Politecnico di Milano
SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE
Master Degree Course in *Mathematical Engineering*

École Polytechnique Fédérale de Lausanne
SCHOOL OF BASIC SCIENCES
Master Degree Course in *Computational Science and Engineering*

PDE-aware Deep learning for Inverse Problems in Cardiac Electrophysiology

MASTER THESIS

SUPERVISORS

Prof. Alfio Maria Quarteroni
Prof. Simone Deparis

CO-SUPERVISOR

PhD. Stefano PAGANI

CANDIDATE

Riccardo TENDERINI
Polimi ID : 892249
Sciper Nº : 298776
riccardo.tenderini@mail.polimi.it
riccardo.tenderini@epfl.ch

Academic Year 2019-2020
October 2, 2020

A mia madre

Acknowledgments

Ringrazio i miei relatori *Prof. Alfio Quarteroni* e *Prof. Simone Deparis* per avermi assistito con attenzione e continuità durante questo progetto di tesi, fornendomi sempre tutte le nozioni e gli strumenti necessari per portare a compimento il lavoro. É stato un piacere ed un onore poter collaborare con voi. Il mio ringraziamento si fa poi ancor piú sentito in considerazione delle particolari e sfortunate circostanze che hanno caratterizzato gli ultimi mesi.

Un enorme ringraziamento va al mio correlatore *PhD. Stefano Pagani* per avermi proposto questo bellissimo progetto, per avermi accompagnato ed assistito in ogni fase del suo sviluppo e per aver dipanato tutti i miei dubbi e risolto tutti i miei problemi con prontezza ed efficacia. Sono consapevole di aver rubato una grande parte del suo tempo negli ultimi mesi, ma spero che la buona riuscita del lavoro possa ripagarlo di ció.

Ringrazio anche tutti i membri del progetto *iHeart*; i meeting a cui ho preso parte sono stati occasione di arricchimento umano e professionale, nonchè sede di un confronto costruttivo, rivelatosi molto utile per portare a termine il progetto. Inoltre, desidero ringraziare il *Dr. Antonio Frontera* dell'IRCCS Ospedale San Raffaele di Milano per la disponibilità e per le preziose indicazioni datami circa possibili sviluppi futuri del lavoro.

Desidero ringraziare tutti i miei compagni di corso, e in modo particolare Federico, Filippo, Lucia, Nicola ed Alessandra. Frequentare università prestigiose come il Politecnico di Milano e l'EPFL richiede tanto impegno e numerosi sacrifici, ma poterlo condividere con voi lo ha reso certamente piú piacevole, arricchendolo di uno spessore umano che va ben oltre le mere nozioni scientifiche. Agli ultimi due membri della lista va poi un rigraziamento ancor piú sentito, per aver condiviso con me ogni esperienza in terra elvetica e per avermi sopportato (e supportato) giorno e notte per 18 lunghi mesi.

Un grandissimo ringraziamento va a tutti i miei amici che, pur nella mia prolungata lontananza da casa, mi hanno fatto sempre sentire la loro vicinanza e il loro supporto. In particolare, un ringraziamento dal cuore va a Fabio, Sirio e Stefano, amici da una vita; sono stato lontano per un po' e lo saró per altri 3 anni, ma con voi ogni distanza si azzera ed è come se vi avessi sempre al mio fianco.

Un ringraziamento speciale va a tutta la mia famiglia e in particolare a mia sorella Giulia, a mio padre Giancarlo e a mia madre Laura (senza dimenticare la piccola Moon). Il modo in cui mi avete sostenuto, incoraggiato, spronato, consolato in tutti questi anni, ed ancor di piú in questo ultimo difficile periodo, è incredibile e non smetteró mai di esservene grato. Vi voglio bene! A te poi, mamma, ho voluto dedicare questo lavoro; sei per me il migliore esempio in tutto ció che fai e la forza e il coraggio con cui stai affrontando le avversitá della vita sono il piú grande insegnamento che mi possa mai essere impartito.

Dulcis in fundo volevo ringraziare la mia fidanzata Valentina. Sei senza ombra di dubbio la parte migliore di me, colei che sa farmi riflettere quando tendo a seguire l'istinto, colei che sa liberarmi dalle prigioni della mente quando tendo ad intrappolarmi nel pensiero, colei che porta la luce quando mi adombro e il suono quando mi ovatto su me stesso. Mi rendo conto che stare al mio fianco non sia un mestiere facile (e piú mi invecchio, peggio è), ma tu lo fai con una gioia impareggiabile, contagiandomi ogni giorno con il tuo entusiasmo.

Riccardo

Abstract

The last decades have been characterized by the explosive growth of available data and computational resources. This has led to stunning advances in the field of Deep Learning and also to significant improvements in the numerical approximation of physical problems via PDEs. Despite their parallel success, the two fields have maintained a clean separation margin, at least until the last few years; recently, indeed, some works addressing an integration between those have started to appear and PDE-aware DL models have been introduced. These models proved to be successful in the so-called *small data regime*, i.e. in frameworks characterized by a scarcity of precise and easily retrievable data, which is anyway compensated by proper knowledge of some of the physical laws that govern the problem of interest. A context of this kind is the one of the Inverse Problem of Electrocardiography, whose aim is to estimate the electric potential at the epicardium from its knowledge in a discrete set of points in the torso. Such problem has become central in contemporary biomedical research, configuring as the theoretical basis of ECGI, but, due to its ill-posedness, all State-Of-Art numerical algorithms feature severe weaknesses. Because of this, we present here a PDE-aware DL model, named Space-Time Reduced-Basis Deep Neural Network (ST-RB-DNN), which estimates the solution to the Inverse Potential Problem both leveraging available data and exploiting the knowledge of some underlying physical laws. This last goal is carried out in two ways: predicting the epicardial potential as projected onto a Space-Time-Reduced subspace, generated from the training dataset, and inserting a RB-solver of the Forward Problem within the network architecture. The project proposes itself as a methodological analysis and additional efforts should be made in order to apply it in the clinical setting. Several numerical tests have been conducted, both on a simplified test case, employing idealized geometries and using 12-lead ECG signals as input, and in a slightly more realistic setting, making use of human-shaped geometries and processing surface potentials, recorded by electrodes placed on the torso.

Keywords: Deep Learning, Partial Differential Equations, PDE-aware Deep Learning, Cardiac Electrophysiology, Inverse Problem of Electrocardiography, ECGI

Sommario

Gli ultimi decenni sono stati caratterizzati da un'enorme crescita sia della quantità di dati utilizzabili che di risorse computazionali. Ció ha portato ad incredibili sviluppi nel campo del *Deep Learning*, nonché a significativi miglioramenti nell'approssimazione numerica di fenomeni fisici, tramite PDE. Nonostante i contemporanei progressi, tra i due campi è stato sempre mantenuto un chiaro margine di separazione; nell'ultimo decennio, tuttavia, sono stati realizzati diversi lavori mirati ad una loro integrazione ed i *PDE-aware DL models* sono stati introdotti. Tali modelli si sono dimostrati efficaci nel cosiddetto regime degli *small data*, i.e. in contesti caratterizzati da una povertà di dati precisi e facilmente misurabili, la quale viene peró compensata dalla conoscenza di alcune delle leggi fisiche governanti il fenomeno d'interesse. Un esempio in tal senso è offerto dal Problema Inverso dell'Elettrocardiologia, il cui obiettivo è stimare il potenziale all'epicardio a partire dalla sua conoscenza in alcuni punti del torso. Tale problema è divenuto centrale nella ricerca biomedica contemporanea, configurandosi come la base teorica dell'ECGI, ma gli algoritmi numerici correntemente utilizzati sono caratterizzati da significative lacune. In ragione di ció, presentiamo un *PDE-aware DL model*, chiamato *Space-Time Reduced-Basis Deep Neural Network* (ST-RB-DNN), che stima la soluzione del Problema Inverso dell'Elettrocardiologia sfruttando sia l'abbondanza di dati sia la conoscenza di alcune delle leggi fisiche sottostanti. Quest'ultimo obiettivo è realizzato per mezzo di due elementi: la predizione del potenziale epicardico proiettato su un sottospazio ridotto sia in spazio sia in tempo, generato a partire dal *training dataset*, e l'inserimento di un solutore RB del Problema Diretto nell'architettura della rete. Il progetto si propone come un'analisi metodologica. Vari test numerici sono stati condotti, sia in un caso *benchmark* semplificato, impiegando geometrie idealizzate ed utilizzando segnali *12-lead ECG* come input, sia in un contesto piú realistico, facendo uso di migliori geometrie e processando 158 segnali, misurati tramite elettrodi posizionati sul torso.

Parole Chiave: Deep Learning, Equazioni alle Derivate Parziali, PDE-aware Deep Learning, Elettrofisiologia Cardiaca, Problema Inverso dell'Elettrocardiografia, ECGI

Contents

List of Figures

List of Tables

List of Abbreviations

Adam adaptive moment estimation 17, 96, 107, 114, 126
AGD accelerated gradient descent 114, 126
AI artificial intelligence 11, 15, 140
ALV anterior left ventricle 45, 106, 112
ANN artificial neural network 15
AP Aliev-Panfilov 56, 58, 59, 65, 67, 68, 78, 104–106, 139
APD action potential duration 56
ARV anterior right ventricle 45, 106, 112, 125
AV atrio-ventricular 42–44, 50, 144, 146
aVF augmented vector foot 47, 48, 51, 113, 125
aVL augmented vector left 47, 48, 51
aVR augmented vector right 47–49, 51, 113, 125, 131

BDF backward differentiation formulas 38–40
BiCG bi-conjugate gradient 62
BiCGSTAB bi-conjugate gradient stabilized 62, 64
BSPM body surface potential map 82, 83, 88, 117, 131, 134, 139

CD coordinate descent 17
CL cycle length 56
CNN convolutional neural network 16–19, 31, 33, 38, 100
CRNN convolutional recurrent neural network 92, 140
CT computed tomography 11, 82

DEIM discrete empirical interpolation method 22, 32–35, 37
DFT discrete Fourier transform 87, 91, 98, 101–104, 119–130, 134–136, 138, 152–154
DIF decimation in frequency 154
DIT decimation in time 153, 154
DL deep learning 11–13, 15, 17, 18, 20, 29–32, 41, 52, 65, 72, 76, 79, 82, 86–88, 91, 99, 102, 104, 107, 131, 137, 139, 140, 149
DNN deep neural network 15, 16, 31
DOF degree of freedom 23–26, 33, 37, 64, 65, 72, 75–77, 89, 96, 100, 113, 139

EBT epicardial breakthrough 44, 45, 69, 106, 112, 117, 125, 131, 134
ECG electrocardiogram 6, 11, 13, 18, 31, 41, 44–51, 58, 65, 67–71, 78, 79, 81, 82, 88, 91, 98–101, 104, 105, 107, 111–113, 116–119, 123–125, 127–136, 138, 140
ECGI electrocardiographic imaging 11, 13, 82, 88, 91, 92, 104, 117
EIM empirical interpolation method 22, 32–35, 37
EP electrophysiology 12, 26, 41, 45, 52, 53, 55, 56, 58, 59, 61, 65, 66, 69, 75, 79, 82, 90–92, 104, 105, 137, 140

FE finite elements 12, 13, 20, 22, 35, 59, 61, 62, 64, 65, 83–85
FFT fast fourier transform 102, 103, 153, 154
FHN FitzHugh Nagumo 56, 60

FOM full order model 13, 20–24, 26, 27, 32–35, 37–40, 59, 65, 72, 75–77, 79–81, 89, 90, 93, 96, 100, 106, 113, 117, 130, 131, 137, 138

GD gradient descent 17
GMRes generalized minimal residual 86
GSVD generalized singular value decomposition 85, 88

IGA IsoGeometric Analysis 35
ILV inferior left ventricle 45, 106, 112, 125
IRV inferior right ventricle 45, 106

L-BFGS limited-memory Broyden-Fletcher-Goldfarb-Shanno 114, 126
LA left arm 46, 47
LBBB left bundle branch block 69, 71, 106
LEA latest epicardial activation 44, 45, 69, 96
LL left leg 46, 47

MAE mean absolute error 90, 112, 116, 127, 128, 130, 138
ML machine learning 15, 29, 92
MLP multiple layer perceptron 16–19, 31, 33–37, 92, 101, 102, 124
MOR model order reduction 83, 89
MSE mean squared error 34, 38, 90, 112, 116, 127, 128, 130

Nadam Nesterov adaptive moment estimation 107, 114, 126, 127
NLP natural language processing 11, 18, 30
NN neural network 12, 15, 17, 30, 31, 33–35, 37, 38, 72, 76, 88–96, 98–102, 107, 113, 116, 137, 140

ODE ordinary differential equation 11, 29, 31, 63, 64, 67, 137, 139, 151, 153

PCA principal component analysis 147
PDE partial differential equation 11–13, 15, 20–24, 26, 27, 29–35, 37, 38, 41, 52, 64, 65, 67, 72, 76, 82, 86, 87, 99, 137, 139, 140, 151, 153, 155
PINN physically informed neural network 29, 31, 33, 37, 114
POD proper orthogonal decomposition 6, 7, 15, 21, 23–26, 35, 36, 72–81, 90, 93, 107, 113, 114, 119, 124–126, 130, 137, 147

RA right arm 46, 47
RB reduced basis 15, 20–26, 32–40, 52, 72, 75, 76, 89–91, 93, 95, 97, 98, 100–103, 112, 113, 124, 139, 140
RB-CNN reduced basis convolutional neural network 35, 37–40, 89, 99, 140
RB-DNN reduced basis deep neural network 12, 13, 29, 31–38, 88, 93, 95
RBBB right bundle branch block 69, 106
RL right leg 46
RMSProp root mean squared propagation 17, 114, 126
RNN recurrent neural network 18, 31, 33
ROM reduced order modeling 20, 22, 31, 32, 34, 35, 37, 41, 52, 72, 77–81, 89, 90, 94, 96, 97, 107, 109–111, 113, 118, 119, 121–123, 129–137

SA sino-atrial 43, 44
SGD stochastic gradient descent 17, 95, 96, 114, 126
SL semi-lunar 43, 44, 144
SNR Signal-to-Noise ratio 99, 106, 140
ST-LSPG Space-Time Least-Squares Petrov-Galerkin 24, 26, 37–39, 72
ST-RB-DNN Space-Time reduced basis deep neural network 13, 87–92, 94, 96, 98, 100–104, 107, 109–111, 113–140
SVD singular value decomposition 21, 24, 25, 73, 74, 85, 147–149

Chapter 1

Introduction

The 21^{st} century has been unarguably characterized by the stunning growth of machine learning and data analytics, which has been made possible thanks to the abundance of available data and to the progresses in terms of computational power and resources. In particular, Deep Learning (DL) and Artificial Intelligence (AI) have established themselves as pillars of the contemporary scientific development and they have already yielded several breakthrough results in a variety of fields, as image recognition [3], text classification [4] or Natural Language Processing (NLP) [5] just to name a few. Beside of DL progresses, recent years have also witnessed significant advances in the numerical approximation of Partial Differential Equations (PDEs) and Ordinary Differential Equations (ODEs), which have led to an overall consolidation of such field. Indeed, nowadays many physical phenomena can be simulated employing classical numerical methods in a variety of fields (from thermodynamics to fluid mechanics, from electromagnetism to quantum physics) and incredible levels of accuracy have been reached, again thanks to the improvements in terms of computational power and also to the enhancement of imaging techniques.

Despite the parallel upgrades of the two aforementioned fields, few attempts aimed at integrating them have been performed, at least until the last decade. On the one side, indeed, the lack of solid theoretical foundations of DL has hindered its use in the well-established field of numerical analysis; on the other side, instead, the efficient application of classical numerical methods within the DL framework appeared to be a non-trivial task. In the very recent years, anyway, several works aimed at merging DL and classical numerical methods have started to appear (see [1, 2, 6–9] for instance). The general idea is that numerical methods, which are able to encode the physics of the phenomena of interest, can ease the learning process of DL models, at least in frameworks where the problem dynamics are well-known and can be precisely simulated, whereas the availability of precise and non-invasively measured data is scarce (i.e. the so-called *small data regime*). In simpler terms, equipping a DL-based model with the knowledge of the physical laws underlying the phenomenon at hand should ease its ultimate predictive task, especially in contexts where data abundance cannot be exploited.

A relevant instance of *small data regime* is the one of the Inverse Problem of Electrocardiography (see Section 4.3), which aims at estimating the electric potential at the heart epicardial surface from the knowledge of its value in a discrete set of points in the torso. Indeed, in order to train a classical DL model in this context, it would be necessary to collect a huge amount of data, made of measurements of the electric potential both over the torso (easy collectable via non-invasive techniques) and at the epicardium (hardly collectable with invasive procedures); in actual practice assembling such a dataset is unfeasible. The important role of the Inverse Problem of Electrocardiography in nowadays biomedical research is justified by the fact that, as extensively presented in [10], it serves as theoretical basis for Electrocardiographic Imaging (ECGI), a novel imaging modality for non-invasive mapping of cardiac electrical activity, which makes use of body-surface ECG signals and of ECG-gated thoracic Computed Tomography (CT) scans. While the employment of classical DL models is prevented by data scarcity, the usage of classical numerical methods is the State-Of-Art in the field; anyway, as it will be made clear in

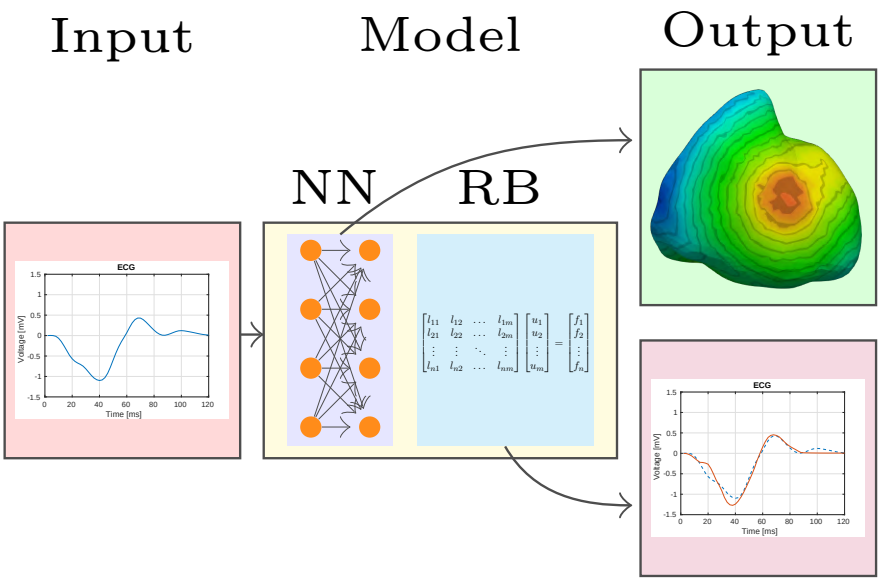


Figure 1.1: Basic structure of the developed PDE-aware DL model

Section 4.3, such methods are effective in approximating the Forward Problem (i.e. the estimation of the potential in the torso, being given the one at the epicardium), but they feature severe lacks and weaknesses when they come to the Inverse Problem, because of its ill-posed nature.

The Inverse Problem of Electrocardiography appears then as an ideal framework for taking advantage of the integration between DL models and classical methods for the numerical approximation of the cardiac electrical activity. In this project we made a first attempt in this direction, by developing a PDE-aware DL model, inspired by the Reduced Basis Deep Neural Networks (RB-DNNs) introduced by *Dal Santo et al.* in [2]; the basic structure can be visualized in Figure 1.1. Our model tries to approximate the solution of the Inverse Problem, by mapping signals recorded on (or inside) the torso to epicardial potentials (from which the activation map is lately derived, at post-processing stage). In doing this, it does not just exploit classical DL paradigms (encoded by the model section in blue, named "NN"), but it also takes advantage of the knowledge of the problem physics, which is involved by means of the numerical approximation of the Forward Problem. In particular, the Finite Elements (FE) approximations of the heart electrophysiology (EP) and of the Forward Problem of Electrocardiography are employed to:

1. Generate the training dataset, made of both the input signals (model section in red, named "Input"), measured on the torso, and of the epicardial potentials returned in output (model sections in green and purple, as a whole named "Output")
2. Generate a Reduced Basis in space for the potentials in the torso and a Reduced Basis in Space-Time for the ones at the epicardial surface. This configures as a fundamental aspect of the model, since it allows on the one side to lighten its complexity and on the other side to estimate the desired potential as projected onto a subspace generated from "physical" solutions, thus belonging by construction to a lower-dimensional and physically-consistent manifold.
3. Reconstruct the input signals from the estimated epicardial potentials within the network architecture, via an embedded tensorial Reduced Basis solver (encoded by the model section in cyan, named "RB"). In this way, the loss functional can be constructed as the weighted average between the error on the (reduced) epicardial potentials and the one on the reconstructed signals; the second contribution configures as a physically-aware regularization term and it allows to visualize the neural network (NN) as a deep autoencoder.

All the methodological details and the results of the numerical tests can be found in Chapter 5. What is important to underline from the very beginning is that this work proposes itself as a preliminary research and it is not oriented towards the development of models that can be used in the clinical setting. Because of this, we have restricted ourselves to simplified benchmark cases, employing idealized geometries and coarse meshes, making several simplifying assumptions and keeping the complexity of the DL architectures as low as possible. Several upgrades should be made in order to develop models that can be used in actual practice, especially at data generation stage; the main ones will be discussed in Section 6.2. Anyway, the reduced complexity of the proposed models and the limited amount of computational resources employed for their training (see Appendix C) suggest them to be potentially used with success even in more detailed and complex frameworks.

Project Outline

Chapter 2 is devoted to the presentation of the most important preliminary concepts, which are necessary for a full understanding of the analysis carried out alongside the report. In particular: Section 2.1 contains a brief description of Deep Neural Networks, with a more specific focus on Multiple Layer Perceptrons (Subsection 2.1.1) and on Convolutional Neural Networks (Subsection 2.1.2). Section 2.2 presents the Reduced Basis Method, which allows to efficiently solve parametrized PDEs (see Subsection 2.2.1) by projecting them onto a suitably constructed lower-dimensional subspace; the method is analyzed both in the stationary case (Subsection 2.2.2) and in the time-dependent one (Subsection 2.2.3).

Chapter 3 is dedicated to PDE-aware DL models, i.e. DL models that take advantage of the knowledge of the physics of the problem at hand (expressed via PDEs) to improve their learning capabilities. In particular: Section 3.1 offers an overview on the field of PDE-aware Neural Networks. Section 3.2 presents the RB-DNNs introduced in [2], focusing both on their structure and goals (Subsection 3.2.1) and on the numerical results that have been achieved (Subsection 3.2.2).

In Chapter 4, the heart electrophysiology is analyzed, both from the clinical point of view and from the one of its numerical approximation. In particular: Section 4.1 provides a clinical description of the cardiac electrical activity, describing the heart conduction system (Subsection 4.1.1) and the techniques used to measure and evaluate it, with particular focus on the ECGs (Subsection 4.1.2). Section 4.2 features the description of the models (Subsection 4.2.1) and of the numerical methods (Subsection 4.2.2) employed to approximate the heart electrophysiology and Forward Problem of Electrocardiography; also Subsection 4.2.3 presents the numerical results obtained adopting a Full Order Model (FOM) approximation of the heart electrophysiology (via the FE method), while Subsection 4.2.4 discusses how Model Order Reduction techniques can be put in place while solving the Forward Problem and presents some additional numerical results in this sense. Finally Section 4.3 is devoted to the Inverse Problem of Electrocardiography and to ECGI; specifically Subsection 4.3.1 provides the mathematical definition of the Inverse Potential Problem and discusses its main properties, while Subsection 4.3.2 presents the most widely employed numerical methods.

Chapter 5 presents the PDE-aware DL models (called Space-Time Reduced Basis Deep Neural Networks (ST-RB-DNNs)) that have been built in order to provide a physically-consistent and data-driven approximation of the solution to the Inverse Problem of Electrocardiography. In particular: Section 5.1 features a general presentation of ST-RB-DNN models, highlighting the main motivations that have driven us towards their development (Subsection 5.1.1) and providing a description of their general architecture (Subsection 5.1.2). Section 5.2 is devoted to the presentation of the models that have been actually tested in the project; specifically Subsections 5.2.1 and 5.2.2 present the two different ST-RB-DNN models that have been implemented, while Subsection 5.2.3 presents the numerical results obtained with such models, both on a simplified benchmark case, using ECG signals as input, and on a more realistic one, where Body Surface Potentials coming from 155 different electrodes are processed.

In Chapter 6 the conclusions are drawn. In particular: Section 6.1 offers a final summary, highlighting the main motivations and findings of the work, while Section 6.2 lists the most relevant limitations of the project and discusses some possible further developments.

Also, Appendix A offers a more detailed overview on the functionalities of the cardiovascular system; Appendix B presents other important numerical methods employed in the project as the (Randomized) Singular Value Decomposition and the Fast Fourier Transform Algorithms; Appendix C describes the computational environment in which the numerical tests have been carried out.

Finally, at the beginning of each Section of Chapters 2, 3 and 4 is reported a scheme analogous to the one of Figure 1.1, where the portions of the model that are related to the concepts discussed within the Section are highlighted. This should allow the reader to better orientate in the report, following the common thread.

Chapter 2

Preliminary Concepts

The aim of this short chapter is to provide a brief overview on Deep Neural Networks (DNNs) (Section 2.1) and on PODs and the Reduced Basis (RB) method (Section 2.2). We refer the interested reader to [11] for a more extensive and exhaustive explanation of DNNs and to [12] and [13] for a more detailed analysis of the RB method. Many of the concepts presented here, as well as the basic notation, are borrowed from [2], which sets as our starting point for the development of this project.

2.1 Deep Neural Networks

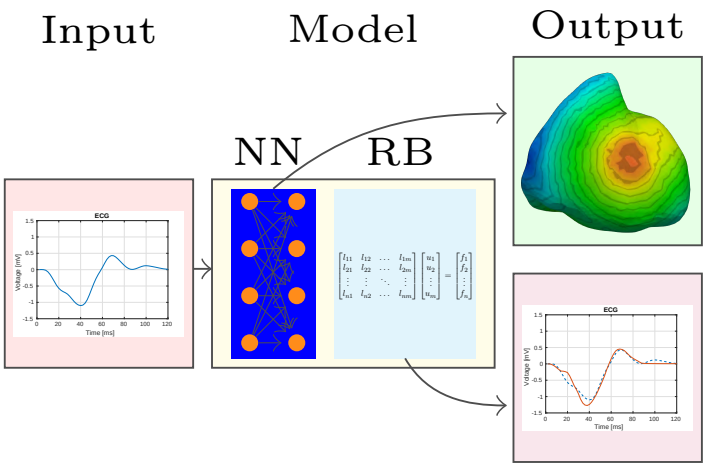


Figure 2.1: Basic structure of the developed PDE-aware DL model, where the NN is highlighted

From a very general point of view, Artificial Neural Networks (ANNs) are Machine Learning (ML) models that are designed to simulate the way the human brain analyzes and processes information; those are the foundation of AI and proved to be able to solve problems, in a variety of fields, that would prove otherwise impossible for standard ML models and even for humans. ANNs are characterized by having a *self-learning* ability, which makes them able to produce better and better results, as long as more and more data are available. The learning process occurs during a so-called *training* phase, where a set of *trainable parameters* is computed in a way that, given a dataset for which both input and output values are known, a precise reconstruction of the underlying I/O relationship can be derived.

Among ANNs, DNNs represent the widest subclass and they are characterized by the flanking of several non-linear modules, each of whom is composed by multiple units, called *neurons* or *nodes*. In particular, the *input layer* processes the raw data, the *output layer* gives the final

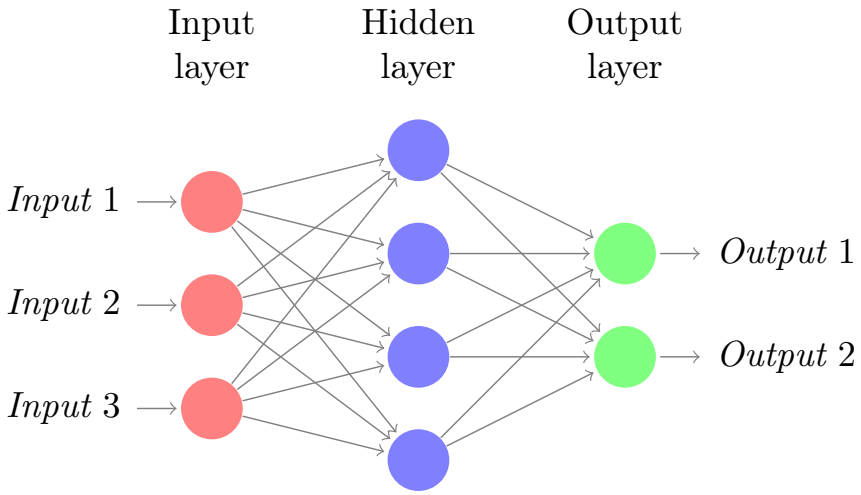


Figure 2.2: Structure of a MLP neural network with 1 hidden layer (so $L = 3$), $N_{in} = 3$ and $N_{out} = 2$. Image adapted from [2]

output and the internal *hidden layers* perform suitable non-linear operations on the data, so that an accurate functional representation of the I/O relationship is (hopefully) achieved in the end.

