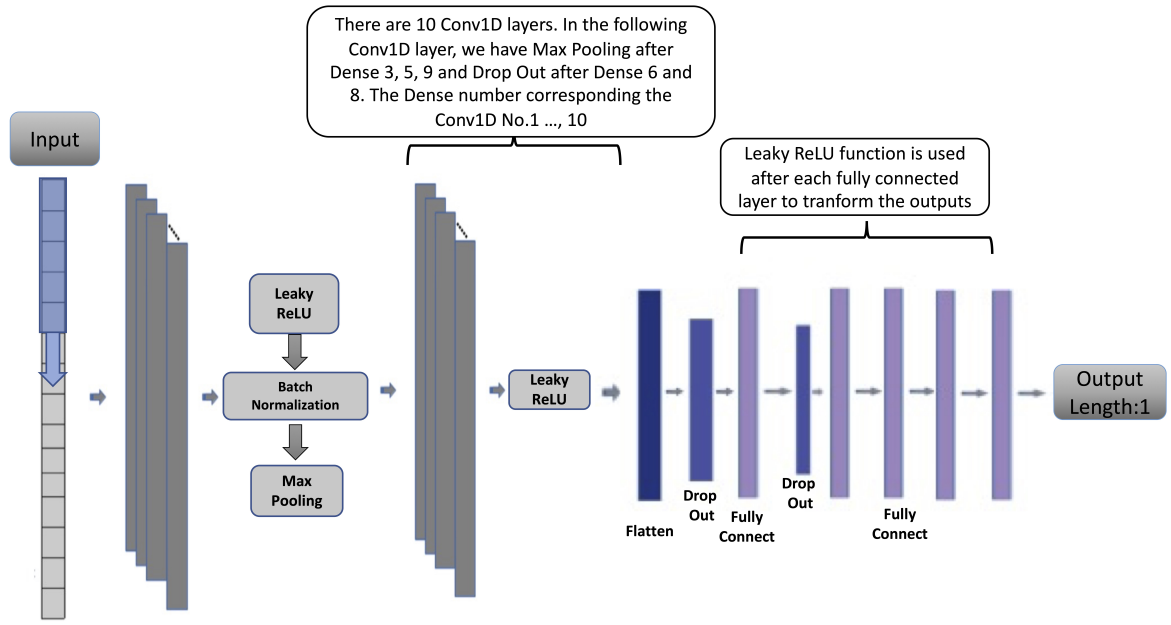


Figure 3.4: Architecture of the 1D-CNN model

as NO-K-blocker whereas the ones coming from experiments of K-blocker drugs will be labeled as K-blocker.

1D-CNN methodology and architecture

The architecture of the implemented 1D-CNN method (which is common to the univariate and multivariate 1D-CNNs) is shown in Figure 3.4. It is based on the traditional AlexNet [152] proposed in [153]. In the CNN method, the raw signals are used as input. The weights for the first hidden layer are initialized with random normal distributed numbers. Then, the kernel window with a height of 5 and a width of 1 unit slides across the input time series to do a cross-correlation operation [126, Chapter 6, section 2] with 1 stride from top to bottom. The example of a cross-correlation operation can be seen in Figure 3.5. The results from those cross-correlation operations constitute the features extracted from the first convolutional layer (the convolutional layer can be called Conv1D) and the Conv1D has 16 extractors to do this kind of computation named convolution filters.

The output from the first Conv1D will be transformed by a leaky ReLU function (3.3.2) before the next process. As ReLU (3.3.1) could not be used as negative features are transformed to 0. So, to avoid losing information from negative features in the following computation, the leaky ReLU (3.3.2) has been chosen to transform the output from all Conv1D layers and fully connected layers.

After the first Conv1D layer and the activation process, the Batch Normalization layer has been used to speed up the training process and reduce the sensitivity of the initialization of the convolutional neural network [145]. Then the max-filter of the max-pooling layer [126, Chapter 6, section 5] extracts the maximum values from the defined region which has a height of 5 and a width of 1, as has been proposed in [153]. There is also a dropout layer with a rate of 50% added between several Conv1D layers. Then

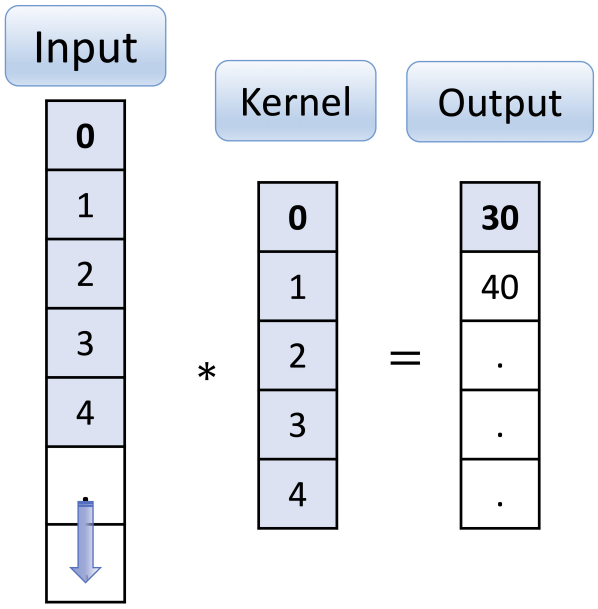


Figure 3.5: One-dimensional cross-correlation operation. The shaded portions are the first output element as well as the input and kernel tensor elements used for the output computation: $0 \times 0 + 1 \times 1 + 2 \times 2 + 3 \times 3 + 4 \times 4 = 30$

the other convolution layers are following the same logic as the first convolution layer. In summary, our model includes 10 Conv1D layers and each layer includes: 16, 16, 26, 26, 32, 52, 52, 72, 72, and 84 filters. The max pooling or dropout layers have been put between several Conv1D layers.

At the end of the Conv1D layers, the flattened tensor reshapes the outputs from the previous layer in a one-dimensional array. After flattening the layer, there is a dropout layer that reduces 50% of the parameters. Then some fully connected layers were added to the dropout layer. Every neuron in a fully connected layer is connected to every neuron in the next fully connected layer. There are 5 fully connected layers that have 872, 328, 128, 64, and 16 neurons respectively. Between the last fully connected layer and the output, the sigmoid activation function (3.3.3) has been used to transform the input to the output.

In the training phase of the network, we used Binary Cross Entropy as the loss function and the weights of hidden layers were computed by using an Adam optimizer for CNN models.

3.3.4 2D-CNN

Pre-processing for 2D-CNN

Before presenting the 2D-CNN method, let us explain how images have been generated from the signals. The main idea is to associate an image to a pair of signals constructed in the data pre-processing step for the multivariate CNN. The image is a portrait in which the first signal of the pair is a function of the second signal of the pair.

Since the depolarisation phase and the repolarisation phase are very different in terms

of magnitude and duration, we will consider them separately and two images will be associated with each pair of signals after a renormalization step. To be more precise, reusing the notations (3.3.6) or (3.3.7), we define

$$\mathfrak{p} = \max_{1\leq i\leq 2, 1\leq j\leq N_t} \mathfrak{S}_{ij} \text{ and } \mathfrak{q} = \min_{1\leq i\leq 2, 1\leq j\leq N_t} \mathfrak{S}_{ij}$$

and then, we define the following points in the 2D space:

$$\left(\frac{\mathfrak{S}_{1j}-\mathfrak{q}}{\mathfrak{p}-\mathfrak{q}}, \frac{\mathfrak{S}_{2j}-\mathfrak{q}}{\mathfrak{p}-\mathfrak{q}}\right)$$

for $1 \leq j \leq N_t$.

The images have been generated from the data of these points considering the following process. Every image is divided into a certain number of squares. For the depolarisation and repolarisation phase, the graph is divided into 30×30 and 60×60 squares respectively. The images have been generated by counting how many curve points lay in each square. The number of points included in each square also represents the density of the pixels and the density is presented by the colour of the squares in the image. Examples of images are given in Figure 3.6 and 3.7. We expect to observe that the pixels diverge along the diagonals when a drug causes an alteration of the signals.

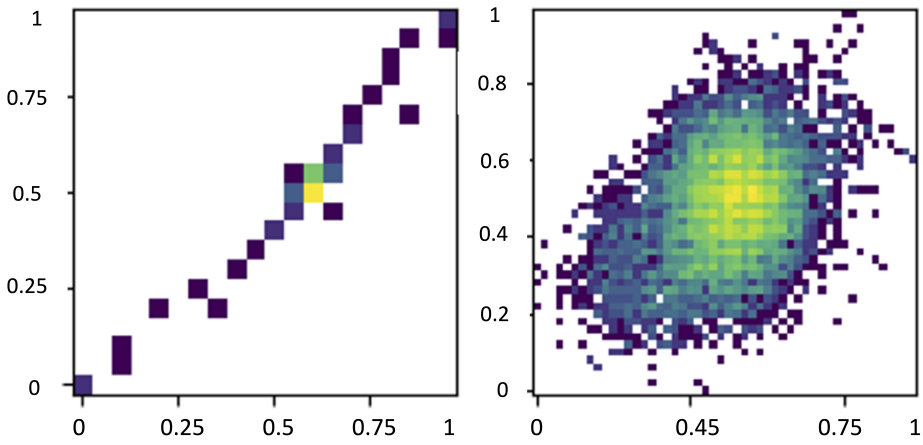


Figure 3.6: Example of image obtained by pairing signals both in period $\mathfrak{P}_1$ as in (3.3.6). Left: depolarisation phase, right: repolarisation phase

2D-CNN methodology and architecture

These generated images are the inputs of the 2D-CNN method. The size of the image corresponds to height× width × colour channels. In contrast to 1D-CNN method, the kernel for 2D-CNN slides crosses the input along with two dimensional directions which are the height and width of the image like in Figure 3.8. The sizes for kernel and max pooling window correspond to height × width × colour channel.

In the 2D-CNN model, the kernel size is $3 \times 3 \times 3$, the window size is $2 \times 2 \times 3$ for max-pooling and a 20% dropout rate. The architecture of proposed 2D-CNN model is

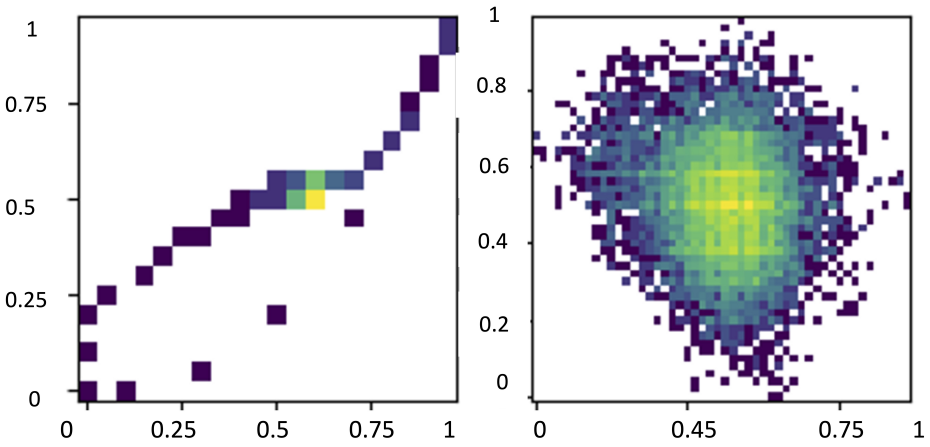


Figure 3.7: Example of the image obtained by pairing a signal coming from period $\mathfrak{P}_1$ and a signal coming from period $\mathfrak{P}_2$ as in (3.3.7). Left: depolarisation phase, right: repolarisation phase

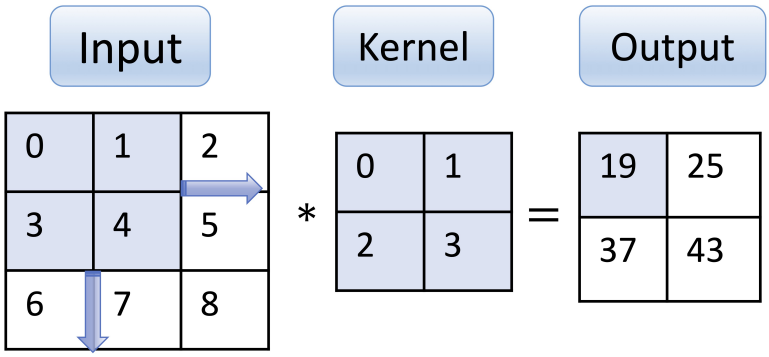


Figure 3.8: Two-dimensional cross-correlation operation. The shaded portions correspond to the first output element as well as the input and kernel tensor elements used for the computation of this output: $0 \times 0 + 1 \times 1 + 3 \times 2 + 4 \times 3 = 19$, figure from [126, Chapter 6, section 4.1].

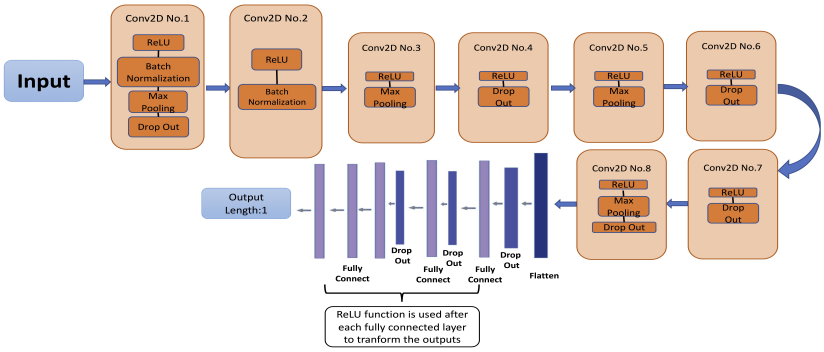


Figure 3.9: Architecture of the 2D-CNN model

Table 3.4: Details for 2D CNN model: number of layers and their type, number of neurons, kernel size, whether it has Batch Normalization, max pooling (and, if so, the size) and dropout (and, if so, the rate)

Number of Hidden Layers	Number of Filters	Number of Neurons	Kernel Size	Batch Normalization	Max Pooling	Dropout
Conv2D Layer 1	16	None	3	Yes	Yes, pool size(2, 2)	Yes, 20%
Conv2D Layer 2	16	None	3	Yes	No	No
Conv2D Layer 3	32	None	3	No	Yes, pool size(2, 2)	No
Conv2D Layer 4	32	None	3	No	No	Yes, 20%
Conv2D Layer 5	54	None	3	No	Yes, pool size(2, 2)	No
Conv2D Layer 6	54	None	3	No	No	Yes, 20%
Conv2D Layer 7	62	None	3	No	No	Yes, 20%
Conv2D Layer 8	62	None	3	No	Yes, pool size(2, 2)	Yes, 20%
Flatten Layer 9	None	None	None	No	No	Yes, 20%
Dense Layer 10	None	872	None	No	No	Yes, 20%
Dense Layer 11	None	328	None	No	No	Yes, 20%
Dense Layer 12	None	164	None	No	No	No
Dense Layer 13	None	64	None	No	No	No
Dense Layer 14	None	16	None	No	No	No

shown in Figure 3.9. There are 8 hidden two dimensional convolution layers (Conv2D) and the input of image data will be passed to those sets of Conv2D layers. The results from the last Conv2D layer will be flattened by a Flatten layer. Then, the flattened parameters will be pass to a set of fully connected layers. The results from each hidden layer will be transformed by a ReLU activation function (3.3.1). The outputs from the last hidden layer will be passed to the fully connected output layer of 1 neuron with the sigmoid activation function (3.3.3). Binary Cross Entropy was used as a loss function and the weights of hidden layers were computed by using an Adam optimizer.

More details of the architecture of the 2D-CNN model are given in Table 3.4 which includes the name of hidden layers, the number of filters and the kernel size for each Conv2D layer and the number of neurons in the fully connected layers. Table 3.4 also lists the information regarding whether there is a Batch Normalization layer, Max Pooling layer or Dropout process after each Conv2D or fully connected layer. The techniques used in Table 3.4 refer to [126, Chapter 6], [149, Chapter 9].

3.4 Setup of binary classification

This section will present the details of the design of the classification tasks for testing the four ANN methods presented above (MLP, Univariate 1D-CNN, Multivariate 1D-CNN and 2D-CNN). The tests have performed for two different scenarios:

- *Scenario 1*: The training set is composed of random signals taken from the recordings of all the 11 K blockers of our data set. The test set is composed of random signals taken from the same drugs.
- *Scenario 2*: The training set is composed of random signals taken from the recordings of only 7 K blockers among the 11. The test set is composed of random signals from the 4 remaining K blockers.

For each scenario, the four ANN methods have been tested to predict the label of the signals in the test set. The obtained results make it possible to compare the ANN

methods in terms of performances, data processing costs and network training costs. The performances of the networks are evaluated using the classical criteria [153] which are listed in Table 3.5. The ANN methods have implemented using the Sequential model from $TensorFlow^{TM}$ [154].

Table 3.5: Criteria to evaluate the classification methods

Criterion	Meaning
False negative (FN)	The classification result where positive training data are evaluated as negative
Percentage of FN (FN%)	$FN\% = \frac{FN}{TP+FN}$
False positive (FP)	The classification result where negative training data are evaluated as positive
True negative (TN)	The classification result where negative training data are evaluated as negative
True positive (TP)	The classification result where positive training data are evaluated as positive
Accuracy	The percentage of signals correctly classified, $\frac{TP+TN}{TP+TN+FP+FN}$
Precision	The percentage of predicted controls cases that were correctly classified, $\frac{TP}{TP+FP}$
Recall	The percentage of actual control cases that were correctly classified, $\frac{TP}{TP+FN}$
AUC	Area under the curve of a receiver operating characteristic (ROC) curve

For *scenario 1*, different methods have been implemented in the following way:

1. The MLP method takes the rescaled features array of size 64 as input. There are 29116 signals have been selected.

2. The Univariate 1D-CNN method takes single signals as inputs. 27831 signals have been randomly chosen from NO-K-blocker experiments and 30000 signals from K-blocker experiments (with a third of the signals in plate 1, a third of the signals in plate 2 and a third of the signals in plate 3, each plate containing signals from 4 tested drugs).

3. The Multivariate 1D-CNN method takes pairs of signals as inputs. The same signals have been used as inputs for the Univariate CNN method.

4. The 2D-CNN takes images as inputs. The images were generated from the same set of signals as the Multivariate 1D-CNN. By loading the images to the 2D-CNN, each image corresponds to a matrix of size $97 \times 181 \times 3$ (corresponding to height, width and colour channels).

During the model training phase, there are 20% of the signals randomly distributed in the test set and 80% of the signals in the training set. In the training set, 20% have been randomly chosen as a validation set.

For *scenario 2*, the same signals have been used as in *scenario 1* for the four types of ANNs. However, the signals have been divided in a different way since the training is made by using signals coming only from a part of the drugs. More precisely, the network is trained by using the signals coming from only 7 K blockers (Loratadine, Ibutilide, Mexiletine, Droperidol, Chlorpromazine, Clozapine, Dofetilide). In the training part, 20% of signals have been randomly chosen as the validation set and the rest of the signals are in the training set. Then, the signals coming from the 4 remaining K blockers (Clarithromycine, Cisapride, Bepridil, Azimilide) are used to test the networks.

3.5 Binary classification results and methods evaluations

In this subsection, the classification results obtained for each scenario and each ANN method are presented and the evaluations of each method will be listed.

3.5.1 Results of binary classification

The classification results are listed in Table 3.6 for *scenario 1* and Table 3.7 for *scenario 2*. In general, the performances of MLP and multivariate 1D-CNN methods are superior with 96.82% and 99.26% of accuracy for *scenario 1* and 98.33% and 99.13% of accuracy for *scenario 2*, respectively. A special attention has to be paid to the error rate of K-blocker wrongly classified as NO-K-blocker, because in practice this may have more critical outcomes. Again, the MLP and multivariate 1D-CNN methods give satisfactory results with respectively 0.2% and 0.14% being false negative (FN). A study on the signals which have been wrongly classified allows to notice that a large part of them corresponds to drugs whose channel effect is complex either because they correspond to the mixed blockers, like Chlorpromazine and Clozapine, or because it has been classified as a non-K blocker (Diltiazem) whereas this drug partly blocks the K channel at high concentration. So, the classification error is partly related to the fact that classifying the drug as K-blocker or non-K blocker may sometimes be reductive due to their complex effect on the ionic channel activity. There will be more discussion about this point in the next section.

Table 3.6: Classification results of the tested ANNs with *scenario 1*

Metrics	**Classifier**			
	MLP	Univariate 1D-CNN	Multivariate 1D-CNN	2D-CNN
Accuracy	**96.82%**	86.58%	**99.26%**	88.50%
Precision	**92.79%**	86.98%	**98.70%**	89.92%
Recall	**99.74%**	85.98%	**99.86%**	87.52%
AUC	**99.63%**	94.21%	**99.90%**	95.65%
FN%	**0.20%**	13.80%	**0.14%**	12.95%

Table 3.7: Classification results of the tested ANNs with *scenario 2*

Metrics	**Classifier**			
	MLP	Univariate 1D-CNN	Multivariate 1D-CNN	2D-CNN
Accuracy	**98.33%**	64.63%	**99.13%**	84.42%
Precision	**99.90%**	65.62%	**98.76%**	88.43%
Recall	**96.00%**	61.47%	**99.67%**	82.42%
AUC	**98.80%**	70.85%	**99.63%**	92.27%
FN%	**2.72%**	36.24%	**0.41%**	19.77%

It is interesting to notice that, contrary to the other methods, the performances of MLP and multivariate 1D-CNN methods with *scenario 2* are similar to the ones with *scenario 1*. This suggests that these methods will have good predictive capabilities for testing new drugs whose action on ionic channels is still unknown.

Table 3.8: Data Processing and Training Costs of the tested ANNs

Classifier	Data Processing Cost	Training Time for Each Epoch	Number of Epochs needed	Training Time
MLP	**Very High**	1 minutes	20 epochs	**20 minutes**
Univariate 1D-CNN	Very Low	3.5 minutes	100 epochs	5.8 hours
Multivariate 1D-CNN	**Very Low**	3.67 minutes	20 epochs	**1.2 hours**
2D-CNN	Low	6.67 minutes	60 epochs	6.7 hours

3.5.2 Methods evaluations

Considering the showed classification results, MLP and multivariate 1D-CNN methods can provide the most promising analysis of experimental data set. Besides the evaluation of the performance of the ANN methods, additional factors have to be considered such as the data processing costs and the model training costs to have a more comprehensive evaluation of each method. In terms of industrial implementation, it is important to consider which ANN methods would be easier to deploy. The data processing costs and model training costs are listed in Table 3.8. Depending on the size of the data set and computational power, the time demand would be different. The data processing costs are ranked from very low to very high. In terms of training costs, the training time for each epoch and the total training time are considered to get a well-trained model. In summary, MLP needs the shortest training time but has the largest data processing costs. Multivariate 1D-CNN has very low data processing costs and needs 1.2 hours of training time. In other words, compared to MLP, Multivariate 1D-CNN has a lower requirement for data processing but higher computation requirements for training the network.

