UNIVERSITY
OF OSLO

Master's thesis

Pathological Electrical Wave Simulation in the Heart Using Physics Informed Neural Networks

Adam Jakobsen

Computational Science: Physics

60 ECTS study points

Department of Physics

Faculty of Mathematics and Natural Sciences

Autumn 2024

Adam Jakobsen

Pathological Electrical Wave Simulation in the Heart Using Physics Informed Neural Networks

Supervisors:

Gabriel Balaban, Molly Maleckar, Vajira Tambawita,

Thus Nguyen and Morten Hjorth-Jensen

Abstract

Cardiovascular disease is the leading cause of mortality globally, with a staggering 18 million deaths every year. Arrhythmia, one of the numerous heart conditions contributing to this significant mortality rate, presents considerable challenges in terms of both diagnosis and treatment. Computational models of cardiac electrophysiology (EP) can provide valuable insights for the understanding and treatment of arrhythmias and other heart-related disorders. Specifically, personalized cardiac EP modeling has shown significant promise in the detection and management of arrhythmias. However, the substantial computational requirements of these models pose a notable challenge in their integration into clinical practice. In contrast, neural networks, once trained, can offer predictions within a very short timeframe. In particular, Physics-Informed Neural Networks (PINNs) can combine the theoretical knowledge of a physical system with data, presenting a promising method for personalized electrophysiology simulations.

We have developed a novel PINN model that utilizes spatial and temporal coordinates along with local conductivities as input parameters to predict the spatio-temporal evolution of action potentials. 2D magnetic resonance images (MRI) were used to establish a computational domain and to infer local conductivities.

Our study demonstrates the ability of PINNs to precisely recreate action potential dynamics from sparse *in-silico* voltage data. Importantly, we demonstrate that trained PINNs are able to interpolate and extrapolate with great accuracy using local conductivity inputs, even beyond the scope of the training data, and are able to generate predictions in a remarkably short time span. This work represents a significant advancement towards the clinical application of personalized cardiac electrophysiology models as well as a step towards general-purpose PINN models that can be used in varying scenarios without needing retraining or fine-tuning.

Acknowledgements

I would like to extend my heartfelt gratitude to my thesis supervisors, Gabriel Balaban, Molly Maleckar, Vajira Tambawita, Thu Nguyen, and Morten Hjort-Jensen, whose guidance and support have been indispensable in the completion of this thesis. Thank you for helping me prepare for new academic experiences, such as conferences and publishing papers. You have all made this thesis work a very enjoyable experience, and I couldn't have wished for a better supervision team.

I want to thank Gabriel Balaban and Molly Maleckar for providing excellent supervision and mentorship. Your support and encouragement has given me the confidence to push my boundaries and achieve more than I thought possible at this stage in my academic career.

Thank you to Thu Nguyen and Vajira Tambawita for your guidance and providing valuable feedback whenever needed.

I am also thankful to Morten Hjorth-Jensen, Tone Skramstad and the rest of the people at CCSE for providing an excellent environment for both academic pursuits and social bonding.

Additionally, I would like to extend my gratitude to Simula Research Laboratory for offering an outstanding work environment with wonderful people, a great cafeteria, and indispensable complementary coffee.

I also want to thank my family and friends, whose encouragement and support have been invaluable throughout this journey. A special thanks goes to my parents, Latifa Jakobsen and Per Oskar Jakobsen, for always believing in me, encouraging me, and providing financial support in times of need. Lastly, but not least, I want to thank my partner, Leah Hansen, for never-ending emotional support throughout and for helping me with proof reading my acknowledgments.

Contents

Acronyms

AP Action Potential

AU Arbitrary Units

BC Boundary Conditions

EP Electrophysiology

FEM Finite Elements Method

IQR Interquartile Range

RK4 4th order Runge-Kutta

MRI Magnetic Resonance Imaging

NN Neural Network

PINN Physics-Informed Neural Network

TP Transmembrane Potential

PDE Partial Differential Equation

ODE Ordinary Differential Equation

RA Right Atrium

RV Right Ventricle

LA Left Atrium

LV Left Ventricle

AD Automatic Differentiation

ECG Electrocardiogram

NN Neural Network

List of symbols and variables

β_1, β_2	Decay rates for the first and second moment estimates in the Adam optimizer
ϵ	Parameter used in the Aliev-Panfilov ionic model
η	Learning rate
λ	Constant used to simplify bidomain equations to monodomain
λ_d	Weight for the data loss term
λ_{BC}	Weight for the boundary conditions loss term
λ_{IC}	Weight for the initial conditions loss term
λ_{ODE}	Weight for the ODE loss term
λ_{PDE}	Weight for the PDE loss term
$\mathcal{L}$	Hybrid loss function
$\mathcal{L}_{BC}$	Loss term ensuring boundary conditions
$\mathcal{L}_{data}$	Loss term ensuring agreement with voltage-data
$\mathcal{L}_{IC}$	Loss term ensuring initial conditions
$\mathcal{L}_{ODE}$	Loss term ensuring the ODE is satisfied
$\mathcal{L}_{PDE}$	Loss term ensuring the PDE is satisfied
$\mathcal{L}_V$	Loss term for the voltage component

$\mathcal{L}_W$	Loss term for the recovery variable component
μ_1,μ_2	Parameters used in the Aliev-Panfilov ionic model
Σ	Conductivity tensor
σ	Standard deviation of noise
Σ_e	Conductivity tensor for extracellular domain
Σ_i	Conductivity tensor for intracellular domain
Σ_m	Harmonic mean tensor
τ	Time in temporal units
θ	Parameters optimized in training
φ	Scalar potential
φ_e	Extracellular potential
φ_i	Intracellular potential
a	Parameter used in the Aliev-Panfilov ionic model
A_m	Surface to volume ratio of the cell membrane
B	Magnetic field
b	Parameter used in the Aliev-Panfilov ionic model
C_m	Membrane capacitance
D	Components of the spatially varying local diffusion tensor
E	Electric field
g_{el}	Longitudinal extracellular conductivity
g_{et}	Transverse extracellular conductivity
g_{il}	Longitudinal intracellular conductivity

g_{it} Transverse intracellular conductivity

I_m Transmembrane current per unit area

I_{ion} Ionic current

J Current density

k Parameter used in the Aliev-Panfilov ionic model

k_1, k_2, k_3, k_4 Intermediate variables used in Runge-Kutta integration scheme

N Number of spatio-temporal points for training

N_c Number of collocation points

n_s Number of spatial locations sampled

N_{BC} Number of boundary condition points

N_{IC} Number of initial condition points

RL^1 Relative L^1 error

RL^2 Relative L^2 error

V Adimensional potential

V_m Transmembrane potential

W Recovery variable

Chapter 1

Introduction

1.1 Motivation

Cardiovascular disease is the leading cause of death worldwide. According to the World Health Organization (WHO), an estimated 17.9 million people die each year due to cardiovascular disease [23]. Arrhythmia, which is characterized by an irregular heart rhythm, is particularly concerning, as it can cause sudden cardiac arrest, often leading to sudden death without warning.

Computer simulations have been an integral tool in cardiovascular research for several decades, helping to elucidate complex cardiac mechanisms alongside experimental and clinical studies. In 1952, Hodgkin and Huxley laid the groundwork for computational models of excitable tissue [14], although it soon became apparent that significant advancements in both algorithm and computational power would be necessary in order to simulate the electrical activity of these tissues. By 1960, Denis Noble had harnessed the power of the University College London Ferranti Mercury computer to simulate the action potential using a modified version of the Hodgkin-Huxley equations [21]. This pioneering work demonstrated the potential of computational approaches to complement and extend experimental and clinical research in cardiac electrophysiology.

Today, in light of decades of computational advances, computer simulations can help to understand the mechanisms behind cardiac arrhythmias and to design personalized treatment strategies based on simulations specific to each

patient [32]. For example, Pop et al. [26] employed models using magnetic resonance imaging (MRI) data from pig hearts and showed good correspondence of computer simulations with experimental heart voltage data. This method accurately predicted ventricular tachycardia [6], a form of arrhythmia, as well as *in-vivo* electrophysiology (EP) studies.

EP simulations are typically performed by solving a system of reaction-diffusion equations [24] and provide detailed information about the local electric properties of the heart. However, the steep electric potential gradients that occur in the heart require EP simulations to have high spatial and temporal resolution to achieve accurate results. This requirement makes traditional EP simulations computationally expensive and time-consuming, limiting their clinical applications. In contrast, neural networks, once trained, have been shown to provide accurate predictions in a fraction of a second compared to the 12+ hours required by traditional simulation methods [38]. Recently, Physics-Informed Neural Networks(PINNs), popularized by Raissi et al.[27], has surfaced as a powerful way to solve ordinary differential equations(ODEs) and partial differential equations (PDEs). These neural networks take advantage of the knowledge of the physical system, allowing for a flexible and efficient framework that can adapt to a wide range of physical phenomena without the need for extensive reconfiguration and present a promising alternative to traditional methods such as Finite Element Methods(FEM). Unlike FEM, which relies on mesh-based discretizations and can become progressively more computationally intensive with increasing problem complexity, PINNs offer a mesh-free solution, promising fast evaluation times once trained [10].

1.2 State of the (He)art

The field of PINN-based computational cardiac electrophysiology is at an emerging stage. As of now, the literature on this topic is sparse, consisting of five studies, reflecting the novelty of this research area. The work of Sahli et al. [29] in 2020 marks the first contribution to this young field. This particular study provided

a solution for activation mapping in cardiac EP using PINNs. The authors were able to predict conduction velocities (CV), a measure of how fast electrical signals propagate through the tissue, as well as activation times. Succeedingly, in 2022, Nazarov et al. [20] mainly concentrated on a phenomenological model, while Martin et al. [13] employed a canine atrial model [1]. Both of these were conducted using structured one-dimensional and two-dimensional computational domains, without using anatomically correct domains for training PINNs. Notably, Martin et al. [13] have demonstrated the effectiveness of PINNs even in cases with noisy and sparse data, as well as in solving inverse problems to estimate tissue properties.

Building upon these foundations, recent studies have further explored the application of PINNs in cardiac electrophysiology. Xie and Yao [36] proposed a physics-constrained deep learning approach for inverse ECG modeling, integrating the same cardiac electrophysiological model as [13] to model spatio-temporal cardiac electrodynamics from sparse simulated sensor observations. Their method demonstrated significant improvements in handling measurement noise and model uncertainty in inverse electrocardiogram problems. In another study, Xie and Yao [37] presented a physics-constrained deep active learning framework for spatio-temporal modeling of cardiac electrodynamics by employing a non-MRI-derived 3D geometry of the heart. By using dropout layers, the authors were able to quantify model uncertainties and re-create action potential from sparse measurements in the presence of noise.

While not being a physics-informed model, a recent publication in the HeartRhythm Journal has also expanded upon prior research by employing operator-learning neural networks and patient-specific three-dimensional geometries. Nevertheless, this study concentrated solely on the activation phase of the heart, without addressing the complete depolarization (activation) and repolarization processes, as in the other studies mentioned [38].

Although significant progress has been made in the application of deep learning models for EP simulations, there is a clear need for further research to integrate more complex theoretical models of EP dynamics and patient-specific data, such as MRI of the heart. Leveraging MRI data to construct computational domains

that mirror the unique anatomical features of an individual's heart is an important step in personalized healthcare, as every patient's heart is unique. Moreover, MRIs provide insights into intricate aspects of heart tissue, including fiber direction and areas of scarring, which influence the heart's electrical conduction. Incorporating MRIs into simulations helps in creating more accurate models of the heart and in bridging the gap between clinical practices and theoretical models by ensuring that the simulations are grounded in real, observable phenomena. This level of detail, combined with the predictive capabilities of deep-learning, can thus potentially make the models more relevant and applicable to clinical settings.

1.3 Objective

The overarching objective of this thesis is to explore the application of PINNs in cardiac EP simulations by addressing the following goals and hypotheses.

Goals:

1. Evaluate the efficacy of PINNs in simulating electric waves in the heart as compared to classical methods.

2. Investigate the application of PINNs in the simulation of electric waves in 2D MRI-slice-based heart geometries with scar location data.

3. Investigate the ability of PINNs to generalize to scenarios with varying electrical conductivities.

Hypotheses:

1. The use of PINNs will result in faster simulations of electric waves in 2D MRI-slice-based heart geometries compared to traditional methods.

2. PINNs can accurately simulate electrophysiological wave propagation in 2D MRI-slice-based heart geometries, including the incorporation of image informed scar tissue data.

3. A trained PINN can effectively extrapolate PDE parameters, such as conductivities, beyond the range of training data, providing accurate and reliable predictions.

In summary, these goals and hypotheses aim to examine the efficacy, adaptability, and predictive strength of PINNs in cardiac electrophysiology simulations. This thesis pushes the boundaries of cardiac PINN techniques by incorporating scars from late gadolinium-enhanced MRI, advancing toward patient-specific models. Additionally, the conductivity extrapolations represent a step toward general-purpose PINN models that can be used in varying scenarios without needing retraining or fine-tuning.

Chapter 2

Background

In this chapter, we provide the necessary theoretical and physiological background in cardiac electrophysiology and the application of PINNs in this context. We begin with an overview of the human heart's anatomy and the fundamental principles of cardiac electrophysiology, including the cellular mechanisms of excitability and the propagation of electrical signals through the heart tissue. We then dive into the mathematical modeling of cardiac electrophysiology, detailing the ionic models and the monodomain model classically used to describe the electrical activity of the heart. Following this, we introduce the concept of PINNs, explaining their architecture, the role of automatic differentiation, and how they are utilized to solve differential equations governing cardiac electrophysiology.

2.1 Cardiac Electrophysiology

We begin with an in-depth exploration of the physiological and theoretical foundations of cardiac electrophysiology. The structure of the human heart is described first, highlighting the roles of the various chambers and the pathways of blood flow. Next, we examine the cellular mechanisms that enable cardiac muscle cells, or cardiomyocytes, to generate and propagate electrical signals, focusing on the concepts of membrane potential, depolarization, and repolarization. This discussion transitions into the mathematical representation of these processes through ionic models, particularly the Aliev-Panfilov model, which captures the

dynamics of action potentials. Finally, we introduce the monodomain model, a volume-averaged approach to modeling the heart's electrical activity. This model simplifies computational complexity while maintaining accuracy in simulating the propagation of electrical waves through cardiac tissue.

2.1.1 The human heart

The human cardiovascular system is powered by the heart, which acts as a pump to circulate blood throughout the body. The human heart is an amazing organ, pumping an incredible 7600 liters of blood each day without pause, a feat that no other muscle in the body can match. As illustrated in Figure 2.1, the heart is made up of four chambers: the right atrium (RA), the right ventricle (RV), the left atrium (LA), and the left ventricle (LV). The heart can be seen as two pumps that work in sequence, with the RA and RV carrying deoxygenated blood from the veins to the lungs through the pulmonary arteries. The blood is then oxygenated in the lungs before being sent through the pulmonary veins to the LA and then the LV. Finally, oxygenated blood is pumped out of the heart through the aorta [15].

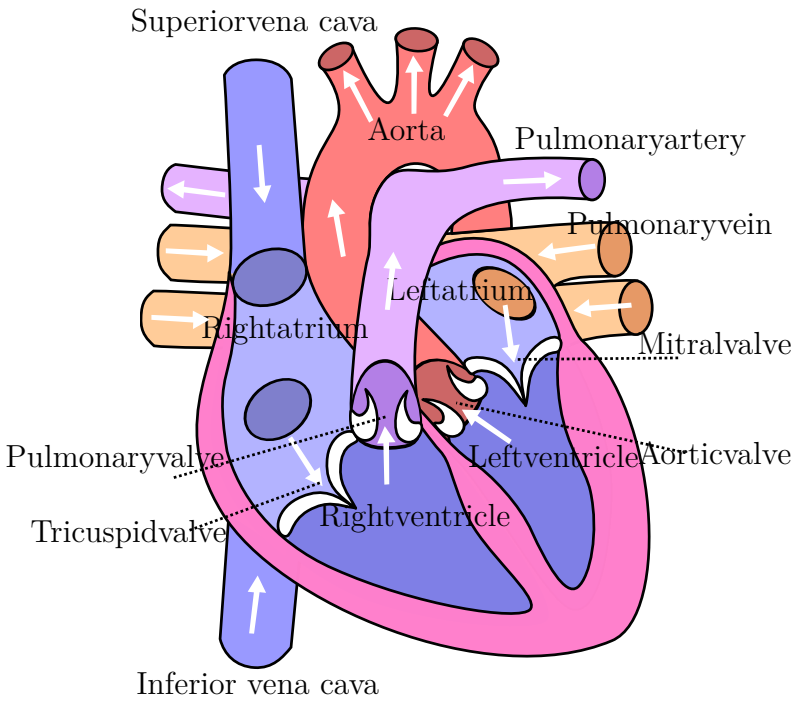

Figure 2.1: Diagram of the human heart with arrows indicating the direction of the blood-flow. Illustrated by Eric Pierce via Wikimedia Commons, licensed under CC BY-SA 3.0

The heartbeat is generated by the collective contraction of cardiac muscle cells, known as cardiomyocytes. This process is managed by a complicated signal transmission system, where contractions are initiated by the sinoatrial (SA) node, forming what is known as a functional syncytium [5]. The SA is located in the RA and produces electrical impulses that cause the atria to contract, thus pumping blood into the ventricles. The electrical impulses, transmitted through local activation of the cardiomyocytes, are then conducted to the atrioventricular node (AV), with a brief pause to allow the ventricles to fill with blood, then further on through the cardiac conduction system's His-Purkinje network to activate the ventricles to contract [12].

2.1.2 The cell membrane and excitable tissue

Cardiomyocytes (heart muscle cells) are excitable, which means that they can respond to electrical stimuli. At rest, these cells maintain ionic concentrations that differ from those of their surroundings. Because ions carry electrical charge,

a potential difference arises between the inside and outside of the cell, known as the transmembrane potential (TP). When electrical stimuli are applied to excitable cells, the TP increases. If the stimulus is strong enough, the cell's conductive properties change, leading to a flow of positive ions into the cell. This results in the TP increasing from a negative value to a positive value or close to zero, a process referred to as membrane depolarization. Heart cells stay in this depolarized state for a significant amount of time, typically a few hundred milliseconds, called the plateau phase. After depolarization, the potential returns to its resting value during a process called repolarization. The entire cycle consisting of depolarization followed by repolarization is termed a cardiac action potential (AP) [5].