2.1.1 Multiple Layer Perceptron (MLP) Networks

Among the family of DNNs, we will consider two sub-families: Multiple Layer Perceptron (MLP) Networks and Convolutional Neural Networks (CNNs). MLPs configure as the baseline models among the set of DNNs: indeed they are simple *feed-forward* networks (i.e. networks without any feedback loop) in which several layers, made of a variable number of *neurons*, are composed together, side-by-side, in such a way that all nodes of layer $l-1$ are connected to all nodes of layer l. Figure 2.2 provides a graphical representation of the structure of a MLP network.

Here we introduce some basic notation.

- Each layer is made of $N^{(l)}$ neurons $\left(l \in \{1, \dots, L\}\right)$, being L the total number of layers of the network. Readily, the dimensionality of the input N_{in} must match the number of nodes in the first layer $N^{(0)}$ and the dimensionality of the output N_{out} must match the number of nodes of the last layer $N^{(L)}$.
- $\boldsymbol{x}^{(l)}$ denotes the set of input values of layer l; notice that, because of the structure of this network, $\boldsymbol{x}^{(l)}$ also coincides with the set of output values of layer $l-1$. It belongs to $\mathbb{R}^{N^{(l-1)}}$. Trivially, $\boldsymbol{x}^{(0)}$ coincides with the set of raw input data to be processed.
- $\boldsymbol{y}^{(l)}$ denotes the set of output values of layer l. It belongs to $\mathbb{R}^{N^{(l)}}$. Trivially, $\boldsymbol{y}^{(L)}$ coincides with the final output of the network.
- $w_{ij}^{(l)}$ is a scalar, representing the value of the weight associated to the edge going from neuron i $\left(i \in \{1, \cdots, N^{(l-1)}\}\right)$ of layer $l-1$ to neuron j $\left(j \in \{1, \cdots, N^{(l)}\}\right)$ of layer l. Weights from layer $l-1$ to layer l can be then stored in a matrix $\boldsymbol{W}^{(l)} \in \mathbb{R}^{N^{(l-1)} \times N^{(l)}}$.
- $\boldsymbol{b}^{(l)}$ is a vector, belonging to $\mathbb{R}^{N^{(l)}}$, representing a bias term, i.e. a zero-order term to be summed up to the linear combination of the inputs by the weights inside the argument of the layer activation function, as it will be made clear by equation (2.3)
- $\sigma^{(l)}$ represents the so-called activation function of neurons in layer l. Typically, but not compulsorily, all neurons in the same layer feature the same activation function. Classical choices, that proved to be very successful in terms of approximating power of the network, are the *ReLU* (Rectified Linear Unit) function

$$ReLU(x) = x^{+} = \max(0, x) \tag{2.1}$$

and the sigmoid function

$$sigmoid(x) = \frac{1}{1+e^{-x}} \tag{2.2}$$

In particular the sigmoid function (or its multi-label generalization, called *softmax* function) is used as activation function of the output layer of networks that aim at performing binary (or multiple) classification tasks, being it able to map real inputs into probabilities, i.e. into the interval $[0; 1]$

The working pipeline of MLPs is summarized by the following equation:

$$\boldsymbol{y}^{(l)} = \sigma^l(\boldsymbol{W}^{(l)}\boldsymbol{x}^{(l)} + \boldsymbol{b}^{(l)}) = \sigma^l(\boldsymbol{W}^{(l)}\boldsymbol{y}^{(l-1)} + \boldsymbol{b}^{(l)}) \tag{2.3}$$

Notice that the extension to the case in which multiple input datapoints are processed together is straightforward; in such a situation, indeed, it is enough to define $\boldsymbol{x}^{(l)}$ and $\boldsymbol{y}^{(l)}$ no longer as vectors but as matrices, featuring a number of rows coinciding with the dimensionality N_μ of the input.

During the training phase, the MLP network (and, in general, any NN) tries to learn optimal values for its trainable parameters, i.e. the weights and biases of its layers; those can be embodied into a single vector $\boldsymbol{\Theta} = \{\boldsymbol{\theta}^{(0)}, \dots, \boldsymbol{\theta}^{(L)}\}$, with $\boldsymbol{\theta}^{(l)} =: (\boldsymbol{W}^{(l)}, \boldsymbol{b}^{(l)})$ for simplicity of notation. Such learning process is carried out by minimizing, in the trainable parameters' space, a loss function, whose aim is to give a measure of the discrepancy between the desired output values (available for the training dataset) and the ones predicted in output by the network. A generic expression of the loss function is:

$$\mathcal{L}(\boldsymbol{\Theta}; \boldsymbol{Y}, \boldsymbol{Y}^{(L)}) = \frac{1}{N_{\boldsymbol{\mu}}} \sum_{i=1}^{N_{\boldsymbol{\mu}}} l(\boldsymbol{\Theta}; \boldsymbol{Y}, \boldsymbol{Y}^{(L)}) \tag{2.4}$$

i.e. the loss function is computed as the average loss across the $N_{\boldsymbol{\mu}}$ different training datapoints. Here $\boldsymbol{Y}$ denotes the matrix storing the desired output and $\boldsymbol{Y}^{(L)}$ the matrix storing the output predicted by the NN. A classical choice is to rely on the quadratic loss function defined as:

$$\mathcal{L}(\boldsymbol{\Theta}; \boldsymbol{Y}, \boldsymbol{Y}^{(L)}) = MSE_{\boldsymbol{\Theta}}(\boldsymbol{Y}, \boldsymbol{Y}^{(L)}) = \frac{1}{N_{\boldsymbol{\mu}}} \sum_{i=1}^{N_{\boldsymbol{\mu}}} \sum_{j=1}^{N_{out}} (\boldsymbol{Y}_{ij} - \boldsymbol{Y}_{ij}^{(L)}(\boldsymbol{\Theta}))^2 \tag{2.5}$$

The search for global minima of $\mathcal{L}$, with respect to $\boldsymbol{\Theta}$, is far from being trivial, since it is in general non-convex and it shows multiple local-minima points, in which classical iterative minimization algorithms (Gradient Descent (GD), Stochastic Gradient Descent (SGD), Coordinate Descent (CD)...) can get stuck. However, the problem can be circumvented by performing the optimization task employing optimized versions of the SGD algorithm, among which the Adaptive Moment Estimation (Adam) method (see [14]) and the Root Mean Squared Propagation (RMSProp) method (see [15]) are by far the most widely used. Additionally, the training process of NNs can be very efficiently carried out by exploiting the *backpropagation* algorithm, i.e. a specialized version of the chain rule, discovered in the 60s, popularized by *Rumelhart et al.* in [16] and easily applicable to NNs; for a detailed description of this breakthrough algorithm in the context of NNs and for a more detailed overview on DL models and algorithms, we refer the reader to [11].

2.1.2 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs or ConvNets) are a well-known family of DL architectures, which have been inspired by the natural visual perception mechanism; indeed the predecessor of CNNs can be identified in the *neocognitron* developed by *Fukishima* in [17], which has been clearly influenced by the discovery of receptive fields in animals visual cortex. A detailed and precise description of the history, characteristics, recent developments, current applications and limitations of CNNs can be found in [18]; most of the material present in this section is borrowed from such work. The very first CNN, called *LeNet-5*, has been developed by *LeCun*

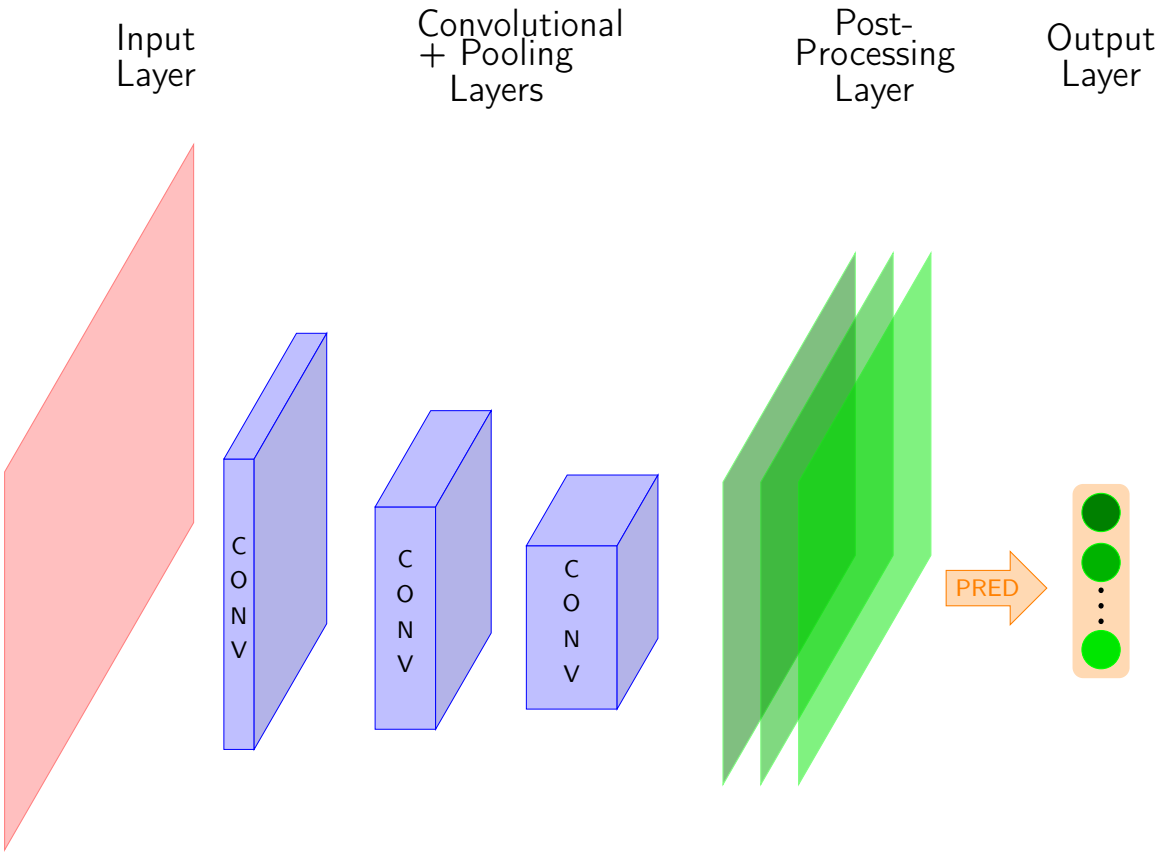


Figure 2.3: Structure of a CNN with no pre-processing layers, 3 convolutional+pooling layers and 1 post-processing layer

et al. in 1990 [19] in the context of handwritten digits recognition. From such point on, and especially after the starting of the new millennium, work on CNN architectures (and, more in general, in the field of DL) has flourished, giving birth to many different and successful models, that have been applied in a broad variety of contexts. Among those, *AlexNet* [20], *VGGNet* [21], *GoogleNet* [22] and *ResNet* [23] surely deserve a special mention, having proved extremely successful and having identified the progressive State-Of-Art in the field.

A great advantage of CNNs with respect to more standard feed-forward MLP-like models lies in the fact that they have been successfully employed in many different areas. Surely the field of image processing and computer vision (thus including image analysis, image classification, object detection, text detection and recognition, action recognition, scene labeling...) is the one where CNNs have been mostly used and have shown their best performances; this is somehow natural if we recall that these models set their roots in biological discoveries linked to animals visual perception. Anyway, CNN models have been also used for speech recognition and NLP (where anyway Recurrent Neural Network (RNN) showed better performances) and for the analysis of time series. This last applicative area is the one of most interest to us, since we would like to extract useful information from ECG signals, that are actually an example of time series. At this aim, it surely deserves a mention the work of *Rajpurkar et al.* [24]; the authors, using a 34-layers CNN featuring a *ResNet*-like architecture, managed to develop an algorithm able to exceed the performances of a board of certified cardiologist in detecting a wide range of heart arrhythmias, just being given ECG signals. Such a result allows us to recognize CNNs as models potentially able to extract lots of useful information from ECG signals, thus being useful in epicardial activation maps reconstruction, as it will be made clear in Chapter 5.

The structure of a generic CNN model can be seen in Figure 2.3. CNN architectures are char-

acterized by the presence of different types of layers, as convolutional layers (from which the name), pooling layers and classical fully-connected layers. *Convolutional layers* take their name from the fact that they perform convolutional operations along specified dimensions of the input, using several convolutional kernels (or channels); each entry of each kernel is learned via the backpropagation algorithm [16]. Via different convolutional kernels, CNN models compute different feature maps; every feature map is obtained by first convolving the input, along specified dimensions, with a learned kernel and lately applying, point-wise, an activation function to the results of the convolution operation. In mathematical terms, supposing a 2D-convolutional layer is considered, we can write the feature value at location (i, j) in the k-th feature map of the l-th layer as:

$$\boldsymbol{y}^{(l)}_{(i,j);k} = \sigma^{(l)}\big(\boldsymbol{W}^{(l)}_k * \boldsymbol{x}^{(l)}_{(i,j)} + \boldsymbol{b}^{(l)}_k\big) \tag{2.6}$$

where $\boldsymbol{W}^{(l)}_k$ and $\boldsymbol{b}^{(l)}_k$ are the weight vector and bias term of the k-th filter of the l-th layer, while $\boldsymbol{x}^{(l)}_{(i,j)}$ is the input patch centered at location (i, j) of the l-th layer. A very important observation is that, in generating a feature map, the kernel is shared by all the locations of the input (indeed $\boldsymbol{W}^{(l)}_k$ does not depend on the location (i, j)); this is a key point, since such weight sharing allows to drastically reduce the number of hyperparameters (easing in turn the training process) and to force the training algorithm to extract globally-relevant features. Each convolutional layer is furthermore characterized by different hyperparameters, as the activation function (which is commonly taken equal across the different channels), the dimensionality of the kernel, the number of channels, the type of padding (i.e how to pad the input if, at certain locations, the kernel does not fit inside it) and the regularization strategy and parameters.

Pooling layers aim at reducing the resolution of the feature maps, by performing operations that map every patch of the input into a single numerical value. The dimensionality of the patches and the amount of stride (i.e. how many entries of the input have to be "skipped" when passing from a patch to the subsequent one) configure as the main hyperparameters of such layers; the bigger the dimensionality of patches and strides, the more significant the overall complexity reduction. Pooling layers are usually placed between one convolutional layer and the subsequent one, so that the feature map of a pooling layer is connected to the one of the preceding convolutional layer. Denoting the pooling function as f_{pool}, the operation done in a pooling layer can be written as:

$$\boldsymbol{y}^{(l_p)}_{(i,j);k} = f_{pool}(\boldsymbol{y}^{(l)}_{(m,n);k}) \quad \forall\ (m, n) \in \mathcal{R}_{(i,j)} \tag{2.7}$$

being $\boldsymbol{y}^{(l_p)}_{(i,j);k}$ the output of the pooling layer placed after the l-th convolutional layer and $\mathcal{R}_{(i,j)}$ a local neighborhood around location (i, j). Typical pooling operations are average pooling (i.e. take the average value of the patch) and max pooling (i.e. take the maximum value of the patch). Notice that pooling layers are deterministic layers.

Finally, convolutional layers and pooling layers can be complemented by *fully-connected layers* (typically *ReLU*-activated), which aim at performing some more high-level reasoning. They can be either placed before or after the convolutional+pooling block; in case they are placed before (*pre-convolutional fully-connected layers*), they behave as pre-processing layers who extract from the input features on top of which the convolutional operations are expected to act better (like, for instance, removing noise that does not carry any useful information). In case, instead, they are placed after the convolutional+pooling block (*post-convolutional fully-connected layers*), they take all neurons of the previous layer and connect them all together, with the goal of generating some global semantic information, that allows to better reconstruct the input-output mapping. Ultimately, in most applications, the output layer of a CNN architecture is yet a fully-connected layer, but whose activation function depends of the type of task to be carried out, as in the case of MLPs.

2.2 Reduced Basis Method

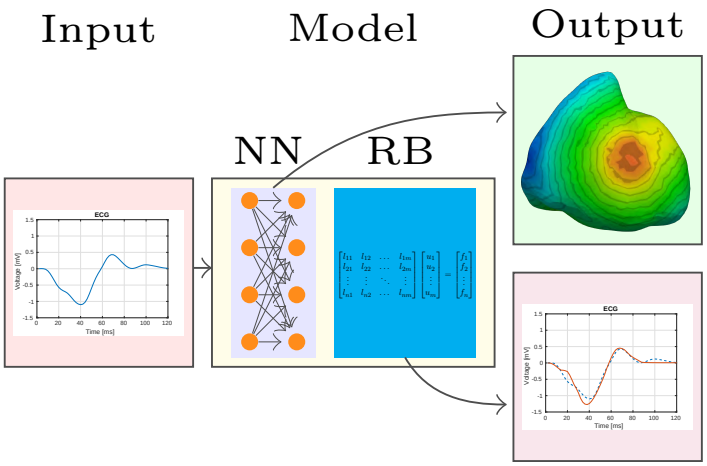


Figure 2.4: Basic structure of the developed PDE-aware DL model, where the RB-solver is highlighted

Reduced Order Modeling (ROM) techniques, like the Reduced Basis (RB) method, find their main applicative field in the context of parametrized PDEs. Thus, in this Section, we first introduce parametrized PDEs (Subsection 2.2.1) and we lately move towards the description of the RB method, both applied to steady (Subsection 2.2.2) and unsteady (Subsection 2.2.3) problems.

2.2.1 Parametrized Partial Differential Equations

Let us consider some parameter space $\mathcal{P} \in \mathbb{R}^p$, $p \geq 1$ and denote as $\boldsymbol{\mu} \in \mathcal{P}$ a parameter vector, whose entries encode some physical and/or geometrical properties of the problem at hand. Then, defining as Ω the computational domain and as $\partial\Omega$ its boundary, we can define a steady parametrized PDE as:

$$\begin{cases} \mathcal{N}[u;\boldsymbol{\mu}] = f(\boldsymbol{\mu}) & in\ \Omega \\ c_1 \dfrac{\partial u}{\partial n} + c_2 u = \bar{u}(\boldsymbol{\mu}) & on\ \partial\Omega \end{cases} \tag{2.8}$$

where $\mathcal{N}[\cdot,\boldsymbol{\mu}]$ denotes the differential operator modeling the problem, $f(\boldsymbol{\mu})$ the forcing term, $\bar{u}(\boldsymbol{\mu})$ the boundary datum and $u = u(\boldsymbol{\mu})$ the parameter-dependent solution that we are looking for.

For each value of $\boldsymbol{\mu}$, the solution of (2.8) can be approximated employing a FOM, like, for instance, the FE method. Such an approach leads to the solution of a system of the form

$$\boldsymbol{N}(\boldsymbol{\mu})\boldsymbol{u} = \boldsymbol{f}(\boldsymbol{\mu}) \tag{2.9}$$

where $\boldsymbol{N}(\boldsymbol{\mu})$ represents a parameter-dependent $N_h \times N_h$ matrix and $\boldsymbol{f}(\boldsymbol{\mu})$ a parameter-dependent N_h-dimensional vector, being N_h the number of DOFs resulting after the FOM approximation and discretization. The information coming from the boundary conditions are somehow embedded in $\boldsymbol{N}(\boldsymbol{\mu})$ and $\boldsymbol{f}(\boldsymbol{\mu})$. Solving such a big system (that, in real applicative fields, can reach dimensions of the order of millions/billions) for several distinct parameter values could be very expensive from the computational point of view. ROM techniques allow then for the computation of a good approximation of the FOM solution for any parameter value, just being given a set of FOM solutions, computed for some carefully pre-selected parameter values and able to show the range of all (or at least of most of) the possible outcomes of the problem at hand.

2.2.2 Reduced Basis Method for Steady Parametrized PDEs

The RB method lies on the assumption that the parameter-dependent and N_h-dimensional FOM solutions of the PDE can be expressed as a linear combination of n_h basis functions, with $n_h \ll N_h$; such functions can be derived from solutions of the PDE itself, computed for some suitably selected parameter values. Here we provide an outline of the method, as divided in two phases: an *offline* phase and an *online* phase.

During the *offline* phase, a number $N_{\boldsymbol{\mu}}$ of FOM solutions is computed, at some selected parameter values $\{\boldsymbol{\mu}_i\}_{i=1}^{N_{\boldsymbol{\mu}}}$; the underlying idea is that such solutions are representative of the vast majority of the possible outcomes of the system. These solutions are then assembled into the so-called *snapshots* matrix $\boldsymbol{S} \in \mathbb{R}^{N_h \times N_{\boldsymbol{\mu}}}$, where they are stored column-wise, so that $\boldsymbol{S}$ can be written as $\boldsymbol{S} = [\boldsymbol{u}(\boldsymbol{\mu_1})|\boldsymbol{u}(\boldsymbol{\mu_2})|\dots|\boldsymbol{u}(\boldsymbol{\mu_{N_{\mu}}})]$. Since the final aim is to express the solution of the PDE, for any parameter value, as a linear combination of basis functions deriving from the *offline*-computed FOM solutions, the next step in the algorithm consists in building up a (lower-dimensional) basis, starting from such solutions. This task can be carried out in different ways; the approach followed in this work is to perform a Singular Value Decomposition (SVD) on the snapshots matrix $\boldsymbol{S}$, which amounts at writing $\boldsymbol{S}$ as:

$$\boldsymbol{S} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{Z}^T \tag{2.10}$$

with $\boldsymbol{U} \in \mathbb{R}^{N_h \times N_h}$, $\boldsymbol{Z} \in \mathbb{R}^{N_{\boldsymbol{\mu}} \times N_{\boldsymbol{\mu}}}$ orthogonal matrices and $\boldsymbol{\Sigma} = diag(\sigma_1, \dots, \sigma_{N_{\boldsymbol{\mu}}}) \in \mathbb{R}^{N_h \times N_{\boldsymbol{\mu}}}$ being the diagonal matrix storing the singular values of $\boldsymbol{S}$ in descending order. The final Reduced Basis, which is also an orthonormal one, is then derived by selecting the first $n_h \ll N_h$ columns of $\boldsymbol{U}$ and it is usually denoted as $\boldsymbol{V} \in \mathbb{R}^{N_h \times n_h}$. The number n_h, more than being selected *a priori*, is obtained in such a way that the approximation error is minimized in Euclidean norm; this is achieved by selecting n_h as the minimum value of $j \in \{1, \dots, N_{\boldsymbol{\mu}}\}$ such that:

$$1 - \frac{\sum_{i=1}^{j} \sigma_i^2}{\sum_{i=1}^{N_{\boldsymbol{\mu}}} \sigma_i^2} \leq \epsilon_{POD}^2 \tag{2.11}$$

being ϵ_{POD} a tolerance value, selected by the user and whose value typically depends of the complexity and the characteristics of the problem at hand. This part is known as the POD construction of the Reduced Basis and it constitutes the final step of the *offline* phase of the RB method. According to [25], which discusses the SVD algorithm implemented in *MATLAB*, the computational complexity of the procedure is $\mathcal{O}(\max(N_h, N_{\boldsymbol{\mu}})^2)$, thus equaling in our case $\mathcal{O}(N_h^2)$, being typically $N_h > N_{\boldsymbol{\mu}}$. The computational burden can be anyway reduced by relying on probabilistic algorithms for the construction of approximate matrix decompositions, as the ones described in [26]; in the case of the SVD, an extremely fast computation of a truncated decomposition (considering only $K < \min(N_h, N_{\boldsymbol{\mu}})$ singular values/vectors) is possible via the Randomized SVD algorithm. Few additional details on the SVD and a brief description of the Randomized SVD algorithm can be found in Appendix B.1. Incidentally, a Reduced Basis can be generated in different ways rather than by leveraging the SVD of the snapshots' tensor; for instance, greedy algorithms can be employed, as discussed in [27].

The *online* phase of the RB method consists in solving the PDE at hand, for any parameter value $\tilde{\boldsymbol{\mu}} \in \mathcal{P}$, exploiting the fact that the solution can be expressed in terms of the pre-computed lower-dimensional basis, encoded in $\boldsymbol{V}$. From the algebraic point of view, this just amounts at solving the linear system of equations:

$$\boldsymbol{N}_{RB}(\tilde{\boldsymbol{\mu}})\boldsymbol{u}_{RB} = \boldsymbol{f}_{RB}(\tilde{\boldsymbol{\mu}}) \tag{2.12}$$

where

$$\boldsymbol{N}_{RB}(\tilde{\boldsymbol{\mu}}) = \boldsymbol{V}^T\boldsymbol{N}(\tilde{\boldsymbol{\mu}})\boldsymbol{V} \qquad \in \mathbb{R}^{n_h \times n_h} \tag{2.13a}$$

$$\boldsymbol{f}_{RB}(\tilde{\boldsymbol{\mu}}) = \boldsymbol{V}^T\boldsymbol{f}(\tilde{\boldsymbol{\mu}}) \qquad \in \mathbb{R}^{n_h} \tag{2.13b}$$

$$\boldsymbol{u}_{RB} = \boldsymbol{V}^T\boldsymbol{u} \qquad \in \mathbb{R}^{n_h} \tag{2.13c}$$

The unknown $\boldsymbol{u}_{RB}$ which is computed by solving (2.12) contains the expansion coefficients of the approximate solution with respect to the Reduced Basis encoded in matrix $\boldsymbol{V}$. To project again such a solution onto the FOM space, a pre-multiplication by $\boldsymbol{V}$ is therefore necessary, so that:

$$\boldsymbol{u} \approx \boldsymbol{V}\boldsymbol{u}_{RB} \tag{2.14}$$

Finally, notice that such a construction is yet not completely independent from the FOM dimension N_h; indeed, while the dimensionality of the linear system to be solved is actually $n_h \ll N_h$, the assembling procedure of such linear system requires, for every new parameter value, the execution of the matricial operations described in (2.13), which are definitely depending on N_h. This bottleneck can be overcome if the problem at hand shows an affine dependence of the FOM arrays on the parameter values, meaning that:

$$\boldsymbol{N}(\boldsymbol{\mu}) = \sum_{q_n=1}^{Q_n} \Theta_n^{q_n}(\boldsymbol{\mu})\boldsymbol{N}^{q_n} \tag{2.15a}$$

$$\boldsymbol{f}(\boldsymbol{\mu}) = \sum_{q_f=1}^{Q_f} \Theta_f^{q_f}(\boldsymbol{\mu})\boldsymbol{f}^{q_f} \tag{2.15b}$$

with $\boldsymbol{N}^{q_n} \in \mathbb{R}^{N_h \times N_h},\ q_n \in \{1, \dots, Q_n\}$ and $\boldsymbol{f}^{q_f} \in \mathbb{R}^{N_h},\ q_f \in \{1, \dots, Q_f\}$. This readily implies that also the RB arrays show an affine dependence on the parameter values, so that:

$$\boldsymbol{N}_{RB}(\boldsymbol{\mu}) = \sum_{q_n=1}^{Q_n} \Theta_n^{q_n}(\boldsymbol{\mu})\boldsymbol{N}_{RB}^{q_n} \tag{2.16a}$$

$$\boldsymbol{f}_{RB}(\boldsymbol{\mu}) = \sum_{q_f=1}^{Q_f} \Theta_f^{q_f}(\boldsymbol{\mu})\boldsymbol{f}_{RB}^{q_f} \tag{2.16b}$$

with $\boldsymbol{N}_{RB}^{q_n} = \boldsymbol{V}^T\boldsymbol{N}^{q_n}\boldsymbol{V} \in \mathbb{R}^{n_h \times n_h},\ q_n \in \{1, \dots, Q_n\}$ and $\boldsymbol{f}_{RB}^{q_f} = \boldsymbol{V}^T\boldsymbol{f}^{q_f} \in \mathbb{R}^{n_h},\ q_f \in \{1, \dots, Q_f\}$. In this way, all the operations with computational complexity depending on N_h can be executed during the *offline* phase; indeed, if the $\boldsymbol{\mu}$-independent RB affine components of $\boldsymbol{N}$ and $\boldsymbol{f}$ are pre-computed (i.e. $\{\boldsymbol{N}_{RB}^{q_n}\}_{q_n=1}^{Q_n},\ \{\boldsymbol{f}_{RB}^{q_f}\}_{q_f=1}^{Q_f}$), during the *online* phase only suitable linear combinations of those, with weights given by $\boldsymbol{\mu}$-dependent scalar coefficients (i.e. $\{\Theta_n^{q_n}(\boldsymbol{\mu})\}_{q_n=1}^{Q_n},\ \{\Theta_f^{q_f}(\boldsymbol{\mu})\}_{q_f=1}^{Q_f}$), have to be performed. Notice that such a procedure can be followed only if the affine parametrization of the FOM arrays holds true for the problem at hand. In case, instead, no affine decomposition takes place on either $\boldsymbol{N}$ or $\boldsymbol{f}$, it is yet possible to derive an approximated one relying on the Empirical Interpolation Method (EIM) (see [28]) or on its discrete version Discrete Empirical Interpolation Method (DEIM) (see [29]) and then basically stick to the same pipeline.

