Master's Thesis

Double Master's Degree in Industrial Engineering and Automatic Control and Robotics

Applying Reinforcement Learning in Treatment Strategies for Cardiogenic Shock Patients

DISERTATION

Author: Roger Pallarès López
Director: Nicholas Houstis, MD, PhD
Coadvisor: Aaron Aguirre, MD, PhD
Speaker: Cecilio Angulo Bahon, PhD

Call: April 2021

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

ABSTRACT

Treating patients with cardiogenic shock is definitely a medical challenge. Physicians in the cardiac intensive care unit have to constantly face a stressful environment where they have to make quick decisions on a vast amount of patients. They must reason about the complex physiology of critical illness and take actions based on a mental model of the patient's physiology, together with mental predictions of possible patient's outcomes to interventions. This approach is clearly life-saving and irreplaceable, yet it can lead to errors and compromise patient results. A computational tool equipped with quantitative knowledge of physiology with the ability to systematically evaluate the patient's state, could help physicians with valuable suggestions for action.

In this master's thesis it has been proposed to train a policy with reinforcement learning to make therapy decisions on simulated patients with cardiogenic shock due to decompensated heart failure. The Burkhoff and Tyberg cardiovascular hemodynamics model was implemented to simulate cardiogenic shock patients. The deep Q-network with experience replay algorithm was developed to train policies interacting with the simulator. Improvements on the policy performance were carried out by introducing the domain randomization technique during the training phase. Such a technique demands a set of parameter ranges to define the domain. Therefore, these parameter ranges were obtained from real patient's data by means of a system identification tool.

Results showed that using an encoder to identify parameters from patient's data can be promising, yet improvements are needed. Furthermore, it was demonstrated that domain randomization increases robustness on policies, giving their ability to operate in unseen environments. The variation of penalty terms on the reward showed different behaviors on the trained policies, suggesting the need for such penalties to obtain the desired policy performance. Overall, a policy was trained and tested satisfactorily on a simulated patient with cardiogenic shock.

These findings suggested that a policy can learn from simulated data and propose therapy decisions that in some cases could resemble to the actions made by physicians. The outcomes obtained from this approach are encouraging and further analysis and assessment on policies with real data should be performed.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. Motivation

Heart failure (HF) affects more than 64 million people worldwide. In the US, the incidence of HF was 6.9 million in 2020 and is expected to increase to nearly 8.5 million in 2030. Moreover, the financial burden of HF on healthcare systems is high and will increase in the future. In the US, the total cost of care for HF in 2020 was estimated at $43.6 billion. The annual total cost of care is projected to $69.7 billion by 2030 if no improvements are achieved [31].

Caring for a patient with heart failure can be challenging, especially when critically ill in a decompensated state known as cardiogenic shock. Physicians in the cardiac intensive care unit (CCU) make decisions in a complex and stressful environment, rarely assisted by decision support tools. Currently, each physician makes these decisions based on their mental model of the patient's physiology, often oversimplified, together with mental predictions of possible outcomes of treatment. Furthermore, they must act fast in data rich environments, where the data are also complex, including patients' vital signs, imaging data, and continuous waveforms such as intravascular pressure monitoring or ECG telemetry. This can lead to fatigue and cognitive overload. Moreover, in addition to treatment decisions physicians deal with patient families and life or death decision planning. These challenges and more, such as physician-to-physician variation in decision making, can lead to errors and compromise patient results.

These challenges motivate projects that aim to predict patient outcomes, group patients with similar characteristics or even identify optimal treatment strategies adapted to each patient, projects that are being developed in the Aguirre-Houstis Lab in the Center of Systems Biology (CSB) group at Massachusetts General Hospital (MGH), Harvard Medical School (HMS). All these studies are approved by the Mass General Brigham Human Research Committee (MGBHRC), under the IRB number: #2020P003053. Concretely, there is a current project that proposes to train an algorithm to make therapy decisions in patients with cardiogenic shock due to decompensated heart failure (DHF).As discussed above, costs associated with hospitalizations for DHF can reach up to 60% of the total expenditures for the treatment of HF [19]. Therefore, this project addresses a common, resource intensive healthcare problem.

The present work, entitled "Applying Reinforcement Learning in Treatment Strategies for Cardiogenic Shock Patients", corresponds to the Master's Thesis of the Dual Master's Degree in Industrial

Engineering and Automatic Control and Robotics. This work is framed within the aforementioned project, in which the main goal is to develop a computational tool for helping physicians with a focused subset of decisions known as tailored therapy which is used to treat DHF. This tool, equipped with quantitative knowledge of physiology, past action-outcome events and the ability to systematically evaluate all the data, could aid physicians with valuable tailored therapy decisions, which could treat the patient's DHF state. The present Master's thesis is aligned with the objective of developing such a tool.

1.2. Project scope

This project is an initial study to derive a computational tool that could help physicians make optimal therapy decisions for patients with cardiogenic shock. In brief, patient data are extracted and employed to estimate the space of CV parameters that the simulator should work to reproduce patient states. Then, a policy is trained in simulation (modeling cardiogenic shock in heart failure patients), whose actions directly affect the parameters of the simulator. Finally, results from the policy behavior and the training process are assessed. The whole project implementation is performed in *Python 3*.

Because of the time frame and the circumstances that involve this project, some limitations must be kept in mind: First, the simulator implemented to train the policy is a simplified version of the final simulator. The one used in this project only includes the modelization of hemodynamics. However, the complete simulator will also include a model of oxygen transportation, in order to capture effects of oxygen delivery. Second, actions of the obtained policy affect directly to parameters of the CV model. In a future step, a mapping between drug administration and CV parameter changes will be developed. Thereby, policy actions will be realistic therapy decisions. Finally, patient's data are used to define the CV parameter space in which the simulator should work. In a next step, out of the scope of this project, data will also be used to further train and evaluate policies.

To sum up, the scope of this project is the implementation in *Python 3* of an algorithm that through simulation trains a policy to perform a simplification version of therapy decisions, affecting directly to the parameters of the CV model. Therefore, it is a first step towards the final implementation of a controller that would learn from a more realistic simulator, its training would be refined with patient-outcome historical data, and its actions would be truly physicians' decisions, such as certain drug administrations.

1.3. Project objectives

The general objective of this thesis is to obtain a policy with reinforcement learning (RL) to make therapy decisions in simulated patients with cardiogenic shock due to decompensated heart failure. This broad goal is subdivided into the following specific objectives:

- Implement a cardiovascular (CV) hemodynamics model. Concretely, understand and reproduce the Burkhoff and Tyberg cardiovascular model.
- Implement the Deep Q-Network (DQN) reinforcement learning algorithm to train policies employing the CV model as simulator.
- Develop a system identification (SI) tool for estimating CV model parameters from MGH patient data.
- Employ the DQN algorithm with optimized hyperparameters to train policies that learn therapy strategies from simulated cardiogenic shock patients, in which the parameters of the CV simulator are obtained from real patient data.

2. BACKGROUND AND STATE OF THE ART

Theoretical background related to the project is explained and presented in this chapter. Furthermore, this information is complemented with examples from published works, with a section specifically dedicated to reinforcement learning applied to the healthcare field (Section 2.4).

2.1. The Cardiovascular System

Basic physiological concepts that appear throughout this thesis are explained in this section. Moreover, decompensated heart failure and tailored therapy are introduced and described in more detail.

2.1.1. The Heart and Circulation

The cardiovascular system consists of the heart, which is a muscular pump, and a closed system of vessels called arteries, veins, and capillaries. It has four main functions, as explained below:

- The main function is the rapid convective transport of O_2, glucose, aminoacids and other components to the tissues, and the rapid washout of metabolic waste, such as CO_2 or urea.
- Another function is that acts as a control system. It distributes hormones to the tissues and secretes bioactive agents itself.
- Moreover, the cardiovascular system is essential for the regulation of body temperature, bringing heat from deep organs to the skin.
- Finally, in reproduction, it is also responsible of the hydraulic mechanism for genital erection.

The heart is a muscular organ about the size of a fist and it is the most complex component of the cardiovascular system. Its function is to pump the blood and circulate it through the vascular system. It is formed by four distinct chambers: the right atrium, the right ventricle, the left atrium and the left ventricle.

The right atrium receives blood from the superior and inferior vena cava, which through the tricuspid valve, ejects that blood into the right ventricle. The right ventricle is in charge of ejecting the received blood to the pulmonary artery, which passes the pulmonary valve. On the left side of the heart, the left atrium receives blood from the pulmonary veins and transmit it to the left ventricle through the mitral valve. Finally, the left ventricle ejects blood to the aorta, through the aortic valve. In Figure 2.1, a simplified illustration of the heart is shown.

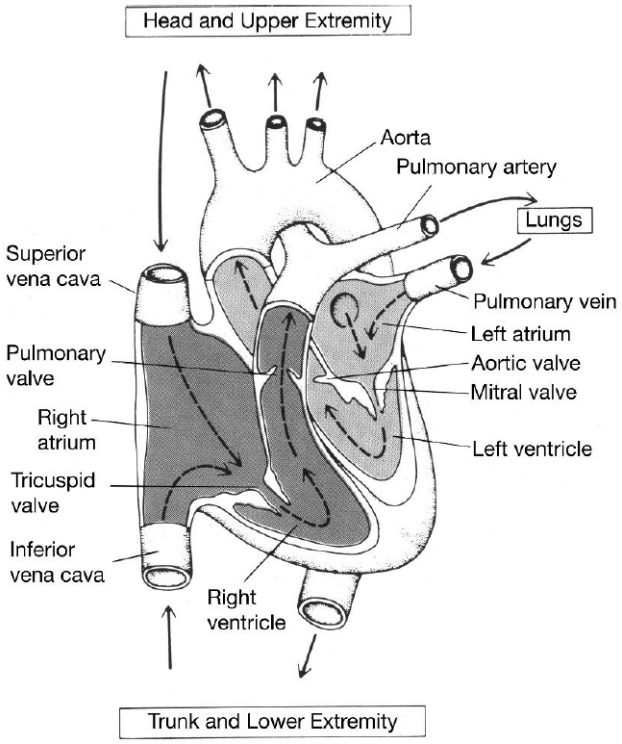
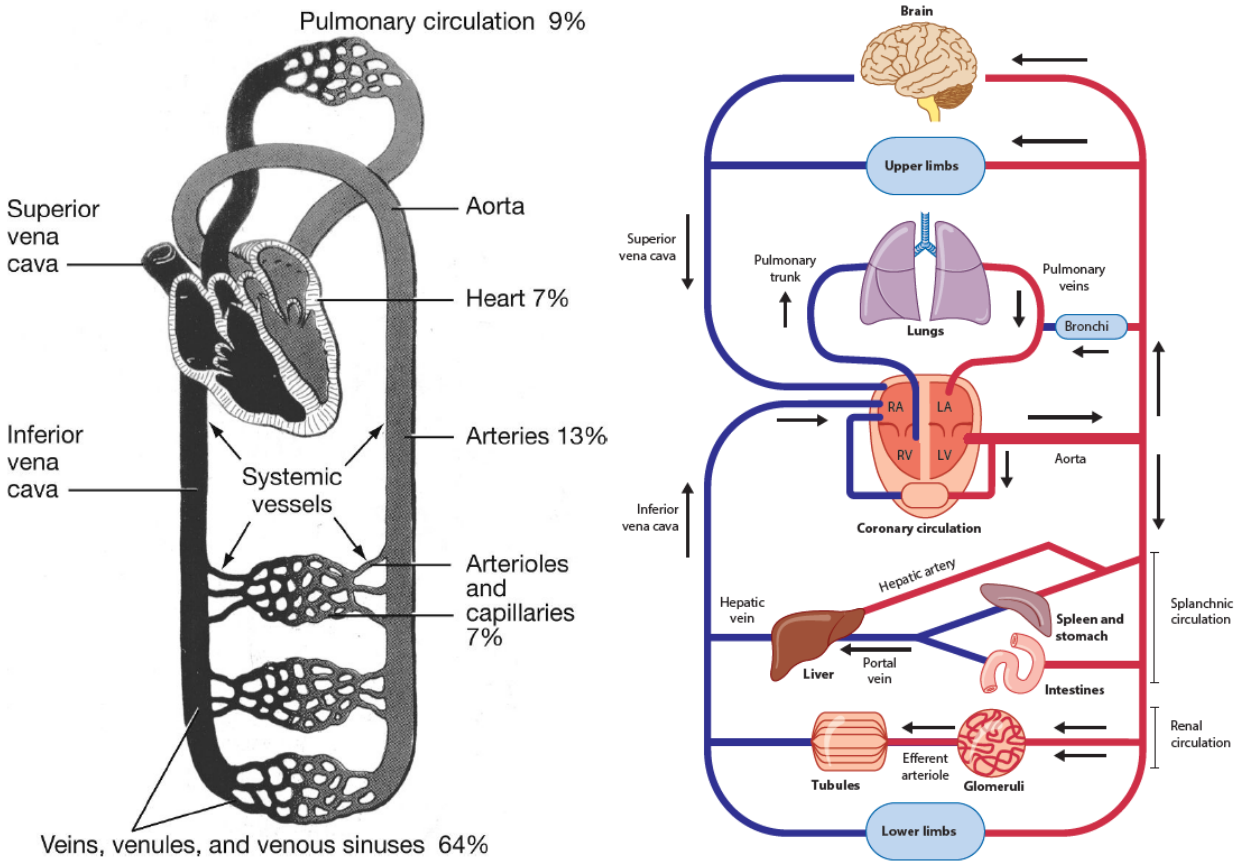


Figure 2.1 Schematic diagram of the heart with the four chambers and valves. Extracted from [12].

In addition to the heart, a vascular system is needed to deliver oxygenated blood to tissues and transport waste products to removal centers such as the lungs, liver, and kidneys. This circulatory system forms a closed loop for the blood flow. The left heart receives blood rich in O_2 from the lungs and pumps this blood into the systemic arteries. The systemic arteries, beginning with the aorta, branch out to smaller arteries that finally become the capillaries, where the exchange of O_2 and CO_2 takes place. Leaving the systemic capillaries, the blood enters the systemic veins, through which it flows in vessels that progressively augment their size to the vena cava. Then, the right heart pumps blood into the pulmonary arteries that distribute the blood to the lungs. The smallest branches of these arteries are the pulmonary capillaries, where CO_2 leaves and O_2 enters the blood. Finally, the oxygenated blood leaves the pulmonary capillaries and enters the pulmonary veins, through which it flows back to the left heart. A red blood cell takes approximately one minute to complete this circuit in an individual at rest.

The circulation described above is illustrated in Figures 2.2a and 2.2b. In Figure 2.2a the distribution of blood volume over the circuit is shown. Note that most of the volume is gathered in the systemic veins. On the other hand, in Figure 2.2b, a more illustrative circulatory system is shown, where the most important vital organs are presented. Note that vessels colored red carry blood rich in O_2 and vessels in blue carry blood depleted of O_2 and enriched in CO_2.



(a) (b)

Figure 2.2 Illustrations of the circulatory system. a) Schematic diagram, showing both the systemic and pulmonary circulations, the heart and the distribution of blood volume. Extracted from [12]. b) Illustration of human circulation with some vital organs. Red lines are blood rich in O_2 and the blue lines blood that contains more CO_2. Extracted from[8].

In addition to its topology, the function of the cardiovascular system is characterized by a set of physiologic variables. Since the blood is nearly incompressible, the volume (V) is a convenient measure of its quantity. In general, blood volume is measured in milliliters [mL]. Moreover, its time derivative is another key variable: the blood flow (Q), usually measured in liters per minute [L/min]. Also, the pressure (P) is used to describe the blood inside vessels, which in general is measured in millimeters of mercury [$mmHg$].

The total blood flow generated by the heart, termed the cardiac output (CO) and measured in liters per minute [L/min], is especially important as it determines the rate of oxygen delivery to tissues. It is decomposed further into two cardiac state variables, heart rate and stroke volume. Heart rate (HR) refers to the number of muscle contractions the heart completes per minute, measured in beats per minute [bpm]. Stroke volume (SV) refers to the volume of blood ejected with each beat.

Hence, cardiac output can be quantified as the product of these variables:

$$CO = SV \times HR \tag{2.1}$$

Additional relations exist between the physical variables defined above. Blood vessels are considered to have a resistance to blood flow, and a compliance in response to distending pressure. The simplest models of these relationships are linear [9]:

$$Q = \frac{\Delta P}{R} \tag{2.2}$$

$$V = CP \tag{2.3}$$

where R is known as the resistance of the vessel, in general measured in $[mmHg \cdot s/mL]$; and C is the compliance of the vessel, usually measured in $[mL/mmHg]$.

The heart generates flow by repeatedly ejecting blood in a characteristic "cardiac cycle". Each cycle (heart beat) consists of a contraction (systole) and a relaxation (diastole) that results in the ejection of a quantity of blood, the previously defined stroke volume (SV). The entire cycle is divided into four phases, based on the position of the heart valves: ventricular filling, isovolumetric contraction, ejection and isovolumetric relaxation.

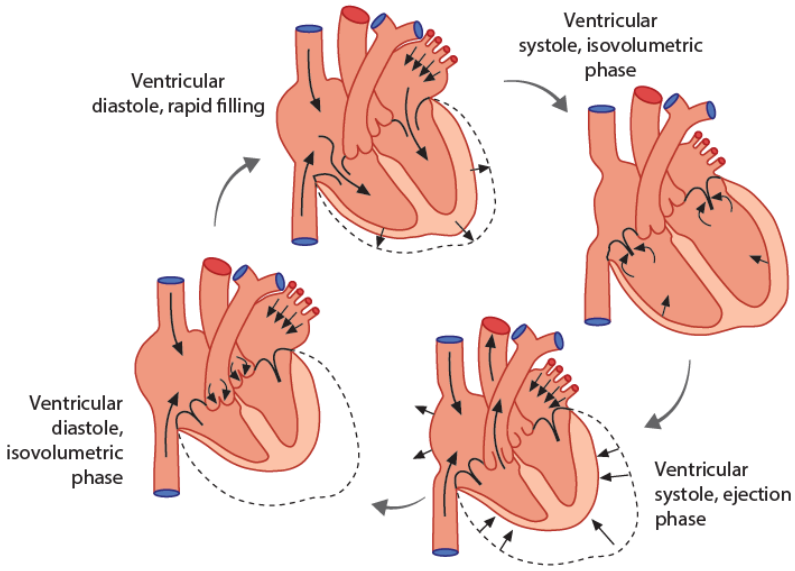


Figure 2.3 The four phases of the cardiac cycle. From top and clockwise: filling, isovolumetric contraction, ejection and isovolumetric filling. Adapted from [8].

In ventricular filling, both tricuspid and mitral valves are open, and pulmonary and aortic valves are closed. The heart is relaxed and filled with blood. It is usually the longest phase of the cycle, taking two thirds of it. In isovolumetric contraction, all four valves are closed. The heart starts contracting

and because of ventricles have temporarily become a closed chamber, pressure of blood rises quickly. During ejection, mitral and tricuspid valves are closed, and pulmonary and aortic valves are open. The heart is contracted and blood is ejected to the arteries. Finally, in isovolumetric relaxation, again all four valves are closed. Heart relaxes and since it is again a closed chamber, pressure drops rapidly. The explained cycle can be appreciated in Figure 2.3, in which the four phases are depicted.

The state of a cardiac chamber during the cardiac cycle, e.g. the left ventricle, can be represented by the so called PV loop. This loop expresses the chamber's state in terms of its volume and pressure as a function of time. It is very useful for discussing the heart performance in a quantitative manner. Moving counter-clockwise, the four phases can be appreciated. Figure 2.4 shows an example of PV loop of a subject resting. For example, the stroke volume (SV) can be determined, since it is defined as the end-diastolic minus the end-systolic volumes.

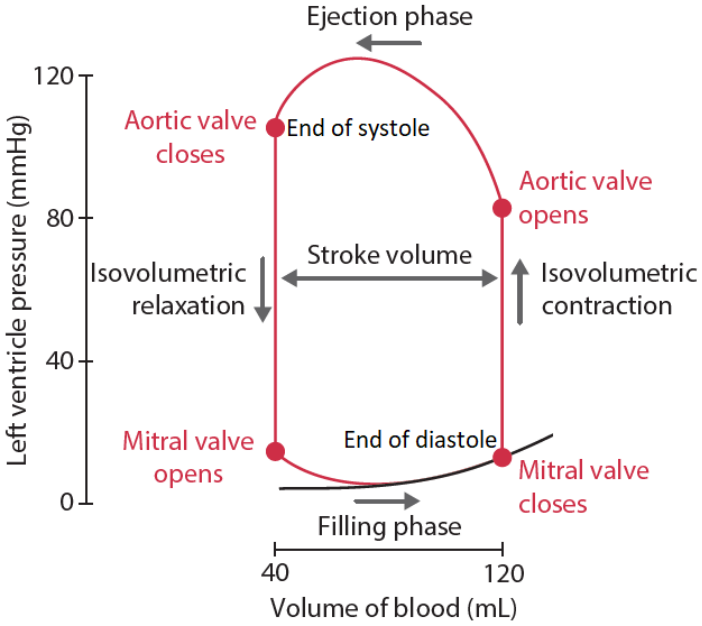


Figure 2.4 Example of a left ventricle pressure volume loop (PV loop). Adapted from. [8]

2.1.2. Decompensated Heart Failure

Heart failure can be defined as the inability of the heart to pump sufficient blood to meet the body's full range of demands and/or to do so at abnormally high pressure. Decompensated heart failure (DHF) is a more advanced form when O_2 delivery by the heart no longer meets the body's minimum demands and requires immediate therapeutic intervention. As presented in the introduction (Section 1.1), costs in healthcare for DHF reach approximately 60% of the total costs related to the treatment of heart failure. It is estimated that its prevalence is around 10 % in the population over 70 years of age. Moreover, mortality rate among patients discharged during the first 90 days is approximately of 10%, with about 25% of readmission in this period [19].

The most severe form of DHF is cardiogenic shock (CS). In CS O_2 delivery by the heart is so impaired that rapid organ damage occurs, in particular those organs with higher O_2 demands such as the kidneys and brain. As a result CS is rapidly fatal. The treatment strategy for CS should be carefully guided by patient's vital signs, physical examination, laboratory tests, echocardiography, etc. in order to reverse the decompensated state [14]. Tailored therapy [29] is a clinical algorithm that follows this strategy.

2.1.3. Tailored Therapy

As presented Section 2.1.2, tailored therapy is a clinical algorithm that uses patient's physiology data to choose treatments that reverse the decompensated state avoiding CS. The goal is to restore O_2 transport by restoring the heart's normal loading conditions (preload and afterload) and inotropic state. This is achieved by dosing vasoactive medications and using artificial mechanical support.

However, what are the heart's loading conditions and inotropic state? Preload, also known as the left ventricular end-diastolic pressure (LVEDP), is the amount of ventricular stretch at the end of diastole. It directly affects the ventricular filling of the heart. Therefore, a high preload is related with increased stroke volume (SV), and vice versa, a low preload results in a lower SV. Afterload is the condition that affects the amount of resistance that the heart must overcome to open the aortic valve and eject the blood volume into the aorta. It is both related to the aortic pressure and the SV. Thereby, an increased afterload results in a high resistance and thus, a lower stroke volume and higher pressures. If the afterload is low, the opposite happens. Finally, inotropy is the heart state related to the heart's force of contraction and its elastance. It is mainly linked to the SV, and an increment of inotropy causes a reduction of the end-systolic volume and thus a higher SV [15].

Left ventricular pressure volume (PV) loops mentioned in Section 2.1.1 are a very useful tool for visualizing such effects mentioned in the previous paragraph. In Figure 2.5, both preload and afterload as well as inotropy effects are schematized.

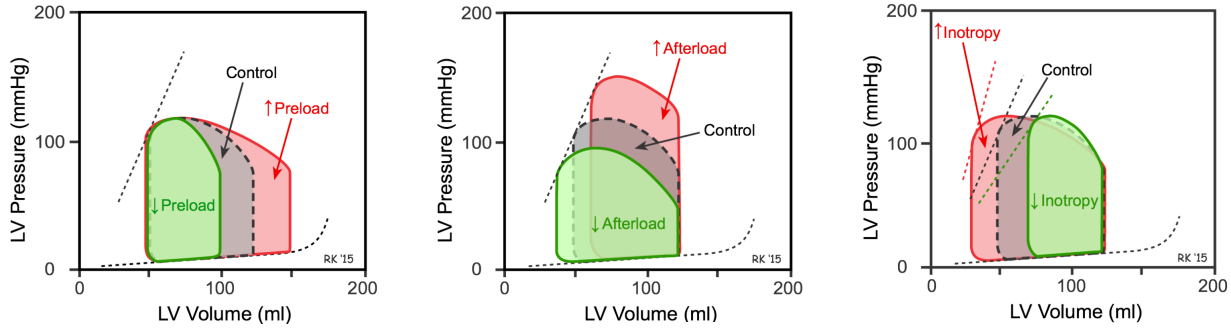


Figure 2.5 PV loops showing the effect of the heart's loading conditions and inotropic state. Extracted from [15]. a) Effect of preload related to SV. b) Effect of afterload related to both SV and pressure. c) Effect of inotropy related to SV.

2.2. Cardiovascular System Modeling

A cardiovascular (CV) system model is a mathematical abstraction of cardiovascular physiology that is used to understand and quantify the system and its functions. Many modeling choices are available, and as described in Section 2.1.1 the CV system may have multiple functions. As a result, one may ask themselves what function or part of the cardiovascular system wants to model. Is the heat transfer from organs to the skin desired to be modeled? Or the fluid dynamics of the blood? Furthermore, one must determine the scale at which to model the system. Should the model describe interactions between cells? Or are macro-scale measurements such as pressure sufficient?

Cardiovasculer system modeling therefore embraces many types of models and distinct scales. In this section a brief review of some modeling approaches is presented, with an emphasis on the type of model that is employed in this project.

2.2.1. Multiscale Modeling Approaches

The cardiovascular system can be modeled at different length and time scales [35]. For example, one could model at the scale of a cell in order to study and predict effects of perturbations such as genetic mutations, the effects of certain drugs, or disease consequences on cells. One could also model at the scale of tissue. These models are used to study and quantify arrhythmias, which can be viewed as emergent properties of the cardiac system at the scale of tissue. Such models can also be used to investigate anatomical abnormalities and interactions with devices. Zooming out, there are organ scale models which can be used to model patient-specific geometry and cardiac structure. Finally, there are the whole body models, which can quantify functional variables such as pressure or volume in a cardiovascular compartment. Examples of the aforementioned models can be seen in Figure 2.6.