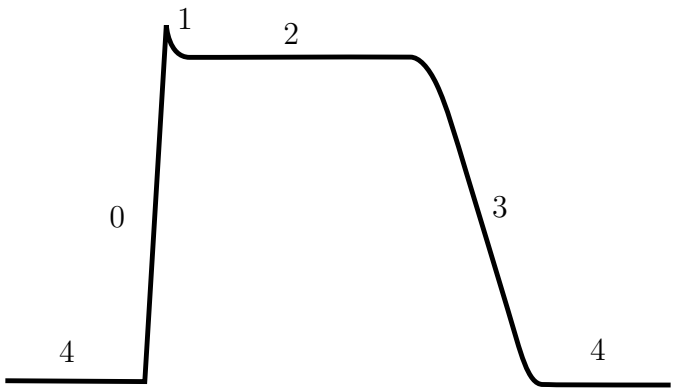

Figure 2.2: Illustration of ventricular action potential with its five phases: (0) *depolarization*, (1) *initial repolarization*, (2) *plateau*, (3) *repolarization*, (4) *Resting potential*. Illustrated by Wikimedia Commons user Ksheka , licensed under CC BY-SA 3.0

We now shift our focus from the physiological basis of overall cardiac function, excitability and conduction to a theoretical point of view, considering the mathematical modeling of excitable tissue. The electrical behavior of excitable cells can be represented as an electrical circuit composed of a combination of resistors and capacitors, with variable resistors acting as ion channels that enable ions to move in and out of the cell.

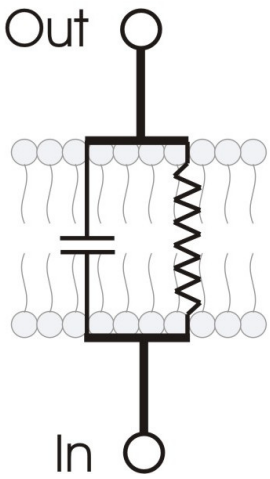

Figure 2.3: Resistor-capacitor circuit model of the cell membrane. Adaptation of illustration made by Wikipedia user Synaptidude, licensed under GNU Free Documentation License

In a capacitor, the charge is stored in two conductive plates separated by a dielectric and the ability of a capacitor to store charge is termed its capacitance. Since the cell membrane acts as an insulator with selective permeability to various ions, it can be viewed as a capacitor. The membrane capacitance, C_m, is defined as

$$C_m = \frac{Q}{V}, \tag{2.1}$$

where Q is the charge across the cell membrane and V is the electric potential needed to hold that charge. Resistors represent the ion channels that allow the flow of ions across the membrane. The resistance of these channels, R, is inversely proportional to their conductance.

Kirchhoff's current law states that the net current flowing into a junction must be equal to the net current flowing out of the junction. In the context of the cell membrane, this can be written as

$$I_{ion} + I_{cap} = 0, \tag{2.2}$$

where I_{ion} is the net ionic current across the membrane and I_{cap} is the capacitive current due to the charging and discharging of the membrane. Since current can be defined as $\frac{dQ}{dt}$, it follows that

$$C_m \frac{dV}{dt} + I_{ions} = 0. \tag{2.3}$$

This basic equation forms the basis for more complex models of excitable cells.

2.1.3 The Aliev-Panfilov Ionic Model

In order to be able to represent the action potential in its entirety, we need an ionic model that accounts for both depolarization and repolarization. The Aliev-Panfilov model uses two state variables to do this:

$$I_{ion} = -kV(V-a)(V-1) - VW, \tag{2.4}$$

$$\frac{dW}{d\tau} = (\epsilon + \frac{\mu_1 W}{V+\mu_2})(-W - kV(V-b-1)), \tag{2.5}$$

where V is an adimensional variable that diffuses across neighboring cells, and W is a non-diffusible variable, called a recovery variable which governs the restitution properties of the action potential. V is given in arbitrary units(AU) and relates to the transmembrane potential by $V = \frac{V_m - v_r}{v_n}$, where $v_r = -80\text{mV}$ is the resting potential of the cardiac tissue and $v_n = 100\text{mVAU}^{-1}$ is a normalization constant. Furthermore, time τ is measured in temporal units(TU) such that $t[\text{ms}] = 12.9\tau[\text{TU}]$. The parameters a, b, k, μ_1, μ_2 and ϵ are generally selected based on empirical data to mimic the observed electrical signals [13].

2.1.4 The Monodomain Model

Up until now, we have focused on studying excitable tissue at the cellular level. However, it is not practical to model an organ as large as the heart by modeling each individual cell separately due to computational and other constraints. Hence, we move over to a volume averaging approach and consider the body as a volume conductor. The following derivations are based on [17].

For a volume conductor, Farday's law states that [9]

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \tag{2.6}$$

where $\mathbf{E}$ is the electric field and $\mathbf{B}$ is the magnetic field. In the quasi-static approximation, the temporal variations of the magnetic field are negligible due to

the relatively slow ion movements compared to the propagation of electromagnetic waves. This simplifies (2.6) to

$$\nabla \times \mathbf{E} = 0, \tag{2.7}$$

which states that the electric field has zero curl, and can thus be expressed as the gradient of a scalar potential ϕ:

$$\mathbf{E} = -\nabla\phi. \tag{2.8}$$

This relationship states that the electric field points in the direction of the steepest decrease in voltage, which entails that positively charged ions will move from high to low potential, and the opposite for negatively charged ions.

Since the velocity of charges can be ignored, the current density $\mathbf{J}$ is described by Ohm's law

$$\mathbf{J} = \Sigma\mathbf{E}, \tag{2.9}$$

where Σ is the conductivity tensor.

Combining (2.9) and (2.8), we can thus describe the current density in terms of the gradient of ϕ as

$$\mathbf{J} = -\Sigma\nabla\phi \tag{2.10}$$

The bidomain model considers the heart as composed of two overlapping continuous domains, namely the intracellular (subscripted i) and the extracellular domains (subscripted e) [33]. Let ϕ_i and ϕ_e be intracellular and extracellular potentials, respectively. The transmembrane potential, V_m, is then defined as the potential difference across the cell membrane such that

$$V_m = \phi_i - \phi_e. \tag{2.11}$$

From (2.10), we can describe the current densities in the two domains as

$$\mathbf{J_i} = -\Sigma_i\nabla\phi_i \tag{2.12}$$

and

$$\mathbf{J_e} = -\Sigma_e\nabla\phi_e. \tag{2.13}$$

Any current leaving or entering one of the domains must do so through the membrane. Therefore, the change in current density in the intracellular and extracellular domain must be equal in magnitude with opposite sign [17].

$$\nabla \cdot \mathbf{J_i} = -\nabla \cdot \mathbf{J_e} = A_m I_m, \tag{2.14}$$

where I_m is the transmembrane current per unit area and A_m is the surface to volume ratio of the cell membrane.

Combining (2.13) and (2.12) with (2.14) yields our first bidomain equation

$$\nabla \cdot (\Sigma_i \nabla \phi_i) = -\nabla \cdot ((\Sigma_i + \Sigma_e) \nabla \phi_e). \tag{2.15}$$

The transmembrane current can be described by a capacitive current and an ionic current

$$I_m = C_m \frac{\partial V_m}{\partial t} + I_{ion}. \tag{2.16}$$

Combining this with (2.14) yields our final bidomain equation

$$\nabla \cdot (\Sigma_i \nabla V_m) + \nabla \cdot (\Sigma_i \nabla \phi_e) = A_m (C_m \frac{\partial V_m}{\partial t} + I_{ion}) \tag{2.17}$$

The bidomain model, defined by (2.15) and (2.17), is computationally expensive. By assuming equal anisotropy rates, such that $\Sigma_e = \lambda \Sigma_i$, where λ is a constant, the bidomain equations can be simpliflied to a single equation named the monodomain equation

$$\nabla \cdot (\Sigma_m \nabla V_m) = A_m (C_m \frac{\partial V_m}{\partial t} + I_{ion}), \tag{2.18}$$

where it can be shown that the monodomain model is equivalent to the bidomain model if the conductivity tensor is given by the harmonic mean tensor Σ_m[22]:

$$\Sigma_m = (\Sigma_i \Sigma_e)(\Sigma_i + \Sigma_e)^{-1}. \tag{2.19}$$

Additionally, in the monodomain model, it is reasonably assumed that no current is leaving the heart. This is imposed by no flux Neumann boundary conditions as

$$(\Sigma_m \nabla V_m) \cdot \vec{n} = 0, \tag{2.20}$$

where $\vec{n}$ is the boundary normal.

2.1.5 Monodomain combined with the Aliev-Panfilov Ionic Model

Having introduced the components of the Aliev-Panfilov ionic model and the monodomain model, we now proceed to integrate these models together. In order to do this, we need to ensure dimensional consistency. Considering the ionic current I_{ions} described in (2.4) in the Aliev-Panfilov model, we have

$$I_{ion} = \left[\frac{\text{AU}}{\text{TU}}\right]. \tag{2.21}$$

Thus, it is necessary to determine a scaling coefficient, κ, to make sure that the units of the capacitive current and the diffusion term in (2.18) align with I_{ion}. Consequently, we then get

$$\kappa C_m A_m \frac{\partial V_m}{\partial t} = \left[\frac{\text{AU}}{\text{TU}}\right], \tag{2.22}$$

and

$$\kappa \nabla \cdot (\Sigma_m \nabla V_m) = \left[\frac{\text{AU}}{\text{TU}}\right]. \tag{2.23}$$

We also have that

$$V_m = v_n V - 80\text{mV} \quad \text{and} \quad t = \tau_n \tau,$$

where $v_n = 100\frac{\text{mV}}{\text{AU}}$ and $\tau_n = 12.9\frac{\text{ms}}{\text{TU}}$ are normalization constants. We then get

$$\begin{aligned} \frac{\partial V_m}{\partial t} &= v_n \frac{\partial V}{\partial t} \\ &= v_n \frac{\partial V}{\partial \tau} \frac{\partial \tau}{\partial t} \\ &= \frac{v_n}{\tau_n} \frac{\partial V}{\partial \tau}, \end{aligned} \tag{2.24}$$

and similarly

$$\begin{aligned} \nabla V_m &= \nabla(v_n V - 80) \\ &= v_n \nabla V. \end{aligned} \tag{2.25}$$

Substitution of (2.24) into (2.22) gives

$$\kappa C_m A_m \frac{v_n}{\tau_n} \frac{\partial V}{\partial \tau} = \left[\frac{\mathrm{AU}}{\mathrm{TU}}\right], \tag{2.26}$$

and since $\frac{\partial V}{\partial \tau} = \left[\frac{\mathrm{AU}}{\mathrm{TU}}\right]$, we must have

$$\kappa = \frac{\tau_n}{v_n C_m A_m}.$$

Similarly, we get

$$\kappa \nabla \cdot (\Sigma_m v_n \nabla V) = \left[\frac{\mathrm{AU}}{\mathrm{TU}}\right],$$

and we can thus define a diffusion tensor $D = \kappa v_n \Sigma_m$. In isotropic conditions, the conductivity is the same in all directions and the diffusion tensor can be reduced to a scalar value.

Finally, we obtain the following coupled equations with rescaled units

$$\frac{dV}{d\tau} = \nabla \cdot (D \nabla V) - kV(V-a)(V-1) - VW, \tag{2.27}$$

and

$$\frac{dW}{d\tau} = (\epsilon + \frac{\mu_1 W}{V + \mu_2})(-W - kV(V - b - 1)). \tag{2.28}$$

2.2 Physics Informed Neural Networks

Having established the necessary physiological basis and theoretical background of the governing physics in this study, we now transition into the concept of PINNs. For a detailed description of neural networks the reader is referred to the work of Goodfellow et al. [8]. Our emphasis herein is on the framework of PINNs based on [19]. A PINN is a type of neural network designed to solve both forward and inverse problems involving PDEs and ODEs. The general idea behind PINNs is that we can construct a loss function such that, when minimized, the governing equations are satisfied by incorporating their residuals into the loss function. In

the approach first suggested by Raissi et al.[27], a feed forward neural network (FFNN) is used to represent a target function.

2.2.1 Feed Forward Neural Network

An FFNN is a computational model consisting of multiple layers of interconnected nodes. In these networks, as the name suggests, the information moves in only one direction, from input nodes, through hidden layers and to output nodes.

As described in [19], let $\mathcal{N}(\theta, \mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$ be a neural network with inputs $\mathbf{x} \in \mathbb{R}^n$ and m outputs, where θ are tunable parameters consisting of weights, $W^\ell \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$, and biases $\mathbf{b}^\ell \in \mathbb{R}^{N_\ell}$ and N_ℓ is the number of nodes in layer $\ell \in [0, L]$. Regardless of the underlying architecture of the neural network, the resulting outputs will be linear transformations of the inputs. Consequently, introducing a non-linear activation function, f, is crucial in order to represent non-linear relationships. The neural network is thus a composite function defined as

$$\begin{aligned} \text{input layer:} \quad & \mathcal{N}^0(x) = x \in \mathbb{R}^n, \\ \text{hidden layers:} \quad & \mathcal{N}^\ell(x) = f(W^\ell \mathcal{N}^{\ell-1}(x) + b^\ell) \in \mathbb{R}^{N_\ell}, \\ \text{output layer:} \quad & \mathcal{N}^L(x) = W^L \mathcal{N}^{L-1}(x) + b^L \in \mathbb{R}^m, \end{aligned}$$

2.2.2 Automatic differentiation

Training a neural network involves determining the optimal parameters θ^*. Central to this process is the backpropagation algorithm[28], which is a specific case of Automatic Differentiation (AD). The iterative procedure leverages the chain rule to update the network's weights and biases with the goal of minimizing a loss function. Considering the fact that the neural network represents a composite function, AD applies the chain rule repeatedly to compute the derivatives of the loss function with respect to the parameters θ. PINNs also require computing the

derivatives of the neural network outputs with respect to its inputs, which can be achieved by means of AD. For simplicity, consider a neural with one input, x, one hidden layer, and one output y. The forward propagation then yields

$$\begin{aligned}\mathcal{N}^1 &= W^1 x + \mathbf{b}^1 \\ y &= W^2 a^1 + \mathbf{b}^2, \quad \text{where } a^1 = f(\mathcal{N}^1).\end{aligned}$$

Then by applying the chain rule, we can find the derivative of the output with respect to the input as follows

$$\begin{aligned}\frac{\partial y}{\partial x} &= \frac{\partial y}{\partial a^1}\frac{\partial a^1}{\partial x} \\ &= \frac{\partial y}{\partial a^1}\frac{\partial a^1}{\partial \mathcal{N}^1}\frac{\partial \mathcal{N}^1}{\partial x} \\ &= W^2 W^1 f'(\mathcal{N}^1),\end{aligned}$$

which demonstrates an important advantage of computing derivatives via automatic differentiation, primarily the fact that these derivatives no not depend on any discretization.

2.2.3 PINN loss

We assign the first output to represent our estimate of V, and the second output to represent the estimate of the recovery variable W such that

$$\mathcal{N}^L = \begin{bmatrix}\tilde{V} \\ \tilde{W}\end{bmatrix}. \tag{2.29}$$

We can thus define residuals of Equations (2.27) and (2.28) in terms of the neural network outputs and derivatives as

$$R_V = -\frac{\partial \tilde{V}}{\partial \tau} + \nabla \cdot (D\nabla \tilde{V}) - k\tilde{V}(\tilde{V} - a)(\tilde{V} - 1) - \tilde{V}\tilde{W}, \tag{2.30}$$

and

$$R_W = -\frac{\partial \tilde{W}}{\partial \tau} + \left(\epsilon + \frac{\mu_1 \tilde{W}}{\tilde{V} + \mu_2}\right)(-\tilde{W} - k\tilde{V}(\tilde{V} - b - 1)). \tag{2.31}$$

A hybdrid loss function $\mathcal{L}$ can then be defined as a combination of components which ensure both data fitting and consistency with the governing equations:

- $\mathcal{L}_{\text{data}}$ ensures agreement with both voltage and recovery variable data for N_d samples within the spatial domain Ω :

$$\mathcal{L}_{\text{data}} = \frac{1}{N_d}\sum_{i=1}^{N_d}\left(V(\vec{x}_i,t_i) - \tilde{V}(\vec{x}_i,t_i)\right)^2 + \frac{1}{N_d}\sum_{i=1}^{N_d}\left(W(\vec{x}_i,t_i) - \tilde{W}(\vec{x}_i,t_i)\right)^2 \quad \vec{x}_i \in \Omega \tag{2.32}$$

- $\mathcal{L}_{PDE}$ and $\mathcal{L}_{ODE}$ account for the physical laws by evaluating the residual given by (2.30) and (2.31) evaluated at N_c collocation points within Ω :

$$\mathcal{L}_{PDE} = \frac{1}{N_c}\sum_{i=1}^{N_c}\left(R_V(\vec{x}_i,t_i)\right)^2 \quad \vec{x}_i \in \Omega \tag{2.33}$$

$$\mathcal{L}_{ODE} = \frac{1}{N_c}\sum_{i=1}^{N_c}\left(R_W(\vec{x}_i,t_i)\right)^2 \quad \vec{x}_i \in \Omega \tag{2.34}$$

- $\mathcal{L}_{\text{BC}}$ ensures no flux Neumann boundary conditions on N_{BC} points sampled from the boundary $\partial\Omega$:

$$\mathcal{L}_{\text{BC}} = \frac{1}{N_{BC}}\sum_{i=1}^{N_{\text{BC}}}\left(\vec{n}\cdot\left(D(\vec{x}_i)\nabla\tilde{V}(\vec{x}_i,t_i)\right)\right)^2 \quad \vec{x}_i \in \partial\Omega : \tag{2.35}$$

- and finally $\mathcal{L}_{\text{IC}}$ ensuring that the network's output $\tilde{V}$ at the initial time t_0 is close to the initial potential V_0 on N_{ic} sampled data points

$$\mathcal{L}_{\text{IC}} = \frac{1}{N_{\text{IC}}}\sum_{i=1}^{N_{\text{IC}}}\left(V(\vec{x}_i,t_0) - \tilde{V}(\vec{x}_i,t_0)\right)^2 \quad \vec{x} \in \Omega \tag{2.36}$$

We then get

$$\mathcal{L} = \lambda_d\mathcal{L}_{data} + \lambda_{PDE}\mathcal{L}_{PDE} + \lambda_{ODE}\mathcal{L}_{ODE} + \lambda_{BC}\mathcal{L}_{V_{BC}} + \lambda_{IC}\mathcal{L}_{IC}, \tag{2.37}$$

where λ_d, λ_{PDE}, λ_{ODE}, λ_{BC}, λ_{IC} are hyperparameters that weight the importance of each term in the loss function. Assigning these weights will be further discussed in the following chapter.