2.2.3 Reduced Order Modeling Techniques for Unsteady Parametrized PDEs

ROM techniques can be also applied to time-dependent parametrized PDEs, which in a general form read as:

$$\begin{cases} \dfrac{\partial u}{\partial t} + \mathcal{N}[u; \boldsymbol{\mu}] = f(t; \boldsymbol{\mu}) & in\ \Omega \times [t_0; T] \\ c_1 \dfrac{\partial u}{\partial n} + c_2 u = \bar{u}(t; \boldsymbol{\mu}) & on\ \partial\Omega \times [t_0; T] \\ u(t = 0) = u_0(\boldsymbol{\mu}) & in\ \Omega \end{cases} \tag{2.17}$$

where $\mathcal{N}[u; \boldsymbol{\mu}]$, $f(t; \boldsymbol{\mu})$ and $\bar{u}(t; \boldsymbol{\mu})$ are defined as in (2.8) (aside of the time dependency), while $u_0(\boldsymbol{\mu})$ encodes the initial condition. Notice that here we assume the differential operator in space $\mathcal{N}[\cdot; \boldsymbol{\mu}]$ to be independent of time. Upon a discretization in space, for instance via the FE

method, (2.17) writes as:

$$\begin{cases} \dfrac{d\boldsymbol{u}}{dt} + \boldsymbol{N}(\boldsymbol{\mu})\boldsymbol{u} = \boldsymbol{f}(t;\boldsymbol{\mu}) \quad in\ [t_0;T] \\ \boldsymbol{u}(t=0) = \boldsymbol{u}_0(\boldsymbol{\mu}) \end{cases} \tag{2.18}$$

The FOM solution can be retrieved performing a partition of the time domain $\left\{[t^{(l)};t^{(l+1)}]\right\}_{l=0}^{N_t-1}$, such that $t^{(0)} = t_0$ and $t^{(N_t)} = T$ and choosing some time marching scheme (as linear multistep methods or Runge-Kutta multistage methods) for the approximation of the time derivative.

The Standard RB Method

The more classical way of proceeding involves the application of model order reduction along the space dimension only, basically replicating what is done in the steady case for all the timesteps $\{t^{(l)}\}_{l=0}^{N_t}$. Before getting there, it is necessary to introduce an additional concept, which is the one of mode-n unfolding of a tensor. Consider to be given tensor $\mathcal{T}$ of order N, which can be visualized as a matrix in N dimensions. Given then a positive number $n \leq N$, we can perform the mode-n unfolding of the tensor $\mathcal{T}$ by reshaping it into a 2D matrix, having a number of rows equal to the cardinality of $\mathcal{T}$ along the n-th dimension and a number of columns equal to the product between all the cardinalities of $\mathcal{T}$ in the dimensions different from n. For instance, suppose to have a tensor $\mathcal{T} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$; its mode-1 unfolding leads to a 2D matrix $\boldsymbol{T}_{(1)} \in \mathbb{R}^{N_1 \times N_2 N_3}$ that can be so written as:

$$\boldsymbol{T}_{(1)} = \left[\begin{array}{cccc|cccc|c|cccc} t_{111} & \cdots & \cdots & t_{1N_21} & t_{112} & \cdots & \cdots & t_{1N_22} & \cdots & t_{11N_3} & \cdots & \cdots & t_{1N_2N_3} \\ t_{211} & \cdots & \cdots & t_{2N_21} & t_{212} & \cdots & \cdots & t_{2N_22} & \cdots & t_{21N_3} & \cdots & \cdots & t_{2N_2N_3} \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \vdots & \cdots & \vdots & \ddots & \ddots & \vdots \\ t_{N_111} & \cdots & \cdots & t_{N_1N_21} & t_{N_112} & \cdots & \cdots & t_{N_1N_22} & \cdots & t_{N_11N_3} & \cdots & \cdots & t_{N_1N_2N_3} \end{array}\right]$$

The mode-2 and mode-3 unfoldings of $\mathcal{T}$ can be constructed in a similar way.

So, suppose to compute the solution to (2.18), using some time-marching scheme, for $N_{\boldsymbol{\mu}}$ different parameter values and to store it in a third-order tensor $\mathcal{S} \in \mathbb{R}^{N_{\boldsymbol{\mu}} \times N_h \times N_t}$. The Reduced Basis along the spatial dimension can be computed performing a POD over the mode-1 unfolding of the snapshots' tensor, which corresponds at reshaping $\mathcal{S}$ such that the N_h spatial degree of freedom (DOF) span over the rows, while both the time DOFs N_t and the number of snapshots $N_{\boldsymbol{\mu}}$ vary over the columns. By doing so, we get:

$$\boldsymbol{S}_{(1)} = \boldsymbol{U}_s \boldsymbol{\Sigma}_s \boldsymbol{Z}_s^T \qquad \in \mathbb{R}^{N_h \times N_t N_{\boldsymbol{\mu}}} \tag{2.19}$$

$$\boldsymbol{\phi}_i = \boldsymbol{u}_s^i, \qquad i \in \mathbb{N}(n_h) \tag{2.20}$$

with $\boldsymbol{u}_s^i$ identifying the i-th column of $\boldsymbol{U}_s$ and with $n_h \ll N_h$ provided that only a minority of relevant singular values is kept at the reduction stage, if the spatial POD tolerance ϵ_{POD}^s is taken sufficiently high in (2.11). This computation requires $n_h N_h$ storage, since the final Reduced Basis in space is represented by a matrix $\boldsymbol{V}_s \in \mathbb{R}^{N_h \times n_h}$.

Once the Reduced Basis in space $\boldsymbol{V}_s$ is computed, the classical RB method can be applied also to the unsteady parametrized PDE (2.18), just by projecting all the quantities onto the dimensionality reduced subspace in space. Thus, (2.18) can be rewritten as:

$$\begin{cases} \dfrac{d\boldsymbol{u}_{RB}}{dt} + \boldsymbol{N}_{RB}(\boldsymbol{\mu})\boldsymbol{u}_{RB} = \boldsymbol{f}_{RB}(t;\boldsymbol{\mu}) \quad in\ [t_0;T] \\ \boldsymbol{u}_{RB}(t=0) = \boldsymbol{u}_{0_{RB}}(\boldsymbol{\mu}) \end{cases} \tag{2.21}$$

where

$$\boldsymbol{N}_{RB}(\boldsymbol{\mu}) = \boldsymbol{V}_s^T \boldsymbol{N}(\boldsymbol{\mu}) \boldsymbol{V}_s \qquad \in \mathbb{R}^{n_h \times n_h} \tag{2.22a}$$

$$\boldsymbol{f}_{RB}(t;\boldsymbol{\mu}) = \boldsymbol{V}_s^T \boldsymbol{f}(t;\boldsymbol{\mu}) \qquad \in \mathbb{R}^{n_h},\ t \in [t_0;T] \tag{2.22b}$$

$$\boldsymbol{u}_{RB} = \boldsymbol{V}_s^T \boldsymbol{u} \qquad \in \mathbb{R}^{n_h} \tag{2.22c}$$

$$\boldsymbol{u}_{0_{RB}}(\boldsymbol{\mu}) = \boldsymbol{V}_s^T \boldsymbol{u}_0(\boldsymbol{\mu}) \qquad \in \mathbb{R}^{n_h} \tag{2.22d}$$

Then, choosing a time marching scheme to approximate the time derivative of $\boldsymbol{u}_{RB}$, it is possible to compute a solution to (2.21) by solving a linear system of dimension $n_h \ll N_h$ at any discrete time instant. Denoting by $C(N)$ the cost of solving a linear system of dimension N (with $N \in \mathbb{N}$), which depends on the type of solver that is used and that monotonically grows with N, the overall computational complexity of the *online* phase of the proposed method is $C_{RB} = C(n_h)N_t$. Such quantity is significantly smaller compared to the corresponding one got using the FOM approximation, which is $C_{FOM} = C(N_h)N_t$. As a side remark, notice that, under the assumptions that $\mathcal{N} \perp\!\!\!\perp t$ and that the time step Δt is constant over time (so, definition of a homogeneous partition of the time interval $[t_0;T]$ and usage non-adaptive time marching schemes), the matrix of the linear system is time-independent too. Thus, it is possible to leverage LU/Cholesky decompositions to speed up the computation of the solution.

Leveraging Dimensionality Reduction in Time

The straightforward application of the classical RB approach to unsteady problems features the evident drawback that no dimensionality reduction is performed across the temporal dimension, thus attaining a complexity that depends on the number of timesteps N_t. Several methods have been proposed to overcome this issue, among which the Space-Time Least-Squares Petrov-Galerkin (ST-LSPG) projection approach proposed by *Choi, Carlberg* in [30] is definitely worth a mention; indeed it allows to compute the solution to the unsteady parametrized PDE at hand at all Space-Time DOFs and for any $\boldsymbol{\mu} \in \mathcal{P}$ just by solving an underdetermined linear system, arising from the minimization of the residual in a weighted l^2-norm. A detailed description of the algorithm goes beyond the aim of the project and it won't be provided in here; anyway it is crucial to understand how the authors managed to reduce the time dimensionality, encoding the information coming from the spatio-temporal solutions to the unsteady PDE at hand in a bunch of coefficients. In particular, three different ways to compute a Reduced Basis along the time dimension have been proposed.

1. **Fixed Temporal Subspaces via T-HOSVD (Temporal High-Order SVD)**: this approach is the most straightforward and it just consists in performing the SVD of the mode-2 unfolding of the snapshot tensor $\mathcal{S}$ (i.e. a SVD on the matrix that has the time DOFs organized over rows and with both spatial DOFs and training parameters spanning over the columns). Hence, we get:

$$\boldsymbol{S}_{(2)} = \boldsymbol{U}_t \boldsymbol{\Sigma}_t \boldsymbol{Z}_t^T \qquad \in \mathbb{R}^{N_t \times N_h N_{\boldsymbol{\mu}}} \tag{2.23}$$

$$\boldsymbol{\psi}_j = \boldsymbol{u}_t^j \qquad j \in \mathbb{N}(n_t) \tag{2.24}$$

   with $\boldsymbol{u}_t^j$ identifying the j-th column of matrix $\boldsymbol{U}_t$ and with $n_t \ll N_t$ provided that only a minority of relevant singular values is kept at reduction stage, if the temporal POD tolerance is taken sufficiently high in (2.11). In simpler terms, this procedure derives a basis for the time dynamics of the problem by observing the evolution of the FOM solution at all the spatial DOFs. This approach requires $n_t N_t$ storage, since the Reduced Basis in time is represented by the matrix $\boldsymbol{V}_t \in \mathbb{R}^{N_t \times n_t}$.

2. **Fixed Temporal Subspaces via ST-HOSVD (Spatio-Temporal High-Order SVD)**: this approach is maybe more complex than the previous one, but definitely less expensive from the computational point of view, since it leverages the previously computed spatial

Reduced Basis $\boldsymbol{V}_s$. In particular, the temporal Reduced Basis is computed by performing a SVD on the mode-2 unfolding of the projection of the snapshots' tensor $\mathcal{S}$ onto the reduced spatial subspace, with respect to its first component. Formally, this means that we have to compute the following "reduced" tensor:

$$\mathcal{S}_{\boldsymbol{V}_s} =: \mathcal{S} \times_1 \boldsymbol{V}_s \in \mathbb{R}^{n_h \times N_t \times N_{\boldsymbol{\mu}}} \tag{2.25}$$

$$(\mathcal{S} \times_1 \boldsymbol{V}_s)_{ijk} = \sum_{l=1}^{N_h} \mathcal{S}_{ljk} \boldsymbol{\phi}_{il} \tag{2.26}$$

being $\boldsymbol{\phi}_{il}$ the l-th component of the i-th spatial basis vector, thus equaling $\boldsymbol{V}_{s_{li}}$. Basically, this computation can be seen as the unsteady counterpart of the multiplication by the transpose of the basis matrix $\boldsymbol{V}$, which is done when performing the projection onto the RB space in the steady case. Given this new tensor $\mathcal{S}_{\boldsymbol{V}_s}$, performing the SVD on its mode-2 unfolding, we get that:

$$\boldsymbol{S}_{\boldsymbol{V}_s(2)} = \boldsymbol{U}_{t,\boldsymbol{V}_s} \boldsymbol{\Sigma}_{t,\boldsymbol{V}_s} \boldsymbol{Z}^T_{t,\boldsymbol{V}_s} \qquad \in \mathbb{R}^{N_t \times n_h N_{\boldsymbol{\mu}}} \tag{2.27}$$

$$\psi_j = \boldsymbol{u}^j_{t,\boldsymbol{V}_s} \qquad j \in \mathbb{N}(n_t) \tag{2.28}$$

being $\boldsymbol{u}^j_{t,\boldsymbol{V}_s}$ the j-th column of the matrix $\boldsymbol{U}_{t,\boldsymbol{V}_s}$ and $n_t \ll N_t$ provided that only a minority of relevant singular values is kept at reduction stage, if the temporal POD tolerance is taken sufficiently high in (2.11). In simpler terms, the current approach computes a reduced temporal basis by observing the dynamics over time of all the coefficients resulting from the projection of the solutions onto the dimensionality reduced subspace in space; heuristically, those are expected to be less subject to noise than the plain values of the solution at all the spatial DOFs, thus suggesting this method to be able to provide a better encoding of the information relative to time-evolution. The storage requirement is again $n_t N_t$, but a significant saving of computational resources is achieved in the POD, by means of the projection step, since $n_h \ll N_h$; the temporal basis is stored in the matrix $\boldsymbol{V}_t \in \mathbb{R}^{N_t \times n_t}$.

3. **Tailored Temporal Subspaces via ST-HOSVD (Spatio-Temporal High-Order SVD)**: this last approach is a further generalization of the previous one and it allows to compute a different temporal basis, depending on the corresponding spatial basis vector $\boldsymbol{\phi}_i$. In other words, each element of the spatial Reduced Basis is equipped with its own "customed" temporal basis. Such a method performs again the SVD on the space-projected snapshots' tensor $\mathcal{S}$, but it does so independently for each element of the Reduced Basis in space. Thus, for each $i \in \{1, \dots, n_h\}$, first the projection of the snapshots' tensor $\mathcal{S}$ onto the space spanned by $\boldsymbol{\phi}_i \equiv \boldsymbol{V}_{s_i}$ (i-th column of $\boldsymbol{V}_s$) is computed, i.e.

$$\mathcal{S}_{\boldsymbol{V}_{s_i}} =: \mathcal{S} \times_1 \boldsymbol{V}_{s_i} \in \mathbb{R}^{N_t \times N_{\boldsymbol{\mu}}} \tag{2.29}$$

$$(\mathcal{S} \times_1 \boldsymbol{V}_{s_i})_{jk} = \sum_{l=1}^{N_h} \mathcal{S}_{ljk} \boldsymbol{V}_{s_{il}} \tag{2.30}$$

Then its SVD is derived, so that:

$$\boldsymbol{S}_{\boldsymbol{V}_{s_i}} = \boldsymbol{U}_{t,\boldsymbol{V}_{s_i}} \boldsymbol{\Sigma}_{t,\boldsymbol{V}_{s_i}} \boldsymbol{Z}^T_{t,\boldsymbol{V}_{s_i}} \qquad \in \mathbb{R}^{N_t \times N_{\boldsymbol{\mu}}} \tag{2.31}$$

$$\boldsymbol{\psi}^i_j = \boldsymbol{u}^j_{t,\boldsymbol{V}_{s_i}} \qquad i \in \mathbb{N}(n_h),\ j \in \mathbb{N}(n^i_t) \tag{2.32}$$

being $\boldsymbol{u}^j_{t,\boldsymbol{V}_{s_i}}$ the j-th column of the matrix $\boldsymbol{U}_{t,\boldsymbol{V}_{s_i}}$ and $n^i_t \ll N_t$ the cardinality of the temporal Reduced Basis corresponding to the spatial Reduced Basis element $\boldsymbol{\phi}_i \equiv \boldsymbol{V}_{s_i}$. Notice that, in this case, the storage required is $\sum_{i=i}^{n_h} n^i_t N_t =: n_{st} N_t$, which is higher with respect to the previous cases (n_{st} is expected to be quite much higher than n_t) since a different temporal Reduced Basis is computed for each element of the Reduced Basis in space. Anyway, precisely because the temporal bases are tailored to each element of the spatial one, it is reasonable to expect them to be lower-dimensional with respect to the ones of the

previous approaches. This entails that the information coming from each spatio-temporal solution to the PDE at hand can be encoded in a lower number of coefficients, i.e. in $\sum_{i=i}^{n_h} n_t^i =: n_{st} < n_h n_t$ coefficients. Incidentally, the fact that the temporal reduced bases are obtained from the coefficients resulting from the dimensionality reduction in space, and not from the plain values of the solution at the spatial DOFs, allows this method to be relatively robust to noise, as highlighted also for the second approach. The temporal bases are ultimately stored in the matrices $\{\boldsymbol{V}_t^i\}_{i=1}^{n_h}$ such that $\boldsymbol{V}_t^i \in \mathbb{R}^{N_t \times n_t^i}, \quad i \in \{1, \dots, n_h\}$.

As it will be made clear in Chapters 4 and 5, in the context of this project we have not exploited the spatio-temporal dimensionality reduction in solving a time-dependent parametrized PDE (as done by the ST-LSPG projection method for instance). Conversely, we faced the need of encoding the information coming from spatio-temporal FOM solutions into the lowest number of coefficients, yet retaining a good degree of accuracy. Because of this, as better reported in Subsection 4.2.4, on the one side we employed the standard RB method to solve a steady generalized Laplace equation at different discrete time instants and on the other side we encoded the information coming from spatio-temporal solutions to the heart EP problem using the *Tailored Temporal Subspaces via ST-HOSVD* approach. In the last part of the current section, we thus consider such approach to properly define the spatio-temporal reduced subspace, its basis functions and the operations needed to perform the projection of a FOM solution onto such space and the re-projection of a reduced solution back onto the FOM space.

The space spanned by the $N_{\boldsymbol{\mu}}$ FOM snapshots defining the training dataset can be expressed as:

$$\mathcal{ST} = \boldsymbol{u}^0(\boldsymbol{\mu}) \otimes \mathcal{O} + span\{\boldsymbol{u}(\cdot, \boldsymbol{\mu}_i) - \boldsymbol{u}^0(\boldsymbol{\mu}_i)\}_{i=1}^{N_{\boldsymbol{\mu}}}; \tag{2.33}$$

where $\mathcal{O} : \{t^{(l)}\}_{l=0}^{N_t} \rightarrow \mathbf{1}_{N_t}$ is a functional that maps the set of all time instants into a N_t-dimensional vector of ones. Basically, the operation $\boldsymbol{u}^0(\boldsymbol{\mu}) \otimes \mathcal{O}$ has the effect of giving as output a $N_h \times N_t$ matrix, where the initial condition is replicated over each column (i.e. over each time instant). Notice that the initial condition is subtracted from the solution at each time instant in the "spanned space" and summed up with a separate term; this can be very useful when lately performing PODs to find the dimensionality reduced subspaces. Indeed, resorting to homogeneous initial conditions may be of help in finding a better reduced subspace, since the phenomenon of inertia that the initial condition exerts on the solution at the first time instants is "hidden".

The idea is to write the spatio-temporal reduced subspace as an outer product between a reduced subspace in space and reduced subspace in time, plus a term encoding the initial condition contribution. Thus, we aim at getting something of the form:

$$\mathcal{ST} \approx \mathcal{ST}_{red} = \boldsymbol{u}^0(\boldsymbol{\mu}) \otimes \mathcal{O} + \mathcal{S} \otimes \mathcal{T} \tag{2.34}$$

being $\mathcal{S} =: span\{\boldsymbol{\phi}_i\} \subset \mathbb{R}^{N_h}$ the reduced subspace in space and $\mathcal{T} =: span\{\boldsymbol{\psi}_j\}_{j=1}^{n_t} \subset \mathbb{R}^{N_t}$ the reduced subspace in time. Following the last of the proposed alternatives, we end up defining a different temporal reduced subspace for each basis function $\boldsymbol{\phi}_i$ of the reduced subspace in space $\mathcal{S}$. Thus we can express the spatio-temporal reduced subspace as:

$$\mathcal{ST} \approx \mathcal{ST}_{red} = \boldsymbol{u}^0(\boldsymbol{\mu}) \otimes \mathcal{O} + \oplus_{i=1}^{n_h} span(\boldsymbol{\phi}_i) \otimes \mathcal{T}_i; \tag{2.35}$$

with $\mathcal{T}_i = span\{\boldsymbol{\psi}_j^i\}_{j=1}^{n_t^i} \subset \mathbb{R}^{N_t}$ and n_t^i denoting the number of temporal basis functions corresponding to the spatial basis function $\boldsymbol{\phi}_i$. Given this formulation of the space-time reduced subspace, the generic space-time basis function can be written as:

$$\boldsymbol{\pi}_{\mathcal{F}_{(i,j)}} = \boldsymbol{\phi}_i \otimes \boldsymbol{\psi}_j^i \in \mathbb{R}^{N_h \times N_t} \qquad i \in \mathbb{N}(n_h), \quad j \in \mathbb{N}(n_t^i) \tag{2.36}$$

where $\mathcal{F} : (i, j) \rightarrow \sum_{k=1}^{i-1} n_t^k + j$ is a mapping from the spatial-basis and temporal-basis indexes to the spatio-temporal-basis indexes. Finally, the generic parametrized solution to the unsteady

PDE at hand $\boldsymbol{u}(\boldsymbol{\mu})$ can be approximated as:

$$\boldsymbol{u}(\boldsymbol{\mu}) \approx \boldsymbol{u}^0(\boldsymbol{\mu}) \otimes \mathbf{1}_{N_t} + \sum_{i=i}^{n_{st}} \boldsymbol{\pi}_i \hat{u}_i \tag{2.37}$$

where n_{st} defines the total number of spatio-temporal basis functions and $\{\hat{u}_i\}_{i=1}^{n_{st}}$ are the so-called generalized coordinates, which represent the expansion coefficients of $\boldsymbol{u}(\boldsymbol{\mu})$ with respect to the spatio-temporal basis functions. So, considering that the basis is orthonormal, we can write the k-th generalized coordinate of $\boldsymbol{u}(\boldsymbol{\mu})$ as:

$$\hat{u}^k = \sum_{i=1}^{N_h} \sum_{j=1}^{N_t} \boldsymbol{\pi}_{kij} \boldsymbol{u}(\boldsymbol{\mu})_{ij} \qquad k \in \mathbb{N}(n_{st}) \tag{2.38}$$

where $\boldsymbol{\pi}_{kij}$ denotes the coordinates i (in space) and j (in time) of the k-th space-time basis element $\boldsymbol{\pi}_k$. Equation (2.38) expresses then how a FOM solution can be projected onto the spatio-temporal dimensionality reduced subspace, while equation (2.37) allows to re-project onto the FOM space a spatio-temporal reduced solution, expressed via its generalized coordinates $\{\hat{u}_k\}_{k=0}^{n_{st}}$.

Chapter 3

PDE-aware Neural Networks and RB-DNN models

As anticipated in Chapter 1, a key element of the present work is the idea of merging together in a unique model data-driven DL techniques and more classical physics-aware numerical methods. Such a choice is motivated by the recent growth of both fields, which ended up intersecting in a common area that we ultimately wish to explore. On the one side, indeed, in recent times, with the abundance of available data, ML and data analytics techniques have drawn significant attention and have yielded several breakthrough results in the more disparate fields. On the other side, classical numerical methods for the modeling of physical phenomena via PDEs/ODEs have undergone a significant stage of development too; nowadays the dynamics of many physical systems can be simulated with high degree of accuracy and precision and at reasonable computational cost thanks to such recent advances. Thus it seems reasonable to try to join these two fast-growing components, trying to take from both the best parts, in a sort of mutual relationship by means of which the weaknesses of one approach are mitigated by the strengths of the other. Such a strategy provided to give an appreciable pay-off in the so-called *small data regime*, i.e. in contexts characterized by data scarcity, where the application of baseline DL techniques features an increased complexity. Indeed, if underlying physical laws explaining the behaviour of some of the variables involved in the system are known, then the performances of data-driven approaches can be significantly improved by taking advantage of those.

This chapter will be divided in two Sections; in Section 3.1 a more detailed overview on the field of PDE-aware ML models will be given, posing particular attention to Physically Informed Neural Networks (PINNs) (see [1]) and to RB-DNNs (see [2]). In Section 3.2, instead, a more exhaustive description of RB-DNN models will be given, discussing their structure and goals (see Subsection 3.2.1) and showing the results of their application to both steady and unsteady parametrized PDE problems (see Subsection 3.2.2). This choice is motivated by the fact that the architectures of the PDE-aware DL models developed within this project have been inspired by the ones of the RB-DNNs.

3.1 PDE-aware Neural Networks: a Brief Overview

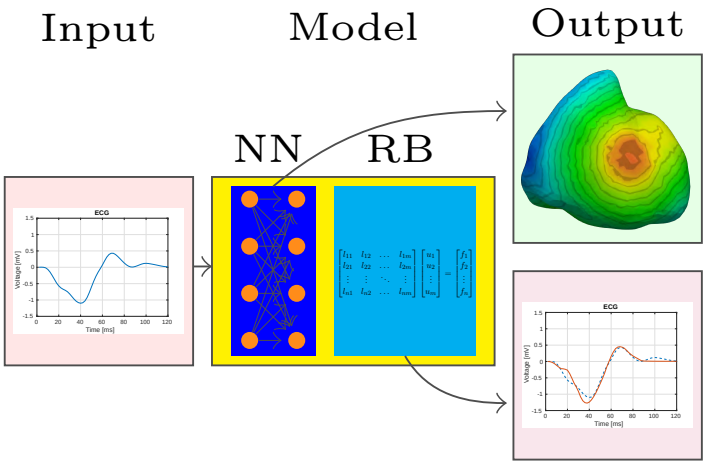


Figure 3.1: Basic structure of the developed PDE-aware DL model, where the model highlighted

Recent years have been unarguably characterized by the explosive growth of available data and of computing resources; this has induced major advances in the context of machine learning and data analytics, which have yielded several breakthrough results in the most disparate fields, from image recognition [3] to text categorization [4] and natural language processing (NLP) [5]. In particular, DL-based algorithms have experienced a huge success in the last decade, being able to provide (approximate) data-driven solutions to problems on top of which "classical" algorithmic paradigms used to experience lots of difficulties and showed many lacks and weaknesses.

Among the fields where DL-based models have opened the way towards new horizons and perspectives, anyway, there's no place for the one related to the numerical approximation of PDEs. Surely a reason for that relies in the fact that classical numerical methods for the modeling of physical phenomena via PDEs have experienced a massive development too in recent times, reaching stunning degrees of accuracy and precision in a variety of heterogeneous contexts. Another motivation may be surely found in the reluctance of the numerical analysis community to adopt algorithms that suffer of a lack of theoretical foundations, as the DL-based ones. Indeed, while classical Finite Elements and Finite Volumes methods are equipped with theorems and corollaries delivering precise errors bounds and stability constraints, the properties of NNs have only been partially discovered, especially when they come to the applicative stage. It is in fact true that the characterization of NNs as universal functions approximators has been well-known for many years (see [31]), but it is still obscure and often demanded to heuristics and personal intuition how to design optimal NN architectures, choosing the hyperparameters so that an ideal trade-off between model complexity and efficiency is achieved. Recent advances in defining a (not yet rock) solid mathematical theory of NNs can be for instance found in [32,33].

This mutually exclusive trend has started being reverted only in the very recent years, where several studies bringing proposals for the integration of DL-based models into the well-established and precise world of numerical analysis have appeared. The general key point of such proceeding relies in the fact that it may be possible to establish a win-win mutual relationship between the two fields. On the one side, indeed, DL could benefit of numerical analysis because it would be enriched with a solid theoretical foundation that would make it less inclined towards reasonings based on heuristics and experience. On the other side, instead, classical numerical analysis could take advantage of DL models to accomplish all those tasks on top of which it has always shown its limitations and that, instead, proved to be carried out efficiently by leveraging data abundance. Clearly, the merging process could be boosted if more rigorous theoretical results on the integration between DL models and classical numerical methods were available; at this aim it surely deserves a mention [6] by *Kutyniok et al.*, where the authors analyze, both theoretically

and numerically, the properties of *ReLU*-activated DNNs in approximating the solution maps of parametrized PDEs, also leveraging ROM techniques.

According to [1], an ideal situation in which the aforementioned integration can be performed is the so-called *small-data* regime i.e. situations in which the amount of data at disposal is either limited or partial or subject to a high degree of inaccuracy, because of the high cost/complexity of data acquisition procedures. In such a context, all classical DL-based methods (DNNs, CNNs, RNNs, etc...) traditionally feature severe problems in terms of robustness, generalization and convergence. If the phenomenon generating the data is anyway characterized by the presence of some underlying physical laws, expressible by means of PDEs/ODEs, then classical numerical methods can come to the rescue of the newborn DL-based ones. Indeed the knowledge of the physics of the problem at hand can be, somehow, made available to the NN model, acting as a physically-aware regularization agent, that (hopefully) eases the NN design and training, improving in turn the overall model performances.