3.6 Beyond binary classification

As already mentioned in the previous section, it may be too restrictive to assign a class (K-blocker or NO-K-blocker) to a drug because this binary assignment does not reflect the complex behaviour of drugs on ion channels. In particular, once a drug has been identified as a K-blocker, a natural refinement could be to classify it as a pure K-blocker or a multi-channel blocker with the different possible combinations K+Na, K+Ca or K+Na+Ca. It may also be important for this classification to consider each concentration separately since they may have a different impact on the ion channels. In this section, we present some tests on multiclass classification. In particular, we have started to investigate two scenarios:

1. Classification according to molecule concentration. Given a signal, we try to identify to which level of molecule concentration it corresponds.

2. Classification as pure K-blocker or multi-channel blocker.

For all these tests, we considered the multivariate 1D-CNN presented in Chapter 3, Section 3.3.

3.6.1 Classification according to concentration.

In the preliminary study that is presented here, we made the simple hypothesis that, when the drug concentration increases, the identification of the molecule becomes more accurate. Conversely, we assume that when the drug concentration is lower, we have a larger classification error. The idea behind this hypothesis comes from the observation that it is more difficult to distinguish signals obtained for low concentrations from the ones of the control group. It is to be noted that this assumption can be naive because when concentration increases, for instance, a drug that primarily has an impact on the hERG channel can have effects on other channels. But we have chosen to disregard these more complex behaviours (like counteract of multiple blockades with higher concentrations) in this exploratory study.

We consider here 5 different classes: the first one corresponds to controls (baseline recording which is prior to drug addition), and the other four correspond to the four different levels of concentrations, $C_1 \leq \ldots \leq C_4$. These are reported in Table 3.1. To start with, we considered here a set of data composed of the signals obtained for the control and the drug Ibutilide.

The training set consists of 80% of the signals recorded for the Ibutilide assessment (meaning that we have data for controls and for the four concentrations). The training procedure is the same as the multivariate 1D-CNN presented in Section 3.3 and 3.4.

The obtained results are depicted in Figure 3.10. We can see that, for the lowest concentration, C_1, we obtain, as expected, less accuracy, as 8% of the signals are classified as controls, and part of the signals are classified as C_2. For higher concentrations, the results are encouraging, as we obtain an accuracy larger than 94%. Therefore, this means that we have, at least for some compounds, the possibility to have a finer classification than the binary one, with multiple levels of blockade intensity. Further tests are needed in order to have a finer assessment of the performances of this classification with different kinds of drugs, in particular when they are multi-channel blockers.

3.6.2 Multi-blockers classification.

We describe in this section the compound classification according to different types of blockade. We have simultaneously considered pure potassium blockers and multi-channel blockers corresponding to the following four classes:

- K blockers corresponding to Droperidol, Ibutilide, Dofetilide, Clarithromycine, and Cisapride
- K+Na blockers corresponding to Mexiletine
- K+Ca blockers corresponding to Clozapine and Loratadine
- K+Na+Ca blockers corresponding to Chlorpromazine, Azimilide, and Bepridil

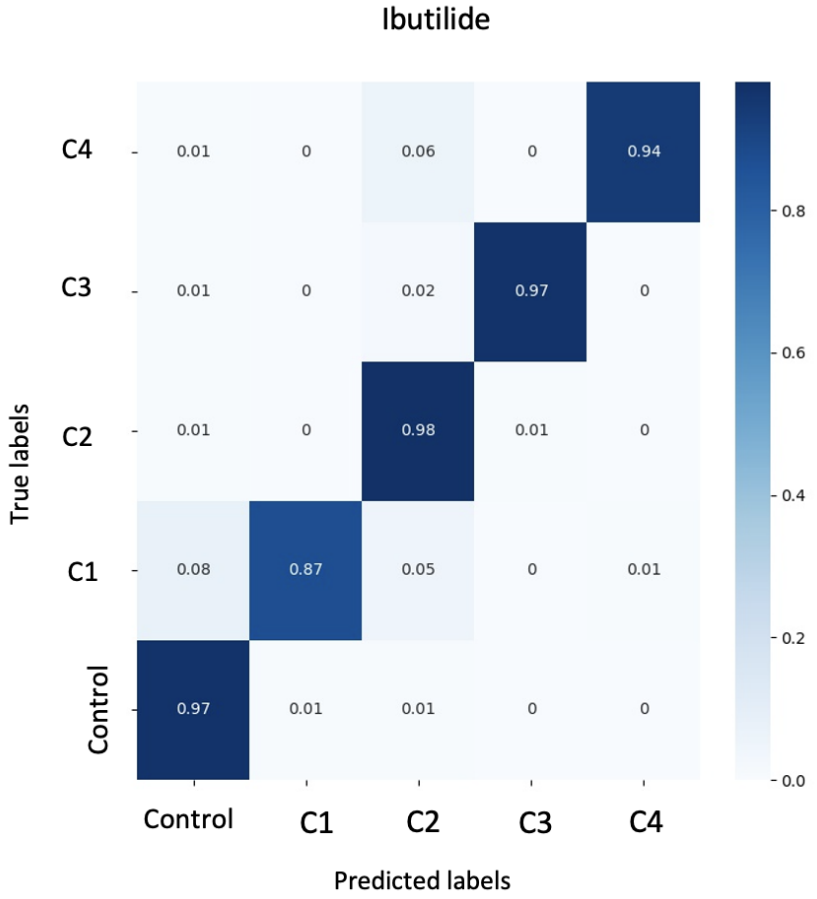


Figure 3.10: Classification of concentrations of Ibutilide

The training set consists of 80% of the signals recorded in the experiments when adding the above listed compounds. The training procedure was the same as the multivariate 1D-CNN presented in Section 3.3 and 3.4.

The results of this classification are depicted in Figure 3.11. We can see that multi-channel blockers like K+Na+Ca blockers have a higher chance of being misclassified as pure K blockers. This can be illustrated by some multi-channel blockers that may share some similar characteristics as pure K blockers. When just the potassium or two channels at a time are blocked, the accuracy is larger than 92%, showing the potential of distinguishing between different blockade mechanisms.

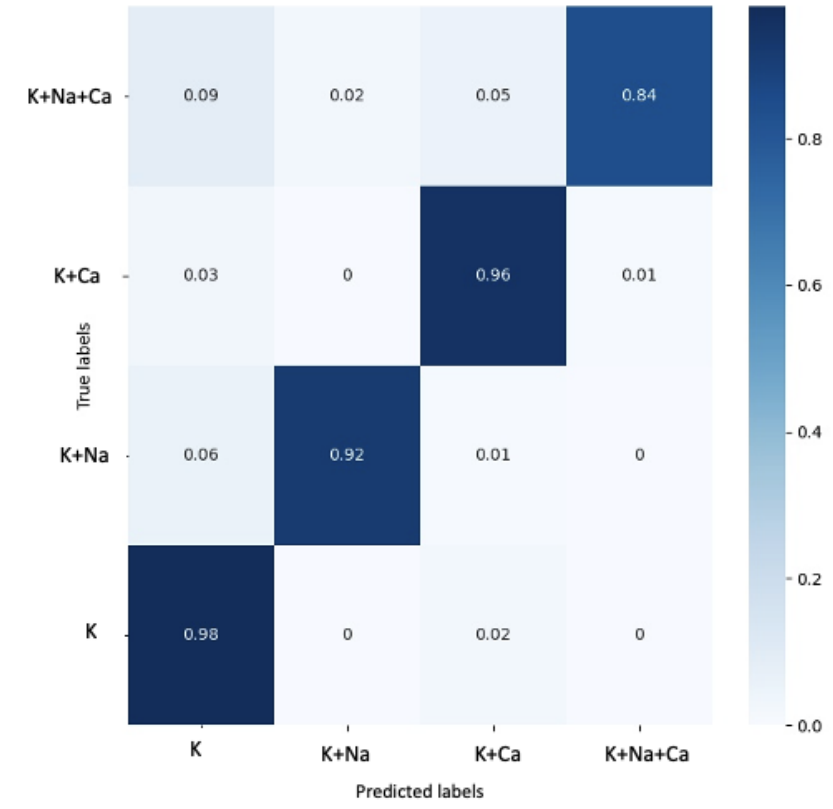


Figure 3.11: Classification of different drug effects

3.7 Replicator neural networks (RNNs) for anomaly detection

In hiPSC CM drug safety assays, a substantial volume of MEA signals is recorded. Before beginning any signal analysis, a series of data preprocessing steps is needed. These preprocessing steps include data cleaning, normalization, etc. The principal objective of this preprocessing is to ensure the data quality, which is paramount for subsequent analysis.

However, among all the collected FP signals, some signals may deviate from the main signal shape. This discrepancy can often be attributed to suboptimal cell conditions or recording circumstances. Such abnormal recordings introduce potential biases that could

compromise the experiments conclusions. Consequently, it becomes necessary to exclude these problematic recordings from the data set. Traditionally, this has been accomplished through manual selection and removal, a labor-intensive process that consumes significant resources.

In this section, we aim to explore methods able to automatically and reliably identify poorly recorded FP signals. If we consider this challenge as a semi-supervised learning problem, we could use an RNN-based approach. RNN is an example of autoencoder which was already presented in Chapter 2, section 2.3.1. We will test a linear and nonlinear autoencoder to perform this task.

Autoencoders are usually set up to perform anomaly detection, as presented in [124]. They are used in a semi-supervised way. An autoencoder is trained in order to well reproduce signals which are considered as normal. When trying to reconstruct normal signals, it would henceforth have a small reconstruction error. On the contrary, when trying to reconstruct an abnormal signal, it would have a significantly larger reconstruction error. This provides a way to perform a binary classification between normal and abnormal signals, by considering a threshold on the reconstruction error.

We consider here a first test with real FP signals obtained from the drug safety assessment experiments mentioned in Section 3.2. We restrict to the depolarisation part for simplicity as the majority of the anomalies occurring in MEA experiments can be seen in the depolarisation part of the signal. The data set consists of $N_d = 38894$ signals, each signal being denoted by $\mathfrak{s} \in \mathbb{R}^{N_t}$, with $N_t = 320$. In these data, there are 34530 normal signals and 4364 signals considered as anomalies. Among all normal signals, we will take 80% to train and 20% to validate the autoencoder.

We consider, for the reconstruction error, the root relative mean squared error (RRMSE), as done in Chapter 2, section 2.5:

$$RRMSE(\mathfrak{s}, \hat{\mathfrak{s}}) = \frac{\sqrt{\frac{1}{N_t}\sum_{i=1}^{N_t}(\mathfrak{s}_i - \hat{\mathfrak{s}_i})^2}}{\sqrt{\frac{1}{N_t}\sum_{i=1}^{N_t}\mathfrak{s}_i^2}}$$

where $\mathfrak{s}, \hat{\mathfrak{s}} \in \mathbb{R}^{N_t}$ are the true and reconstructed signals respectively. The rationale behind this choice for error reconstruction lies in the robustness with respect to the eventual change of scale in the recorded signals.

In view of performing anomaly detection by classifying the reconstruction based on the error, we have to introduce a threshold. We will set its value by considering the average reconstruction $RRMSE_{\mathfrak{s}}$ in the training set plus its standard deviation η, which is denoted by $\mathfrak{e} = RRMSE_{\mathfrak{s}} + \eta$. When the reconstructed error of a given new datum is higher than this threshold $\mathfrak{e}$, we will predict it as an anomaly.

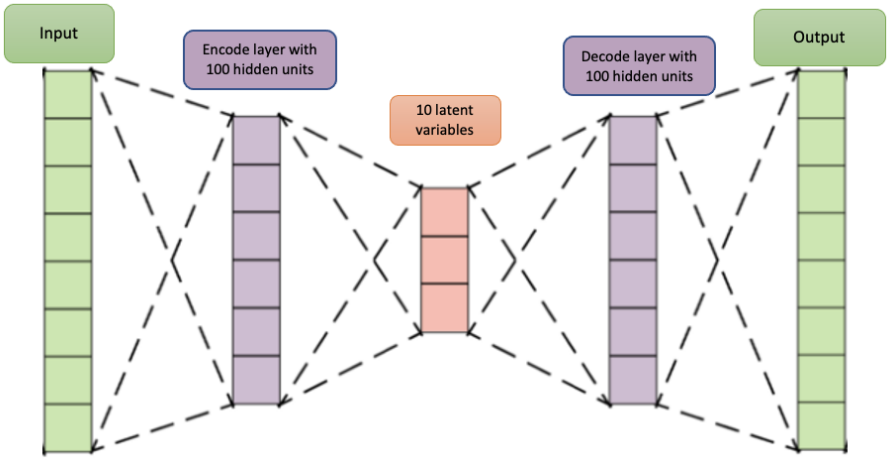
Three different autoencoders have been tested. The first one, that we will refer to as linear autoencoder, is built by considering the following set-up. We arrange the data into a matrix $A \in \mathbb{R}^{N_t \times N_d}$, each datum being a column of this matrix. The singular value decomposition (SVD) of A is denoted $A = USV^T$. Here, U consists of the eigenvectors of AA^T, V consists of the eigenvectors of A^TA and S contains the singular values in decreasing order. According to the idea of principal component analysis (PCA) in [155] and [156], if we select the first $\mathfrak{k}$ column vectors of U (corresponding to the $\mathfrak{k}$ largest

singular values), denoted $\tilde{U}$, for an input datum u, we can define a linear autoencoder as:

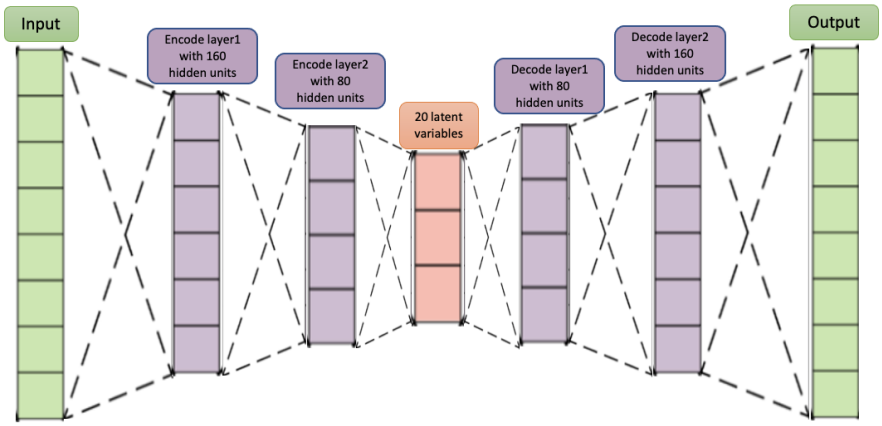
$$\alpha = \psi(u) = \tilde{U}^T u$$

$$\tilde{u} = \psi^\dagger(u) = \tilde{U}\alpha.$$

In the following tests, we will set $\mathfrak{k} = 40$ ($\mathfrak{k}$ should be set depending on specific data). This linear autoencoder will be compared with two other nonlinear autoencoders that have the following architectures, denoted AE1, AE2 respectively, depicted in Figure 3.12.



(a) Nonlinear autoencoder AE1



(b) Nonlinear autoencoder AE2

Figure 3.12: Autoencoder Architectures

For these two non-linear autoencoders, the considered activation function is ELU(2.3.2). The training was performed by using the Adam optimizer, 500 epochs with batch size 20.

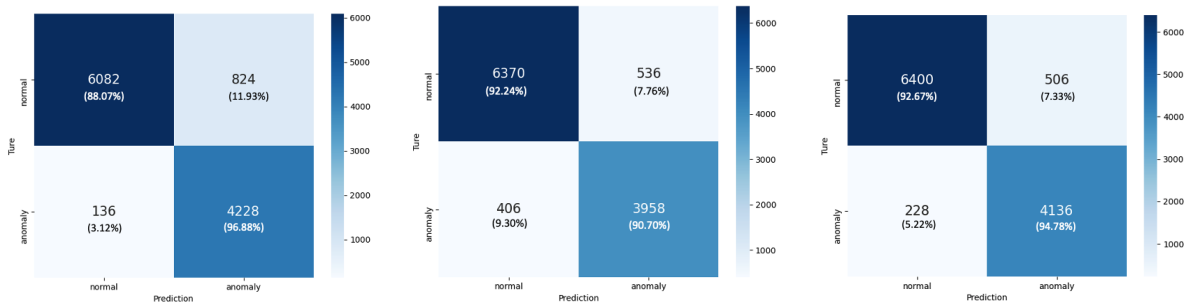
The error thresholds $\mathfrak{e}$ to be used in the classification are reported in Table 3.9. The test set consists of 20% of the normal signals completed by the abnormal signals.

Let us describe the results obtained. We compare the RRMSE of each signal in the test set to the threshold and attribute a label. The results, depicted in Figure 3.13, are presented as percentages. These percentages are obtained by dividing the total number

Table 3.9: Thresholds results for three autoencoders

Methods	Average RRMSE	Standard deviation
Linear autoecnoder	0.0207	0.0756
Nonlinear autoencoder AE1	0.0435	0.1442
Nonlinear autoencoder AE2	0.03051	0.1069

of abnormal and normal signals, respectively.



(a) Results for the linear autoencoder (b) Results for the nonlinear autoencoder AE1 (c) Results for the nonlinear autoencoder AE2

Figure 3.13: Anomaly detection results

Both the linear autoencoder and the non-linear autoencoder AE2 detected anomalies with a large accuracy (around 95%) and the linear autoencoder slightly outperforms AE2. However, the linear autoencoder does not have a good accuracy when predicting normal signals, the error being around 12%. The autoencoder AE2 has more uniform performances, as the error in detecting normal signals is around 7%. The global accuracy of the linear autoencoder is around 91.5% and the accuracies of AE1 and AE2 are around 91.5% and 93.5% respectively.

In this section, we conducted tests using three different autoencoders for anomaly detection. The linear autoencoder stands out as the most simple method, offering the advantage of lower computational costs while maintaining a high level of accuracy. However, in comparison to nonlinear autoencoder AE2, one limitation of the linear autoencoder is its tendency to detect more normal cases as anomalies, potentially leading to the misprediction of normal data.

It is important to highlight that this section represents a preliminary investigation of methods for automatic anomaly detection. To establish a robust and reliable method, there are several aspects to be considered in further studies.

Firstly, concerning the selection of the parameter $\mathfrak{k}$, we should address the question of what criterion determines the optimal value of $\mathfrak{k}$ that results in the lowest overall prediction errors. The same question applies to the nonlinear autoencoder: how do the architecture and dimension of the latent space influence the prediction errors? Likewise, the threshold value is set arbitrarily and we would need to determine a criterion suitable for detecting anomalies. At last, we would like to understand in which situations the linear autoencoder outperforms the nonlinear ones and vice versa.

3.8 Summary: chapter conclusion and discussion

In this chapter, we tested some methods based on ANNs to address specific challenges in the field of safety pharmacology.

In our first study, we explored the performance of four different neural network approaches, namely MLP, univariate 1D-CNN, multivariate 1D-CNN, and 2D-CNN, in a binary classification task involving the classification of K channel blockers and NO-K blockers. Our findings indicated that both MLP and multivariate 1D-CNN yielded the highest accuracy. MLP offers the advantage of having lower training costs, but it requires more extensive data pre-processing compared to multivariate 1D-CNN which requires less computational costs in pre-processing but more in training.

We also presented two exploratory studies in Sections 3.6 and 3.7. First, we worked on a classification according to the concentration of drugs or according to their multi-channel effect. This problem can be viewed as a multiclass classification problem. We showed some results obtained with the multivariate 1D-CNN. Then, for the question related to anomaly detection, we utilised three different autoencoders to automatically spot the anomalies.

Overall, we got encouraging results that lead us to believe that the use of Neural Networks may allow us to achieve a fine classification of drugs and may be of great help in assessing the cardiac safety of drugs in a semi-automatic and high throughput manner.

It is worth noting that these methods were tested on a specific data set. In future works, we aim to assess whether they can maintain their high accuracy when applied to a new data set. Furthermore, we recognise that there is a wide range of alternative ANN methods that could potentially surpass the current methods in terms of computational efficiency and accuracy. In the current context, we are constrained by limited resources, which restrict our ability to compare and analyze a broader range of ANN methods.

To ensure the applicability of these methods in industrial settings, it is imperative to conduct additional tests in various scenarios to validate their robustness and efficiency. Especially, considering the multiclass classification and anomaly detection scenarios, there are still a lot of potential aspects to be considered in the future.

Chapter 4

Two Additional Studies on Cardiovascular Signals

Part 1: Analysis of Data from Patients Affected by Brugada Syndrome

Abstract

This part presents results obtained in the context of a collaboration with the Pharmacology department at Antwerp University and more specifically with Maaike Alaerts. It focuses on investigating Brugada syndrome, a genetic disorder in which the cardiac electrical activity is irregular. Based on data sets of action potentials (AP) recorded for patient derived hiPSC-CMs, we aim to explore methods that would allow to better understand the manifestations of Brugada syndrome. First of all, we will present how Neural Network methods can help to automatically classify patients and healthy individuals. Subsequently, we will introduce a strategy in order to distinguish patients with severe symptoms from patients with non-severe symptoms. This approach combines clustering methods with a classification strategy and provides insights into the severity degrees of the symptoms. In the end, we will present some methods which could help us to understand better the mechanisms related to Brugada syndrome and some prospective studies.

4.1 Introduction

As described in the studies [157, 158, 159], Brugada syndrome is a genetic condition that can lead to dangerous irregular heart rhythms and sudden cardiac death in affected individuals. It has a notably high incidence in various regions worldwide and carries a substantial risk of sudden death.

Certain studies have observed a higher prevalence of this syndrome, particularly in patients with SCN5A mutations. SCN5A gene [160] encodes the alpha subunit of the main cardiac sodium channel Nav1.5. SCN5A mutations can influence inward sodium current (I_{Na}) and cause irregular cardiac electrophysiological functions. However, much

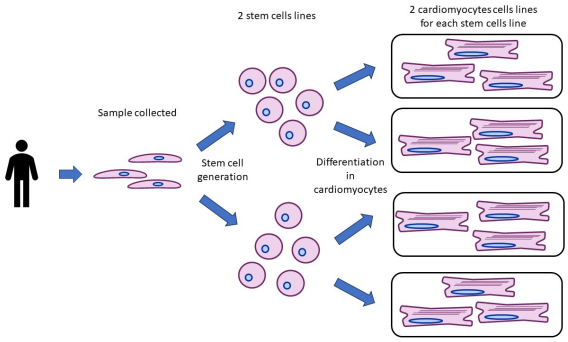


Figure 4.1: The process to generate patient derived hiPSC-CMs

work remains to be undertaken to fully understand the underlying mechanisms of Brugada syndrome. The Antwerp research team has undertaken the task of cloning hiPSC-CMs from patients and healthy individuals and recording cellular signals (Patch Clamp data Action Potential, AP) to delve deeper into the intricacies of the mechanisms of this disease.

Compared with hiPSC-CMs assays for drug safety assessment in Chapter 3, patient derived hiPSC-CMs carry genetic mutations and changes in other genes. These changes make it different from the healthy cells.

One of the central challenges comes from the fact that patients sharing the same genetic mutation exhibit varying degrees of syndrome expression. The objective is to understand the relationship between the genetic mutation and the ion channel activities. Furthermore, the team aims to investigate why some patients exhibit milder or more severe manifestations of the syndrome, shedding light on the underlying factors that contribute to this variability. It should be mentioned that this work is a preliminary investigation of a potentially broad topic, involving multidisciplinary research.