Chapter 3

Method

In this section, we outline the methodologies employed to develop and evaluate our PINN-models. The experiments are structured into two main parts: simulations using simple geometries and simulations based on an MRI-derived geometry. Initially, we use one-dimensional (1D) cable and two-dimensional (2D) square grid setups to generate synthetic data, providing a controlled environment to benchmark our models and compare them with previously published work [13]. We then extend our approach to more complex and anatomically accurate heart models derived from MRI-data, capturing the intricacies of cardiac tissue structure and fiber orientation. All PINN were implemented in PyTorch [25]. The parameters used in the Aliev-Panfilov model, adopted from the original paper by Aliev and Panfilov [24], are summarized in Table 3.1.

Table 3.1: Aliev-Panfilov Model Parameters

Parameter	**Value**
μ_1	0.2
μ_2	0.3
k	8.0
a	0.15
b	0.15
ϵ	0.002

3.1 1D Cable and 2D square geometry

To reproduce and build upon the research presented in [13], we generate *in-silico* cardiac EP data to train and evaluate our models using two simple geometries: a 1D cable and a 2D structured square grid.

3.1.1 Synthetic Data Generation

We generated *in-silico* cardiac electrophysiological (EP) data, which served as the training and evaluation dataset. This is done by approximating the second derivatives using a central finite difference and employing a 4th-order Runge-Kutta (RK4) iterative scheme, as shown in Alg.(1), for the time evolution. We impose no-flux Neumann boundary conditions in both cases.

The spatial domain was discretized using a uniform grid spacing h:

- For the 1D case, the second spatial derivative was approximated using the central difference method:

$$\frac{\partial^2 V}{\partial x^2} \approx \frac{V_{i-1} - 2V_i + V_{i+1}}{h^2} \tag{3.1}$$

  where V_i is the value of V at the i-th grid point.

- For the 2D case, the Laplacian $\nabla^2 V$ was approximated using:

$$\nabla^2 V \approx \frac{V_{i-1,j} + V_{i+1,j} + V_{i,j-1} + V_{i,j+1} - 4V_{i,j}}{h^2} \tag{3.2}$$

  where $V_{i,j}$ is the value of V at the grid point (i, j).

Additionally, we constructed additional data sets with added noise sampled from a normal distribution with standard deviation σ.

Algorithm 1 Fourth-Order Runge-Kutta (RK4) Method for Aliev-Panfilov Model

procedure RUNGE-KUTTA 4TH ORDER

Define $f(V, W)$ and $g(V, W)$ ▷ Right-hand sides of the differential equations

$\Delta t \leftarrow$ time step size

for $i = 1, 2, \ldots, n$ **do** ▷ Loop over time steps

$k_1 V \leftarrow \Delta t \cdot f(V_i, W_i)$

$k_1 W \leftarrow \Delta t \cdot g(V_i, W_i)$

$k_2 V \leftarrow \Delta t \cdot f\left(V_i + \frac{k_1 V}{2}, W_i + \frac{k_1 W}{2}\right)$

$k_2 W \leftarrow \Delta t \cdot g\left(V_i + \frac{k_1 V}{2}, W_i + \frac{k_1 W}{2}\right)$

$k_3 V \leftarrow \Delta t \cdot f\left(V_i + \frac{k_2 V}{2}, W_i + \frac{k_2 W}{2}\right)$

$k_3 W \leftarrow \Delta t \cdot g\left(V_i + \frac{k_2 V}{2}, W_i + \frac{k_2 W}{2}\right)$

$k_4 V \leftarrow \Delta t \cdot f\left(V_i + k_3 V, W_i + k_3 W\right)$

$k_4 W \leftarrow \Delta t \cdot g\left(V_i + k_3 V, W_i + k_3 W\right)$

$V_{i+1} \leftarrow V_i + \frac{1}{6}(k_1 V + 2k_2 V + 2k_3 V + k_4 V)$

$W_{i+1} \leftarrow W_i + \frac{1}{6}(k_1 W + 2k_2 W + 2k_3 W + k_4 W)$

3.1.2 Evaluation

In order to evaluate the performance, we calculated the root mean squared error (RMSE) across the evaluation points as outlined in [13].

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\tilde{V}_i - V_{\mathrm{ref}_i})^2} \tag{3.3}$$

To investigate the variability of the models, we trained 10 distinct models for each combination of training sample sizes ($n = 10^2, 10^3, 10^4$) and noise levels ($\sigma = 0.0, 0.01, 0.1$). We calculated the RMSE values for each model and reported the interquartile range (IQR) to measure the variability in the predictions. The IQR, representing the middle 50% of the RMSE values, provided an insight into the consistency of model performance across different training scenarios.

3.2 MRI based geometry

In this section, we present the methodology concerning more complex and anatomically accurate 2D geometries derived from MRI data.

3.2.1 Synthetic Data generation for the MRI based geometry

As shown in Figure 3.1, a computational model was derived from a late gadolinium-enhanced MRI 2D-slice acquired from from the Royal Brompton Hospital, using a previously described protocol [11].

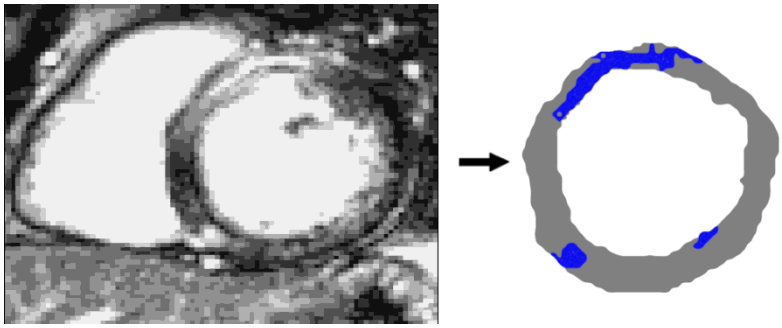

Figure 3.1: Generation of a Computational Heart Model from MRI Data. The left panel shows a raw 2D MRI slice. The right panel presents the resulting computational model that has been extracted from the MRI data, where the blue region marks an area with scar tissue and reduced conductivity.

We generate a computational mesh composed of triangles with edge length of 250 μs consisting of 72434 vertices. The image processing and mesh generation methodologies are outlined in detail by Balaban et al. [2]. We carried out FEM simulations using openCARP [22] under both isotropic and anisotropic conditions with an initial stimulus given by a circular pulse with strength $500\mu\text{A/cm}^2$ with a duration of 2μs. The simulations were carried out using a time step of 20μs, with V_m and W outputed every 5ms for a total of 211 time steps. The simulations all consiststed of a single heartbeat with a duration of 1050ms. In the isotropic conductivity scenario the longitudinal and transverse conductivities were as shown in Table 3.2.

Parameter	Value
g_{et}	1
g_{it}	1
g_{il}	0.2
g_{it}	0.2

Table 3.2: Isotropic conductivities

Under anisotropic conditions, the data was generated from multiple FEM simulations with fixed extracellular conductivities (g_{el} = 1 S/m, g_{et} = 1 S/m) and intracellular transverse conductivity (g_{it} = 1 S/m), while varying intracellular longitudinal conductivity (g_{il}) from 0.5 to 1 in steps of 0.05. Addtionally, MRI-derived scar tissue is introduced into the computational model as conductive heterogeneities in the regions marked in blue in Figure 3.1, with conductivities as shown in Table 3.3.

Parameter	Value
g_{il}	0.0775 S/m
g_{it}	0.0143 S/m
g_{el}	0.2785 S/m
g_{et}	0.0512 S/m

Table 3.3: Conductivities in the scar region.

3.2.2 Fibers

In the heart muscle, cardiomyocytes are interconnected to form elongated fibers, wherein excitation propagates more rapidly longitudinally than in the transverse direction [31]. Consequently, the orientation of these fibers determines the preferred direction of electrical wave propagation in the heart. Thus precise modeling to approximate these configurations is essential for achieving accurate simulations which reflect biological reality within pre-defined tolerance. We inferred fiber orientations from MRI using a widely-used rule-based method

described in [3].

The fiber orientations are initially defined for each triangle in the generated computational mesh. To translate this rotational data to the points used in a PINN, we employ a local averaging approach. For each target vertex, all triangles containing that vertex are identified. The fiber field at each point is then calculated by averaging the fiber fields of the neighboring triangles, weighted by the area of each triangle:

$$e_f(\text{point}) = \sum_{\text{neighboring triangles } i} \text{area}(\text{triangle}_i) \cdot e_f(\text{triangle}_i)$$

Finally, the fiber vector at each point is normalized.

Assuming that the intracellular and extracellular conductivity tensors are given by

$$\Sigma_i = \begin{bmatrix} g_{il} & 0 \\ 0 & g_{it} \end{bmatrix} \tag{3.4}$$

and

$$\Sigma_e = \begin{bmatrix} g_{el} & 0 \\ 0 & g_{et} \end{bmatrix}, \tag{3.5}$$

where g_{il} is the longitudinal intracellular conductivity, g_{it} is the transverse intracellular conductivity and similarly for the extracellular conductivities g_{el} and g_{el}.

We can calculate the local conductivity tensors $\tilde{\Sigma}_i, \tilde{\Sigma}_e$ by means of the local fiber vector $e_f(x, y)$ using tensor rotation. We define the local rotation tensor

$$R = \begin{bmatrix} e_f^x & e_f^y \\ -e_f^y & e_f^x \end{bmatrix} \tag{3.6}$$

so that

$$\tilde{\Sigma}_i = R\Sigma_i R^T \tag{3.7}$$

$$\tilde{\Sigma}_e = R\Sigma_e R^T \tag{3.8}$$

$$\tag{3.9}$$

and finally

$$\Sigma_m = \tilde{\Sigma}_i \tilde{\Sigma}_e (\tilde{\Sigma}_i + \tilde{\Sigma}_e)^{-1}. \tag{3.10}$$

3.2.3 Boundary normals

In the context of calculating normals when training a PINN, it is important to remember that we are dealing with a point cloud rather than a continuous surface. Considering that the normal of a point isn't well defined, we need to resort to an approximation. Given a point $\mathbf{P}$ and its two nearest neighbors $\mathbf{N}_1$ and $\mathbf{N}_2$, the line segment between the neighboring points, $\mathbf{l}$, is given by

$$\mathbf{l} = \mathbf{N}_1 - \mathbf{N}_2.$$

We can then approximate the boundary normal vector at point $\mathbf{P}$ as the normal vector of $\mathbf{l}$. This can be done by using the cross product of $\mathbf{l}$ with a unit vector orthogonal to the plane of interest (in this case, the z-axis unit vector $\mathbf{k} = [0, 0, 1]$) such that

$$\mathbf{n} = \mathbf{l} \times \mathbf{k}$$

The resulting normal vector $\mathbf{n}$ is then projected onto the xy-plane by discarding its z-component:

$$\mathbf{n}_{xy} = [n_x, n_y].$$

The orientation of the normal vector is then adjusted based on whether the point is on the inner or outer boundary.

Finally, all normal vectors were normalized to unit length:

$$\mathbf{n}_{xy} = \frac{\mathbf{n}_{xy}}{\|\mathbf{n}_{xy}\|}.$$

3.2.4 Evaluation

In evaluating the performance of our MRI-based models, we employ the relative L^1 norm, RL^1, defined as

$$RL^1 = \frac{\sum_i |\tilde{V}_i - V_i|}{\sum_i |V_i|}, \tag{3.11}$$

where $\tilde{V}_i$ and V_i are the PINN approximated solution and the reference FEM solution, respectively. We also consider the relative L^2 error, RL^2, for comparison with existing literature [36], which is defined as

$$RL^2 = \frac{\sqrt{\sum_i (\tilde{V}_i - V_i)^2}}{\sqrt{\sum_i V_i^2}}. \tag{3.12}$$

The relative L^1 error provides a scale-invariant assessment of model performance, ensuring that errors are evaluated proportionally to the magnitude of the targets. The relative L^1 error is less sensitive to slight phase shifts as compared to the relative L^2 norm, making it more robust in scenarios where small temporal misalignments might otherwise result in disproportionately large error values.

3.3 Optimization

3.3.1 Adam Optimizer

In order to find the optimal parameters θ^*, we need to minimize (2.37). One of the most commonly used optimization algorithms in training PINNs is Adaptive Moment Estimation (Adam) [16]. Adam is an extension of the stochastic gradient descent (SGD) algorithm that computes adaptive learning rates for each parameter. The algorithm maintains two moving averages for each parameter: the first moment and the second moment. These moments are used to adapt the learning rates of the parameters, which helps in converging more quickly and avoiding local minima more effectively. These properties make Adam a preferred choice for training complex neural network architectures such as PINNs [34]. For more details on the Adam optimizer, the reader can refer to the original paper by Kingma and Ba [16].

Algorithm 2 Adam Optimizer [16]

Initialize:

θ_0 (initial parameters)

$m_0 = 0$ (first moment vector)

$v_0 = 0$ (second moment vector)

$t = 0$ (time step)

η (learning rate)

$\beta_1 = 0.9$ (decay rate for first moment)

$\beta_2 = 0.999$ (decay rate for second moment)

$\epsilon = 10^{-8}$ (small constant to prevent division by zero)

while not converged **do**

  $t = t + 1$

  $g_t = \nabla_\theta L(\theta_{t-1})$ (compute gradients of the loss function L w.r.t. θ)

  $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$ (update biased first moment estimate)

  $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$ (update biased second moment estimate)

  $\hat{m}_t = m_t / (1 - \beta_1^t)$ (compute bias-corrected first moment estimate)

  $\hat{v}_t = v_t / (1 - \beta_2^t)$ (compute bias-corrected second moment estimate)

  $\theta_t = \theta_{t-1} - \alpha \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$ (update parameters)

return θ_t

3.3.2 Learning rate decay

The choice of learning rate is arguably the most important hyperparameter in training neural networks, as it significantly influences the convergence speed and the quality of the final model [35][8]. An appropriate learning rate ensures efficient progress in the loss landscape, while an excessively high learning rate can cause the training process to overshoot minima or even diverge, and a too-low learning rate can result in a prolonged training process that might get stuck in suboptimal local minima [8].

In the context of using the Adam optimizer, which adjusts the learning rate for each parameter based on the first and second moments of the gradients, the

learning rate parameter η acts as an approximate upper bound on the effective step size [16]. To optimize the training process, we employed an exponential learning rate scheduler as suggested in [34], which systematically decreases the learning rate η over the course of training. This scheduler is implemented in PyTorch and follows the formula $\eta_t = \eta_0 \cdot \exp(-\gamma t)$, where η_0 is the initial learning rate, γ is the decay rate, and t represents the epoch or iteration number.

3.4 Input scaling

Scaling of neural network inputs (in our case physical quantities) is an important step in training neural networks, ensuring that these inputs (features) contribute equally to the learning process. Differences in scales of features or very large features can lead to several issues that negatively impact model performance.

Firstly, features with larger scales can dominate the learning process, causing features with smaller scales to be effectively ignored. This imbalance can skew the model's predictions and reduce its overall performance [8].

Secondly, large feature values can saturate activation functions, particularly those like the sigmoid or tanh functions, which are prone to saturation. When these functions saturate, their gradients approach zero. This issue significantly hampers the learning process, as the model updates the weights very slowly [7].

In one commonly used scaling approach, feature scaling is performed as a preprocessing step before training the neural network [34]. Consequently, the differential equations governing the physics of the problem must be scaled accordingly to maintain consistency when evaluating the residuals given by (2.30) and (2.31).

An alternative approach (which we have not seen in PINN litterature) involves scaling the features within the network itself, making the scaling operation part of the computational graph so that the residuals of the governing equations are evaluated correctly. This can be achieved in two ways: by using a frozen scaling layer (fixed parameters) or by incorporating scaling as the first step in the forward pass. We introduce a scaling layer, which offers the added benefit of storing

the scaling factors together with the model's optimal parameters, simplifying the process for future inference. We employ min-max scaling, which transforms each feature individually so that they are within a given range $[a, c]$:

$$x' = a + \frac{x - \min(x)(c - a)}{\max(x) - \min(x)}, \tag{3.13}$$

where x' is the scaled feature x, while $\max(x)$ and $\min(x)$ denote that largest and smallest values of the specific feature in the training set.

Let $\vec{x} \in \mathbb{R}^i$ denote the inputs, where i is the number of input nodes. If $\vec{x}_{\max} \in \mathbb{R}^i$ and $\vec{x}_{\min} \in \mathbb{R}^i$ are the feature wise largest and smallest values respectively, then we can rescale the inputs by

$$\vec{x}' = \vec{a} + (x - \vec{x}_{\min}) \odot (\vec{c} - \vec{a}) \oslash (\vec{x}_{\max} - \vec{x}_{\min}), \tag{3.14}$$

where $\vec{x}'$ is the rescaled inputs, $\odot$ is element-wise multiplication and $\oslash$ is element-wise division. Each element of $\vec{a} \in \mathbb{R}^i$ and $\vec{c} \in \mathbb{R}^i$ have values a and c respectively. Now, let $W_s \in \mathbb{R}^{i \times i}$ be the weights matrix of the scaling layer and $\vec{b}_s \in \mathbb{R}^i$ be the bias. We then want our outputs from the scaling layer to be the rescaled inputs given by (3.15) such that

$$W_s\vec{x} + \vec{b}_s = \vec{a} + (x - \vec{x}_{\min}) \odot (\vec{c} - \vec{a}) \oslash (\vec{x}_{\max} - \vec{x}_{\min}). \tag{3.15}$$

Re-arranging the right-hand side, we get

$$W_s\vec{x} + \vec{b} = \vec{x} \oslash [\vec{x}_{\max} - \vec{x}_{\min}] + [\vec{a} - \vec{x}_{\min} \odot (\vec{c} - \vec{a}) \oslash (\vec{x}_{\max} - \vec{x}_{\min})]\,. \tag{3.16}$$

We then see that W_s must be a diagonal matrix where the diagonal is given by

$$\operatorname{diag}(W_s) = \vec{1} \oslash [\vec{x}_{\max} - \vec{x}_{\min}]\,, \tag{3.17}$$

and the bias is given by

$$\vec{b}_s = [\vec{a} - \vec{x}_{\min} \odot (\vec{c} - \vec{a}) \oslash (\vec{x}_{\max} - \vec{x}_{\min})]\,. \tag{3.18}$$

3.5 Multi-Objective Loss balancing

The hybrid loss function defined in (2.37) belongs to the category of Multi-Objective Optimization, where many components have different units of measurement, which inevitably creates a disparity in their magnitudes. This can lead to an imbalance where some loss components dominate the optimization process, favoring the components with the highest magnitude. To address this issue, a recently published study by Bischof et al. [4] proposes a self-adaptive loss balancing scheme for PINNs, namely *Relative Loss Balancing with Random Lookback*(ReLoBRaLo). The ReLoBRaLo method combines elements from previous adaptive loss balancing techniques and introduces new features such as random lookback and "temperature scaling" to dynamically adjust the contributions of different loss components. This results in more stable and efficient training of PINNs by preventing any single loss term from dominating the optimization process and allowing the network to achieve better overall performance.