Among all the works on PDE-aware NNs, surely [1] by *M. Raissi, P. Perdikaris, G.E. Karniadakis* deserves a special mention, representing somehow the State-Of-Art in the field. In such work, the authors, leveraging the property of NN as universal functions approximators, train simple $Tanh$-activated MLP networks in such a way that they provide data-driven solutions to PDEs, estimating also some of the underlying characteristic parameters. Such networks take the name of Physically Informed Neural Networks (PINNs) and they have been presented in the form of two different models. In the continuous-in-time model, the network takes as input the coordinates of a point $(\boldsymbol{x}, t)$ in the computational domain and it produces as output the value of the solution to the PDE at that point; the awareness of the physics of the problem is achieved by minimizing, in the loss function, the Mean Squared Error on the residuals of the PDE at hand, computed via the technique of automatic differentiation (see [34]), together with the coupled initial and boundary conditions. Conversely, the discrete-in-time model exploits Runge-Kutta time marching schemes with very high number of stages (of the order of 100) in order to provide the solution to the PDE at a certain point $\boldsymbol{x}$ and at several discrete time instants $\{t_j\}_{j=0}^{s}$ with $t_j = t_0 + j\Delta t$, just being given $(\boldsymbol{x}, t_0)$ as input; in this case the loss is constructed as the Sum Squared Error over the values of the solution at $\{(x, t_j)\}_{j=0}^{s}$. PINNs' performances proved to reach incredible levels of accuracy for both the continuous-in-time and the discrete-in-time model on a broad variety of problems, from the simple $1D$ Burgers' equation to the much more complicated cases of the Navier-Stokes equations or of the complex-valued Schrodinger equation.

As explained in Chapter 1, the aim of this project is to develop physically-aware DL models that could provide approximate solutions to the inverse problem of electrocardiography (see Section 4.3). Thus, we would like our model to receive as input "external" measurements (like ECG signals for instance) and to produce an output descriptive of the time evolution of the epicardial extracellular potential. In view of this, PINNs seem not too adequate to the task: indeed it could be possible to design a PINN-inspired model that receives as input both the coordinates of a point $(\boldsymbol{x}, t)$ and the body surface potentials (eventually just in a time window including the target time instant t), and that gives as output the value of the epicardial extracellular potential at $(\boldsymbol{x}, t)$. This may configure as a possible extension of the present work, but at first we preferred to accomplish the task at hand by leveraging a different type of PDE-aware architectures, which combine DL with ROM techniques. Several works have been already made in this sense (as [7–9] for instance), but the one we took inspiration from is *Data Driven Approximation of Parametrized PDEs by Reduced Basis and Neural Networks* by *N. Dal Santo, S. Deparis, L. Pegolotti*, where DL-based models, embedding a Reduced Basis solver as a deterministic layer within the architecture, are presented. Such models take the name of Reduced Basis Deep Neural Networks (RB-DNNs) and, having exerted a significant influence on our work, we have devoted the entire subsequent Section to their description and to a brief discussion on their performances.

3.2 RB-DNN models

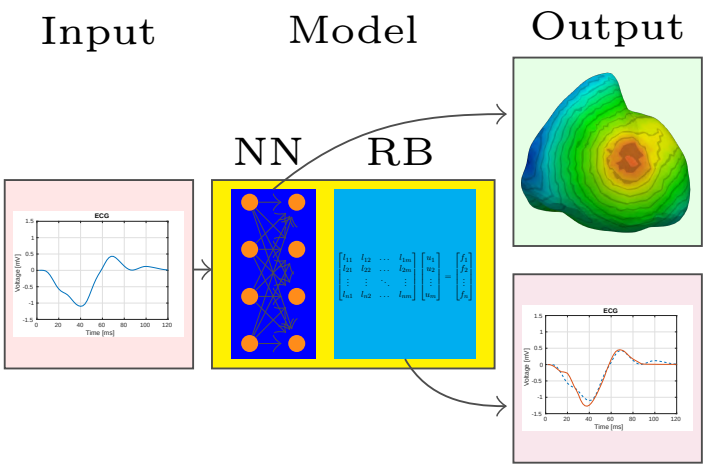


Figure 3.2: Basic structure of the developed PDE-aware DL model, where the model highlighted

3.2.1 Description, Structure and Goals of RB-DNN models

The aim of the current Subsection is to provide a more detailed description of RB-DNN models, focusing on their structure and architecture and highlighting their main points of strength and goals. The discussion provided here is in large part borrowed from [2] and it just takes into account the case of steady parametrized PDEs (see Subsection 2.2.1), admitting an affine decomposition of the FOM/ROM arrays with respect to the parameter values (see (2.15) - (2.16)). The transition towards the case of unsteady problems, will be faced in Subsection 3.2.2; the hypothesis of affine parametrization, as said in Section 2.2, can be relaxed by taking advantage of EIM (see [28]) and DEIM (see [29]) algorithms.

Let us suppose we are given:

- A steady parametrized PDE, as the one in (2.8); let us call $\boldsymbol{\mu} \in \mathcal{P} \subset \mathbb{R}^p$, $p \geq 1$ the vector whose entries encode the characteristic parameters and let us suppose that the measurement of such quantities is either not available or very inaccurate or very complex/expensive/invasive.
- A set of points $\mathcal{P}_{in} = \{p_i\}_{i=1}^{N_{in}}$ at which we know the value of the solution to the PDE
- A set of points $\mathcal{Q}_{out} = \{q_j\}_{j=1}^{N_{out}}$ at which we desire to estimate the value of the solution

The final aim is, given the values of the solution to the PDE at the input points $\mathcal{P}_{in}$ for $N_{\boldsymbol{\mu}}$ unknown parameter values, to compute the values of the solution at the desired output points $\mathcal{Q}_{out}$ for the corresponding parameter values as well as the values of the parameters themselves. The choices of the sets of locations $\mathcal{P}_{in}$ and $\mathcal{Q}_{out}$ can be different; for instance, it is possible to set $\mathcal{P}_{in} \equiv \mathcal{Q}_{out}$, thus designing a PDE-aware autoencoder able to reconstruct the values of the underlying parameters. Conversely, we may also choose $\mathcal{P}_{in}$ as made of points placed in regions of the computational domain where measurements are expected to be less complex and/or more precise, while $\mathcal{Q}_{out}$ defines regions where measurements of the solution values are typically difficult or very inaccurate. In this setting, the RB-DNN architecture is expected to perform a sort of extrapolating task, reconstructing the values of the solution in regions of the domain where no data was originally available.

The idea presented in [2] is to carry out this task by means of a combination of ROM numerical methods (in particular, using the RB method introduced in Section 2.2) and DL techniques; this finally results in the assembling of the RB-DNN architecture. As anticipated in the previous Subsection, the element of novelty of this approach with respect to already proposed ones, in

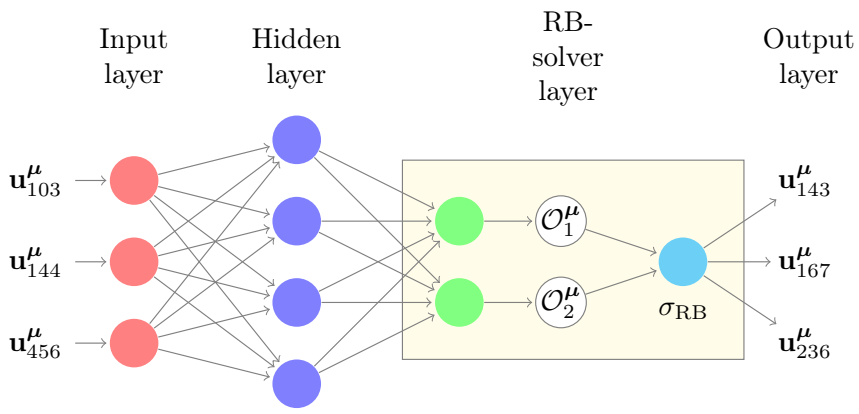


Figure 3.3: Structure of a simple RB-DNN architecture, with $N_{in} = 3$, $N_{out} = 3$ and 2 characteristic parameters $\{\mathcal{O}^{\boldsymbol{\mu}}_i\}^2_{i=1}$ "internally" estimated by the trainable MLP part of the model. The numbers reported as subscript of the input and output entries represent the indices associated to the DOFs chosen to assemble the input and output datapoints. Image taken from [2]

the context of data-driven numerical approximation of PDEs (like PINNs, for instance), is to exploit the underlying physics of the problem at hand not in the expression of the loss function, but in a self-structured deterministic layer, which uses the RB method as its exotic non-linear activation function.

The structure of a simple RB-DNN can be found in Figure 3.3; it is immediate to visualize it as made of two distinct parts, placed one after the other. In the first part, we find a MLP network made of $L - 1$ layers, each of whom can be constituted by a different number of neurons; in [2] they are all said to exhibit a *ReLU* as activation function, but different choices can be made in this sense, according to the problem at hand. The MLP part of the NN takes as input the values of the solution at the input locations $\mathcal{P}_{in}$ and it is responsible for providing, as output, the values of the characteristic parameters of the PDE. In other words, the presence of a mapping between the values of the solution measured at the input locations $\mathcal{P}_{in}$ and the values of the characteristic parameters is postulated, i.e.

$$\begin{aligned} \mathcal{M} : \quad \mathcal{S}_{in} \quad &\to \mathcal{P} \\ \{\tilde{\boldsymbol{u}}(p_i;\boldsymbol{\mu})\}_{p_i \in \mathcal{P}_{in}} \quad &\mapsto \boldsymbol{\mu} \end{aligned} \tag{3.1}$$

with $\mathcal{S}_{in}$ identifying the space spanned by the values of the solution measured at the input locations $\{\tilde{\boldsymbol{u}}(p_i;\boldsymbol{\mu})\}_{p_i \in \mathcal{P}_{in}}$. The aim of the trainable part of the model is then to estimate such mapping from the data, leveraging the well-established capability of NNs as universal function approximators (see [31]).

The following remarks are in order. First the MLP part of the RB-DNN could produce estimates of either the parameter values $\boldsymbol{\mu}$ or of the parameter-dependent coefficients of the affine parametrization of the FOM/RB arrays, i.e. $\{\Theta^{q_n}_n(\boldsymbol{\mu})\}^{Q_n}_{q_n=1}$ and $\{\Theta^{q_f}_f(\boldsymbol{\mu})\}^{Q_f}_{q_f=1}$, eventually approximated via EIM-DEIM methods. Second, the architecture of the trainable part of the RB-DNN does not have to be compulsorily a simple MLP, just made of several flanked fully-connected layers. As it will be discussed in the next Subsection, for instance, in case unsteady problems are faced and input data are organized as time series, placing either a CNN or a RNN in the trainable part may be of great help, since it can allow to exploit the temporal dynamics characterizing the input data and thus to extract features that are more effective for the ultimate prediction task.

Following the notation adopted in Section 2, we can express the operations done by the MLP among its fully-connected layers as:

$$\boldsymbol{y}^{(l)} = \sigma^{(l)}(\boldsymbol{W}^{(l)}\boldsymbol{x}^{(l)} + \boldsymbol{b}^{(l)}) \tag{3.2}$$

In the end, the model returns either p or $Q_f + Q_n$ final outputs in layer $L-1$, depending on the nature of the quantities that it is expected to estimate; whichever the case, following the notation of [2], let us denote such quantities as $\mathcal{O}_i^{\boldsymbol{\mu}}$.

In the second part of the network we find the real element of novelty: a RB solver, embedded inside the NN architecture as a non-trainable layer (also called *lambda-layer*). In a perfectly deterministic way, then, this last layer takes as input the values given as output by the MLP (so either the parameters or the parameter-dependent coefficients of the affine parametrization of the FOM/RB arrays) and it efficiently computes the values of the solution at the output locations $\mathcal{Q}_{out}$. We can synthesize the action of the RB-layer as:

$$\{\tilde{\boldsymbol{u}}(q_i;\tilde{\boldsymbol{\xi}}(\tilde{\boldsymbol{\mu}}))\}_{q_i\in\mathcal{Q}_{out}} = \boldsymbol{y}^{(L)} = \sigma_{\text{RB}}(\boldsymbol{y}^{(L-1)}) = \sigma_{\text{RB}}(\boldsymbol{W}^{(L-1)}\boldsymbol{x}^{(L-1)} + \boldsymbol{b}^{(L-1)}) \tag{3.3}$$

being $\tilde{\boldsymbol{\xi}}(\tilde{\boldsymbol{\mu}}) = \tilde{\boldsymbol{\mu}}$ or $\tilde{\boldsymbol{\xi}}(\tilde{\boldsymbol{\mu}}) = [\Theta_n^1(\tilde{\boldsymbol{\mu}}),\dots,\Theta_n^{q_n}(\tilde{\boldsymbol{\mu}}),\Theta_f^1(\tilde{\boldsymbol{\mu}}),\dots,\Theta_f^{q_f}(\tilde{\boldsymbol{\mu}})]$ the vector containing either the inputs to the deterministic RB-solver layer and $\sigma_{\text{RB}} : \mathbb{R}^s \to \mathbb{R}^{N_{out}}$, with $s = p$ or $s = Q_n + Q_f$, the functional representation of the RB method. Actually, the last layer does not just solve a dimensionality reduced problem, since this alone would not be enough to retrieve the values of the solution at the output locations; indeed, once the approximated reduced solution $\tilde{\boldsymbol{u}}_{RB}$ is available, it is also re-projected onto the FOM space and evaluated/interpolated at the desired output locations. As a whole, then, the action of the output layer can be written as:

$$\sigma_{\text{RB}}(\tilde{\boldsymbol{\xi}}) = \boldsymbol{R}_{out}^T\boldsymbol{V}\tilde{\boldsymbol{u}}_{RB}(\tilde{\boldsymbol{\xi}}) = \boldsymbol{R}_{out}^T\boldsymbol{V}\boldsymbol{N}_{RB}^{-1}(\tilde{\boldsymbol{\xi}})\boldsymbol{f}_{RB}(\tilde{\boldsymbol{\xi}}) \tag{3.4}$$

being $\boldsymbol{R}_{out}$ a matrix belonging to $\mathbb{R}^{N_h\times N_{out}}$ which allows to evaluate the solution at the target output locations in $\mathcal{Q}_{out}$.

It is crucial to underline that the choice of resorting to a ROM technique to solve the PDE at hand is almost compulsory, if the solver layer is embedded inside the network architecture and thus involved in the overall training process via backpropagation algorithm. Indeed, the computation of the solution to the PDE, given the values of the characteristic parameters, has to be performed $N_{\boldsymbol{\mu}}N_{epochs}$ times, being $N_{\boldsymbol{\mu}}$ the number of training datapoints, generated for different parameter values, and N_{epochs} the number of training epochs. In standard applicative contexts this number can be very high, thus leading to an exceed of both memory and wall time constraints, at least on "standard" machines, if the FOM problem has to be solved. Reducing the dimensionality of the problem at hand via ROM techniques, instead, makes it feasible to embody the computation of the parametrized PDE approximated solution within the NN model. Readily, the choice of the RB method, among the set of ROM techniques, is not mandatory, but it seems to be the most straightforward, at least in a general framework. Also, the hypothesis of affine parametrization (or of its approximation via EIM/DEIM algorithms) allows to further minimize the amount of computations needed to assemble the RB-projected linear system to be solved, thus ultimately leading to a significant speed up in the training/testing wall times.

Finally, the loss functional to be minimized is defined simply as the Mean Squared Error (MSE) on the values of the solution at the output locations $\mathcal{Q}_{out}$, i.e.

$$\mathcal{L}(\Theta;\boldsymbol{y},\boldsymbol{y}^{(L)}) = MSE_\Theta(\boldsymbol{y},\boldsymbol{y}^L) = MSE_\Theta(\{\boldsymbol{u}(p_j;\boldsymbol{\mu})\}_{j\in\mathcal{Q}_{out}},\{\tilde{\boldsymbol{u}}(p_j;\tilde{\boldsymbol{\mu}}\}_{j\in\mathcal{Q}_{out}}) \tag{3.5}$$

where $\{\boldsymbol{u}(p_j;\boldsymbol{\mu})\}_{j\in\mathcal{Q}_{out}}$ are the real values of the solution, measured at the output locations, while $\{\tilde{\boldsymbol{u}}(p_j;\tilde{\boldsymbol{\mu}}\}_{j\in\mathcal{Q}_{out}}$ are the estimated values of the solution at the output locations, corresponding to the estimated parameter values. The major observation that can be made looking at the formulation of the loss functional is that it does not contain any explicit information regarding the values of the characteristic parameters $\boldsymbol{\mu}$; thus, such values do not have to be known neither at training stage and they configure as a physically-aware *by-product*, estimated by the hidden layers of the RB-DNN model.

Few remarks are worth to follow.

- The value of the solution at the input/output locations may not coincide with the numerical value encoded in its vectorial representation; this may happen either because the FOM basis is not interpolatory (think about high-order splines in IsoGeometric Analysis (IGA)) or because, even with classical P1 FE, the sensors which are supposed to detect the input values are located in points of the domain that cannot be "traced" efficiently with a reasonably simple mesh. Such an issue can be tackled employing interpolation techniques; in the case of interest, anyway, the usage of Lagrangian basis functions and the selection of the grid points as locations of the input/output values prevents us from any effort in this direction.
- The structure of the RB layer must be adapted to whether the trainable part of the NN provides an estimate of the parameters $\boldsymbol{\mu}$ or directly of the parameter-dependent coefficients $\{\Theta_n^{q_n}(\boldsymbol{\mu})\}_{q_n=1}^{Q_n}$ and $\{\Theta_f^{q_f}(\boldsymbol{\mu})\}_{q_f=1}^{Q_f}$.
- In case the training dataset comes from numerical simulations rather than from real physical measurements (as in our case of interest), it is immediate to observe that it can be used both as NN training dataset and as the *snapshots*' set for the construction of the Reduced Basis via POD in the offline phase of the RB method, thus lightening significantly the overall computational burden. Also, in case the training appears to require a larger amount of data, it is possible to follow a two-steps data generation pipeline; in a first phase, the FOM problem is solved for $N_{\boldsymbol{\mu}}^{FOM}$ different parameter values and from such solutions the Reduced Basis is computed via POD. Once this is done, provided that the accuracy of such basis matches the requirements in terms of accuracy, it is possible to solve (at low cost) the ROM problem for other different $N_{\boldsymbol{\mu}}^{ROM}$ parameter values; the union of the two datasets can be ultimately used as training dataset for the RB-DNN model.
- The dimensionality of the proposed models is reasonably small, being typically made of $4-5$ fully-connected layers featuring from 512 to 32 neurons each. Thus, the training phase can be carried out in few minutes in a simple computational environment as the one described in Appendix C, while the testing on fresh data samples just takes fractions of a second, involving only the evaluation of a feed-forward model and the resolution of a small linear system.

3.2.2 Applications of RB-DNN models: an overview on the obtained results

This Subsection is devoted to a brief discussion of the results obtained with RB-DNN models on several test cases. In particular, in the first part a summary of the results on steady parametrized PDEs contained in [2] is given; in the second part, instead, an extension of the model to handle unsteady parametrized problems, named Reduced Basis Convolutional Neural Network (RB-CNN), is presented and few numerical results obtained on a simple affinely parametrized benchmark problem are discussed.

RB-DNN model on Steady Parametrized PDEs

In [2], the performances of the RB-DNN model have been evaluated on 3 different test cases, with increasing level of complexity, namely a simple affinely parametrized advection-diffusion problem, a non-affinely parametrized elliptic problem and the steady Navier-Stokes problem. In the last two test cases, an approximated affine parametrization of the FOM/RB arrays has been retrieved via EIM/DEIM algorithms. In all cases, the performances of the model have been evaluated for different settings of its main hyperparameters, i.e. the number of layers and of nodes per layer in the trainable MLP part of the network, the number of input and output locations N_{in} and N_{out}, the number of training snapshots $N_{\boldsymbol{\mu}}$ and, for the non-affinely parametrized problems, the number of affine basis functions chosen in the EIM/DEIM algorithms.

The RB-DNN model proved to give very good results on all the problems it has been tested on, both in terms of estimation of the underlying parameters and of reconstruction of the values of the

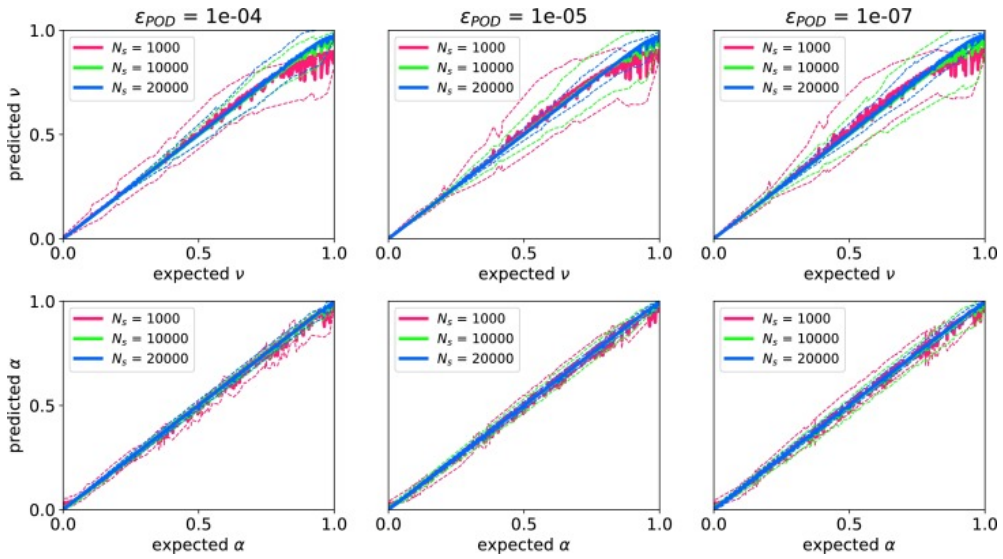


Figure 3.4: Estimation of the two parameters of the affinely parametrized advection-diffusion test case ν (top row) and α (bottom row) (normalized over the respective ranges), for different POD tolerances of the RB problem and different number of training samples. The results refer to the test dataset, which is composed of 600 data points, and are averaged over 30 trainings of the RB-DNN network. The regions between dashed lines display the 95% confidence intervals corresponding to the average lines with the matching colors. Image taken from [2].

solution at the output locations. As explanatory test case, Figure 3.4 shows the results of the estimation of the two parameters involved in the formulation of the affinely parametrized advection-diffusion problem; in particular $\nu \in [0.5; 10]$ represents the diffusion coefficient, while $\alpha \in [0; \pi/6]$ defines the characteristic angle of the advection field $b(\boldsymbol{x}; \alpha) =: \sin(\alpha) b_1(\boldsymbol{x}) + \cos(\alpha) b_2(\boldsymbol{x})$. It is immediate to see that the errors on both parameters are small (quantitatively, they are of the order of $10^{-2} - 10^{-3}$ for both ν and α), with the only exception of the ones got on ν for high values of ν itself; a heuristic justification for this is provided in the paper. Moreover, the accuracy of the model increases as the number of training samples increases and as the tolerance of the POD used to generate the Reduced Basis decreases; these trends are both expected and reasonable. Average relative errors on the values of the solution at the output locations (which are taken in a portion of the domain where no input point has been sampled, so that $\mathcal{P}_{in} \cap \mathcal{Q}_{out} = \emptyset$) are of the order of $10^{-3} - 10^{-4}$ for all the considered architectures, thus indicating again an appreciable degree of accuracy.

Another relevant test performed in [2] aims at comparing the behaviour of the RB-DNN architecture with the one of other two architectures, that do not show any embedded RB-solver layer. In particular, the comparison of the RB-DNN model is performed, again on the simple affinely parametrized test case, with the following three networks:

1. MLP_{all}: a simple MLP network. It takes as input the values of the solution at some randomly sampled locations $\mathcal{P}_{in}$ and it gives, as output, the values of the solution at some different locations $\mathcal{Q}_{out}$ as well as the values of the parameters; those must be involved explicitly in the loss function in this case, since there's no physical layer allowing for an implicit retrieval.
2. $MLP_{\boldsymbol{\mu}}$: a MLP network, identical to the previous one, except the fact that it gives as output only the values of the characteristic parameters.
3. MLP_{out}: a MLP network, identical to the first one, except the fact that it gives as output only the values of solution at the output locations.

The findings are very important: indeed it has been observed that, regardless of the number of input-output locations and of the size of the training dataset, the behaviour of the RB-DNN model is very close to the one of the MLP networks in terms of the estimation of both the parameters (where MLP_{all} and $MLP_{\boldsymbol{\mu}}$ have been considered) and of the values of the solution

(where MLP_{all} and MLP_{out} have been considered). Such a result tells us that the RB-DNN model is able to provide a very good approximation of the characteristic parameters, comparable to the one of a NN featuring them as output, despite retrieving them just a a by-product. Also, a strong dependency on the size of the training dataset has been found in terms of accuracy (both for RB-DNN and MLP-based networks), while the number of input-output locations appeared to play a minor role. Finally, the time taken by the training process is very short (of the order of 10 minutes on a simple laptop as the one described in Appendix C); this configures as an unquestionable strength.

Other than the results discussed in [2], further investigations on the performances of RB-DNN models have been carried out by the author of the present work in [35]. In particular, the study of the dependency of the model performances on the network sizing and the possibility of adding a trainable regularizer layer in parallel to the RB-solver have been analyzed, considering a simple affinely parametrized elliptic problem as test case.

RB-CNN model on Unsteady Parametrized PDEs

While PINNs have been developed and tested on time-dependent PDE problems from the very beginning (see [1]), in [2] the performances of RB-DNNs have been evaluated only on steady problems. The author of the present report, anyway, has already investigated the possibility of employing RB-DNN architectures on problems showing a dependency with respect to time. In particular in [35] an initial feasibility study on the theme has been carried out, discussing how time dependent data could be processed and how a dimensionality reduced solver couls be embedded within the already developed RB-DNN architecture; in [36] such reasoning has been enriched with further observations and some preliminary tests have been performed.

Two main ways of realizing the extension of RB-DNNs models to unsteady problems, depending on the way the dimensionality reduction of the parametrized PDE problem is tackled.

1. **Spatial Dimensionality Reduction & Multistep Time Marching Scheme**: the first possibility is represented by the usage of ROM techniques that allow to perform a dimensionality reduction along the spatial dimension only, keeping unchanged the number of DOFs along the temporal one. The "time-iterated" RB method described in the beginning of Subsection 2.2.3 represents the most straightforward choice in this sense. In such a context, the idea would be to develop a model that takes as input the values of the solution for a pre-defined number of initial equispaced time instants; extracts from such values the relevant features of the parametrized problem at hand (i.e. the parameters or the parameter-dependent coefficients); using a multistep time marching scheme, predicts the values of the solution at one or more subsequent timesteps and returns them in output. Notice that the data given as input would be used by the RB-projected multistep solver to predict the values of the solution at the desired subsequent timesteps; because of this, either FOM data in $\mathcal{P}_{in} \equiv \mathcal{Q}_{out} \equiv \Omega_h$ or RB-projected data have to be passed, yielding to a significant applicative limitation with respect to the steady models.
2. **Spatio-Temporal Dimensionality Reduction**: the second possibility would be to use a spatio-temporal dimensionality reduction technique (as the ST-LSPG method by *Choi & Carlberg*; see [30]). In this case, the value of the solution at all the spatio-temporal FOM locations would be available just by solving, in the Least-Squares sense, an underdetermined linear system, that can be also fast-assembled at online stage taking advantage of affine parametrization (eventually approximated via EIM-DEIM algorithms). Thus, the overall structure and application of the model would be similar to the one of the RB-DNNs described in [2]. Also, the input data would be used only to infer the parameter values, not being directly employed by the RB solver: this allows to potentially choose $\mathcal{P}_{in} \neq \mathcal{Q}_{out} \neq \Omega_h$. The model complexity may be anyway increased if the parameters are time-dependent; in such case, smart ways of estimating their trends over time should be designed, in order to reduce as much as possible the amount of quantities to be estimated.

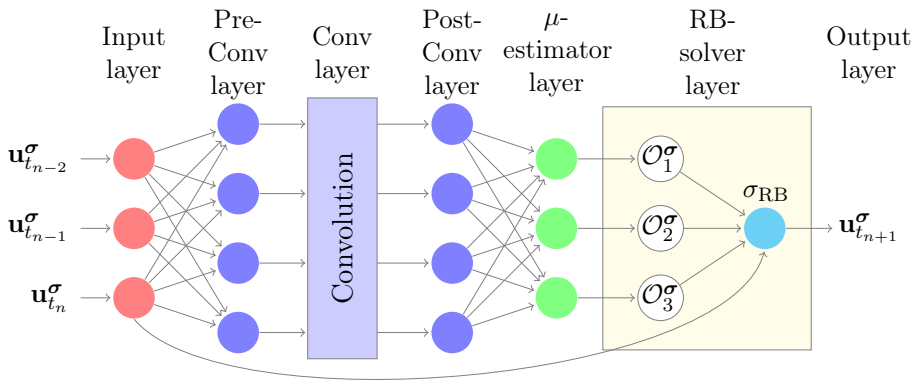


Figure 3.5: Architecture of the RB-CNN model, developed in [36] as first extension of the RB-DNN model to handle unsteady problems. It is based on spatial dimensionality reduction via classical RB method and the time marching is achieved via implicit Backward Differentiation Formulas (BDF) multistep methods. Image taken from [36]

Among the two, the second alternative features unquestionable advantages, as the possibility of "freely" choosing the input data and of performing a one-shot estimation, even if the parameters are time-dependent (provided that a smart way of estimating their trends is found). Surely, then, the implementation of a network of such kind configures as the most straightforward extension in the context of RB-DNNs towards the goal of handling unsteady problems. Anyway, designing an efficient and robust ST-LSPG solver, able to deal with both linear and non-linear problems and that can be embodied within the structure of a trainable NN, is not an easy task to carry out; because of this, in [36] the author of the present work has started tackling the problem of the extension of RB-DNN models to unsteady problems implementing and testing the first alternative.