4.1.1 The AP data set

In this section, we will give some information about the data set of action potentials (AP) provided by Antwerp team. The AP recordings come from 5 patients suffering from Brugada syndrome. For each patient, there are 2 cell lines and each cell line has 2 differentiations (we need to note that the differentiation process can itself be a source of variability and increase the challenges in this study). An example of this process is described in Figure 4.1. In addition, the same process has been implemented for healthy individuals in order to get the same quantity of hiPSC-CMs APs which correspond to the control cell lines. Each AP recording has been segmented by beat and each beat is denoted $\mathfrak{s}_a \in \mathbb{R}^{N_t}$ where $N_t = 15000$ corresponds to the number of time steps.

The data generated are labelled as follows: BrS8C2_D1 and D2, BrS8C3_D1 and D2, BrS9C7_D1, BrS10C3_D1, and BrS12C1_D1 and D2. BrS is followed by the patient code, C is followed by the cell line code, and D is followed by the differentiation code. For the control cell lines, we have the following data: M4550C3_D1 and D2, M4550C9_D1 and D2, M4550C15_D1 and D2, F199001C2_D1 and D2, and F199001C3_D1 and D2.

4.1.2 The goals of this study

In this work, we will investigate the following questions:

1. **Distinguishing patients from healthy individuals:** This question can be viewed as a classification problem in the same vein as the problems studied in Chapter 3. Answering to this question therefore consists in labelling AP recordings according to two classes: control and patient. In Section 4.2, we will approach this classification problem using neural network methods and perform a comparison between an MLP classifier and a multivariate 1D-CNN. Both methods were presented in Chapter 3, Section 3.3.

2. **Distinguishing patients with severe symptoms from non-severe symptoms:** The question is now the following: given the data collected on the different cell lines and differentiations of the same individual, is it possible to deduce whether the syndrome manifests itself through severe symptoms or not?

   As already mentioned, we observe a large variability in the data set. In particular, for the same patient, there are signals (in certain cell lines and differentiations) which correspond to a ventricular shape whereas others have an atrial shape. Due to this variability, we will break this question into two distinct parts:

   - **Handling Variability in the Data:** We will explore methods to handle the inherent variability in the data. The goal is to separate the dataset into subgroups according to the AP signals' shape. To this end, we will use a clustering method, whose input is generated by a dictionary of agnostic and electro-physiology derived features (biomarkers). The details are explained in Section 4.3.1.
   - **Distinguishing Severe and Non-Severe Syndrome Patients:** After dealing with this data variability problem, we will introduce an approach which could help to discriminate between patients with severe and non-severe forms of the syndrome. This step is performed by exploiting the clustering of the APs shapes, and a notion of similarity between signals. The results are presented in Section 4.3.2.

4.2 Binary classification between healthy and non-healthy individuals

Given an AP signal, we wish to classify it as healthy (labelled as 0) or non-healthy (labelled as 1). In this section, we present a comparison between two different classification methods: MLP and multivariate 1D-CNN.

For the classification based on MLP, we need to extract well-adapted biomarkers associated to the signals. The computation of these features is similar to the one presented in Chapter 3, Section 3.2.2. There are however some variations due to the difference of shapes between FP signals and AP signals. For instance, the depolarisation of AP (like Figure 1.1) is followed by the plateau phase which does not exist as is in the FP (like Figure 3.1 and 3.2). Therefore, some features are not computed separately for depolarisation and repolarisation phases.

More specifically, we compute a dictionary of features $\mathfrak{f} \in \mathbb{R}^9$ containing: action AP duration denoted as APD, amplitude (APA), amplitude in the centre (APC), maximum and minimum slope (APS_{max} and APS_{min}), 25%, 50%, 75%, and 90% of the area under

the AP curve ($APCT_{0.25}$, $APCT_{0.50}$, $APCT_{0.75}$, and $APCT_{0.90}$). In our classification scheme, we considered that obtaining a value above 0.5 classifies the cell line as unhealthy, whereas a value below 0.5 corresponds to a control cell line. The proximity of a prediction to either 1 or 0 indicated the model's level of certainty. For the architecture, the choice of the NN parameters and the training setup of the MLP, we considered the same as in Chapter 3, Section 3.3.2. Moreover, to ensure a robust evaluation, we randomly selected 20% of the entire feature data set as unknown samples in the test set.

The MLP model achieved an accuracy of 75.45% on correctly predicted controls and patients in the test set. However, upon closer examination, we observed that the average confidence levels, as depicted in Figure 4.2, were not particularly high for certain patient cell lines, namely BrS12C1, BrS12C4, and BrS8C3-D1. Furthermore, a noteworthy issue arose with patient BrS8C3-D2, as most of its samples were classified as controls.

Given the limitations in both accuracy and confidence levels, we explored an alternative method to improve our classification results.

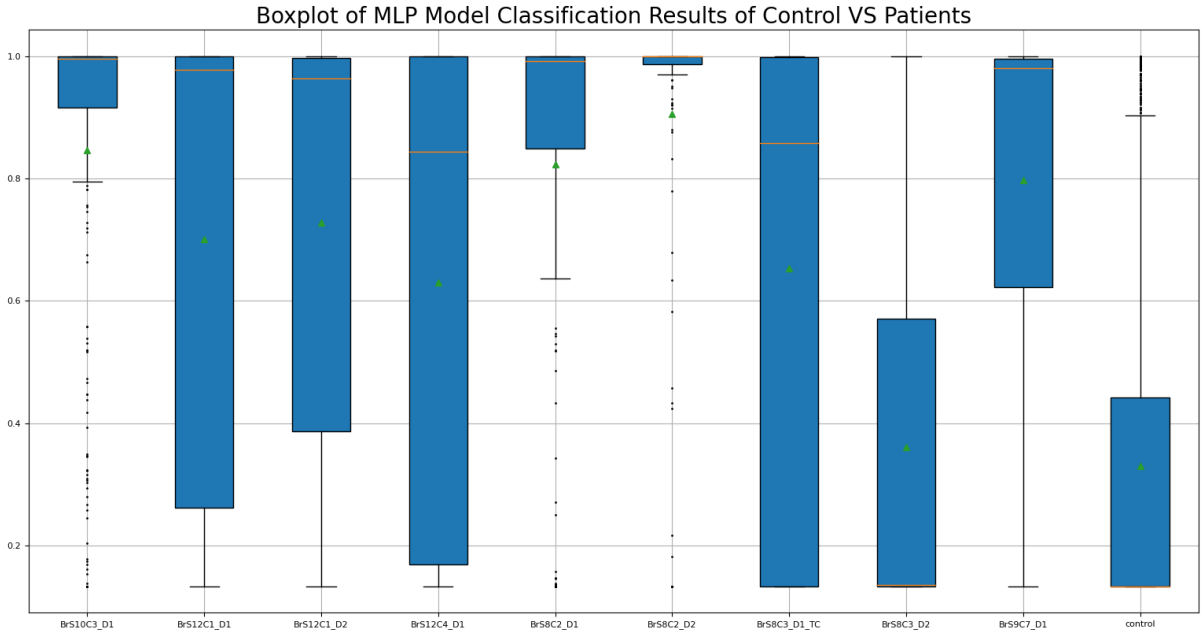


Figure 4.2: Results from MLP model to predict control versus patients

In our second approach, we employed the multivariate 1D-CNN method, which has the capability to use the action potential (AP) signals directly as input and autonomously extract features from the signals. This method can consider features which we did not take into account for the construction of the MLP input, thereby enhancing its ability to discriminate between controls and patients. The details of the multivariate 1D-CNN method, including the architecture, NN parameters and training setup are the same as the ones detailed in Chapter 3, Section 3.3.3.

In order to define the input of the multivariate 1D-CNN method, we organized the APs as an array $\mathfrak{S} \in \mathbb{R}^{6\times N_t}$, where each beat of the APs is $\mathfrak{s}_a \in \mathbb{R}^{N_t}$. In the case of controls, we randomly selected one trace and paired it with another control AP. This pairing process was repeated three times to create a matrix containing six APs. For patients, we randomly selected one trace and paired it with a control AP. Again, this pairing process was repeated three times to create a matrix with six APs.

To train the multivariate 1D-CNN model, we included 80% of all APs from control and patient cell lines. Then, the model was tested on 20% of the unknown APs. The results demonstrated 97% accuracy in predicting controls versus patients, as depicted in Figure 4.3. The model successfully classified all the patients with high confidence, except for patient BrS8C3-D2, for which the confidence level was lower and the number of misclassified samples was higher.

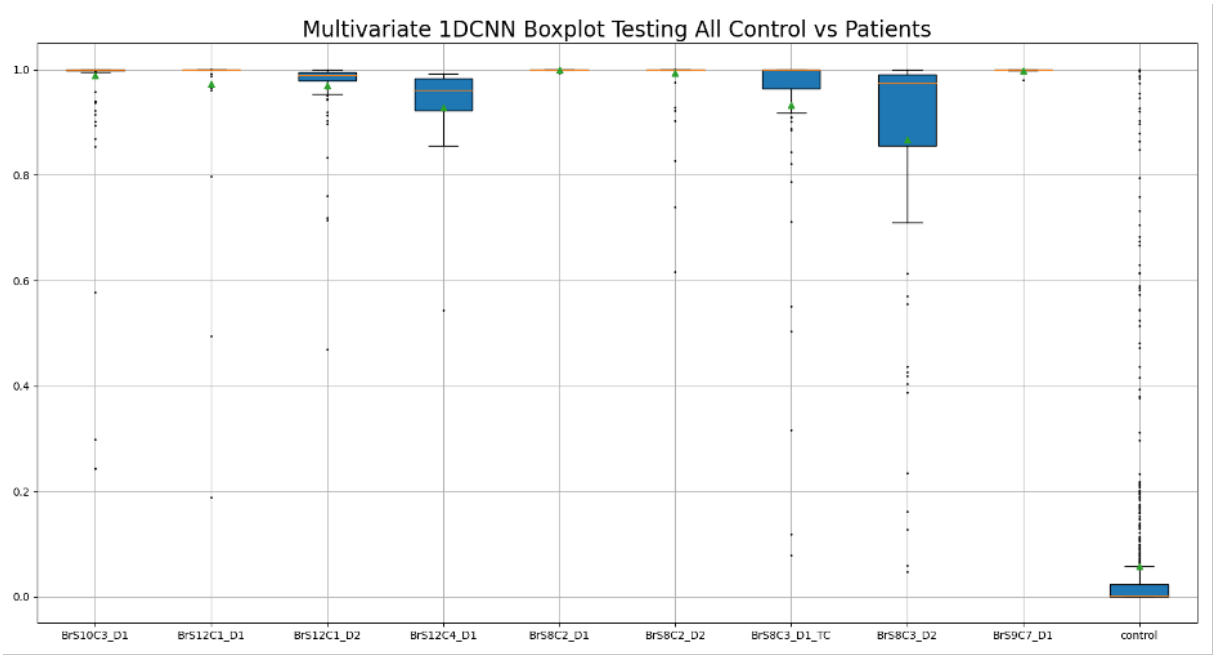


Figure 4.3: Results from multivariate 1D-CNN model to predict control versus patients

Subsequently, we extended the evaluation of the multivariate 1D-CNN model by testing its ability to predict unknown patients. This involved training the multivariate 1D-CNN model using a portion of a patients data and then using the trained model to predict the label corresponding to patients data which had never been used during the model training. This evaluation strategy allows us to assess the model capability to generalize and perform predictions on unknown data.

The results of these evaluations are presented in Figures 4.4, 4.5, and 4.6. The Figures are in agreement with our previous observations, indicating that the 1D-CNN model performs well in classifying the patients. It exhibits a high degree of confidence in accurately classifying patients (more than 90% accuracy), except for patient BrS8C3-D2, where, although the average confidence level remains above 0.5, there is a notable presence of misclassified samples and increased uncertainty. It seems to be related that, for this patient cell line, the impact of the genetic disorder is lower but further investigations are needed in order to better explain this difference.

To conclude this section, we noticed that MLP method could not provide a good classification between patients and healthy individuals whereas the multivariate 1D-CNN was able to achieve a high-level of accuracy with good confidence bounds. This points towards some questions that merit further investigation in the future. In particular, it would be interesting to study which features are used by the CNN and to understand their interpretation in order to complement the physiological biomarkers classically used in an MLP type of approach.

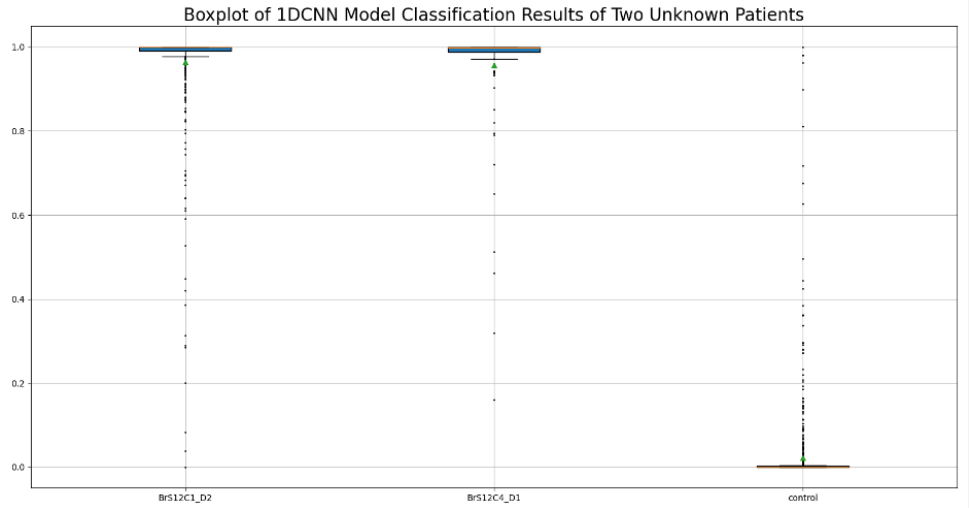


Figure 4.4: Results from multivariate 1D-CNN model to predict control versus unknown patients 1

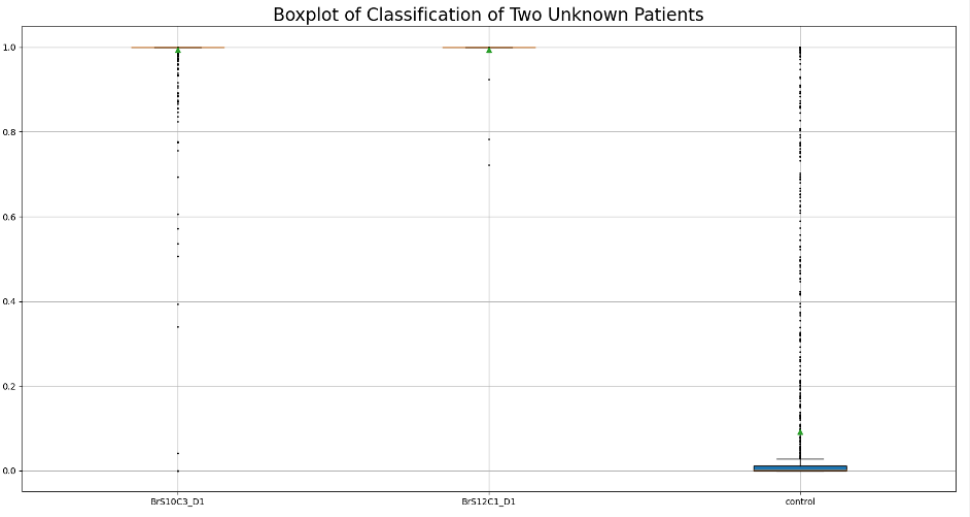


Figure 4.5: Results from multivariate 1D-CNN model to predict control versus unknown patients 2

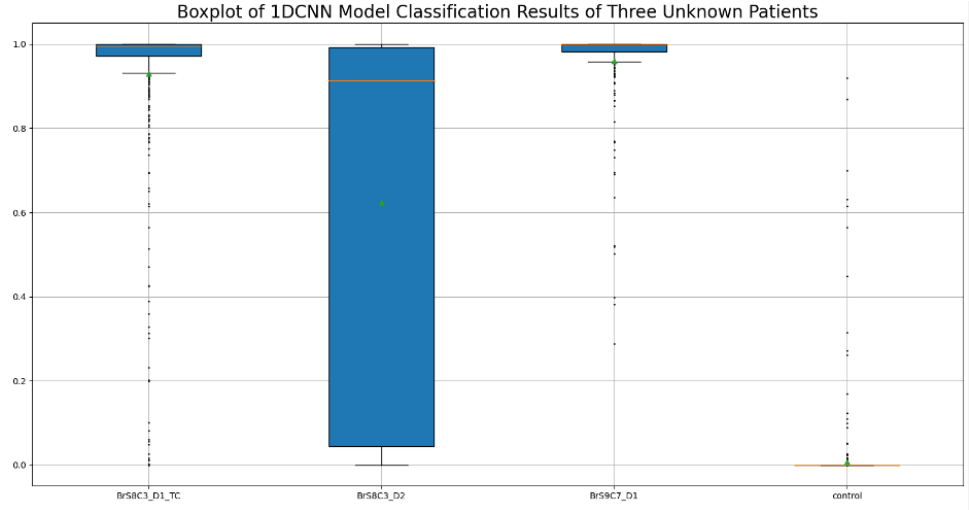


Figure 4.6: Results from multivariate 1D-CNN model to predict control versus unknown patients 3

4.3 Distinguishing patients with severe and non-severe symptoms

In this section, we will present a study on the classification between patients with a severe manifestation of the pathology and patients with mild effects. This section will be divided into two parts: a variability analysis on the control cell data and the identification of patients with severe and non-severe syndrome.

4.3.1 Data variability analysis

One of the difficulties we encountered when attempting to distinguish severe forms from less severe forms comes from the variability in the data set. The diverse range of AP shapes makes indeed this task tough. So in a first step, we analysed the "beat-to-beat" variability for the control group (cell lines and differentiation of cells derived from healthy individuals) by clustering the AP signals according to their shape.

We used the Density Based Spatial Clustering of Applications with Noise (DBSCAN) method, that was introduced in [161]. DBSCAN is a density-based clustering method, whose aim is to discover clusters of arbitrary size.

The first step of DBSCAN consists in defining a neighbourhood around each data point, according to a notion of distance, and a specified radius. It also requires that this neighbourhood contains at least a minimum number of data points. Then, the method identifies the centre of those neighbourhoods. This process is repeated until all centre points have been identified and included in clusters. The algorithm iterates through each centre point and expands the cluster by connecting centre points within their neighbourhoods. For the non-centre points, the method assigns them to a nearby cluster if the cluster is within the defined radius neighbourhood. Otherwise, the points are labelled as noise.

DBSCAN stands out for its ability to determine the number of clusters based on the given radius and minimum number of data points. It is robust in handling outliers, which

is suitable for our problem.

The inputs required for DBSCAN are the electro-physiology derived features mentioned for the MLP method presented in Section 4.2. The features were normalised from 0 to 1 by min-max normalisation. We denote by $\mathfrak{F} \in \mathbb{R}^{N_d \times 9}$ the array of features and define $\mathfrak{f}_{j_{min}} = \min_{1 \leq i \leq N_d} \{\mathfrak{F}_{ij}\}$ and $\mathfrak{f}_{j_{max}} = \max_{1 \leq i \leq N_d} \{\mathfrak{F}_{ij}\}$ where $1 \leq j \leq 9$. Then, the normalised features are defined by:

$$\hat{\mathfrak{F}}_{ij} = \frac{\mathfrak{F}_{ij} - \mathfrak{f}_{j_{min}}}{\mathfrak{f}_{j_{max}} - \mathfrak{f}_{j_{min}}}, 1 \leq i \leq N_d, 1 \leq j \leq 9$$

For the DBSCAN algorithm, we considered the Euclidean distance. Moreover, the hyperparameter $\mathfrak{d} > 0$ which corresponds to the maximum distance between two samples for one to be considered as in the neighbourhood of the other has been set to $\mathfrak{d} = 0.3$. At last, the minimum number of samples required in a neighbourhood for a point to be considered as a centre point was equal to 20.

However, the first results did not meet our expectations, as different shapes were assigned to the same cluster. This can be explained by the fact that very different shapes could have similar biomarkers. For the sake of brevity, we do not report these first results.

To correct this, we tried to enrich the dictionary of features to be used in the clustering process. To do so, we considered an autoencoder-based approach. In this case, the input of the autoencoder is now the full AP signals that we denote by $\mathfrak{s}_a$. The autoencoder has 5 hidden layers which have respectively 5000, 300, 20, 300, and 5000 hidden units. The autoencoder was trained to reconstruct the APs with small error ($RRMSE_{\mathfrak{s}_a} \leq 10^{-1}$). Then we computed the values of the latent variables $\alpha \in \mathbb{R}^{20}$, and we considered them as extra features.

The results from the clustering of control cell data are shown in Figure 4.7. We can see that the data set of control APs is divided into $N_c = 36$ different clusters and that this method is able to qualitatively group the APs on their shapes.

4.3.2 Severe and non-severe syndrome identification

The method that we propose to identify if a patient has severe or non-severe syndrome effects relies on the use of this clustering applied to control cell data. Having now an AP coming from a patient, we extract the features of the augmented dictionary presented before and we compute the distance between the patient features and the centres of every cluster to see to what extent the patient AP is far from the population of control APs. For each cluster, we compare the distance to a threshold (set at $\mathfrak{d} = 0.3 + 0.5$). If the distance is smaller than the threshold we can consider that this patient datum could belong to this cluster. If multiple clusters have a distance smaller than defined $\mathfrak{d}$, the patient datum could belong to the cluster which has the minimum distance. If so, this beat is somehow similar to part of the beats in the control population. On the contrary, if the sample does not belong to any of the control clusters, it will be considered as an outlier (labelled -1), as it is far apart with respect to all the control data.

In this section, we will present some results to classify the data recorded for two patients: BrS10 and BrS12.

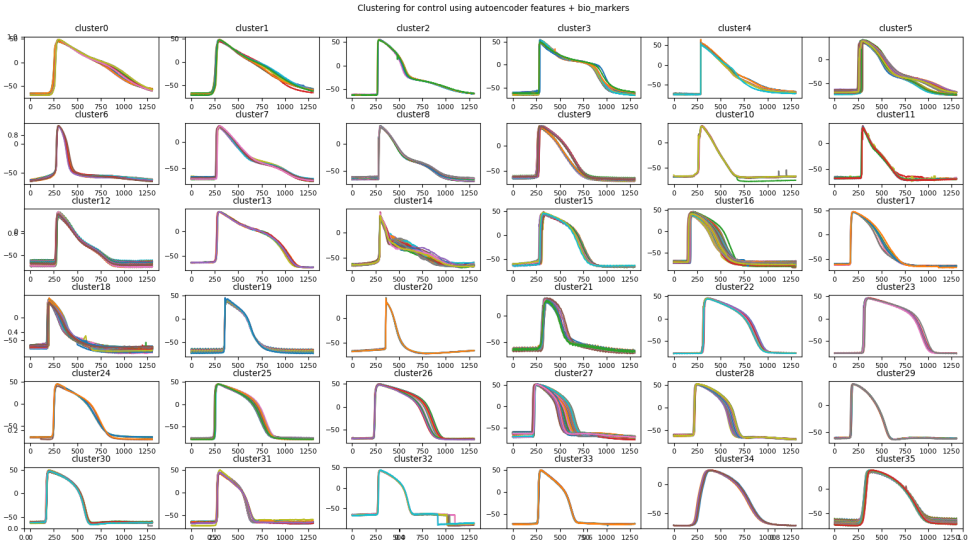


Figure 4.7: Clustering results for control cell lines

The results for these two patients are summarised in the Tables 4.1-4.2. In these tables, we listed the cluster number and the number of beats assigned to them. For BrS10, we can see that there are 248 beats, corresponding to 12.67% of the total number of beats, that have been labelled as outliers, which means not similar to any of the control APs.