Initialization: At the beginning of training, initial losses for each component are recorded. Each loss component is assigned an initial weight of 1.

Dynamic Weight Calculation: At each iteration, weights for the loss components are updated based on the relative changes in their respective loss values. The weight λ_i for each loss component $\mathcal{L}_i$ is calculated as:

$$\lambda_i(t) = \alpha\lambda_i^{\text{hist}}(t) + (1-\alpha)\lambda_i^{\text{bal}}(t, t-1), \tag{3.19}$$

where

$$\lambda_i^{\text{bal}}(t, t') = k \cdot \frac{\exp\left(\frac{L_i(t)}{\tau L_i(t')}\right)}{\sum_{j=1}^{k} \exp\left(\frac{L_j(t)}{\tau L_j(t')}\right)}, \tag{3.20}$$

and

$$\lambda_i^{\text{hist}}(t) = \rho\lambda_i(t-1) + (1-\rho)\lambda_i^{\text{bal}}(t, 0). \tag{3.21}$$

In this context, $\lambda_i^{\text{bal}}(t, t')$ defines the scaling in accordance with the relative improvement between a previous iteration t' and the current iteration t. In

this case, τ is a temperature parameter controlling the sharpness of the softmax operation. $\lambda_i^{\text{hist}}(t)$ determines if the scaling is calculated from the previous timestep or the improvement from the initial iteration. α is the exponential decay rate, and ρ is a Bernoulli random variable (taking values 0 or 1).

3.6 Architecture

The architectures of the PINN models are illustrated in Figure 3.2 and hyperparameters for each experiments are summarized in Table 3.4. In the case of isotropic conductivities, the network takes the spatial coordinates $\vec{x}$ and time t as inputs. For anisotropic and heterogeneous conductivities, the model is designed to take the components of the spatially varying local diffusion tensor (defined in 2.1.5) as inputs in addition to spatio-temporal coordinates.

The PINNs were implemented using PyTorch[25]. All models utilize the Glorot uniform initialization [7] for the weights, and biases are initialized to zero.

The network architectures used for the 1D and 2D structured geometries are adopted from [13], while the architectures for the MRI-based geometries were manually tuned to balance the complexity and computational efficiency. The isotropic model uses fewer layers and nodes, while the anisotropic model requires more complexity to capture locational and directional dependencies in conduction. We use the tanh activation function throughout.

Table 3.4: Hyperparameter settings by experiment

Hyperparameter	**1D Cable**	**2D Square**	**MRI Isotropic**	**MRI Anisotropic**
Max Epochs	20000	30000	50000	100000
Activation	tanh	tanh	tanh	tanh
Inputs	2	3	3	7
Hidden Layers	4	5	5	7
Nodes	32	64	128	$[128, \ldots, 128, 256]$
Weight Initialization	Glorot uniform	Glorot uniform	Glorot uniform	Glorot uniform
η	5×10^{-3}	5×10^{-3}	5×10^{-4}	2×10^{-4}
N_c	10^4	10^4	2×10^4	5×10^4
N_{BC}	10^2	10^3	2×10^3	2×10^3

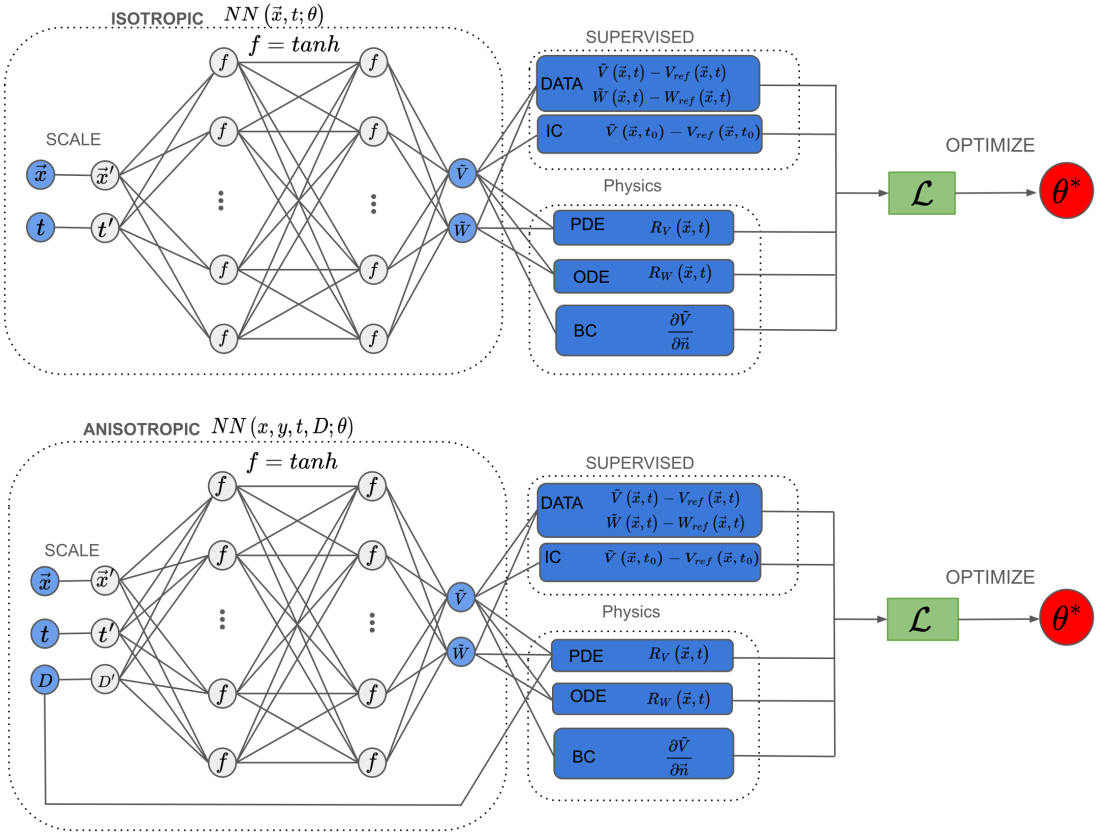

Figure 3.2: Architecture of the PINNs used in this study. The top section illustrates the isotropic model, which takes spatio-temporal coordinates $\vec{x}$ and t as inputs. The bottom section shows the anisotropic model with inputs x, y, t, D. All models use tanh activation functions and include supervised learning components (DATA and IC) as well as physics-based constraints (PDE, ODE, BC). The loss function $\mathcal{L}$ is optimized to determine the best parameters θ^*.

3.7 Training Procedures

This section outlines the training procedures employed for various models. Common in all training procedures, is the use of full-batch training, as evidence suggests that large batch sizes are best suitable for training PINNs, providing faster training times and improved accuracy [30]. Additionally, we use early-stopping, a form of regularization where training is stopped before overfitting happens. This is done by monitoring an appropriate metric computed across a validation dataset and saving the model with the lowest validation error as described in [19]. Finally,

all models in this study utilize gradient clipping during training. Gradient Clipping is a commonly used heuristic [8] to prevent excessively large gradients and has been demonstrated to accelerate the convergence of neural networks [39]. This technique involves rescaling the gradients to a predefined threshold, maintaining their direction while diminishing their magnitude.

The differences in training procedures are mainly concerning data selection and loss balancing:

- For the 1D and 2D structured geometry models, we selected n spatio-temporal datapoints sampled uniformly. The remaining data was split equally into validation and evaluation sets. In this experiment, all loss weights in (2.37) are set to one as done in [13].

- In the case of the MRI-based geometry with isotropic conductivities, we sample time-series data at n_s spatial locations. The remaining points were reserved for validation and evaluation (equally split). Loss weights are balanced as described in Section 3.5.

- For the MRI-based geometry with anisotropic conductivities, training is done with a subset of the datasets ($g_{il} = 0.5, 0.6, 0.7, 0.8, 0.9$), from which we sample $N = 10^6$ spatio-temporal points for training. A validation set is also constructed by sampling 10^6 spatio-temporal points. The selection of training data leaves the remaining conductivity scenarios for evaluating the unbiased predictive capabilities of the model outside the training range. Loss weights are balanced as described in Section 3.5.

Chapter 4

Results

This chapter presents the findings of our study, organized into distinct experiments. Each experiment is designed to build upon the previous one, adding complexity. We begin with simple models to establish a baseline and progressively move towards more complex and realistic simulations.

At this stage, we restate our guiding hypotheses:

- **Hypothesis 1:** The use of PINNs will result in faster simulations of electric waves in 2D MRI-slice-based heart geometries compared to traditional methods.
- **Hypothesis 2:** PINNs can accurately simulate electrophysiological wave propagation in 2D MRI-slice-based heart geometries, including the incorporation of image informed scar tissue data.
- **Hypothesis 3:** A trained PINN can effectively extrapolate PDE parameters, such as conductivities, beyond the range of training data, providing accurate and reliable predictions.

4.1 1D cable

We begin this exploration by considering a 2cm 1D cable domain. This experiment lays the foundation for the following experiments and allows us to compare our findings with existing literature [13]. Additionally, we compare our PINN solution

to a standard neural network (NN) trained with only the data without the physics loss components. The standard NN architecture and hyperparameters are the same as the PINNs in this experiment.

4.1.1 Data and Training 1D cable

We generated synthetic data as outlined in 3.1. The spatial domain was discretized with a step size of $\Delta x = 0.1$, and the time step used was $\Delta t = 0.005$. Different combinations of training sample sizes ($n = 100, 1000, 10000$) and noise levels ($\sigma = 0, 0.01, 0.1$) were used to create the datasets. For each combination, 10 separate models were trained, generating 10 RMSE values by computing the error across evaluation points consisting of the remaining data. The hyperparameters are outlined in Section 3.6.

The training loss convergence of a 1D model trained with $n = 100$ and $\sigma = 0.1$ is illustrated in Figure 4.1. The plot displays the evolution of different components of the loss, $\mathcal{L}_{PDE}$, $\mathcal{L}_{ODE}$, $\mathcal{L}_{BC}$, $\mathcal{L}_{IC}$, and $\mathcal{L}_{data}$, on a logarithmic scale. The graph shows a general decrease in the loss values, indicating successful training and convergence of the model. Notably, the $\mathcal{L}_{ODE}$ and $\mathcal{L}_{PDE}$ components demonstrate significant reduction, highlighting the model's ability to satisfy the underlying physical equations. The $\mathcal{L}_{data}$ loss component also decreases consistently, reflecting accurate fitting to the provided data points. Overall, the training process results in well-balanced optimization across most loss components, except for the boundary condition component, which increases to begin with before converging.

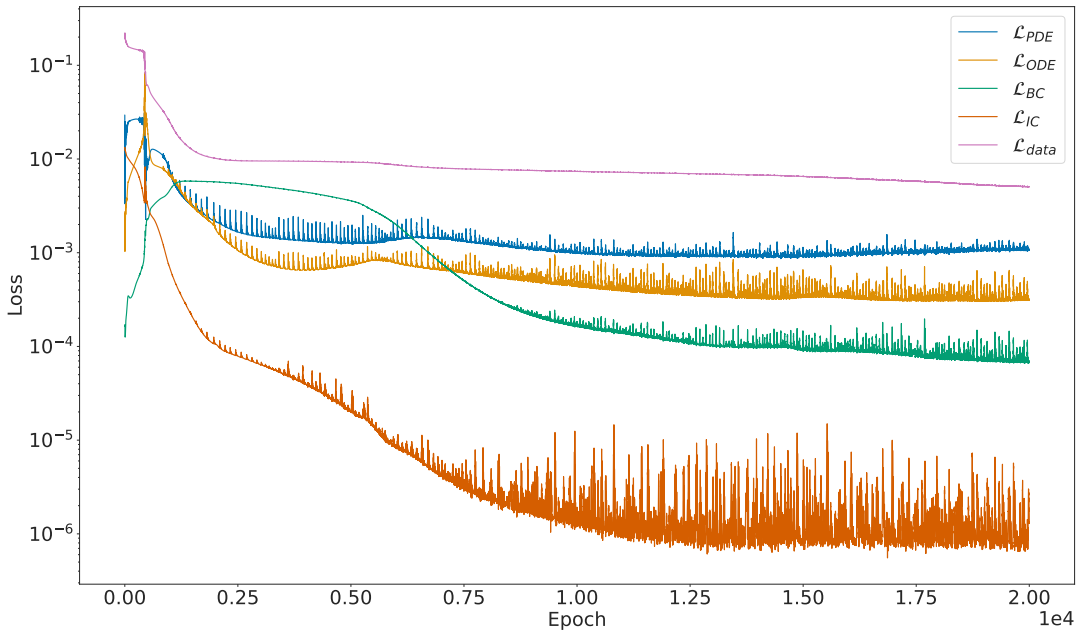

Figure 4.1: Training loss convergence of 1D model trained with $n = 100$ and $\sigma = 0.1$. The plot shows the evolution of different components of the total loss on a logarithmic scale.

4.1.2 Predictions: 1D cable

Figure 4.2 illustrates the comparison between the PINN prediction and the reference RK4 solution for the 1D cable model. Additonally, the figure shows the predicted solution from a standard neural network (NN) trained with only the data without the physics loss components. The state variable V is plotted as a function of time t at a randomly selected spatial location x. The figure demonstrates that the PINN prediction closely follows the RK4 solution, indicating the effectiveness of the PINN in capturing the dynamics of the system. Even with a noise level of $\sigma = 0.1$ (approximately 10% of the maximum value of V) and a limited number of spatio-temporal points ($n = 100$), the PINN successfully approximates the reference solution with considerable accuracy with RMSE $= 0.07$AU. In comparison, the standard NN achieves an RMSE of 0.15AU shows tendencies to overfit the data by interpolating the training data rather than learning the underlying dynamics.

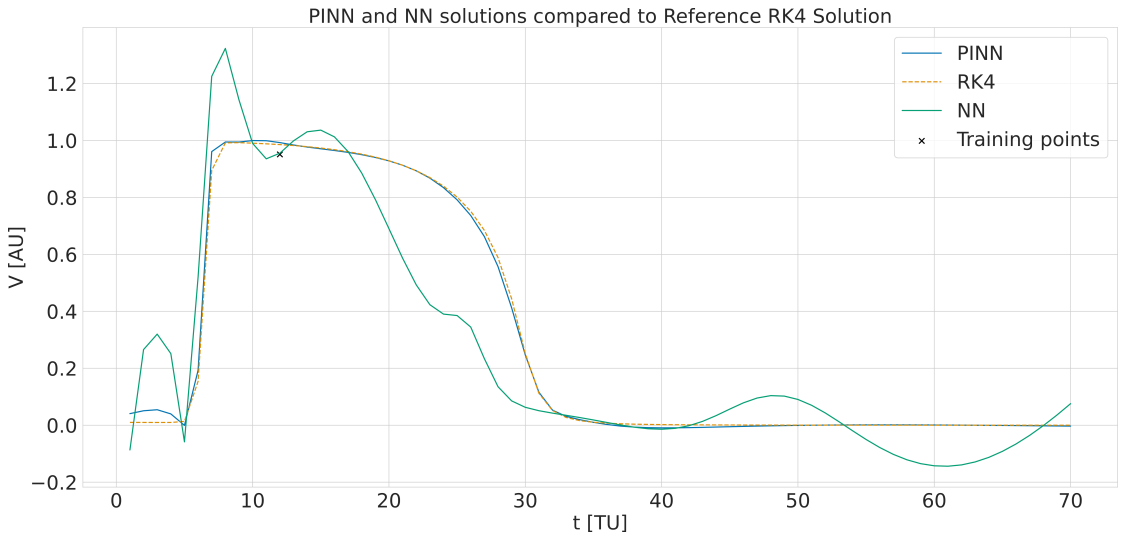

Figure 4.2: Comparison between the PINN prediction and the reference RK4. The blue line represents the PINN prediction, the orange line represents the RK4 solution, and the green crosses indicate the training data for the specific spatial location. The number of training points is $n = 100$ and the noise level is $\sigma = 0.1$.

Figure 4.2 illustrates the comparison between the PINN prediction, NN prediction, and the reference RK4 solution for the 1D cable model. The figure demonstrates that the PINN prediction again closely follows the RK4 solution. The PINN successfully approximates the reference solution with considerable accuracy with RMSE $= 0.02$AU when trained using $n = 1000$ data points and a noise level with a standard deviation of $\sigma = 0.1$. In comparison, the standard NN achieves an RMSE of 0.05AU and shows less tendencies to overfit the data as in the previous case (Figure 4.2).

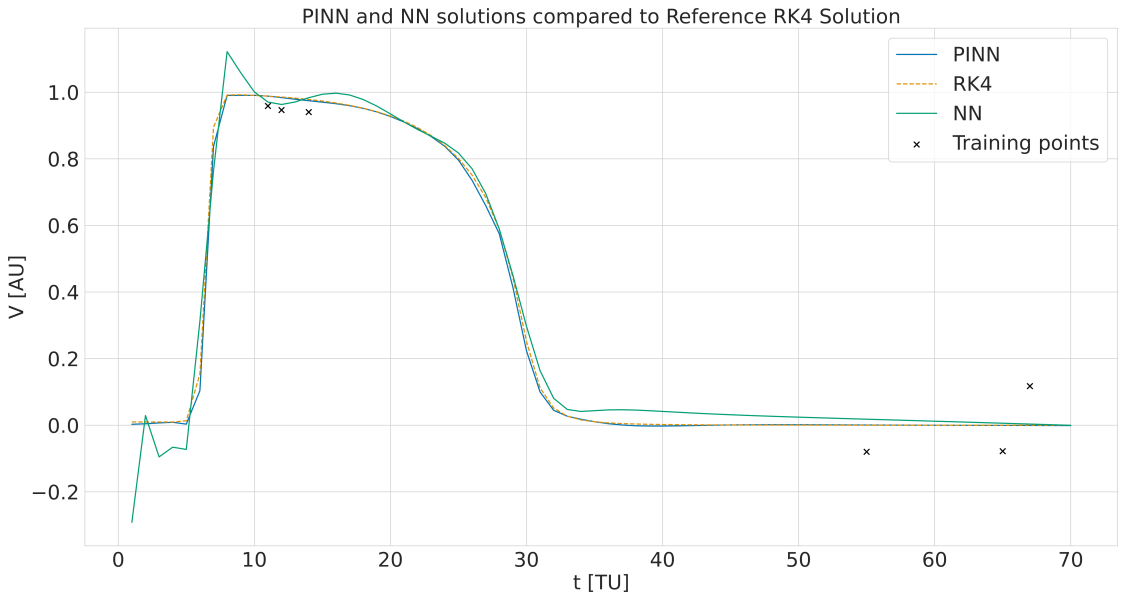

Figure 4.3: Comparison between the PINN prediction and the reference RK4. The blue line represents the PINN prediction, the orange line represents the RK4 solution, and the green crosses indicate the training data for the specific spatial location. The number of training points is $n = 1000$ and the noise level with $\sigma = 0.1$.