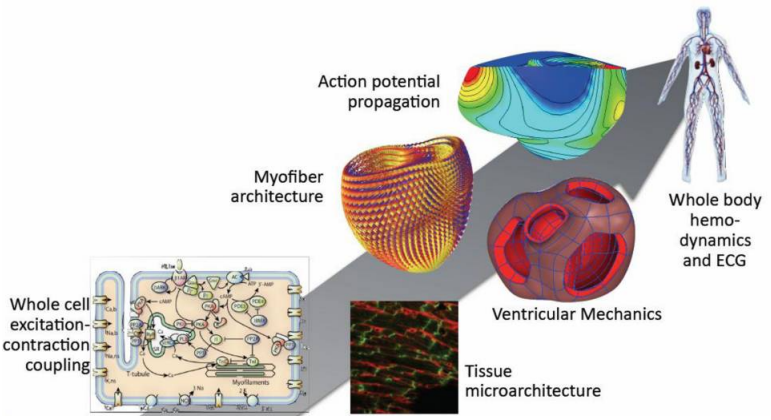


Figure 2.6 Multiscale modeling. From bottom to the top: cell scale modeling, tissue/fiber scale modeling, organ (cardiac electrical and mechanical) modeling and finally whole body modeling. Adapted from [35].

In this work, a whole body model is used to simulate the physical variables that describe the blood circulation.

2.2.2. Cardiovascular Hemodynamics Modeling Approaches

Models that describe the physical variables that quantify cardiovascular characteristics are called hemodynamic models. A range of such models describe the CV system at distinct levels of detail. There are three main classes of hemodynamics models. These include, in order of decreasing complexity: three dimensional (3D) hemodynamic models, one dimensional (1D) hemodynamic models and zero dimensional (0D, or lumped parameter) hemodynamic models [18].

3D hemodynamic models are those that study the blood as a vector field. Blood can be represented as an incompressible fluid. Its fluid dynamic behaviour can then be approximated by a Newtonian fluid model governed by the Navier-Stokes equations. Arterial vessels usually have compliant walls, which deform significantly due to the pulsation of blood pressure. In order to model this compliance, elastodynamic equations are coupled with Navier-Stokes equations. Due to their complexity, such models are usually used to describe local hemodynamic information of vessels in detail. However, they are rarely used to simulate the entire circulatory system.

1D hemodynamic models are those that study the blood as a wave propagating in a tube. Again, blood is considered an incompressible fluid. The 1D partial differential equations that govern this wave can be obtained by integrating the Navier-Stokes equations over the vessel's cross-section. These models, much less complex than the 3D models, are usually used to simulate pressure and flow waveforms at any point of the arterial network, such as the central aortic pressure. Nevertheless, their complexity and computational demands make them challenging to use for simulating an entire circulatory system.

0D hemodynamic or lumped parameter models are those in which the circulatory system is often modeled by applying Windkessel theory [37]. The hypothesis underlying this theory is that the pressure in a deformable fluid container is homogeneous in space. Consequently, the circulatory system is grouped into different compartments that represent a specific part of it. These models are defined with ordinary differential equations, making them relatively simple to solve. Due to their simplicity, these models are not employed to study wave propagation or pressure waves in detail, but rather they are useful to estimate blood pressure changes [1], simulate the whole circulatory system [3] or for teaching purposes [13].

The type of model implemented in this project is a lumped parameter model, as it serves as a simulator used by a reinforcement learning algorithm to train policies. This model is explained in detail in Section 3.2.

2.3. Machine Learning

This section is devoted to describe the basics of machine learning topics that appear in this thesis. A brief introduction to artificial neural networks is explained, together with the fundamental theory related to reinforcement learning.

2.3.1. Artificial Neural Networks

Artificial neural networks (ANNs) are a type of supervised learning method that aim to learn a function from a set of input-output (or label) data. The simplest entity of an ANN is the so called perceptron. A perceptron is a linear combination of variables passed to an activation function (f):

$$y = f(\mathbf{x}\boldsymbol{\omega}^T + b) \tag{2.4}$$

where $\mathbf{x}$ is the vector of inputs, $\boldsymbol{\omega}$ the vector of weights and b the bias. Regarding activation functions, there are many different types, including ReLu, Sigmoid or ELU [7].

Therefore, an ANN is formed by a set of perceptrons (or nodes) that are organized into different layers and linked among them. Those layers are: the input layer, the output layer and the hidden layers. The simplest architecture of ANN is called a multilayer perceptron (MLP) or feedforward neural network. In these kind of networks, the nodes of each layer are connected to the next layer, thus information travels forward, from the input layer to the output layer. In Figure 2.7 a simple architecture of a MLP is shown.

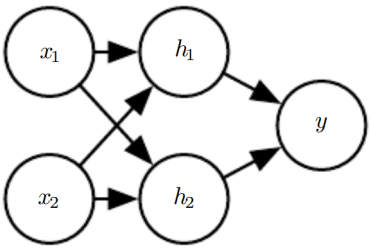


Figure 2.7 Schema of a simple artificial neural network. Input nodes are denoted with x, hidden nodes with h and output node with y. Adapted from [7].

In order to train (optimize) an ANN, a function that quantifies the difference between the labels (or data outputs) and the outputs of the ANN is needed. This function is generally called the loss function:

$$\mathcal{L}(y(\boldsymbol{\omega}), \tilde{y}) \tag{2.5}$$

in which $y(\boldsymbol{\omega})$ are the outputs of the ANN that depend on its weights $\boldsymbol{\omega}$ and $\tilde{y}$ are the labels.

Finally, the training (optimization) of the ANN is carried out by employing the backpropagation algorithm. This algorithm is based on updating the weights of the ANN "backwards" using the

gradient descent method. It computes the gradient of the loss function with respect to the weights and propagates this error updating them:

$$\boldsymbol{\omega}^{t+1} = \boldsymbol{\omega}^{t} - \alpha \frac{\partial \mathcal{L}(y(\boldsymbol{\omega}), \tilde{y})}{\partial \boldsymbol{\omega}} \tag{2.6}$$

where t denotes the update iteration and α is the learning rate. Iterating over the whole dataset, backpropagation is applied each time a batch (subset) of data is introduced to the ANN, and repeated a certain amount of epochs. An epoch is defined as the moment when all the dataset has been introduced to the ANN in batches.

2.3.2. Reinforcement Learning

Reinforcement learning (RL) is one of three well-known machine learning paradigms, alongside supervised learning and unsupervised learning. It is an area of machine learning concerned with how a virtual agent should choose actions when interacting with an environment, with the goal of maximizing the notion of a cumulative reward [30] (Figure 2.8).

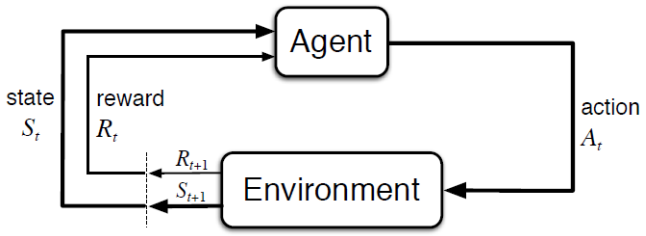


Figure 2.8 Reinforcement learning framework agent-environment. Extracted from [30].

The environment is typically modeled as a Markov Decision Process (MDP). A MDP is a discrete-time stochastic control process. It provides a mathematical framework for modeling decision making in situations where outcomes are partly random and partly under the control of an agent. A MDP is composed of a tuple of 4 elements:

$$(\mathcal{S},\, \mathcal{A},\, \mathcal{P}_a(s, s'),\, \mathcal{R}_a(s, s')) \tag{2.7}$$

where $\mathcal{S}$ is the set of states called the state space; $\mathcal{A}$ is the set of actions called the action space; $\mathcal{P}_a(s, s') : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$ is called the transition function, it determines the probability of going from state s to state s' taking action a; finally, $\mathcal{R}_a(s, s') : S \times S \times A \to \mathbb{R}$ is the instantaneous reward, and it defines the reward for going from state s to state s' when taking action a.

Two key functions in RL are the policy and the value functions. The policy is a mapping between states and actions that determines what action the agent has to take given the state and the set of possible actions. The optimal policy determines the actions that maximize a cumulative reward.

$$\text{Policy:} \quad \pi(s, a) : \mathcal{S} \times \mathcal{A} \to \mathcal{A} \tag{2.8}$$

On the other hand, the value function is defined as the expected return starting from state s_0 and successively following the policy π:

$$V_\pi(s) = \mathrm{E}[R] = \mathrm{E}\left[\sum_{t=0}^{T} \gamma^t r_t \mid s_0 = s\right] \tag{2.9}$$

where R is the return, T is the horizon, $\gamma \in [0, 1)$ is the discount factor and $r_t \in \mathcal{R}_{a_t}(s, s'|s = s_t, s' = s_{t+1})$, given transition from s to s' by $a_t = \pi(s_t, a)$ is the reward at step t. Note that an episode is defined as the whole agent interaction with the environment until the horizon T is reached.

The concepts described above are shared across a wide range of RL algorithms. However, there are also many important distinction between algorithms. For instance, there are algorithms that have discrete state and action spaces, others that have both continuous and others that can have continuous state space and discrete action space. Moreover, algorithms can be on-policy if they optimize the policy using actual values of it, or off-policy if they do not use them. Finally, algorithms can be model-based if they have a model of the environment while training (optimizing) the policy, or model-free, if they train the policy without a model and learn directly from the environment.

With respect to the distinction between model-free vs. model-based algorithms, model-free algorithms are usually employed for those tasks where the underlying dynamics can be difficult to model. For example, in [21] a model-free RL algorithm was used to play Atari games. Also, the rubik's cube problem was solved with a robotic hand vastly trained in simulation [22]. The authors reported the RL had achieved meta-learning, meaning that their model-free RL algorithm was able to learn the dynamics of the environment, rather than memorizing all pair of cause-effect relations. On the other hand, problems that are easy to simulate such as board games have been better solved with model-based RL algorithms. For example AlphaZero was able to outperform any human or engine in chess, shogi and Go [28].

Nevertheless, it must be pointed out that during the last years, model-based RL algorithms have also been able to solve problems where the underlying dynamics are complex. For example, in [34] they could find "universal" policies for robotic tasks where the dynamics were not completely known. By introducing a system identification model that was capable of detecting the parameters of the environment, they could introduce the estimated parameters to the policy and therefore optimize it over different environment setups. However, the most astonishing achievement is the case of MuZero [27]. MuZero is the successor of AlphaZero that builds itself the model that uses to run the RL algorithm and optimize its policy. It learns a model that predicts the quantities most directly relevant to planning, which makes it outperform on not only board games like AlphaZero, but also in Atari games, where previously model-based RL algorithms have historically struggled.

2.4. Reinforcement Learning in Healthcare

In recent years, RL techniques have been introduced into healthcare applications for use as decision support tools. There are different clinical settings in the intensive care unit (ICU) where RL has been studied. For example, RL was used for the management of invasive mechanical ventilation in the ICU [24, 33], for learning optimal treatment strategies for sepsis [17] or even for learning effective cancer treatment regimes that dynamically adapt to patients [36]. However, no applications have been reported for the critical care unit (CCU) or specifically for cardiogenic shock.

Regardless of any application of RL to ICU problems, there are constraints that affect on those and must be taken into consideration. First, RL algorithms cannot learn through experimentation (trial and error) on actual patients. Learning from patient data and evaluation of trained policies must only happen through training on databases of past physician actions and their associated patient outcomes (off-policy learning and evaluation [10]).

Second, these databases tend to have a small number of examples relative to the needs of RL to train a policy. For example, for complex tasks, RL may need millions of examples or more to obtain a trained policy. This motivates the need to either find algorithms that are extremely sample-efficient [6], or to use simulators to create many more cause-effect samples.

Furhtermore, RL approaches to ICU problems have tended to use model-free algorithms, wherein they begin with no explicit knowledge of physiology or medicine. They learned to make decisions about patients simply by observing a complex dataset generated by a collection of physicians in thousands of patients. Published results reflect the limitations of this approach, for example making decisions that ignore basic cause effect relationships well understood by physicians [11].

Although there are some studies in recent years that have applied RL approaches to the healthcare field, many challenges remain in order to build robust policies that could operate in the real-world. In particular, RL applied to cardiogenic shock is a novel and promising area of RL development.

3. MATERIALS AND METHODS

This chapter describes the tools and techniques used in this thesis. Detailed explanations are provided for data acquisition, simulation of cardiovascular physiology, the reinforcement learning framework, and finally the system identification tool.

3.1. Clinical Data Acquisition

The patient data used to train the system identification tool were obtained from two Massachusetts General Hospital (MGH) databases. The step-by-step process, beginning with data extraction from bedside monitors and ending with the final dataset, is presented next.

3.1.1. Massachusetts General Hospital Databases

MGH systematically monitors and records all patient data in an electronic medical record (EMR). Data come from different sources, e.g. vitals signs recorded at the bedisde, past clinic visits, medication administration, etc. Once data are recorded, they are stored in two principle databases: Electronic Data Warehouse (EDW) and Bedmaster (BM).

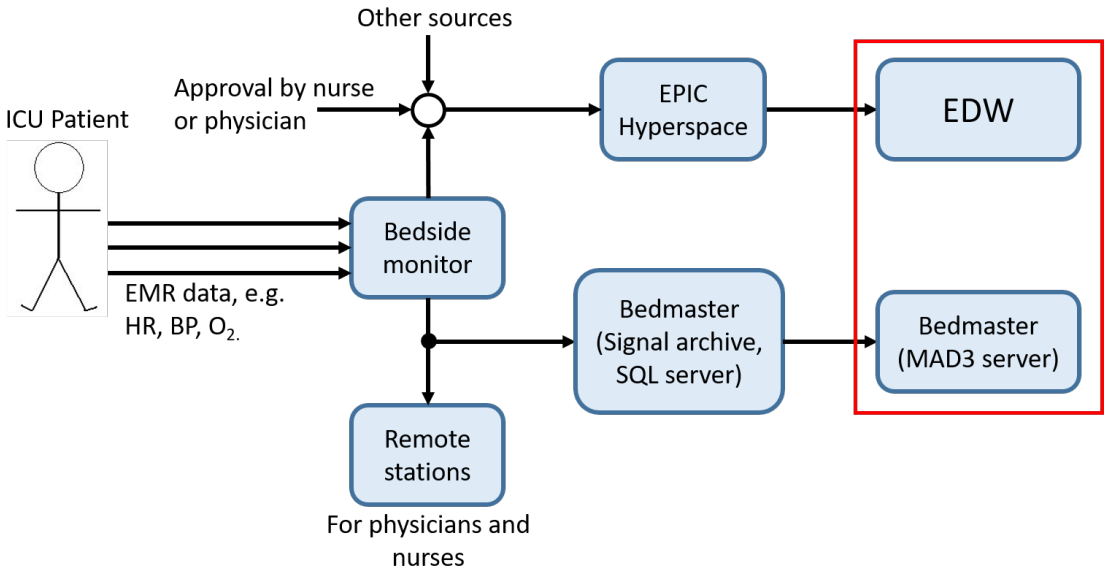


Figure 3.1 Data workflow. From ICU patient to EDW and Bedmaster databases. Framed in red both EDW and BM databases.

The workflow from the extraction of ICU patient's data to their storage into the aforementioned databases is schematized in Figure 3.1. Patient data such as ECG telemetry or blood pressure are tracked by a bedside monitor and displayed onto remote stations for physicians and nurses to review. A subset of these data are checked and approved by a nurse or physician and then sent to EPIC Hyperspace (a commercial EMR) and subsequently to EDW. Other sources of data, like lab measurements or physician notes are also incorporated into EPIC and EDW. There are also a collection of densely-sampled physiologic signals recorded from patients that are viewed in real-time and sent to the BM database without filtering by a nurse or physician.

The first of these databases, EDW, is a SQL-based database that contains information from EPIC and other data sources. Data formats range from numerical values to dates, notes or labels. Information is organized in several tables that can be queried, obtaining a set of `csv` files. Typical patient information found in this database includes: vital signs at hospital admission, their movements between MGH departments, demographics, diagnoses, chart-review-items (vital signs measurements, among others), laboratory value measurements, medication administration, surgeries, important events and patient history data (such as tobacco consumption).

The second database, BM, contains two types of data: patient vital signs and patient waveforms. The data are stored in a collection of `mat` files that can be accessed on a server (MAD3). These files are stored in subfolders corresponding to each MGH department and stripped of any patient identifiers; the only identifying information that is retained is the hospital bed number from which the data were recorded and the start time of recording. Vital signs are sampled every two seconds, and have information such as a patient's heart rate, systolic blood pressure, respiration rate, etc. On the other hand, waveforms are sampled at $120Hz$ or $240HzHz$ and contain signals such as ECG, systemic arterial pressure, central venous pressure, etc.

3.1.2. Data Curation

Data curation is the act of organizing and integrating data collected from various sources, so as to preserve it over time and make it available for applications. Part of the Houstis-Aguirre Lab team dedicated time and effort to curate the data by developing a pipeline to extract information from both EDW and BM databases and create `hd5` files (Figure 3.2). Two key components of this pipeline are the matching algorithm and the tensorization process.

As mentioned in the previous section (Section 3.1.1) BM data are de-identified. In order to associate a BM file with the patient from which it was measured, it must be matched with patient identifiers in the EDW database. The matching algorithm does this task by comparing hospital bed and start time information in BM files against bed movement information stored in EDW.

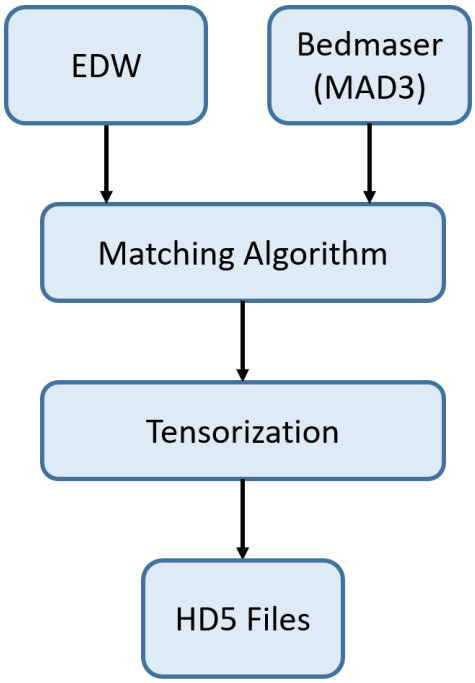


Figure 3.2 Data workflow. From EDW and Bedmaster databases to HD5 files.

Once a patient's data from both databases are matched, the tensorization process follows. Tensorization is the operation of getting the desired data from EDW and BM and organizing it hierarchically in `hd5` files, one for each patient. Two important operations in this process include: time alignment between data from both databases, since EDW data is stored in local time but BM data in UTC; and corrections on BM time arrays, as sometimes there are non-monotonicities such as jumps back in time.

Storing the data in `hd5` files has several advantages. First, there is one file per patient; so if a cohort of patients is desired, it is easy to collect the desired data. Second, information in `hd5` is easy to organize hierarchically and has the flexibility to be compressed into different formats. Moreover, `hd5` files have the characteristic of only loading in RAM memory the desired data, instead of the entire patient record. These features and more make this type of files suitable for data analysis applications.

3.1.3. Cohort of Patients

The cohort of patients chosen for this project are MGH patients who underwent coronary artery bypass grafting (CABG). CABG is a type of surgery that improves blood flow to the heart. It is used for people who have severe coronary artery disease, a condition that narrows the lumen of the coronary arteries, thereby reducing blood flow to the heart muscle. Patients who undergo CABG surgery sometimes develop complications, in particular cardiogenic shock (CS) [20]. Therefore, this cohort is interesting and relevant for this thesis as it has data that can describe CS patient states.

Figure 3.3 shows a flowchart for selecting CABG patients who satisfy criteria of data availability and data quality. The initial CABG cohort consisted of 2774 patients and 2812 encounters that occurred in MGH from March 2016 to February 2021. Note that an encounter refers to a specific instance when a patient was admitted to the hospital. Therefore, a given patient that underwent more than one CABG operation, during separate hospital stays, will have more than one encounter. Moreover, only the first 12 hours after the surgery were analyzed, since it is the period of time in which patients are heavily monitored in order to diagnose any complications that may occur.

From the initial cohort, 1002 patients and 1039 encounters were removed from the data set because they did not have all the desired signals used in this work. Moreover, those patients who had signals which overlapped less than 20% of the total time, i.e., less than 2.5 hours, were also removed from the cohort. Finally, after outlier removal, patients whose signals overlapped for less than 20% of the 12 hours time window were again removed from the data set. The end result of patient selection cut down the initial cohort of 2774 patients and 2812 encounters to 1372 patients and 1373 encounters, which equals 48.8% of the initial cohort.

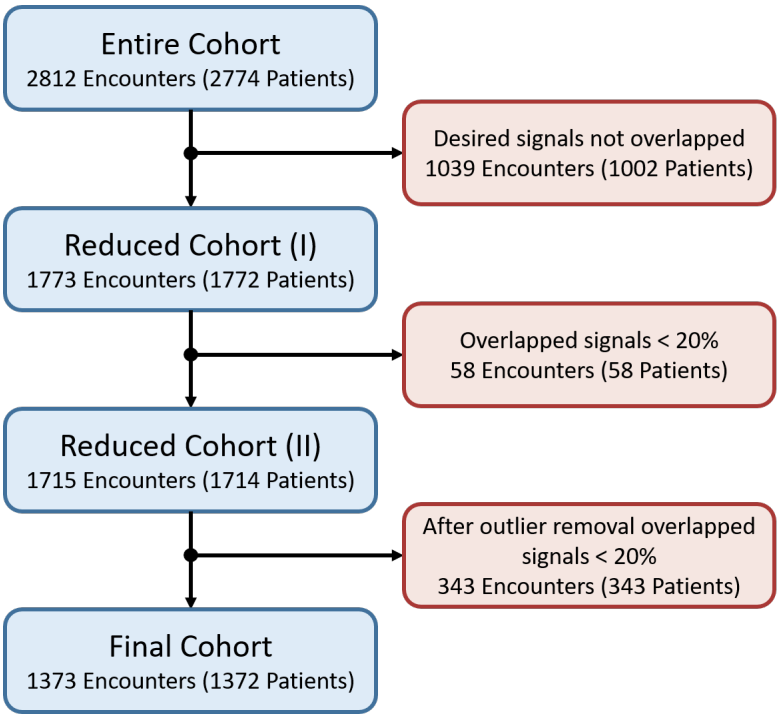


Figure 3.3 Cohort reduction due to signal missingness and outlier removal.

Demographic information for the final cohort is presented in Table 3.1. Patients were predominantly male (80%), at a mean age of 69 years, with a mean length of hospital stay of about 12 days. Moreover, the mortality of this cohort is nearly 6%.

	Mean	Standard Deviation	Min Value	Max Value	Percentage [%]
Age [yr]	68.83	9.68	24	93	
Sex (Male/Female)					79.4/20.6
Weight [kg]	86.50	18.40	42.30	183.10	
Height [m]	1.72	0.09	1.40	2.03	
Length of stay [h]	298.76	255.46	111.97	3788.77	
End of stay type (Alive/Deceased)					94.3/5.7

Table 3.1 Demographic data of the final cohort ($n = 1373$).

3.1.4. Preprocessing and Final Dataset

As justified and explained in Section 3.4, the signals used in this project are the patient's systemic arterial pressure, central venous pressure, pulmonary artery pressure, cardiac output, and heart rate (Table 3.2). All of these signals come from BM database, although it must be noted that information from EDW was also employed to determine which patients underwent CABG, as well as to obtain demographic information (Table 3.1).

Signal Name	Signal Type	Symbol	Units	Description
Arterial pressure	Waveform (240 Hz)	$P_{a,s}$	*mmHg*	Patient's arterial pressure measured with a catheter placed in the radial artery.
Central venous pressure	Waveform (240 Hz)	$P_{v,s}$	*mmHg*	Patient's venous pressure measured with a catheter placed in the jugular vein.
Pulmonary artery pressure	Waveform (240 Hz)	$P_{a,p}$	*mmHg*	Patient's pulmonary artery pressure measured with a catheter placed in the pulmonary artery.
Cardiac output	Vital sign (0.5 Hz)	*CO*	*L/min*	Patient's cardiac output measured with thermodilution
Heart rate	Vital sign (0.5 Hz)	*HR*	*bpm*	Patient's heart rate measured from the ECG telemetry.

Table 3.2 List of signals used in the project. Included information: the signal type and between parentheses their sample frequency, the equivalent mathematical symbol used throughout the whole study, the units, and a brief description of how they are measured.

Once the final cohort of patients was selected, a set of preprocessing steps were performed in order to clean and transform the data required for CV parameter estimation using the system identification tool. First, signals were cleaned out of outliers by determining those values that were outside physiological ranges (Table 3.3 present these ranges). Second, all five signals were resampled at $120Hz$ and cut so as to have the same sample points and start/end time. Third, a *Butterworth* low-pass filter with a cut-off frequency of $8Hz$ was passed on the waveforms to remove remaining noise. The cut-off frequency of $8Hz$ was chosen as it is double of the highest heart rate, in this way ensuring no information from the cardiac cycles could be lost. Fourth, the signals were cut into cardiac cycles, by detecting the local minima of the arterial pressure signal (i.e. the diastolic pressure). Finally, corrupted or noisy signals that did not correspond to true cardiac cycles were removed, since artifacts in the signal sometimes introduced local minima that did not correspond to

the diastolic pressure. This was achieved by employing the dynamic time warping (DTW) technique, computing the similarity between a template cardiac cycle and the cycles cut from the signals.

Signal Name	Min Value	Max Value	Units
Arterial pressure	20	250	*mmHg*
Central venous pressure	0.5	40	*mmHg*
Pulmonary artery pressure	5	60	*mmHg*
Cardiac output	1.5	15	*L/min*
Heart rate	20	250	*bpm*

Table 3.3 Minimum and maximum achievable values for each cardiovascular system measurement, used to define ranges for outlier removal.

As a final step, systemic arterial pressure, central venous pressure and pulmonary artery pressure were resampled at 50 samples per cardiac cycle. On the other hand, for cardiac output and heart rate, the mean over all the cycle was computed, taking only one value for the entire cycle. After performing all the aforementioned preprocessing steps, the final dataset used for parameter estimation contained 776341 cardiac cycles with information from all five signals. Summary statistics from the final dataset is shown in Table 3.4.

Name	Symbol	Min	Max	Mean	Std Dev	Units
Arterial pressure	$P_{a,s}$	24.75	204.70	74.63	19.93	*mmHg*
Central venous pressure	$P_{v,s}$	0.82	19.77	7.88	3.24	*mmHg*
Pulmonary artery pressure	$P_{a,p}$	3.49	48.48	20.24	6.00	*mmHg*
Heart rate	HR	27.36	194.88	86.77	9.54	*bpm*
Cardiac output	CO	3.49	48.48	20.24	6.00	*mL*

Table 3.4 Summary statistics for the final dataset (n=776341) with systemic arterial pressure ($P_{a,s}$), central venous presure ($P_{v,s}$), pulmonary artery pressure ($P_{a,p}$), heart rate (HR) and cardiac output (CO) signals.

3.2. Cardiovascular Hemodynamics Model

The cardiovascular (CV) model used to simulate the whole body hemodynamics during reinforcement learning training is the Burkhoff and Tyberg model [1]. This model is a 0D or lumped parameter model (see Section 2.2.2), in which the whole CV circulatory system is divided and grouped into 6 compartments: the left ventricle, the systemic arteries, the systemic veins, the right ventricle, the pulmonary arteries, and the pulmonary veins. Each compartment has its own cardiovascular parameters. Also, note that atria are included as part of the venous compartments and not modeled independently.