For BrS12, there are 180 beats, corresponding to 25.86% of the total number of beats, that have been labelled as outliers. This ratio is much bigger with respect to the one for patient BrS10. This result is an indicator that for patient BrS12, there exist more APs that are not similar to the control data. Henceforth, it is plausible to infer that BrS12 is experiencing more severe symptoms with respect to BrS10. According to the prior information we got from the Antwerp team, patient BrS12 indeed shows more severe symptoms compared with patient BrS10.

Table 4.1: Clustering results of BrS10 respect controls

Cluster number	13	12	-1	5	9	15	33	25	16	19	7	32	10	29	11	17	26	8	27	28	24	2	35	1	34	21	23
number of beats	520	311	248	236	162	133	113	23	22	22	20	20	18	17	14	10	8	7	7	5	5	5	3	3	1	1	1

Table 4.2: Clustering results of BrS12 respect controls

Cluster number	-1	14	23	21	10	25	22	17	15	0	32	20	11	8	30	1	13
number of beats	180	116	108	69	53	48	28	15	14	14	14	11	10	9	5	2	1

This work is a preliminary investigation to distinguish severe syndrome effects from less severe syndrome effects on patients. The results on the two patients that have been analysed are encouraging. However, further tests are needed to understand whether the ratio of outliers can be directly related to the severity of the pathology, and how much it is affected by the level of variability in the experiments. This question has to be studied in collaboration with physiologists and pharmacologists who are working on the Brugada syndrome. A further step in the understanding of this syndrome consists in investigating

how the same gene mutation could have a different impact on the activity of different ion channels activity in cells. This will be explained in the next section.

4.4 Perspectives

As mentioned previously, to be able to conclude whether one patient is affected by more severe syndrome, we need to investigate the mechanisms related to the syndrome, and, in particular, which ion channels have been altered and to what extent.

To explore these questions, we can rely on a description of the electrophysiological mechanisms of cardiac cells by a mathematical model.

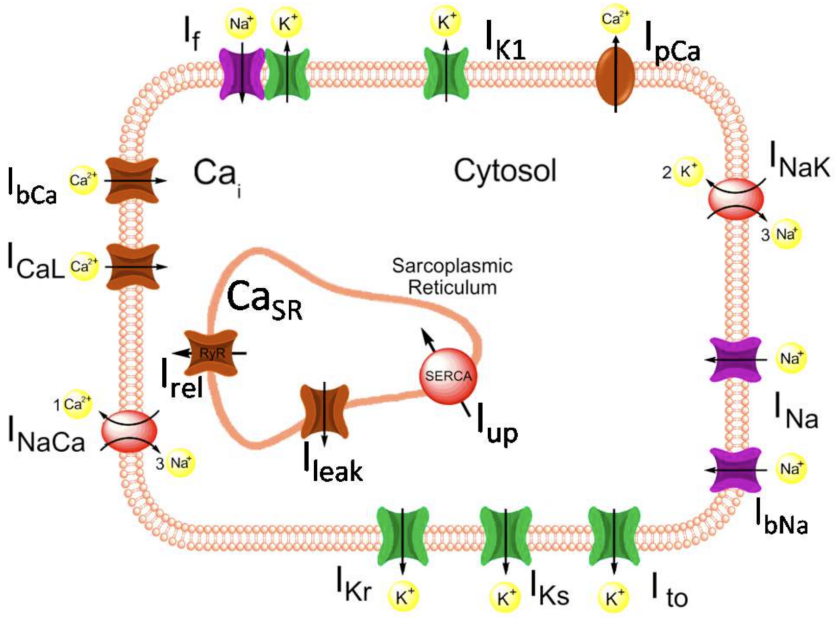


Figure 4.8: Diagram of ion channels in Paci model [21]

According to [21], Paci model has been introduced to describe the evolution of the action potential and the ion channel activities of hiPSC-CMs like in Figure 4.8. The parameters in each ion channel have a specific physical meaning. Henceforth, estimating the values of the parameters given a certain AP signal could provide some insight on the ion channel activities of the cells.

This parameter estimation can be carried by minimising the classical objective function which was stated in Chapter 2, Section 2.2.1, equation (2.2.5) and which, in our case, corresponds to the discrepancy between the experimental AP signal and the AP signal given by the model.

In these preliminary tests, we used CMAES [162] algorithm which is a commonly used global optimisation method already presented in Chapter 1, section 1.3. As prior information, we know that the majority of the patients carry an SCN5A mutation and SCN5A inhibits sodium channel Nav1.5 at a certain level. Since Paci model contains 49 parameters, working on the identification of all the parameters simultaneously is out of range. So, to begin with, we decided to focus on five parameters which we assume are the five most important parameters that may be influenced by genetic mutation: the

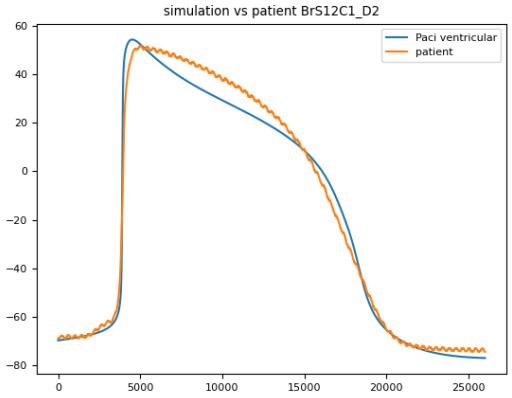


Figure 4.9: Ventricular like AP simulation result using Paci model

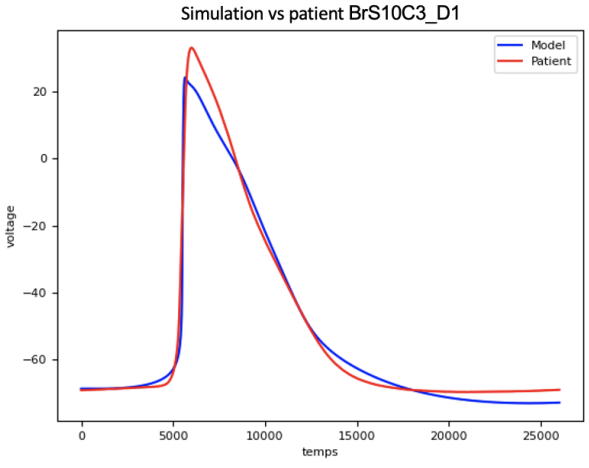


Figure 4.10: Atrial like AP simulation result using Paci model

gate of sodium g_{Na}, the calcium channel g_{Cal}, the potassium channel g_{Kr}, the gate of the background sodium channel $g_{b_{Na}}$ and hiPSC-CM membrane capacities C_m.

Then, we set the initial guess for CMAES method by considering the values reported in [21], except for g_{Na} and $g_{b_{Na}}$ where we take half the value given in [21]. Once obtained the values of the parameters given by CMAES method, we can perform a simulation and compare it to the data. An example is presented in Figure 4.9. In [21], the parameters for ventricular like AP are [3671.2, 8.64e−5, 29.8667, 0.9, 9.87109e−11]. The optimised parameters obtained by using CMAES are [3726.78, 3.69e−4, 57.92, 0.5076, 9.87e−11]. These optimised parameters were used to generate the simulation in blue. We observe that, compared to the original values of Paci model, we have the following behaviour for the parameters associated to the AP of BrS12C1_D2: an increase of g_{Cal} and g_{Kr} and a decrease in $g_{b_{Na}}$.

Another example is presented in Figure 4.10, in which we only try to identify $[g_{Na},\ g_{Cal},\ g_{Kr},\ g_{b_{Na}}]$. As an initial guess for the optimisation, we considered $[6646.185, 8.64\text{e}{-5}, 29.8667, 0.9]$ (which corresponds to the original suggested parameters for Atrial like AP in Paci model). In that case, the parameters obtained by CMAES method are $[8.84, -9.7287, 2.82, -0.45]$. We can observe, in this case, a decrease in g_{Na}, g_{Cal}, g_{Kr}, $g_{b_{Na}}$.

In both cases, we can see that the simulated APs do not completely fit to the experimental data.

These two calibrations provided encouraging results. However, they pointed towards some methodological limitations that have to be further investigated.

1. Firstly, due to the complexity of the models (for example Paci model), which involve a large number of parameters, the parameter estimation turns out to be a challenging task. The computational cost of using global optimisation methods could be prohibitive. In these preliminary tests, we selected five parameters were the effective ones. However, this may not be a good idea as it rules out the possibility to investigate some of the mechanisms described by the model. When we increase the number of parameters to optimise, it not only increases the computation costs but also the risk that CMAES method finds combinations of parameters which reduce the cost but correspond to non-physiological regimes or to situations in which the AP is close to the experimental one, but the internal state of the system is different. The cause of this is twofold: first, we do not have information about the identifiability of the parameters in the Paci model. Due to its complexity, an analytical study is, at present, out of reach. Methods of practical identifiability evaluation should be used. Second, for every parameter, physiological ranges should be specified, and eventual regularisation terms to be added to the cost function.

   The challenges of high-dimensional parameter estimation motivated the work presented in Chapter 1. To deal with the computation costs of global optimisation methods, like CMAES, or convergence issues of local optimisation methods, like gradient descent, the LVGF method has been proposed and tested on simpler ODE systems. The LVGF has shown some advantages compared with the gradient descent algorithm to minimise the classical objective function. LVGF is beneficial in estimating parameters which are varying and observable in a population.

2. As pointed out in [21], Paci model can be combined with some adult ionic current from other models to produce more variety in terms of AP shape. Only a single

model with a fixed ionic channel may have some limitations to be able to describe all possible shapes of AP. To consider various AP shapes in the experimental data, one idea is to include in the Paci model some ion channels considered in other electrophysiology models, as, for instance, the O'Hara-Rudy model (ORd model) which is a simulation model of undiseased human cardiac ventricular action potential done by ([19]).

3. A challenging aspect of the present study is related to the variability of the AP shapes. A first strategy to address the study of the mechanisms triggered by the Brugada syndrome consists in solving the parameter estimation for all individual cells and study how the values of the parameters differ in the healthy and in the patient populations. This involves a significant computational burden. Besides that, suitable comparison criteria should be introduced. Indeed, as there are different AP shapes, it is not straightforward to set up a comparison. Moreover, we do not know a prior how many different mechanisms can be triggered by the disease. In view of addressing this point, it could be interesting to consider population based approaches (a similar idea is mentioned in [163], [164], and [165])

Part 2: Comparison of Statistical, Machine Learning, and Mathematical Modelling Methods to Investigate the Effect of Ageing on Dog's Cardiovascular System

The work presented in this part has been published in an ESAIM proceeding (Vol.70) and is entitled: *Comparison of statistical, machine learning, and mathematical modelling methods to investigate the effect of ageing on dog's cardiovascular system.*

This work has been carried out in the CEMRACS 2021 summer school, and it is a collaboration with Elham Ataei Alizadeh (Boehringer Ingelheim Pharma GmbH & Co KG, Biberach an der Riss, Germany), Sara Costa Faya (Sorbonne Université and COMMEDIA team, Inria, Paris, France), my supervisors Damiano Lombardi and Sylvain Bernasconi, Pieter-Jan Guns (Laboratory of Physiopharmacology, University of Antwerp, Antwerp, Belgium) and Michael Markert (Boehringer Ingelheim Pharma GmbH & Co KG, Biberach an der Riss, Germany).

Abstract

The aim of this work is to provide a preliminary comparison of different classes of methods to automatically detect the effect of ageing from *in vivo* data. The application which motivated this work is related to safety pharmacology, whose major goal is to determine, in a pre-clinical phase, whether a drug is potentially dangerous for the health [1]. In particular, we are going to compare statistical, machine learning and mathematical modelling methods.

4.5 Introduction

This work has been motivated by some questions arising in safety pharmacology. Safety pharmacology studies are designed to identify and assess the potential clinical risk of undesirable drug properties before they enter clinical trials, as described in [166].

Many drug development processes must proceed through several stages to be sure for a product to be safe, efficacious, and has passed all regulatory requirements. The preclinical stage encompasses the use of *in vitro* and *in vivo* studies to develop a drug that can safely and effectively be administered for clinical trials. *In vivo* studies performed in animals are essential to drug development because they have the ability to evaluate the effects a drug has on a living organism. A particular care is taken in assessing adverse effects and drug-drug interactions that cannot be observed *in vitro* [167].

When an animal participates in an experiment in safety pharmacology studies, one can anticipate that the compound tested might have an effect on the organism. It is therefore essential to know when the animal can participate again in an experiment after a sufficient wash-out period. This is of particular importance in cross-over design studies (for details, the reader can refer to [168]). Before an animal will be used in a study, it has to undergo clinical evaluation as well as physiological tests to monitor the condition its cardiovascular system. When these initial tests are successfully performed the particular animal can be labelled as "healthy" and participate in the experiment. Age is one of the factors that has an impact on the function of the cardiovascular system. The effect

of age on cardiovascular function in laboratory dogs can be related to decreased blood flow, blood velocity, arterial compliance and distensibility, as well as increased ventricular systolic and diastolic stiffness as a result of prolonged duration of myocardial contraction phase (see [169], [170], and [171]). In this study, we are going to test the capabilities of several methods to detect the effect of age using some cardiovascular data collected from laboratory dogs. In addition to detecting ageing, we are interested in determining whether we can find unique individual fingerprints in the cardiovascular data of the dogs. Roughly speaking, this could provide some insight into whether ageing might impact interindividual variability.

Among all the questions that pharmacologists and physiologists have, we have tried to answer a pair of questions:

(Q_1) Can we assess if an animal is getting old? This question can be reformulated as follows: given its data at an initial time and assuming that its initial state is conforming to the experimental protocol how far its cardiovascular system is, at subsequent times, from this initial state?

(Q_2) Is it possible to identify an animal by its haemodynamic data by a computer algorithm?

Data about the cardiovascular activity of animals are available over several weeks taken by telemetry. For instance, each hour of recording can be a few gigabytes for each dog. Plus, there are variabilities across several weeks of recording. Therefore, it is necessary to use mathematical methods that can automatically analyse large data sets collected from those initial tests since it is not possible to analyse them manually.

This is a preliminary work to test different methods to answer these questions using a large data set from 4 dogs. Then, the methods can be verified and improved in future work.

4.5.1 Methods

Different viewpoints might be used to address these questions. In this work, we have compared statistical, machine learning, and mathematical modelling methods to analyse some *in vivo* cardiovascular data of 4 dogs. One of the main goals of the project is to assess which methods perform better on these kinds of tasks. In particular, we will compare:

1. Statistical methods: Physiologists and pharmacologists typically use statistical methods. This technique involves extracting a set of features from the signals and analyzing them statistically. As a first step, we will compute the empirical estimators mean, median, and other statistical criteria to determine if there is any significant difference between the cardiovascular function of young and old animals. Two-tailed Wilcoxon (Mann-Whitney) test will be presented to assess the effect of age on the individual features. The final step in the analysis will be the K-Means clustering (which could be interpreted as an unsupervised learning approach).

2. Machine learning: Given a database of signals and the outcome of the questions, we can build a map to learn a relationship between the data and the outcome. We will use artificial neural networks, which are typical machine learning algorithms.

They were used for instance in ECG analysis [172], cardiac arrhythmia prediction [173, 174], and drug safety studies [175, 176].

3. Mathematical modelling: By exploiting a priori information about the system, we build a set of equations to simulate the phenomenon under investigation. These equations provide a way to link observable quantities to the outcome. We will use a parametric 0-d model to simulate the global circulation (in the spirit of [177, 178, 179, 180, 181, 182, 183]). The parameters of the model, once calibrated, will make it possible to investigate the changes in the animals' cardiovascular system with age.

More precisely, we would like to analyse the advantages and disadvantages of each approach in terms of accuracy and computational cost.

4.5.2 Structure

In Section 4.6 we have presented the experimental data set we have used. In Section 4.7 we have described methods for analyzing the statistics. In Section 4.8 we have explained the Multilayer Perceptron (MLP) method that was implemented to detect the ageing of the dogs and distinguish the dogs. In Section 4.9 we have presented the model for the left part of the heart coupled with a model of the global circulation and the obtained results after calibration. Finally, some conslusions are given in Section 4.10.

4.6 Experimental Data Sets

In this project, we have used cardiovascular data gathered from 4 dogs: two Beagles (Hexe and Happy) and two Labrador-mix mongrel dogs (Simba and Roxy), involved in safety pharmacology studies for several years. During the study, dogs are in pairs in a group housing system. The dogs are calm and in resting mode during the study. A placebo injection was given to the dogs one hour after the study began. In order to avoid the effects of animal excitability during this period, the data during administration time have been excluded. Water is available to them *ad libitum*, and meals are provided after the study period ends. While avoiding all disturbances on the conscious animals is not possible due to the natural environment, the laboratory team managed to minimize cardiovascular disruption during data collection by creating a calm and regular environment and avoiding any intrusion or potential impact.

The data are acquired by telemetry from awake and non-anaesthetized animals for many hours (for more details, the reader can refer to [184]). This data set includes values of the Electrocardiogram (ECG), Arterial Pressure (AP), and Left Ventricular Pressure (LVP) signals, recorded every 2 milliseconds. For each dog, we had data corresponding to two different periods of the dog's life: the first one when the dog was 6-7 years old ("*Historical data*"), the second one when it was 8-9 years old ("*New data*"). We have used the cardiovascular data that has been recorded when the animal was younger and "healthy" state (we also call it "Historical" data) to compare with the recent recorded data, helping to decide whether or not this particular animal is "healthy". Each data file includes a seven hours continue recording of a placebo cardiovascular safety pharmacology study (pharmacologically inactive substances [185]).

Concerning the measurement of the arterial pressure, according to [186], the signal was sampled either by a catheter in the abdominal aorta or in the femoral artery. The LVP is measured by inserting a catheter connected to a fluid-filled transducer into the left ventricle [187]. From each complete cardiac beat (which in dogs at rest is around 0.8 s), we have extracted 9 quantities (those choices of 9 quantities have been decided by the data provider). All these features have been done by Notocord-Hem™ software. Human supervision and some manual corrections were needed to ensure the features are correctly extracted (especially for ECG). From the ECG we compute (see Figure 4.11):

1 QT interval in ms.

2 QRS interval in ms.

3 RR interval in ms.

4 PR interval in ms.

From the LVP signal we extract the following parameters:

5 Left ventricle systolic pressure, LVP systolic, measured in mmHg.

6 Left ventricle diastolic pressure, LVP diastolic, measured in mmHg.

7 Maximum of the left ventricular pressure, LVP dpdt(max), measured in mmHg s^{-1}.

From the AP signal we extract:

8 Systolic AP, measured in mmHg.

9 Diastolic AP, measured in mmHg.

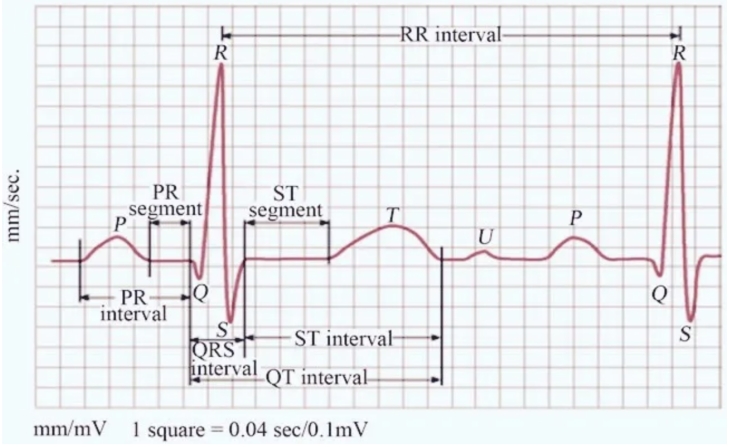


Figure 4.11: Cardiac parameter calculation from the raw ECG signal. See reference [188]

4.7 Statistical Analysis

In this section, we present the statistical analysis of the data. This has been performed by using solely the 9 features extracted from the telemetry data. In the next section, we will present the methods that have been used and their results.

4.7.1 Methodology

The data set consists of 9 cardiovascular features for every cardiac beat for a total amount of 10887 beats for the young and old animals. We had, henceforth, roughly $N_s = 10^4$ samples. The first analysis consists of computing the mean, first quartile, median, third quartile, and standard deviation for each feature individually by using empirical estimators. Due to the distribution-free nature of the data, a Mann-Whitney U test with a significance level of 0.05 was performed to confirm the results and interpretation provided by the statistical moments estimated (in the spirit of [189]). The goal was to understand whether age is influencing the features, individually. In Mann-Whitney test, the U_1 and U_2 values were computed as

$$U_1 = n_1 n_2 + n_1(n_1 + 1)/2 - R_1,$$
$$U_2 = n_1 n_2 + n_2(n_2 + 1)/2 - R_2.$$

In these formulas, n_1 and n_2 represent the sample sizes for the "Historical" sample and the "New" sample, and R_1 and R_2 represent the sum of the ranks for the "Historical" and the "New" cardiovascular sample, respectively.

As a second method, K-Means clustering has been used in order to determine whether cardiovascular functionality is unique and how age can affect this function (the reader is referred to [190, 191]). This algorithm was used to identify homogeneous subgroups within the data, such that the data points within each cluster are as similar as possible based on euclidean-based distance. Prior to applying the method, we have normalised the feature values and mapped them into the unit hypercube $[0, 1]^d$. As a similarity metric, we have used the standard $\ell^{2,d}$ norm (the Euclidean distance in renormalised space). The d value for this study is equal to 9 because we have nine cardiovascular features. The way to assign data points to clusters is to compute the squared distance between them and the cluster centroid (arithmetic mean of all the data points in a cluster) at a minimum.

To perform several tests with different purposes using the K-Means algorithm, we specify a different number of clusters that is estimated in all cases using the clustering gap method, based on [192]. Let $k \in \mathbb{N}$ be the number of clusters, we denote the Euclidean distance between two points D_r, the sum of all pairwise distances would be: $W_k = \sum_{r=1}^{k} \frac{1}{2n_r} D_r$. The gap statistic is defined as:

$$\mathrm{Gap}_n(k) = E_n^*\{\log(W_k)\} - \log(W_k),$$

where E_n^* is the expectation under a sample of size n from the reference distribution. Gap statistic is computed to estimate the most optimized K for clustering. The number of K is selected based on the overall behaviour of uniformly drawn samples, where the greatest jump in within cluster distance occurred. For each number of clusters k, the algorithm compares $\log(W(k))$ with $E_n^*\{\log(W_k)\}$ where the latter is defined via bootstrapping [193]. To eliminate the sampling noise from the data, the optimal K value will only be determined if the change is larger than the others. Figure 4.13 illustrates how K-Means clustering can estimate that we have two different data sets for the first data union of "Historical" and "New" data of one dog as an example.