4.1.3 Evaluation 1D cable

Figure 4.4 displays the RMSE values for different combinations of training sample sizes, n, and noise levels sampled from a normal distribution with standard deviation σ. The IQRs are larger for higher noise levels as well as for smaller training sample sizes, showing greater variability in the results. We see that as the number of training points increases, the addition of noise becomes less significant. Additionally, we see some outliers where the models have converged to a less optimal solution.

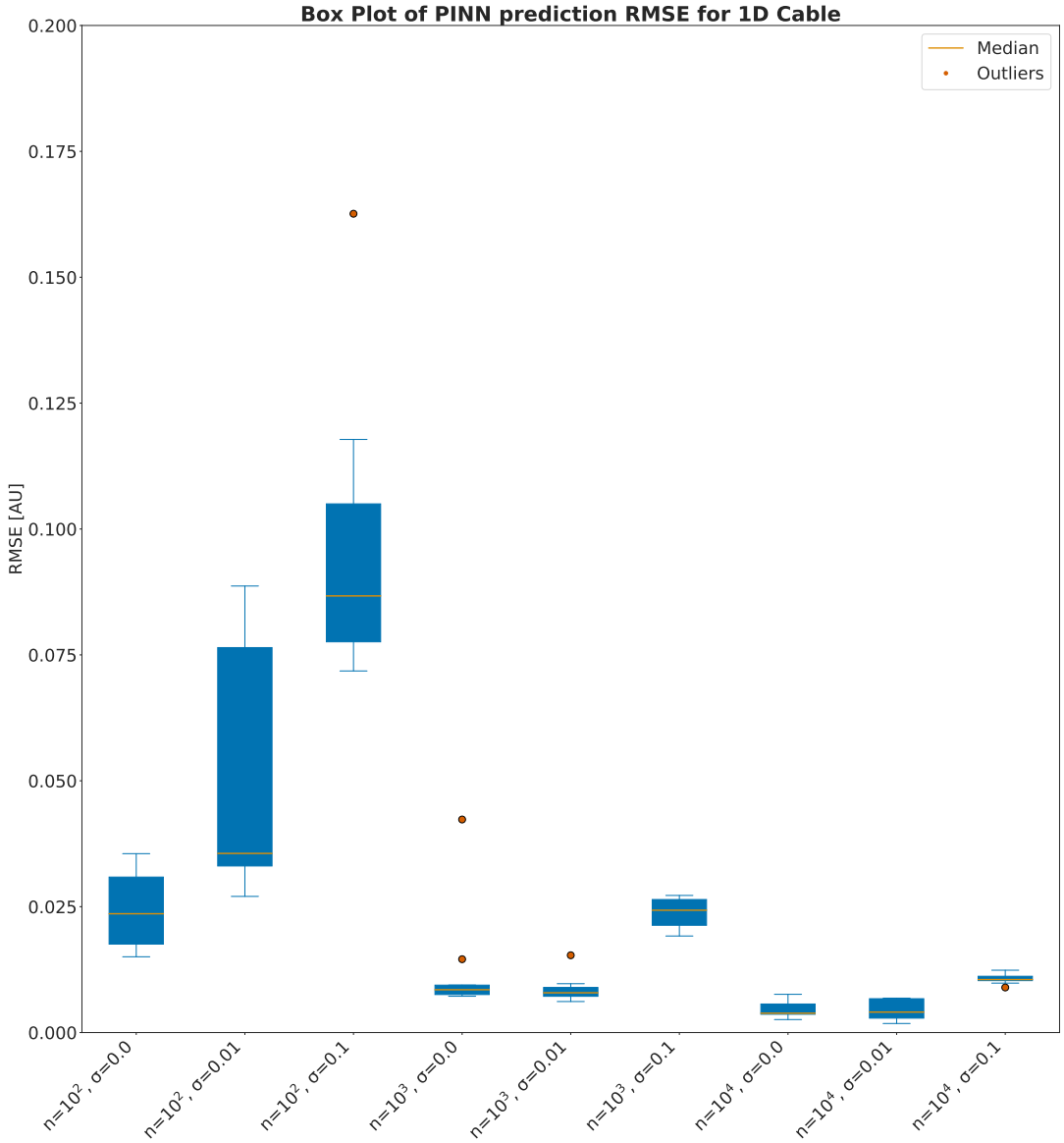

Figure 4.4: Box plot of RMSE values computed across evaluation points for different combinations of training sample sizes ($n = 10^2, 10^3, 10^4$) and noise levels ($\sigma = 0.0, 0.01, 0.1$). The blue colored boxes represent the IQR, indicating the middle 50% of the computed RMSE. The orange lines within the boxes show the median values. The whiskers (thin blue lines) extend from the smallest to the largest values, excluding outliers (orange dots).

4.2 2D structured geometry

4.2.1 Data and Training on a 2D Square

Building on the 1D simulations, we now consider a 1cm × 1cm 2D structured grid domain. The spatial domain was discretized with a step size of $\Delta x = 0.1$, and the time step used was $\Delta t = 0.005$. As for the 1-D cable, different combinations of training sample sizes ($n = 1000, 10000, 10000$) and noise levels ($\sigma = 0, 0.01, 0.1$) were used to create the datasets. Due to the added complexity from the additional spatial dimension, the neural network inputs now include 2D spatial coordinates and time (x, y, t).

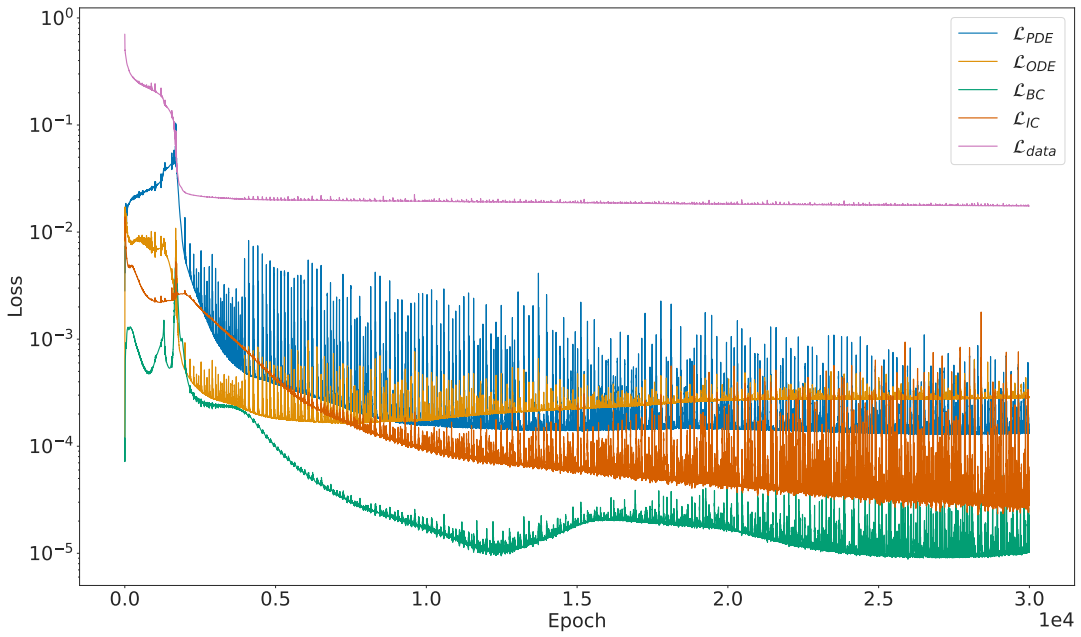

Figure 4.5: Training loss convergence of 1D model trained with $n = 100$ and $\sigma = 0.1$. The plot shows the evolution of different components of the total loss on a logarithmic scale.

The training loss convergence of a 2D square model trained with $n = 1000$ and $\sigma = 0.1$ is illustrated in Figure 4.5. The plot displays the evolution of different components of the loss, $\mathcal{L}_{PDE}$, $\mathcal{L}_{ODE}$, $\mathcal{L}_{BC}$, $\mathcal{L}_{IC}$, and $\mathcal{L}_{data}$, on a logarithmic scale. The graph shows a general decrease in the loss values, indicating successful

training and convergence of the model. Notably, the $\mathcal{L}_{ODE}$ and $\mathcal{L}_{PDE}$ components demonstrate significant reduction, but with larger fluctuation than seen in the 1D case (Figure 4.1). The $\mathcal{L}_{data}$ loss component decreases rapidly in the beginning of training, followed by a minimal, but steady decrease.

4.2.2 Predictions on a 2D Square

Figure 4.6 shows a comparison between the PINN prediction and the reference RK4 solution for the 2D structured grid model. The scalar field V is depicted across the spatial domain at a randomly chosen time point. The PINN accurately captures planar wave dynamics with some deviations at the wave front and resting potential area (0AU), even in the presence of noise with a standard deviation of about 10% of the maximum value of V.

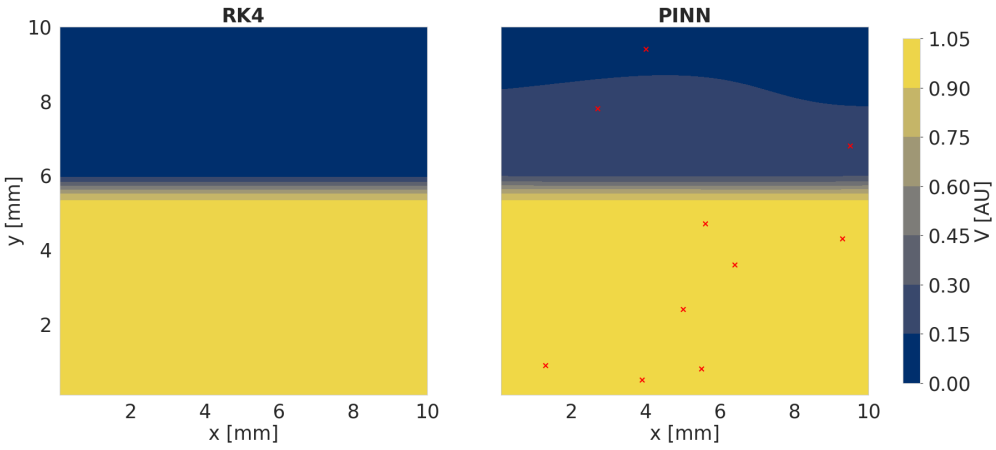

Figure 4.6: Comparison of the state variable V between the RK4 reference solution and the PINN prediction in a square domain. The red crosses indicate the training data points for this specific time step. The number of training points is $n = 1000$ and the noise level is $\sigma = 0.1$.

4.2.3 Evaluation of results on a 2D Square

Figure 4.7 displays the RMSE values for different combinations of sample sizes and noise levels, with RMSE computed across evaluation points. For each combination

of n and σ, 10 RMSE values are generated by training 10 separate models. Similarly to the 1D scenario, the interquartile ranges are larger for higher noise levels as well as for smaller training sample sizes, and as the number of training points increases, the addition of noise becomes less significant. Certain models in the 2D scenario display RMSE values that deviate significantly from the norm, both on the higher and lower ends.

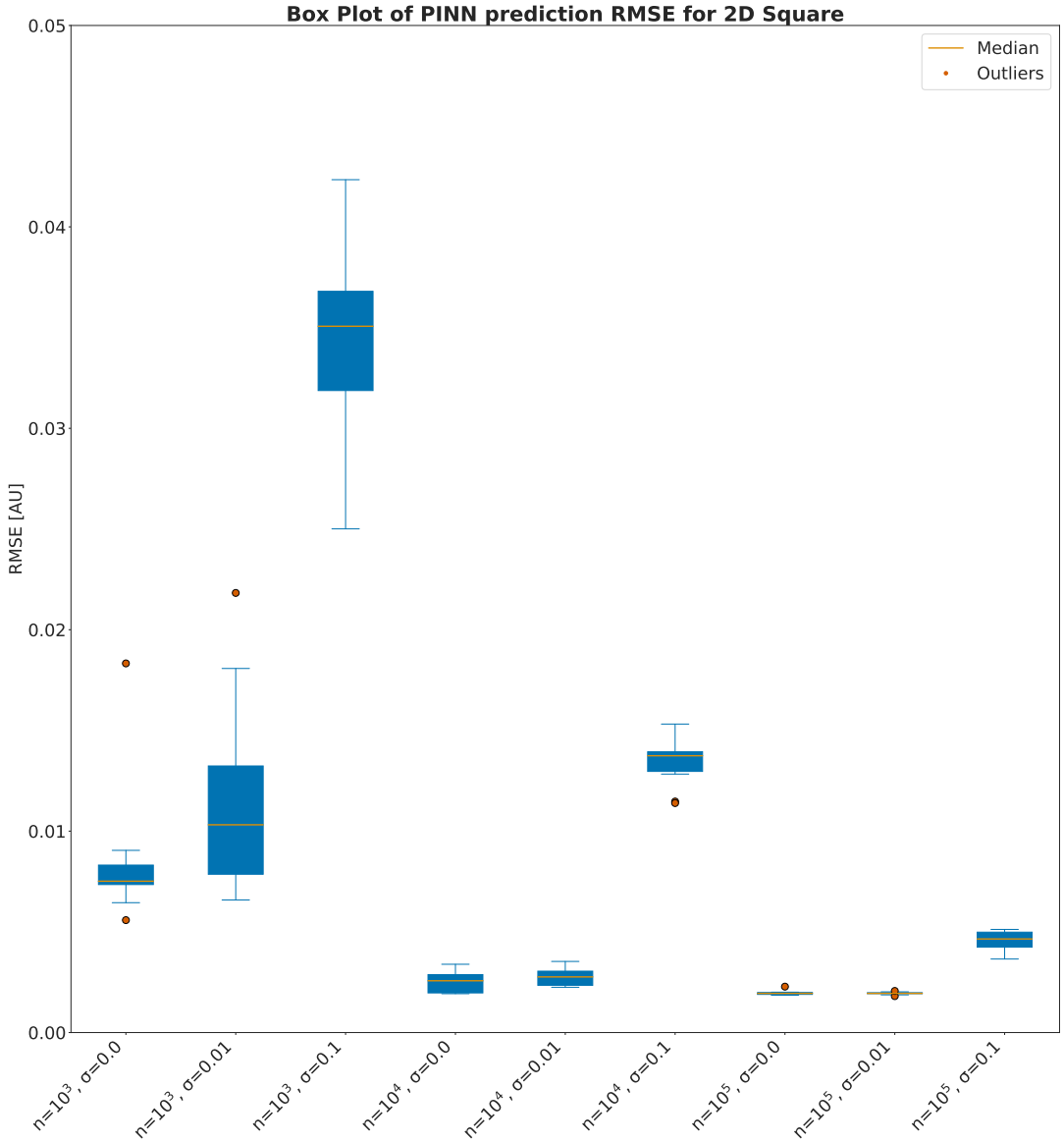

Figure 4.7: Box plot of RMSE values computed across evaluation points for different combinations of training sample sizes ($n = 10^3, 10^4, 10^5$) and noise levels ($\sigma = 0.0, 0.01, 0.1$). The blue colored boxes represent the IQR, indicating the middle 50% of the computed RMSE values. The orange lines within the boxes show the median values. The whiskers (thin blue lines) extend from the smallest and largest values, excluding outliers (orange dots).

4.3 MRI-based 2D geometry with isotropic conductivities

Extending to more complex and anatomically accurate models, the third experiment involved an MRI-based geometry with isotropic conductivities.

4.3.1 Data and training for an MRI-based 2D geometry with isotropic conductivities

We generate data from Finite Element Method (FEM) simulations based on an MRI-derived geometry consisting of 72434 vertices as described in Section3.2. The simulations are carried out using a time step of 20μs, output every 5ms for a total of 211 time steps. The simulation consists of a single heartbeat with a duration of 1050ms

Parameter	Value
g_{et}	1
g_{it}	1
g_{il}	0.2
g_{it}	0.2

Table 4.1: Isotropic conductivities

Time series data where then sampled at n_s spatial locations, each with $n_t = 211$ timesteps, to mimic a clinical setup where measurements are sparse in terms of spatial resolution, although the temporal resolution is usually high. This approach was employed to investigate the effectiveness of PINNs in reproducing a detailed transmembrane potential map from these sparse measurements. Furthermore, we look into the variability of the models' performance across different combinations of number of training samples($n_s = 20, 100, 1000$) and added noise ($\sigma = 0, 0.08, 0.8$). In doing so, we train ten separate models for each combination.

4.3.2 Predictions for an MRI-based 2D geometry with isotropic conductivities

Figure 4.8 displays the spatial distribution of the membrane potential (V_m) in a 2D MRI-based geometry at two different time points, 100 ms and 530 ms. The model was trained using 100 randomly sampled spatial locations including the complete time series with 211 timesteps. The relative L^1 error between the FEM and PINN results is provided, indicating an error of 0.02 for both time points, while the relative L^2 error is 0.05 and 0.04 at 100ms and 530ms respectively. The errors computed accross the whole simulation were $RL^1 = 0.02$ and $RL^2 = 0.05$. This indicates a very close match between the FEM and PINN solutions, and demonstrates the effectiveness of PINNs in handling complex geometrical structures derived from MRI data. In particular, this experiment shows that PINNs can reproduce the dynamics of electrophysiological wave propagation from sparse measurements and noisy measurements, critical in a clinical setting where data are often limited and have various sources of noise limiting their quality. However, some discrepancies at the wave-fronts can be observed at $t = 100$ms.