In order to model each compartment, it is assumed that large arteries and veins are primarily compliance vessels, meaning that only small pressure gradients are needed to propel blood flow (cardiac output) through them. In contrast, their change in volume can be highly significant. On

the other hand, resistance to blood flow is concentrated in the tissues themselves (mainly arterioles and valves), where volume changes are less important but resistances give rise to large blood pressure drops. Moreover, compliance and resistance effects are modeled linearly (Equations 2.2 and 2.3, respectively). In Figure 3.4a the six compartments with the main CV parameters are schematized. In Figure 3.4b an equivalent electrical model is shown, which is helpful for understanding the equations presented in Section 3.2.2.

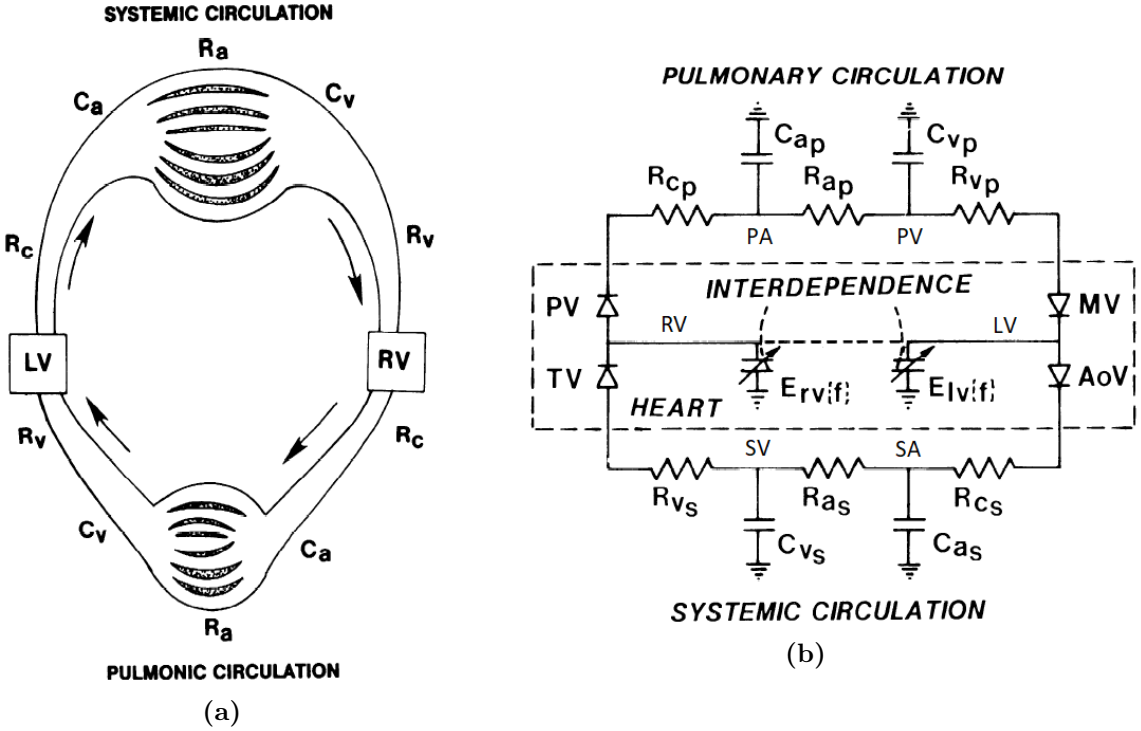


Figure 3.4 Diagrams of the Burkhoff and Tyberg model. a) Six compartment model (parameters explained in Section 3.2.1). RV and LV stand for right and left ventricles, respectively. Extracted from [1]. b) Equivalent electrical circuit of the CV model (parameters explained in Section 3.2.1). RV and LV stand for right and left ventricles, respectively. SA and SV, systemic arteries and veins, respectively. PA and PV, pulmonic arteries and veins, respectively. MV, AoV, TV and PV, mitral, aortic, tricuspid and pulmonar valves, respectively. Adapted from [26].

3.2.1. Model Parameters

Model parameters can be divided into three groups, including those that characterize the heart, the circulation, or both. Heart parameters are those related to the time-varying ventricular contraction, as described next. The end-systolic elastance (E_{es}) denotes the highest elastance of the heart during the cardiac cycle. The moment of time at which this occurs is the end of the systole (T_{es}). The unstressed volume (V_0) is the volume at which end-systolic pressure (P_{es}) is equal to zero. The heart relaxes with exponential decay with a time constant of relaxation (τ). Finally, there are scaling and exponent factors that characterize the end-diastolic pressure-volume relationship (EDPVR) A and B, respectively. Note that each half (right and left) of the heart has its its own values for these

parameters.

The circulation parameters are those related to the vessels and tissues. The characteristic resistance (R_c) is the resistance of the aortic and pulmonar valves, and proximal arteries. The arterial resistance (R_a) is the resistance of the arterioles. The venous resistance (R_v) is the resistance to venous return, and mitral and tricuspid valves. The arterial compliance (C_a) is the compliance of the arteries. The venous compliance (C_v) is the compliance of the veins. Note that each half of the CV circuit (systemic and pulmonic) has its own values for these parameters.

Finally, there are a set of parameters common to multiple components of the circulatory system. These include the heart rate (HR) and the circulatory volumes: total blood volume (V_T); stressed blood volume (V_S), which is related to the volume that determines intravascular pressure; and unstressed blood volume (V_U), which is the volume that remains constant regardless the circulatory pressure. A summary of all the model parameters are presented in Table 3.5, including their symbol as used in this project, their units, and typical values for a healthy man.

Parameters	Symbol	Value		Units
Heart Parameters		**RV**	**LV**	
End-systolic elastance	E_{es}	0.7	3.0	*mmHg/mL*
Unstressed volume	V_0	0	0	*mL*
Time to end systole	T_{es}	0.175	0.175	*s*
Time constant of relaxation	τ	0.025	0.025	*s*
Scaling factor for EDPVR	A	0.35	0.35	*mmHg*
Exponent for EDPVR	B	0.023	0.033	mL^{-1}
Circulation parameters		**Pul**	**Sys**	
Arterial resistance	R_a	0.03	0.90	*mmHg·s/mL*
Characteristic resistance	R_c	0.02	0.03	*mmHg·s/mL*
Venous resistance	R_v	0.015	0.015	*mmHg·s/mL*
Arterial compliance	C_a	13.00	1.32	*mL/mmHg*
Venous compliance	C_v	8.00	70.00	*mL/mmHg*
Common parameters				
Heart rate	HR	75		*bpm*
Total blood volume	V_T	5500		*mL*
Total stressed blood volume	V_S	750		*mL*
Total unstressed blood volume	V_U	4750		*mL*

Table 3.5 Baseline parameters. CV parameters with values appropriate for a healthy 75 kg man. RV and LV stand for right and left ventricle, respectively. Pul and Sys stand for pulmonic and systemic circulation, respectively.

By tuning the values of these parameters, different CV states can be defined, such as a healthy subject at rest or during exercise, a patient with specific CV pathologies, a patient recovering from

surgery, and so on. These parameters can also be modified by drugs. As explained in Section 3.3.2, the reinforcement learning policy varied several of these parameters to simulate physician actions in the treatment of patients.

3.2.2. Model Equations

The equations describing the CV model are explained in this section. In order to denote the pressure or volumen in a specific CV compartment the following subscripts are used:"a" for arteries, "v" for veins, "s" for systemic circulation, "p" for pulmonic circulation, "LV" for left ventricle, and "RV" for right ventricle. Moreover, specific moments in the cardiac cycle are denoted by the subscripts "es" and "ed" which stand for end-systole and end-diastole, respectively.

The ventricular pumping characteristics are represented by a time varying function ($\epsilon(t)$) that relates the ventricular pressure ($P(t)$) with the ventricular volume ($V(t)$). The $\epsilon(t)$ function is modeled as an increasing sine wave during systole and an exponential decay during diastole. Moreover, the ventricular pressure ($P(t)$) is split up into the end-systolic pressure ($P_{es}(t)$) and end-diastolic pressure ($P_{ed}(t)$). The end-systolic pressure ($P_{es}(t)$) is linearly related to the ventricular volume ($V(t)$) by the end-systolic elastance (E_{es}). The end-diastolic pressure ($P_{ed}(t)$) is related to the ventricular volume ($V(t)$) non-linearly with an exponential function.

The equations for the left ventricle are the following:

$$\epsilon_{LV}(t) = \begin{cases} \frac{1}{2}\Big(\sin\Big(\frac{\pi}{T_{es}}t - \frac{\pi}{2}\Big) + 1\Big) & \text{for} \quad t < \frac{3T_{es}}{2} \\ \frac{1}{2}\exp\Big(-\frac{t-3T_{es}/2}{\tau}\Big) & \text{for} \quad t \geq \frac{3T_{es}}{2} \end{cases} \tag{3.1}$$

$$P_{es,LV}(t) = E_{es,LV}(V_{LV}(t) - V_{0,LV}) \tag{3.2}$$

$$P_{ed,LV}(t) = A_{LV}\big(\exp\big(B_{LV}(V_{LV}(t) - V_{0,LV})\big) - 1\big) \tag{3.3}$$

$$P_{LV}(t) = \big(P_{es,LV}(t) - P_{ed,LV}(t)\big)\epsilon_{LV}(t) + P_{ed,LV}(t) \tag{3.4}$$

On the other hand, the equations for the right ventricle are:

$$\epsilon_{RV}(t) = \begin{cases} \frac{1}{2}\Big(\sin\Big(\frac{\pi}{T_{es}}t - \frac{\pi}{2}\Big) + 1\Big) & \text{for} \quad t < \frac{3T_{es}}{2} \\ \frac{1}{2}\exp\Big(-\frac{t-3T_{es}/2}{\tau}\Big) & \text{for} \quad t \geq \frac{3T_{es}}{2} \end{cases} \tag{3.5}$$

$$P_{es,RV}(t) = E_{es,RV}(V_{RV}(t) - V_{0,RV}) \tag{3.6}$$

$$P_{ed,RV}(t) = A_{RV}\big(\exp\big(B_{RV}(V_{RV}(t) - V_{0,RV})\big) - 1\big) \tag{3.7}$$

$$P_{RV}(t) = \big(P_{es,RV}(t) - P_{ed,RV}(t)\big)\epsilon_{RV}(t) + P_{ed,RV}(t) \tag{3.8}$$

The pressure ($P(t)$) and volume ($V(t)$) from the systemic and pulmonic arteries and veins are

related linearly by their respective compliances (C):

$$P_{a,s}(t) = V_{a,s}(t)/C_{a,s} \tag{3.9}$$

$$P_{v,s}(t) = V_{v,s}(t)/C_{v,s} \tag{3.10}$$

$$P_{a,p}(t) = V_{a,p}(t)/C_{a,p} \tag{3.11}$$

$$P_{v,p}(t) = V_{v,p}(t)/C_{v,p} \tag{3.12}$$

Changes in pressure ($P(t)$) and volume ($V(t)$) in the six compartments are described with a set of six differential equations, where flow (derivative of volume) is related linearly to the pressure by the appropriate resistance:

$$\frac{\mathrm{d}V_{LV}(t)}{\mathrm{d}t} = \frac{P_{v,p}(t) - P_{LV}(t)}{R_{v,p}}\alpha_{LV} - \frac{P_{LV}(t) - P_{a,s}(t)}{R_{c,s}}\beta_{LV} \tag{3.13}$$

$$\frac{\mathrm{d}V_{a,s}(t)}{\mathrm{d}t} = \frac{P_{LV}(t) - P_{a,s}(t)}{R_{c,s}}\beta_{LV} - \frac{P_{a,s}(t) - P_{v,s}(t)}{R_{a,s}} \tag{3.14}$$

$$\frac{\mathrm{d}V_{v,s}(t)}{\mathrm{d}t} = \frac{P_{a,s}(t) - P_{v,s}(t)}{R_{a,s}} - \frac{P_{v,s}(t) - P_{RV}(t)}{R_{v,s}}\alpha_{RV} \tag{3.15}$$

$$\frac{\mathrm{d}V_{RV}(t)}{\mathrm{d}t} = \frac{P_{v,s}(t) - P_{RV}(t)}{R_{v,s}}\alpha_{RV} - \frac{P_{RV}(t) - P_{a,p}(t)}{R_{c,p}}\beta_{RV} \tag{3.16}$$

$$\frac{\mathrm{d}V_{a,p}(t)}{\mathrm{d}t} = \frac{P_{RV}(t) - P_{a,p}(t)}{R_{c,p}}\beta_{RV} - \frac{P_{a,p}(t) - P_{v,p}(t)}{R_{a,p}} \tag{3.17}$$

$$\frac{\mathrm{d}V_{v,p}(t)}{\mathrm{d}t} = \frac{P_{a,p}(t) - P_{v,p}(t)}{R_{a,p}} - \frac{P_{v,p}(t) - P_{LV}(t)}{R_{v,p}}\alpha_{LV} \tag{3.18}$$

where α_{LV}, β_{LV}, α_{RV} and β_{RV} are the representation of mitral, aortic, tricuspid and pulmonary valves, respectively. All valves are modeled as ideal valves allowing flow in only one direction:

$$\alpha_{LV} = \begin{cases} 1 & \text{if} \quad P_{LV}(t) < P_{v,p} \\ 0 & \text{otherwise} \end{cases} \tag{3.19}$$

$$\beta_{LV} = \begin{cases} 1 & \text{if} \quad P_{LV}(t) > P_{a,s} \\ 0 & \text{otherwise} \end{cases} \tag{3.20}$$

$$\alpha_{RV} = \begin{cases} 1 & \text{if} \quad P_{RV}(t) < P_{v,s} \\ 0 & \text{otherwise} \end{cases} \tag{3.21}$$

$$\beta_{RV} = \begin{cases} 1 & \text{if} \quad P_{RV}(t) > P_{a,p} \\ 0 & \text{otherwise} \end{cases} \tag{3.22}$$

Finally, the sum of the volume ($V(t)$) in all six compartments remains constant over time, and it is equal to the stressed volume (V_s):

$$V_s = V_{LV}(t) + V_{a,s}(t) + V_{v,s}(t) + V_{RV}(t) + V_{a,p}(t) + V_{v,p}(t) \tag{3.23}$$

3.2.3. Model Implementation and Assessment

The whole CV model was implemented as a *Python* class in order to be modular and easily embeddable in the reinforcement learning framework. Differential equations were discretized and solved using the Euler's method. For example, for any volume ($V(t)$):

$$\frac{\mathrm{d}V(t)}{\mathrm{d}t} = Q(t) \rightarrow \frac{V_{k+1} - V_k}{t_k} = Q_k \rightarrow V_{k+1} = V_k + t_k \cdot Q_k \tag{3.24}$$

where t_k is the time step, which was chosen to be adaptive to the heart rate (HR). This time step adaptation was implemented because it was desired to simulate as fast as possible, while avoiding integration step instabilities. Since HR determines the speed of the cycle, and thus, the speed of the evolution of the dynamic variables ($P(t)$ and $V(t)$), adapting the time step to this parameter was suitable. Time step adaptation corresponded to HR ranges:

$$t_k[ms] = \begin{cases} 8 & \text{for} \quad HR < 75 \\ 5 & \text{for} \quad 75 \leq HR < 140 \\ 3 & \text{for} \quad 140 \leq HR < 180 \\ 1 & \text{for} \quad 180 \leq HR < 200 \\ 0.5 & \text{for} \quad HR \geq 200 \end{cases} \tag{3.25}$$

In order to assess the correctness of the implementation and debug possible errors, the *Harvi* simulator was used. *Harvi* is a more complex CV model also developed by Burkhoff, which is based on the model described in this thesis. Available to the lab team, its use helped determine the correctness of the model by comparing the evolution of pressures and volumes with baseline parameters.

In the Results chapter (Section 4.1) PV loops from a healthy and unhealthy subject are presented and explained. Furthermore, a qualitative analysis comparing the arterial pressure obtained in simulation and from a real patient is performed. Advantages, similarities, and limitations of this CV model are discussed.

3.3. Reinforcement Learning Framework

The desired policy that performs therapy decisions on simulated patients was obtained by reinforcement learning (RL) training. RL was chosen over any other control techinque as it was considered suitable for the cardiogenic shock problem. The decision making nature of tailored therapy, together with the need to build a controller that is adaptive, learns from historical and data, and can handle both continuous and discrete actions, lead to consider RL appropriated for this problem. Therefore, a RL framework was developed to implement the deep Q-Network (DQN) algorithm that using the presented CV model as simulator, was able to train policies in different conditions. This section is meant to explain all the methodology related to the RL framework developed in this project.

3.3.1. Deep Q-Network Algorithm

The DQN algorithm [21] is a model-free RL algorithm that tries to find the policy that maximizes the total return (see Equation 2.9). It works with continuous state spaces ($\mathcal{S}$) and discrete action spaces ($\mathcal{A}$) where action-triggered state transitions can be stochastic. In order to maximize the total return and find the optimal policy, DQN optimizes the action-value or Q-function ($Q(s,a)$). This function determines the expected return achievable by following a policy from given state ($s \in \mathcal{S}$) and then taking some action ($a \in \mathcal{A}$). Computing the optimal Q-function ($Q^*(s,a)$) can be challenging, thus the *Bellman equation* is commonly employed to optimize it recursively:

$$Q_{i+1}(s_t, a_t) = Q_i(s_t, a_t) + \alpha \cdot (r_t + \gamma \cdot \max_{a \in \mathcal{A}} Q_i(s_{t+1}, a) - Q_i(s_t, a_t)) \tag{3.26}$$

where $Q_i(s_t, a_t)$ is the current Q-function, $\alpha \in (0, 1]$ is the learning rate, $\gamma \in [0, 1)$ is the discount factor, and $\max_a Q_i(s_{t+1}, a)$ is the estimate of optimal future value. This value iteration algorithm converges to the optimal Q-function if enough iterations are performed: $Q_i(s,a) \rightarrow Q^*(s,a)$ if $i \rightarrow \infty$, and the optimal policy can be computed from the optimal Q-function as:

$$\pi^*(s,a) = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\ Q^*(s,a) \tag{3.27}$$

In DQN, the Q-function is approximated with a neural network (NN) ($Q(s,a) \rightarrow Q(s,a,\omega)$) where ω are the weights of the NN. To train this NN and find the optimal Q-function, DQN takes advantage of the *Bellman equation* convergence, thereby, with enough iterations $Q_{i+1}(s_t, a_t) = Q_i(s_t, a_t)$ and Equation 3.26 becomes:

$$r_t + \gamma \cdot \max_{a \in \mathcal{A}} Q_i(s_{t+1}, a) - Q_i(s_t, a_t) = 0 \rightarrow Q_i(s_t, a_t) = r_t + \gamma \cdot \max_{a \in \mathcal{A}} Q_i(s_{t+1}, a) \tag{3.28}$$

The equality from Equation 3.28 can be used to define a loss function for NN training:

$$\mathcal{L} = E[(\hat{y} - Q(s, a, \theta))^2] \tag{3.29}$$

where $\hat{y} = r + \gamma \max_a \hat{Q}(s', a, \theta)$ are the labels obtained from the target Q-function ($\hat{Q}(s', a, \theta)$). The target Q-function ($\hat{Q}(s', a, \theta)$) is a copy of the trained Q-function ($Q(s', a, \theta)$) updated at periodic increment steps of the algorithm. Note that this implies that the training labels are changing over time as $Q(s, a)$ evolves. The purpose of this target Q-function is to hold the training labels constant for a tunable number of iterations to help the algorithm converge. Finally, the data used to train the Q-function are obtained from a technique called *experience replay*. This technique uses the agent's experiences at each time step as data for training, which include: the current state and action, the reward obtained and the next state ($e_t = (s_t, a_t, r_t, s_{t+1})$).

Algorithm 1: Deep Q-learning with Experience Replay

Initialize replay memory $\mathcal{D}$ to capacity N;
Initialize Q-function $Q(s, a, \omega)$ with random weights;
Initialize target Q-function $\hat{Q}(s, a, \omega)$ with same weights as $Q(s, a, \omega)$;
for *episode = 1, M* **do**
  Initialize environment with states s_0;
  for *t = 1, T* **do**
    With probability ϵ select a random action a_t
    otherwise select $a_t = \max_{a \in \mathcal{A}} Q(s_t, a, \omega)$;
    Execute action a_t and observe reward r_t and next state s_{t+1};
    Store experience $e_t = (s_t, a_t, r_t, s_{t+1})$ in $\mathcal{D}$;
    Sample random batch of experiences $e_1, \ldots, e_J$ from $\mathcal{D}$;
    For each experience e_j set:
    $$\hat{y}_j = \begin{cases} r_j, & \text{for terminal } s_{j+1} \\ r_j + \gamma \cdot \max_a \hat{Q}(s_{j+1}, a, \omega) & \text{for non-terminal } s_{j+1} \end{cases}$$
    Perform a gradient descent step on $E(\hat{y}_j - Q(\phi_j, a_j, \omega))^2, \quad j = 1, \ldots, J$;
    Every C steps, copy $\hat{Q}(s, a, \omega)$ with same weights as $Q(s, a, \omega)$;
  end
end

Combining the pieces of the DQN results in the full algorithm schematized in Algorithm 1. First, both trained and target Q-function are initialized with random weights, as well as a buffer (replay memory $\mathcal{D}$) with capacity N. Then, M episodes are performed. Note that an episode consists of an agent's entire interaction with the environment, from initial to final interaction step. For each episode, the environment is initialized and the agent starts interacting with it. Action selection is carried out according to an ϵ-greedy policy, in which a random action is taken with probability ϵ and policy action ($a_t = \max_{a \in \mathcal{A}} Q(s_t, a, \omega)$) is taken with probability $1 - \epsilon$. At each environment step, the whole experience ($e_t = (s_t, a_t, r_t, s_{t+1})$) is stored in the replay memory ($\mathcal{D}$). Finally, a batch of experiences are taken and the policy is updated performing a gradient descent step according to the loss function in Equation 3.29. This is done until reaching the horizon T, when this process is

repeated for the next episode. Moreover, every C steps of the algorithm $\hat{Q}(s, a, \omega)$ is updated with the same weights as $Q(s, a, \omega)$.

3.3.2. Reinforcement Learning for Cardiogenic Shock

In this project the RL framework was adapted to the cardiogenic shock (CS) problem with the aim of computing policies that will restore cardiac output, a central function of the heart. The Markov Decision Process (MDP) was determined in terms of the cardiovascular (CV) model defined in Section 3.2 and actions that would emulate the therapeutic decisions a clinician makes to treat CS, often referred to as tailored therapy (see Section 2.1.3). The elements of an MDP, namely the state space ($\mathcal{S}$), the action space ($\mathcal{A}$) and the instantaneous reward function ($\mathcal{R}_a$) are specified below. Note that the fourth element, the transition function ($\mathcal{P}_a$) is the cardiovascular model itself.

State space

The state space included all pressures ($P(t)$) and volumes ($V(t)$) from the six compartments plus the cardiac output (CO). Concretely, cardiac output was computed with Equation 2.1, and the mean of each pressure and volume was taken over one cycle:

$$\bar{X} = \frac{1}{L_{cycle}} \sum_{k=1}^{L_{cycle}} X_k \tag{3.30}$$

being L_{cycle} the length of the cardiac cycle in terms of samples, and X any of the pressures and volumes of the six compartments: $X \in \{V_{LV}, V_{a,s}, V_{v,s}, V_{RV}, V_{a,p}, V_{v,p}, P_{LV}, P_{a,s}, P_{v,s}, P_{RV}, P_{a,p}, P_{v,p}\}$.

Hence, the state space contained 13 different continuous states (cardiac output plus pressures and volumes for the 6 compartments), becoming a 13-dimensional continuous state space ($\mathcal{S} \in \mathbb{R}^{13}$).

Action space

The action space was defined to mimic tailored therapy in a simplified way. Rather than reproducing the actions of specific medications given to patients, the actions here were modeled as direct changes to a subset of the CV model parameters. Drugs and other therapies (fluid administration, blood transfusion) also modify the physiology but often in complex ways, for example by altering, in effect, multiple parameters simultaneously. What is more, drug effects can be highly specific to an individual and complex to model, so they were considered beyond the scope of this project.

Given that some of the most important targets of tailored therapy include the loading conditions and the inotropic state of the heart (as explained in Section 2.1.3), the parameters chosen to be varied by the policy actions were the end-systolic elastance (E_{es}), the heart rate (HR) and the stressed blood volume (V_s). Changes to those parameters would simulate the effect of vasoactive medications used in tailored therapy, such as norepinephrine [4], dobutamine [25], intravenous fluids, and

diuretics [16]. Moreover, the policy could choose whether these parameters should be increased, decreased or remain constant. The entire action space therefore, consists of 27 possible discrete actions ($\mathcal{A} \in \{0, \ldots, 26\}$). Table 3.6 shows all combinations of parameter changes that lead to the 27-action space.

Action	V_s	HR	E_{es}	Action	V_s	HR	E_{es}
0	*Constant*	*Constant*	*Constant*	14	*Decrease*	*Decrease*	*Increase*
1	*Constant*	*Constant*	*Decrease*	15	*Decrease*	*Increase*	*Constant*
2	*Constant*	*Constant*	*Increase*	16	*Decrease*	*Increase*	*Decrease*
3	*Constant*	*Decrease*	*Constant*	17	*Decrease*	*Increase*	*Increase*
4	*Constant*	*Decrease*	*Decrease*	18	*Increase*	*Constant*	*Constant*
5	*Constant*	*Decrease*	*Increase*	19	*Increase*	*Constant*	*Decrease*
6	*Constant*	*Increase*	*Constant*	20	*Increase*	*Constant*	*Increase*
7	*Constant*	*Increase*	*Decrease*	21	*Increase*	*Decrease*	*Constant*
8	*Constant*	*Increase*	*Increase*	22	*Increase*	*Decrease*	*Decrease*
9	*Decrease*	*Constant*	*Constant*	23	*Increase*	*Decrease*	*Increase*
10	*Decrease*	*Constant*	*Decrease*	24	*Increase*	*Increase*	*Constant*
11	*Decrease*	*Constant*	*Increase*	25	*Increase*	*Increase*	*Decrease*
12	*Decrease*	*Decrease*	*Constant*	26	*Increase*	*Increase*	*Increase*
13	*Decrease*	*Decrease*	*Decrease*				

Table 3.6 Action space of 27 possible discrete actions as combinations of increasing, decreasing or remaining constant to the stressed volume (V_s), heart rate (HR) and end-systolic elastance (E_{es}) parameters.

Regarding the numerical degree of change elicited by each action, stochastic variation was introduced, following a Gaussian distribution ($\mathcal{N}(\mu, \sigma^2)$). To make this variation as realistic as possible, the change was based on the study in [25], which reported a statistical analysis of the impact of the drug dobutamine on heart rate. For the end-systolic elastance and stressed volume, similar variation in the effect size was employed. Specifically, the parameter change induced by an action were defined as follows:

$$\begin{cases} |\Delta V_s| = \mathcal{N}(30.0, 2.675^2)\, mL \\ |\Delta HR| = \mathcal{N}(4.0, 0.357^2)\, bpm \\ |\Delta E_{es}| = \mathcal{N}(0.15, 0.0134^2)\, mmHg/mL \end{cases} \tag{3.31}$$

Note that the effect size is expressed as an absolute value, since a policy action can either increase or decrease the parameter value.