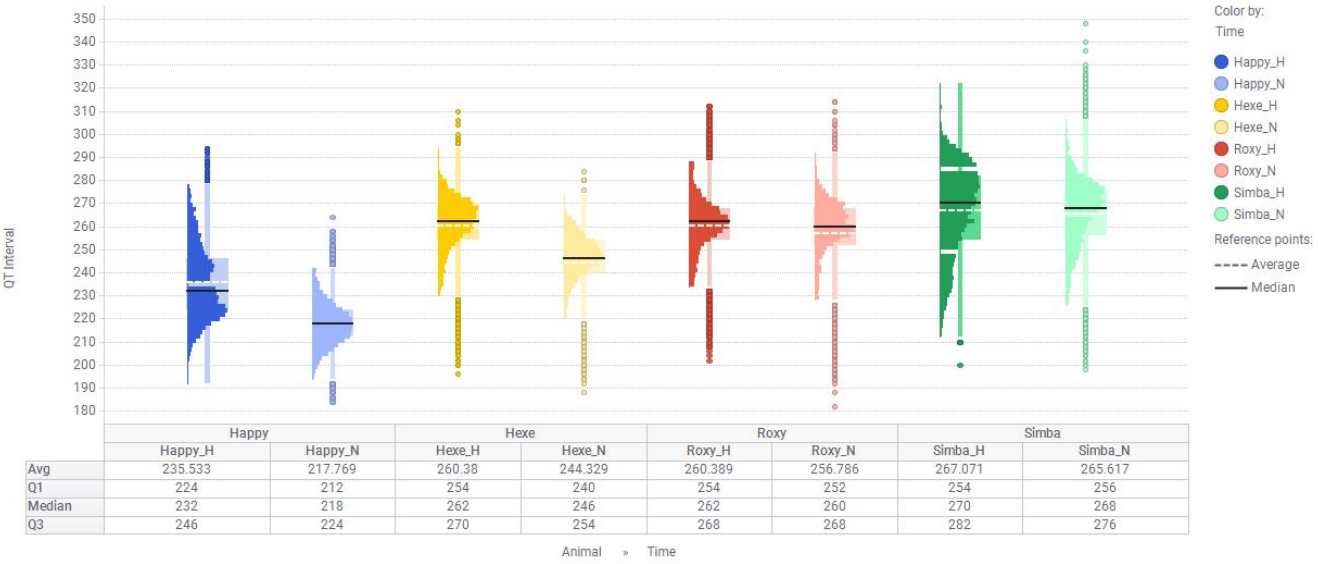
As a first step, we have considered, as a data set, the union of the "Historical" and "New" data for each dog. As the second data union, we have shuffled all the cardiovascular data of the dogs in both "Historical" and "New" data. W_k has been used to measure the

pertinence of the results derived from K-Means on this database.

In order to determine whether age dominates variability within an individual dog to a greater or lesser extent, we divided the first data union into two and the second data union into four groups.

4.7.2 Results from Statistical Analysis

For each cardiovascular feature, we have made box-plot charts to compare the changes over time between "Historical" and "New" data for each dog in order to dynamically analyze the effect of ageing on each cardiovascular characteristic. Figure 4.12 shows the box-plot charts[1] of the overall QT interval comparison between dogs as an example of the statistical calculation based on ECG features.



	Happy		Hexe		Roxy		Simba	
	Happy_H	Happy_N	Hexe_H	Hexe_N	Roxy_H	Roxy_N	Simba_H	Simba_N
Avg	235.533	217.769	260.38	244.329	260.389	256.786	267.071	265.617
Q1	224	212	254	240	254	252	254	256
Median	232	218	262	246	262	260	270	268
Q3	246	224	270	254	268	268	282	276

Figure 4.12: QT interval overview of all the dogs at two different ages.

Results answering Q_1

It has been found that the Mann-Whitney analysis has confirmed that ageing has a significant effect, which has a large and detectable impact on each cardiovascular feature as they relate to ageing.. These results are presented in Table 4.3.

To answer Q_1, we have provided Table 4.4 that shows the impact of ageing on each of our dogs on a case-by-case basis. In order to gain a clearer understanding of the extent to which each dog has been affected by ageing over time, it is necessary to average out the changes between "Historical" and "New" data. It has been observed that Simba's cardiovascular data has remained relatively stable over time. The cardiovascular feature values of Roxy are opposite to the ones of Happy and Hexe (we should also note that Happy and Hexe are Beagles, while Simba and Roxy are Mongrels).

[1] All the charts have been drawn by TIBCO Spotfire® platform.

Table 4.3: Two-tailed Mann-Whitney U test to determine the effects of ageing on each dog cardiovascular features

Cardiovascular Features	Happy		Simba		Roxy		Hexe	
	U value	p-value	U value	p-value	U value	p-value	U value	p-value
LVP Systolic	$6.19 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$2.83 \cdot 10^7$	$5 \cdot 10^{-3}$	$4.44 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$4.64 \cdot 10^7$	$2.2 \cdot 10^{-16}$
LVP Diastolic	$1.89 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$1.78 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$4.82 \cdot 10^6$	$2.2 \cdot 10^{-16}$	$1.56 \cdot 10^7$	$2.2 \cdot 10^{-16}$
LVP dpdt(max)	$7.47 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$2.95 \cdot 10^7$	$18 \cdot 10^{-15}$	$3.29 \cdot 10^7$	$1.4 \cdot 10^{-4}$	$1.08 \cdot 10^8$	$1.4 \cdot 10^{-4}$
QT duration	$7.51 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$3.04 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$3.27 \cdot 10^7$	$1.4 \cdot 10^{-3}$	$9.39 \cdot 10^7$	$2.6 \cdot 10^{-2}$
PR duration	$4.42 \cdot 10^7$	$1 \cdot 10^{-4}$	$3.17 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$2.11 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$5.66 \cdot 10^7$	$2.2 \cdot 10^{-16}$
RR duration	$5.65 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$2.82 \cdot 10^7$	$1.1 \cdot 10^{-2}$	$2.39 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$6.29 \cdot 10^7$	$2.2 \cdot 10^{-16}$
QRS duration	$3.39 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$2.53 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$4.97 \cdot 10^6$	$2.2 \cdot 10^{-16}$	$6.78 \cdot 10^7$	$2.2 \cdot 10^{-16}$
APR Systolic	$8.25 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$3.96 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$5.39 \cdot 10^6$	$2.2 \cdot 10^{-16}$	$4.98 \cdot 10^7$	$2.2 \cdot 10^{-16}$
APR Diastolic	$1.93 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$4.01 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$6.02 \cdot 10^7$	$2.2 \cdot 10^{-16}$	$4.36 \cdot 10^7$	$2.2 \cdot 10^{-16}$

Table 4.4: Total median of "Historical" and "New" cardiovascular data parameter for per each animal and their comparison.

Name	Data	LVP Systolic	LVP Diastolic	LVP dpdt(max)	QT duration	PR duration	RR duration	QRS duration	APR Systolic	APR Diastolic	Avg of all
Happy	Historical	130.73	-4.63	3173.83	232	126	708	40	136.52	90.79	
	New	125.61	-1.83	2746.58	218	126	572	42	123.62	104.47	
	New-Historical	-5.12	2.8	-427.25	-14	0	-136	2	-12.9	13.68	-64.0877
Simba	Historical	114.86	0.48	2075.2	270	114	1256	40	131.99	86.23	
	New	115.72	2.19	2075.2	268	112	1258	40	122.53	76.64	
	New-Historical	0.86	1.71	0	-2	-2	2	0	-9.46	-9.59	-2.0533
Roxy	Historical	98.75	-3.66	1434.33	262	120	1010	46	104.93	81.22	
	New	90.69	1.7	1403.81	260	128	1228	60	92.03	62.83	
	New-Historical	-8.06	5.36	-30.52	-2	8	218	14	-12.9	-18.39	19.2766
Hexe	Historical	117.55	-4.63	3021.24	262	122	978	44	128.62	84.32	
	New	120.48	1.586	2136.23	246	120	832	42	130.21	89.66	
	New-Historical	2.93	6.216	-885.01	-16	-2	-146	-2	1.59	5.34	-114.9926

In the next step, K-Means clustering has been examined on merged "Historical" and "New" data sets for each dog (independently) in order to determine whether or not ageing can comprehensively change the cardiovascular characteristics of individual animals. To avoid over-fitting, data points were grouped into chunks, and for each cardiovascular feature the median of 100 data points was computed.

As a positive outcome of the K-Means clustering test, the data were roughly clustered according to age. The number of samples that fall into the wrong cluster was computed by assuming "Historical" data are labeled cluster 1 and "New" data are labeled cluster 2. The evaluation of the results of this clustering has been shown in Table 4.5. This table shows the rates of wrong labeled K-Means clustering for each animal in each cluster of "Historical" and "New". Regarding to this table, the accuracy of K-Means clustering for distinguishing "Historical" and "New" data of Happy, Simba, Roxy, and Hexe are, respectively, 77%, 52%, 87%, and 94%.

Table 4.5: K-Means cardiovascular clustering of one animal in different ages.

	"Historical" wrong labeled / "Historical" cluster	"New" wrong labeled / "New" cluster	Total wrong labeled / Total number
Happy	45/106 = 0.42	0/86 = 0	45/192 = 0.23
Simba	52/84 = 0.61	20/66 = 0.30	72/150 = 0.48
Roxy	7/78 = 0.09	15/82 = 0.18	22/160 = 0.13
Hexe	0/105 = 0	14/103 = 0.13	14/208 = 0.06

Results answering Q_2

To check if it is possible to identify an animal by its haemodynamic data and answering Q_2, we have considered a data set containing data from two dogs. The purpose of this test is to examine inter- and intra-dog variability in order to determine if age is more relevant than individual cardiovascular characteristics. The input data was the union of 4 data sets, two dogs in two different ages. As soon as the data set is divided into

two clusters by the K-Means algorithm, it is automatically divided by two individual animals' cardiovascular data. This result illustrates that each individual animal has unique cardiovascular characteristics. As shown in Table 4.6, K-Means clusters the cardiovascular data of each individual dog. The average number of total incorrect classifications was 16 percent, which indicates an 84 percent success rate for distinguishing cardiovascular data between two individual dogs. The K-Means clustering method is thus capable of recognizing differences in age, individual animals, and individual files (each animal in each age group) with acceptable accuracy. Let us point out that if the number of clusters increases, or if the age, gender, and strain of animals are similar, then the accuracy of the K-Means clustering will be reduced.

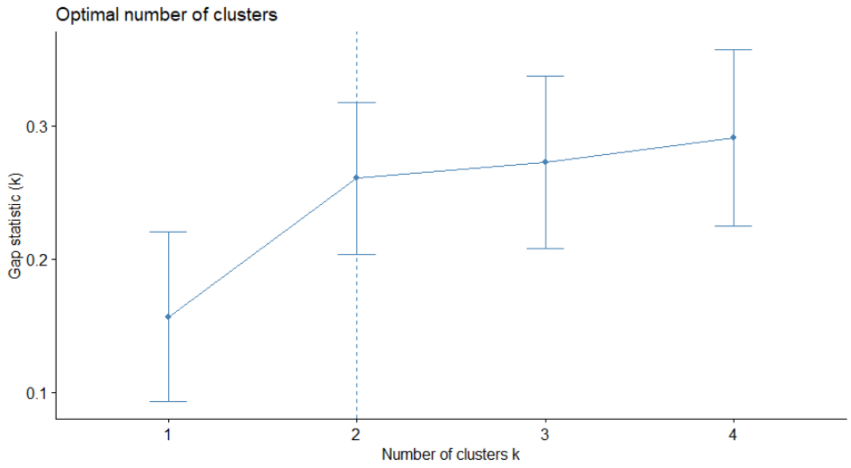


Figure 4.13: K-Means clustering indicates the optimal number of clusters using gap for "Historical" and "New" data combination file for one dog. Gap is within cluster distance, and number of cluster is considering possible cluster number.

Table 4.6: K-Means cardiovascular clustering of two animals in different ages.

Happy-Simba	Happy wrong labeled / Happy cluster	Simba wrong labeled / Simba cluster	Total wrong labeled / Total number
	$22/192 = 0.11$	$20/150 = 0.13$	$42/342 = 0.12$
Happy-Roxy	Happy wrong labeled / Happy cluster	Roxy wrong labeled / Roxy cluster	Total wrong labeled / Total number
	$0/192 = 0$	$16/160 = 0.1$	$16/352 = 0.04$
Happy-Hexe	Happy wrong labeled /Happy cluster	Hexe wrong labeled / Hexe cluster	Total wrong labeled / Total number
	$53/192 = 0.27$	$38/208 = 0.18$	$91/400 = 0.22$
Simba-Roxy	Simba wrong labeled / Simba cluster	Roxy wrong labeled / Roxy cluster	Total wrong labeled / Total number
	$0/150 = 0$	$46/160 = 0.28$	$46/310 = 0.14$
Simba-Hexe	Simba wrong labeled / Simba cluster	Hexe wrong labeled / Hexe cluster	Total wrong labeled / Total number
	$45/150 = 0.3$	$75/208 = 0.36$	$120/358 = 0.33$
Roxy-Hexe	Roxy wrong labeled / Roxy cluster	Hexe wrong labeled / Hexe cluster	Total wrong labeled / Total number
	$46/160 = 0.28$	$0/208 = 0$	$46/368 = 0.12$

4.8 Machine Learning Analysis

Considering the answer to Q_1, we would like to detect if the dogs are still healthy by comparing the "Historical" with the "New" data. We can see this problem either as a binary classification or a semi-supervised learning problem. Firstly, we have considered a binary classification. For each dog, the input consists of the set of cardiovascular features introduced in Section 4.6. Given these features, we would like to determine if they were

recorded at the younger age (class 0) or at the older age (class 1) for each dog separately. If we get a high classification score, we would conclude that their health conditions have changed. Otherwise, we would consider that they have a similar health condition as when they were young. We have used an MLP to perform this classification.

Secondly, if we consider the ageing assessment as a semi-supervised learning problem, we could use a Replicator Neural Networks (RNN) method on the cardiovascular signals. RNN is usually set up to perform anomaly detection in [194]: the RNN takes the raw signals as input and tries to reconstruct the input itself, as usually done in autoencoders [195]. In the present context, this would translate as follows: we train an RNN model on the "Historical" signals (normal case). Then we can use trained RNN to reconstruct test samples from "Historical" and "New" signals. By comparing the errors in the reconstructions, we would like to be able to classify whether the signal is coming from a young or old animal. Due to the small changes in the signals and the variabilities between signals, the RNN method is not very successful in performing the detection of dogs' age effects and for sake of brevity, we will not discuss any further the results in this article. However, we have tested the RNN method on another data set reported in the Chapter 3, section 3.7.

Moreover, we would also like to see if we can distinguish 4 dogs by using their cardiovascular features (answering Q_2). We can view this question as a multi-class classification problem (see [154]). We have tried to classify each dog to see if they are accurately classified in their class[2].

4.8.1 MLP Method used to Analysis *In vivo* Data

In this part, we have tested MLP to answer Q_1 and Q_2. According to [196], MLP is one of the most commonly used artificial neural networks in medical decision support to help analysing cardiovascular data. MLP consists of multiple layers which include the input layer to receive the features, several hidden layers which are the true computational engines of the MLP, and an output layer that produces a decision or prediction results. MLP is often applied to supervised learning problems especially binary classification.

We can train a set of hidden parameters of MLP that are able to learn the relationships between input-output. Then, the trained parameters can be optimized by Back-propagation which computes the gradient of the loss/error with respect to the weights in the network.

We have trained a MLP to discriminate between the "Historical" and "New" state of 4 dogs to answer the first question. "Historical" and "New" data correspond the two classes in the binary classification task. "Historical" and "New" data are mixed in the model training phase. We have used the data set consisting of around 10000 samples of cardiovascular features for each state of each dog to train the MLP model.

Those features are normalized using $MinMaxScaler$, explained in [197]. We have used 80% of the samples to train and validate (with cross validation), and 20% of samples to test the MLP model. Our MLP model consists of 3 fully connected hidden layers, which have 9, 6, and, 3 hidden units. The first hidden layer receives 9 cardiovascular features (LVP Systolic, LVP Diastolic, LVP dpdt(max), RR interval, PR duration, QT interval, QRS duration, APR Systolic, APR Diastolic). The outputs from each hidden layer are

[2]All the Machine Learning models were implemented in Python using $TensorFlow^{TM}$.

transformed by a ReLU function and the results in the output layer will be transformed to a decision boundary by a *sigmoid* function.

4.8.2 Results from MLP Method Answering Q_1

To evaluate the performance of the MLP method, we have computed the success rate which is defined as the number of correctly predicted samples divided by the total number of samples of the dogs as in Table 4.7. We have high success rates, 98.24%, 87.40%, 97.81%, and 98.33% to discriminate between the "Historical" and "New" state of our 4 dogs. As Simba has the lowest success rate than other dogs, maybe Simba has fewer changes in its cardiovascular performance compared to the other 3 dogs.

Table 4.7: Results from MLP model to discriminate "Historical" and "New" state of the dogs.

	Happy	Simba	Roxy	Hexe
Success Rate	98.24%	87.40%	97.81%	98.33%

4.8.3 Results from MLP Method Answering Q_2

We also would like to check if we can identify a dog by using an MLP, to perform a multi-class classification. According to [198], multiclass classification makes the assumption that each sample is assigned to one and only one label. We have labelled 4 classes Happy, Simba, Roxy, and Hexe. If there are significant differences in their cardiovascular features for each dog, we expect the model will have a high score to label them correctly, which means we can identify a specific dog among the 4. In the multi-class classification, we have used the same cardiovascular features as input. The difference between MLP used for binary classification and multi-class classification is in the output layer, the number of outputs being equal to the number of classes and the results will be transformed by a *softmax* function. In the training and testing phase, only "Historical" data of 4 dogs were used to train and test the multi-class classification model. From the results reported in Figure 4.14, we can conclude that over 93% of the time, we can identify one dog among 4 dogs and 7% of the time, we miss identifying the dog.

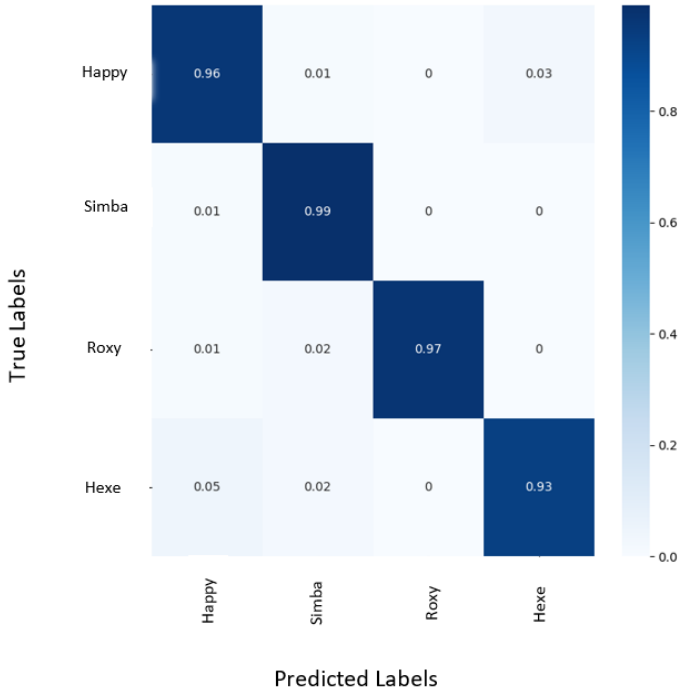


Figure 4.14: Classification results for identifying dogs using MLP method.

4.9 Mathematical Modelling Analysis

In this part of the project, we have considered a 0-d model (concerning 0-d models the reader is referred to [177, 178, 179, 180, 181, 182, 183]) for the left part of the heart coupled to systemic circulation which makes it possible to describe the main observable haemodynamics quantities and their evolution in time. Once the data are reproduced, we can calibrate the model by estimating the values of the parameters. The calibration is performed for the "Historical" state and for the "New" state of the animal and the values for the parameters in each situation will help to assess if the dog has changed significantly with age.

4.9.1 Analog Circuit Model for the Left Ventricle

The heart beat (concerning the physiology the reader is referred to [199]) is a two stage pumping action over a period of about one second subdivided in 2 stages: systole and diastole. During systole, the pressure in the left ventricle increases, exceeding the left atrium pressure. Then the mitral valve closes and the aortic valve opens and the blood flows into the aorta and out to the rest of the body. In diastole, the rate of contraction of the myocardium begins to slow and the aortic valve closes. When the ventricle relaxes, the pressure in the left ventricle falls and when it decreases below the pressure in the left atrium, the mitral valve opens and this lets blood flow from the left atrium to the ventricle.

The goal is to build a 0-d model of the left part of the heart coupled to system circulation by an electrical analogy [200].

The $0-d$ electric circuit model (see [183]) for the left ventricle is shown in Figure

4.15.

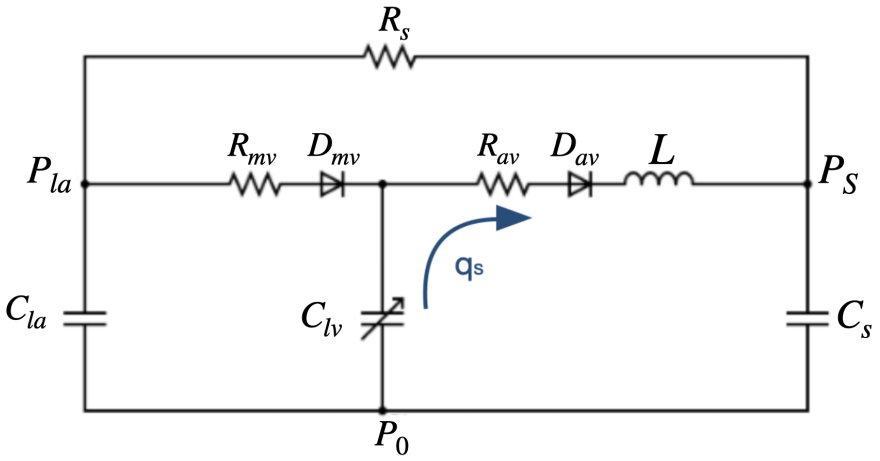


Figure 4.15: Electric circuit model.

The valves are represented by diodes in series with a resistor: R_{mv} and D_{mv} for the mitral valve and R_{av} and D_{av} for the aortic valve and R_s is the resistance of systemic circulation. The coil L represents the inertia of blood in the aorta. The compliances of the left atrium and systemic circulation are represented by C_{la} and C_s, respectively. We have used a time-varying left ventricular compliance $C_{lv}(t)$ to represent the action of the heart muscle. The elastance $E(t)$ is the reciprocal of the compliance ($E(t) = 1/C_{lv}(t)$) and it represents the contractile state of the left ventricle. It relates to the ventricle's pressure and volume (detailed in [201]) according to the expression:

$$E(t) = \frac{P_{lv}(t)}{V_{lv}(t) - V_0},$$

where $P_{lv}(t)$ is the left ventricular pressure, $V_{lv}(t)$ is the left ventricular volume and V_0 is a reference volume. The time variations in this model are due to the cyclical nature of the ventricle elastance and it changes as a function of time within one cardiac cycle. In our case, we are using the ECG to model the activation of the ventricle and, more precisely, the peaks of the QRS complex. The QRS complex (see Figure 4.11) corresponds to the electrical forces generated by ventricular depolarisation and represents the pumping action of the ventricles. From the ECG, we save the times at which the R peaks of the QRS complex occur and this is what we call the *activation times* of the ECG.