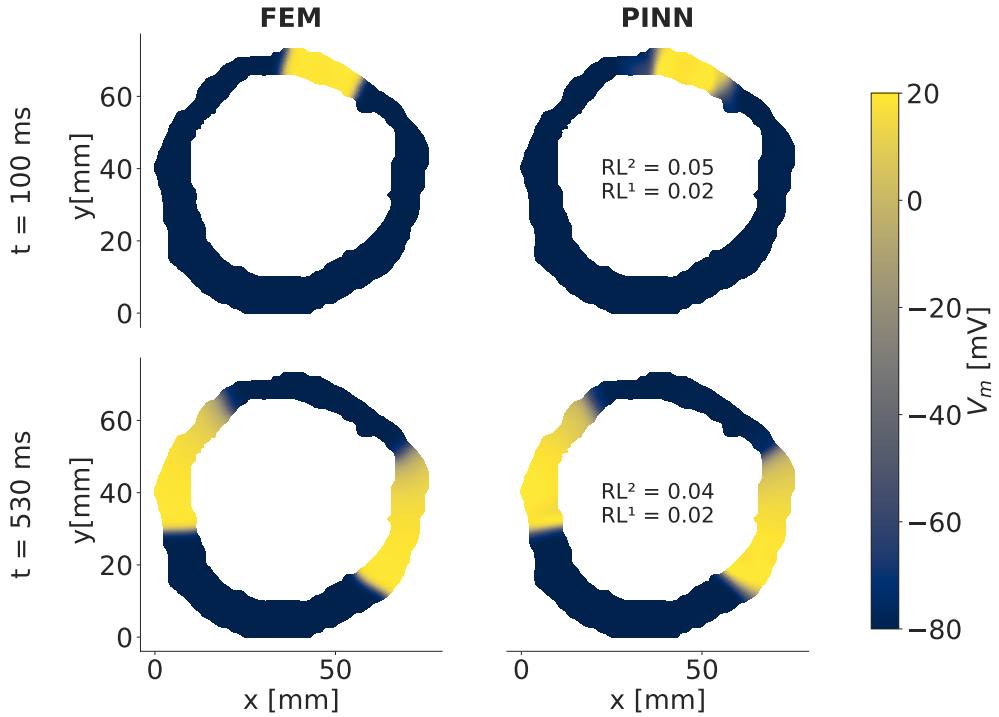

Figure 4.8: Predictions with isotropic and homogeneous conductivities $g_{il} = g_{it} = 0.2$ S/m. The left column shows the results from the FEM at $t = 100$ ms and $t = 530$ ms, while the right column shows the predictions from the PINN at the same time points. The corresponding relative L^1 errors are computed for each timestep and are 0.02 at both $t = 100$ ms and $t = 530$ ms. The number of spatial locations used in training is $n_s = 100$ with noise level with $\sigma = 8$ mV.

4.3.3 Evaluation of an MRI-based 2D geometry with isotropic conductivities

Figure 4.9 displays the relative L^1 error computed across evaluation points for different combinations of sample sizes and noise levels. As seen previously, the IQRs are larger for higher noise levels and a sparse training sample size, showing greater variability in the results. As the number of training points increases, the RL^1 values decrease and the addition of noise becomes less significant. The IQRs also become smaller, indicating more consistent performance across the 10 runs.

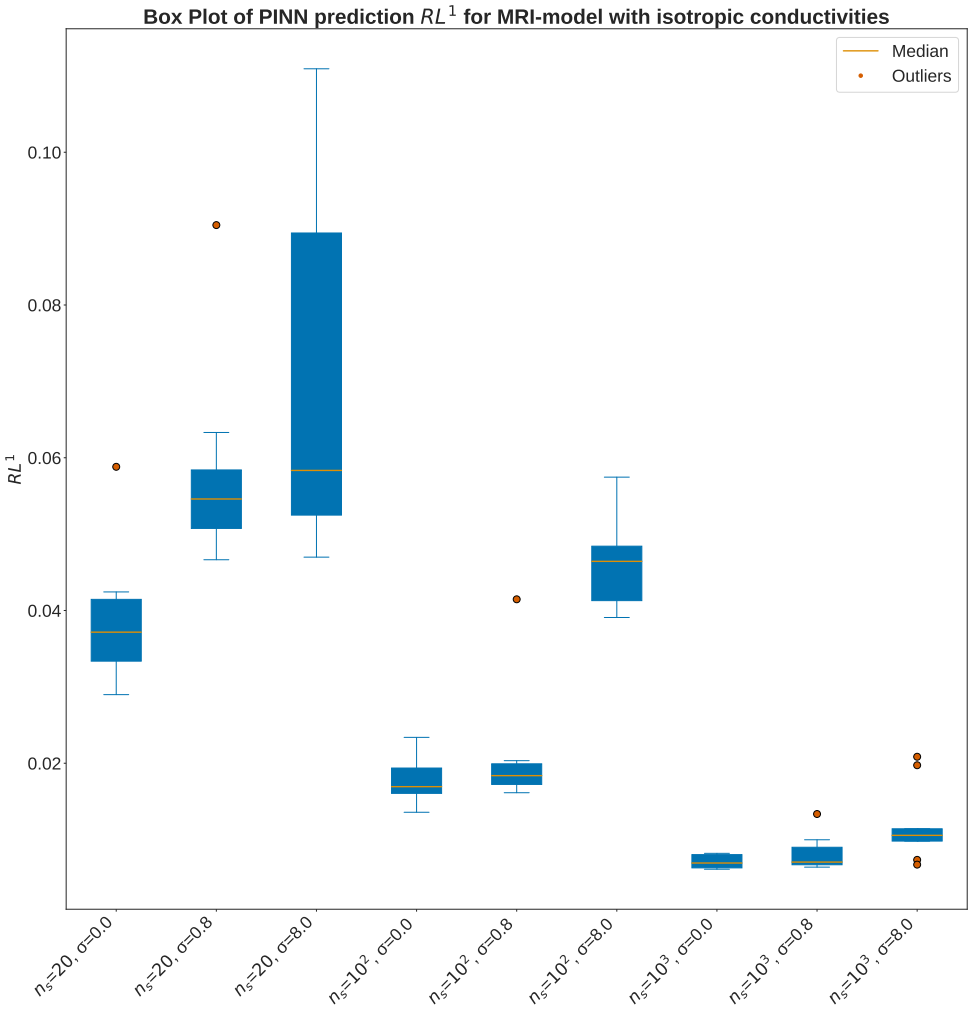

Figure 4.9: Box plot of the relative L_1 error of PINN predictions for the MRI-based model with isotropic and homogeneous conductivities. The x-axis labels indicate different training configurations, varying the number of spatial locations (n_s) and the standard deviation of noise (σ).

4.4 MRI-based geometry with Anisotropic Conductivities

In this experiment, we extend the MRI model to handle variable conductivity values as inputs, enbaling predictions with differing conductivities without retraining.

4.4.1 Data and Training for an MRI-based geometry with Anisotropic Conductivities

The data was generated from multiple FEM simulations with fixed extracellular conductivities ($g_{el} = 1$ S/m, $g_{et} = 1$ S/m) and intracellular transverse conductivity ($g_{it} = 1$ S/m), while varying intracellular longitudinal conductivity (g_{il}) from 0.5 to 1 in steps of 0.05. Conductive heterogeneities, representing scars, were introduced in the specified regions, marked in blue in Figure 4.10, with conductivities as shown in Table 4.2. The model was trained with a subset of the datasets ($g_{il} = 0.5, 0.6, 0.7, 0.8, 0.9$), from which we sample $N = 10^6$ spatio-temporal points.

Parameter	Value
g_{il}	0.0775 S/m
g_{it}	0.0143 S/m
g_{el}	0.2785 S/m
g_{et}	0.0512 S/m

Table 4.2: Scar conductivities.

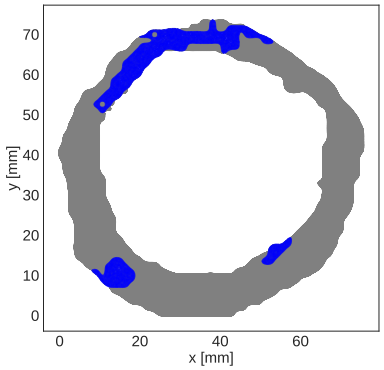

Figure 4.10: Heterogeneous conductivity map. Conductivities in the scar-region (blue) are fixed and shown in Table 4.2. The grey area has conductivities with varying g_{il}.

Figure 4.11 presents the convergence behavior of the loss function components during the training of an anisotropic and heterogeneous MRI-based model. The plot illustrates the evolution of different loss terms, namely $\mathcal{L}_{PDE}$, $\mathcal{L}_{ODE}$, $\mathcal{L}_{BC}$, $\mathcal{L}_{IC}$, and $\mathcal{L}_{data}$, over training steps on a logarithmic scale.

We see all loss components decrease rapidly during the initial phase of training, and the loss being dominated by the data component.

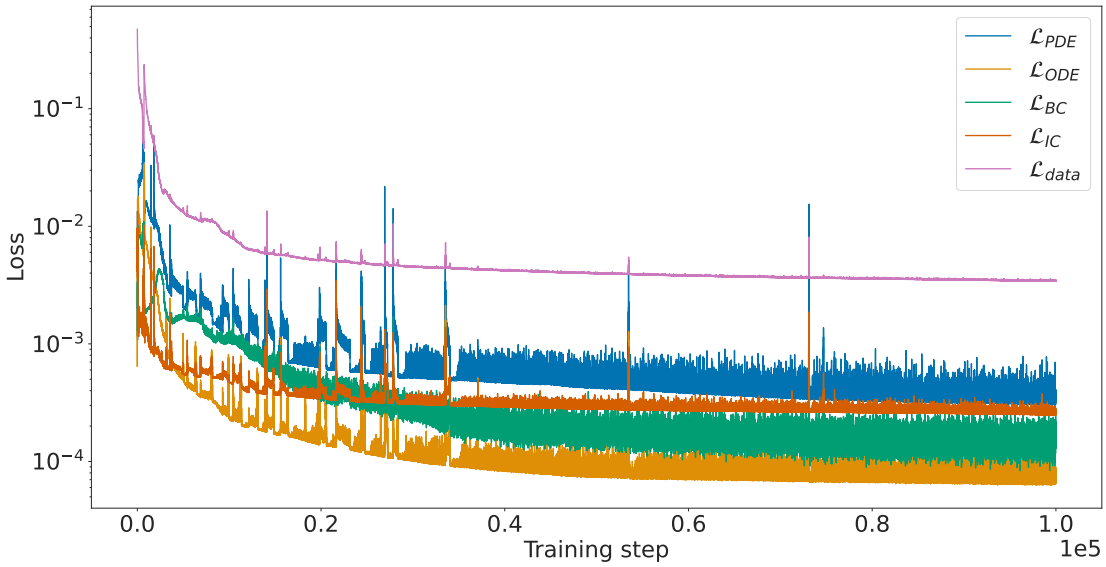

Figure 4.11: Anisotropic and heterogeneous MRI-based model training loss convergence during training on a logarithmic scale. The plot shows the evolution of different components of the total loss.

4.4.2 Predictions for an MRI-based geometry with Anisotropic Conductivities

Figures 4.12 and 4.13 present a comparison between the FEM reference solution and the PINN predictions when using conductivity inputs beyond the range of the training data. Both figures are produced using our best performing model in this experiment. In both figures, we see the PINN solution is able to accurately represent the transmembrane potential. Notably, we can see the wavefront slowing down when reaching the low-conduction zone(blue area in Figure 4.10) as the wave front curves around the scar area at $t = 100$ms in Figure 4.12, and at both $t = 100ms$ and $t = 530ms$ in Figure 4.13. The deceleration of the wave is also

evident, as the wavefront has propagated more to the right compared to the left between the two specified time points.

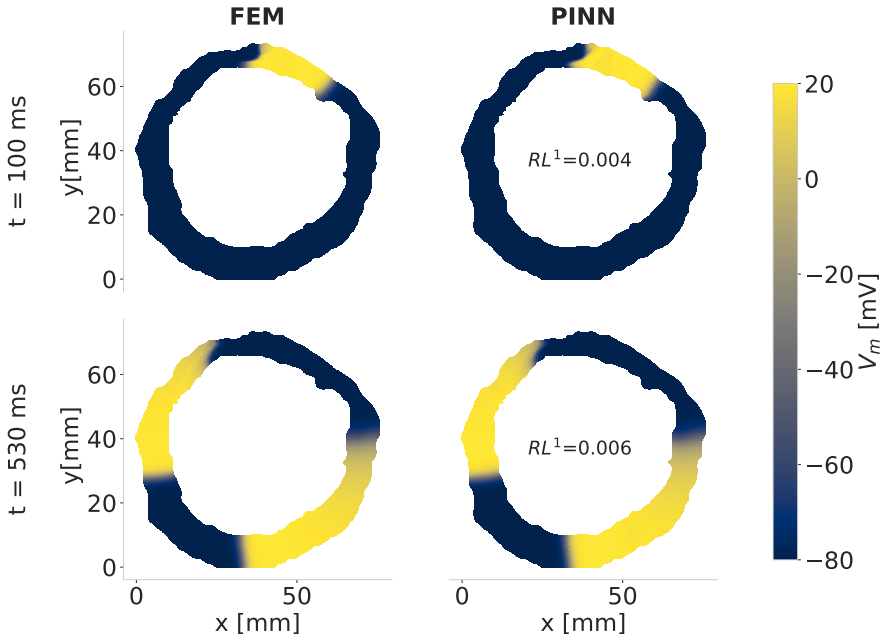

Figure 4.12: Interpolated conductivity predictions of the best performing anisotropic model using $g_{il} = 0.55$, which is not included in the training data. The left column shows the results from the FEM at $t = 100$ ms and $t = 530$ ms, while the right column shows the predictions from the PINN at the same time points. The corresponding relative L_1 errors (RL_1) are computed for each timestep and are 0.004 at $t = 100$ ms and 0.006 at $t = 530$ ms. The color bar indicates the membrane potential V_m in mV.

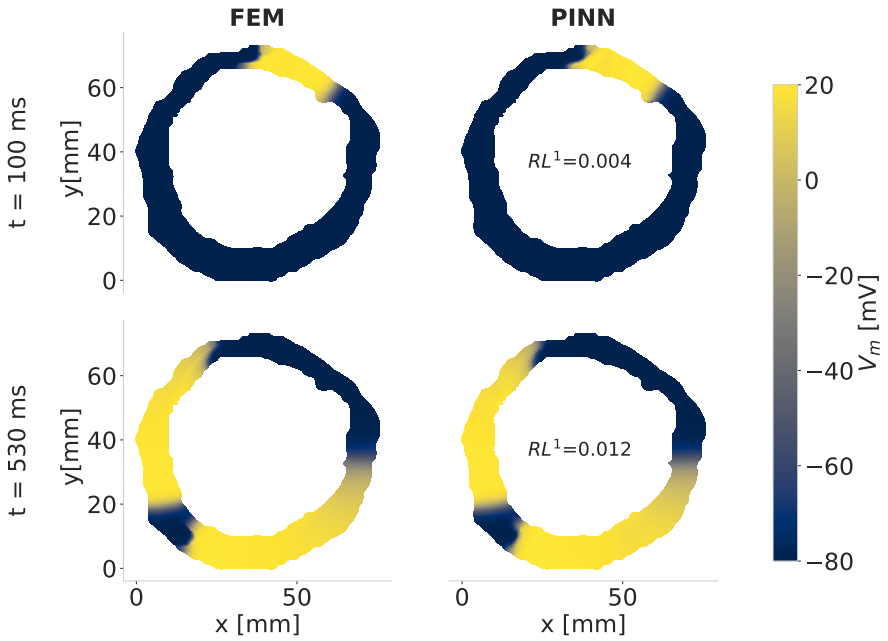

Figure 4.13: Extrapolated conductivity predictions of the best performing anisotropic model using $g_{il} = 1$, which is beyond the range of the training data. The left column shows the results from the FEM at $t = 100$ ms and $t = 530$ ms, while the right column shows the predictions from the PINN at the same time points. The corresponding relative L_1 errors (RL_1) are computed for each timestep and are 0.004 at $t = 100$ ms and 0.012 at $t = 530$ ms. The color bar indicates the membrane potential V_m in mV.

4.4.3 Model Evaluation of an MRI-based geometry with Anisotropic Conductivities

Figure 4.14 shows the relative L^1 error computed with a model trained on a data set with varying longitudinal intracellular conductivities (g_{il}). The results are highlighted for three cases: training data, interpolation, and extrapolation. The relative L^1 error increases slightly during extrapolation beyond the range of the training data set, while it remains comparable to the training data during interpolation. This figure illustrates the model's performance and its capability to generalize across different conductivity inputs.

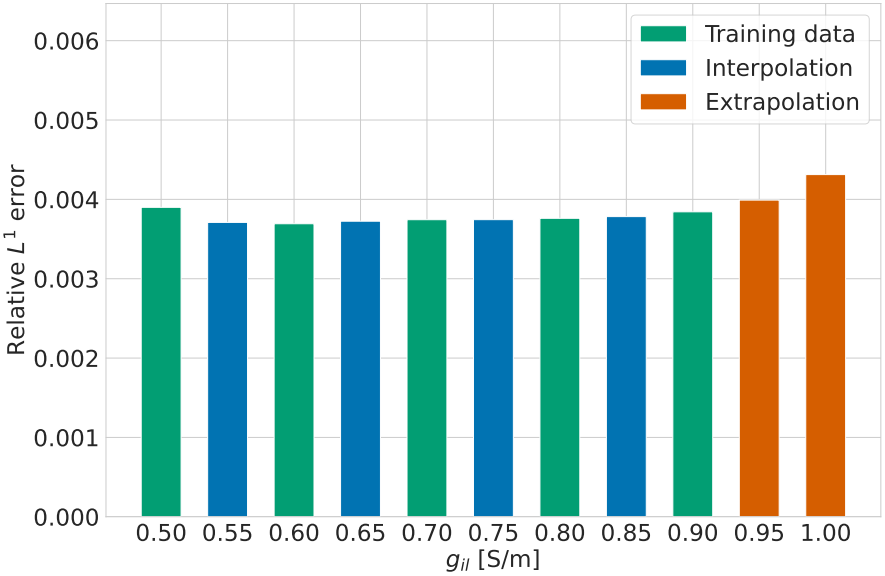

Figure 4.14: Relative L^1 error computed from the best performing MRI-based model with anisotropic and heterogeneous conductivities trained on a dataset generated by varying longitudinal intracellular conductivities g_{il}.