Reward function

The instantaneous reward function was implemented as a shaped function to resemble how physicians evaluate their goals. The most common approach is to track the deviation of a physiological parameter from a goal value and then act accordingly. The reward included both a positive reward term as well as penalty terms.

The positive reward ($\mathcal{R}^{+}(s,a)$) tracked the cardiac output since the main objective was to restore it to a normal value. To encourage the policy to achieve this, a sigmoid-like function that gives reward of $+1$ when cardiac output is higher than $5.2L/min$ and 0 when it is below $3.8L/min$ was implemented. The shape of such a function can be appreciated in Figure 3.5.

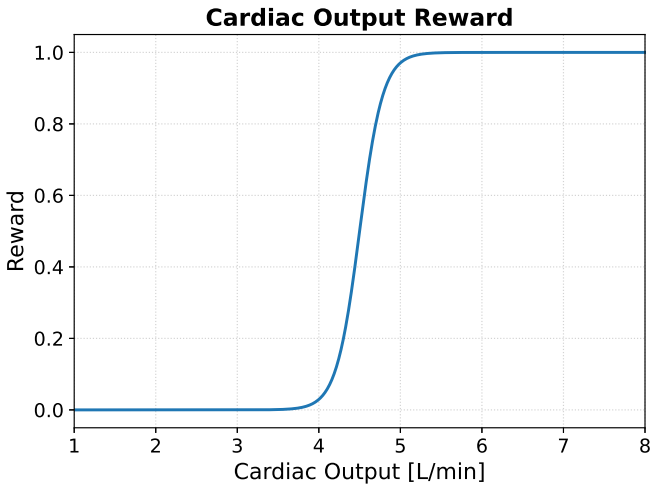
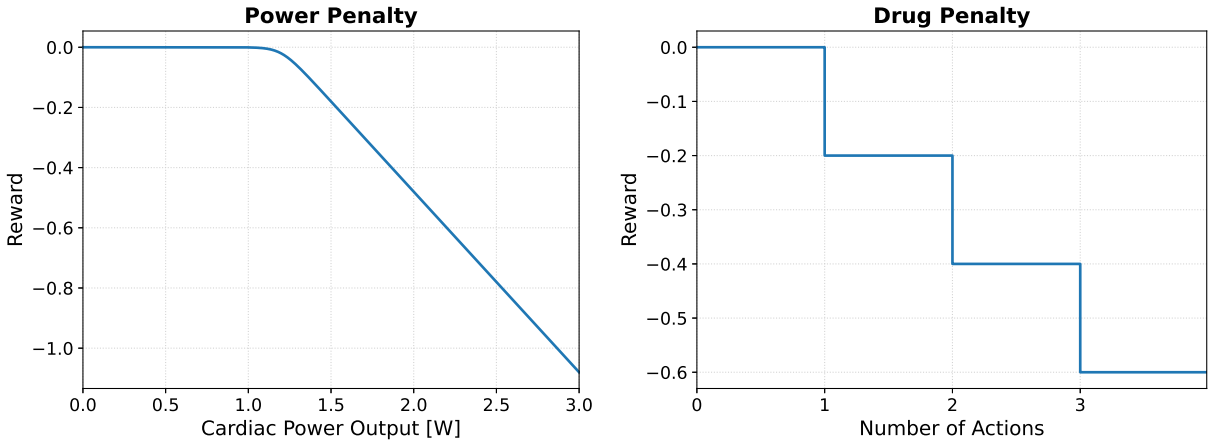


Figure 3.5 Shape of the reward function related to cardiac output.

Conversely, two penalty terms were included to restrict policy behavior in certain ways. The first of these was a "drug penalty" ($\mathcal{R}^{-}_{dp}(s,a)$) for those actions that changed a parameter value. A value of 0.2, 0.4 and 0.6 was subtracted from the reward every time an action was chosen that modified one, two or three of the CV model parameters, respectively. This penalty was meant to encourage the policy to use as few drug actions as possible. The second penalty term was for states of high cardiac power. This power penalty term ($\mathcal{R}^{-}_{pp}(s,a)$) considered the cardiac power output ($CPO = \bar{P}_{a,s} \times CO/451$), which is a measure of how much power the heart is consuming. It was shaped as a ReLu-like function that started to subtract reward, proportional to the CPO, above $1.1W$. It is well appreciated clinically that operating the heart at high power outputs is detrimental to myocardial recovery and is often unsustainable. This penalty therefore encouraged the policy to avoid high power outputs when choosing actions to augment cardiac output. Figure 3.6 represents both penalty functions.



(a) **(b)**

Figure 3.6 Shape of penalty terms. a) Drug action penalty term ($\mathcal{R}^-_{dp}(s,a)$). b) Power penalty term ($\mathcal{R}^-_{pp}(s,a)$).

To sum up, the resulting instantaneous reward was a sum of the positive reward plus a combination of the penalties explained above:

$$\mathcal{R}(s,a) = \mathcal{R}^+(s,a) + \mathcal{R}^-_i(s,a) \tag{3.32}$$

where $\mathcal{R}^+(s,a)$ stands for the positive reward and $\mathcal{R}^-_i(s,a)$ for any combination of the penalties introduced in the previous paragraph ($i \in \{pp, dp, dp+pp\}$).

To complete the MDP, terminal states were included that would end an episode before reaching the simulation horizon (T). These states were added to simulate patient mortality in the event that a bad strategy was followed by the policy. Specifically, the terminal state occurred if one of the following criteria were met:

- The cardiac output (CO) was lower than 2 L/min.
- The stressed volume (V_s) was below 300 mL or above 2500 mL.
- The heart rate (HR) was lower than 20 bpm or higher than 250 bpm.
- The end-systolic elastance (E_{es}) was lower than 0.1 $mmHg/mL$ and higher than 10 $mmHg/mL$.

3.3.3. Policy Transfer to Real World

An important challenge that a policy must face is transfer to the real world. Simplifications and deviations that the simulator may have from reality could lead to catastrophic policy behaviors, and likely failing to perform the desired task. Healthcare applications carry additional challenges, as real patient data tends to be limited relative to the training needs of RL, and training policies on real patients is not feasible or ethical. These challenges are commonly known as the *sim2real*

gap, and exist due to inconsistencies between the physical world and the model. In order to close or at least narrow this gap, there are strategies that can be performed in simulation [32].

A first strategy is to develop a system identification tool, which can improve the physical model and make the simulator more realistic. Another strategy is domain randomization (DR), an approach that trains a policy using a whole set of simulated environments, each with different properties. In the CS application, this would be analogous to training a policy on a range of patients, each with a unique set of parameters that determine the properties of their CV system. Training a policy that works across such a spectrum of environments increases the likelihood it will perform well in the real world environment. This is plausible because the real system may be close to a subset of those simulated environments, or because training a single policy in multiple environments increases its robustness.

The simplest strategy for randomizing the domain is to use uniform sampling. Each randomized parameter (ξ_i) is bounded by an interval ($\xi_i \in [\xi_i^{low}, \xi_i^{high}]$ $i = 1, \ldots, N$) where N are the total number of randomization parameters. During policy training, each parameter is uniformly sampled within the range.

In the present thesis, uniform DR was used to train the policy. Parameter ranges were found by employing a system identification tool on the CABG cohort (described in Section 3.1.3). The parameters randomized and the strategy followed to identify their ranges are explained in detail in Section 3.4.

3.3.4. Reinforcement Learning Implementation and Assessment

The whole RL framework was implemented in *Python*. The *Gym* environment library was utilized to define the MDP, and combine the CV simulator with the DQN + experience replay algorithm. The policy was modeled with a neural network (NN). Both modeling and NN training were performed with *TensorFlow* (TF). The ϵ-greedy action selection strategy was defined with an exponential decay, decreasing the ϵ value as training moved forward. The exponential function contained a decay rate (δ) and maximum/minimum values ($\epsilon_{max}, \epsilon_{min}$). As a result, policy training explored using a greater fraction of random actions at the beginning of training, and it exploited the best policy actions at the end of training.

Regarding the simulation of CS patients, this was achieved by reducing the end-systolic elastance (E_{es}) and/or increasing the resistance of the systemic veins ($R_{v,s}$) in the CV simulator. Changing those parameters led to a very low cardiac output, where the policy had to learn how to restore it at every episode of the training process. The goal was to restore the cardiac output to, at least, $5.0 L/min$.

The RL training workflow was set up to perform the following tasks at each step of an episode: take an action according to ϵ-greedy strategy; apply the action to the CV simulator by changing the appropriate parameters; simulate with the new set of parameters until convergence; compute the 13 states, the reward and evaluate terminal state criteria; store experience $e_t = (s_t, a_t, r_t, s_{t+1})$; get batch of experiences and update policy. Figure 3.7 schematizes this workflow.

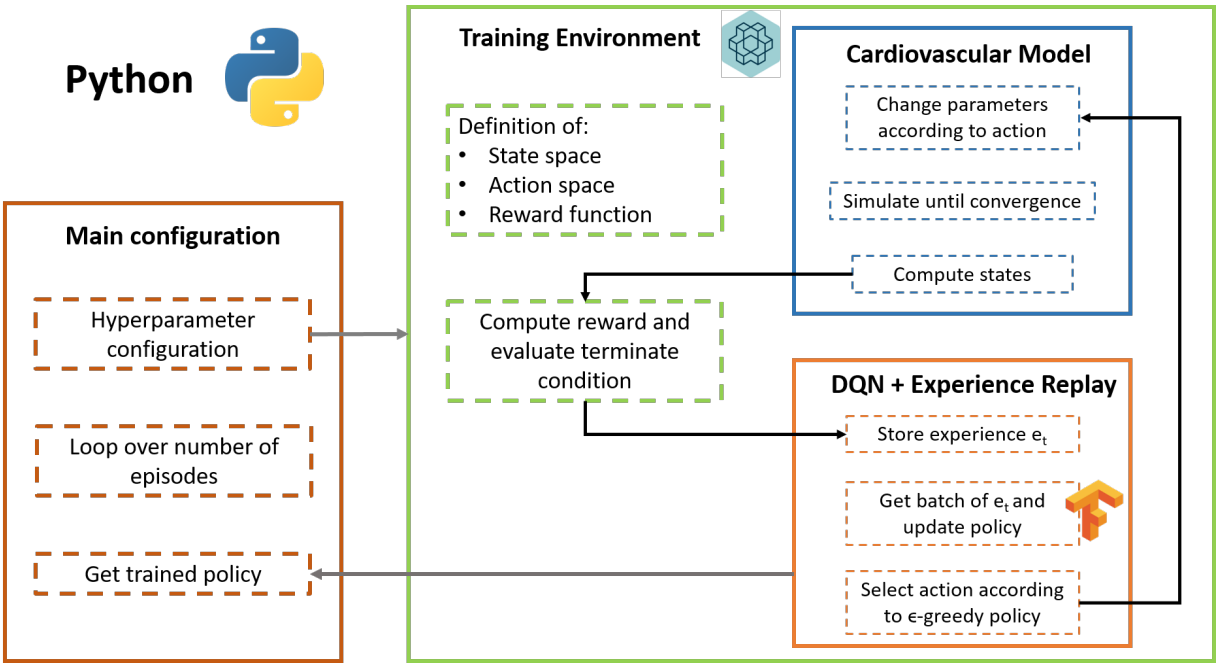


Figure 3.7 Schematic workflow of the RL framework to train policies in simulated CS patients.

The RL framework contained several hyperparameters that configured the training environment, the DQN algorithm and the NN to approximate the Q-function. The set of hyperparameters included: the total number of episodes (M); the simulation horizon (T); the capacity (N) of the replay memory ($\mathcal{D}$); the amount of steps where the target Q-function ($\hat{Q}(s', a, \theta)$) is copied (C); the discount factor (γ); the maximum and minimum epsilon ($\epsilon_{max}, \epsilon_{min}$) of the ϵ-greedy strategy; the decay rate (δ) of the ϵ-greedy strategy; the batch size of experiences (e_t); the learning rate (α) for NN training; the NN activation function; the NN hidden architecture (number of hidden layers and neurons per layer); and the parameters (β_1 and β_2) of the NN optimizer.

The majority of hyperparameters were optimized in order to obtain the best training results. However, some were already preset. Concretely, the number of episodes (M) to ensure policy convergence; the simulation horizon (T) so the policy had enough time to treat the patient; and finally, the maximum and minimum epsilon values (ϵ_{max}, ϵ_{min}), balancing the ϵ-greedy strategy during all the training process.

The hyperparameter optimization was performed through Bayesian Optimization using Gaussian Processes as function approximators [5]. *SkOpt* library was used for this optimization task. More-

over, the process was parallelized over all computer resources with the help of *Ray* library, as this optimization is quite time consuming. The optimization was performed by maximizing the accumulated reward ($\sum_{t=0}^{T} \mathcal{R}(s,a)$) averaged over the last 50 episodes. A list of all the hyperparameters with their final value is presented in Table 3.7.

Episodes (M)	Horizon (T)	Memory Capacity (N)	Sync Steps (C)	Discount Factor (γ)	Max Eps (ϵ_{max})	Min Eps (ϵ_{min})	Decay Rate (δ)	Batch Size	Learning Rate (α)	Activation Function	Hidden Architecture	Optimizer Parameters
350	150	57000	670	0.83	0.95	0.05	0.0001	450	0.0001	ELU	16 - 32	0.81; 0.97

Table 3.7 RL hyperparameters with optimized values through Bayesian Optimization.

Several analyses were performed to assess the results of the presented implementation (Section 4.3). These were grouped into the following sets:

1. **Assessment of DR robustness**. Trained policies with and without DR were compared to evaluate the importance to include DR during the training phase of the policy.
2. **Comparison of restricted policies.** Three policies with restricted actions (one without actions that change V_s, another without actions that change HR. and another without actions that change E_{es}) were trained and compared in order to assess their limitations to reverse the decompensated state of CS patients, and thus, find out what are the most important actions.
3. **Importance of penalty terms.** Three policies with different reward functions were analyzed to evaluate the value of the penalty terms explained above. To do so, a policy with only the cardiac output reward ($\mathcal{R}(s,a) = \mathcal{R}^{+}(s,a)$), a policy with both cardiac output and power penalty term reward ($\mathcal{R}(s,a) = \mathcal{R}^{+}(s,a) + \mathcal{R}^{-}_{pp}(s,a)$), and a policy with cardiac output, power and drug penalty term reward ($\mathcal{R}(s,a) = \mathcal{R}^{+}(s,a) + \mathcal{R}^{-}_{pp+dp}(s,a)$) were trained and compared.
4. **Final policy assessment**. As a final evaluation, the policy that performed the best on the previous analyses was tested in a more complex environment. A CS patient was not only simulated with initial low cardiac output, but also disturbances while the policy was treating the patient were included.

In all cases, sample efficiency during training was compared by counting how many episodes were needed for convergence, as well as the maximum accumulated reward achieved. Also, performance of the policy was assessed by determining how many policy steps (or actions) it needed to restore the cardiac output, as well as its behavior once it reached the goal of $5.0 L/min$. Assessments included the analysis of cardiac output (CO), mean arterial pressure ($\bar{P}_{a,s}$), end-systolic elastance (E_{es}), stressed volume (V_s) and heart rate (HR) evolution, as well as the actions taken by the policy at each step. In some cases, it has not been necessary to assess all these variables to discuss the results, and figures of such variables have been included in the Appendix, Section A.2.

3.4. System Identification for Parameter Estimation

In order to reduce the *sim2real* gap with domain randomization, realistic parameter ranges derived from patients must be found. To identify these ranges, a system identification tool was developed for the purpose of estimating a subset of the Burkhoff CV model parameters (Section 3.2.1), employing the data from the CABG cohort (Section 3.1.3). Concretely, the subset of parameters that were estimated included: the left and right end-systolic elastances ($E_{es_{LV}}$ and $E_{es_{RV}}$); the systemic and pulmonic arterial resistances ($R_{a,s}$ and $R_{a,p}$); the four compliances ($C_{a,s}$, $C_{v,s}$, $C_{a,p}$ and $C_{v,p}$); and finally, the stressed volume (V_s). Nine parameters in all.

As stated in [23], the Burkhoff and Tyberg CV model is structural identifiable for the aforementioned subset of parameters if the following physiological variables are known: the systemic arterial pressure ($P_{a,s}(t)$), the pulmonary artery pressure ($P_{a,p}(t)$), the stroke volume (SV) and the heart rate (HR). This fact justifies the selection of signals in the final dataset (Section 3.1.4). Furthermore, the central venous pressure ($P_{v,s}(t)$) was also included as it was available on the patient's recorded signals. It was thought that including this last signal could add valuable information for the parameter estimation.

3.4.1. Modification of Autoencoder for System Identification

System identification was framed as a supervised learning problem. A deep multilayer perceptron (MLP) was trained to estimate the parameters from the signals on the dataset. However, it should be noted that no labels were available, as these CV parameters are not estimated in the hospital (if they were, the identification problem would have been already solved). Therefore, a strategy based on a modified autoencoder was implemented to train the deep MLP.

Briefly, an autoencoder is a specific architecture of neural network (NN) based on an encoder, decoder and a latent space. Usually, the aim of an autoencoder is to learn a representation (latent space) of a set of data by encoding it. It is trained by introducing the same information as input and labels, thus the autoencoder learns to encode the data and then to decode it. In Figure 3.8 a representation of an autoencoder is shown.

In the approach taken in this project, an autoencoder was trained to estimate the CV parameters by forcing the latent space to be the specific space of nine parameters described above. This specification could be achieved by training the decoder and the encoder separately. First, the decoder was trained with simulated data to mimic the CV simulator – given the nine parameters it would produce a cardiac cycle's worth of state data. It should be noted that ideally, the simulator could have been used as decoder. However, due to the parallelization needs over a GPU and the backpropagation algorithm for NN training, this other approach was followed.

With the decoder able to generate state data from parameters, the encoder could be trained to

generate parameters using real patient data as input. In this framework only the encoder was trained with real data. It was assumed that the decoder, which reproduces the CV simulator output, would be capable of generating state data (e.g. intracardiac and intravascular pressures and volumes) that are sufficiently similar to real patient data, provided that the encoder converted these data to a latent space close to the nine CV parameters. Thus, the trained encoder in this setup can be seen as a system identificatation algorithm for estimating the subset of CV parameters.

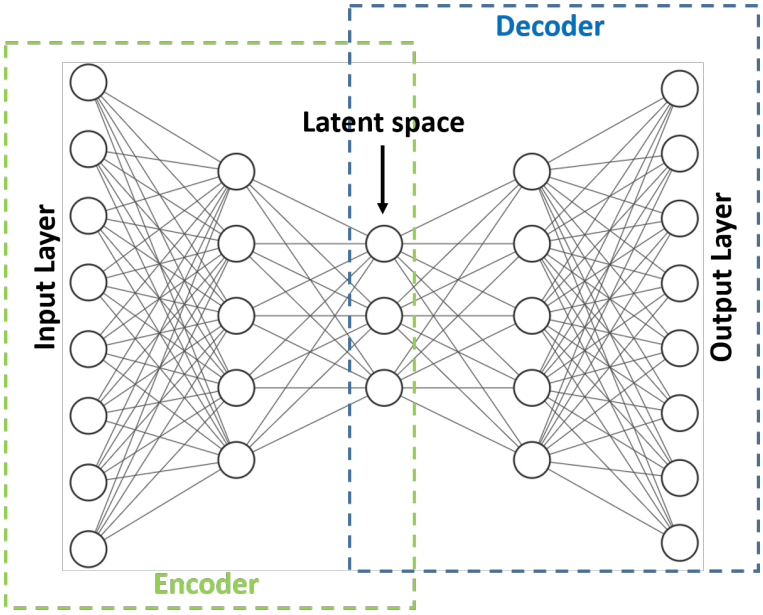


Figure 3.8 Schematic representation of an autoencoder architecture.

3.4.2. System Identification Implementation and Assessment

Both the architecture and the training of the autoencoder were performed with *TensorFlow* (TF) library. In order to train the decoder with simulated data, 0.7 million cardiac cycles that contained information about the arterial pressure ($P_{a,s}(t)$), the central venous pressure ($P_{v,s}(t)$) the pulmonary artery pressure ($P_{a,p}(t)$), the stroke volume (SV) and heart rate (HR) were generated using the simulator. Moreover, the corresponding CV parameters employed to generate these data were stored for the training process.

The input vector of the decoder was the subset of nine CV parameters plus the heart rate:

$$\mathbf{x_{dec}} = [E_{es_{LV}}, E_{es_{RV}}, R_{a,p}, R_{a,s}, C_{a,s}, C_{v,s}, C_{a,p}, C_{v,p}, V_s, HR]^T \in \mathbb{R}^{10} \tag{3.33}$$

The output vector consisted of the three pressures resampled at 50 samples per cycle plus the stroke volume repeated 10 times:

$$\mathbf{y_{dec}} = [\mathbf{P_{a,s}}, \mathbf{P_{v,s}}, \mathbf{P_{a,p}}, \mathbf{SV}]^T \in \mathbb{R}^{160} \tag{3.34}$$

The loss function was the mean squared error between the output vector and the data (simulated pressures and stroke volume) $\hat{\mathbf{y}}_{\mathbf{sim}}$:

$$\mathcal{L} = MSE(\mathbf{y_{dec}}, \hat{\mathbf{y}}_{\mathbf{sim}}) \tag{3.35}$$

In contrast to the decoder, the encoder was trained with both simulated data and the patients dataset. Recalling that the dataset contained the mean value of the cardiac output and heart rate, but not the stroke volume, this last was computed using Equation 2.1. The input vector of the encoder was the three pressures resampled at 50 samples per cycle, the stroke volume repeated 10 times and the heart rate repeated also 10 times:

$$\mathbf{x_{enc}} = [\mathbf{P_{a,s}}, \mathbf{P_{v,s}}, \mathbf{P_{a,p}}, \mathbf{SV}, \mathbf{HR}]^T \in \mathbb{R}^{170} \tag{3.36}$$

The output vector consisted of the subset of nine CV parameters:

$$\mathbf{y_{enc}} = [E_{es_{LV}}, E_{es_{RV}}, R_{a,p}, R_{a,s}, C_{a,s}, C_{v,s}, C_{a,p}, C_{v,p}, V_s]^T \in \mathbb{R}^9 \tag{3.37}$$

Given that the encoder was trained within the whole autoencoder architecture, the loss function used for training was the mean squared error between the decoder's output (state) vector and the patient's state data (pressures and stroke volume) $\hat{\mathbf{y}}_{\mathbf{real}}$:

$$\mathcal{L} = MSE(\mathbf{y_{dec}}, \hat{\mathbf{y}}_{\mathbf{real}}) \tag{3.38}$$

Once the encoder was trained, the entire CABG dataset was passed to it again to find the corresponding latent space vector, i.e, estimates of the nine CV parameters across the whole patient dataset. Figure 3.9 schematizes the entire process.

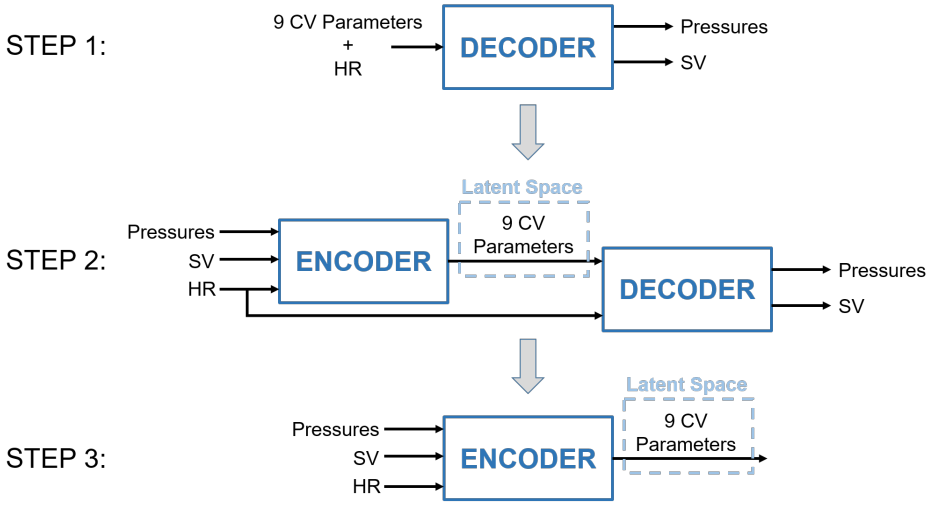


Figure 3.9 Process followed to obtain the system identification tool. Step 1: train the decoder with simulated data. Step 2: train the encoder with the whole autoencoder architecture using both simulated and real patient data. Step 3: estimate the 9 CV parameters from the CABG dataset.

Note that in order to improve the training process for both encoder and decoder, the data were normalized. Parameter vectors ($\mathbf{x_{dec}}$ and $\mathbf{y_{enc}}$) were normalized so that the min and max values fell in the range of $[-1, 1]$. Cardiac cycle vectors ($\mathbf{y_{dec}}$ and $\mathbf{x_{enc}}$) were standardized to have zero mean and unit standard deviation. Values for doing such normalization processes were found from the simulated data (Table A.1 in the Appendix (Section A.1) shows these values). Moreover, dropout, L_2 regularization and the early stopping technique were applied to improve encoder and decoder generalization. Note that early stopping is a method that determines the number of epochs to train the NN. It does so by finding when the validation loss starts to increase, a sign that the model starts to overfit.

Just as in the RL framework, hyperparameters of the encoder and decoder were optimized with Bayesian Optimization (again employing *SkOpt* and *Ray* libraries). In this case, the optimization was performed by maximizing the accuracy of the validation set. Concretely, data was split into a training set of 75% and a validation set of 25%. A five fold cross validation process was performed and the resulting validation accuracy was the score to maximize. Note that the accuracy was measured with the cosine similarity between the output vector and labels, as the NN generates a vector of data:

$$acc = \cos(\theta) = \frac{\mathbf{y_{dec}} \cdot \hat{\mathbf{y}}_{\mathbf{real}}}{\|\mathbf{y_{dec}}\| \|\hat{\mathbf{y}}_{\mathbf{real}}\|} \tag{3.39}$$

A list of all the hyperparameters with their final value is presented in Table 3.8. Moreover, an scheme of the architecture used for both encoder and decoder is shown in Figure 3.10.

	Epochs (E)	Batch Size	Learning Rate (α)	Regularization (L2)	DropOut	Activation Function
Encoder	40	512	0.0001	0.001	0.05	ELU
Decoder	12	512	0.0003	0.001	0.1	ELU

Table 3.8 Autoencoder hyperparameters with optimized values through Bayesian Optimization.