We approximated $E(t)$ by the following expression:

$$E(t) = \sum_{k=1}^{m} a \cdot [\tanh[b \cdot (t - w_k)] - \tanh[b \cdot (t - w_k - d)]] + h,$$

where $m \in \mathbb{N}^*$, a, b, d and h are constants, $t \in \mathbb{R}$ is the time and $w \in \mathbb{R}^m$ are the activation times of the ECG.

In order to get the equations for the circuit, three different cases have been considered:

- **Case 1: Filling.** The mitral valve is **opened** and the aortic valve is **closed** so the

left ventricle is being filled.

- **Case 2: Ejection.** The mitral valve is **closed** and the aortic valve is **opened** so blood is being ejected from the left ventricle.

- **Case 3: Isovolumic phase.** Both valves are **closed** and the capacitor has either just being filled (isovolumic contraction) or emptied (isovolumic relaxation).

Each phase of operation of the left ventricle can be modeled by a system of linear ordinary differential equations (ODEs). However, the whole system is non-linear because each diode has two states so we have three different cases (first diode is on and second one is off, first diode is off and second one is on, or both diodes are off). Each case is modeled by a different equivalent circuit since when a diode is off, it acts as a wire and we have open-circuit and when it is on, we have short-circuit.

The system of ODEs has been derived by applying Kirchoff's Current Law and Ohm's Law (the reader is referred to [202] for an overview on the topic) to the analog circuit model as follows:

CASE 1: If $P_{la}(t) > P_{lv}(t)$ and $P_{lv}(t) < P_s(t)$

For the first node, with left atrial pressure (LAP) $P_{la}(t)$, we have that

$$C_{la}\frac{dP_{la}}{dt} = \frac{P_{lv} - P_{la}}{R_{mv}} - \frac{P_{la}}{R_s}.$$

For the one in the middle with value $P_{lv}(t)$

$$C_{lv}\frac{dP_{lv}}{dt} + P_{lv}\frac{dC_{lv}}{dt} - P_0\frac{dC_{lv}}{dt} = \frac{P_{la} - P_{lv}}{R_{mv}},$$

and since $E_{lv}(t) = 1/C_{lv}(t)$, we can rewrite the previous expression as

$$\frac{dP_{lv}}{dt} = E\left(\frac{P_{la} - P_{lv}}{R_{mv}}\right) + \frac{P_{lv}}{E}\frac{dE}{dt} - \frac{P_0}{E}\frac{dE}{dt},$$

where $P_0 \in \mathbb{R}$ is just a reference value. Then, for the node on the right, with arterial pressure $P_s(t)$,

$$C_s\frac{dP_s}{dt} = \frac{P_{la} - P_s}{R_s},$$

and, for the aortic flow $q_s(t)$

$$\frac{dq_s}{dt} = 0.$$

The system of equations for the filling phase is then

$$\begin{cases} C_{la}\dfrac{dP_{la}}{dt} &= \dfrac{P_{lv}-P_{la}}{R_{mv}}-\dfrac{P_{la}}{R_s}, \\ \dfrac{dP_{lv}}{dt} &= E\left(\dfrac{P_{la}-P_{lv}}{R_{mv}}\right)+\dfrac{P_{lv}}{E}\dfrac{dE}{dt}-\dfrac{P_0}{E}\dfrac{dE}{dt}, \\ C_s\dfrac{dP_s}{dt} &= \dfrac{P_{la}-P_s}{R_s}, \\ \dfrac{dq_s}{dt} &= 0. \end{cases}$$

CASE 2: If $P_{lv}(t) > P_{la}(t)$ and $P_{lv}(t) > P_s(t)$
Proceeding in an analogue way, we get the following system of ODEs

$$\begin{cases} C_{la}\dfrac{dP_{la}}{dt} &= \dfrac{P_s-P_{la}}{R_s}, \\ \dfrac{dP_{lv}}{dt} &= \dfrac{P_{lv}}{E}\dfrac{dE}{dt}-Eq_s-\dfrac{P_0}{E}\dfrac{dE}{dt}, \\ C_s\dfrac{dP_s}{dt} &= \dfrac{P_{la}-P_s}{R_s}+q_s, \\ \dfrac{dq_s}{dt} &= \dfrac{P_{lv}-P_s}{L}-\dfrac{q_s}{L}R_{av}. \end{cases}$$

CASE 3: If $P_{lv}(t) > P_{la}(t)$ and $P_{lv}(t) < P_s(t)$
In this case, both valves are closed (both diodes are off) and there is no current flow. Then the equations are

$$\begin{cases} C_{la}\dfrac{dP_{la}}{dt} &= \dfrac{P_s-P_{la}}{R_s}, \\ \dfrac{dP_{lv}}{dt} &= \dfrac{P_{lv}}{E}\dfrac{dE}{dt}-\dfrac{P_0}{E}\dfrac{dE}{dt}, \\ C_s\dfrac{dP_s}{dt} &= \dfrac{P_{la}-P_s}{R_s}, \\ \dfrac{dq_s}{dt} &= 0. \end{cases}$$

4.9.2 Results from Mathematical Modelling

To solve the equations in each case according to time we have used a backward differentiation formula (BDF) solver that is implemented in the Python built-in solver *odeint* (see [203] and [204] for more details). In order to solve the model, we need to give as **input** the parameters $(R_s, R_{mv}, R_{av}, C_a, C_s, L, P_0, a, h)$ and the peaks of the ECG. Then we get as **output** the LVP, the AP, the LAP and the aortic flow.

To obtain the optimized parameters for each dog we have used the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) optimization algorithm that is implemented in Python in the *pycma* module [162]. We need to give as **input** an initial estimation of the parameters and the experimental data for the AP, LVP and ECG. The initial standard

deviation was chosen as $\sigma_0 = 0.4$ and we have used a population size 1000 times larger than the default value. The objective function needs also to be specified. We have chosen the following function to be minimized

$$J = \sqrt{\sum_{i=1}^{n} \left| P_{lv}^{(i)} - \hat{P_{lv}}^{(i)} \right|^2 + \left| P_s^{(i)} - \hat{P_s}^{(i)} \right|^2},$$

where $P_{lv}^{(i)}$ is the experimental value of the left ventricle pressure at each time, $\hat{P_{lv}}^{(i)}$ is the predicted value of the left ventricle pressure at each time, $P_s^{(i)}$ is the experimental value of the arterial pressure at each time, $\hat{P_s}^{(i)}$ is the predicted value of the arterial pressure at each time and $n \in \mathbb{N}^*$.

The **output** of the CMA-ES algorithm are the optimized parameters of the model.

Results answering Q_1

For each dog, we have estimated the parameters for the "Historical" and "New" state. Table 4.8 shows the optimized model parameters for each dog at both times. The solution of the model for its corresponding parameters for each dog is plotted in Figures 4.16, 4.17, 4.18 and 4.19. Although the heart pace can look irregular in Figures 4.16-4.19, the signal cannot be considered arrhythmia since dogs have a pronounced vagal tone which leads to this "normal" unnormal rhythm and this has no clinical relevance.

We have computed the difference between the values predicted by our model and the values observed. For that purpose, we have found the local maxima of the LVP and we have computed the difference between the real signal and the simulated one at each peak (local maxima) for the LVP. The relative error is

$$\xi(P_{lv}) = \sqrt{\frac{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\frac{1}{n}\sum_{i=1}^{n}(y_i)^2}},$$

where y is the value of the pressure at each peak in the true signal, $\hat{y}$ is the value in the predicted signal and $n \in \mathbb{N}^*$. For the AP we have taken into account not only the difference between maxima but also the one between minima

$$\xi(P_s) = \sqrt{\frac{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\frac{1}{n}\sum_{i=1}^{n}(y_i)^2}} + \sqrt{\frac{\frac{1}{n}\sum_{i=1}^{n}(w_i - \hat{w}_i)^2}{\frac{1}{n}\sum_{i=1}^{n}(w_i)^2}},$$

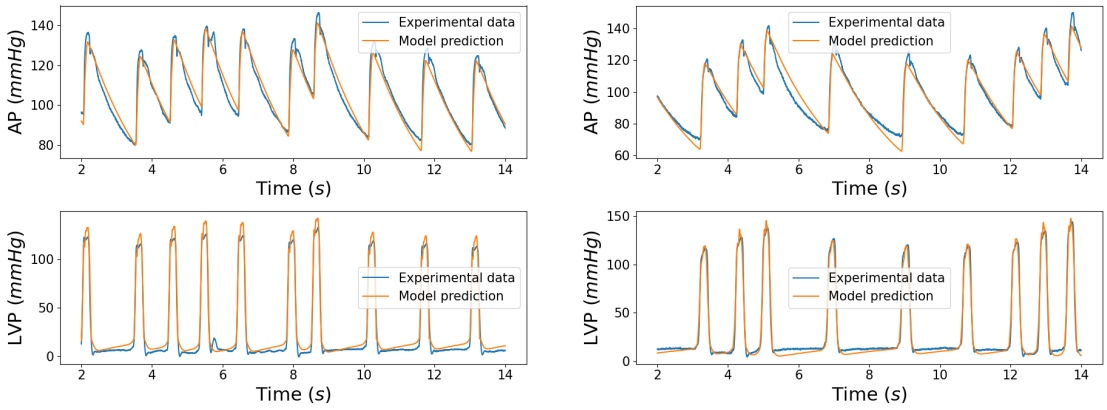
where w is the value of the pressure at each minimum peak in the true signal, $\hat{w}$ is the value of the pressure at each minimum peak in the predicted signal and $n \in \mathbb{N}^*$.

We want to address if the animals are getting old or not by the change in the parameters in the electrical model. To check that, we have made the following steps:

1. We have computed the optimized parameters of the model for each dog in the "Historical" state and also in the "New" state (they are shown in Table 4.8).
2. Afterwards, we have run the parameter optimization in the 0-d model for each dog in the "New" state considering as initial guess the optimized parameters for the "Historical" state. The purpose of doing this is to see if having as initial test the parameters of the "Historical" state, the optimizer could find good values for the

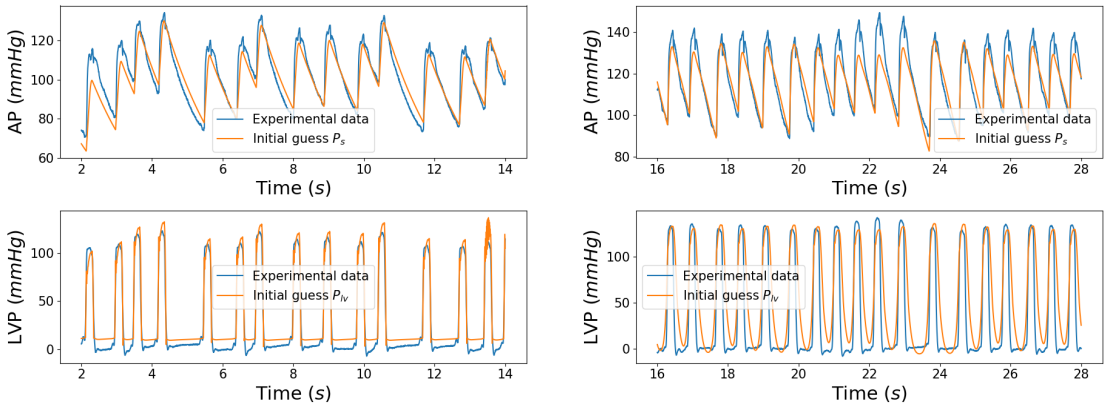
Table 4.8: Model parameters (from left to right) for Happy "Historical" (2015), Happy "New" (2018) , Simba "Historical" (2018), Simba "New" (2021), Roxy "Historical" (2018), Roxy "New" (2021), Hexe "Historical" (2018) and Hexe "New" (2020). The relative error for the LVP and for the AP is also shown for each dog in both states ("Historical" and "New").

Parameters	Happy "H"	Happy "N"	Simba "H"	Simba "N"	Roxy "H"	Roxy "N"	Hexe "H"	Hexe "N"
R_s [mmHg · s/cm^3]	$1.62 \cdot 10^{-3}$	$1.20 \cdot 10^{-1}$	$8.06 \cdot 10^{-2}$	$2.67 \cdot 10^{-1}$	8.78	3.87	$1.45 \cdot 10^{-3}$	$6.26 \cdot 10^{-2}$
R_{mv} [mmHg · s/cm^3]	$2.06 \cdot 10^{-6}$	$1.75 \cdot 10^{-2}$	$5.12 \cdot 10^{-5}$	$1.47 \cdot 10^{-4}$	$1.26 \cdot 10^{-1}$	$1.12 \cdot 10^{-2}$	$7.02 \cdot 10^{-7}$	$7.38 \cdot 10^{-3}$
R_{av} [mmHg · s/cm^3]	$1.85 \cdot 10^{-5}$	$1.18 \cdot 10^{-3}$	$6.33 \cdot 10^{-4}$	$2.45 \cdot 10^{-3}$	$3.57 \cdot 10^{-4}$	$8.18 \cdot 10^{-4}$	$3.55 \cdot 10^{-6}$	$2.95 \cdot 10^{-4}$
C_a [cm^3/mmHg]	$3.65 \cdot 10^{4}$	7.01	$1.36 \cdot 10^{2}$	$5.18 \cdot 10^{1}$	$2.61 \cdot 10^{-1}$	1.72	$4.59 \cdot 10^{4}$	$1.0808 \cdot 10^{1}$
C_s [cm^3/mmHg]	$1.21 \cdot 10^{3}$	9.39	$2.99 \cdot 10^{1}$	9.27	$3.26 \cdot 10^{-1}$	$9.67 \cdot 10^{-1}$	$1.71 \cdot 10^{3}$	$1.63 \cdot 10^{1}$
L [mmHg · s^2/cm^3]	$1.68 \cdot 10^{-7}$	$9.18 \cdot 10^{-6}$	$1.1151 \cdot 10^{-5}$	$5.26 \cdot 10^{-5}$	$1.98 \cdot 10^{-4}$	$1.60 \cdot 10^{-4}$	$6.30 \cdot 10^{9}$	$3.41 \cdot 10^{-6}$
P_0 [mmHg]	$1.66 \cdot 10^{4}$	$6.41 \cdot 10^{1}$	$3.05 \cdot 10^{3}$	$7.71 \cdot 10^{2}$	$1.63 \cdot 10^{-1}$	$9.52 \cdot 10^{-1}$	$2.25 \cdot 10^{3}$	$3.16 \cdot 10^{1}$
a [mmHg/cm^3]	$9.08 \cdot 10^{-3}$	$6.90 \cdot 10^{-1}$	$9.27 \cdot 10^{-2}$	$1.95 \cdot 10^{-1}$	$1.07 \cdot 10^{1}$	2.39	$5.36 \cdot 10^{-3}$	$2.87 \cdot 10^{-1}$
h [mmHg/cm^3]	$1.12 \cdot 10^{-2}$	0.00	$6.92 \cdot 10^{-2}$	$6.07 \cdot 10^{-2}$	$3.36 \cdot 10^{-3}$	$3.87 \cdot 10^{-3}$	$1.23 \cdot 10^{-3}$	$1.40 \cdot 10^{-6}$
Relative errors								
$\xi(P_s)$	10.83 %	10.56 %	6.39 %	9.39 %	5.22 %	8.53 %	10.13 %	9.51 %
$\xi(P_{lv})$	8.38 %	4.11 %	8.94 %	3.99 %	3.51 %	7.68 %	6.17 %	4.17 %



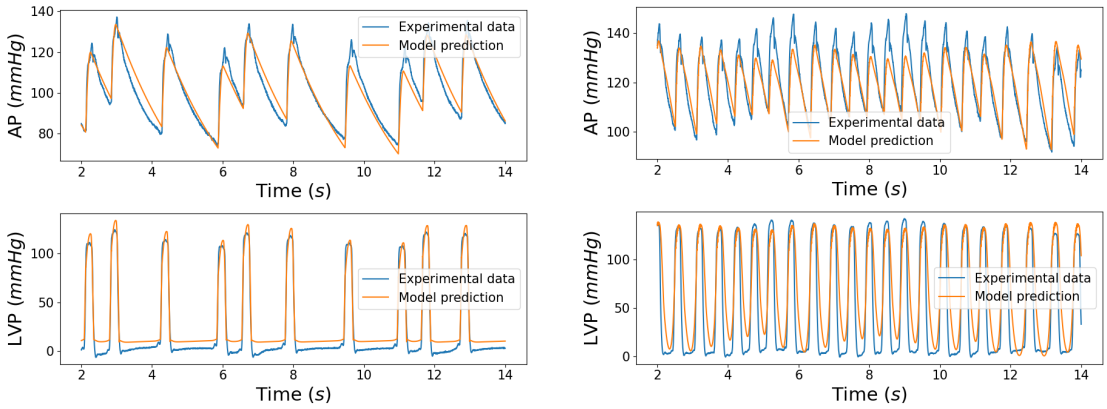
(a) AP and LVP for Simba 2018 ("Historical") (b) AP and LVP for Simba 2021 ("New")

Figure 4.16: Experimental data compared with the model prediction for Simba.



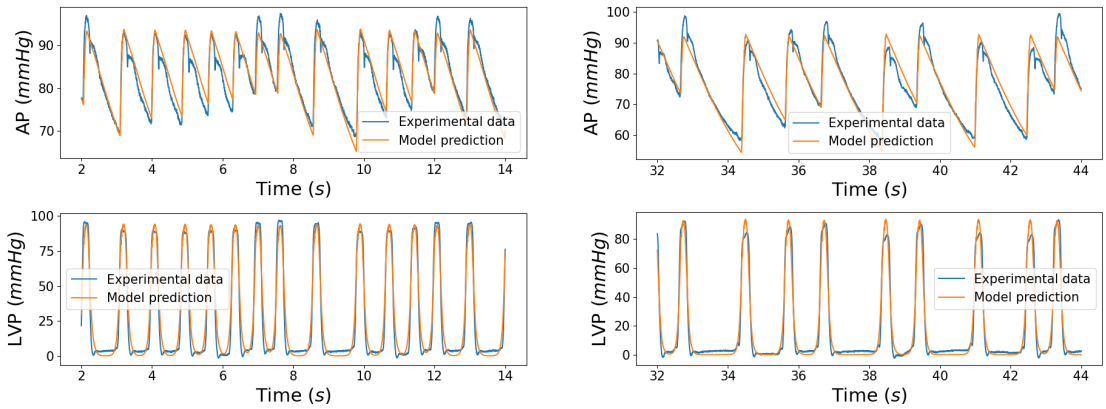
(a) AP and LVP for Happy 2015 ("Historical") (b) AP and LVP for Happy 2018 ("New")

Figure 4.17: Experimental data compared with the model prediction for Happy.



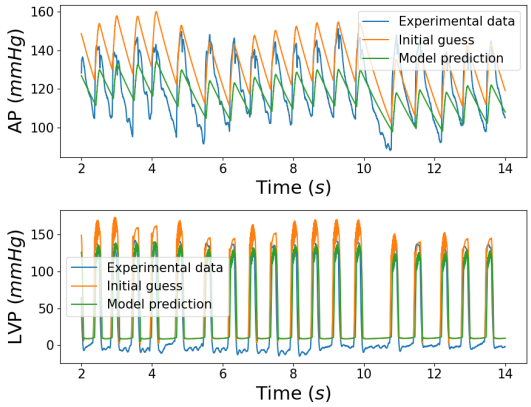
(a) AP and LVP for Hexe 2018 ("Historical") (b) AP and LVP for Hexe 2020 ("New")

Figure 4.18: Experimental data compared with the model prediction for Hexe.

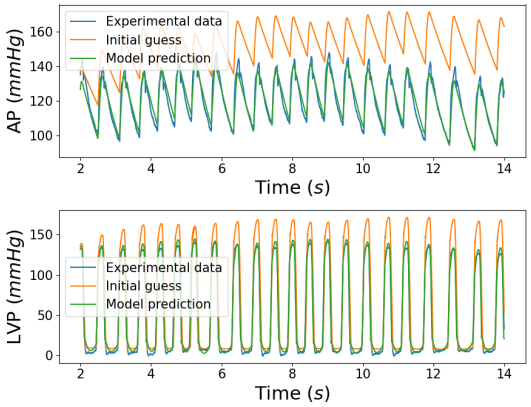


(a) AP and LVP for Roxy 2018 ("Historical") (b) AP and LVP for Roxy 2021 ("New")

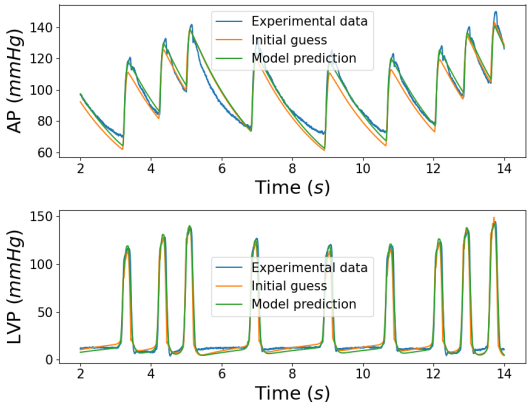
Figure 4.19: Experimental data compared with the model prediction for Roxy.



(a) AP and LVP for Happy 2018 starting with parameters of Happy 2015



(b) AP and LVP for Hexe 2020 starting with parameters of Hexe 2018



(c) AP and LVP for Simba 2021 starting with parameters of Simba 2018

Figure 4.20: Experimental data compared with the model prediction when we start the optimization from the optimized "Historical" parameters as initial guess.

"New" state. The model prediction after the optimization is shown in green in Figure 4.20. The initial guess is shown in orange in the same Figure.

3. We have also directly solved the model in the "New" state, with the parameters of the "Historical" state. The purpose is to see if we can match the real data in the "New" state with the optimized parameters of the "Historical" state (without running the optimization). This is plotted in Figure 4.20 in orange.

After repeating the simulations for several chunks of beats, we have observed that Simba barely changes. With the optimized parameters in the "Historical" state we can reproduce the data in the "New" state. However, we have observed significant changes for Happy, Roxy and Hexe. In Happy and Hexe we have observed that as the dogs are getting older the resistances increase and the capacitances decrease as a general tendency. The capacitance gives us the ability of the vessels to get elastic and since the capacitances decrease, the vessels become less elastic with the years. On the contrary, we have observed in Roxy a different tendency. Roxy has experienced a decrease in the resistances and an increase in the capacitances which is the opposite behaviour as in Hexe and Happy and may be interpreted as her features have improved with age.