Figure 4.15 illustrates the variability from training five separate models and shows the relative L^1 error ranges across training, interpolation, and extrapolation sets. Each round marker represents the mean relative L^1 error, and the whiskers indicate the standard deviation, providing a sense of the variability in the predictions.

The training data (green), interpolation data (blue), and extrapolation data (orange) show relatively consistent error ranges, with standard deviations reflecting the spread of the error values. The plot suggests that the model maintains stable performance across different conductivity values, with no significant increase in error for extrapolated data, highlighting the robustness of the model in handling varying input conditions. The consistent error across different sets demonstrates that the model can generalize well beyond the training data, which is crucial for practical applications involving variable conductivity values as discussed further in following Chapters.

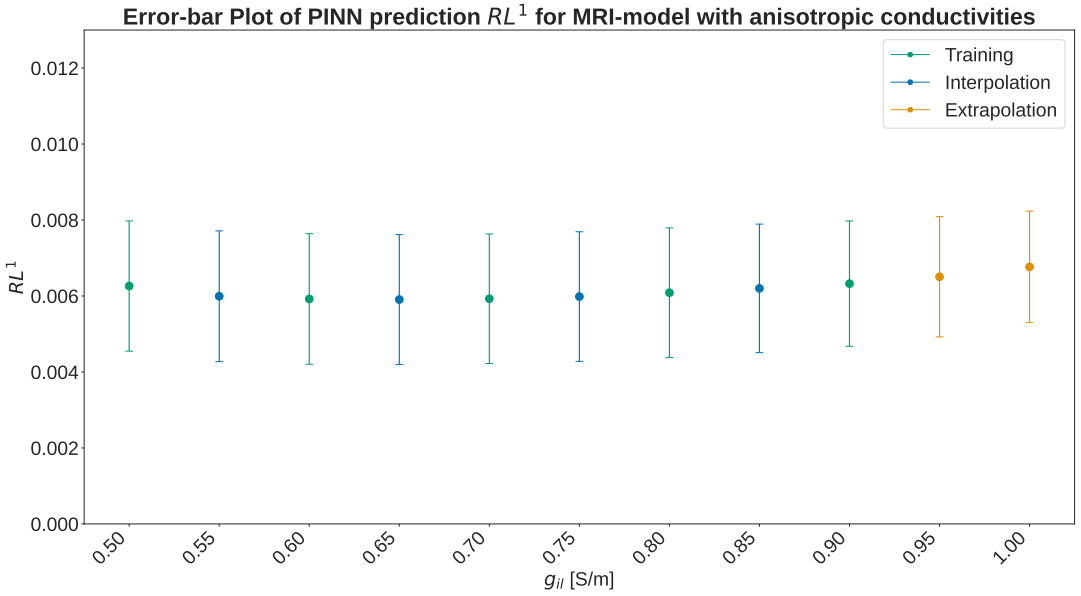

Figure 4.15: Error bar plot of the relative L_1 error of PINN predictions for the MRI-based model with anisotropic and heterogeneous conductivities. The figure is generated by training 5 separate models. The data set is generated by varying longitudinal intracellular conductivities g_{il}. The round markers indicate the mean and the whiskers indicate the standard deviation.

The performance comparison in terms of inference time between the traditional FEM and the PINN approach is presented in Table 4.16. Simulation times using the FEM approach averaged 684 ± 60 seconds when running on a CPU (AMD Ryzen 4800H laptop CPU), indicating a significant computational effort required for a single simulation.

In contrast, the PINN implementation demonstrated a markedly faster performance with an average CPU time of only 3.62 ± 0.09 seconds. Furthermore, when using a GPU (NVIDIA RTX3070 mobile), the PINN performance is enhanced even further, achieving an average computation time of just 0.22 ± 0.01 seconds per simulation, which is over $3000\times$ faster than FEM. The training time is typically around $6 - 10$h when parallelizing training on five NVIDIA RTX 2080Ti. In comparison, generating the complete dataset with varying conductivities takes approximately 2h with a time step of 20μs.

	FEM	PINN
CPU	684 ± 60 s	3.62 ± 0.09 s
GPU	-	0.22 ± 0.01 s

Figure 4.16: Comparison of FEM and MRI-based anisotropic PINN prediction inference time. The PINN time excludes the training time.

Chapter 5

Discussion

This study demonstrates the potential of PINNs for efficient simulations of cardiac electrophysiology. The training of PINNs was performed using various geometrical configurations, including 1D and 2D structured geometries, as well as MRI-based geometries. The main, novel, contributions of this work are as follows:

- An MRI-derived Cardiac Electrophysiology PINN model which includes image-informed scar regions with reduced conductivity.
- Extension of the PINN model to variable conductivity values, which removes the need for the PINN to be retrained when simulating scenarios with differing conductivities, as occurs in real-world physiological and pathophysiological scenarios.

5.1 1D cable and 2D Square geometry

The PINNs successfully reproduced the electrical wave propagation in 1D cable and 2D square geometries. The models showed high accuracy in capturing the dynamics. Specifically, the RMSE values were consistently low across different training sample sizes and noise levels. For both the 1D and 2D scenarios, our results exhibit a strong concordance with the RMSE values reported in [13], which specified RMSE $\leq 2.5 \times 10^{-2}$ for the 1D case when using 100 training and RMSE $\leq 3.0 \times 10^{-2}$ for the 2D structured grid scenario with 1.4×10^{5}

training points, without added noise in both cases. If we exclude the outliers seen in Figure 4.4, which were models getting stuck early on in training, we get RMSE $\leq 3.6 \times 10^{-2}$ in the 1D scenario. In the 2D scenario (Figure 4.7), we have $2.0 \times 10^{-3} \geq$ RMSE $\leq 1.5 \times 10^{-2}$ when using 10^4 training points, even with normally distributed noise with a standard deviation of approximately 10% of the maximum V value. Nevertheless, our models demonstrated significant variability when trained on smaller and noisier datasets (Figures 4.4 and 4.7). Additionally, We conducted a more extensive investigation into the variability of the models compared to [13], which trained only 5 models, whereas we trained 10 models. Furthermore, the authors utilized DeepXDE [19], a library designed for Physics-Informed machine learning and employed a different optimization process consisting of first training on the data only with Adam for a few epochs, followed by training using the complete loss and finally finishing training with the L-BFGS optimization algorithm [18]. Therefore, variations in both implementation and training procedures could have contributed to these observed differences. The presence of extreme outlier in Figures 4.4 and 4.7, can likely be attributed to the inherent stochastic nature of neural network training. Specifically, from the random initialization of the neural network parameters.

5.2 MRI-based 2D geometry with isotropic conductivities

Extending the analysis to more anatomically accurate models, the novel isotropic MRI-based 2D geometry experiments further highlight the strengths of PINNs. The comparison between FEM and PINN results, particularly the relative L_1 error values of 0.02 at both the 100 ms and 530 ms time points demonstrates the effectiveness of PINNs in handling complex geometrical structures derived from MRI data. In particular, this experiment shows that PINNs can reproduce the dynamics of electrophysiological wave propagation from sparse measurements (100 spatial points) and noisy measurements ($\sigma = 8.0$), which is crucial in a clinical setting where data are often limited. Although this is a proof-of-concept

and further refinement is needed, it provides a promising foundation for future clinical applications, supporting Hypothesis 2 by showing improved computational efficiency and maintaining accuracy in simulating these complex dynamics. In contrast to the work by Xie and Yao [36], which utilized a 3D geometry with a relatively sparse 1094 vertices, our 2D model employed 72434 vertices and achieved an RL^2 error of 0.05 compared to their RL^2 error of 0.08. Although these results are not directly comparable due to the dimensional difference, our initial experiments (structured geometries) suggest that there isn't significant loss in accuracy when transitioning from 1D to 2D with our approach. Therefore, similar accuracy could be maintained when extending our work from 2D to 3D. This comparison is particularly relevant as prior work in this field is scarce, and Xie and Yao [36] provide an example of an isotropic model in a 3D geometry, demonstrating that accurate PINN solutions are possible in such complex structures. However, it is important to note that the 3D study did not cover the modeling of scars nor MRI-derived geometries, which are unique contributions of our work.

5.3 MRI-based 2D geometry with anisotropic conductivities

The anisotropic MRI-based 2D geometry experiment demonstrates the generalization capabilities of PINNs. Figures 4.15 and 4.14 reveal several noteworthy patterns. Contrary to initial expectations, there is no marked distinction in the relative L^1 error between training, interpolation, and extrapolation datasets. This could be attributed to the majority of the training dataset being unseen by the model, resulting in similar performance across different subsets. Additionally, there appears to be a slight increase in error with higher conductivity values. This trend could be explained by the fact that larger conductivity corresponds to faster wave speeds, which makes the model more susceptible to errors due to slight phase shifts.

This novel approach, unprecedented in the context of cardiac electrophysiology, is particularly noteworthy as it illustrates the possibility to create a neural

network based solver capable of running different simulations in a fraction of a second by simply inputting different conductivities. In contrast, the FEM method necessitates a full repetition of the process for every new simulation. Additionally, the results show that the wave propagation slows down in regions of low conductivity, indicating that the PINN has successfully learned how conductivity affects wave propagation. These findings support Hypotheses 1 and 3, and underscore the significant advantages of using PINNs over traditional methods in terms of both speed and adaptability. The PINN training process can be slow and resource-intensive, presenting a significant investment in computational resources and time. However, our study demonstrates that this investment can be highly beneficial. Once trained, our PINN model can efficiently handle varying conductivities as inputs, offering significant flexibility, adaptability and fast inference times.

Chapter 6

Conclusions

6.1 Conclusions

This research investigated the use of PINNs for simulating cardiac electrophysiology on 1D and 2D structured geometries, as well as MRI-based 2D unstructured geometries in both isotropic and homogeneous conductivity conditions, and anisotropic heterogeneous conductivity scenarios. The findings indicate that PINNs are highly effective in reproducing the dynamics of electrophysiological wave propagation with high accuracy, even in the presence of sparse and noisy data. In the structured geometries, the models demonstrated accurate performance across different training sample sizes and noise levels, with a good accordance with previously reported RMSE values [13].

In the context of MRI-based geometries, PINNs proved capable of handling complex anatomical details, accurately simulating the effects of scar tissue and varying fiber orientations on electrical conduction. The anisotropic model experiments demonstrated the generalization capabilities of PINNs, showing only a slight increase in error during extrapolation beyond the training data range and providing fast simulation times in a fraction of a second once trained. These results underscore the significant advantages of using PINNs over traditional methods like FEM in terms of inference speed, adaptability, and computational efficiency.

Overall, this study confirms the feasibility of using PINNs for personalized cardiac electrophysiology simulations, paving the way for potential clinical

applications that require rapid and accurate simulations.

6.2 Limitations

Although the application of PINNs in cardiac electrophysiology presents several advantages, there are some limitations.

- We trained and tested our PINNs using highly specific simulation scenarios, with pre-specified geometries, biophysics models, electrical properties, and boundary and initial conditions. Though we demonstrated that PINNS can extrapolate to novel electrical conductivity values, there is much more work to be done with regards to Cardiac EP PINN generalizability.

- The training process for PINNs is computationally expensive, requiring dedicated GPU resources and long training times $(6 + \mathrm{h})$ for the anisotropic MRI-based model. The memory requirement when calculating the residuals was a substantial limitation, requiring around 40Gb of dedicated GPU-memory.

- Looking into the variability of the models reveals high variation when data are both sparse and noisy. In clinical settings, consistently obtaining high-quality, high-resolution data can be challenging, which may affect the robustness and reliability of the models.

- For clinical applications where decisions based on model predictions can directly impact patient outcome, quantifying the uncertainty of these predictions is essential. Developing methods to quantify uncertainty in PINN predictions can enhance the reliability and safety of these models in clinical practice.

- The current ionic current model uses the recovery variable, W, which isn't measurable in practice.

- The objective of this study was to explore the potential of PINNs within cardiac EP simulations as a proof of concept, and as such we did not

conduct an extensive hyperparameter search. The performance of PINNs is highly sensitive to the choice of hyperparameters and determining the optimal set involves navigating a vast search space. The slow training process complicates this task further, especially when employing traditional methods like grid or random search. However, there are more advanced techniques, such as Bayesian optimization, that leverage prior knowledge from previous trials to explore the search space more efficiently, though these methods were not fully explored.

- Neural networks, including PINNs, often act as "black boxes," making it difficult to interpret their internal workings and the reasons behind their predictions. This lack of interpretability can be a significant barrier to clinical adoption, where understanding the rationale behind a model's prediction is essential.

- Small sample sizes when testing variability, especially in the anisotropic case (only 5 samples).

6.3 Future Work

The promising results of this study open several avenues for future research in the field of computational cardiology using PINNs.

Expanding the current 2D MRI-based models to three-dimensional simulations will offer a more thorough and anatomically precise depiction of cardiac electrophysiology. Within the existing framework, this extension can be readily achieved by adding an input node and modifying the diffusion term to accommodate 3D spatial derivatives.

Integrating cardiac electrophysiology with cardiac mechanics can provide a more complete model of the human heart.

In the presence of electrically disturbing scars, pathological electrophysiological waves called reentry can form. Reproducing such waves with a PINN has not been performed in MRI-based geometries. Future work should include modeling reentry

in MRI-based geometries.

Applying PINNs to inverse problems, such as identifying scar regions or potential ablation sites, could significantly aid in the diagnosis and treatment of arrhythmias.

Appendix A

Effects of input scaling and loss balance

In this chapter, we analyze and discuss the impact of input scaling and balancing the loss terms on the performance four MRI-based anisotropic PINN model.

A.1 Input Scaling

As discussed in 3.4, input scaling is an important step in training neural networks, ensuring that input features contribute equally during training. Figure A.1 shows the convergence of the training loss components of an anisotropic and heterogeneous MRI-based model. We see most (apart from $\mathcal{L}_{IC}$) loss components plateauing at higher values compared to 4.11. The final values of the loss components are summarized in TableA.1.

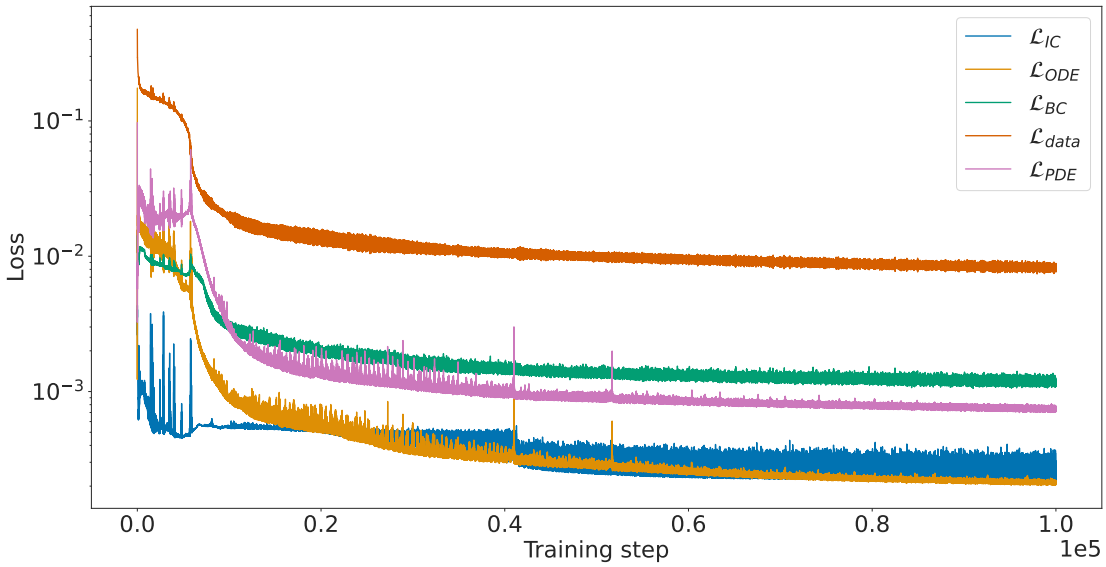

Figure A.1: Anisotropic and heterogeneous MRI-based model training loss convergence during training on a logarithmic scale. The plot shows the evolution of different components of the total loss when the model is trained without input scaling.

component	With scaling	Without scaling
$\mathcal{L}_{PDE}$	3.7×10^{-4}	7.1×10^{-4}
$\mathcal{L}_{ODE}$	7.0×10^{-5}	2.1×10^{-4}
$\mathcal{L}_{BC}$	1.6×10^{-4}	1.2×10^{-3}
$\mathcal{L}_{IC}$	2.5×10^{-4}	2.2×10^{-4}
$\mathcal{L}_{data}$	3.4×10^{-3}	7.7×10^{-3}

Table A.1: Final training losses with and without input scaling

Figure A.2 shows the relative L^1 error when training without scaling the inputs. The plot shows higher errors in general, compared to scaling (Figure 4.14) as well as an increase of the error for higher conductivities. Considering the difference in magnitudes of the inputs (the largest conductivity values being of two orders of magnitude smaller than the largest spatio-temporal coordinate values), this suggests that the conductivity inputs become less significant during training due to the absence of input scaling.

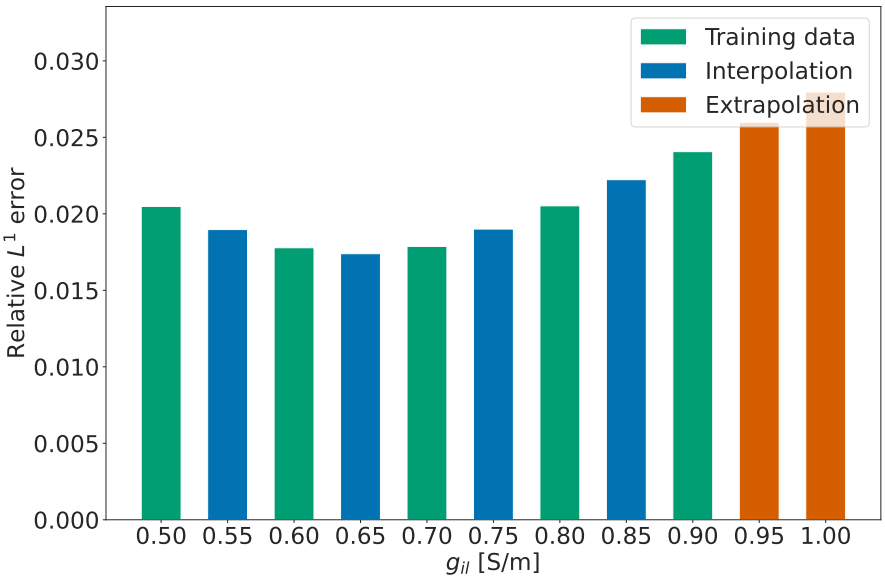

Figure A.2: Relative L^1 error computed from MRI-based model with anisotropic and heterogeneous conductivities trained without input scaling.