The developed model has been called Reduced Basis Convolutional Neural Network (RB-CNN), since convolutional layers have been employed; its architecture can be visualized in Figure 3.5 and it is structured as follows:

- **Input** (Red in figure): the NN takes as input the FOM solutions to the PDE at hand, projected over the RB subspace generated from the training data and for M_{in} consecutive equispaced timesteps
- **Trainable Part of the Network** (Blue in figure): the trainable part of the architecture is made of three regions. In the first region, several $ReLU$-activated fully-connected layers process the data, extracting features on top of which convolutional operations can be more effective. In the second region, $1D$ convolutional layers perform convolutions on the data, along their temporal dimension, in order to extract features representative of the time evolution process; additionally max-pooling operations, put in series to the convolutions, allow to reduce the dimensionality of the data and to provide a regularizer effect. Finally, in the third region, another set of flanked $ReLU$-activated fully-connected layers combines the quantities estimated by the convolutions to ultimately estimate the characteristic parameters (Green in figure). The usage of CNN architectures, able to exploit the time-dependent nature of the input data, justifies the change in the name of the model.
- **Reduced Solver Layer** (Cyan in figure): the deterministic solver layer takes as input the estimated parameters and the values of the RB-projected solution at the previous $M_{steps} \leq M_{in}$ timesteps; then, using an implicit BDF time marching scheme with M_{steps} steps (with $2 \leq M_{steps} \leq 6$), it estimates the value of the RB-projected solution at the subsequent timestep
- **Output**: the network returns the values of the RB-projected solution at the timestep following the ones given as input and the values of the estimated characteristic parameters at such timestep. The loss is built as the MSE over the estimated values of the solution.

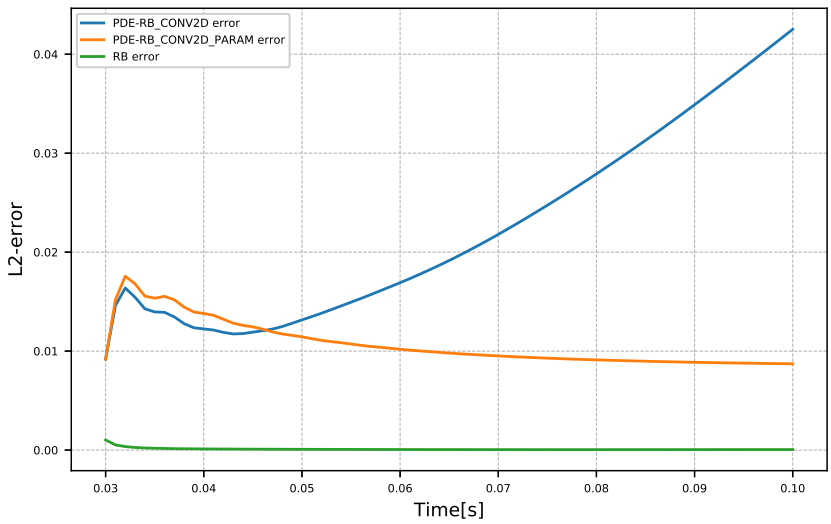


Figure 3.6: Trend of the relative errors (with respect to a reference FOM solution) in $L^2(\Omega)$-norm for the BDF6+RB solver fed with the "real" parameter values (in Green), the BDF6+RB solver fed with the parameter values estimated by a RB-CNN model applied over the first $M_{in} = 30$ timesteps (in Orange) and the RB-CNN model applied iteratively over time (in Blue). Image taken from [36]

Readily, this model needs to be applied iteratively over time and, in order to be started, it is necessary to know the values of the solution at least for $M_{in} \geq M_{steps}$ initial equispaced time instants. This poses severe limitations on its applicability. In [36] the model performances have been evaluated on top of a simple affinely parametrized elliptic problem, characterized by a piecewise constant diffusivity coefficient over the computational domain (i.e. the so-called *thermal block problem*). Figures 3.7-3.8 show the performances of the model in terms of both parameters' estimation and solution reconstruction at one-step, using the BDF6 time marching scheme. Figure 3.6 displays instead the trend of the errors arising from the application of the RB-CNN model either iteratively over time (predicting new parameter values at every cycle - Blue) or only at the first timestep (the value of the solution at the subsequent ones is then obtained by applying the RB-projected BDF method, using the parameter values originally estimated by the model - Orange).

It quite immediate to observe that the RB-CNN performances are very good in terms of one-step prediction; in particular the relative error on the solution equals 0.094% in $L^2(\Omega)$-norm, while the one on the parameters is of the order of 1.5%. Anyway, when the model is applied starting from a set of solutions known only at some initial time instants, then the errors increase; for instance, the relative $L^2(\Omega)$-norm errors on the solution values relax at the 1% level if the parameters (assumed to be constant) are estimated just once, while they show a diverging trend if the RB-CNN model is applied iteratively over time. Such a behavior, also considering that it has been obtained on a problem that does not show too complicated dynamics, clearly drives towards the choice of second alternative of the ones proposed before; the implementation and testing of a ST-LSPG-based model will be the subject of a future study by the author of this work. Finally, the training time is of the order of 45 minutes in a computational environment as the one described in Appendix C, thus being absolutely feasible even on machines featuring constraints in terms of memory occupation and of computational power.

One-step parameters estimation for the RB-CNN model with BDF6

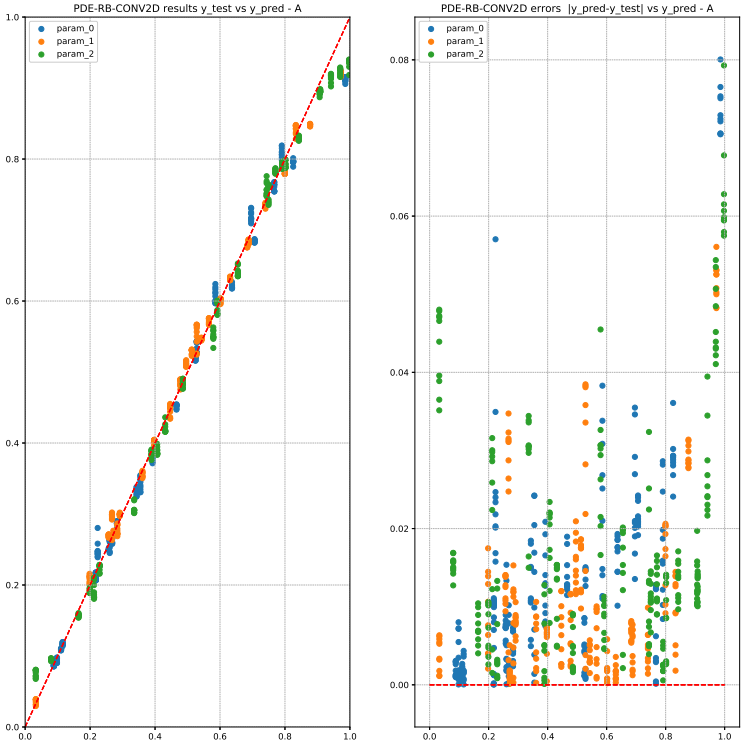


Figure 3.7: Results of the estimation of the 3 characteristic parameters of the unsteady thermal block problem by the RB-CNN model, with BDF6 method in the RB-solver layer and optimal hyperparameters choice. On the right, the plot shows the estimated values of the parameters (in y) vs the true ones (in x). On the left, the plot shows the absolute error on the parameters estimation (in y) vs the true values of the parameters (in x). Image taken from [36]

One-step solution estimation for the RB-CNN model with BDF6

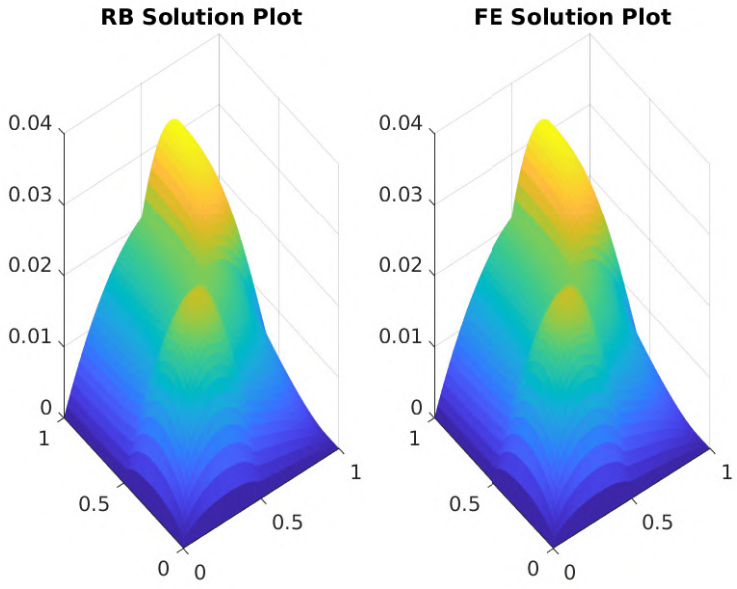


Figure 3.8: Results of the one-step estimation of the solution of the unsteady thermal block problem by the RB-CNN model, with BDF6 method in the RB-solver layer and optimal hyperparameters choice. The reported solutions refer to a parameter value and to a time instant which have been randomly sampled from the ones available in the test set. On the right, the plot shows the reconstructed solution; on the left, the reference FOM solution is displayed. Image taken from [36]

Chapter 4

Heart Electrophysiology

In this chapter, we recall the fundamentals of cardiac electrophysiology and we describe the State-Of-Art numerical methods that are employed to approximate it.

In particular, Section 4.1 provides an illustration of the cardiac electrical activity (Subsection 4.1.1) and an explanation of how such activity can be non-invasively recorded and analyzed via ECGs (Subsection 4.1.2). Additionally, Appendix A contains an overview on the heart anatomy and on the cardiovascular system. In Section 4.2 the numerical methods employed in this project to approximate heart EP are reported. Specifically Subsection 4.2.1 describes how heart EP can be modeled via PDEs; Subsection 4.2.2 explains how the considered models have been approximated numerically and Subsection 4.2.3 shows the achieved numerical results, both in terms of epicardial activation maps and of ECG signals. Additionally, Subsection 4.2.4 provides an explanation of how ROM techniques across both space and time dimensions have been put in place to reduce the overall computational burden; numerical results to assess their performances are also reported. Finally, Section 4.3 defines and discusses the Inverse Problem of Electrocardiography; in particular Subsection 4.3.1 provides the mathematical description of the problem, while Subsection 4.3.2 features a small literature review, listing and briefly detailing the most relevant numerical methods.

4.1 Electrical Activity of the Heart

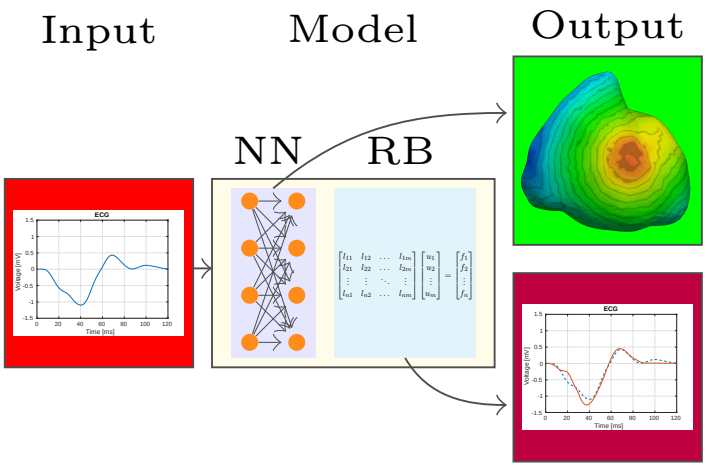


Figure 4.1: Basic structure of the developed PDE-aware DL model, where the input and the outputs are highlighted

The vast majority of the information contained in this Section are taken from the book *Fisiologia*, by Cindy L. Stanfield [37].

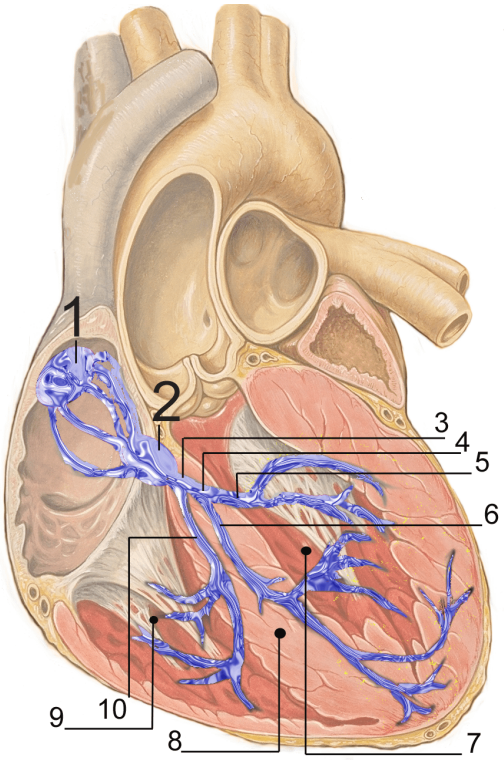


Figure 4.2: Structure of the heart electrical conduction system. 1.SA node 2.Atrio-ventricular (AV) node 3.His bundle 4.Left bundle branch 5.Left Posterior Fascicle 6.Left anterior fascicle 7.Left ventricle 8.Interventricular septum 9.Right ventricle 10.Right bundle branch
Image by J. Heuser - Own work, based upon https://commons.wikimedia.org/wiki/File:Heart_anterior_view_coronal_section.jpg by Patrick J. Lynch (Patrick J. Lynch; illustrator; C. Carl Jaffe; MD; cardiologist Yale University Center for Advanced Instructional Media), CC BY 2.5, https://commons.wikimedia.org/w/index.php?curid=1734607

4.1.1 The Heart Electrical Conduction System

In order for the heart to adequately pump the blood into the vascular system, the cardiac muscle has to contract in a synchronous way; the two atria have to contract first and lately the two ventricles. Such contractions are coordinated by a complex electrical conduction system, which determines the sequence of the excitation of the cardiac muscular cells and that will be described in the following.

The Structure of the Cardiac Conduction System

The cardiac muscle does not receive commands from the central nervous system in order to contract; indeed all its contractions generate from signals that originate within the cardiac muscle itself. Because of this peculiar property, the contractile activity of the cardiac muscle is referred to as *myogenous*. The capability of the heart to generate signals that periodically activate its contractions, auto-generating its own rhythm, is called **autorhytmicity**. Such a property is due to the action of a small fraction of "modified" muscular cells, called *autorhytmic cells*, that are essential for the pumping activity of the heart, coordinating all its beats. As a whole, the autorhytmic cells constitute the **heart conduction system**, which can be visualized in Figure 4.2; the cells that generate the contractile force are instead called **contractile cells**.

There are two types of autorhytmic cells:

1. **Pacemaker Cells**: as suggested by their name, these cells determine the rhythm (or pace) of the cardiac beat, generating the regular train of action potentials. Despite being distributed almost everywhere across the heart, pacemaker cells are mainly located in two

distinct regions of the myocardium: the **sino-atrial (SA) node** (on the superior wall of the right atrium) and the **AV node** (close to the tricuspid valve in the interatrial septum). The discharge frequencies of the SA node and of the AV node are anyway different; indeed the former are higher than the latter, which in turn implies, being the two nodes themselves connected by conduction fibers, that the SA node coordinates the depolarization of the AV node and, ultimately, of the whole heart.

2. **Conduction Fibers**: they are specialized in quickly conducting the action potentials generated by the pacemaker cells from one point of the myocardium to another one, triggering in this way the contractions of the different regions of the cardiac muscle. It is worth clarifying, at this point, that all the fibers of the cardiac muscle are able to lead action potentials; anyway, the conduction fibers differ from the contractile ones since they have a larger diameter, thus being able to perform a much faster electrical conduction ($\approx 4\ m/s$ vs. $\approx 0.5\ m/s$).

Origin and Conduction of the Electrical Impulse during a Heart Beat

The rapid transmission of the action potentials from the pacemaker cells to the conduction and contractile fibers is made possible by the presence of **gap junctions**, that allow the ions to flow from one cell to another. Other than by gap junctions, the cardiac muscular fibers are also connected by *desmosomes*, areas where protein fibers adhere one with the other, creating a physical link that gives a good mechanical resistance.

The sequence of electrical events that, in physiological circumstances, are responsible for the heart beat is the following:

1. An action potential is started at the SA node. From such node, the action potential travels towards the AV node via the *internodal ways*, which are part of the heart conduction system. While moving along the internodal ways, the signal also diffuses through the atrial muscular mass, via the *interatrial ways*, triggering the atrial contraction.
2. The impulse is conducted by the cells of the AV node, which anyway transmit the action potential at a lower velocity compared to the one of the other cells of the conduction system. This implies that, before advancing towards the ventricles, the electrical impulse is delayed by approximately 100 ms (*AV nodal delay*); such a delay is fundamental, since it allows the atrial contraction to have been completed before the ventricular one gets started.
3. From the AV node, the impulse travels through the *AV bundle* (also called **His bundle**), which is located in the interventricular septum. Notice that the AV node and the His bundle are the only electrical connections between the atria and the ventricles, which are otherwise separated by the fibrous skeleton.
4. The signal briefly travels through the His bundle and then it spreads into the branches of the **left bundle** and of the **right bundle**, which lead the impulse to the left and right ventricle respectively.
5. From the left and right bundles, the impulse travels through a dense net of conduction fibers, called **Purkinje fibers**, which become more and more numerous and small as their distance from the original branches increases. Such a net diffuses through the ventricular myocardium, ascending from the apex of the heart to the semi-lunar (SL) valves, and it allows the electrical impulse to reach all the ventricular cells, ultimately triggering the ventricular contraction.

The cardiac beat is almost always triggered by the action potentials that are originated at the SA node; this is due to two reasons. The first one is that the SA node discharge frequency is higher than the one of the AV node (70 impulses/min vs. 50 impulses/min). Secondly, the action potentials that are originated at the SA node have to travel through the AV node before reaching the ventricles; when such passage happens, the pacemaker cells of the AV node enter in a refractory period that prevents them from generating other action potentials. Anyway, it is important to remark that, if the SA node stops generating action potentials, then the AV node

is actually able to generate them on its own, triggering the ventricular contraction; in the same way, the AV node can take over the heart beat in case the internodal ways connecting it to the SA node are for some reason blocked. Additionally, if even the AV node is not able to trigger the ventricular contraction, then the heart has another emergency system; indeed some cells of the Purkinje fibers, called *idioventricular pacemaker cells*, can take over the heart beat, yet being characterized by a lower discharge frequency ($\approx 30 - 40$ impulses/min).

Apart from describing how the electrical impulse travels through the heart conduction system, it is also important to depict how it spreads in the whole heart, reaching the contractile fibers and triggering the muscular contractions of the atria (first) and of the ventricles (lately). At this aim, it is possible to say that the ordered propagation of the electrical impulses in the cardiac muscle generates a **depolarizing wavefront**, commonly called "excitation wave". The term "depolarizing" is used because the potential lies at about $-65\ mV$ at rest, while it reaches at most $30 - 40\ mV$ upon excitation, thus moving towards $0\ mV$ and reducing its magnitude. Muscular contraction follows the propagation of this wave. According to the sequence of events described before, the excitation wave originates at the SA node and then diffuses from the endocardium to the epicardium, fully depolarizing the atria. After that, the wave travels towards the AV node, following the conduction fibers of the AV bundle. The AV node acts as a bottleneck, inducing a delay of about 100 ms that allows atrial contraction to be completed before the ventricular one is started; this is fundamental, in order for the blood to pass from the atria to the ventricles, through the AV valves, obeying to the pressure gradients. Once the electrical impulses reach the left and the right bundles and, lately, the Purkinje fibers, they are quickly driven down to the apical region of the ventricles. During such travel, the interventricular septum is depolarized and several studies, as the one reported in [38], have assessed that this activation proceeds from the left surface to the right surface of the septum; this phenomenon has implications also in terms of ECG waves, as it will be made clear in Subsection 4.1.2. Lastly, the depolarization wavefront spreads through the whole ventricular myocardium, ascending until the basal region; such a bottom-top progression actually makes sense, considering that the blood leaves the ventricles from the SL valves, that are located at their top.

Epicardial Breakthroughs (EBTs) Localization

In this paragraph we discuss the localization of Epicardial Breakthroughs (EBTs), defined as the sites of emergence of a radially propagating wavefront at the epicardial surface. Our references are [38] and [39], where, despite under different working hypothesis, the authors managed to reconstruct epicardial activation maps, from which both the EBTs positions and the areas of Latest Epicardial Activation (LEA) can be identified.

In the work by *Durrer et al.* [38], the information regarding the time course and the instantaneous distribution of the excitation of the normal human heart have been obtained by studying reperfused isolated hearts, taken from seven individuals who died from various cerebral condition, but who had no history of any cardiac disease. Upon exciting the hearts synchronously in three endocardial areas for 5 ms, their activation maps have been obtained, both at the endocardium and at the epicardium. Focusing on epicardial activation, that serves us as a reference to assess the consistency of the numerical simulations, early EBTs have been found in the area *pretrabecularis* (i.e. the anterior paraseptal area) of the right ventricle, $20 - 25\ ms$ after the initial endocardial activation. From such area an almost radial spread of the excitation towards both the apex and the base has been observed, with LEAs having been identified at the posterobasal area of the right ventricle. The isochrones are not concentric, but rather activation proceeds tangentially, ultimately reaching the aforementioned LEAs. Larger variations have been instead noticed in the epicardial activation of the left ventricle; indeed three EBT areas have been identified, namely (1) a small anterior paraseptal area, close to the AV sulcus; (2) an anterior paraseptal area, located halfway between apex and base; (3) a posterior paraseptal area, about halfway between the apex and the base. Also, a small EBT area at the posterior apex has been sometimes detected; EBT areas tended to become confluent after 30 ms and,

apart from individual variability, concentric isochronic lines of propagation of the depolarizing wavefront have been observed. The LEAs of the left ventricle reflected the variability of its activation pattern; in most cases, anyway, they have been found in the posterobasal paraseptal region, despite having been sometimes detected also in a more lateral location, basically at the left ventricle free wall. Finally, it is worth remarking that the isochronic lines encoding the propagation of the depolarizing wavefront have been found to be quite irregular. This is mainly due to the fact that the excitation does not spread over the epicardial surface, but rather from the endocardium to the epicardium; as a consequence, slight variations in the wall thickness induce relevant differences in the epicardial activation of adjacent areas.

The study of *Wyndham et al.* [39] aims at describing only the ventricular epicardial activation. As stated by the authors, the study proposes itself to resolve the question whether the data collected in [38] on reperfused hearts actually reflected the epicardial activation in the intact human heart; the findings have given an overall positive answer to such question, despite some elements of novelty have been brought to light. In particular, the data have been obtained by recording the epicardial potentials of 11 patients featuring ECG signals with normal QRS complex (so, somehow, healthy ECGs) and undergoing open heart surgery due to a coronary artery disease. Upon constructing epicardial activation maps from the collected data, EBTs have been identified in three to five sites for each patient; the earliest EBT has been always found in the anterior paraseptal area of the right ventricle (as in [38]). Subsequent EBTs have been then found in the inferior right ventricle (often closer to the base than to the apex), in the anterolateral left ventricle (paraseptally, closer to the base than to the apex) and in the inferior left ventricle (paraseptally, somehow halfway between the apex and the base). Notice that all such EBT areas have been also found in [38], apart from the one located in the inferior right ventricle; this configures as the main novelty element of this study. Regarding the timing, all EBT sites appeared $7-48\ ms$ after the QRS onset, with the earliest one occurring at $7-25\ ms$. LEAs have always occurred at the basal segments, in accord with [38] and confirming the fact the human heart tends to activate in a more or less apex-to-base direction (other than from the endocardium to the epicardium). Specifically 5 patients showed LEAs at the anterior right ventricle (ARV), while 4 featured it at the inferior right ventricle (IRV) and 2 at the inferior left ventricle (ILV). None featured LEA at the anterior left ventricle (ALV).

4.1.2 Recording of the Heart Electrical Activity: the Electrocardiogram (ECG)

The electrocardiogram (ECG) is a non-invasive means of monitoring the electrical activity of the heart, that consists in the recording of the flux of electrical current that crosses the heart during a cardiac cycle. In particular, ECG signals are recorded by means of electrodes placed on the skin of the patient; since the heart EP is highly synchronized (as already described in Subsection 4.1.1), relatively high potentials can be detected even at skin level and, thus, a partitioning of the different phases of the cardiac cycle can be inferred.

All the information given in the following are borrowed from [37], [40] and [41].

Placement of the ECG Electrodes

Several different ECG lead systems have been developed throughout the medical history; among those the 12-lead ECG system is definitely the most used in actual practice. As the name suggests, the 12-lead system features 12 leads, that are derived by means of 9 (or 10) electrodes placed on the patient skin; at this point it is also useful to clarify the difference between **ECG leads** and **ECG electrodes**. An electrode is an conductive pad which is attached to the skin, enabling in this way the recording of the electrical currents; a lead, instead, is a graphical description of the electrical activity of the heart and it is obtained by comparing the information coming from different electrodes. Thus the number of electrodes and of leads is typically different, as it happens for instance in the 12-lead ECG system.

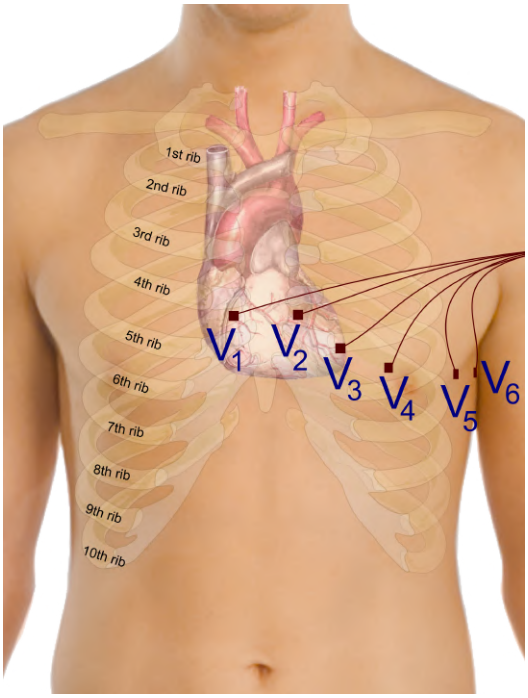


Figure 4.3: Positions of the 6 chest electrodes in the 12-lead ECG system. Recall that the 4 limb leads can be placed both far down the limbs (arms/legs) and closer to the hips/shoulders.
Image by Mikael Häggström - Own work, CC0, https://commons.wikimedia.org/w/index.php?curid=20064293

The positioning of the electrodes in the 12-lead ECG system can be visualized in Figure 4.3. The electrodes can be clearly divided into two groups:

1. **Limb or Peripheral Electrodes**: those are 4 electrodes, placed at the "periphery" of the human body, i.e. at the left and right arms, avoiding thick muscle (respectively **left arm (LA)** and **right arm (RA)**) and at the left and right legs, avoiding bony prominences (respectively **left leg (LL)** and **right leg (RL)**). These electrodes can be placed both far down on the limbs or closer to the hips/shoulders, as long as they are symmetric. Actually, the RL electrode can be omitted, reducing to 3 the number of peripheral electrodes; by doing so, the remaining three electrodes form what is referred to as the **Einthoven's triangle**, an imaginary equilateral triangle built around the heart, that takes its name from Willem Einthoven, the dutch physiologist who developed the first "rudimental" ECG machine in 1895, winning for that the *Nobel Prize in Physiology or Medicine* in 1924.
2. **Chest or Precordial Electrodes**: those are 6 electrodes, placed on the left half of the chest. In particular their positions are the following:
   - **V1**: at the 4^{th} intercostal space, in the right parasternal line
   - **V2**: at the 4^{th} intercostal space, in the left parasternal line
   - **V3**: midway between V2 and V4
   - **V4**: at the 5^{th} intercostal space, in the left mid-clavicular line.
   - **V5**: horizontally even with V4, in the left anterior axillary line
   - **V6**: horizontally even with V4, in the left mid-axillary line

Other than the 9 (or 10) physical electrodes, the 12-lead ECG system makes also use of a virtual electrode, ideally recording the *center-of-the-heart potential* and located in the center of Einthoven's triangle. Such electrode is known as **Wilson's central terminal** and its potential (called *Wilson's potential*) is produced by averaging the measurements of the three limb electrodes, so that

$$V_W = \frac{1}{3}(RA + LA + LL)$$

Definition of the ECG leads

The leads that characterize the 12-lead ECG system are obtained by combining the potentials measured by the aforementioned electrodes; in particular they are all *bipolar*, i.e. they result from the difference between the potential measured by an electrode (called positive or exploring electrode) and the one measured by another electrode (called negative or reference electrode). They can be classified as follows:

- **Limb Leads (or Einthoven's leads)**: Leads I, II and III are called limb leads and are computed just by subtracting (in pairs) the electric potentials measured by the three peripheral electrodes LA, RA and LL, which form the Einthoven's triangle. In particular:
  - Lead I is the difference between the left arm electrode LA and the right arm one RA, so that
  $$I = LA - RA$$
  - Lead II is the difference between the left leg electrode LL and the right arm one RA, so that
  $$II = LL - RA$$
  - Lead III is the difference between the left leg electrode LL and the left arm one LA, so that
  $$III = LL - LA$$

  Notice that, by Kirchhoff's law, it trivially holds that
  $$I + III = II$$
  so one of the leads is redundant and it does not bring any new amount of information with respect to the other two.