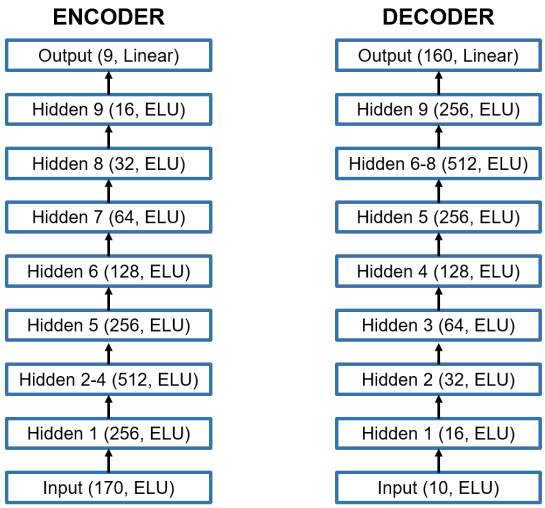


Figure 3.10 NN architectures of both encoder and decoder. Each box represents a layer. Between parenthesis are presented the number of neurons and the activation function.

In order to assess the autoencoder implementation and get the estimation of the nine CV parameters, diverse stages were defined and evaluated separately. First, the decoder training was apprised with its training curves and comparing the waves obtained with baseline parameters (Table 3.5). Pressure waves and stroke volume were compared with the ones obtained using the simulator. The root mean square error (RMSE) was used to get quantitative results. Then, the encoder training was assessed also with its training curves. Again, baseline parameters were compared to the ones obtained by the encoder, when introducing simulated waves. The absolute relative error was obtained to quantitatively describe differences. Finally, the encoder was employed to estimate the distribution of the nine CV parameters using CABG data. Mean and standard deviation of the distribution of each parameter were computed and compared to literature parameters. The estimated mean and standard deviation were also used to define the ranges of the domain randomization technique.

4. RESULTS AND DISCUSSION

Results from this project are presented and discussed in this chapter. First, results related to the implementation of the Burkhoff and Tyberg cardiovascular (CV) model are presented. Second, the training of the system identification tool and the set of CV parameters it produces is shown. Finally, the results of reinforcement learning (RL) for treating cardiogenic shock are described, including both policy training and performance.

4.1. Cardiovascular Simulation with Burkhoff and Tyberg Model

In this section the CV model implemented as a simulator is presented. Simulations from a typical healthy and a typical heart failure patient are described and compared. Next, limitations of the model are pointed out by comparing a simulated arterial pressure wave to a real waveform from a CABG patient.

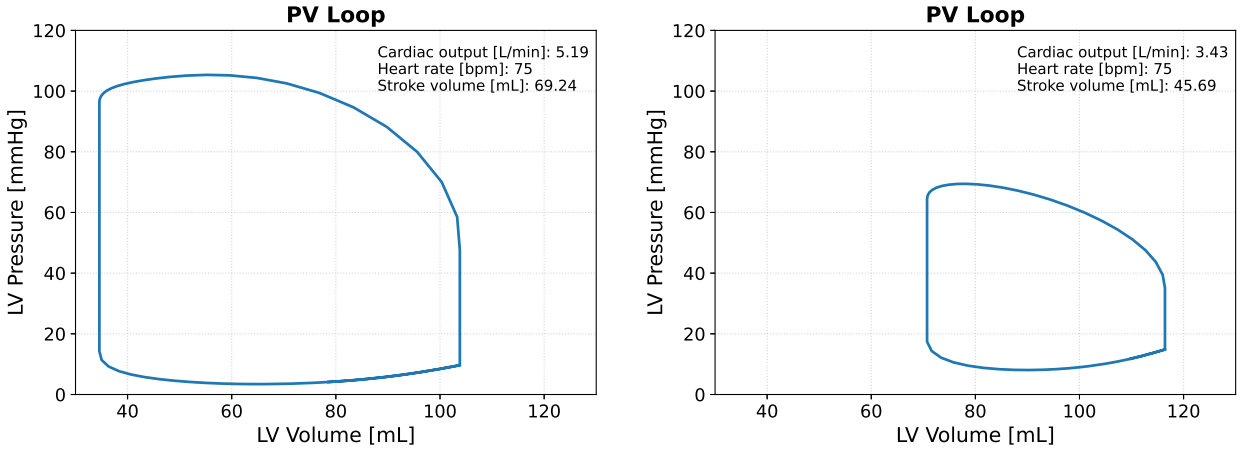
4.1.1. Analysis of Simulated Patients

A healthy individual and patient with a cardiomyopathy were simulated with the Burkhoff and Tyberg CV model. Healthy values of the CV parameters (Table 3.5) were employed to simulate the healthy subject. In contrast, the patient with a cardiomyopathy was modeled by a decreasing the value of the left ventricle end-systolic elastance ($E_{es_{LV}} = 1 mmHg/mL$); the remaining parameters were fixed at healthy values. Figure 4.1 presents the left ventricle pressure-volume (PV) loop from both patients.

The PV loop presents relevant information about the heart's state. The changes in ventricular pressure associated with changes in volume that occur during a cycle have valuable information to determine whether a subject's heart function is impaired. For instance, information about the volume can be assessed. A healthy patient (Figure 4.1a) exhibits a stroke volume (SV) equal to the difference between end-diastolic and end-systolic volumes (vertical lines) of almost $70mL$, which when multiplied by the heart rate (HR), gives a cardiac output (CO) of $5.2L/min$. On the other hand, the patient (Figure 4.1b) with a cardiomyopathy exhibits a SV of about $46mL$, which when multiplied by the HR, gives a CO of $3.4L/min$. Cardiac outputs lower than $4L/min$ with HR above $60bpm$ are abnormal and may reflect cardiac pathology.

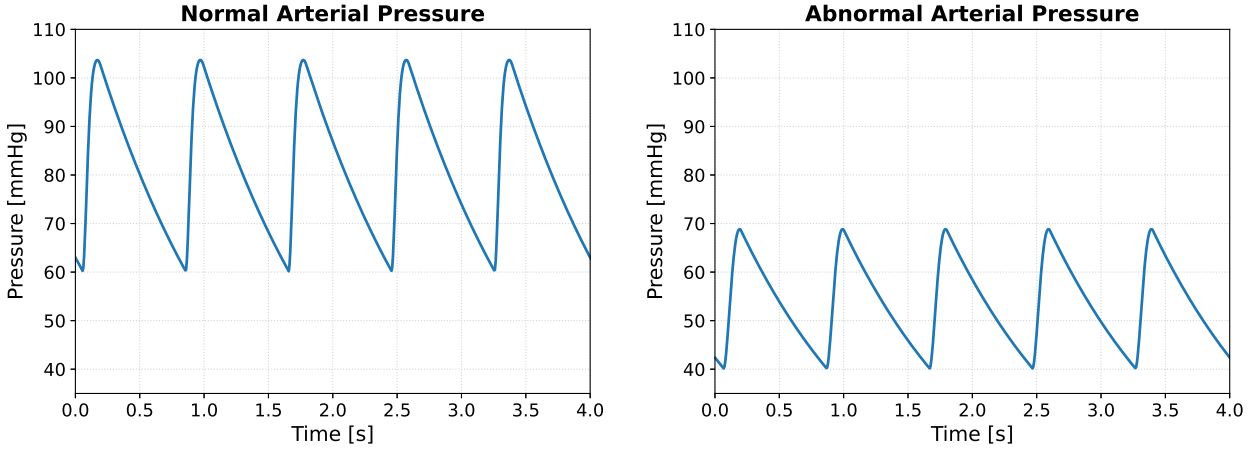
In addition to cardiac state, the PV loop captures information on the pressure and the relationships

between pressure and volume during the cardiac cycle. Systolic pressure can be approximated by the highest pressure value on the PV loop, and diastolic pressure when the aortic valve opens (top of the right vertical line). The end systolic pressure-volume relationship (ESPVR), which describes contractility (end-systolic elastance E_{es}), can be obtained by dividing the pressure by the volume when the aortic valve closes, i.e., end-systolic pressure and volume (top of the left vertical line). For example, the healthy subject exibits an end-systolic pressure (P_{es}) of $100mmHg$ and an end-systolic volume (V_{es}) of $34mL$. The ratio of these values gives the LV end-systolic elastance ($E_{es_{LV}} = 3mmHg/mL$). The same can be done for the sick patient ($P_{es} = 70mmHg$, $V_{es} = 70mL$), giving an LV end-systolic elastance that is characteristic of cardiomyopathy ($E_{es_{LV}} = 1mmHg/mL$).



(a) **(b)**

Figure 4.1 Left ventricle pressure-volume loop. a) Healthy subject with baseline parameters. b) Subject with cardiomyopathy represented by a reduction of the left ventricle end-systolic elastance ($E_{es_{LV}} = 1mmHg/mL$).



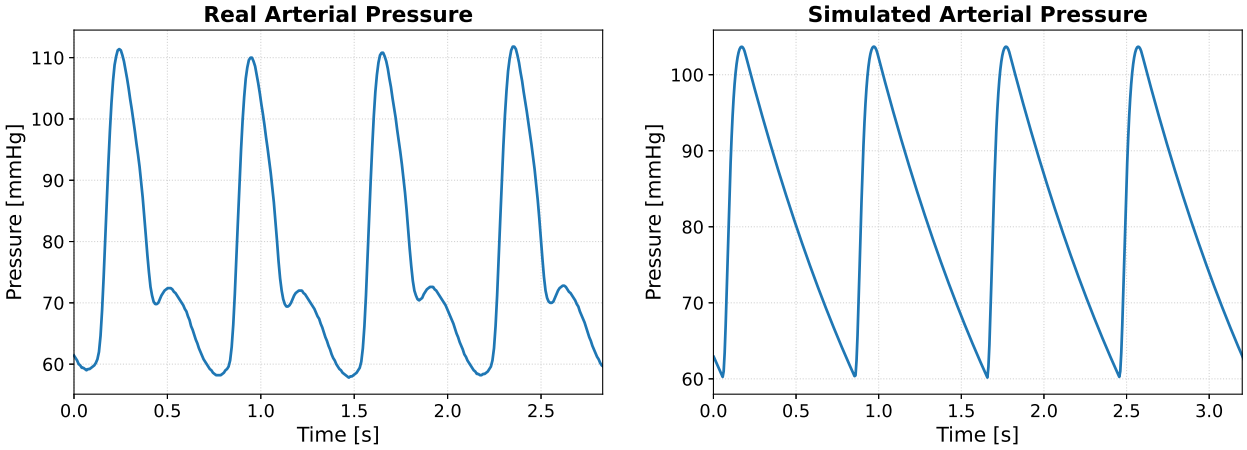
(a) **(b)**

Figure 4.2 Simulated arterial pressure ($P_{a,s}$). a) Healthy subject with baseline parameters. b) Subject with cardiomyopathy represented by a reduction of the left ventricle end-systolic elastance ($E_{es_{LV}} = 1mmHg/mL$).

The arterial pressure is the one of the most common waveforms used by physicians. It also includes important information about the patient's state. Figure 4.2 displays arterial pressure tracings generated by the simulator for a healthy subject and for a patient. Typical features that are extracted from this tracing include the *HR*, the systolic (maximum), diastolic (minimum) and mean pressures. Normal values for systolic pressure range between $90 - 120 mmHg$, for diastolic pressure between $50 - 80 mmHg$, and for the mean between $70 - 100 mmHg$. It can be appreciated that in the simulated healthy patient, all pressure values lie inside the normal ranges ($P_{sys} = 104 mmHg$, $P_{dias} = 40 mmHg$, $P_{mean} = 54.5 mmHg$). However, the patient with cardiomyopathy exhibits hypotension: systolic, diastolic and mean pressures are all below the normal ranges ($P_{sys} = 69 mmHg$, $P_{dias} = 40 mmHg$, $P_{mean} = 54.5 mmHg$). This is due to the fact that this patient is assigned a low contractility (low end-systolic elastance) in simulation, which results in a low stroke volume and therefore a low arterial pressure.

4.1.2. Comparison with Real Data

A typical arterial pressure wave from a CABG patient was extracted and compared with a simulated one to evaluate limitations of the Burkhoff and Tyberg CV model. Note that the focus of the comparison is on the shape of the waves rather than the exact pressure values. Both waves are showed in Figure 4.3.



(a) **(b)**

Figure 4.3 Comparison of real and simulated arterial pressure ($P_{a,s}$). a) Real arterial pressure from one CABG cohort patient. b) Simulated arterial pressure.

Three main differences between the real wave and the simulated one can be identified: variation in the amplitude of the real patient, a second peak when the pressure is decreasing, and a smaller curvature at the diastolic pressure during the transition between diastole and systole. The variation in the amplitude present in the real arterial pressure is due to respiration and its interactions with the CV system, an effect that is not considered in this CV model. As stated in [2], respiration varies the amplitude of the arterial pressure wave and this can be appreciated as a low frequency

component (respiratory rate) of the signal. The second peak that appears in the wave is known as the dicrotic notch, which is a consequence of the aortic valve closing. Detecting the dicrotic notch allows one to differentiate between systole and diastole in the aortic pressure wave. The lumped parameter nature of the CV model used in this project does not permit simulation of this second peak. Finally, the smother diastolic pressure transition in the real patient might be due to a non-ideal aortic valve, which unlike the CV model, does not open instantaneously.

In conclusion, it could be said that the Burkhoff and Tyberg CV model can represent reasonable values and relationships between diastolic, systolic, and mean pressures and volumes from all six compartments (left ventricle, systemic arteries, systemic veins, right ventricle, pulmonic arteries and pulmonic veins). Moreover, a good approximation of the these states results in a good description of physiological variables such as cardiac output, stroke volume or even cardiac power output, which are central to this project. At the same time, finer details in the shape of the waveform may be poorly represented or simply not modeled at all. Therefore, the effects of respiration and the dicrotic notch, among others, cannot be assessed in this particular model. To do so, one could extend the model to include a pulmonary system and its interactions with the heart and vasculature. Alternatively, it could be more suitable to develop a 1D or 3D model, as presented in Section 2.2.2.

4.2. System Identification

Results related to the autoencoder and estimation of CV parameters are presented and discussed in this section. First, the results related to the training and evaluation of the decoder are presented. Then, the results of training and evaluation of the encoder are described. Finally, the estimated CV parameters are discussed and ranges are defined for use with domain randomization during RL policy training.

4.2.1. Decoder Training

The decoder used in the autoencoder to estimate the CV parameters was trained with simulated data to reproduce the behavior of the CV simulator. Recall that 0.7 million samples were generated for this training process. Each sample contained as input data the CV parameters and the labels were a single cardiac cycle of the three pressures ($P_{a,s}$, $P_{v,s}$,$P_{a,p}$) and the stroke volume (SV) (explained in detail in Section 3.4). Figure 4.4 shows the decoder training curves with the optimized hyperparameters presented in Table 3.8.

From the evolution of the validation loss (Figure4.4a) it can be appreciated that the neural network (NN) starts to overfit at around 12 epochs. Following the early stopping technique, 12 epochs were chosen for training the decoder. Furthermore, one aspect of these curves that is worth noting is the fact that at the beginning, the validation loss is lower than the training loss. This may happen

because of the regularization used in training, in particular dropout and L_2 regularization. Also, the accuracy achieved (at 12 epochs) is around 0.996. Considering that the accuracy in this project is computed as the cosine similarity between the NN output vector and the label vector, this value represents a difference of about 5.1^o between these two vectors.

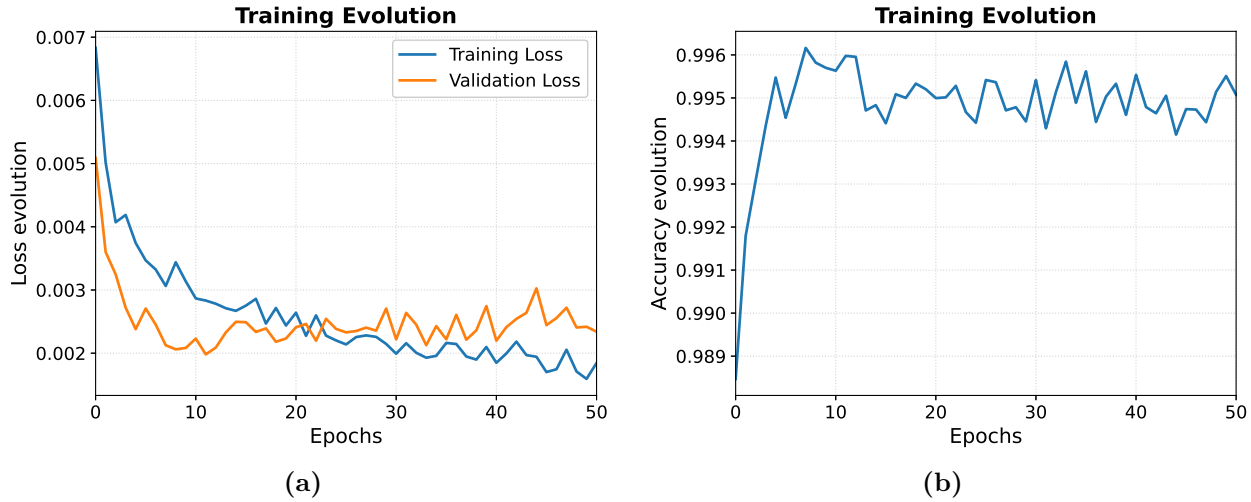


Figure 4.4 Decoder training curves. a) Loss evolution of training (blue) and validation (orange) sets. b) Accuracy evolution of validation set.

To further evaluate the decoder, a set of baseline parameters (Table 3.5), not used for training, were tested as inputs to the NN to generate the expected pressure waves and stroke volume. A comparison of the generated waves from both the decoder and simulator is shown in Figure 4.5. Also, the root mean squared error ($RMSE$) of the pressure waves and stroke volume are presented in Table 4.1.

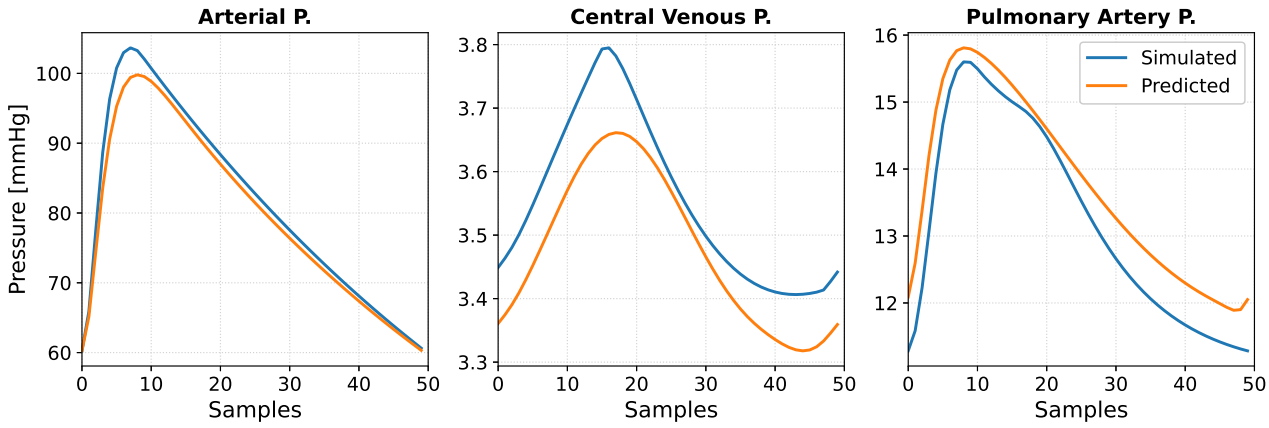


Figure 4.5 Comparison of generated pressure waves from simulator (blue) and decoder (orange). Baseline parameters used. a) Arterial pressure ($P_{a,s}$). b) Central venous pressure ($P_{v,s}$). c) Pulmonary artery pressure ($P_{a,p}$).

	$P_{a,s}$ [mmHg]	$P_{v,s}$ [mmHg]	$P_{a,p}$ [mmHg]	SV [mL]
RMSE	2.0455	0.0822	0.5628	5.3604

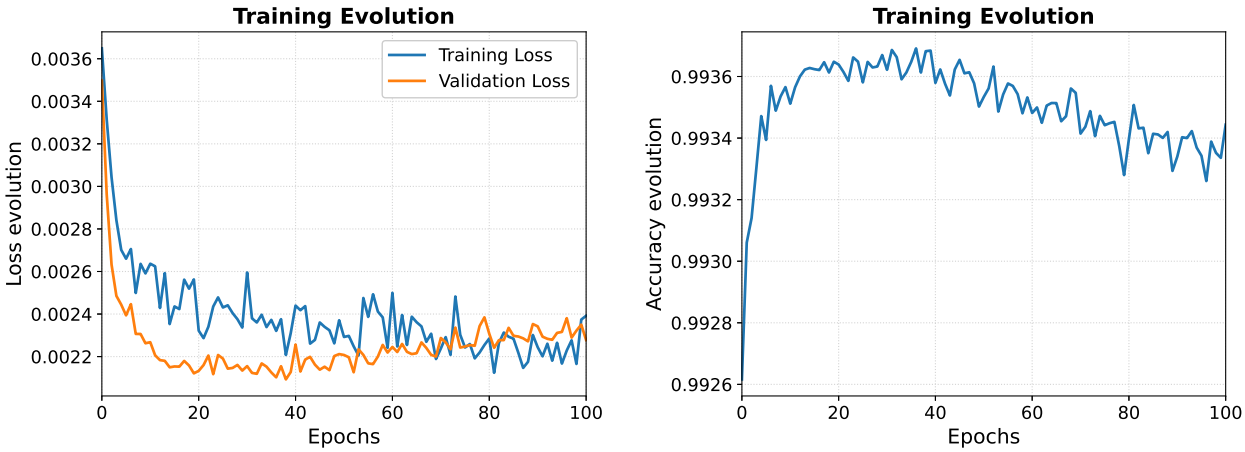
Table 4.1 Root mean squared error (RMSE) between generated pressure waves (arterial pressure ($P_{a,s}$), central venous pressure ($P_{v,s}$), pulmonary artery pressure ($P_{a,p}$)) and stroke volume (SV) from simulator and decoder.

The arterial pressure wave is very similar with a modest error at the systolic value, where the decoder predicts a maximum of $100mmHg$ when the true value is $105mmHg$. The RMSE is low compared to the scale of this signal, with a value of $2.0455mmHg$. Regarding the central venous pressure, the most notable errors occur at the maximum and minimum of the wave. However, these differences are low, at $0.1mmHg$. Together with the low RMSE ($0.0822mmHg$), it is fair to say that the decoder generated an accurate central venous pressure wave. With respect to the pulmonary artery pressure wave, the highest difference appears at the end of the wave, though again the absolute magnitude of this error was small. The RMSE of this predicted wave compared to the simulated one is $0.563mmHg$. Finally, the decoder generated a stroke volume of $63.88mL$, which is quite comparable to the true value ($SV = 69.24mL$), with a difference of only $5.36mL$. Overall, the training results and the decoder's performance on a set of baseline CV parameters demonstrate that it approximates the CV simulator quite well.

4.2.2. Encoder Training and Cardiovascular Parameter Estimation

Once the decoder learned to reproduce the CV simulator, the encoder used in the autoencoder was trained in two stages. First it was trained with simulated data, and then it was trained with the real data from CABG cohort (776341 samples). Recall that in this case both input and output data were a cycle of the three pressures ($P_{a,s}$, $P_{v,s}$,$P_{a,p}$) and the stroke volume (SV), as the encoder was trained within the full autoencoder architecture (explained in detail in Section 3.4). Figure 4.4 shows the encoder training curves using real data with the optimized hyperparameters presented in Table 3.8.

From the trajectory of the validation loss curve (Figure4.4a) it is clear that the encoder NN starts to overfit at around 40 epochs. As a result, 40 epochs were chosen for training the encoder. As with training the decoder, the validation loss is smaller than the training loss, likely due to the regularization effect explained previously. The accuracy achieved (at 40 epochs) with the encoder is around 0.993, which in terms of cosine similarity between the NN output vector and the label vector, equates to a difference of about 6.8^o between these two vectors.



(a) **(b)**

Figure 4.6 Encoder training curves with real data. a) Loss evolution of training (blue) and validation (orange) sets. b) Accuracy evolution of validation set.

To evaluate the encoder, the baseline CV parameters (Table 3.5) were first used to generate a set of the expected pressure waves and stroke volume. This state data was then used as input for the the encoder to generate a set of predicted parameters. A comparison of the nine CV parameters estimated by the encoder with the true values is presented in Table 4.2. Right ventricle end-systolic elastance ($E_{es_{RV}}$) is the parameter with the greatest estimation error, with an absolute relative error (RE) of 237%. This poor approximation may occur because the estimated left ventricle end-systolic elastance ($E_{es_{LV}}$) is lower than the true value, so that a larger value of $E_{es_{RV}}$ may compensate for this loss. Pulmonic arterial resistance ($R_{a,p}$) and pulmonic arterial compliance ($C_{a,p}$) also exhibit notable estimation errors, with an absolute RE of 40.4% and 30.1%, respectively. In contrast, systemic arterial resistance ($R_{a,s}$), systemic venous compliance ($C_{v,s}$) and stressed volume (V_s) are properly approximated, with absolute RE of 0.32%, 2.17% and 4.92%, respectively.

	$E_{es,RV}$ [mmHg/mL]	$E_{es,LV}$ [mmHg/mL]	$R_{a,p}$ [mmHg·s/mL]	$R_{a,s}$ [mmHg·s/mL]	$C_{a,p}$ [mL/mmHg]	$C_{a,s}$ [mL/mmHg]	$C_{v,p}$ [mL/mmHg]	$C_{v,s}$ [mL/mmHg]	V_s [mL]
True parameter	0.7	3.0	0.03	0.9	13	1.32	8	70	750
Estimated parameter	2.36	2.39	0.018	0.903	16.91	1.17	7.27	68.50	713.1
Absolute RE [%]	236.72	20.48	40.37	0.32	30.11	11.40	9.07	2.17	4.92

Table 4.2 Comparison of true and estimated parameters from simulated waves. Absolute relative error (RE) is also computed.

Further evaluation of the full autoencoder output was carried out. The pressure waves and stroke volume obtained from the baseline parameter set were compared to the generated ones from the autoencoder. Results are plotted in Figure 4.7. Also, the root mean squared error (RMSE) of the pressure waves and stroke volume are presented in Table 4.3.

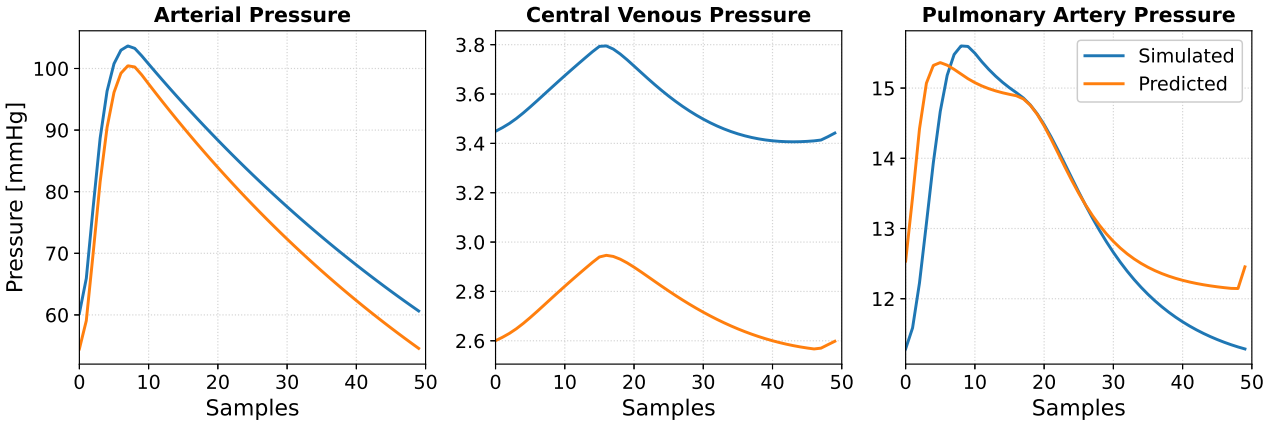


Figure 4.7 Comparison of generated pressure waves from simulator (blue) and autoencoder (orange). Baseline parameters used. a) Arterial pressure ($P_{a,s}$). b) Central venous pressure ($P_{v,s}$). c) Pulmonary artery pressure ($P_{a,p}$).