To answer properly Q_1, we need to set a threshold of the error that determines that the animal is changing with age. In other words, at which relative error do we claim that the "Historical" parameters do not fit the "New" state because the dog has changed? To establish that threshold, we have solved the model for each dog for several chunks with the parameters of Table 4.8. The maximum of the error that we got for each animal, is going to be our threshold to claim whether the dog experienced changes with age or not. The thresholds to determine if the model is fitting the data are 14 % (Happy), 15 % (Simba), 16 % (Roxy) and 13 % (Hexe).

To study the accuracy of the mathematical modelling method answering Q_1, we have solved the model for each dog with our calibrated parameters in other chunks of beats than the ones used for setting the threshold. First, we have considered chunks in the "Historical" state and we have used the "Historical" parameters to solve the model (same applies for the "New" state). If we get a relative error that is less than the threshold, then it means that the model has worked. But if we get a relative error that is bigger than the threshold, it means that the model suggests that the dog changed but it is a mistake since we are just using the calibrated "Historical" parameters but for a different chunk. We have repeated this computation for a certain number of chunks and we have computed the success rate (number of correctly predicted chunks divided by the total number of tested chunks), that was always higher than 95 %.

Therefore, we can conclude that we are able to identify from the parameters if a dog is changing with age or not.

Results answering Q_2

The second question that we wanted to answer is if we can discriminate between dogs. If we look at the optimized parameters of the model that are shown in Table 4.8, we can see that they substantially change between different dogs. We have solved the model at a given state ("New" or "Historical") with the data of a given dog but using the optimized parameters for the same state but for another dog. The result of doing that for every dog, was that the relative error in the left ventricular pressure ($\xi(P_{lv})$) was sometimes below

the values of the thresholds but the value for the relative error in the arterial pressure ($\xi(P_s)$) was always above 32 % (higher than the thresholds).

Although it would be necessary to do the study with a larger number of chunks in order to withdraw a strong conclusion, it is already possible to spot some differences between the dogs since the calibrated parameters for one dog can never reproduce the data of a different one. Therefore, we can claim that it is possible to identify an animal by its haemodynamic data after doing the calibration of the model since with the optimized parameters of a dog, it is not possible to reproduce the data of another one.

4.10 Conclusion and Discussion

In this paper, we have tried to analyse some *in vivo* experiments data for addressing the effect of ageing in an animal by using statistical, machine learning and mathematical modelling methods. On the whole, we are able to identify some changes in dogs' cardiovascular data in the "New" state compared to the "Historical" state. Moreover, we can discriminate not only between "New" and "Historical" state of the dog, but also distinguish one dog among the others.

In terms of accuracy, cost, and robustness, we have compared the advantages and disadvantages of each method. Although the way how we answer the questions is very different in mathematical modelling and in statistical and machine learning approaches, we can set a link on how to compare the accuracy of the methods. With respect to the answer to question Q_1 in the statistical method, changes between the "New" and "Historical" state of cardiovascular data for each dog in a positive or negative direction are always measurable, as shown in the statistical results. However, the K-Means clustering method could recognize and cluster these changes for Happy, Simba, Roxy, and Hexe with an accuracy of 77%, 52%, 87%, and 94%, respectively. Machine learning method can correctly (above 97% for Happy, Roxy, and Hexe) distinguish between the "New" and "Historical" state of the dog. When "New" and "Historical" states are more similar for Simba, the error of the machine learning method would be higher (12.60%). Regarding mathematical modelling, we can see that the relative error between the values predicted by our model with the optimized parameters and the values observed, is typically below the thresholds. The accuracy of this approach was above 95 %.

According to question Q_2, K-Means clustering accurately identified 84% of dogs among four cardiovascular data of two dogs of different ages. Machine learning method can identify one dog among 4 dogs with more than 93% accuracy. To test the accuracy of the identifiability of the dogs with the mathematical modelling approach, we had the same logic as the one with ageing but we have tried the calibrated parameters of one dog on another dog and see if they could fit. It was never the case so the accuracy in this case is around 100 %.

Regarding the cost and robustness, K-Means and MLP use the extracted cardiovascular features as input and they would require computational cost in the feature extraction process. Considering that the number of samples and the size of the input is not large, K-Means only takes very small computational cost. We could also train a MLP with high accuracy (above 93%) and very little training cost (less than 20 minutes). In particular, MLP is robust to beats to beats variability and provides a promising result for the above biological questions. Regarding the computational cost for mathematical

modelling, it requires a light pre-processing of the data. Before running the optimization, the only thing we need to do is to extract the activation times of the ECG and this is almost instantaneous. However, the optimization process to obtain the parameters of the model takes on average 7 hours, although it depends a lot on the initial guess that we consider. Once we have the parameters, model integration in time is cheap from a computational point of view (few seconds). Regarding the robustness of the parameter estimation procedure in mathematical modelling, we have tried so far to run the model with the same parameters for other intervals to see if we are able to reproduce other windows of time with the parameters fitted from one chunk. Usually, we are able to fit other time intervals if we do not go very far from the original one. It is also important to take into account that if the activity of the dog changes (if it is playing, eating, making digestion...) the parameters could change. In the future, we would like to get the model parameters for a larger number of chunks and analyze their variability of them in the "New" and "Historical" states in order to give a reliable interval for the parameters in each state and to have a better notion of the accuracy of this approach. Same applies for the identifiability of each dog in order to know what is the interval in which the parameters move in each dog. In fact, the parameters may significantly change between different dogs because it is possible that there are several combinations of parameters that fit the data. As stated before, it would be necessary to do a study with a larger set of chunks in order to analyze in detail the variability of the parameters. This will be the object of a further investigation. However, we could already spot differences between the animals based on the parameters (the parameters of a dog do not reproduce the data of another one). Regarding the ageing process, it is also possible to assess if the dog is getting old (excluding Simba, we have shown that with the parameters of the "New" state we were not able to fit the data in the "Historical" state).

4.10.1 Limitations and Perspectives

The limitation of the statistical algorithms is that when the number of animals is high or strains and genders of dogs are similar, the accuracy may decrease. Statistical methods are modelled using extracted features which require computational resources and the number and choices of features may have an impact on the analysis of the results (missing some information from the raw signals).

Similarly, MLP needs extracted features as input which other types of neural networks like Convolutional Neural Network (CNN) in [205] do not. CNN can process and learn the important parameters from the signals directly to perform a binary classification. By using CNN, we can avoid the feature extraction process step. However, as the dimension of the signal data is usually high, the CNN method would need more training costs. We can also consider the autoencoder method calibrated by some cardiovascular features to do anomaly detection like in [123] to perform a semi-supervised learning. In general, the neural network method has a "black box" nature. This method can't help us to understand the mechanism of dogs' cardiovascular system. So we won't be able to use it to answer some questions like how much and which element of cardiovascular function has changed. However, in a mathematical modelling approach, the parameters have a link with age which can give us some information related to how age changes cardiovascular functionality (that is to say if the dog gets older or younger). Moreover, if the number of dogs increases in the future, which means the number of classes also increases, the accuracy

of correctly identifying the dogs decreases for machine learning and statistical methods. To test the neural network method and statistical method for a larger number of dogs, we need to find a training strategy that can include enough dogs and acceptable model accuracy. For the mathematical modelling method, it is necessary to do the optimization of the parameters for each dog in each state ("Historical" and "New") and this process takes a lot of time. Consequently, if the number of dogs is high, it will be computationally very demanding in terms of time to answer the fore-mentioned questions.

In the future perspective, we would like to take into account the data of more dogs to test our methods. It is also very important from the mathematical modelling point of view to do the study with more chunks to be able to perform classification tasks on the parameters. We would also like to work on more experiments with different species and strains to gather more information about the effect of ageing on cardiovascular performance. Discovering the impact factors on cardiovascular functionality helps us to understand the reason why some animals have more changes in their cardiovascular data by comparing with others.

Acknowledgements

Elham, Sara and Haibo are ESR-fellows of the INSPIRE European Training Network. INSPIRE receives funding from the EU Horizon 2020 Research and Innovation programme, under the Marie Skłodowska-Curie GA 858070 [1].

We appreciate all support from Pieter-Jan Guns as the scientific coordinator of INSPIRE European Training Network, Michael Markert, Georg Rast, Karin Graf, Jessica Schiwon, Thomas Trautmann and Florian Krause from Boehringer Ingelheim Pharma GmbH & Co KG, Damiano Lombardi from Inria, Muriel Boulakia from Université Paris-Saclay, UVSQ, CNRS, Laboratoire de Mathématiques de Versailles and Sylvain Bernasconi from NOTOCORD and Instem companies.

Bibliography

[1] Pieter-Jan Guns. "inspire": A european training network in safety pharmacology creating opportunities for 15 phd students. *Journal of Pharmacological and Toxicological Methods*, 105:106838, 2020.

[2] PhRMA. Pharmaceutical research and manufacturers of america. 2016 biopharmaceutical research industry profile, 2016.

[3] Peter Ferdinandy, István Baczkó, Peter Bencsik, Zoltan Giricz, Anikó Görbe, Pal Pacher, Zoltan V Varga, Andras Varro, and Rainer Schulz. Definition of hidden drug cardiotoxicity: paradigm change in cardiac safety testing and its clinical implications. *European heart journal*, 40(22):1771–1777, 2019.

[4] Clay W Scott, Matthew F Peters, and Yvonne P Dragan. Human induced pluripotent stem cells and their use in drug discovery for toxicity testing. *Toxicology letters*, 219(1):49–58, 2013.

[5] Chunbo Yang, Jumana Al-Aama, Miodrag Stojkovic, Bernard Keavney, Andrew Trafford, Majlinda Lako, and Lyle Armstrong. Concise review: cardiac disease modeling using induced pluripotent stem cells. *Stem Cells*, 33(9):2643–2651, 2015.

[6] Angela Di Baldassarre, Elisa Cimetta, Sveva Bollini, Giulia Gaggi, and Barbara Ghinassi. Human-induced pluripotent stem cell technology and cardiomyocyte generation: progress and clinical applications. *Cells*, 7(6):48, 2018.

[7] Bert Sakmann and Erwin Neher. Patch clamp techniques for studying ionic channels in excitable membranes. *Annual review of physiology*, 46(1):455–472, 1984.

[8] Fitzwilliam Seibertz, Markus Rapedius, Funsho E Fakuade, Philipp Tomsits, Aiste Liutkute, Lukas Cyganek, Nadine Becker, Rupamanjari Majumder, Sebastian Clauß, Niels Fertig, et al. A modern automated patch-clamp approach for high throughput electrophysiology recordings in native cardiomyocytes. *Communications Biology*, 5(1):969, 2022.

[9] Micha E Spira and Aviad Hai. Multi-electrode array technologies for neuroscience and cardiology. *Nano-Enabled Medical Applications*, pages 567–602, 2020.

[10] Dai Kusumoto and Shinsuke Yuasa. The application of convolutional neural network to stem cell biology. *Inflammation and regeneration*, 39(1):1–7, 2019.

[11] Augustus O Grant. Cardiac ion channels. *Circulation: Arrhythmia and Electrophysiology*, 2(2):185–194, 2009.

[12] Shawn M Lamothe, Jun Guo, Wentao Li, Tonghua Yang, and Shetuan Zhang. The human ether-a-go-go-related gene (herg) potassium channel represents an unusual target for protease-mediated damage. *Journal of Biological Chemistry*, 291(39):20387–20401, 2016.

[13] Alan L Hodgkin and Andrew F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500, 1952.

[14] George W Beeler and Harald Reuter. Reconstruction of the action potential of ventricular myocardial fibres. *The Journal of physiology*, 268(1):177–210, 1977.

[15] Ching-hsing Luo and Yoram Rudy. A model of the ventricular cardiac action potential. depolarization, repolarization, and their interaction. *Circulation research*, 68(6):1501–1526, 1991.

[16] S Inada, JC Hancox, H Zhang, and MR Boyett. One-dimensional mathematical model of the atrioventricular node including atrio-nodal, nodal, and nodal-his cells. *Biophysical journal*, 97(8):2117–2127, 2009.

[17] Victor A Maltsev and Edward G Lakatta. Synergism of coupled subsarcolemmal ca2+ clocks and sarcolemmal voltage clocks confers robust and flexible pacemaker function in a novel pacemaker cell model. *American Journal of Physiology-Heart and Circulatory Physiology*, 296(3):H594–H615, 2009.

[18] KJ Sampson, V Iyer, AR Marks, and RS Kass. A computational model of purkinje fibre single cell electrophysiology: implications for the long qt syndrome. *The Journal of physiology*, 588(14):2643–2655, 2010.

[19] Thomas O'Hara, László Virág, András Varró, and Yoram Rudy. Simulation of the undiseased human cardiac ventricular action potential: model formulation and experimental validation. *PLoS computational biology*, 7(5):e1002061, 2011.

[20] Frank B Sachse. *Computational cardiology: modeling of anatomy, electrophysiology, and mechanics*, volume 2966. Springer Science & Business Media, 2004.

[21] Michelangelo Paci, Jari Hyttinen, Katriina Aalto-Setälä, and Stefano Severi. Computational models of ventricular-and atrial-like human induced pluripotent stem cell derived cardiomyocytes. *Annals of biomedical engineering*, 41:2334–2348, 2013.

[22] Akwasi D Akwaboah, Bright Tsevi, Pascal Yamlome, Jacqueline A Treat, Maila Brucal-Hallare, Jonathan M Cordeiro, and Makarand Deo. An in silico hipsc-derived cardiomyocyte model built with genetic algorithm. *Frontiers in Physiology*, 12:675867, 2021.

[23] Emanuela Abbate, Muriel Boulakia, Yves Coudière, Jean-Frédéric Gerbeau, Philippe Zitoun, and Nejib Zemzemi. In silico assessment of the effects of various compounds in mea/hipsc-cm assays: modeling and numerical simulations. *Journal of pharmacological and toxicological methods*, 89:59–72, 2018.

[24] Fabien Raphel, Muriel Boulakia, Nejib Zemzemi, Yves Coudière, Jean-Michel Guillon, Philippe Zitoun, and Jean-Frédéric Gerbeau. Identification of ion currents components generating field potential recorded in mea from hipsc-cm. *IEEE Transactions on Biomedical Engineering*, 65(6):1311–1319, 2017.

[25] Elisa Passini, Oliver J Britton, Hua Rong Lu, Jutta Rohrbacher, An N Hermans, David J Gallacher, Robert JH Greig, Alfonso Bueno-Orovio, and Blanca Rodriguez. Human in silico drug trials demonstrate higher accuracy than animal models in predicting clinical pro-arrhythmic cardiotoxicity. *Frontiers in physiology*, 8:668, 2017.

[26] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[27] Shunichi Amari. A theory of adaptive pattern classifiers. *IEEE Transactions on Electronic Computers*, EC-16(3):299–307, 1967.

[28] Seppo Linnainmaa. *The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors*. PhD thesis, Master's Thesis (in Finnish), Univ. Helsinki, 1970.

[29] Mohsen Sharifi, Dan Buzatu, Stephen Harris, and Jon Wilkes. Development of models for predicting torsade de pointes cardiac arrhythmias using perceptron neural networks. *BMC bioinformatics*, 18(14):93–102, 2017.

[30] Sebastian Polak, Barbara Wiśniowska, Malidi Ahamadi, and Aleksander Mendyk. Prediction of the herg potassium channel inhibition potential with use of artificial neural networks. *Applied Soft Computing*, 11(2):2611–2617, 2011.

[31] Pietro Delre, Giovanna J Lavado, Giuseppe Lamanna, Michele Saviano, Alessandra Roncaglioni, Emilio Benfenati, Giuseppe Felice Mangiatordi, and Domenico Gadaleta. Ligand-based prediction of herg-mediated cardiotoxicity based on the integration of different machine learning techniques. *Frontiers in Pharmacology*, 13:951083, 2022.

[32] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[33] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[34] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. Ieee, 2017.

[35] Miyuki Sakai, Kazuki Nagayasu, Norihiro Shibui, Chihiro Andoh, Kaito Takayama, Hisashi Shirakawa, and Shuji Kaneko. Prediction of pharmacological activities from chemical structures with graph convolutional neural networks. *Scientific reports*, 11(1):525, 2021.

[36] Xia Sun, Long Ma, Xiaodong Du, Jun Feng, and Ke Dong. Deep convolution neural networks for drug-drug interaction extraction. In *2018 IEEE International conference on bioinformatics and biomedicine (BIBM)*, pages 1662–1668. IEEE, 2018.

[37] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[38] Hervé Bourlard and Yves Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59(4-5):291–294, 1988.

[39] Maxime Barrault, Yvon Maday, Ngoc Cuong Nguyen, and Anthony T Patera. An 'empirical interpolation'method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667–672, 2004.

[40] Alfio Quarteroni, Andrea Manzoni, and Federico Negri. *Reduced basis methods for partial differential equations: an introduction*, volume 92. Springer, 2015.

[41] Peter Benner, Mario Ohlberger, Albert Cohen, and Karen Willcox. *Model reduction and approximation: theory and algorithms*. SIAM, 2017.

[42] Joshua L Barnett, Charbel Farhat, and Yvon Maday. Neural-network-augmented projection-based model order reduction for mitigating the kolmogorov barrier to reducibility of cfd models. *arXiv preprint arXiv:2212.08939*, 2022.

[43] Romit Maulik, Bethany Lusch, and Prasanna Balaprakash. Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders. *Physics of Fluids*, 33(3), 2021.

[44] Kookjin Lee and Kevin T Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *Journal of Computational Physics*, 404:108973, 2020.

[45] Michele Milano and Petros Koumoutsakos. Neural network modeling for near wall turbulent flow. *Journal of Computational Physics*, 182(1):1–26, 2002.

[46] Paolo Conti, Giorgio Gobat, Stefania Fresca, Andrea Manzoni, and Attilio Frangi. Reduced order modeling of parametrized systems through autoencoders and sindy approach: continuation of periodic solutions. *Computer Methods in Applied Mechanics and Engineering*, 411:116072, 2023.

[47] Jinling Zhao, Lei Hu, Yingying Dong, Linsheng Huang, Shizhuang Weng, and Dongyan Zhang. A combination method of stacked autoencoder and 3d deep residual network for hyperspectral image classification. *International Journal of Applied Earth Observation and Geoinformation*, 102:102459, 2021.

[48] Pengzhi Li, Yan Pei, and Jianqiang Li. A comprehensive survey on design and application of autoencoder in deep learning. *Applied Soft Computing*, page 110176, 2023.

[49] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders. *Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook*, pages 353–374, 2023.

[50] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[51] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[52] Lovedeep Gondara. Medical image denoising using convolutional denoising autoencoders. In *2016 IEEE 16th international conference on data mining workshops (ICDMW)*, pages 241–246. IEEE, 2016.

[53] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[54] Peng Xiong, Hongrui Wang, Ming Liu, and Xiuling Liu. Denoising autoencoder for eletrocardiogram signal enhancement. *Journal of Medical Imaging and Health Informatics*, 5(8):1804–1810, 2015.

[55] Stuart C Kramer and Harold W Sorenson. Bayesian parameter estimation. *IEEE Transactions on Automatic Control*, 33(2):217–222, 1988.

[56] James Vere Beck and Kenneth J Arnold. *Parameter estimation in engineering and science*. James Beck, 1977.

[57] Richard C Aster, Brian Borchers, and Clifford H Thurber. *Parameter estimation and inverse problems*. Elsevier, 2018.

[58] Ake Bjorck. *Numerical Methods for Least Squares Problems*, volume 51. SIAM, 1996.

[59] Philip E Gill and Walter Murray. Algorithms for the solution of the nonlinear least-squares problem. *SIAM Journal on Numerical Analysis*, 15(5):977–992, 1978.

[60] John E Dennis Jr and Robert B Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations*. SIAM, 1996.

[61] Roger Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2000.

[62] Hua Liang and Hulin Wu. Parameter estimation for differential equation models using a framework of measurement error in regression models. *Journal of the American Statistical Association*, 103(484):1570–1583, 2008.

[63] I-Chun Chou and Eberhard O Voit. Recent developments in parameter estimation and structure identification of biochemical and genomic systems. *Mathematical biosciences*, 219(2):57–83, 2009.

[64] Carmen G Moles, Pedro Mendes, and Julio R Banga. Parameter estimation in biochemical pathways: a comparison of global optimization methods. *Genome research*, 13(11):2467–2474, 2003.

[65] D Magill. Optimal adaptive estimation of sampled stochastic processes. *IEEE Transactions on Automatic Control*, 10(4):434–439, 1965.

[66] LA Rastrigin and Y Rubinstein. The comparison of random search and stochastic approximation while solving the problem of optimization. *Automatic Control*, 2:23–29, 1969.

[67] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Oxford University Press*, 1970.

[68] Stephen Brooks. Markov chain monte carlo method and its application. *Journal of the royal statistical society: series D (the Statistician)*, 47(1):69–100, 1998.

[69] AHG Rinnooy Kan and Gerrit T Timmer. Stochastic global optimization methods part i: Clustering methods. *Mathematical programming*, 39:27–56, 1987.

[70] Carlos Andrés Pena-Reyes and Moshe Sipper. Evolutionary computation in medicine: an overview. *Artificial Intelligence in Medicine*, 19(1):1–23, 2000.

[71] Nikolaus Hansen and Andreas Ostermeier. Convergence properties of evolution strategies with the derandomized covariance matrix adaptation: The (/i,)-es. *Eufit*, 97:650–654, 1997.

[72] Marco Dorigo, Mauro Birattari, and Thomas Stutzle. Ant colony optimization. *IEEE computational intelligence magazine*, 1(4):28–39, 2006.

[73] Scott Kirkpatrick, C Daniel Gelatt Jr, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[74] Djurdje Cvijović and Jacek Klinowski. Taboo search: an approach to the multiple minima problem. *Science*, 267(5198):664–666, 1995.

[75] Marcio Schwaab, Evaristo Chalbaud Biscaia Jr, José Luiz Monteiro, and José Carlos Pinto. Nonlinear parameter estimation through particle swarm optimization. *Chemical Engineering Science*, 63(6):1542–1552, 2008.

[76] Leonardo Gabrielli, Stefano Tomassetti, Stefano Squartini, and Carlo Zinato. Introducing deep machine learning for parameter estimation in physical modelling. In *Proceedings of the 20th international conference on digital audio effects*, 2017.

[77] Viktor Grimm, Alexander Heinlein, Axel Klawonn, Martin Lanser, and Janine Weber. Estimating the time-dependent contact rate of sir and seir models in mathematical epidemiology using physics-informed neural networks. *Electronic Transactions on Numerical Analysis*, 56:1–27, 2022.

[78] Alireza Yazdani, Lu Lu, Maziar Raissi, and George Em Karniadakis. Systems biology informed deep learning for inferring parameters and hidden dynamics. *PLOS Computational Biology*, 16(11):1–19, 11 2020.

[79] Xianghao Kong, Koji Yamashita, Brandon Foggo, and Nanpeng Yu. Dynamic parameter estimation with physics-based neural ordinary differential equations. In *2022 IEEE Power & Energy Society General Meeting (PESGM)*, pages 1–5. IEEE, 2022.

[80] Tamirat Temesgen Dufera, Yadeta Chimdessa Seboka, Carlos Fresneda Portillo, et al. Parameter estimation for dynamical systems using a deep neural network. *Applied Computational Intelligence and Soft Computing*, 2022, 2022.

[81] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021.

[82] Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed deeponets. *Science advances*, 7(40):eabi8605, 2021.

[83] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pages 5301–5310. PMLR, 2019.