- **Augmented Limb Leads (or Goldberger's Leads)**: Leads Augmented Vector Right (aVR), Augmented Vector Left (aVL) and Augmented Vector Foot (aVF) are called augmented limb leads; they are still derived from the three limb electrodes, but they all use the *Goldberger's central terminal* as their negative pole. The Goldberger's central terminal has actually a different value for each of the three augmented limb leads, being it computed as the average of the potentials measured by the two "not-considered" electrodes. In particular:
  - Lead aVR is computed as the difference between the right arm potential RA and the average of the remaining two peripheral electrodes LA and LL, so that
  $$aVR = RA - \frac{1}{2}(LA + LL) = \frac{3}{2}(RA - V_W) = -\frac{1}{2}(I + II)$$
  - Lead aVL is computed as the difference between the left arm potential LA and the average of the remaining two peripheral electrodes RA and LL, so that
  $$aVL = LA - \frac{1}{2}(RA + LL) = \frac{3}{2}(LA - V_W) = \frac{1}{2}(I - III)$$
  - Lead aVF is computed as the difference between the left left leg potential LL and the average of the remaining two peripheral electrodes LA and RA, so that
  $$aVF = LL - \frac{1}{2}(LA + RA) = \frac{3}{2}(LL - V_W) = \frac{1}{2}(II + III)$$

  Deriving from the 3 limb electrodes, the augmented limb leads are redundant with respect to the Einthoven's leads, not adding any additional information about the cardiac electrical activity; anyway, they allow to enlarge the field of view on the frontal plane, making it easier and more immediate to evaluate the frontal cardiac electrical axis and, in a broader sense, the electrical state of the heart.

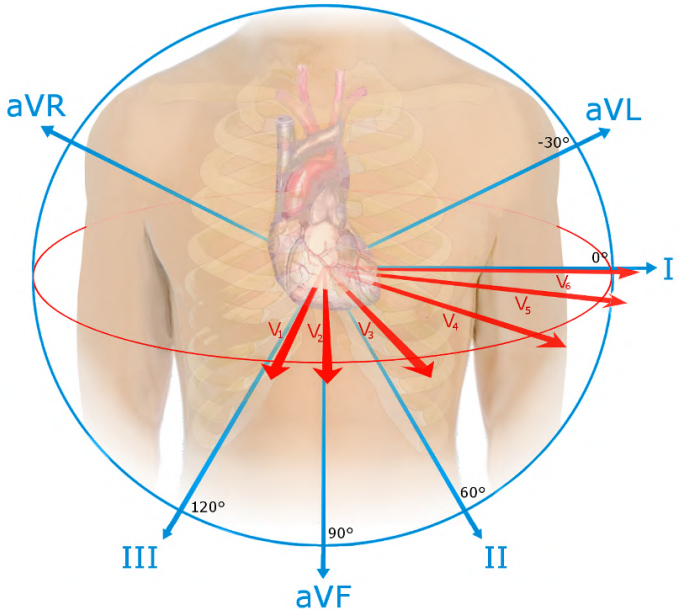


Figure 4.4: Directions of the electric vectors associated to the 12 ECG leads.
Image by Npatchett - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=39235260

- **Chest or Precordial Leads**: Leads V1, V2, V3, V4, V5 and V6 are called chest leads, since they are all obtained by computing the difference between the potential recorded at the corresponding chest electrode and the Wilson's potential V_W, so that

$$V_i = \tilde{V}_i - V_W \quad \forall\, i \in \{1, \dots, 6\}$$

  being $\tilde{V}_i$ the potential recorded by the i-th precordial electrode.

The placement of the ECG electrodes and the computation of the ECG leads is obviously not random; indeed all ECG leads are derived in such a way that they can give information about the electrical activity of the heart from different angles, as represented in Figure 4.4. The 6 limb leads (i.e. I, II, III, aVR, aVL and aVF) provide information about the electrical activity that is happening at different angles across the vertical frontal plane; indeed they form the basis of the *hexaxial reference system* (or *Cabrera reference system*), which is used to estimate the cardiac electrical axis in the frontal plane. At this aim, it is worth remarking that often lead aVR is replaced by its opposite (named -aVR), so that the gap between leads I and II is filled and a uniform span of the angles from $-30°$ (aVL) to $120°$ (III) is obtained. Based on their positions, the 6 limb leads can be furthermore divided into two groups: the **inferior (diaphragmal) limb leads** (II, aVF, III), which mainly observe the inferior aspect of the left ventricle, and the **lateral limb leads** (aVL, I, -aVR), that primarily observe the lateral aspect of the left ventricle. Conversely, the 6 precordial leads are characterized by electric vectors who span in the horizontal plane, since their reference electrode, the Wilson's central terminal, is ideally placed in the center of the thorax; thus they allow to monitor the heart electrical activity somehow horizontally from right to left, offering a different point of view with respect to the limb leads. A possible classification of the precordial electrodes is the following:

- **Septal Leads (V1-V2)**: they are located in front of the heart, thus primarily observing the interventricular septum and occasionally displaying waves related to the aspect of the right ventricle.
- **Anterior Leads (V3-V4)**: they are basically in front of the left ventricle, thus mainly observing the electrical activity of its anterior wall
- **Anterolateral Leads (V5-V6)**: they are placed on the side of the thorax, thus monitoring the activity of the lateral wall of the left ventricle.

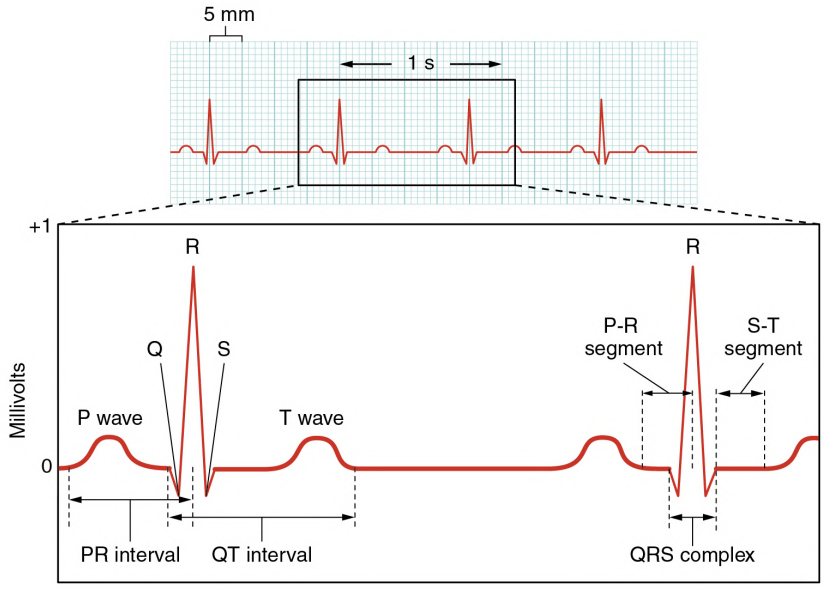


Figure 4.5: Schematic view of two physiological ECG waveforms.
Image by OpenStax College - Anatomy & Physiology, Connexions Web site. http://cnx.org/content/col11496/1.6/, Jun 19, 2013., CC BY 3.0, https://commons.wikimedia.org/w/index.php?curid=30148219

Notice that none of the 12 leads is adequate to monitor the electrical activity of the right ventricle, which is partially seen only by the two septal precordial leads V1 and V2; this is surely a limitation, since pathological conditions localized only at right ventricular level, as right ventricular ischemia/infarction for instance, cannot be detected. On the other side, anyway, the shapes of the body surface signals are mainly influenced by the electrical activity of the left ventricle, since the thickness of its myocardial layer is much larger than the one of the right ventricle and of the two atria, thus inducing a much larger flux of ions and, ultimately, much larger currents. Because of this, monitoring the activity of the left ventricle alone is already able to give a lot of information about the electrical state of the heart and it allows to detect a very broad set of common pathological conditions.

Standard ECG Waves Morphology

A schematic view of a "standard" ECG signal is shown in Figure 4.5. The various deflections in ECG signals are commonly denoted in alphabetic order, adopting uppercase letters for significant deflections (P, Q, R, S, T, U) and lowercase letters for the ones having smaller amplitudes (p, q, r, s, t ,u). The temporal sequence of these waves tends to remain constant (at least in physiological conditions), reflecting the depolarization/repolarization cycle of the heart, but their amplitude and polarity can vary, depending on the considered lead.

The following waves can be observed:

- **P wave**: the P wave is associated to atrial depolarization; its polarity is positive in all leads but in aVR.
- **QRS complex**: the QRS complex is actually a sequence of waves and it represents the largest deflection in the ECG signal. It is associated to ventricular depolarization and it results in a succession of peaks, whose amplitude and polarity highly depend on the considered lead. The three waves (Q, R and S) may not be detected in all leads and that they can even merge; this induces a high degree of variability in the QRS complex morphology.
- **T wave**: the T wave is associated to ventricular repolarization and, in physiological conditions, it has the same polarity as the P wave (thus being positive in all leads but in aVR).

More specifically, the start of the T wave is thought to correlate with the start of the repolarization of epicardial cells (which repolarize first, despite having been depolarized last); the peak of the T wave is related to the full repolarization of the epicardium; the end of the T wave corresponds to the repolarization of the midmyocardial cells.

- **U wave**: after the T wave, it is sometimes present a small deflection known as U wave. Its origin is yet not fully understood: some studies link it to the repolarization of the papillary muscles or of the Purkinje fibers, while others relate it to late repolarization of some midmyocardial cells or to afterpotentials due to the ventricular stretch.

Other than the waves themselves, also the duration of the intervals and the segments between one wave and another carry a lot of information. The most relevant intervals and segments are the following:

- **PR interval**: the PR interval is computed as the amount of time elapsed between the onset of the P wave and the one of the R wave (or, more in general, of the QRS complex). Since the P wave is associated to the depolarization of the atria and the QRS complex to the one of the ventricles, the PR interval gives an estimate of the AV conduction time (i.e. of the AV nodal delay).
- **PR segment**: the PR segment is defined as the fraction of ECG signal going from the end of the P wave to the onset of the QRS complex; it is important since it defines the *isoelectric line* of the ECG curve.
- **QT interval**: the QT interval is computed as the amount of time elapsed between the onset of the Q wave (i.e. of the QRS complex) and the end of the T wave; thus it accounts for an estimate of the total time needed by the ventricles to depolarize and repolarize and, thus, to contract (ventricular systole).
- **ST segment**: the ST segment is defined as the part of the ECG curve going from the end of the S wave (at the so-called *J-point*) to the onset of the T wave. It gives an estimate of the duration of the plateau phase characterizing the action potential of all contractile cells and it is important since it may be altered in a wide range of pathological conditions. In particular, acute myocardial ischemia can be detected from the ST segment, since it causes *ST segment deviation* (either elevation or depression); the amount of deviation is determined by comparing the magnitude of the signal at the PR segment and at the J-point.
- **RR interval**: the RR interval is computed as the distance (in time) between two subsequent peaks of the R wave, thus representing the amount of time elapsed between two consecutive heart beats.

An example of a healthy 12-lead ECG recording can be found in Figure 4.6, while Table 4.1 summarizes the main features of a physiological 12-lead ECG.

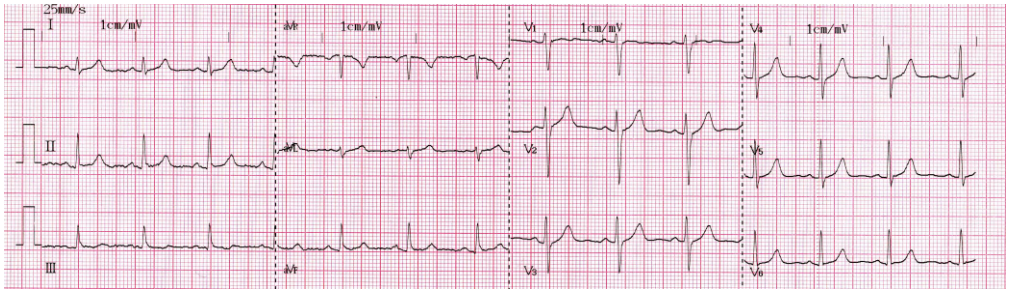


Figure 4.6: Sample of 12-lead ECG waveforms in physiological conditions.
Image by Ptrump16 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=77817932

Component	Morphology	Amplitude (mV)	Duration (ms)	Notes
P wave	• positive in I,II,aVF, $v_4 - V_6$ • diphasic in $V_1 - V_3$ • negative in aVR	≈ 0.2	≤ 110	Shape is generally smooth
QRS complex	• small Q waves in I, aVL, V_5, V_6 • large upright R waves in I, II, $V_4 - V_6$ • large deep S wave in aVR, V_1, V_2	• ≥ 0.5 in at least one limb lead • ≥ 1 in at least one chest lead • ≤ 3	• ≤ 120 • Q wave: ≤ 40	From V_1 to V_6 the R waves get taller and the S waves get smaller
T wave	• same polarity as QRS in at least 5 limb leads • positive in I, II, $V_2 - V_6$ • negative in aVR	• ≥ 0.2 in $V_3 - V_4$ • ≥ 0.1 in $V_5 - V_6$	$160 - 270$	Rounded and asymmetrical, with more gradual ascent than descent
PR interval	//	//	$120 - 220$	//
ST segment	isoelectric, with a slight upward concavity towards the T wave	//	≈ 80	It can be elevated up to 0.2 mV in some chest leads and it is never depressed more than 0.05 mV
QT interval	//	//	• ≤ 450 in males • ≤ 470 in females • ≥ 300	//
TQ interval	//	//	$550 - 700$	//
RR interval	//	//	$850 - 1000$	//

Table 4.1: Standard features of healthy 12-lead ECG recordings. In particular, the morphology, amplitude and duration of each wave is reported, together with the duration of the most relevant intervals

4.2 Numerical Approximation of Heart Electrophysiology

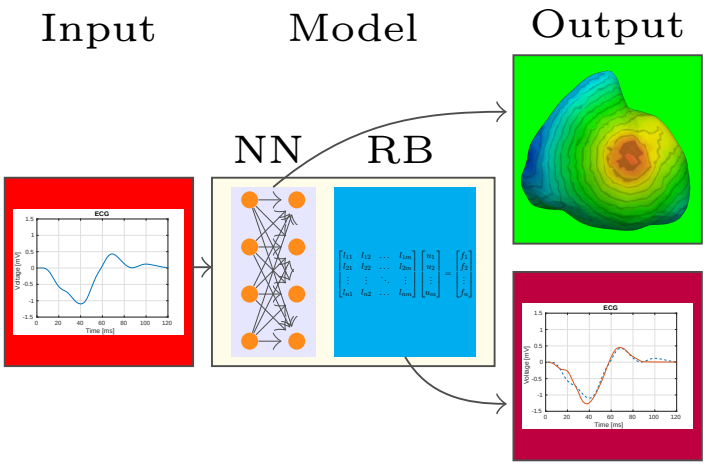


Figure 4.7: Basic structure of the developed PDE-aware DL model, where the input and the outputs are highlighted

This section aims at describing the mathematical models that have been used for the numerical approximation of the cardiac EP. Since a precise and exhaustive description of such models is not the main goal of this work, we suggest the interested reader to refer to *Mathematical Cardiac Electrophysiology* by Franzone et al. [41] or to *Computing the Electrical Activity in the Heart* by Sundnes et al. [42] for a more complete and organic presentation.

Subsections 4.2.1, 4.2.2 and 4.2.3 investigate how heart EP can be modeled and approximated numerically; thus they are linked to the dataset generation process (sections red, green and purple of Figure 4.7). Conversely, Subsection 4.2.4 considers how ROM techniques can be applied to reduce the computational burden of heart EP simulations, thus being related to the RB section (in cyan) of the model in Figure 4.7.

4.2.1 Modeling

The human body consists of billion of cells, that are connected by means of various coupling mechanisms. When constructing mathematical models for the electrical activity in the tissues, one possible approach would be to model each cell as a separate unit and lately to couple them together using proper models for the known coupling mechanisms. However, the significantly large number of cells prevents from choosing this approach, unless a very small portion of tissue is taken into account; also, this would lead to a level of detail which typically goes far beyond what it is needed and useful at clinical level. Thus, it emerges the need of finding more efficient techniques, that, rather than looking precisely at the microscopic cellular scale, are able to offer a more macroscopic understanding of the phenomenon under investigation. At this aim, a standard approach is to study volume averaged quantities, i.e. identifying a quantity observed at point P as the average of such quantity in some small volume centered around P. The choice of the volume directly relates with the level of refinement that is desired and, as a rule of thumb, it must be small compared to the scale of the problem to be studied, but big compared to the microscopic cellular scale.

For the case of electrophysiology, applying a local volume averaging technique allows to model the human body as a volume conductor, for which the Maxwell equation relating the electric and the magnetic field is:

$$rot(E) + \frac{\partial B}{\partial t} = 0 \qquad in\ \Omega \times [t_0, T] \tag{4.1}$$

where E and B represent the electric and of the magnetic fields respectively, Ω a reference

domain and t_0 and T the initial and final times respectively (for a more detailed description of Maxwell's equations, see e.g. [43]). Furthermore, we can add the hypothesis that the two fields are *quasi-static*, meaning that the effect of their temporal variations may be disregarded, without any significant loss in accuracy; under the *quasi-static condition*, equation (4.1) becomes:

$$rot(E) = 0 \qquad in\ \Omega \times [t_0, T] \tag{4.2}$$

which in turn entails, via the Stokes theorem, that the electric field E is conservative, under the hypothesis that the domain Ω is simply connected. Thus, we may write E as the negative gradient of a potential u, i.e.

$$E = -\nabla u \qquad in\ \Omega \times [t_0, T] \tag{4.3}$$

Finally, the current density in the volume conductor can be written as

$$J = \boldsymbol{D}E = -\boldsymbol{D}\nabla u \qquad in\ \Omega \times [t_0, T] \tag{4.4}$$

where $\boldsymbol{D}$ is the conductivity tensor characterizing the considered medium.

The Heart Tissue: the Bidomain Model

The volume conductor modeling assumption introduced before can be efficiently used to model the propagation of the electric signal from the heart to the torso (as it will be made clear in the following), but it is definitely not suitable to model the depolarization/repolarization cycles that characterize heart EP. Indeed, as already explained in Section 4.1, the heart muscular cells are excitable cells, that are able to actively respond to an electrical stimulus and whose behavior mainly depends on the potential difference that is established across the cell membrane. Thus, in order to properly simulate the electrical activity of the heart, it is necessary to come up with a mathematical model which is able to describe how the intracellular and the extracellular potentials evolve over time.

A model of such kind is the **bidomain model**, introduced by Tung in [44]. The model makes use of a volume-averaging approach similar to the one described before for the volume conductor, but it additionally partitions the heart tissue in two distinct subdomains: the *intracellular domain* Ω_i and the *extracellular domain* Ω_e. Anyway, upon a homogenization process, the two domains can be assumed as continuous, so that they fill the complete volume of the heart muscle; the justification for this comes from the fact that both domains are actually connected and spread all over the cardiac tissue. On the one side, indeed, the extracellular domain is clearly connected, since it is the "free space" between the cells; on the other side, instead, also the intracellular domain can be viewed as connected, since the muscle cells are bridged by *gap-junctions*, that allow ions and small molecules to pass from one cell to another without entering the extracellular domain. Furthermore, the two domains are separated by the *cell membrane* Γ, which, acting as an electrical insulator, allows to establish a potential difference at its borders; having anyway assumed that both Ω_i and Ω_e are continuous and that fill the whole heart volume, we must extend such hypothesis also to the cell membrane Γ.

The previous hypothesis brings with it the counter-intuitive entailment that each point in the domain is simultaneously part of the intracellular domain, of the extracellular domain and of the cell membrane. As a consequence, all points in the heart volume domain are associated to three different potentials:

1. u_i: the intracellular potential
2. u_e: the extracellular potential
3. $v := u_i - u_e$: the transmembrane potential

Under the *quasi-static* assumption, we can define the currents in the two domains as:

$$J_i = -\boldsymbol{D}_i \nabla u_i \qquad in\ \Omega_H \times [t_0; T] \tag{4.5a}$$

$$J_e = -\boldsymbol{D}_e \nabla u_e \qquad in\ \Omega_H \times [t_0; T] \tag{4.5b}$$

being $\boldsymbol{D}_i$ and $\boldsymbol{D}_e$ the conductivity tensors in the intracellular and extracellular domains respectively and Ω_H the reference heart domain.

As said before, the cell membrane acts as an electrical insulator between Ω_i and Ω_e; therefore it is natural to expect an accumulation of charge in the two domains. Anyway, due to the small thickness of the membrane, any accumulation of charge on one side of it would induce an accumulation of the same amount of charge, with opposite sign, on the other side; upon the domain homogenization process, this yields to a null overall charge accumulation at any point and at any time instant, i.e.

$$\frac{\partial}{\partial t}(q_i + q_e) = 0 \qquad in\ \Omega_H \times [t_0; T] \tag{4.6}$$

being q_i and q_e the intracellular and extracellular charges, respectively.

In each domain, the net current into a point must be equal to the sum of the rate of charge accumulation at that point and of the current exiting the domain at that point, since the cell membrane behaves as a capacitor; in mathematical terms, this writes as:

$$-div(J_i) = \frac{\partial q_i}{\partial t} + A_m I_{ion} - A_m I^i_{app} \qquad in\ \Omega_H \times [t_0; T] \tag{4.7a}$$

$$-div(J_e) = \frac{\partial q_e}{\partial t} - A_m I_{ion} + A_m I^e_{app} \qquad in\ \Omega_H \times [t_0; T] \tag{4.7b}$$

where I_{ion} is the ionic current across the cell membrane, assumed to be positive if directed from the intracellular domain to the extracellular one and $I^{i,e}_{app}$ are externally applied currents, typically responsible for the initial activation of the heart tissue in the intracellular and extracellular domains. Notice that currents are typically measured per unit area of the cell membrane, while the other quantities are measured per unit volume; thus current terms are pre-multiplied by the constant A_m, which represents the area of cell membrane per unit volume. Also, the following compatibility condition must hold:

$$\int_{\Omega_H} I^i_{app} = \int_{\Omega_H} I^e_{app} \tag{4.8}$$

The combination of (4.6) and (4.7) yields to:

$$div(J_i) + div(J_e) = A_m(I^i_{app} - I^e_{app}) \qquad in\ \Omega_H \times [t_0; T] \tag{4.9}$$

which, in terms of electric potentials via (4.5), becomes:

$$-div(\boldsymbol{D}_i \nabla u_i) - div(\boldsymbol{D}_e \nabla u_e) = A_m(I^i_{app} - I^e_{app}) \qquad in\ \Omega_H \times [t_0; T] \tag{4.10}$$

Ultimately, the amount of charge separated by the cell membrane Γ relates to the transmembrane potential v and to cell membrane capacitance per unit area C_m as:

$$v = u_i - u_e = \frac{q_i - q_e}{2A_m C_m} \qquad in\ \Omega_H \times [t_0; T] \tag{4.11}$$

Rearranging the terms and taking time derivatives, we find:

$$A_m C_m \frac{\partial v}{\partial t} = \frac{1}{2}\frac{\partial (q_i - q_e)}{\partial t} \qquad in\ \Omega_H \times [t_0; T] \tag{4.12}$$

The combination of (4.5), (4.6), (4.7) and (4.12) yields to:

$$div(\boldsymbol{D}_i \nabla u_i) = A_m\Big(C_m \frac{\partial v}{\partial t} + I_{ion} - I^i_{app}\Big) \qquad in\ \Omega_H \times [t_0; T] \tag{4.13}$$

Equations (4.10) and (4.13) properly describe the variations of the potentials u_i, u_e and v.

The last step, prior to the definition of proper boundary and initial conditions, is to rewrite the equations so that the intracellular potential u_i disappears and the only unknowns are u_e and v. In particular, performing the substitution $u_i = u_e + v$ in (4.13) and in (4.10) and rearranging the terms, we get:

$$A_m\Big(C_m\frac{\partial v}{\partial t} + I_{ion}\Big) - div(\boldsymbol{D}_i\nabla v) - div(\boldsymbol{D}_i\nabla u_e) = A_m I^i_{app} \qquad in\ \Omega_H \times [t_0;T] \quad (4.14)$$

$$-div(\boldsymbol{D}_i\nabla v) - div((\boldsymbol{D}_i + \boldsymbol{D}_e)\nabla u_e) = A_m(I^i_{app} - I^e_{app}) \quad in\ \Omega_H \times [t_0;T] \quad (4.15)$$

This formulation of the bidomain equations, originally introduced in [45] and [46], has the great advantage of being suitable to operator splitting techniques, that allow for an extremely fast and efficient computation of the solution. Thus, it has become the standard formulation.

In order to develop a well-posed problem, we must enrich (4.14)-(4.15) with proper boundary and initial conditions. Additionally, a model for the ionic current I_{ion} must be introduced and conductivity tensors able to account for the anisotropy of the heart tissue must be defined.

Boundary Conditions If we assume the heart to be surrounded by a non-conductive medium, then the flux of both the intracellular and the extracellular potential at the domain boundary $\partial\Omega_H$ must be set to 0; thus the boundary conditions to be set on the problem unknowns u_e and v are

$$\boldsymbol{n}_H \cdot (\boldsymbol{D}_i\nabla v + \boldsymbol{D}_i\nabla u_e) = 0 \qquad on\ \partial\Omega_H \times [t_0;T] \quad (4.16)$$

$$\boldsymbol{n}_H \cdot (\boldsymbol{D}_e\nabla u_e) = 0 \qquad on\ \partial\Omega_H \times [t_0;T] \quad (4.17)$$

being $\boldsymbol{n}_H$ the outward unit normal vector to $\partial\Omega_H$.

Initial Conditions Since only the time derivative of the transmembrane potential v is involved in the problem formulation, the system must be supplemented only with an initial condition on such variable, of the form:

$$v(\cdot, t_0) = v_0 \qquad in\ \Omega_H \quad (4.18)$$

Typically the heart is assumed to be "deactivated" at the initial time instant, thus v_0 is taken as constant and equal to a value between $-80\ mV$ and $-90\ mV$. In case the transmembrane potential is supposed to be normalized in $[0;1]$ (as required by many ionic models), then v_0 is chosen equal to 0.

Ionic Model The way the ionic current is computed configures as a very important aspect for the modeling of cardiac EP. Over the last century, lots of research efforts have been devoted towards the development of models which are able to describe how action potentials can be initiated (by excitable cells as pacemaker cells in the heart or neurons) and propagated. In particular, several physiology-based models have been developed; such models, among which the Hodgkin-Huxley model (1952) appears as a milestone (see [47]), aim at investigating even the most relevant sub-cellular processes, which bring to the actuation and propagation of action potentials.

Because of their extreme level of refinement, these models are not suitable for the sake of the investigation of phenomena on large spatial and temporal scales. Rather, simplified models are preferred for providing phenomenologically consistent action potentials at the minimal computational cost. All the phenomenological ionic models are expressed in terms of the normalized transmembrane potential $\tilde{v}$ and of the so-called *gating variable* w, that allows to model the refractoriness of cells. They follow the general kinetics

$$\begin{cases} \dfrac{d\tilde{v}}{dt} = f(\tilde{v}, w) \\ \dfrac{dw}{dt} = g(\tilde{v}, w) \end{cases} \quad (4.19)$$

Different expressions of f and g define different ionic models. The actual transmembrane potential v can be then recovered by a rescaling of the normalized one $\tilde{v}$ as

$$v = 100\tilde{v} - 80 \quad [mV] \tag{4.20}$$

where it is assumed that v takes values in the interval $[-80\ mV; 20\ mV]$.

The most famous and widely used reduced ionic model is the FitzHugh Nagumo (FHN) model, of which multiple versions do exist; one of the most employed is characterized by the following expressions for f and g.

$$\begin{aligned} f(v,w) &= -Kv(v-a)(v-1) - w; \\ g(v,w) &= \epsilon(v - \gamma w) \end{aligned} \tag{4.21}$$

Here a plays a key role in handling the fast dynamics due to the inward sodium current and it acts as an oscillation threshold, above which an action potential is fired. In particular a is taken in $[0;1]$ and a typically used value is $a = 0.15$; ϵ and γ are positive constants, whose values are of the order of 10^{-1}; K is a positive constant, commonly set to 8.