	$P_{a,s}$ [mmHg]	$P_{v,s}$ [mmHg]	$P_{a,p}$ [mmHg]	SV [mL]
RMSE	5.1388	0.8236	0.7021	3.8101

Table 4.3 Root mean squared error (RMSE) between generated pressure waves (arterial pressure ($P_{a,s}$), central venous pressure ($P_{v,s}$), pulmonary artery pressure ($P_{a,p}$)) and stroke volume (SV) from simulator and autoencoder.

The generated waves from the full autoencoder exhibit greater differences than when only the decoder was used, as might be expected. The accumulation of error from the encoder together with the decoder could explain this effect. The arterial and central venous pressure waves are lower in all the cycle. The pulmonary artery pressure is higher in general, except for the maxima values. Even though differences between estimated and true values are more noticeable, the RMSE metrics indicate that the autoencoder still performs quite well. Arterial pressure prediction gives an RMSE of $5.14 mmHg$, which compared to its amplitude is quite modest. Predictions of the central venous pressure and pulmonary artery pressures result in RMSE's that are less than $1 mmHg$. Finally, the autoencoder predicted a stroke volume of $65.43 mL$, which compared to the true value ($SV = 69.24 mL$) it is a difference of $3.81 mL$.

After evaluating the autoencoder on simulated data, the encoder was used to find the latent space of the real CABG cohort, that is, estimate the nine CV parameters for each patient in the cohort. Recall that one motivation for performing this estimation was to find realistic parameter ranges to set bounds for domain randomization (DR) applied to reinforcement learning (explained in Section 3.4). Summary statistics of the nine CV parameters estimated, as well as comparison with literature parameters are presented in Table 4.4. Moreover, probability distributions of the estimated parameters are represented in Figure 4.8.

	$E_{es,RV}$ [mmHg/mL]	$E_{es,LV}$ [mmHg/mL]	Ra,p [mmHg·s/mL]	Ra,s [mmHg·s/mL]	Ca,p [mL/mmHg]	Ca,s [mL/mmHg]	Cv,p [mL/mmHg]	Cv,s [mL/mmHg]	Vs [mL]
Literature parameter	0.7	3.0	0.03	0.9	13	1.32	8	70	750
Estimated mean	3.35	3.26	0.026	0.755	10.99	1.37	18.17	51.4	1060.6
Estimated std dev	0.83	0.99	0.081	0.292	8.53	0.89	8.64	23.49	232.8
Absolute *RE* [%]	379.62	8.89	13.36	16.11	15.47	3.43	127.12	26.56	41.42

Table 4.4 Comparison of literature [1] and estimated parameters from CABG cohort. Mean, standard deviation (std dev) and absolute relative error (RE) between baseline and mean are presented.

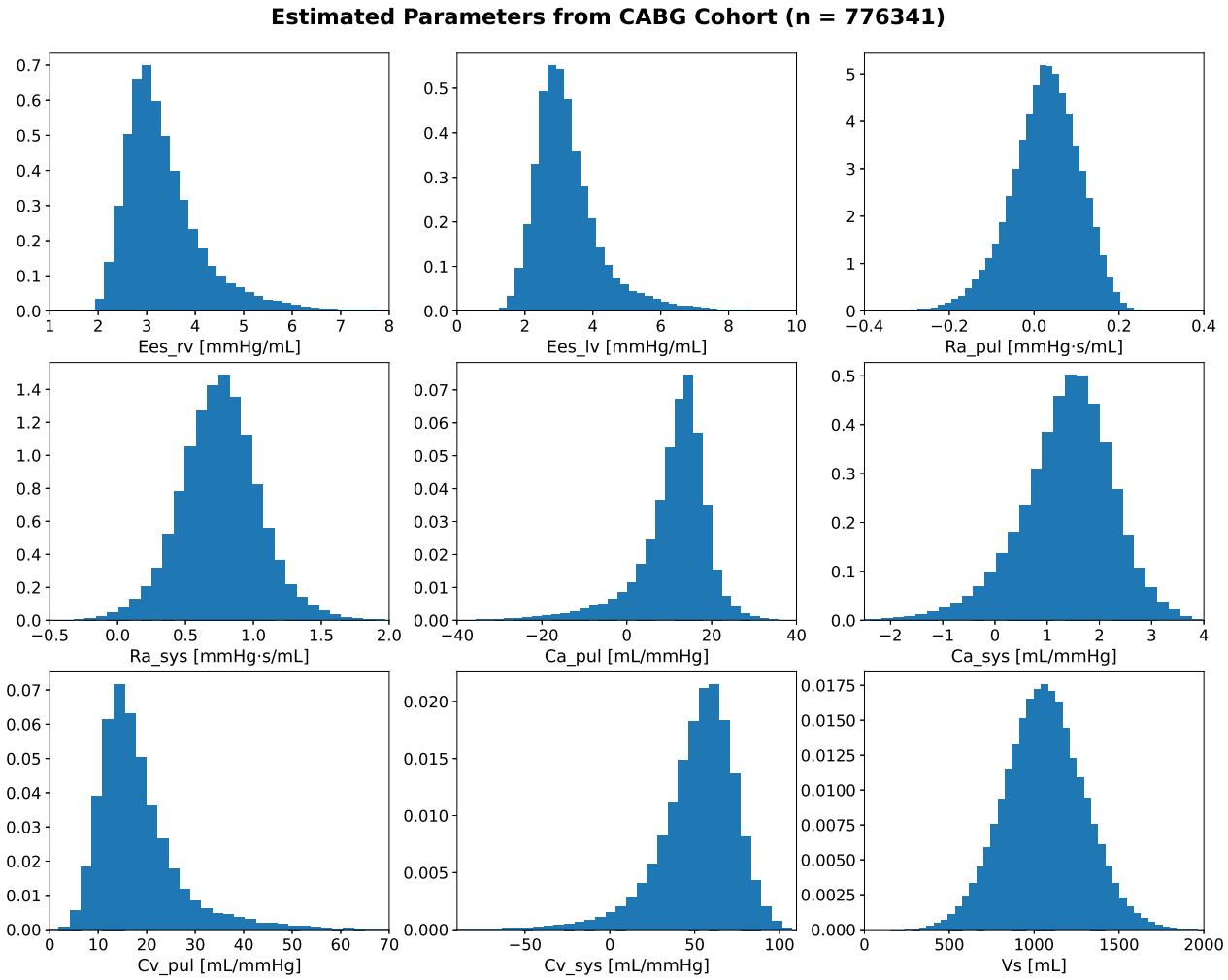


Figure 4.8 Estimated CV parameters from CABG cohort (n = 776341) employing the encoder. From left to right and top to bottom: Right ventricular end-systolic elastance (Ees_rv); Left ventricular end-systolic elastance (Ees_lv); Pulmonic arterial resistance (Ra_pul); Systemic arterial resistance (Ra_sys); Pulmonic arterial compliance (Ca_pul); Systemic arterial compliance (Ca_sys); Pulmonic venous compliance (Cv_pul); Systemic venous compliance (Cv_sys); Stressed volume (Vs).

The means of the parameters estimated by the autoencoder on the real-world CABG cohort data exhibit striking similarities with literature parameters [1] from a healthy 75kg subject. The similarity with healthy values is not surprising given that CABG is commonly performed on patients whose

hearts have not yet been significantly damaged by coronary artery disease. As a result, the means of these distributions in such patients would be expected to be close to healthy values. However, right ventricle end-systolic elastance does differ markedly from the literature value (absolute RE of 380% between estimated mean and literature parameter). Also, pulmonic venous compliance is significantly different, with an absolute RE of 127%. Another problem with the estimation that can be appreciated from Figure 4.8 is that some parameters distributions have left tails with negative values. Negative CV parameter values are not physiologic, and might be the result of limitations of the decoder working as the simulator. During training the encoder may have learned that producing negative values of the CV parameters (latent variables) and passing them to the decoder nevertheless generated reasonable approximations to the pressure waves.

To sum up, although the estimated mean of the nine CV parameters looks promising, this system identification tool needs to be refined and improved. Future work considers to put efforts on improving this strategy to be able to get only positive parameters and a better estimation of certain parameters such as the right ventricle end-systolic elastance. In summary, although the estimated means of the nine CV parameters look promising, this system identification tool needs to be improved. Future work will focus on constraining the output of the encoder to be positive and improving the estimation of certain parameters such as the right ventricle end-systolic elastance.

In order to obtain parameter ranges for the DR, the first approach that was tried was to use the estimated mean $\pm$ standard deviation from Table 4.4. However, those ranges proved to be too wide to for the policy to converge. In fact, as shown in [22], a strategy of gradually enlarging the DR ranges as the policy was learning was an effective way to achieve convergence over a wide ranges of system parameters. In future work, such a strategy will be implemented. For this project the approach that was successful was to reduce the parameter ranges further. Specifically, parameter ranges were set at the estimated mean $\pm$ one third of the standard deviation. Furthermore, the right and left ventricular end-systolic elastances and stressed volume were permitted to change to arbitrarily small or large values to simulate a patient in cardiogenic shock whose cardiac output was low. The final ranges of the nine CV parameters used in the DR were:

$$E_{es_{RV}}\,[mmHg/mL] \in [0.42, 0.98],\;\; E_{es_{LV}}\,[mmHg/mL] \in [0.8, 1.4],$$

$$R_{a,p}[mmHg \cdot s/mL] \in [0.023, 0.053],\;\; R_{a,s}\,[mmHg \cdot s/mL] \in [0.658, 0.852],$$

$$C_{a,p}\,[mL/mmHg] \in [8.16, 13.84],\;\; C_{a,s}\,[mL/mmHg] \in [1.07, 1.66],$$

$$C_{v,p}\,[mL/mmHg] \in [15.29, 21.05],\;\; C_{v,s}\,[mL/mmHg] \in [43.57, 59.23],\;\; V_s\,[mL] \in [650, 850]$$

4.3. Reinforcement Learning for Cardiogenic Shock Patients

This final section presents the results from the reinforcement learning (RL) experiments. First, the effect of DR on policy robustness is assessed. Second, policies with restricted actions are analyzed and compared. Third, the importance of penalty terms is evaluated. Finally, the performance of the best policy is discussed in detail. Note that in all cases, modeling a patient in cardiogenic shock (CS) was achieved by setting the CV parameters of the model in a configuration that resulted in an initial state with a low cardiac output (below $3.5 L/min$). The learning objective was to identify a policy that could restore cardiac output to a normal range (at least $5.0 L/min$) by modulating a subset of the CV parameters (explained in detail in Section 3.3.4). Moreover, the hyperparameters used to train the policies are presented in 3.7.

4.3.1. Assessment of Domain Randomization Robustness

Assessment of the importance of domain randomization (DR) was carried out by training a policy without DR and a policy with DR. The policy trained without DR always interacted with a CV model with the same parameter values (corresponding to shock) at the beginning of every episode. In both cases, the reward function included the reward for cardiac output plus the power penalty term ($\mathcal{R}(s,a) = \mathcal{R}^{+}(s,a) + \mathcal{R}^{-}_{pp}(s,a)$).

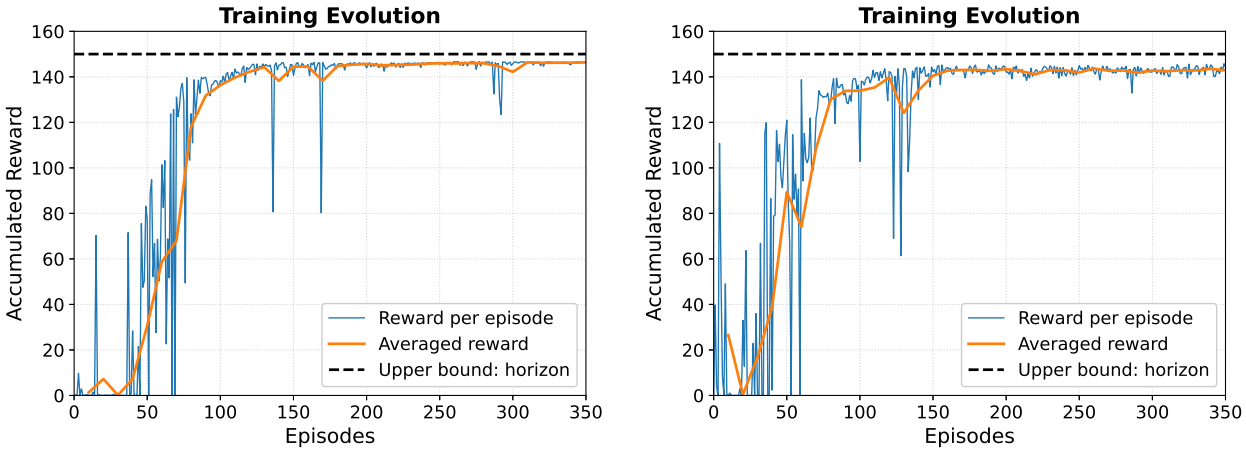
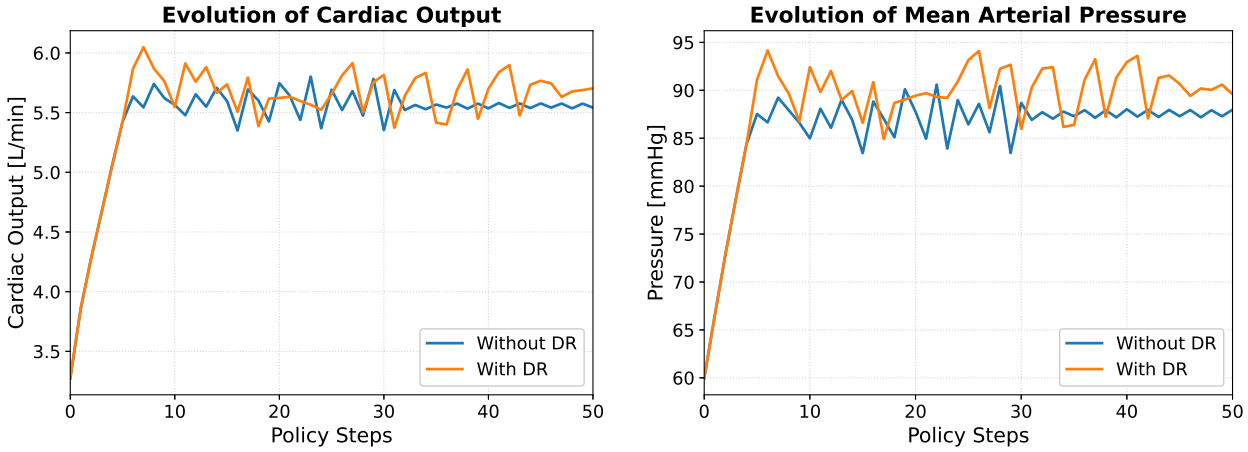


Figure 4.9 Policy training curves with reward per episode (blue), averaged reward over 10 episodes (orange) and training horizon (dashed black). a) Policy trained without domain randomization (DR). b) Policy trained with domain randomization (DR).

The training curves for the two approaches are shown in Figure 4.9. Both policies converge at practically the same moment, at around 150 episodes. However, the policy trained without DR exhibits less noise during training and achieves a slightly higher accumulated reward, 146, compared to 142 with DR. This slight reduction in reward might be expected, as DR creates variability in the CV parameters, forcing the policy to generalize over different CV environments at some cost to

overall performance.

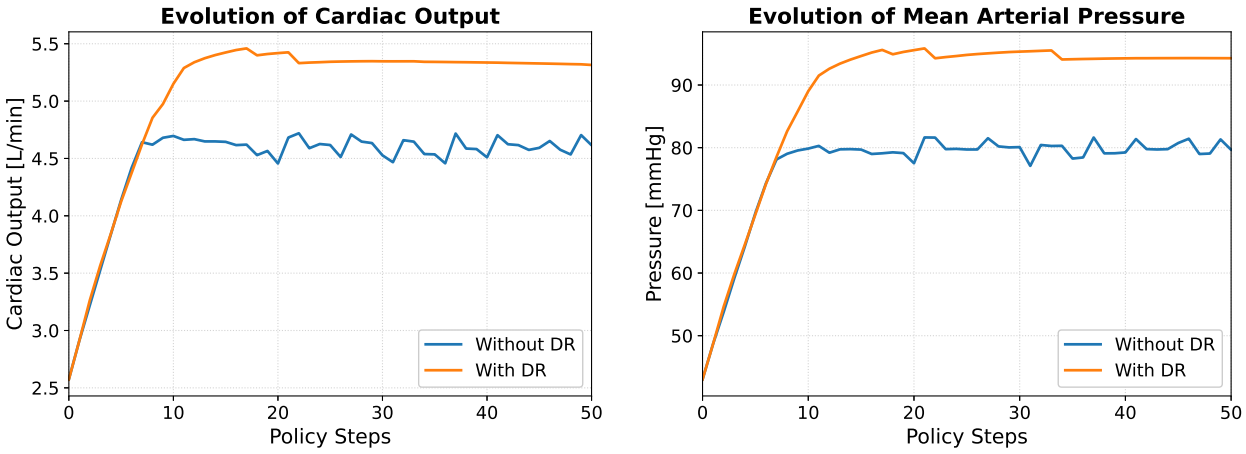
To directly compare the two policies each one was evaluated on a CV environment with the same parameters: the baseline values (Table 3.5) together with low LV end-systolic elastance ($E_{es_{LV}} = 1 mmHg/mL$), which is the same CV environment used to train the policy without DR. Results show that both policies are able to restore the cardiac output (Figure 4.10a) and mean arterial pressure (Figure 4.10b) in the simulated patient, getting final values of around $5.5 L/min$ and $90 mmHg$, respetively. However, the policy trained with DR achieves the goal with moderately more noise and somewhat higher values of cardiac output compare to the policy trained without DR. This may be happening due to a trade-off between robustness achieved by the DR and performance. The policy with DR should be more robust, but at the same time could have poorer performance.



(a) **(b)**

Figure 4.10 Policy performance from training initial conditions without domain randomization (baseline parameters with low LV end-systolic elastance ($E_{es_{LV}} = 1 mmHg/mL$)). Comparison of policies trained without (blue) and with (orange) DR. a) Cardiac output (CO). b) Mean arterial pressure ($\bar{P}_{a,s}$).

In order to assess DR robustness, both policies were evaluated on a set of initial CV parameters not seen by either approach during training. The CV parameters were set at the baseline parameters with even lower LV end-systolic elastance ($E_{es_{LV}} = 0.6 mmHg/mL$) and pathological high systemic venous resistance ($R_{v,s} = 0.06 mmHg \cdot s/mL$). As can be seen in Figure 4.11, the policy trained without DR is not able to restore the cardiac output to a normal value above $5.0 L/min$. On the other hand, the policy trained with DR, despite facing a new environment, is capable of recovering the cardiac output by rising it to $5.3 L/min$. The value of DR is highlighted by this example, in which a policy can still perform well when operating in an unexpected environment. Therefore, training a policy with DR may be a valuable approach for closing the *sim2real* gap in the management of CS.



(a) **(b)**

Figure 4.11 Policy performance from initial conditions not used during training (baseline parameters with low LV end-systolic elastance ($E_{es_{LV}} = 0.6 mmHg/mL$) and high systemic venous resistance ($R_{v,s} = 0.06 mmHg \cdot s/mL$)). Comparison of policies trained without (blue) and with (orange) DR. a) Cardiac output (CO). b) Mean arterial pressure ($\bar{P}_{a,s}$).

4.3.2. Comparison of Restricted Policies

Policies with restricted actions were trained and evaluated to assess the relative importance of different actions. This experiment could also be viewed as modeling a scenario wherein a particular intervention was contraindicated (off-limits, e.g. due to allergy) for a patient. Three different policies were trained: a policy without actions to change the stressed volume (V_s), a policy without actions to change the heart rate (HR), and a policy without actions to change the end-systolic elastance (E_{es}). All three policies were trained with DR and the reward function included both the positive reward for cardiac output plus the power penalty term ($\mathcal{R}(s,a) = \mathcal{R}^{+}(s,a) + \mathcal{R}^{-}_{pp}(s,a)$). For each policy, five rounds of training were performed and the trained policy with the best performance was used for evaluation.

Training curves are presented in Figure 4.12. The policy without HR actions converges faster during training and achieves a final accumulated reward of 141. On the other hand, the policy without E_{es} converges more gradually and achieves a lower accumulated reward of about 120. An open question to be analyzed in the future is whether optimizing the hyperparameters for each choice of action space would improve policy convergence and accumulated reward.

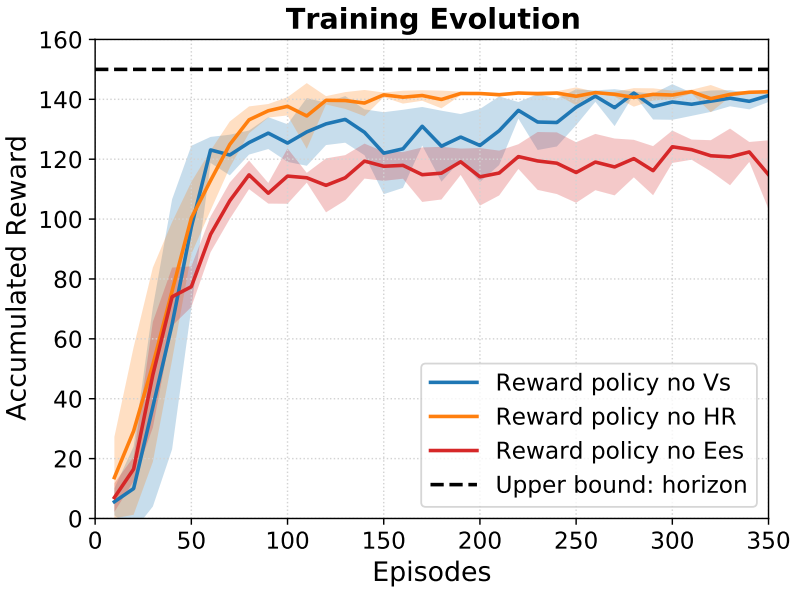
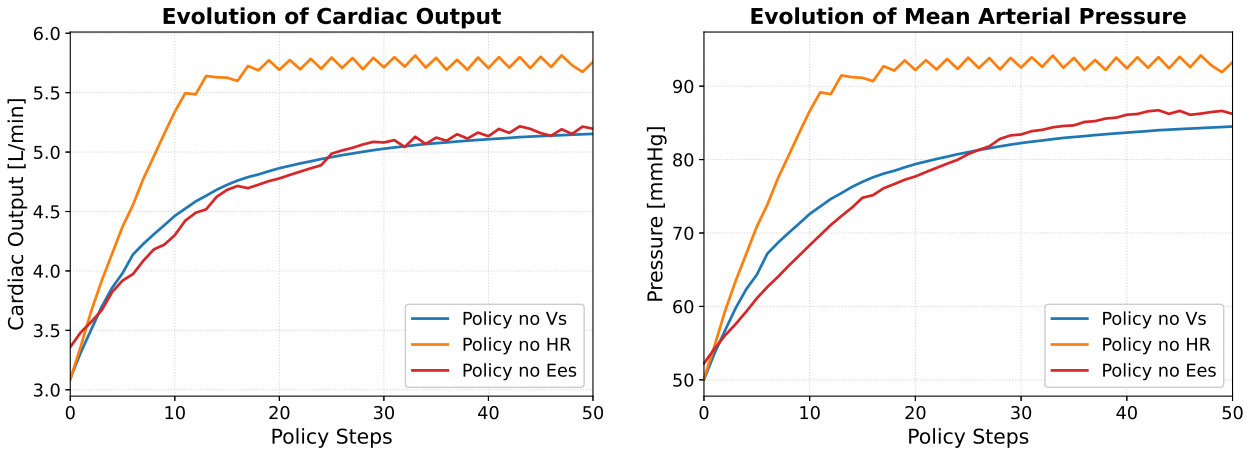


Figure 4.12 Policy training curves with mean (line) ± standard deviation (shaded region) of averaged reward over 10 episodes. 5 policies trained for each case: policy without stressed volume (V_s) actions (blue), policy without heart rate (HR) actions (orange), and policy without end-systolic elastance (E_{es}) actions (red). Horizon represented with black dashed line.



(a) **(b)**

Figure 4.13 Policy performance from the three restricted policies. Initial conditions: baseline parameters with low LV end-systolic elastance ($E_{es_{LV}} = 0.8 mmHg/mL$) and high systemic venous resistance ($R_{v,s} = 0.04 mmHg \cdot s/mL$). Comparison of policies trained without stressed volume (V_s) actions (blue), without heart rate (HR) actions (orange), and without end-systolic elastance (E_{es}) actions (red). a) Cardiac output (CO). b) Mean arterial pressure ($\bar{P}_{a,s}$).

The performance of the three policies was compared by setting up the following initial conditions: baseline parameters with low LV end-systolic elastance ($E_{es_{LV}} = 0.8 mmHg/mL$) and high systemic venous resistance ($R_{v,s} = 0.04 mmHg \cdot s/mL$). Results show that the policy without actions to change HR restores the cardiac output (Figure 4.13a) and mean arterial pressure (Figure 4.13b)

the fastest, obtaining final values of around $5.7L/min$ and $93mmHg$. On the other hand, policies without V_s and E_{es} actions have greater difficulty to treating the simulated patient with CS. In fact, both policies achieve similar performance, taking 30 policy steps to raise the cardiac output to $5L/min$. These results imply that actions that change HR are less important than the actions that change either V_s or E_{es}.