[84] Omar Sallam and Mirjam Fürth. On the use of fourier features-physics informed neural networks (ff-pinn) for forward and inverse fluid mechanics problems. *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, 237:846–866, 2023.

[85] SJ Barrett and WB Langdon. Advances in the application of machine learning techniques in drug discovery, design and development. In *Applications of Soft Computing: Recent Trends*, pages 99–110. Springer, 2006.

[86] Bernhard Scholkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2018.

[87] Ruifeng Liu, Michael Madore, Kyle P Glover, Michael G Feasel, and Anders Wallqvist. Assessing deep and shallow learning methods for quantitative prediction of acute chemical toxicity. *Toxicological Sciences*, 164(2):512–526, 2018.

[88] Mariusz Butkiewicz, Ralf Mueller, Danilo Selic, Eric Dawson, and Jens Meiler. Application of machine learning approaches on quantitative structure activity relationships. In *2009 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 255–262. IEEE, 2009.

[89] Sean Ekins. Progress in computational toxicology. *Journal of pharmacological and toxicological methods*, 69(2):115–140, 2014.

[90] Hongmao Sun. A naive bayes classifier for prediction of multidrug resistance reversal activity on the basis of atom typing. *Journal of Medicinal Chemistry*, 48(12):4031–4039, 2005.

[91] Qiangqiang Ouyang, Wenjian Yang, Yue Wu, Zhongyuan Xu, Yongjun Hu, Ning Hu, and Diming Zhang. Multi-labeled neural network model for automatically processing cardiomyocyte mechanical beating signals in drug assessment. *Biosensors and Bioelectronics*, 209:114261, 2022.

[92] Talasila Bhanu Teja, Mahendran Sekar, Talasila Pallavi, Sivamma Mettu, TE Murthy, Nur Najihah Izzati Mat Rani, Pallaval Veera Bramhachari, and Srinivasa Reddy Bonam. Role of artificial neural networks in pharmaceutical sciences. *Journal of Young Pharmacists*, 14(1), 2022.

[93] Shan Wang, Jinwei Di, Dan Wang, Xudong Dai, Yabing Hua, Xiang Gao, Aiping Zheng, and Jing Gao. State-of-the-art review of artificial neural networks to predict, characterize and optimize pharmaceutical formulation. *Pharmaceutics*, 14(1):183, 2022.

[94] Youjun Xu, Ziwei Dai, Fangjin Chen, Shuaishi Gao, Jianfeng Pei, and Luhua Lai. Deep learning for drug-induced liver injury. *Journal of chemical information and modeling*, 55(10):2085–2093, 2015.

[95] Daniel Jimenez-Carretero, Vahid Abrishami, Laura Fernandez-de Manuel, Irene Palacios, Antonio Quilez-Alvarez, Alberto Diez-Sanchez, Miguel A Del Pozo, and María C Montoya. Tox_ (r) cnn: Deep learning-based nuclei profiling tool for drug toxicity screening. *PLoS computational biology*, 14(11):e1006238, 2018.

[96] Nils Strodthoff, Patrick Wagner, Tobias Schaeffter, and Wojciech Samek. Deep learning for ecg analysis: Benchmarks and insights from ptb-xl. *IEEE Journal of Biomedical and Health Informatics*, 25(5):1519–1528, 2020.

[97] CK Roopa and BS Harish. A survey on various machine learning approaches for ecg analysis. *International Journal of Computer Applications*, 163(9):25–33, 2017.

[98] Zahra Ebrahimi, Mohammad Loni, Masoud Daneshtalab, and Arash Gharehbaghi. A review on deep learning methods for ecg arrhythmia classification. *Expert Systems with Applications: X*, 7:100033, 2020.

[99] Amin Ullah, Syed Muhammad Anwar, Muhammad Bilal, and Raja Majid Mehmood. Classification of arrhythmia by using deep learning with 2-d ecg spectral image representation. *Remote Sensing*, 12(10):1685, 2020.

[100] Alexander Craik, Yongtian He, and Jose L Contreras-Vidal. Deep learning for electroencephalogram (eeg) classification tasks: a review. *Journal of neural engineering*, 16(3):031001, 2019.

[101] U Rajendra Acharya, Shu Lih Oh, Yuki Hagiwara, Jen Hong Tan, and Hojjat Adeli. Deep convolutional neural network for the automated detection and diagnosis of seizure using eeg signals. *Computers in biology and medicine*, 100:270–278, 2018.

[102] Jeffrey Zhang, Sravani Gajjala, Pulkit Agrawal, Geoffrey H Tison, Laura A Hallock, Lauren Beussink-Nelson, Mats H Lassen, Eugene Fan, Mandar A Aras, ChaRandle Jordan, et al. Fully automated echocardiogram interpretation in clinical practice: feasibility and diagnostic accuracy. *Circulation*, 138(16):1623–1635, 2018.

[103] Ali Madani, Ramy Arnaout, Mohammad Mofrad, and Rima Arnaout. Fast and accurate view classification of echocardiograms using deep learning. *NPJ digital medicine*, 1(1):6, 2018.

[104] Mohammad R Keyvanpour and Mehrnoush Barani Shirzad. An analysis of qsar research based on machine learning concepts. *Current Drug Discovery Technologies*, 18(1):17–30, 2021.

[105] Divya C Kernik, Stefano Morotti, HaoDi Wu, Priyanka Garg, Henry J Duff, Junko Kurokawa, José Jalife, Joseph C Wu, Eleonora Grandi, and Colleen E Clancy. A computational model of induced pluripotent stem-cell derived cardiomyocytes incorporating experimental variability from multiple data sources. *The Journal of physiology*, 597(17):4533–4564, 2019.

[106] Jitendra R Raol, Gopalrathnam Girija, and Jatinder Singh. *Modelling and parameter estimation of dynamic systems*, volume 65. Iet, 2004.

[107] John E Dennis, Jr and Jorge J Moré. Quasi-newton methods, motivation and theory. *SIAM review*, 19(1):46–89, 1977.

[108] Philip E Gill and Walter Murray. Quasi-newton methods for unconstrained optimization. *IMA Journal of Applied Mathematics*, 9(1):91–108, 1972.

[109] Yonathan Bard. Comparison of gradient methods for the solution of nonlinear parameter estimation problems. *SIAM Journal on Numerical Analysis*, 7(1):157–186, 1970.

[110] Dorit Wolf and Ralf Moros. Estimating rate constants of heterogeneous catalytic reactions without supposition of rate determining surface steps—an application of a genetic algorithm. *Chemical Engineering Science*, 52(7):1189–1199, 1997.

[111] Niels Baden and John Villadsen. A family of collocation based methods for parameter estimation in differential equations. *The Chemical Engineering Journal*, 23(1):1–13, 1982.

[112] Bruno Van Den Bosch and Leon Hellinckx. A new method for the estimation of parameters in differential equations. *AIChE Journal*, 20(2):250–255, 1974.

[113] Bernd Hartke. Global optimization. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 1(6):879–887, 2011.

[114] Antanas Žilinskas and Anatoly Zhigljavsky. Stochastic global optimization: a review on the occasion of 25 years of informatica. *Informatica*, 27(2):229–256, 2016.

[115] Balthazar Van der Pol. A theory of the amplitude of free and forced triode vibrations, radio review. 1 (1920) 701-710, 754-762. *Selected Scientific Papers*, 1, 1960.

[116] Nik Cunniffe, Frédéric Hamelin, Abderrahman Iggidr, Alain Rapaport, and Gauthier Sallet. Observability, Identifiability and Epidemiology -A primer-. working paper or preprint, May 2023.

[117] Hongyu Miao, Xiaohua Xia, Alan S Perelson, and Hulin Wu. On identifiability of nonlinear ode models and applications in viral dynamics. *SIAM review*, 53(1):3–39, 2011.

[118] D. Csercsik, Gábor Szederkényi, and Katalin M. Hangos. Identifiability of a hodgkin-huxley type ion channel under voltage step measurement conditions. *IFAC Proceedings Volumes*, 43(5):332–337, 2010. 9th IFAC Symposium on Dynamics and Control of Process Systems.

[119] Nathalie Verdière and Carine Jauberthie. Parameter estimation procedure based on input-output integro-differential polynomials. application to the hindmarsh-rose model. In *2020 European Control Conference (ECC)*, pages 220–225, 2020.

[120] Ror Bellman and Karl Johan Åström. On structural identifiability. *Mathematical biosciences*, 7(3-4):329–339, 1970.

[121] Alejandro F Villaverde and Gemma Massonis. On testing structural identifiability by a simple scaling method: Relying on scaling symmetries can be misleading. *PLoS computational biology*, 17(10):e1009032, 2021.

[122] Eliakim H Moore. *On the reciprocal of the general algebraic matrix*, volume 26. Bulletin of the American Mathematical Society, 1920.

[123] Hoang Anh Dau, Vic Ciesielski, and Andy Song. Anomaly detection using replicator neural networks trained on examples of one class. In *Simulated Evolution and Learning: 10th International Conference, SEAL 2014, Dunedin, New Zealand, December 15-18, 2014. Proceedings 10*, pages 311–322. Springer, 2014.

[124] João Pereira and Margarida Silveira. Unsupervised representation learning and anomaly detection in ecg sequences. *International Journal of Data Mining and Bioinformatics*, 22(4):389–407, 2019.

[125] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[126] Aston Zhang, Zachary C Lipton, Mu Li, and Alexander J Smola. Dive into deep learning. *arXiv preprint arXiv:2106.11342*, 2021.

[127] Francesco Camastra and Antonino Staiano. Intrinsic dimension estimation: Advances and open problems. *Information Sciences*, 328:26–41, 2016.

[128] Keinosuke Fukunaga. *Introduction to statistical pattern recognition*. Elsevier, 2013.

[129] Albert Cohen, Ronald DeVore, Guergana Petrova, and Przemyslaw Wojtaszczyk. Optimal stable nonlinear approximation. *Foundations of Computational Mathematics*, 22(3):607–648, 2022.

[130] Guergana Petrova and Przemysław Wojtaszczyk. Lipschitz widths. *Constructive Approximation*, 57(2):759–805, 2023.

[131] Tommi Kärkkäinen and Jan Hänninen. Additive autoencoder for dimension estimation. *Neurocomputing*, 551:126520, 2023.

[132] Lawrence Cayton. *Algorithms for manifold learning.* eScholarship, University of California, 2008.

[133] Nicola Ferri, Peter Siegl, Alberto Corsini, Joerg Herrmann, Amir Lerman, and Renee Benghozi. Drug attrition during pre-clinical and clinical development: Understanding and managing drug-induced cardiotoxicity. *Pharmacology and Therapeutics*, 138(3):470–484, 2013.

[134] Bernard Fermini and Anthony A Fossa. The impact of drug-induced qt interval prolongation on drug discovery and development. *Nature reviews Drug discovery*, 2(6):439–447, 2003.

[135] Antje D Ebert, Ping Liang, and Joseph C Wu. Induced pluripotent stem cells as a disease modeling and drug screening platform. *Journal of cardiovascular pharmacology*, 60(4):408, 2012.

[136] Amanda Garrido, Alban Lepailleur, Serge M Mignani, Patrick Dallemagne, and Christophe Rochais. herg toxicity assessment: Useful guidelines for drug design. *European journal of medicinal chemistry*, 195:112290, 2020.

[137] Fabien Raphel, Tessa De Korte, Damiano Lombardi, Stefan Braam, and Jean-Frederic Gerbeau. A greedy classifier optimization strategy to assess ion channel blocking activity and pro-arrhythmia in hipsc-cardiomyocytes. *PLoS computational biology*, 16(9):e1008203, 2020.

[138] Sophie Kussauer, Robert David, and Heiko Lemcke. hipscs derived cardiac cells for drug and toxicity screening and disease modeling: what micro-electrode-array analyses can tell us. *Cells*, 8(11):1331, 2019.

[139] Guang Deng and LW Cahill. An adaptive gaussian filter for noise reduction and edge detection. In *1993 IEEE conference record nuclear science symposium and medical imaging conference*, pages 1615–1619. IEEE, 1993.

[140] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[141] Yingying Wang, Yibin Li, Yong Song, and Xuewen Rong. The influence of the activation function in a convolution neural network model of facial expression recognition. *Applied Sciences*, 10(5):1897, 2020.

[142] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[143] Jun Han and Claudio Moraga. The influence of the sigmoid function parameters on the speed of backpropagation learning. In *International workshop on artificial neural networks*, pages 195–201. Springer, 1995.

[144] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization. *Advances in neural information processing systems*, 31, 2018.

[145] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[146] Marius-Constantin Popescu, Valentina E Balas, Liliana Perescu-Popescu, and Nikos Mastorakis. Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, 8(7):579–588, 2009.

[147] Gaurav Kumar, Urja Pawar, and Ruairi O'Reilly. Arrhythmia detection in ecg signals using a multilayer perceptron network. In *AICS*, pages 353–364, 2019.

[148] Weili Guo, Haikun Wei, Junsheng Zhao, and Kanjian Zhang. Theoretical and numerical analysis of learning dynamics near singularity in multilayer perceptrons. *Neurocomputing*, 151:390–400, 2015.

[149] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[150] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[151] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In *Shape, contour and grouping in computer vision*, pages 319–345. Springer, 1999.

[152] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[153] Yunan Wu, Feng Yang, Ying Liu, Xuefan Zha, and Shaofeng Yuan. A comparison of 1-d and 2-d deep convolutional neural networks in ecg classification. *arXiv preprint arXiv:1810.07088*, 2018.

[154] Ekaba Bisong. *Building machine learning and deep learning models on Google cloud platform*. Springer, 2019.

[155] Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

[156] Elad Plaut. From principal subspaces to principal components with linear autoencoders. *arXiv preprint arXiv:1804.10253*, 2018.

[157] Begoña Benito, Ramon Brugada, Josep Brugada, and Pedro Brugada. Brugada syndrome. *Progress in cardiovascular diseases*, 51(1):1–22, 2008.

[158] Charles Antzelevitch. Brugada syndrome. *Pacing and clinical electrophysiology*, 29(10):1130–1159, 2006.

[159] Ramon Brugada, Oscar Campuzano, Georgia Sarquella-Brugada, Josep Brugada, and Pedro Brugada. Brugada syndrome. *Methodist DeBakey cardiovascular journal*, 10(1):25, 2014.

[160] Wenjia Li, Lei Yin, Cheng Shen, Kai Hu, Junbo Ge, and Aijun Sun. Scn5a variants: association with cardiac disorders. *Frontiers in physiology*, 9:1372, 2018.

[161] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[162] Nikolaus Hansen. The cma evolution strategy: A tutorial, 2016.

[163] Anna Muszkiewicz, Oliver J Britton, Philip Gemmell, Elisa Passini, Carlos Sánchez, Xin Zhou, Annamaria Carusi, T Alexander Quinn, Kevin Burrage, Alfonso Bueno-Orovio, et al. Variability in cardiac electrophysiology: using experimentally-calibrated populations of models to move beyond the single virtual physiological human paradigm. *Progress in biophysics and molecular biology*, 120(1-3):115–127, 2016.

[164] Bernard Delyon, Marc Lavielle, and Eric Moulines. Convergence of a stochastic approximation version of the em algorithm. *Annals of statistics*, pages 94–128, 1999.

[165] Jingbo Wang and Nicholas Zabaras. Hierarchical bayesian models for inverse problems in heat conduction. *Inverse Problems*, 21(1):183, 2004.

[166] William S Redfern, Ian D Wakefield, Helen Prior, Christopher E Pollard, Timothy G Hammond, and Jean-Pierre Valentin. Safety pharmacology–a progressive approach. *Fundamental & Clinical Pharmacology*, 16(3):161–173, 2002.

[167] Katherine Brake, Ashwini Gumireddy, Amit Tiwari, Harsh Chauhan, and Dunesh Kumari. In vivo studies for drug development via oral delivery: Challenges, animal models and techniques. *vitro research*, 8(8), 2017.

[168] Joseph A 'Chris' Delaney and Samy Suissa. The case-crossover study design in pharmacoepidemiology. *Statistical methods in medical research*, 18(1):53–65, 2009.

[169] Kathryn M Meurs, Matthew W Miller, and Margaret R Slater. Comparison of the indirect oscillometric and direct arterial methods for blood pressure measurements in anesthetized dogs. *Journal of the American Animal Hospital Association*, 32(6):471–475, 1996.

[170] Paul-Emile Roy and Rona George. Recent advances in studies of cardiac structure and metabolism. *American Heart Journal*, 92(3):411, 1976.

[171] Gordon H Templeton, Melvin R Platt, James T Willerson, and Myron L Weisfeldt. Influence of aging on left ventricular hemodynamics and stiffness in beagles. *Circulation research*, 44(2):189–194, 1979.

[172] H Gholam Hosseini, Dehan Luo, and Karen Jane Reynolds. The comparison of different feed forward neural network architectures for ecg signal diagnosis. *Medical engineering & physics*, 28(4):372–378, 2006.

[173] E Roland Adams and Anthony Choi. Using neural networks to predict cardiac arrhythmias. In *2012 IEEE international conference on systems, man, and cybernetics (smc)*, pages 402–407. IEEE, 2012.

[174] Rosaria Silipo and Carlo Marchesi. Artificial neural networks for automatic ecg analysis. *IEEE transactions on signal processing*, 46(5):1417–1425, 1998.

[175] Anna O Basile, Alexandre Yahi, and Nicholas P Tatonetti. Artificial intelligence for drug toxicity and safety. *Trends in pharmacological sciences*, 40(9):624–635, 2019.

[176] Khader Shameer, Kipp W Johnson, Benjamin S Glicksberg, Joel T Dudley, and Partho P Sengupta. Machine learning in cardiovascular medicine: are we there yet? *Heart*, 104(14):1156–1164, 2018.

[177] Fuyou Liang and Hao Liu. A closed-loop lumped parameter computational model for human cardiovascular system. *JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing*, 48(4):484–493, 2005.

[178] Mehran Mirramezani and Shawn C Shadden. A distributed lumped parameter model of blood flow. *Annals of Biomedical Engineering*, 48(12):2870–2886, 2020.

[179] Michael J Moulton, Brian D Hong, and Timothy W Secomb. Simulation of left ventricular dynamics using a low-order mathematical model. *Cardiovascular Engineering and Technology*, 8(4):480–494, 2017.

[180] Patrick Segers, N Stergiopulos, Pascal Verdonck, and Ronny Verhoeven. Assessment of distributed arterial network models. *Medical and Biological Engineering and Computing*, 35(6):729–736, 1997.

[181] Raheem Gul, Christof Schütte, and Stefan Bernhard. Mathematical modeling and sensitivity analysis of arterial anastomosis in arm arteries. *Applied Mathematics and Modelling*, 40:7724–7738, 2016.

[182] Raheem Gul and Stefan Bernhard. Optimal measurement locations for diagnosis of aortic stenosis and aneurysms in a lumped parameter model of systemic circulation using sensitivity analysis. *International Journal of Biomathematics*, 10:175116, 2017.

[183] Yih-Choung Yu, J Robert Boston, Marwan A Simaan, and James F Antaki. Estimation of systemic vascular bed features for artificial heart control. *IEEE Transactions on Automatic Control*, 43:765–777, 1998.

[184] Michael Markert, Thomas Trautmann, Florian Krause, Marius Cioaga, Sebastien Mouriot, Miriam Wetzel, and Brian D Guth. A new telemetry-based system for assessing cardiovascular function in group-housed large animals. taking the 3rs to a new level with the evaluation of remote measurement via cloud data transmission. *Journal of Pharmacological and Toxicological Methods*, 93:90–97, 2018.

[185] Magne Arve Flaten, Terje Simonsen, and Harald Olsen. Drug-related information generates placebo and nocebo responses that modify the drug response. *Psychosomatic Medicine*, 61(2):250–255, 1999.

[186] Barry Fetics, Erez Nevo, Chen-Huan Chen, and David A Kass. Parametric model derivation of transfer function for noninvasive estimation of aortic pressure by radial tonometry. *IEEE Transactions on Biomedical Engineering*, 46(6):698–706, 1999.

[187] Victoria P Le, Attila Kovacs, and Jessica E Wagenseil. Measuring left ventricular pressure in late embryonic and neonatal mice. *Journal of Visualized Experiments*, 60, 2012.

[188] Devona Branch. Ekg interpretation. SlideServe, 16, 2014.

[189] Alfred Bartolucci, Karan P Singh, and Sejong Bae. *Introduction to statistical analysis of laboratory data*. John Wiley & Sons, 2015.

[190] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering. *ACM Computing Surveys*, 31(3):264–323, 1999.

[191] David MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[192] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.

[193] Ian Davidson and Ashwin Satyanarayana. Speeding up k-means clustering by bootstrap averaging. In *IEEE Data Mining Workshop on Clustering Large Data Sets*, 2003.

[194] Simon Hawkins, Hongxing He, Graham Williams, and Rohan Baxter. Outlier detection using replicator neural networks. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 170–180. Springer, 2002.

[195] Mikolai Karpinski, Volodvmvr Khoma, Valerii Dudvkevych, Yuriv Khoma, and Dmytro Sabodashko. Autoencoder neural networks for outlier correction in ecg-based biometric identification. In *2018 IEEE 4th international symposium on wireless systems within the international conferences on intelligent data acquisition and advanced computing systems (IDAACS-SWS)*, pages 210–215. IEEE, 2018.

[196] Hongmei Yan, Yingtao Jiang, Jun Zheng, Chenglin Peng, and Qinghui Li. A multilayer perceptron-based medical decision support system for heart disease diagnosis. *Expert Systems with Applications*, 30(2):272–281, 2006.

[197] VN Ganapathi Raju, K Prasanna Lakshmi, Vinod Mahesh Jain, Archana Kalidindi, and V Padma. Study the influence of normalization/transformation process on the accuracy of supervised classification. In *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pages 729–735. IEEE, 2020.

[198] André CPLF de Carvalho and Alex A Freitas. A tutorial on multi-label classification techniques. *Foundations of Computational Intelligence Volume 5: Function Approximation and Classification*, pages 177–195, 2009.

[199] Marc Thiriet and Kim H Parker. Physiology and pathology of the cardiovascular system: a physical perspective. *Cardiovascular Mathematics: Modeling and simulation of the circulatory system*, pages 1–45, 2009.

[200] Hassanain Ali and Lefta Mossa. Engineering modeling of human cardiovascular system. *Al-Nahrain Journal for Engineering Sciences*, 11(2):307–314, 2008.

[201] Hiroyuki Suga and Kiichi Sagawa. Instantaneous pressure-volume relationships and their ratio in the excised, supported canine left ventricle. *Circulation research*, 35(1):117–126, 1974.

[202] James A Svoboda and Richard C Dorf. *Introduction to electric circuits*. John Wiley & Sons, 2013.

[203] Integration and odes: scipy.integrate.odeint, October 2017.

[204] Alan C. Hindmarsh and Linda R. Petzold. Lsoda, ordinary differential equation solver for stiff or non-stiff system, September 2005.

[205] Tae Joon Jun, Hoang Minh Nguyen, Daeyoun Kang, Dohyeun Kim, Daeyoung Kim, and Young-Hak Kim. Ecg arrhythmia classification using a 2-d convolutional neural network. *arXiv preprint arXiv:1804.06812*, 2018.