In the context of this project, we have employed a modified version of the FHN model, which takes the name of Aliev-Panfilov (AP) model after Rubin Aliev and Alexander Panfilov, who developed it in 1996 [48]. This choice is motivated by the fact that the AP model has been specifically conceived as a variation of the FHN one, aimed at adequately describing the dynamics of the electric pulse propagation in the canine myocardium. In particular, the authors noticed that the FHN model and all its already developed variants where able to successfully describe lots of qualitative aspects of the cardiac excitation, but they all failed in simulating many quantitative aspects, such as the shape of the action potential or the restitution property of the cardiac tissue, defined as the relation linking the action potential duration (APD) with the Cycle Length (CL). This last property is crucial for a correct simulation of the pulse propagation in the myocardium (especially if cases of cardiac arrhythmia are studied), since a quite strong dependency between the two aforementioned quantities is present. The expressions of f and g in the AP model are:

$$\begin{aligned} f(v,w) &= -Kv(v-a)(v-1) - vw \\ g(v,w) &= \left(\epsilon_0 + \frac{\mu_1 w}{v + \mu_2}\right)(-w - kv(v-b-1)) \end{aligned} \tag{4.22}$$

where common values for the cell-related constants are $K = 8$, $a = 0.15$, $b = 0.15$, $\epsilon_0 = 0.002$. The weighting term $\left(\epsilon_0 + \frac{\mu_1 w}{v + \mu_2}\right)$ has been specifically added to account for a finer tuning of the myocardial restitution curve; the values of μ_1 and μ_2 can be adjusted to obtain results as close as possible to the experimental observations. Additionally, in order to get the desired scaling properties, it is necessary also to scale the time variable as

$$t[ms] = 12.9\ t[t.u.] \tag{4.23}$$

As presented in [48], the AP model exceeds the performances of the classical FHN one on cardiomyocites, leading to a much better approximation of the action potential shape and duration and of the cardiac tissue restitution curve.

In the context of the modeling of the cardiac EP, the phenomenological ionic models are embedded in the system by adding the equation describing the behavior of the dimensionless gating variable w and by defining the ionic current as $I_{ion}(v,w) = -f(v,w)$. Thus, if the AP model is considered, the following equations are added to the system:

$$I_{ion}(v,w) = Kv(v-a)(v-1) + vw \tag{4.24}$$

$$\frac{\partial w}{\partial t} = g(v,w) = \left(\epsilon_0 + \frac{\mu_1 w}{v + \mu_2}\right)(-w - Kv(v-b-1)) \tag{4.25}$$

Readily, since the time derivative of the gating variable w is involved in the problem formulation, a proper initial condition on such variable, of the form

$$w(\cdot, t_0) = w_0 \qquad in\ \Omega_H \tag{4.26}$$

must be added. w_0 is commonly taken as constant and equal to 0.

Heart Tissue Anisotropy The conductive properties of the heart muscular tissue are significantly anisotropic; indeed the heart muscle is made of fibers and the conductivity is much higher in the direction of the fibers rather than in the cross-fiber one. Furthermore, the muscle cells are organized in sheets; this allows, at every point in the heart volume, to define three characteristic directions for the conductivity: parallel to the fibers, perpendicular and coplanar to the fibers and orthogonal to the fibers "sheet". Thus, at any point in Ω_H, it is possible to define a set of orthonormal vectors $\boldsymbol{a}_l$, $\boldsymbol{a}_t$ and $\boldsymbol{a}_n$ where $\boldsymbol{a}_l$ is longitudinal to the fibers, $\boldsymbol{a}_t$ is transversal to the fibers and $\boldsymbol{a}_n$ is directed normally to the fibers sheet. Expressed in terms of this basis, the intracellular and extracellular local conductivity tensors read as:

$$\boldsymbol{D}^*_{i,e} = \begin{bmatrix} \sigma_{l_{i,e}} & 0 & 0 \\ 0 & \sigma_{t_{i,e}} & 0 \\ 0 & 0 & \sigma_{n_{i,e}} \end{bmatrix} \tag{4.27}$$

The coefficients $\sigma_{l_{i,e}}$, $\sigma_{t_{i,e}}$ and $\sigma_{n_{i,e}}$ may in general depend on the position $\boldsymbol{x} \in \Omega_H$; for the sake of simplicity, anyway, we will assume them to be constant, working under the so-called *homogeneous anisotropy assumption.*

Defining $\boldsymbol{A}_f$ as the matrix featuring $\boldsymbol{a}_l$, $\boldsymbol{a}_t$ and $\boldsymbol{a}_n$ as columns, at any $\boldsymbol{x} \in \Omega_H$ it holds that $J = \boldsymbol{A}_f J^*$, being J and J^* the current densities expressed with respect to the global and local coordinate system respectively; additionally, $\boldsymbol{A}_f^{-1} = \boldsymbol{A}_f^T$, being $\boldsymbol{A}_f$ orthogonal. Thus

$$J_{i,e} = \boldsymbol{A}_f J^*_{i,e} = \boldsymbol{A}_f \boldsymbol{D}^*_{i,e} E^*_{i,e} = \boldsymbol{A}_f \boldsymbol{D}^*_{i,e} \boldsymbol{A}_f^{-1} E_{i,e} = \boldsymbol{A}_f \boldsymbol{D}^*_{i,e} \boldsymbol{A}_f^T E_{i,e} \tag{4.28}$$

so that the intracellular and extracellular global conductivity tensors are defined at each point $\boldsymbol{x} \in \Omega_H$ as:

$$\boldsymbol{D}_i = \boldsymbol{A}_f \boldsymbol{D}^*_i \boldsymbol{A}_F^T \tag{4.29}$$

$$\boldsymbol{D}_e = \boldsymbol{A}_f \boldsymbol{D}^*_e \boldsymbol{A}_f^T \tag{4.30}$$

Also, taking advantage of the fact that the conductivity tensor is diagonal in the local coordinate system, it is possible to write a generic entry of the conductivity tensor (both intracellular and extracellular) in the global coordinate system as:

$$\boldsymbol{D}_{ij} = \boldsymbol{a}_l^i \boldsymbol{a}_l^j \sigma_l + \boldsymbol{a}_t^i \boldsymbol{a}_t^j \sigma_t + \boldsymbol{a}_n^i \boldsymbol{a}_n^j \sigma_n \qquad for\ i, j = 1, 2, 3 \tag{4.31}$$

Finally, assuming $\sigma_{n_{i,e}} = \sigma_{t_{i,e}}$, we recover the *axially isotropic case* (see [49]) and the conducibility tensor can be ultimately written as

$$\boldsymbol{D}_{i,e} = \sigma_{t_{i,e}} \boldsymbol{I} + (\sigma_{l_{i,e}} - \sigma_{t_{i,e}}) \boldsymbol{a}_l \boldsymbol{a}_l^T \tag{4.32}$$

Coupling with the Torso

To model the transmission of the electric signal from the heart (epicardium) to the body surface, we can rely on the volume conductor approximation introduced at the beginning of the Section; indeed the torso is not made of excitable cells, as the heart, and thus there is no need of coming up with a model able to simulate the behavior of the transmembrane potential.

So, we can consider a domain Ω_T representing the human torso, whose boundary $\partial\Omega_T$ is partitioned into the epicardium $\Gamma_H \equiv \partial\Omega_H$ and the body surface Γ_B. Differently than in the case of

the heart, it is reasonable to assume that there are no current sources or sinks in the medium and that there is no built-up of charge at any point in Ω_T; thus, for any small volume V ,the net current leaving it must be zero. If S defines the surface of V, then it holds that:

$$\int_S \boldsymbol{n} \cdot J_T \; dS = 0 \qquad \forall \; S : S = \partial V \quad \forall \; V \subset \Omega_T \tag{4.33}$$

being $\boldsymbol{n}$ the outward unit normal to S and J_T the current density in the torso. By the divergence theorem, it then holds that:

$$\int_V div(J_T) \; dV = 0 \qquad \forall \; V \subset \Omega_T \tag{4.34}$$

which in turn entails that

$$- \, div(J_T) = 0 \qquad in \; \Omega_T \tag{4.35}$$

Finally, recalling (4.4), we end up with the following generalized Laplace equation

$$- \, div(\boldsymbol{D}_T \nabla u_T) = 0 \qquad in \; \Omega_T \tag{4.36}$$

which describes the diffusion of the electric potential u_T in the torso. $\boldsymbol{D}_T$ represents the torso conductivity tensor; with respect to the case of the heart, an isotropic conduction can be assumed in the torso, but the value of the conducibility may vary from point to point, depending on the type of tissue present at each point.

Finally, regarding the boundary conditions, it is natural to assume a homogeneous Neumann one at the body surface, which models the fact that the human body is surrounded by air or, in general, by an electrical insulating medium. The boundary condition imposed at the heart-torso interface Γ_H, instead, depends on the type of model that has to be constructed. In this case, we have worked under the *isolated heart assumption* (see [50] for additional details); this means that the heart and the torso are seen as completely decoupled, as if the heart were surrounded by an insulating medium when solving its EP. Under such an hypothesis, the boundary conditions on the intracellular and extracellular potentials in the heart EP are imposed as homogeneous Neumann ones (see (4.16) - (4.17)), while the one on the torso potential is expressed as a Dirichlet boundary condition, that forces the torso potential to equal the extracellular cardiac one on Γ_H. In terms of equations, the aforementioned boundary conditions can be written as:

$$\boldsymbol{n}_B \cdot \boldsymbol{D}_T \nabla u_T = 0 \qquad on \; \Gamma_B \tag{4.37a}$$

$$u_T = u_e|_{\Gamma_H} \qquad on \; \Gamma_H \tag{4.37b}$$

being $\boldsymbol{n}_B$ the outward unit normal vector to Γ_B.

<u>*Remark*</u>: Notice that the full-decoupling of the heart from the torso allows to solve the two problems independently one from the other. Indeed the heart EP can be solved first, via the bidomain equations completed with the ones coming from a phenomenological ionic model (as the AP model in our case); then, once the extracellular potential at the epicardium is known, it can be used as a Dirichlet boundary datum to solve the signal transmission problem at the Torso. Such a simplification readily also has its drawbacks, as it is shown in [50]; indeed it does not allow to impose the continuity of the electric potential fluxes at the heart-torso interface (as it is instead done in the monolithic fully-coupled formulation), resulting in estimated ECG signals that often show the correct shape, but that feature abnormal magnitudes. Additionally, observe that the "torso problem" is a steady problem, just modeling the diffusion of the electric potential from the epicardium through the human body at any time instant in $[t_0, T]$; thus no initial condition on the torso potential u_T has to be imposed.

Model Summary

To summarize, we have modeled the human cardiac EP by choosing the **bidomain model**, supplemented with the **AP ionic model** and under the **isolated heart assumption**. In particular, the bidomain equations, supplemented with the ones coming from the AP model and closed by proper initial and boundary conditions, allow to compute the transmembrane potential and the extracellular potential. Under the isolated heart assumption, we decoupled the torso problem from the heart one, deriving the body surface potentials by solving at any time instant a generalized Laplace equation with null forcing term, featuring homogeneous Neumann boundary conditions at the body surface and a Dirichlet boundary condition at the epicardial heart-torso interface. The complete set of equations is reported hereafter in Problem 4.1; equations related to the heart are in red, while the ones set in the torso are in blue. As a further addition, we have made explicit the parametric dependency of some of the terms involved in the formulation; specifically, following the notation used by *Pagani et al.* in [51], we embodied all the scalar parameters involved in the system (from the ones of the ionic model to the ones characterizing the fibers orientation and conducibility) inside a single parameter vector $\boldsymbol{\mu} \in \mathcal{P} \subset \mathbb{R}^p$. Since all our test cases have been run on a fixed geometric setting, we suppose the parameter vector $\boldsymbol{\mu}$ not to contain quantities relative to geometric properties of the domain; as a consequence both domains Ω_H and Ω_T are assumed to be parameter-independent. Furthermore, we have supposed $I^i_{app} = I^e_{app} =: I_{app}$ (which is trivially consistent with the compatibility condition (4.8)).

Problem 4.1: Forward Heart EP Problem with Bidomain Equations

$$
\begin{cases}
A_m\left(C_m \dfrac{\partial v}{\partial t} + I_{ion}(v, w; \boldsymbol{\mu})\right) - div(\boldsymbol{D}_i(\boldsymbol{\mu})\nabla v) - div(\boldsymbol{D}_i(\boldsymbol{\mu})\nabla u_e) = A_m I_{app}(\boldsymbol{\mu}) & in\ \Omega_H \times [t_0, T] \\
-div(\boldsymbol{D}_i(\boldsymbol{\mu})\nabla v) - div((\boldsymbol{D}_i(\boldsymbol{\mu}) + \boldsymbol{D}_e(\boldsymbol{\mu}))\nabla u_e) = 0 & in\ \Omega_H \times [t_0, T] \\
I_{ion}(v, w; \boldsymbol{\mu}) = Kv(v-a)(v-1) + vw & in\ \Omega_H \times [t_0, T] \\
\dfrac{\partial w}{\partial t} = g(v, w; \boldsymbol{\mu}) = \left(\epsilon_0 + \dfrac{\mu_1 w}{v + \mu_2}\right)(-w - Kv(v - b - 1)) & in\ \Omega_H \times [t_0, T] \\
-div(\boldsymbol{D}_T(\boldsymbol{\mu})\nabla u_T) = 0 & in\ \Omega_T \times [t_0, T] \\
n_H \cdot (\boldsymbol{D}_i(\boldsymbol{\mu})\nabla v + \boldsymbol{D}_i(\boldsymbol{\mu})\nabla u_e) = 0 & on\ \Gamma_H \times [t_0, T] \\
n_H \cdot (\boldsymbol{D}_e(\boldsymbol{\mu})\nabla u_e) = 0 & on\ \Gamma_H \times [t_0, T] \\
n_B \cdot \boldsymbol{D}_T(\boldsymbol{\mu})\nabla u_T = 0 & on\ \Gamma_B \times [t_0, T] \\
u_T = u_e|_{\Gamma_H} & on\ \Gamma_H \times [t_0, T] \\
v(\cdot, t_0) = v_0(\boldsymbol{\mu}) & in\ \Omega_H \\
w(\cdot, t_0) = w_0(\boldsymbol{\mu}) & in\ \Omega_H
\end{cases}
$$

4.2.2 Numerical Methods

Weak Formulation

Before moving to the FOM approximation of Problem 4.1, that will be obtained by means of the Galerkin FE method, we state its weak formulation. In particular, since Problem 4.1 features a full-decoupling between the heart and the torso dynamics, the latter configuring just as a steady diffusive problem with boundary conditions given by the former, we can handle the two weak formulations separately.

The Heart For what concerns the heart, the weak formulation of the bidomain model, coupled with any phenomenological ionic model, reads as follows:

Problem 4.2: Bidomain Equations + Ionic Model Weak Formulation

Let $t > t_0$. Given $v_0, w_0 \in L^2(\Omega_H)$, $I_{app}(t;\boldsymbol{\mu}) \in L^2(\Omega_H)$, find $u_e(\cdot,t;\boldsymbol{\mu}) \in H^1(\Omega_H)$, $v(\cdot,t;\boldsymbol{\mu}) \in H^1(\Omega_H)$ and $w(\cdot,t;\boldsymbol{\mu}) \in L^2(\Omega_H)$, with the constraint $\int_{\Omega_H} u_e = 0$, such that $\frac{\partial v}{\partial t}(\cdot,t;\boldsymbol{\mu}) \in L^2(\Omega_H)$, $\frac{\partial w}{\partial t}(\cdot,t;\boldsymbol{\mu}) \in L^2(\Omega_H)$ and

$$\begin{cases} A_m \int\limits_{\Omega_H} \Big(C_m \frac{\partial v}{\partial t} \ + \ I_{ion}(v,w;\boldsymbol{\mu}) \Big) \phi \ + \ \int\limits_{\Omega_H} \boldsymbol{D}_i(\boldsymbol{\mu}) \nabla v \ \cdot \ \nabla \phi \ + \\ \int\limits_{\Omega_H} \boldsymbol{D}_i(\boldsymbol{\mu}) \nabla u_e \cdot \nabla \phi = A_m \int\limits_{\Omega_H} I_{app}(\boldsymbol{\mu}) \phi \\ \int\limits_{\Omega_H} \boldsymbol{D}_i(\boldsymbol{\mu}) \nabla v \cdot \nabla \psi + \int\limits_{\Omega_H} (\boldsymbol{D}_i(\boldsymbol{\mu}) + \boldsymbol{D}_e(\boldsymbol{\mu})) \nabla u_e \cdot \nabla \psi = 0 \\ \int\limits_{\Omega_H} \frac{\partial w}{\partial t} \eta = \int\limits_{\Omega_H} g(v,w;\boldsymbol{\mu}) \ \eta \\ v(t_0;\boldsymbol{\mu}) = v_0(\boldsymbol{\mu}); \quad w(t_0;\boldsymbol{\mu}) = w_0(\boldsymbol{\mu}) \end{cases} \tag{4.38}$$

for all $(\phi, \psi, \eta) \in H^1(\Omega_H) \times H^1(\Omega_H) \times L^2(\Omega_H)$ with $\int_{\Omega_H} \psi = 0$.

As a matter of notation, we can define the following space:

$$H^1_{M_0}(\Omega_H) =: \left\{ \psi \in H^1(\Omega_H) : \int_{\Omega_H} \psi = 0 \right\} \tag{4.39}$$

which contains all the functions of $H^1(\Omega_H)$ that have zero mean in Ω_H; then $\psi \in H^1_{M_0}(\Omega_H)$. Notice that the constraint $\int_{\Omega_H} u_e = 0$ (which in turn implies the same one on the test function $\psi \in H^1(\Omega_H)$) is compulsory to ensure the well-posedness of (4.38); indeed the extracellular potential u_e appears in the equations only through its gradient, thus being known apart from a constant value. A detailed analysis of the well-posedness of the bidomain equations can be found, for instance, in [41], where a Faedo-Galerkin technique is used, and in [52], where only the coupling with the FHN ionic model is anyway considered.

The Torso Regarding the torso problem, the weak formulation of the generalized Laplace equation (4.36) with boundary conditions (4.37a) - (4.37b) reads as follows.

Problem 4.3: Generalized Laplace Equation Weak Formulation

Let $t > t_0$. Given $u_e^{\Gamma_H}(\cdot,t;\boldsymbol{\mu}) \in H^{\frac{1}{2}}(\Gamma_H)$, find $u_T(\cdot,t;\boldsymbol{\mu}) \in H^1(\Omega_T)$ such that $Tr\left(u_T(\cdot,t;\boldsymbol{\mu});\Gamma_H\right) = u_e^{\Gamma_H}(\cdot,t;\boldsymbol{\mu})$ and

$$\int\limits_{\Omega_T} \boldsymbol{D}_T(\boldsymbol{\mu}) \nabla u_T \cdot \nabla \phi_T = 0 \tag{4.40}$$

for all $\phi_T \in H^1_{\Gamma_H}(\Omega_T)$. $Tr(\ \cdot\ ;\Gamma_H) : H^1(\Omega_T) \to H^{\frac{1}{2}}(\Gamma_H)$ denotes the trace operator onto Γ_H.

Under the hypothesis that Γ_H is $\mathcal{C}^1$-continuous, it is possible to use the lifting operation to simplify the expression of (4.40) by defining $\tilde{u}_T =: u_T - u_e^{\Omega_T}$ being $u_e^{\Omega_T}(\cdot,t;\boldsymbol{\mu}) \in H^1(\Omega_T)\ \forall\ t > t_0$ such that $Tr\left(u_e^{\Omega_T};\Gamma_H\right) = u_e^{\Gamma_H} \in H^{\frac{1}{2}}(\Gamma_H)$. In this way, the weak formulation of the torso problem reads as follows.

Problem 4.4: Lifted Generalized Laplace Equation Weak Formulation

Let $t > t_0$. Given $u_e^{\Omega_T}(\cdot, t; \boldsymbol{\mu}) \in H^1(\Omega_T)$ such that $Tr\left(u_e^{\Omega_T}; \Gamma_H\right) = u_e^{\Gamma_H} \in H^{\frac{1}{2}}(\Gamma_H)$, find $\tilde{u}_T(\cdot, t; \boldsymbol{\mu}) \in H^1_{\Gamma_H}(\Omega_T)$ such that

$$\int_{\Omega_T} \boldsymbol{D}_T(\boldsymbol{\mu}) \nabla \tilde{u}_T \cdot \nabla \phi_T = -\int_{\Omega_T} \boldsymbol{D}_T(\boldsymbol{\mu}) \nabla u_e^{\Omega_T} \cdot \nabla \phi_T \tag{4.41}$$

for all $\phi_T \in H^1_{\Gamma_H}(\Omega_T)$. $Tr(\,\cdot\,; \Gamma_H) : H^1(\Omega_T) \to H^{\frac{1}{2}}(\Gamma_H)$ denotes the trace operator onto Γ_H.

Full Order Model: Galerkin Finite Element Approximation

We can now introduce the space and time discretizations of the problem at hand, starting from the weak formulations (4.38) for the bidomain equations on the heart and (4.41) for the torso problem. As before, the full heart-torso uncoupling allows us to handle the two problems separately.

The Heart Let us first consider the bidomain equations for the heart EP, whose weak formulation is given by (4.38). Let us consider three finite-dimensional spaces $X_h^{u_e} \subset H^1_{M_0}(\Omega_H)$, $X_h^v \subset H^1(\Omega_H)$, $X_h^w \subset L^2(\Omega_H)$; they have (usually large) dimensions $N_h^{u_e}, N_h^v$ and N_h^w respectively. Here h represents a parameter related to the mesh size of the computational grid. Let us denote $\{\psi_i\}_{i=1}^{N_h^{u_e}}$ a set of basis functions of the FE space $X_h^{u_e}$, $\{\phi_i\}_{i=1}^{N_h^v}$ a set of basis functions of the FE space X_h^v, and $\{\eta_i\}_{i=1}^{N_h^w}$ a set of basis functions of the FE space X_h^w. We can then express the three variables involved in the problem as:

$$u_{e_h}(\boldsymbol{x}, t; \boldsymbol{\mu}) =: \sum_{i=1}^{N_h^{u_e}} u_i(t; \boldsymbol{\mu}) \psi_i(\boldsymbol{x}) \qquad (\boldsymbol{x}, t) \in \Omega_H \times [t_0, T] \tag{4.42a}$$

$$v_h(\boldsymbol{x}, t; \boldsymbol{\mu}) =: \sum_{i=1}^{N_h^v} v_i(t; \boldsymbol{\mu}) \phi_i(\boldsymbol{x}) \qquad (\boldsymbol{x}, t) \in \Omega_H \times [t_0, T] \tag{4.42b}$$

$$w_h(\boldsymbol{x}, t; \boldsymbol{\mu}) =: \sum_{i=1}^{N_h^w} w_i(t; \boldsymbol{\mu}) \eta_i(\boldsymbol{x}) \qquad (\boldsymbol{x}, t) \in \Omega_H \times [t_0, T] \tag{4.42c}$$

where the three vectors

$$\begin{aligned}
\boldsymbol{u}_{e_h}(t; \boldsymbol{\mu}) &= [u_{e_1}(t; \boldsymbol{\mu}), u_{e_2}(t; \boldsymbol{\mu}), \ldots, u_{e_{N_h^{u_e}}}(t; \boldsymbol{\mu})]^T \\
\boldsymbol{v}_h(t; \boldsymbol{\mu}) &= [v_1(t; \boldsymbol{\mu}), v_2(t; \boldsymbol{\mu}), \ldots, v_{N_h^v}(t; \boldsymbol{\mu})]^T \\
\boldsymbol{w}_h(t; \boldsymbol{\mu}) &= [w_1(t; \boldsymbol{\mu}), w_2(t; \boldsymbol{\mu}), \ldots, w_{N_h^w}(t; \boldsymbol{\mu})]^T
\end{aligned}$$

are obtained by solving the following discrete system:

Problem 4.5: Bidomain Equations + Ionic Model Discrete Formulation

Given $\boldsymbol{\mu} \in \mathcal{P} \subset \mathbb{R}^p$, find $\boldsymbol{u}_{e_h} = \boldsymbol{u}_{e_h}(t;\boldsymbol{\mu})$, $\boldsymbol{v}_h = \boldsymbol{v}_h(t;\boldsymbol{\mu})$ and $\boldsymbol{w}_h = \boldsymbol{w}_h(t;\boldsymbol{\mu})$ such that:

$$\begin{cases} A_m C_m \boldsymbol{M} \dfrac{\partial \boldsymbol{v}_h}{\partial t} + \boldsymbol{A}_{in}^{1,1}(\boldsymbol{\mu})\boldsymbol{v}_h + \boldsymbol{A}_{in}^{1,2}(\boldsymbol{\mu})\boldsymbol{u}_{e_h} + & t \in [t_0, T] \\ A_m \boldsymbol{I}_{ion}(\boldsymbol{v}_h, \boldsymbol{w}_h; \boldsymbol{\mu}) = A_m \boldsymbol{I}_{app}(t;\boldsymbol{\mu}) & \\ \boldsymbol{A}_{in}^{2,1}(\boldsymbol{\mu})\boldsymbol{v}_h + \big(\boldsymbol{A}_{in}^{2,2}(\boldsymbol{\mu}) + \boldsymbol{A}_{ex}(\boldsymbol{\mu})\big)\boldsymbol{u}_{e_h} = 0 & \\ \dfrac{\partial \boldsymbol{w}_h}{\partial t} = g(\boldsymbol{v}_h, \boldsymbol{w}_h; \boldsymbol{\mu}) & t \in [t_0, T] \\ \boldsymbol{v}_h(t_0;\boldsymbol{\mu}) = \boldsymbol{v}_0(\boldsymbol{\mu}); \qquad \boldsymbol{w}_h(t_0;\boldsymbol{\mu}) = \boldsymbol{w}_0(\boldsymbol{\mu}) & \end{cases} \tag{4.43}$$

where we denote the stiffness matrices and the mass matrix as:

$$\big(\boldsymbol{M}\big)_{ij} = \int_{\Omega_H} \phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.44a}$$

$$\big(\boldsymbol{A}_{in}^{1,1}\big)_{ij} = \int_{\Omega_H} \boldsymbol{D}_i(\boldsymbol{x};\boldsymbol{\mu})\nabla\phi_i(\boldsymbol{x}) \cdot \nabla\phi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.44b}$$

$$\big(\boldsymbol{A}_{in}^{1,2}\big)_{ij} = \int_{\Omega_H} \boldsymbol{D}_i(\boldsymbol{x};\boldsymbol{\mu})\nabla\psi_i(\boldsymbol{x}) \cdot \nabla\phi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.44c}$$

$$\big(\boldsymbol{A}_{in}^{2,1}\big)_{ij} = \int_{\Omega_H} \boldsymbol{D}_i(\boldsymbol{x};\boldsymbol{\mu})\nabla\phi_i(\boldsymbol{x}) \cdot \nabla\psi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.44d}$$

$$\big(\boldsymbol{A}_{in}^{2,2}\big)_{ij} = \int_{\Omega_H} \boldsymbol{D}_i(\boldsymbol{x};\boldsymbol{\mu})\nabla\psi_i(\boldsymbol{x}) \cdot \nabla\psi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.44e}$$

$$\big(\boldsymbol{A}_{ex}\big)_{ij} = \int_{\Omega_H} \boldsymbol{D}_e(\boldsymbol{x};\boldsymbol{\mu})\nabla\psi_i(\boldsymbol{x}) \cdot \nabla\psi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.44f}$$

Also, we define:

$$\big(\boldsymbol{I}_{app}(t;\boldsymbol{\mu})\big)_j = \int_{\Omega_H} I_{app}(\boldsymbol{x}, t;\boldsymbol{\mu})\phi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.45a}$$

$$\big(\boldsymbol{I}_{ion}(\boldsymbol{v}_h, \boldsymbol{w}_h;\boldsymbol{\mu})\big)_j = \int_{\Omega_H} I_{ion}(v_h, w_h;\boldsymbol{\mu})\phi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.45b}$$

$$\big(\boldsymbol{g}(\boldsymbol{v}_h, \boldsymbol{w}_h;\boldsymbol{\mu})\big)_j = \int_{\Omega_H} g(v_h, w_h;\boldsymbol{\mu})\eta_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.45c}$$

In order to simplify both the formulation of the problem and the computation of its solution, we resorted to a formulation that is simpler than (4.43) and that has been derived under the assumption that all the three unknowns involved in the problem (i.e. u_e, v and w) belong to the same subspace $H^1(\Omega_H)$. As it is, such a formulation cannot be well-posed, since we have removed the zero-average constraint on u_e; to recover it, we acted at post-processing stage, where the computed value of the epicardial extracellular potential has been rescaled to ensure that $\int_{\Omega_H} u_e = 0$. Such a numerical procedure may easily bring to instability while solving the resulting linear system; issues can be anyway prevented by employing stabilized iterative solvers, as the bi-conjugate gradient stabilized (BiCGSTAB) method (i.e. a variant of the more classical bi-conjugate gradient (BiCG) method, introduced by *Van der Vorst* in [53] and showing much faster and smoother convergence properties).

So, under the last hypothesis, we can define a unique finite-dimensional subspace $X_h \subset H^1(\Omega_H)$ of dimension N_h and denote as $\{\phi_i\}_{i=1}^{N_h}$ the set of its FE basis functions. The unknowns involved in the problem can be then all expressed as in (4.42a)-(4.42b)-(4.42c), but always using $\{\phi_i\}_{i=1}^{N_h}$ as basis functions. Then the discrete-in-space, continuous-in-time system to be solved reads as follows.