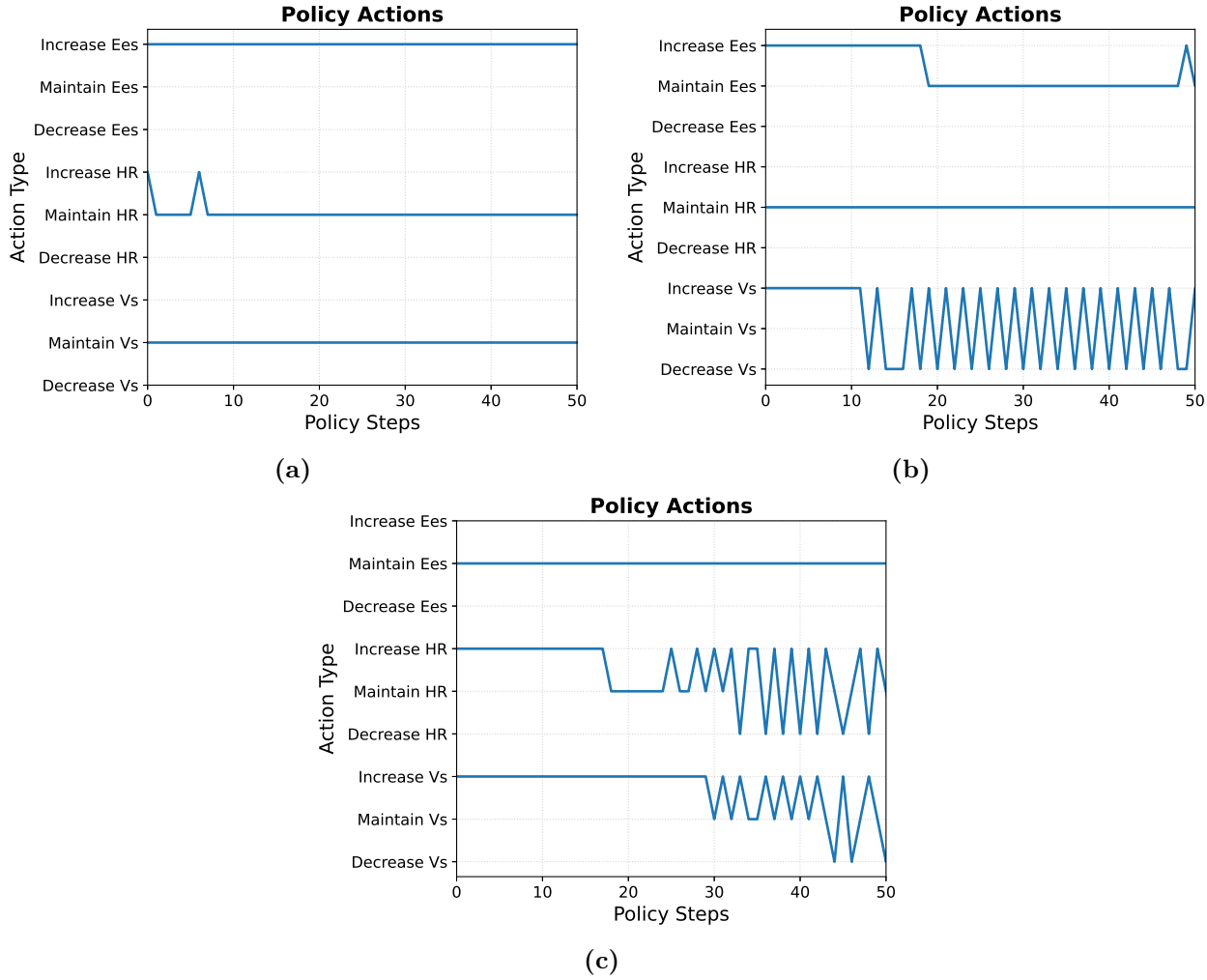


Figure 4.14 Actions from the three restricted policies. a) Policy without stressed volume (V_s) actions. b) Policy without heart rate (HR) actions. c) Policy without end-systolic elastance (E_{es}) actions.

A closer look at the actions taken by each policy (Figure 4.14) reveals further insights. First, the policy without V_s actions (Figure 4.14a) increases the cardiac output simply by continually rising the end-systolic elastance all the time, and it rarely ever uses the HR action. This behavior reinforces the conclusion that changes in HR have smaller impact on cardiac output and thus, actions that manipulate it are less important. Regarding the policy without HR actions (Figure 4.14b), it first increases both E_{es} and V_s to reach the desired cardiac output as fast as possible. Then, in order to maintain the cardiac output at the healthy value, the policy keeps E_{es} constant and oscillates V_s,

increasing and decreasing it successively. This oscillatory behavior occurs because the policy, when reaches the goal cardiac output of $5.5L/min$, it has freedom to do whatever it wants, provided that the cardiac output is maintained. This undesirable behavior is addressed in the next section by introducing a "drug" (action) penalty term. Finally, the policy without E_{es} actions (Figure 4.14c) raises the cardiac output by increasing both HR and V_s. In this case HR actions are successfully employed, which might reflect the fact that at higher blood volumes (increase of V_s), rising HR has a greater effect on increasing cardiac output.

4.3.3. Importance of Penalty Terms

The importance of penalty terms was assessed by training three different policies: one without any penalty terms ($\mathcal{R}(s,a) = \mathcal{R}^{+}(s,a)$), another with the power penalty term ($\mathcal{R}(s,a) = \mathcal{R}^{+}(s,a) + \mathcal{R}^{-}_{pp}(s,a)$), and a third with both power and drug penalty terms ($\mathcal{R}(s,a) = \mathcal{R}^{+}(s,a) + \mathcal{R}^{-}_{pp+dp}(s,a)$). All three policies were trained with DR and for each policy, five rounds of training were performed; the trained policy with the best performance was chosen for further evaluation.

Training curves are presented in Figure 4.15. The policy without penalty terms converged much faster than the other two, after only 60 episodes. On the other hand, the policy with only a power penalty term needed about 200 episodes to converge. Interestingly, the policy with both a power penalty and a drug penalty term took 150 episodes to converge, showing that drug penalty may help on convergence. Regarding total reward, all three policies achieved a similar accumulated reward of 140 upon convergence.

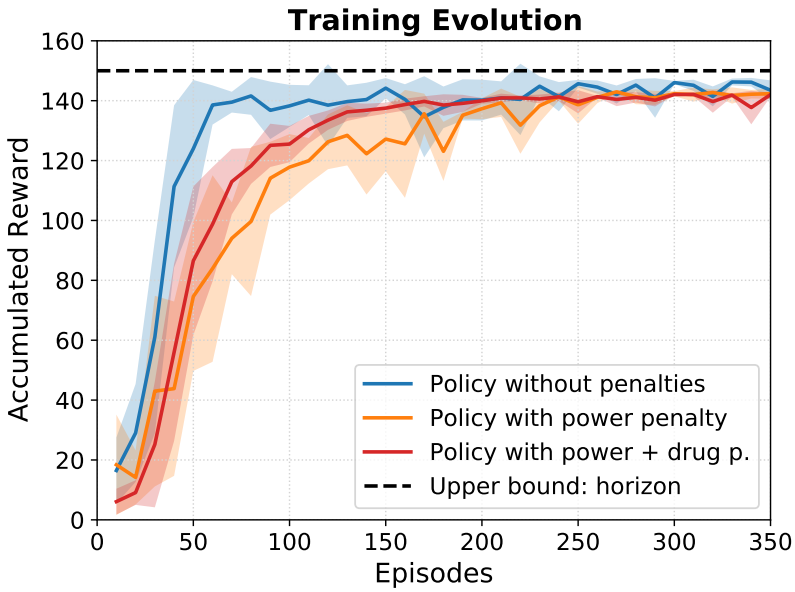
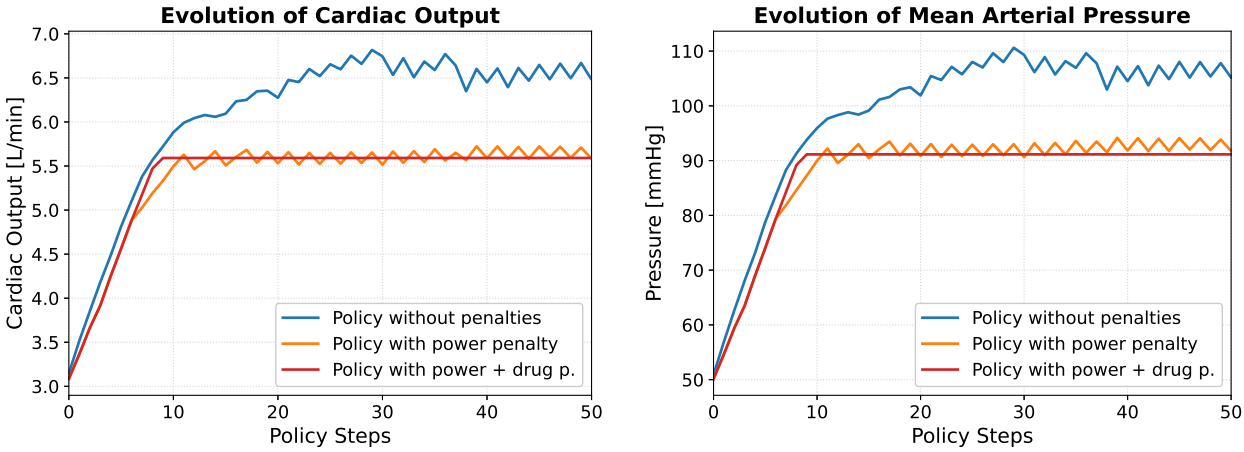


Figure 4.15 Policy training curves with mean (line) ± standard deviation (shaded region) of averaged reward over 10 episodes. 5 policies trained for each case: policy without penalty terms (blue), policy with only power penalty term (orange), and policy with both drug and power penalty terms (red). Horizon represented with black dashed line.

The three policies were evaluated by setting up the same initial conditions as before: baseline parameters with low LV end-systolic elastance ($E_{es_{LV}} = 0.8 mmHg/mL$) and high systemic venous resistance ($R_{v,s} = 0.04 mmHg \cdot s/mL$). Results show that the policy without any penalty terms has freedom to increase cardiac output (Figure 4.16a) as much as it wants, reaching a value of around $6.5 L/min$. This in turn induces hypertension in the simulated patient, a pathologic condition, with values of mean arterial pressure (Figure 4.16b) reaching $110 mmHg$. On the other hand, introducing the power penalty term encourages the policy to not increase the cardiac output excessively (around $5.6 L/min$) and thus avoiding hypertension. However, an undesirable oscillatory behavior occurs as seen in previous experiments. These oscillations can be removed by including the drug penalty term, which penalizes the policy for taking many actions. This last penalty forces the policy to only take actions when necessary, and once the cardiac output level is reached, it is encouraged to take no further parameter altering actions.



(a) **(b)**

Figure 4.16 Policy performance from the three policies with different penalty terms. Initial conditions: baseline parameters with low LV end-systolic elastance ($E_{es_{LV}} = 0.8 mmHg/mL$) and high systemic venous resistance ($R_{v,s} = 0.04 mmHg \cdot s/mL$). Comparison of policies trained without penalty terms (blue), with only power penalty term (orange), and with both drug and power penalty terms (red). a) Cardiac output (CO). b) Mean arterial pressure ($\bar{P}_{a,s}$).

Taking a closer look at cardiac power output (4.17), which is a measure of how much power the heart is consuming, it can be seen that the policy without penalty terms exceeds $1.2 W$. Such high cardiac power outputs are likely detrimental to the heart itself and its recovery. Moreover they result in adverse systemic effects such as hypertension. On the other hand, policies with the power penalty maintained power outputs below $1.2 W$.

Finally, the effect of the drug penalty term can be appreciated in Figure 4.18. Without this penalty (4.18a), the policy behaves in an oscillatory way. First, it takes the appropriate actions to increase the cardiac output fast to $5.5 L/min$, but then it alternates opposing actions (e.g. increase/decrease V_s) to maintain the desired cardiac output level. On the other hand, when adding the drug penalty,

the policy only acts when necessary. As shown in Figure 4.18b, once the policy raises the cardiac output to the desired level, it does not apply any other action.

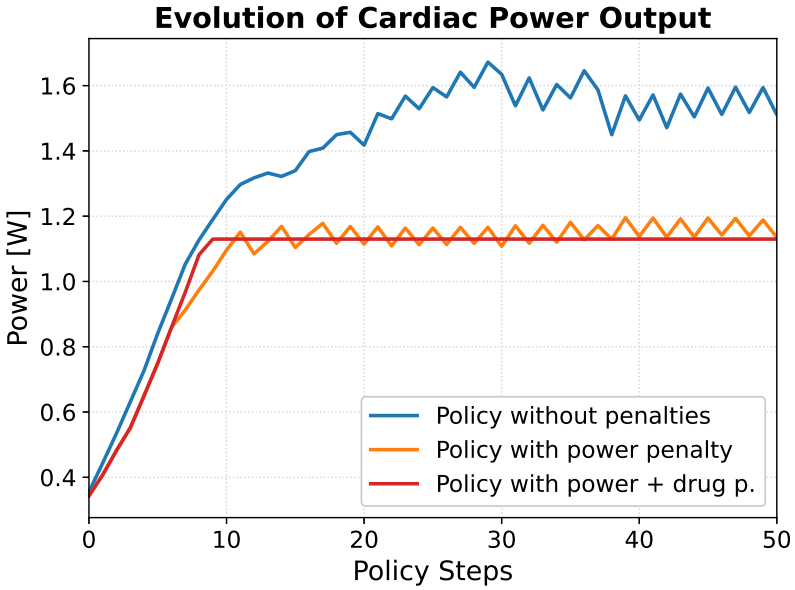


Figure 4.17 Cardiac power output (CPO) comparison from the three policies: without penalty terms (blue), with only power penalty term (orange), and with both drug and power penalty terms (red).

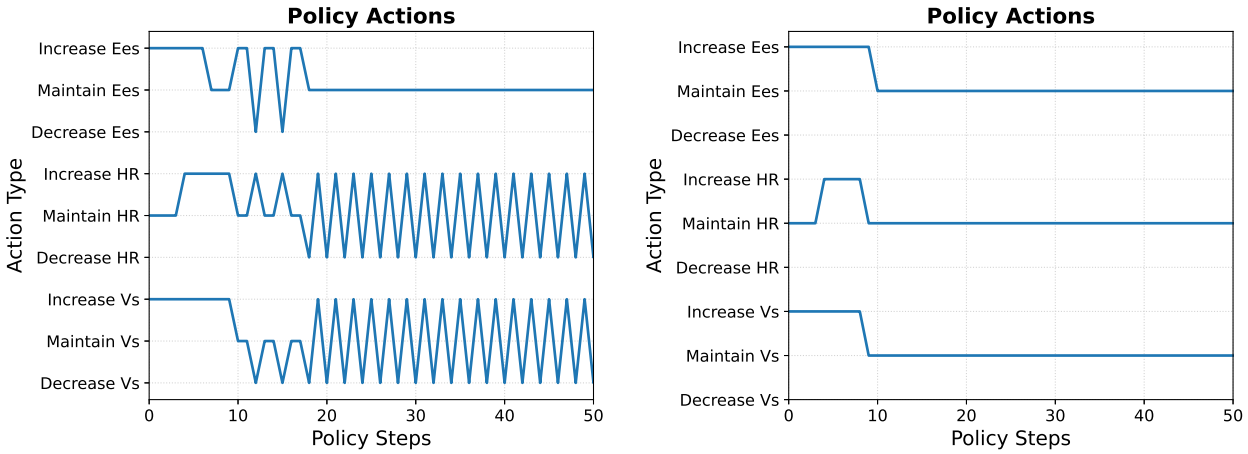


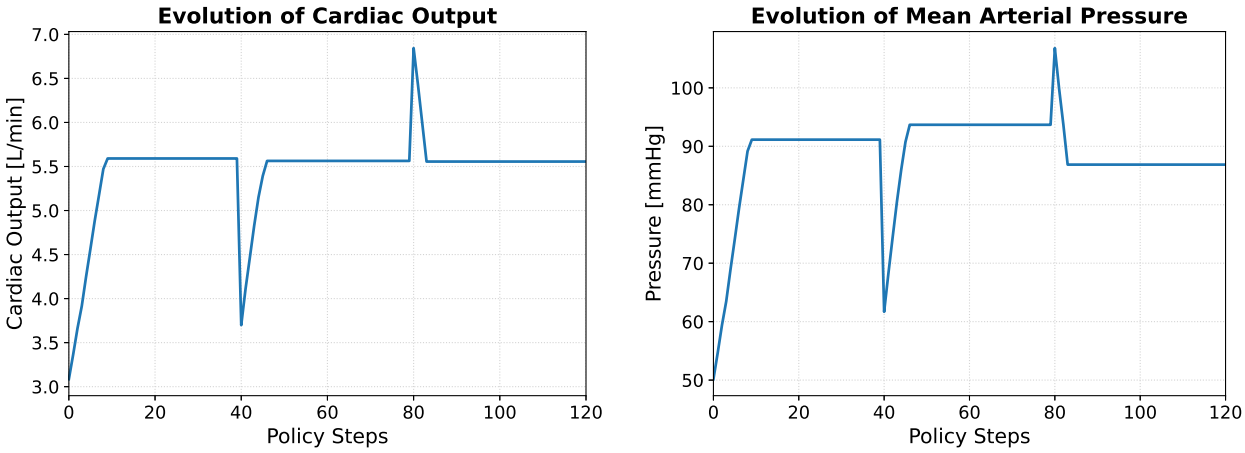
Figure 4.18 Actions from policies with power and with both penalty terms. a) Policy without penalty terms. b) Policy with both power and drug penalty terms.

4.3.4. Final Policy Assessment

A final test was created to assess the policy trained with DR and a reward that included both penalty terms plus the reward to cardiac output ($\mathcal{R}(s,a) = \mathcal{R}^{+}(s,a) + \mathcal{R}^{-}_{pp+dp}(s,a)$). In particular, the policy's performance was evaluated on a more complex environment with time-dependent dis-

turbances. As before, the simulated patient (environment) was initialized with parameters leading to cardiogenic shock. But after the policy had interacted with the environment, two of the parameters were altered, simulating a "relapse" and subsequently "healing". Concretely, initial conditions were set with baseline parameters, a low LV end-systolic elastance ($E_{es_{LV}} = 0.8 mmHg/mL$), and a high systemic venous resistance ($R_{v,s} = 0.03 mmHg \cdot s/mL$). Relapse was modeled with a disturbance after 40 policy steps and consisted of a: sudden reduction in LV end-systolic elastance ($E_{es_{LV}} = 1.0 mmHg/mL$) and increment of systemic venous resistance ($R_{v,s} = 0.05 mmHg \cdot s/mL$). Finally, the healing was modeled with a second disturbance at 80 policy steps, consisting of a: sudden reduction in systemic venous resistance to its healthy value ($R_{v,s} = 0.015 mmHg \cdot s/mL$). These disturbances are depicted in orange in Figure 4.20.

The evolution of cardiac output and mean arterial pressure are shown in Figure 4.19. From the initial conditions, the policy increases both cardiac output and mean arterial pressure to restore the patient from the decompensated state, achieving values of $5.6 L/min$ and $91 mmHg$, respectively. Once the disturbance simulating the patient's relapse appears, the policy reacts to it counteracting the effect and restoring the cardiac output and mean arterial pressure to target values. Finally, after introducing the disturbance that represents healing, the policy notices a high cardiac output and mean arterial pressure and reduces them, obtaining a final cardiac output close to $5.5 L/min$ and a mean arterial pressure of $88 mmHg$.



(a) **(b)**

Figure 4.19 Final policy performance with disturbances. Initial conditions: baseline parameters with low LV end-systolic elastance ($E_{es_{LV}} = 0.8 mmHg/mL$) and high systemic venous resistance ($R_{v,s} = 0.03 mmHg \cdot s/mL$). First disturbance at 40 policy steps: sudden reduction of LV end-systolic elastance ($E_{es_{LV}} = 1.0 mmHg/mL$) and increment of systemic venous resistance ($R_{v,s} = 0.05 mmHg \cdot s/mL$). Second disturbance at 80 policy steps: sudden reduction of systemic venous resistance to its healthy value ($R_{v,s} = 0.015 mmHg \cdot s/mL$). a) Cardiac output ($CO$). b) Mean arterial pressure ($\bar{P}_{a,s}$).

Looking carefully at the evolution of parameters changed by the policy (4.20), one can appreciate that in order to counteract the decompensated state with high systemic venous resistance ($R_{v,s}$)

and low LV end-systolic elastance ($E_{es_{LV}}$), the policy increases $E_{es_{LV}}$, HR and V_s. Specifically, to compensate for the initial conditions, the policy restores $E_{es_{LV}}$ to $2.4 mmHg/mL$, raises HR to $94 bpm$ and increases the Vs to 1025 mL. After the patient relapses, in order to counteract the increase of $R_{v,s}$ and decrease of $E_{es_{LV}}$, the policy raises the heart rate (HR= $113 bpm$) and stressed volume (V_s=1180mL) even further. Finally, when $R_{v,s}$ is restored to its healthy value, the policy reduces all three $E_{es_{LV}}$, HR and V_s to $1.42 mmHg/mL$, $102 bpm$ and $1090 mL$, respectively. This results in decreasing and maintaining the cardiac output close to $5.5 L/min$. Moreover, as shown in Figure 4.21, the policy takes actions only when necessary, remaining neutral when appropriate values of cardiac output are reached.

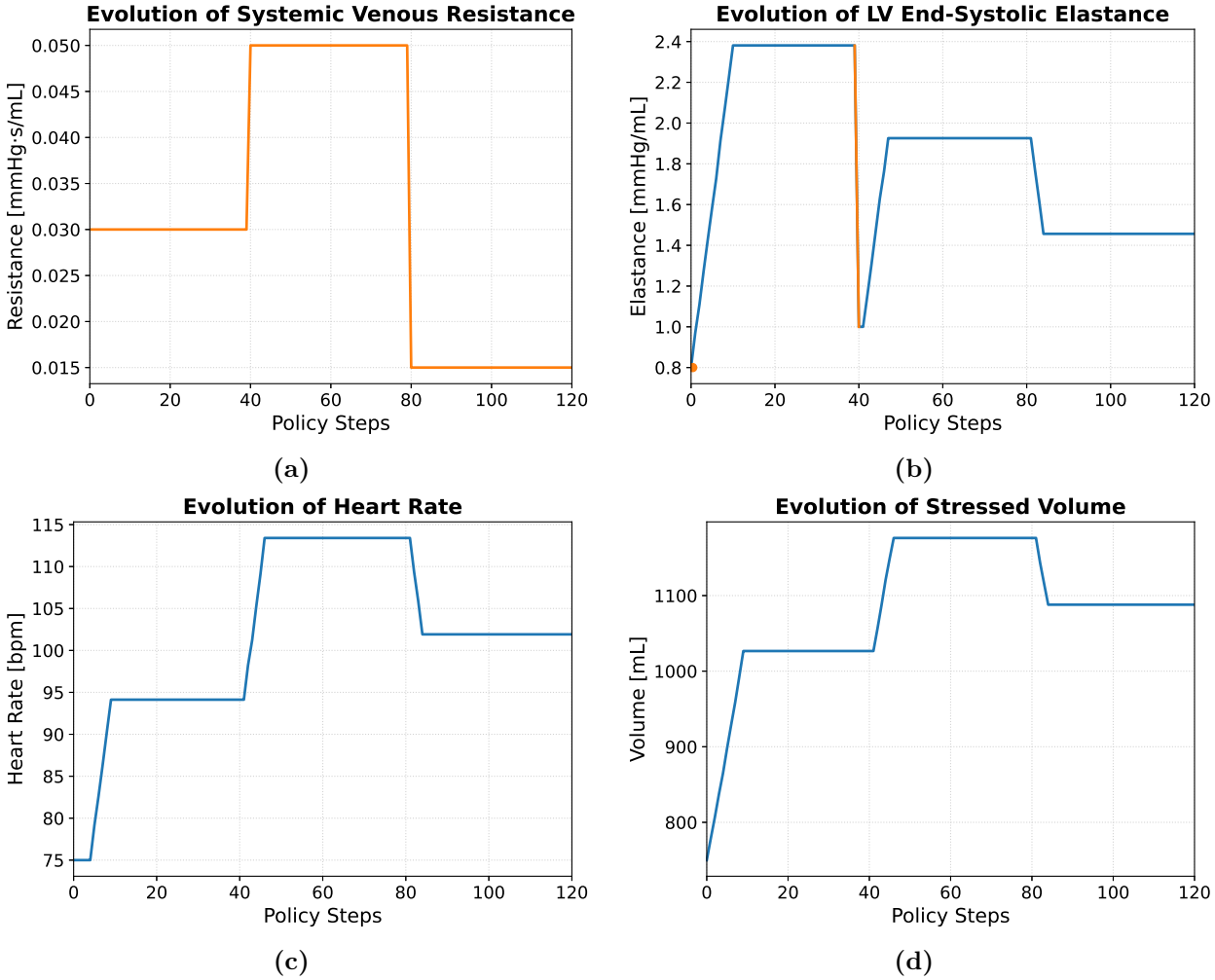


Figure 4.20 Evolution of changed parameters by policy actions plus simulated changes in the systemic venous resistance and LV end-systolic elastance. Changes performed by the policy in blue. Changes performed by the simulator to simulate the patient in orange. a) Systemic venous resistance ($R_{v,s}$). b) Left ventricular end-systolic elastance ($E_{es_{LV}}$). c) Heart rate (HR). d) Stressed volume (V_s).

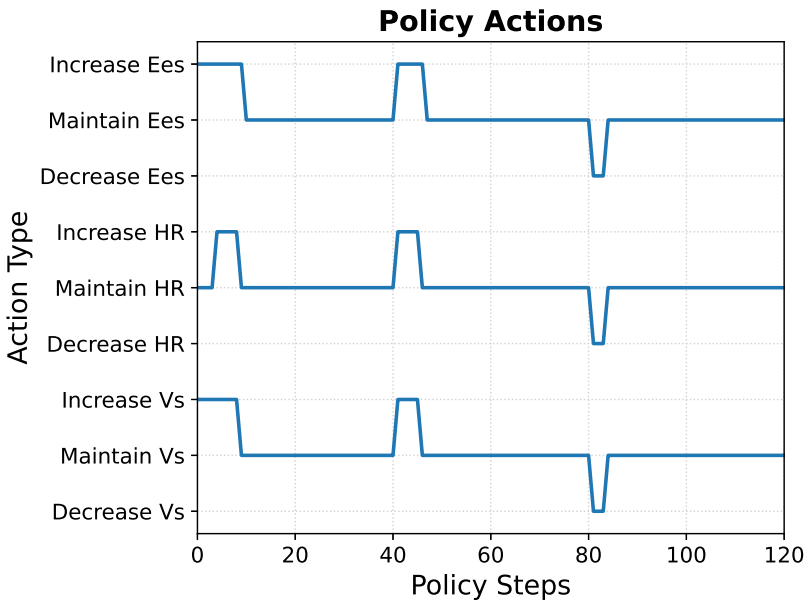


Figure 4.21 Actions from final policy.

After performing all the above experiments, it can be concluded that a policy trained with DR and considering a reward that included both drug and power penalty terms achieves a level of performance notoriously good. This policy can face environments (CV states) not seen before. Also, it can learn patterns that physicians follow, such as introducing intravenous fluids as a first action to rapidly increase cardiac output. Furthermore, this policy learns which actions are the most important, for instance, not changing *HR* until blood volume increases. Finally, such a policy knows the precise moments to make the actions, only when cardiac output is too low or too high. In future work, this findings must be tested with real patient data, where the policy will be assessed with records of real world environments.

5. PROJECT IMPACT AND PLANNING

In this chapter, the environmental and social impacts that this project may cause are considered. Furthermore, the project timeline, the project management methodoogy and the project costs breakdown are detailed.