A.2 Loss balance

As discussed in 3.5, balancing the contribution of each loss component during training is cricial when the different components have different magnitudes. By balancing these losses, we ensure that no single component dominates the optimization process.

Figure A.3 shows the convergence of the training loss components of an anisotropic and heterogeneous MRI-based model. We see all loss components converging faster, but plateauing at higher values compared to 4.11. The final values of the loss components are summarized in TableA.2.

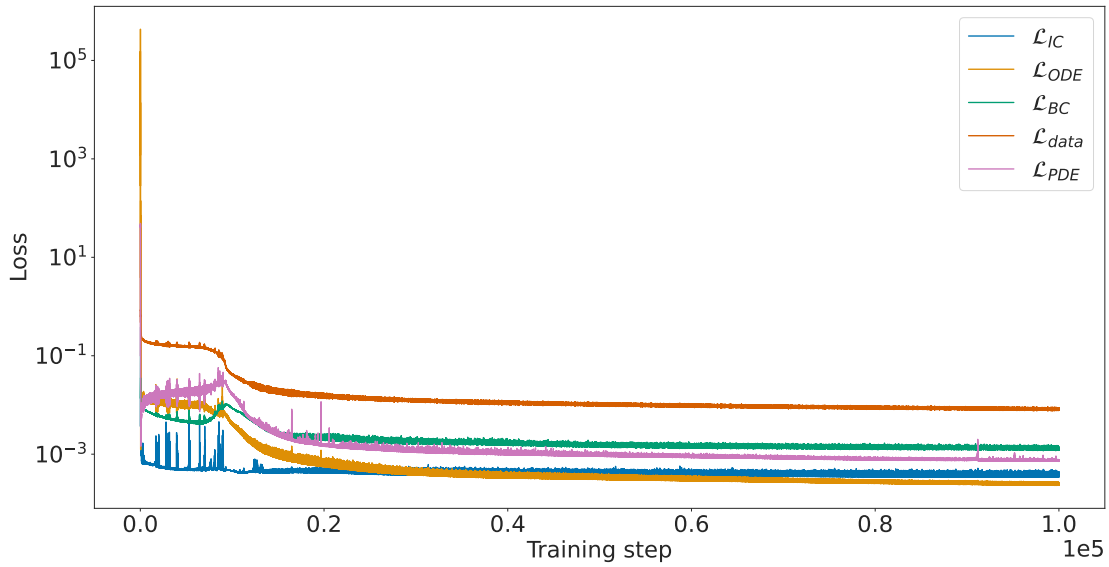

Figure A.3: Anisotropic and heterogeneous MRI-based model training loss convergence during training on a logarithmic scale. The plot shows the evolution of different components of the total loss when the model is trained with all loss weights set to one ($\lambda_i = 1$).

component	ReLoBRaLo	Without ReLoBRaLo
$\mathcal{L}_{PDE}$	3.7×10^{-4}	7.6×10^{-4}
$\mathcal{L}_{ODE}$	7.0×10^{-5}	2.7×10^{-4}
$\mathcal{L}_{BC}$	1.6×10^{-4}	1.3×10^{-3}
$\mathcal{L}_{IC}$	2.5×10^{-4}	3.5×10^{-4}
$\mathcal{L}_{data}$	3.4×10^{-3}	8.1×10^{-3}

Table A.2: Final training losses with and without loss term balancing (ReLoBRaLo)

Figure A.4 illustrates the relative L^1 errors for the model trained without balancing the loss terms. The errors are less consistent at the lower en of $g_i l$ values, indication that the lack of loss term balancing results in less reliable performance.

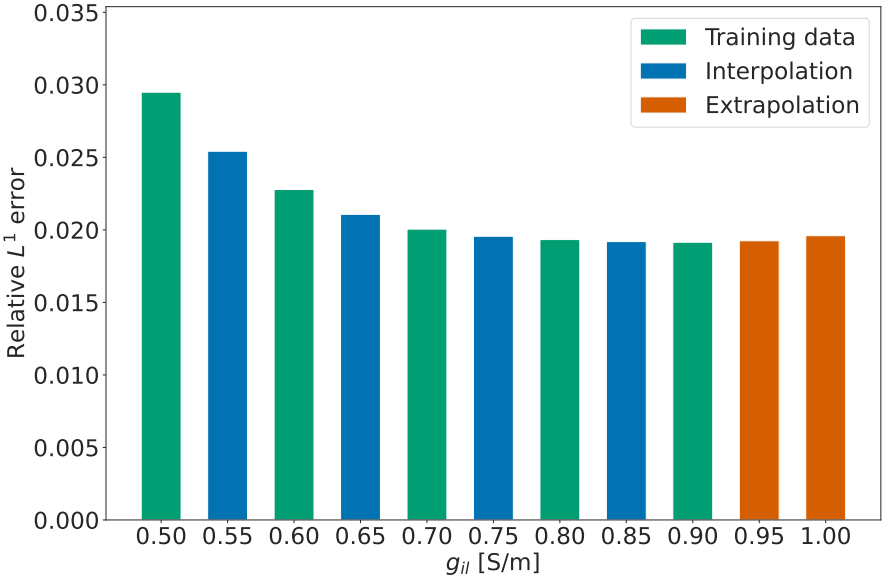

Figure A.4: Relative L^1 error computed from MRI-based model with anisotropic and heterogeneous conductivities trained with $\lambda_i = 1$.

Appendix B

Computing in Cardiology Conference Abstract Submission

Enhancing 2D Patient Specific Electrophysiology with Physics-Informed Neural Networks

Adam Jakobsen, Vajira Thambawita, Thu Nguyen, Mary M Maleckar, Gabriel Balaban

University of Oslo
Oslo, Norway

Background: Neural networks, once trained, can offer predictions for personalized cardiac electrophysiology (EP) within a very short timeframe. Physics-Informed Neural Networks (PINNs) can combine the theoretical knowledge of a physical system with data, presenting a promising method for personalized electrophysiology simulations.

Methods: We have developed a PINN model that utilizes spatial and temporal coordinates along with local conductivities as input parameters to predict the spatio-temporal evolution of action potentials. 2D magnetic resonance images were used to establish a computational domain and infer local conductivities, providing a proof of concept for more detailed personalized 3-D models.

Results: Our study illustrates the capability of PINNs to precisely recreate action potential dynamics from limited in-silico voltage data. Importantly, we show that trained PINNs can accurately interpolate and extrapolate with local conductivity inputs beyond the range of training data and make predictions at a significantly reduced time-frame.

Inference run-time

	FEM	PINN
CPU	684 ± 60 s	3.62 ± 0.09 s
GPU	-	0.22 ± 0.01 s

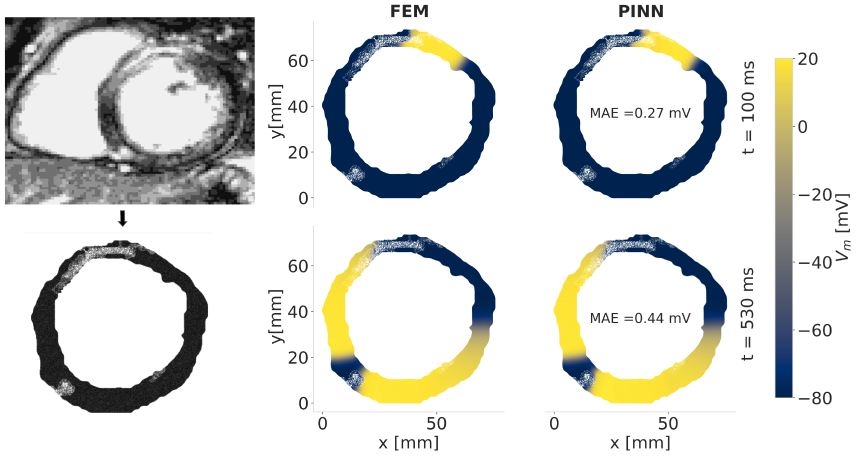

Figure 1. Computational model created from MRI and PINN-predicted transmembrane potential, V_m, compared to the corresponding Finite Elements Method solution.

Bibliography

[1] Rubin R. Aliev and Alexander V. Panfilov. "A simple two-variable model of cardiac excitation." In: *Chaos, Solitons Fractals* 7.3 (1996), pp. 293–301. ISSN: 0960-0779. DOI: https://doi.org/10.1016/0960-0779(95)00089-5. URL: https://www.sciencedirect.com/science/article/pii/0960077995000895.

[2] Gabriel Balaban et al. "Fibrosis Microstructure Modulates Reentry in Non-ischemic Dilated Cardiomyopathy: Insights From Imaged Guided 2D Computational Modeling." In: *Frontiers in Physiology* 9 (2018). ISSN: 1664-042X. DOI: 10.3389/fphys.2018.01832. URL: https://www.frontiersin.org/journals/physiology/articles/10.3389/fphys.2018.01832.

[3] J. D. Bayer et al. "A Novel Rule-Based Algorithm for Assigning Myocardial Fiber Orientation to Computational Heart Models." In: *Annals of Biomedical Engineering* 40.10 (May 2012), pp. 2243–2254. ISSN: 1573-9686. DOI: 10.1007/s10439-012-0593-5. URL: http://dx.doi.org/10.1007/s10439-012-0593-5.

[4] Rafael Bischof and Michael Kraus. "Multi-Objective Loss Balancing for Physics-Informed Deep Learning." en. In: (2021). DOI: 10.13140/RG.2.2.20057.24169. URL: http://rgdoi.net/10.13140/RG.2.2.20057.24169.

[5] *Computing the electrical activity in the heart*. eng. Vol. 1. Monographs in computational science and engineering. Berlin: Springer, 2006. ISBN: 9783540334323.

[6] J. Kevin Donahue, Jonathan Chrispin, and Olujimi A. Ajijola. "Mechanism of Ventricular Tachycardia Occurring in Chronic Myocardial Infarction Scar." In: *Circulation Research* 134.3 (2024), pp. 328–342. DOI: 10.1161/

CIRCRESAHA.123.321553. eprint: https://www.ahajournals.org/doi/pdf/10.1161/CIRCRESAHA.123.321553. URL: https://www.ahajournals.org/doi/abs/10.1161/CIRCRESAHA.123.321553.

[7] Xavier Glorot and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Yee Whye Teh and Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 249–256. URL: https://proceedings.mlr.press/v9/glorot10a.html.

[8] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. Cambridge, MA, USA: MIT Press, 2016.

[9] David J Griffiths. *Introduction to electrodynamics*. Pearson, 2013.

[10] Tamara G. Grossmann et al. *Can Physics-Informed Neural Networks beat the Finite Element Method?* 2023. arXiv: 2302.04107 [math.NA].

[11] Ankur Gulati et al. "Association of Fibrosis With Mortality and Sudden Cardiac Death in Patients With Nonischemic Dilated Cardiomyopathy." In: *JAMA* 309.9 (Mar. 2013), pp. 896–908. ISSN: 0098-7484. DOI: 10.1001/jama.2013.1363. eprint: https://jamanetwork.com/journals/jama/articlepdf/1660382/joc130017_896_908.pdf. URL: https://doi.org/10.1001/jama.2013.1363.

[12] John E Hall. *Guyton and Hall textbook of medical physiology*. eng. Philadelphia, 2011.

[13] Clara Herrero Martin et al. "EP-PINNs: Cardiac Electrophysiology Characterisation Using Physics-Informed Neural Networks." In: *Frontiers in Cardiovascular Medicine* 8 (2022). ISSN: 2297-055X. DOI: 10.3389/fcvm.2021.768419. URL: https://www.frontiersin.org/articles/10.3389/fcvm.2021.768419.

[14] A L Hodgkin and A F Huxley. “Propagation of electrical signals along giant nerve fibers.” en. In: *Proc. R. Soc. Lond. B Biol. Sci.* 140.899 (Oct. 1952), pp. 177–183.

[15] Arnold M Katz. *Physiology of the heart*. eng. 5th ed. Philadelphia: Wolters Kluwer/Lippincott Williams Wilkins Health, 2011. ISBN: 9781608311712.

[16] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[17] G.T. Lines et al. “Mathematical models and numerical methods for the forward problem in cardiac electrophysiology.” In: *Computing and Visualization in Science* 5.4 (July 2002), pp. 215–239. ISSN: 1433-0369. DOI: 10.1007/s00791-003-0101-4. URL: http://dx.doi.org/10.1007/s00791-003-0101-4.

[18] Dong C. Liu and Jorge Nocedal. “On the limited memory BFGS method for large scale optimization.” In: *Mathematical Programming* 45.1–3 (Aug. 1989), pp. 503–528. ISSN: 1436-4646. DOI: 10.1007/bf01589116. URL: http://dx.doi.org/10.1007/BF01589116.

[19] Lu Lu et al. “DeepXDE: A deep learning library for solving differential equations.” In: *SIAM Review* 63.1 (2021), pp. 208–228. DOI: 10.1137/19M1274067.

[20] Iulia Nazarov et al. “Physics-Informed Fully Connected and Recurrent Neural Networks for Cardiac Electrophysiology Modelling.” In: *2022 Computing in Cardiology (CinC)*. Vol. 498. 2022, pp. 1–4. DOI: 10.22489/CinC.2022.188.

[21] Denis Noble, Alan Garny, and Penelope J Noble. “How the Hodgkin-Huxley equations inspired the Cardiac Physiome Project.” en. In: *J. Physiol.* 590.11 (June 2012), pp. 2613–2628.

[22] openCARP consortium et al. *openCARP*. Version v15.0. 2024. DOI: 10.35097/1979. URL: https://git.opencarp.org/openCARP/openCARP.

[23] World Health Organisation. *Cardiovascular diseases*. June 2021. URL: https://www.who.int/health-topics/cardiovascular-diseases#tab=tab_1.

[24] A.V. Panfilov, H. Dierckx, and V. Volpert. "(INVITED) Reaction–diffusion waves in cardiovascular diseases." In: *Physica D: Nonlinear Phenomena* 399 (2019), pp. 1–34. ISSN: 0167-2789. DOI: https://doi.org/10.1016/j.physd.2019.04.001. URL: https://www.sciencedirect.com/science/article/pii/S0167278918305876.

[25] Adam Paszke et al. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. 2019. arXiv: 1912.01703 [cs.LG].

[26] Mihaela Pop et al. "Correspondence Between Simple 3-D MRI-Based Computer Models and In-Vivo EP Measurements in Swine With Chronic Infarctions." In: *IEEE Transactions on Biomedical Engineering* 58.12 (2011), pp. 3483–3486. DOI: 10.1109/TBME.2011.2168395.

[27] M. Raissi, P. Perdikaris, and G.E. Karniadakis. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations." In: *Journal of Computational Physics* 378 (2019), pp. 686–707. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2018.10.045. URL: https://www.sciencedirect.com/science/article/pii/S0021999118307125.

[28] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." In: *Nature* 323.6088 (Oct. 1986), pp. 533–536. ISSN: 1476-4687. DOI: 10.1038/323533a0. URL: http://dx.doi.org/10.1038/323533a0.

[29] Francisco Sahli Costabal et al. "Physics-Informed Neural Networks for Cardiac Activation Mapping." In: *Frontiers in Physics* 8 (2020). ISSN: 2296-424X. DOI: 10.3389/fphy.2020.00042. URL: https://www.frontiersin.org/articles/10.3389/fphy.2020.00042.

[30] Shyam Sankaran et al. "On the impact of larger batch size in the training of Physics Informed Neural Networks." In: *The Symbiosis of Deep Learning*

and Differential Equations II. 2022. URL: https://openreview.net/forum?id=THCvohg1RV.

[31] TOYOMI SANO, NORIYUKI TAKAYAMA, and TAKIO SHIMAMOTO. "Directional Difference of Conduction Velocity in the Cardiac Ventricular Syncytium Studied by Microelectrodes." In: *Circulation Research* 7.2 (Mar. 1959), pp. 262–267. ISSN: 1524-4571. DOI: 10.1161/01.res.7.2.262. URL: http://dx.doi.org/10.1161/01.RES.7.2.262.

[32] Natalia A. Trayanova et al. "Computational cardiology: how computer simulations could be used to develop new therapies and advance existing ones." In: *EP Europace* 14.suppl$_5$ (Nov. 2012), pp. v82–v89. ISSN: 1099-5129. DOI: 10.1093/europace/eus277. eprint: https://academic.oup.com/europace/article-pdf/14/suppl_5/v82/7278648/eus277.pdf. URL: https://doi.org/10.1093/europace/eus277.

[33] Leslie Tung. "A bi-domain model for describing ischemic myocardial d-c potentials." ndltd.org (oai:dspace.mit.edu:1721.1/16177). PhD thesis. Massachusetts Institute of Technology, Cambridge, MA, USA, 1978.

[34] Sifan Wang et al. *An Expert's Guide to Training Physics-informed Neural Networks*. 2023. arXiv: 2308.08468 [cs.LG].

[35] Yanzhao Wu et al. *Demystifying Learning Rate Policies for High Accuracy Training of Deep Neural Networks*. 2019. arXiv: 1908.06477 [cs.LG].

[36] Jianxin Xie and Bing Yao. "Physics-constrained deep active learning for spatiotemporal modeling of cardiac electrodynamics." In: *Computers in Biology and Medicine* 146 (2022), p. 105586. ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2022.105586. URL: https://www.sciencedirect.com/science/article/pii/S001048252200378X.

[37] Jianxin Xie and Bing Yao. "Physics-Constrained Deep Learning for Robust Inverse ECG Modeling." In: *IEEE Transactions on Automation Science and Engineering* 20.1 (2023), pp. 151–166. DOI: 10.1109/TASE.2022.3144347.

[38] Minglang Yin et al. “PO-01-212 A NOVEL DEEP LEARNING MODEL FOR PATIENT-SPECIFIC COMPUTATIONAL MODELING OF CARDIAC ELECTROPHYSIOLOGY.” In: *Heart rhythm.* 20.5 (2023), S163. ISSN: 1547-5271.

[39] Jingzhao Zhang et al. *Why gradient clipping accelerates training: A theoretical justification for adaptivity.* 2020. arXiv: 1905.11881 [math.OC].