5.1. Environmental Impact

The environmental impact of the present project is minimal, as its implementation does not contemplate any increment or reduction of consumption of any energy or material resources. In reference to the work performed and the consequent impact this might cause, it was basically based on documentation and code writing, simulations, training and optimizations using a computer. Thereby, electricity and computers needed to carry out the different parts of the project can be considered.

First, it should be noted that the electrical consumption was low. As estimated in Section 5.5, the power needed for the project was around 1 kW, which is marginal compared to the power that consumes the *Simches Building*, where the lab is located. Therefore, it can be said that the development of this project did not significantly change any electrical consumption.

On the other hand, deterioration of the electronic equipment (workstations and personal laptop) must be borne in mind. Once their useful life is finished, they become the so called e-waste. They must be treated individually, in agreement with the Regulation *310 CMR 30.000: Hazardous Waste Regulations* ordered by the Massachusetts Department of Environmental Protection of the Massachusetts Government. They can be recycled following the Responsible Recycling Practices (R2), the e-Stewards standards and the standard ISO 14001.

5.2. Social Impact

The work developed in this thesis constitutes a step forward towards the automation of a set of therapy decisions that physicians take during the treatment of their patients. Therefore, the implementation of these kind of methods to the ICU and CCU involves an important breakthrough to the patient care automation, allowing professionals to get assistance during their work.

Patient's health state could be predicted before medical actions were taken, thus, giving insight

to physicians on which might be the outcome obtained depending on the strategy implemented. Patient's health evolution could be tracked and analyzed to learn optimal therapy decisions. Therefore, certain trial and error methodologies when giving drugs may be avoided or at least, reduced. Furthermore, information about patients could be shared between professionals that work in different time shifts. This would end with quick and sometimes incomplete meetings where they exchange the patient's health state.

It could be also argued that optimal treatments would be learned from the best physicians or hospitals, and then, be deployed to other hospitals where resources and knowledge are yet limited. The best treatments may be universally achievable for all world population.

In short, the progress in healthcare automation may cause a huge impact that could be quantified in both economic savings (reducing medical resources and time), and social benefits, as those patients who are treated might take advantage of those optimal strategies. Moreover, the physical and mental well-being of professionals would be improved, as they would get help and reduce their stress in their working environment.

5.3. Project Timeline

The project timeline is illustrated with a Gantt diagram (Figure 5.1) in the next page. The whole project was carried out during a complete year, in fact, from April 2020 to April 2021. In total, this counts up to 51 weeks of work, 5 days a week, 8 hours per day (note that on August and December 2 weeks were subtracted, as they were taken for vacations). The main activities performed during the project are separated into reading, implementation and writing tasks.

As appreciated, the first months were primarily dedicated to understanding and documenting the EDW and BM databases, as well as the pipeline development. Concretely, the documentation task took from April to July, and the data curation and pipeline development from May to November. Simultaneously to the previous tasks, the project core started too. From June to July, the readings of cardiovascular system as well as the simulator implementation took place. Next, from July to November the activities related to the Reinforcement Learning algorithm were performed. First, reading reinforcement learning state of the art papers (July - September). Then, implementing the DQN algorithm embedded to the CV simulator (September - October). And finally, to end with this part of the project, developing a domain randomization strategy within the algorithm (November). Following, from December to January, strategies and tools for hyperparameter optimization were learned and implemented. Moreover, the task related to the system identification tool was executed from mid January to mid March. Also, from March to mid April, results from all the different parts of the project were obtained. And last but not least, during the period of February to April, all the work developed in this project was written.

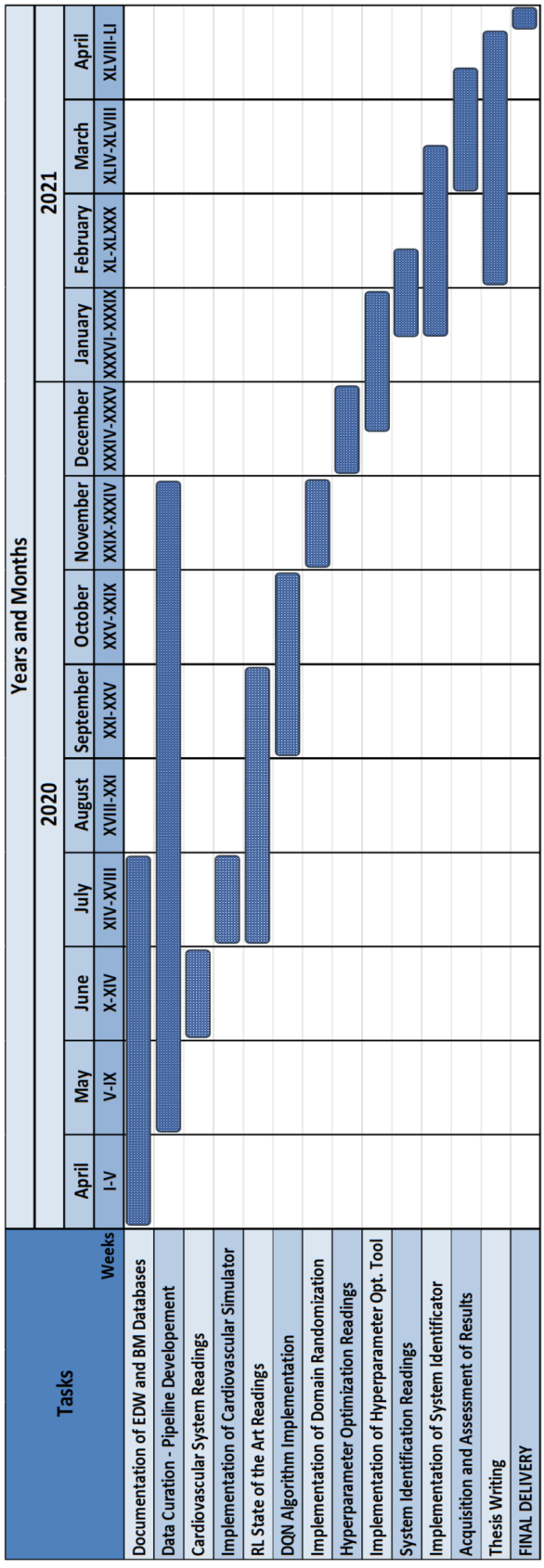


Figure 5.1 Gantt diagram of the project. Number of weeks of each month expressed in roman numeration. Tasks are mainly split in reading, implementation and writing activities.

5.4. Project Management Methodology

The methodology followed to organize and distribute the work over the whole year was based on *Agile*. The *Agile* methodology is a way to manage a project by breaking it up into several phases. It involves constant collaboration with all the group and continuous improvement at every stage. Group teams cycle through a process of planning, executing, and evaluating.

Concretely, the *Scrum* methodology was followed. GitHub served as the environment where the project was developed. With a canvas that contained the work to do, the work in progress and the work done, small tasks called issues were assigned to develop this project. Stand up meetings were carried out biweekely, and retro meetings were held once a month. A postdoctoral researched had the role of *scrum* manager in order to lead the team through the common goals. Moreover, weekely meetings with the lab team and principal investigators were scheduled.

Primary resources needed to carry out the explained methodology were the following. First, a lab space and a meeting room were used to held personal meetings and group work. Moreover, Zoom and Slack applications were provided to do videocalls and organize tasks virtually. Second, GitHub served as the platform to share and unify each student project, building a powerful machine learning environment with different tools. Finally, powerful workstations were bought to perform time and compute consuming tasks related to the project.

5.5. Project Costs

The economic cost of the project consists of four different aspects: the depreciation of the laboratory and personal computers, the cost of the Harvi simulator license, the cost of the supervisors' and student's working time, and finally, an estimation of the electrical energy consumed.

Depreciation of computers is calculated from its total price, their lifespan and the total time they were working. For this project, two workstations (a personal one and a shared one with lab members) as well as a personal laptop were used. A useful life of 8 years and 5 years for the workstations and laptop have been considered, respectively. Workstations and personal laptop are thought to be used 8 hours per day, 5 days a week and 48 weeks a year. Taking the 8 years, it is a total of 15360 hours of useful life for the workstations. On the other hand, considering the 5 years for the personal laptop, it is a total of 9600 hours of useful life.

The lab workstation was only employed when training and optimizing, which has estimated to be a total of 14 entire days, i.e., 336 hours. Regarding the personal workstation and laptop, both were used the whole workday simultaneously. As stated in the previous section (Section 5.3), the total worked time was composed by 8 hours per day, 5 days a week during 51 weeks. Hence, a total of

2040 hours within the project.

Harvi simulator license expires after one year. Taking the hours of a whole year (24 hours per day, 7 days per week, 52 weeks per year), the license lasts 8736 hours. Since the simulator is not depreciated, it has been assumed that its entire price is attributed to the project.

As for the costs related to the student and supervisors, the student worked the 2040 hours derived in the above paragraph. Also, during the task of developing the pipeline, other three students helped on doing so. Therefore, taking from May to November, those three students contributed on 224 hours each. On the other hand, the main supervisor, the director of this thesis, dedicated 2 hours per week in meetings as well as 15 hours of supervision of the thesis writing, which sums up to 117 hours. Moreover, another principal investigator with the role of coadvisor and a postdoctoral researcher also helped on the supervision of this thesis, about 50 hours each. As presented in [38], a graduate research assistant in Massachusetts makes 20 $/h in average. On the other hand, a postdoctoral researcher is paid 28 $/h in average. Finally, principal investigators are paid 66 $/h in average. Furthermore, it must be taken into account that the Massachusetts General Hospital charges with the 68% of all expenses in order to perform research on-site.

Finally, the cost of the electricity consumed has been estimated. Power of all equipment has been upper bounded by its respective power supply. The lab workstation has a power supply of 1600 W, the personal workstation a power supply of 650 W and finally, the personal laptop has a charger of 120 W. Furthermore, a light bulb of 40 W has been considered open all the time the student was working. Finally, the price of electricity in Massachusetts has been averaged during 2020 and taken constant, with a resulting value of 0.2 $/kWh.

Note that the variable cost of electricity in Table 5.2 is computed by multiplying the price of electricity by the total power consumed. As the lab workstation was used much less than the other computers and light bulb, a fraction of 336/2040 of its total power is taken. Thus, the total power can be calculated as:

$$P_{tot} = 1600 \cdot 336/2040 + 650 + 120 + 40 = 1073.53W.$$

The two workstations used were mounted in parts, Table 5.1 shows the summary of their price. Furthermore, in Table 5.2 the total cost of the project is presented. Each cost related to the factors described above is detailed. The total cost of the project is 114009.26 $ (94535.91 €).

		Personal Workstation		Lab Workstation	
PC Component	**Unit Cost [$/unit]**	**Units per Workstation**	**Workstation Cost [$]**	**Units per Workstation**	**Workstation Cost [$]**
GIGABYTE X570 UD AMD Ryzen 3000 (Motherboard)	159.99	1	159.99	1	159.99
EVGA GeForce RTX 2080 Ti XC ULTRA GAMING (GPU)	1625.00	0	0.00	4	6500.00
WD Red Pro WD6003FFBX 6TB (Hard Drive)	194.99	1	194.99	0	0.00
G.SKILL Trident Z Neo Series 32 GB (2 x 16GB) (RAM)	169.99	1	169.99	0	0.00
G.SKILL Trident Z Neo Series 128 GB (4 x 32GB) (RAM)	634.99	0	0.00	1	634.99
AMD RYZEN 7 2700 8-Core 3.2 GHz (4.1 GHz Max) (CPU)	278.99	1	278.99	0	0.00
AMD RYZEN 9 3950 16-Core 3.5 GHz (4.7 GHz Max) (CPU)	724.99	0	0.00	1	724.99
SAMSUNG 970 EVO PLUS M.2 2280 2TB (Solid State Drive)	319.99	1	319.99	1	319.99
EVGA SuperNOVA 650 GA, 80 Plus Gold 650W (Power Supply)	79.00	1	79.00	0	0.00
EVGA 220-P2-1600-X1 1600W ATX12V (Power Supply)	500.56	0	0.00	1	500.56
Logitech MK120 Wired USB Keyboard & Mouse	18.23	1	18.23	1	18.23
Dell UltraSharp U2417H 24" IPS 1080P (Screen)	231.00	1	231.00	1	231.00
TOTAL COST			1452.18		9089.75

Table 5.1 Calculation of workstations cost from each component bought. Price of each component is shown on the middle left column.

Cost factor	Fixed cost [$]	Life expectancy [years]	Variable cost [$/h]	Time referred to project [h]	Cost related to project [$]
Personal Workstation	1452.18	8	0.09	2040	192.87
Lab Workstation	9089.75	8	0.59	336	198.84
Personal Laptop	1463.30	5	0.15	2040	310.95
***Harvi* License**	60.00	1	0.01	8736	60.00
PI Supervisors	-	-	66.00	167	11022.00
PhD Supervisor	-	-	28.00	50	1400.00
Students	-	-	20.00	2712	54240.00
Electrical energy	-	-	0.21	2040	438.00
Project cost					67862.66
MGH cost				68% Project cost	46146.61
TOTAL COST					114009.26

Table 5.2 Calculation of the final project cost. Variable costs of computers and licenses are obtained from dividing their fixed cost by their life expectancy in hours. Variable cost of electrical energy is found from its price multiplied by the power consumption of computers and light.

CONCLUSIONS

Reinforcement learning (RL) for decision support has recently emerged for applications into healthcare. For instance, managing invasive mechanical ventilation, learning optimal treatment strategies for sepsis or treating cancer. This thesis focuses on the application of RL in supporting therapy strategies in cardiogenic shock (CS) patients due to decompensated heart failure.

In this study, a policy was trained in simulated CS patients in order to learn simplified tailored therapy decisions that recovered the patient's decompensated state. First, the Burkhoff and Tyberg cardiovascular (CV) hemodynamics model was implemented to simulate CS patients. This model served as environment in which the policy interacted and learned therapy strategies.

Second, a system identification tool was developed to estimate nine CV model parameters from real data. In particular, a dataset consisting of 1372 patients who underwent CABG surgery was created. From it, 776341 cardiac cycles with relevant hemodynamic information were extracted and used to first, train an autoencoder, and then, use the encoder to estimate the aforementioned parameters. Results showed that the means of the parameters distribution were strikingly similar to literature parameters.

Finally, the deep Q-network (DQN) algorithm was used to train policies on simulated CS patients. Robustness of policies was reinforced by introducing domain randomization during training. Ranges of domain randomization were determined based on the nine estimated CV parameter distributions. Then, an assessment of action importance was carried out by limiting policies to a subset of the whole action space. It was evidenced that actions changing heart rate were less important for restoring cardiac output. Moreover, certain strategies implemented by physicians were also appreciated: the policy learned that introducing intravenous fluids to the patient to increase their blood volume is a relevant strategy to improve cardiac output. On the other hand, evaluation of different reward functions was performed by including different penalty terms. This demonstrated that penalties related to cardiac power consumption and drug administration were necessary to obtain better treatment strategies. Such strategies only included the necessary actions while avoiding undesirable patient states like hypertension.

Having presented the obtained outcomes, some limitations of this work should be addressed. Regarding the system identification tool, the ability of the decoder to reproduce simulated pressures was limited to its training domain. The fact that negative CV parameters were estimated, pointed out the lack of generalization of the decoder outside from its training range.

Related to the RL framework, the simplification of therapy actions directly affecting the CV parameters reduced high variability in patient-to-patient drug administrations. Although stochastic actions were introduced, a drug administration model sensitive to changes in parameters must be developed to account for the interpatient variability that physicians face. Such a model would allow evaluating whether a policy would be robust enough to face patient-to-patient drug effects when making therapy decisions. In addition, it was demonstrated that domain randomization provides robustness in front of unseen and unexpected environments. However, relying on only simulated data limits the assessment of the policy performance in a real scenario.

Future work contemplates first an improvement of the autoencoder. Specifically, retraining the decoder to generalize and achieve better similarity with the simulator. Ideally, it would be desirable to substitute the decoder with the actual simulator. However, this one should be implemented in such a way that could be parallelizable over a GPU and suitable to backpropagate the loss error for neural network training. Moreover, switching to another RL algorithm that accepts continuous actions is also considered. This change, together with a model of the effect of drug administration on CV parameters, would provide a more complex environment that a policy should face to. Finally, a dataset of patient-outcome historical data is sought to be generated. This dataset would serve as a tool to retrain and refine the policy with real physician-to-patient causality relationships, as well as to evaluate the policy performance.

The fundamental value of this thesis lies in the implementation and evaluation of a policy that learns tailored therapy decisions from hemodynamic data. Although these strategies have been studied in simulation, the long term goal contemplates the use of clinical data to assess its potential in a real world scenario. A future implementation is envisioned to work on patient's real time data. By learning the patient's CV physiology, a policy would recommend therapy decisions to improve and counteract decompensated states that lead to CS.

ACKNOWLEDGMENTS

Firstly, I would like to deeply thank my advisor, Dr. Nicholas Houstis, for making of this past year, difficult for all of us in many ways, an unforgettable experience. I truly appreciate his great advice, guidance and mentorship over the past 12 months. This project would not have been possible without his devotion, perseverance and incommensurable support. He not only guided me through this thesis but, with his passion, he also made learning about cardiovascular physiology enjoyable.

I would also like to express my sincere gratitude to Dr. Aaron Aguirre for his constant highly valuable suggestions as well as helping me see the big picture and clinical motivations behind this project. I will always be grateful for both Aaron and Nick for offering us the opportunity to improve our career experiences next to them.

I want to thank Erik Reinertsen for his continuous supervision and involvement in the project to create a cohesive team and an enriching and amiable working environment. Working together with Eric, Raimon, Ridwan, Steven and him has been a huge pleasure. I also want to express a word of gratitude to all Aguirre Lab members, for both showing interest on my project and contributing with any valuable inputs.

Moreover, I would like to thank Dr. Cecilio Angulo for his advice and supervision of my work. Last, but not least, I want to thank my family and friends for always being there for me and, although being apart, supporting me on my new adventure in the U.S.

REFERENCES

[1] Burkhoff, D., and Tyberg, J. V. Why does pulmonary venous pressure rise after onset of LV dysfunction: a theoretical analysis. *Am Journal Physiology 256* (August 1993), 1819–28. DOI: 10.1152/ajpheart.1993.265.5.H1819.

[2] DORNHORST, A. C., HOWARD, P., and LEATHART, G. L. Effects of dobutamine on hemodynamics and left ventricular performance after cardiopulmonary bypass in cardiac surgical patients. *Circulation 6* (October 1952), 553–558. DOI: 10.1161/01.CIR.6.4.553.

[3] Doshi, D., and Burkhoff, D. Cardiovascular Simulation of Heart Failure Pathophysiology and Therapeutics. *Journal of Cardiac Failure 22(4)* (April 2016), 303–11. DOI: 10.1016/j.cardfail.2015.12.012.

[4] Foulon, P., and Backer, D. D. The hemodynamic effects of norepinephrine: far more than an increase in blood pressure! *Annals of Translational Medicine 6(Suppl 1)* (November 2018), S25. DOI: 10.21037/atm.2018.09.27.

[5] Frazier, P. I. A Tutorial on Bayesian Optimization. *arXiv* (July 2018). eprint: 1807.02811.

[6] Fu, J., Levine, S., and Abbeel, P. One-shot learning of manipulation skills with online dynamics adaptation and neural network priors. *RSJ International Conference on Intelligent Robots and Systems (IROS)* (October 2016). DOI: 10.1109/IROS.2016.7759592.

[7] Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016, ch. 6. http://www.deeplearningbook.org.

[8] Herring, N., and Paterson, D. J. *Levick's Introduction to Cardiovascular Physiology*. CRC Press, Boca Raton, FL, 2018, ch. 1 and 2. Sixth Edition.

[9] Hoppensteadt, F. C., and Peskin, C. S. *Modeling and Simulation in Medicine and the Life Sciences*. Springer, Arizona State University and New York University, 2004, ch. 1. Second Edition.

[10] Irpan, A., Rao, K., Bousmalis, K., Harris, C., Ibarz, J., and Levine, S. Off-Policy Evaluation via Off-Policy Classification. *arXiv* (November 2019). eprint: 1906.01624.

[11] Jeter, R., Josef, C., Shashikumar, S., and Nemati, S. Does the "Artificial Intelligence Clinician" learn optimal treatment strategies for sepsis in intensive care? *arXiv* (February 2019). eprint: 1902.03271.

[12] Keener, J., and Sneyd, J. *Mathematical Physiology II: Systems Physiology.* Springer, University of Utah and University of Auckland, 2009, ch. 11. Second Edition.

[13] Kelsey, R., Botello, M., Millard, B., and Zimmerman, J. An online heart simulator for augmenting first-year medical and dental education. *Proc AMIA Symp* (2002), 370–374. PMID: 12463849.

[14] Kitai, T., and Xanthopoulos, A. Contemporary Management of Acute Decompensated Heart Failure and Cardiogenic Shock. *Heart Fail Clin 16(2)* (April 2020), 221–230. DOI: 10.1016/j.hfc.2019.12.005.

[15] Klabunde, R. E. Cardiovascular physiology concepts. https://www.cvphysiology.com/Cardiac%20Function/CF025. [Online; accessed: 1-March-2021].

[16] Klabunde, R. E. Cardiovascular Pharmacology Concepts. https://www.cvpharmacology.com/diuretic/diuretics, November 2017. [Online; accessed: 30-March-2021].

[17] Komorowski, M., Celi, L. A., Badawi, O., Gordon, A. C., and Faisal, A. A. The Artificial Intelligence Clinician learns optimal treatment strategies for sepsis in intensive care. *Nature Medicine 24* (October 2018), 1716–1720. DOI: 10.1038/s41591-018-0213-5.

[18] Liu, H., Liang, F., Wong, J., Fujiwara, T., Ye, W., iti Tsubota, K., and Sugawara, M. Multi-scale modeling of hemodynamics in the cardiovascular system. *Acta Mech. Sin 31* (August 2015), 446–464. DOI: 10.1007/s10409-015-0416-7.

[19] Mangini, S., Pires, P. V., Braga, F. G. M., and Bacal, F. Decompensated Heart Failure. *Einstein (Sao Paulo) 11(3)* (September 2013), 383–91. DOI: 10.1590/s1679-45082013000300022.

[20] Mehta, R. H., Grab, J. D., O'Brien, S. M., Glower, D. D., Haan, C. K., Gammie, J. S., and Peterson, E. D. Clinical Characteristics and In-Hospital Outcomes of Patients With Cardiogenic Shock Undergoing Coronary Artery Bypass Surgery. *Med Eng Phys 117(7)* (February 2008), 876–885. DOI: 10.1161/CIRCULATIONAHA.107.728147.

[21] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing Atari with Deep Reinforcement Learning. *NIPS Deep Learning Workshop* (December 2013). eprint: 1312.5602.

[22] OpenAI, Akkaya, I., Andrychowicz, M., Chociej, M., Litwin, M., McGrew, B., Petron, A., Paino, A., Plappert, M., Powell, G., Ribas, R., Schneider, J., Tezak, N., Tworek, J., Welinder, P., Weng, L., Yuan, Q., Zaremba, W., and Zhang, L. Solving Rubik's Cube with a Robot Hand. *arXiv* (October 2019). eprint: 1910.07113.

[23] Pironet, A., Dauby, P. C., Chase, J. G., Docherty, P. D., Revie, J. A., and 2, T. D. The hemodynamic effects of norepinephrine: far more than an increase in blood pressure! *Med Eng Phys 38(5)* (May 2016), 433–41. DOI: 10.1016/j.medengphy.2016.02.005.

[24] Prasad, N., Cheng, L.-F., Chivers, C., Draugelis, M., and Engelhardt, B. E. A Reinforcement Learning Approach to Weaning of Mechanical Ventilation in Intensive Care Units. *arXiv* (April 2017). eprint: 1704.06300.

[25] Romson, J. L., Leung, J. M., Bellows, W. H., Bronstein, M., Keith, F., Moores, W., Flachsbart, K., Richter, R., Pastor, D., and Fisher, D. M. Respiratory Variations in Blood Pressure. *Anesthesiology 91(5)* (November 1999), 1318–28. DOI: 10.1097/00000542-199911000-00024.

[26] Santamore, W. P., and Burkhoff, D. Hemodynamic consequences of ventricular interaction as assessed by model analysis. *Anesthesiology 260* (January 1991), 146–57. DOI: 10.1152/ajpheart.1991.260.1.H146.

[27] Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., and et al. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature 588* (December 2020). DOI: 10.1038/s41586-020-03051-4.

[28] Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., and Hassabis, D. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science 362* (Dcember 2018), 1140–1144. DOI: 10.1126/science.aar6404.

[29] Stevenson, L. W. Tailored therapy to hemodynamic goals for advanced heart failure. *European Journal of Heart Failure 1* (August 1999), 251–257. DOI: 10.1016/s1388-9842(99)00015-x.

[30] Sutton, R. S., and Barto, A. G. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge MA and London England, 2015. Second Edition.

[31] Urbicha, M., Globe, G., Pantiri, K., Heisen, M., Bennison, C., Wirtz, H. S., and Tanna, G. L. D. A Systematic Review of Medical Costs Associated with Heart Failure in the USA (2014–2020). *Pharmacoeconomics 38* (August 2020), 1219–1236. DOI: 10.1007/s40273-020-00952-0.

[32] Weng, L. Domain Randomization for Sim2Real Transfer. https://lilianweng.github.io/lil-log/2019/05/05/domain-randomization.html, May 2019. [Online; accessed: 1-April-2021].

[33] Yu, C., Liu, J., and Zhao, H. Inverse reinforcement learning for intelligent mechanical ventilation and sedative dosing in intensive care units. *BMC Medical Informatics and Decision Making 19* (April 2019), 1716–1720. DOI: 10.1186/s12911-019-0763-6.

[34] Yu, W., Tan, J., Karen Liu, C., and Turk, G. Preparing for the Unknown: Learning a Universal Policy with Online System Identification. *Robotics: Science and Systems XIII* (July 2017). DOI: 10.15607/rss.2017.xiii.048.

[35] Zhang, Y., Barocas, V. H., Berceli, S. A., Clancy, C. E., Eckmann, D. M., Garbey, M., Kassab, G. S., Lochner, D. R., McCulloch, A. D., Tran-Son-Tay, R., and Trayanova, N. A. Multi-scale Modeling of the Cardiovascular System: Disease Development, Progression, and Clinical Intervention. *Annals of Biomedical Engineering 44(9)* (September 2016), 2642–2660. DOI: 10.1007/s10439-016-1628-0.

[36] Zhao, Y., Kosorok, M. R., and Zeng, D. Reinforcement learning design for cancer clinical trials. *Stat Med 28* (November 2009), 3294–315. DOI: 10.1002/sim.3720.

[37] Zhou, S., Xu, L., Hao, L., Xiao, H., Yao, Y., Qi, L., and Yao, Y. A review on low-dimensional physics-based models of systemic arteries: application to estimation of central aortic pressure. *BioMedical Engineering OnLine 18* (April 2019), 41. DOI: 10.1007/s10409-015-0416-7.

[38] ZipRecruiter. Real salaries from real employers. https://www.ziprecruiter.com/Salaries. [Online; accessed: 30-March-2021].