Problem 4.6: Bidomain Equations + Ionic Model Discrete Formulation 2

Given $\boldsymbol{\mu} \in \mathcal{P} \subset \mathbb{R}^p$, find $\boldsymbol{u}_{e_h} = \boldsymbol{u}_{e_h}(t;\boldsymbol{\mu})$, $\boldsymbol{v}_h = \boldsymbol{v}_h(t;\boldsymbol{\mu})$ and $\boldsymbol{w}_h = \boldsymbol{w}_h(t;\boldsymbol{\mu})$ such that:

$$\begin{cases} A_m C_m \boldsymbol{M}\dfrac{\partial \boldsymbol{v}_h}{\partial t} + \boldsymbol{A}_{in}(\boldsymbol{\mu})\boldsymbol{v}_h + \boldsymbol{A}_{in}(\boldsymbol{\mu})\boldsymbol{u}_{e_h} + & t \in [t_0, T] \\ A_m \boldsymbol{I}_{ion}(\boldsymbol{v}_h, \boldsymbol{w}_h; \boldsymbol{\mu}) = A_m \boldsymbol{I}_{app}(t;\boldsymbol{\mu}) & \\ \boldsymbol{A}_{in}(\boldsymbol{\mu})\boldsymbol{v}_h + \big(\boldsymbol{A}_{in}(\boldsymbol{\mu}) + \boldsymbol{A}_{ex}(\boldsymbol{\mu})\big)\boldsymbol{u}_{e_h} = 0 & \\ \dfrac{\partial \boldsymbol{w}_h}{\partial t} = g(\boldsymbol{v}_h, \boldsymbol{w}_h; \boldsymbol{\mu}) & t \in [t_0, T] \\ \boldsymbol{v}_h(t_0;\boldsymbol{\mu}) = \boldsymbol{v}_0(\boldsymbol{\mu}); \qquad \boldsymbol{w}_h(t_0;\boldsymbol{\mu}) = \boldsymbol{w}_0(\boldsymbol{\mu}) & \end{cases} \tag{4.46}$$

where the mass matrix $\boldsymbol{M}$ and the applied and ionic currents $\boldsymbol{I}_{app}$, $\boldsymbol{I}_{ion}$ are defined as in (4.44a) - (4.45a) - (4.45b) respectively, whereas the stiffness matrices are defined as

$$\big(\boldsymbol{A}_{in}(\boldsymbol{\mu})\big)_{ij} = \int_{\Omega_H} \boldsymbol{D}_i(\boldsymbol{\mu})\phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.47a}$$

$$\big(\boldsymbol{A}_{ex}(\boldsymbol{\mu})\big)_{ij} = \int_{\Omega_H} \boldsymbol{D}_e(\boldsymbol{\mu})\phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.47b}$$

The discretization of the non-linear term involved in the ionic model is

$$\big(\boldsymbol{g}(\boldsymbol{v}_h, \boldsymbol{w}_h; \boldsymbol{\mu})\big)_j = \int_{\Omega_H} g(v_h, w_h; \boldsymbol{\mu})\phi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.48}$$

Additionally, taking advantage of the *axial isotropy assumption* (see (4.32)), both stiffness matrices $\boldsymbol{A}_{in}$ and $\boldsymbol{A}_{ex}$ can be decomposed as:

$$\boldsymbol{A}_{in,ex}(\boldsymbol{\mu}) = \sigma_{t_{i,e}} \boldsymbol{A}^{iso}_{in,ex}(\boldsymbol{\mu}) + (\sigma_{l_{i,e}} - \sigma_{t_{i,e}})\boldsymbol{A}^{fibers}_{in,ex}(\boldsymbol{\mu}) \tag{4.49}$$

being

$$\big(\boldsymbol{A}^{iso}_{in,ex}\big)_{ij} =: \int_{\Omega_H} \nabla\phi_i(\boldsymbol{x}) \cdot \nabla\phi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.50a}$$

$$\big(\boldsymbol{A}^{fibers}_{in,ex}\big)_{ij} =: \int_{\Omega_H} \boldsymbol{a}_l(\boldsymbol{x})\boldsymbol{a}_l(\boldsymbol{x})^T \nabla\phi_i(\boldsymbol{x}) \cdot \nabla\phi_j(\boldsymbol{x})\, d\boldsymbol{x} \tag{4.50b}$$

Regarding the treatment of the non-linear terms and the time discretization, we employed a semi-implicit, first order, one-step time marching scheme; a detailed reference can be found in [49]. Suppose to partition the time interval $[t_0; T]$ into N_t small sub-intervals $\{[t^{(l)}; t^{(l+1)}]\}_{l=0}^{N_t-1}$ such that $t^{(0)} = t_0$ and $t^{(N_t)} = T$; suppose also the intervals to be all of the same length, equal to Δt, and small enough to capture the very fast dynamics of the electric impulse front propagation. The solving process is then divided into three phases:

1. **Solving the ODE associated to the ionic model**: the first step amounts at computing the solution to the ODE that models the behavior of the gating variable w, responsible for cells refractoriness in the phenomenological ionic model. To do so, a simple implicit (backward) Euler method is employed; the transmembrane potential involved in the expression of the right-hand side $g(\cdot,\cdot;\boldsymbol{\mu})$ can be either evaluated at the previous time instant $t^{(l)}$ or extrapolated from its values at some previous k time instants $\{t^{(m)}\}_{m=l-k}^{l}$. Whichever strategy is used for the estimation of the value of $\boldsymbol{v}_h$ at time $t^{(l+1)}$, defining such value as $\tilde{\boldsymbol{v}}_h^{(l+1)}$, the value of $\boldsymbol{w}_h^{(l+1)}$ is computed by solving:

$$\frac{\boldsymbol{w}_h^{(l+1)} - \boldsymbol{w}_h^{(l)}}{\Delta t} = \boldsymbol{g}(\tilde{\boldsymbol{v}}_h^{(l+1)}, \boldsymbol{w}_h^{(l+1)}; \boldsymbol{\mu}) \tag{4.51}$$

   Being $\boldsymbol{g}$ non-linear in its second argument, fixed point methods have to be employed.

2. **Evaluating the ionic current term**: once $\boldsymbol{w}_h^{(l+1)}$ is known, the ionic current term $\boldsymbol{I}_{ion}^{(l+1)}$ can be evaluated at time $t^{(l+1)}$, performing the integration reported in (4.45b). Also in this case, an approximated value of $\boldsymbol{v}_h^{(l+1)}$, denoted as $\tilde{\boldsymbol{v}}_h^{(l+1)}$, is used.
3. **Solving the PDEs associated to the bidomain model**: as last step, once $\boldsymbol{w}_h^{(l+1)}$ is known, the PDEs associated to the bidomain model can be discretized in time, using the implicit Euler method, and solved, so that both $\boldsymbol{v}_h^{(l+1)}$ and $\boldsymbol{u}_{e_h}^{(l+1)}$ can be computed. Thus, the discrete system of equations reads as:

$$\begin{cases} A_m C_m \boldsymbol{M} \dfrac{\boldsymbol{v}_h^{(l+1)} - \boldsymbol{v}_h^{(l)}}{\Delta t} + \boldsymbol{A}_{in}(\boldsymbol{\mu}) \boldsymbol{v}_h^{(l+1)} + \boldsymbol{A}_{in}(\boldsymbol{\mu}) \boldsymbol{u}_{e_h}^{(l+1)} = \\ A_m \big(\boldsymbol{I}_{app}^{(l+1)}(\boldsymbol{\mu}) - \boldsymbol{I}_{ion}^{(l+1)}(\tilde{\boldsymbol{v}}_h^{(l+1)}, \boldsymbol{w}_h^{(l+1)}; \boldsymbol{\mu}) \big) \\ \boldsymbol{A}_{in}(\boldsymbol{\mu}) \boldsymbol{v}_h^{(l+1)} + \big(\boldsymbol{A}_{in}(\boldsymbol{\mu}) + \boldsymbol{A}_{ex}(\boldsymbol{\mu}) \big) \boldsymbol{u}_{e_h}^{(l+1)} = 0 \end{cases} \tag{4.52}$$

Ultimately, written in matricial form, the full-order bidomain equations read as:

$$\left(\frac{A_m C_m}{\Delta t} \begin{bmatrix} \boldsymbol{M} & \boldsymbol{0}_{N_h \times N_h} \\ \boldsymbol{0}_{N_h \times N_h} & \boldsymbol{0}_{N_h \times N_h} \end{bmatrix} + \begin{bmatrix} \boldsymbol{A}_{in}(\boldsymbol{\mu}) & \boldsymbol{A}_{in}(\boldsymbol{\mu}) \\ \boldsymbol{A}_{in}(\boldsymbol{\mu}) & \boldsymbol{A}_{in}(\boldsymbol{\mu}) + \boldsymbol{A}_{ex}(\boldsymbol{\mu}) \end{bmatrix} \right) \begin{bmatrix} \boldsymbol{v}_h^{(l+1)}) \\ \boldsymbol{u}_{e_h}^{(l+1)} \end{bmatrix} = \begin{bmatrix} A_m \Big(\boldsymbol{I}_{app}^{(l+1)} - \boldsymbol{I}_{ion}^{(l+1)} + \dfrac{C_m}{\Delta t} \boldsymbol{M} \boldsymbol{v}_h^{(l)} \Big), \boldsymbol{0}_{N_h} \end{bmatrix}^T \tag{4.53}$$

This first-order semi-implicit approach leads to the separate solution of one non-linear ODE (via fixed points iterations) and of one linear system (via the Preconditioned BiCGSTAB method) at each timestep. Additional improvements, like the usage of adaptive time stepping strategies or possible parallel linear solver implementations, can be found in high detail in [49].

The Torso We can now consider the problem of the propagation of the electric signal from the epicardium to the body surface; its weak formulation is given by (4.41). Let us consider a finite-dimensional space $X_h^t \subset H_0^1(\Omega_T)$ of dimension N_h^t. Let us denote $\{\phi_i^t\}_{i=1}^{N_h^t}$ a set of basis functions of the FE space X_h^t. The torso potential u_t can be then approximated as:

$$u_{t_h}(\boldsymbol{x}, t; \boldsymbol{\mu}) =: \sum_{i=1}^{N_h^t} u_{t_i}(t; \boldsymbol{\mu}) \phi_i^t(\boldsymbol{x}) \qquad (\boldsymbol{x}, t) \in \Omega_T \times [t_0, T] \tag{4.54}$$

where the vector

$$\boldsymbol{u}_{t_h}(t; \boldsymbol{\mu}) = [u_{t_1}(t; \boldsymbol{\mu}), u_{t_2}(t; \boldsymbol{\mu}), \dots, u_{t_{N_h^e}}(t; \boldsymbol{\mu})]^T$$

is computed by solving the following problem:

Problem 4.7: Lifted Generalized Laplace Equation Discrete Formulation

Given $\boldsymbol{\mu} \in \mathcal{P} \subset \mathbb{R}^p$, find $\boldsymbol{u}_{t_h} = \boldsymbol{u}_{t_h}(t; \boldsymbol{\mu})$ such that:

$$\boldsymbol{A}^t(\boldsymbol{\mu}) \boldsymbol{u}_{t_h}(t) = -\boldsymbol{A}^t(\boldsymbol{\mu}) \boldsymbol{u}_{e_h}^{\Omega_T}(t) \qquad t \in [t_0, T] \tag{4.55}$$

where

$$\big(\boldsymbol{A}^t(\boldsymbol{\mu}) \big)_{ij} = \int_{\Omega_T} \boldsymbol{D}_T(\boldsymbol{x}; \boldsymbol{\mu}) \nabla \phi_i^t(\boldsymbol{x}) \cdot \nabla \phi_j^t(\boldsymbol{x}) \, d\boldsymbol{x} \tag{4.56}$$

$$\big(\boldsymbol{u}_{e_h}^{\Omega_T}(t) \big)_j = \int_{\Omega_T} u_e^{\Omega_T}(\boldsymbol{x}, t) \phi_j^t(\boldsymbol{x}) \, d\boldsymbol{x} \qquad t \in [t_0, T] \tag{4.57}$$

Furthermore, Problem 4.7 can be simplified by evaluating the right-hand side term only at the DOFs involved in the Dirichlet boundary condition at Γ_H, i.e.

$$\boldsymbol{A}^t(\boldsymbol{\mu}) \boldsymbol{u}_{t_h}(t) = -\boldsymbol{A}_{\Gamma_H}^t(\boldsymbol{\mu}) \boldsymbol{u}_{e_h}^{\Gamma_H}(t) \qquad t \in [t_0, T] \tag{4.58}$$

where $\boldsymbol{A}^t_{\Gamma_H}(\boldsymbol{\mu}) =: \boldsymbol{A}^t(\boldsymbol{\mu})[\cdot, \{j_{Dir}\}]$, being $\{j_{Dir}\}$ the set of DOFs at which the Dirichlet boundary condition has been imposed.

Notice that Problem 4.7 is a steady problem; thus it has to be solved, independently, at all the discrete time instances $\{t^{(l)}\}_{l=1}^{N_t}$ at which the heart EP has been solved (via (4.52)) and at which, thus, the approximated epicardial extracellular potential $\boldsymbol{u}_{e_h}|_{\Gamma_H}$ acting as Dirichlet boundary condition is available. So, leveraging the time domain discretization employed to compute the numerical solution of the bidomain equations, it is possible to derive the space-time discretized torso potential

$$\left\{\boldsymbol{u}^{(l)}_{t_h}\right\}_{l=1}^{N_t} =: \left\{\boldsymbol{u}_{t_h}(t^{(l)})\right\}_{l=1}^{N_t} \tag{4.59}$$

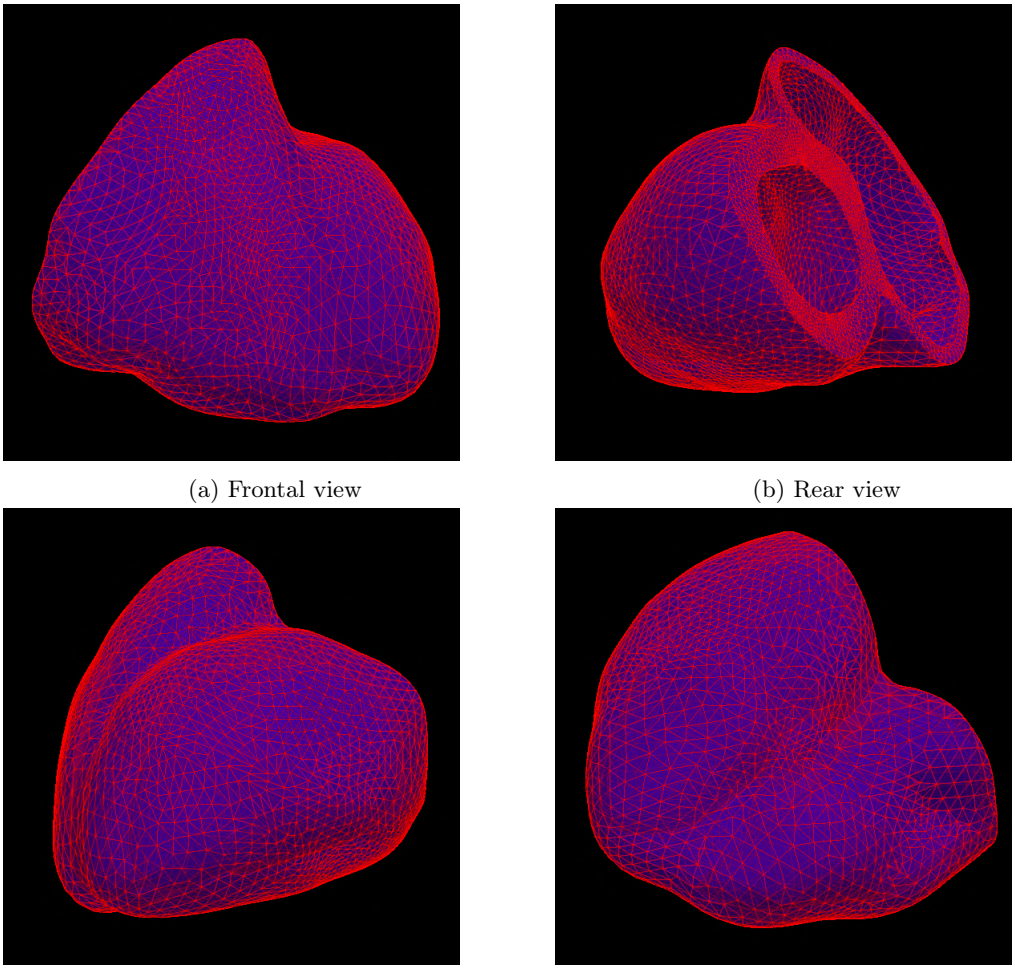
4.2.3 Numerical Results

This Subsection is devoted to a brief presentation of the numerical results obtained by solving the heart bidomain equations and the problem of the transmission of the signal to the torso using the Galerkin FE method (see (4.46) - (4.55)). Before moving to that, it is necessary to provide a description of the geometries and of the computational meshes that have been employed and of the values of the most relevant parameters and data that have been used. This will be done in the next two paragraphs.

Geometries and Computational Meshes

In the context of this project, the FOM problem has to be solved many times, for many different values of the model parameters and data, so that a sufficiently large and variable dataset can be generated and, on top of that, a DL model can be successfully trained. Because of this, we have chosen to run all our simulations using meshes made of a limited number of elements; as a drawback, the results we got cannot be considered accurate and able to precisely reproduce what actually happens in reality. Anyway, the core aim of the present work, as already discussed in Chapter 1, is not to build up a DL model able to effectively handle any kind of real ECG signal, but more to identify a PDE-aware DL model architecture that proves to be able to well estimate the mapping between ECG signals (or, more in general, body surface signals) and epicardial potentials. Thus, a precise reproduction of real ECG signals (that proves to be a very complex task, as discussed for instance by *Boulakia et al.* in [50]) is not actually desired, while our interest is mainly oriented towards the possibility of simulating, in a physiologically and phenomenologically consistent way, but yet with a limited consumption of computational resources, the heart EP and the transmission of the signal to the body surface. Once a proper DL architecture, able to fulfill all our requirements, is found, our *ansatz* (and hope) is that the same architecture, trained with either real data or with more realistic "artificial" ones, will be also able to provide a good mapping between real ECG signals and the corresponding epicardial potentials.

The mesh used to numerically approximate the heart EP via the bidomain equations coupled with the AP ionic model is reported in Figure 4.8 from four different views (frontal, rear, left lateral and inferior). It is made of $27'636$ tetrahedral cells, which result in 7718 vertices; this is a quite coarse mesh, especially in the context of heart EP, whose simulation necessitates of a much higher degree of refinement ($\approx 100'000$ nodes) to be able to precisely capture the fast dynamics of the depolarization wavefront. The geometry only considers the two ventricles (heart biventricle model), excluding the atria; such an assumption is reasonable in the context of electrocardiography, since the contribution of the atria becomes negligible once the depolarization phase of the ventricles is started. In terms of ECG signals, neglecting the atria we cannot be able to visualize the P wave (being it linked to atrial depolarization, see Subsection 4.1.2), but all the other characteristic waves should be almost unaffected.
The mesh that has been used to solve Problem 4.7 is made of $73'607$ tetrahedral elements and of $16'793$ vertices; its underlying geometry represents an idealized human torso, consisting of a cylinder with ellipsoidal basis built around the heart and it is represented in Figure 4.9. In this setting, the heart acts as a boundary condition, so only its epicardium is considered in



(a) Frontal view

(b) Rear view

(c) Left Lateral view

(d) Inferior view

Figure 4.8: Computational mesh employed to numerically approximate the heart EP from four different points of view

the geometry; additionally no design of the different organs of the human body (such as bones or lungs) is made, so that the torso is assumed to be a perfectly isotropic and homogeneous conductor. Differently from the heart geometry of Figure 4.8, the heart present in the torso geometry does not only feature the epicardium of the two ventricles, but also the one of the two atria; this may be problematic from the computational point of view, since the behavior of the depolarization wavefront at the atria appears to be relevant while studying the propagation of the signal through the torso. In order to minimize such contribution and thus to allow the torso potential to be determined just by the ventricular epicardial one, we have decided to impose a homogeneous Neumann boundary condition at the atria, thus rewriting problem (4.36) - (4.37a) - (4.37b) as:

$$\begin{cases} -div\left(\boldsymbol{D}_T(\boldsymbol{\mu})\nabla u_T\right) = 0 & in\ \Omega_T \\ \boldsymbol{n}_B \cdot \boldsymbol{D}_T(\boldsymbol{\mu})\nabla u_T = 0 & on\ \Gamma_B \\ \boldsymbol{n}_{H_A} \cdot \boldsymbol{D}_T(\boldsymbol{\mu})\nabla u_T = 0 & on\ \Gamma_{H_A} \\ u_T = u_e|_{\Gamma_{H_V}} & on\ \Gamma_{H_V} \end{cases} \tag{4.60}$$

being Γ_{H_A} the atrial epicardial surface and Γ_{H_V} the ventricular epicardial surface, so that $\Gamma_{H_A} \cap \Gamma_{H_V} = \emptyset$ and $\Gamma_{H_A} \cup \Gamma_{H_V} = \Gamma_H$, and $\boldsymbol{n}_{H_A}$ the outward unit normal vector to Γ_{H_A}. Other attempts have been made (for instance the imposition of a homogeneous Dirichlet boundary condition, where homogeneity aims at preserving the zero-average constraint already satisfied by the ventricular extracellular potential u_e), but they all ended up giving rise to unrealistic

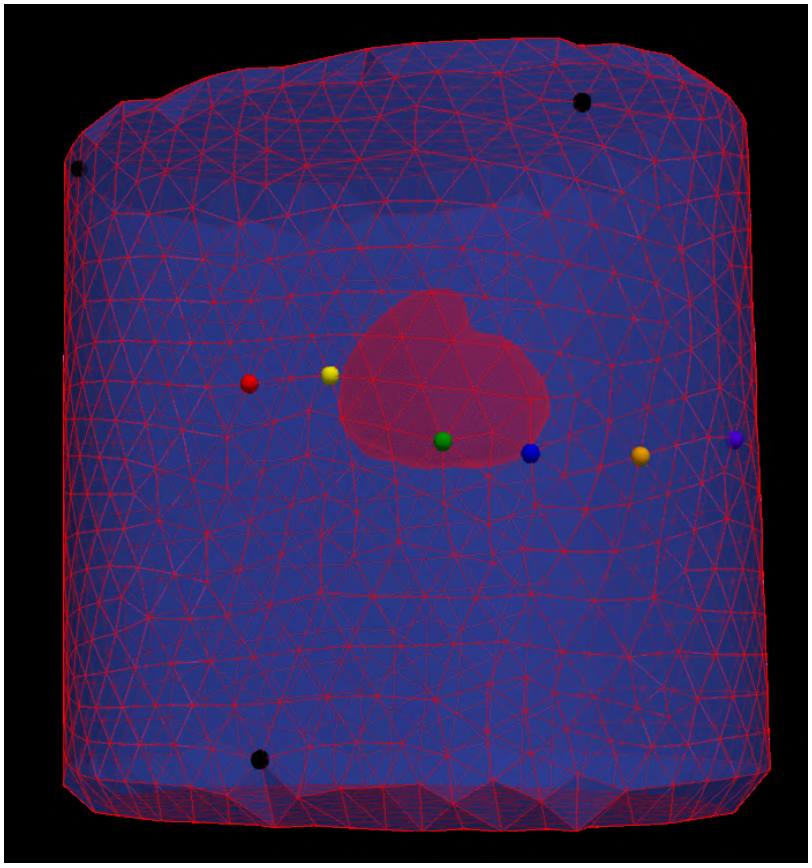


Figure 4.9: Computational Mesh used to numerically approximate the transmission of the signal from the epicardium to the body surface. The geometry represents an idealized torso, constructed as an ellipsoidal cylinder built around the heart. The spheres identify the positions of the 9 ECG electrodes; the three limb electrodes are marked in Black, while the six chest ones follow the *AHA (American Heart Association)* color coding system

waves in the initial part of the ECG traces; the choice of imposing homogeneous Neumann boundary conditions at the atria has been so identified as the best one from a qualitative point of view. Finally, since this geometry only represents the human torso, without arms and legs, it has not been possible to place the three limb electrodes in their standard positions; as it can be seen in Figure 4.9, the electrodes LA and RA have been placed on the left and right shoulder respectively and the electrode LL has been placed in a position that can be thought as being located below the navel. Anyway, as said in Subsection 4.1.2, these electrodes can be also placed closer to the hips/shoulders, as long as their disposition is symmetric and that it allows to build a triangle having its center in the middle of the thorax. Thus we do not expect the results to dramatically change, in case arms and legs are involved in the geometry and the electrodes are moved to their more classical locations.

Model Parameters and Data

All the numerical results presented in this Subsection, as well as the ones that will be shown in Subsection 4.2.4, have been obtained by solving the bidomain equations, coupled with the AP ionic model, and the torso transmission problem (see Problem 4.1). The solutions to this system of PDEs - ODEs show a significant dependency on the model parameters (conducibilities, parameters of the ionic model, etc.) and data (initial and boundary conditions, externally applied current, etc.); thus, before moving to a brief discussion of the results, it is worth reporting the choices that have been made in this sense.

The values of the employed model parameters, that we have been synthetically embodied in the

$\mathbf{K}$	$\mathbf{a}$	$\epsilon_\mathbf{0}$	$\mu_\mathbf{1}$	$\mu_\mathbf{2}$	$\mathbf{b}$	$\mathbf{V_{min}}$	$\mathbf{V_{max}}$
8	$1.5 \cdot 10^{-1}$	$6.5 \cdot 10^{-3}$	$1.0 \cdot 10^{-1}$	$3.0 \cdot 10^{-1}$	$1.5 \cdot 10^{-1}$	$-80\ mV$	$20\ mV$

Table 4.2: Values of the parameters used in the AP ionic model

$\mathbf{A_m}[cm^{-1}]$	$\mathbf{C_m}[\mu F]$	$\sigma_{\mathbf{l_i}}[S\ cm^{-1}]$	$\sigma_{\mathbf{t_i}}[S\ cm^{-1}]$	$\sigma_{\mathbf{l_e}}[S\ cm^{-1}]$	$\sigma_{\mathbf{t_e}}[S\ cm^{-1}]$
$1.40 \cdot 10^3$	$5.0 \cdot 10^{-1}$	$6.0 \cdot 10^{-4}$	$9.0 \cdot 10^{-5}$	$2.25 \cdot 10^{-3}$	$2.25 \cdot 10^{-4}$

Table 4.3: Values of other parameters used in the numerical approximation of the bidomain equations

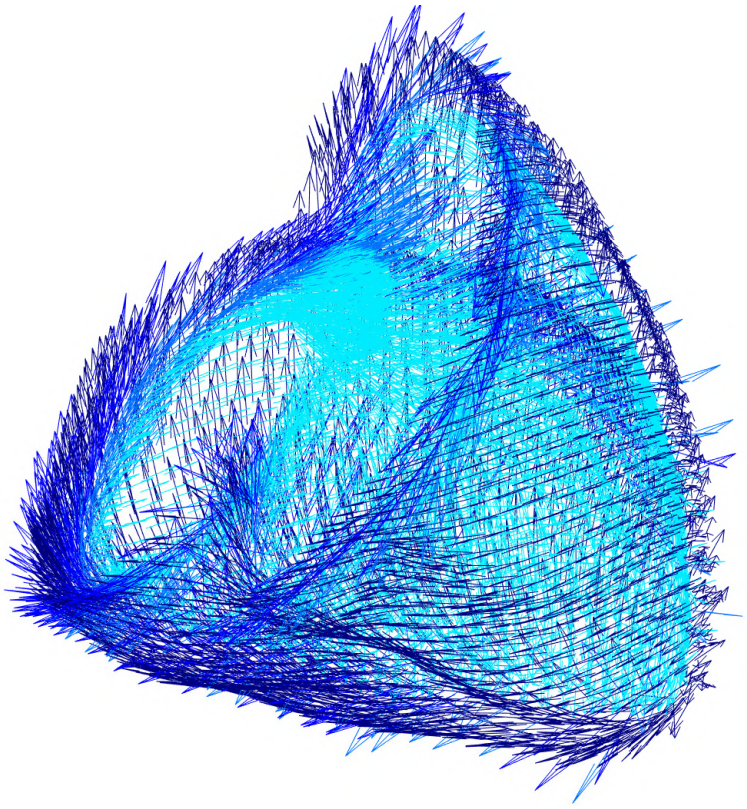


Figure 4.10: Structure of the heart conduction fibers

parameters vector $\boldsymbol{\mu} \in \mathcal{P} \subset \mathbb{R}^p$, are reported in Tables 4.2 (relative to the AP model parameters only) and 4.3. Regarding the AP model, standard parameter values that can be retrieved in literature have been adopted; concerning the other parameter values, instead, A_m and C_m have been set to standard values, while the heart conducibilities $\sigma_{l,t_{i,e}}$ have been chosen so that the resulting ECG signals show waves durations and amplitudes as physiological as possible. The structure of the heart conduction fibers is also reported in Figure 4.10.

Regarding the initial conditions, again standard ones have been imposed, so that

$$v(\cdot, t_0) = 0 \qquad w(\cdot, t_0) = 0 \tag{4.61}$$

Recall that the transmembrane potential v is assumed to be normalized in $[0; 1]$ by the AP ionic model, thus setting it to 0 amounts at assuming a fully inactivated heart at the initial time instant. Also, $t_0 = 0$, while the simulation duration T has been set to 160 ms, so that only the ventricular depolarization process is considered.

Finally, the externally applied current has been designed in such a way that the positions and

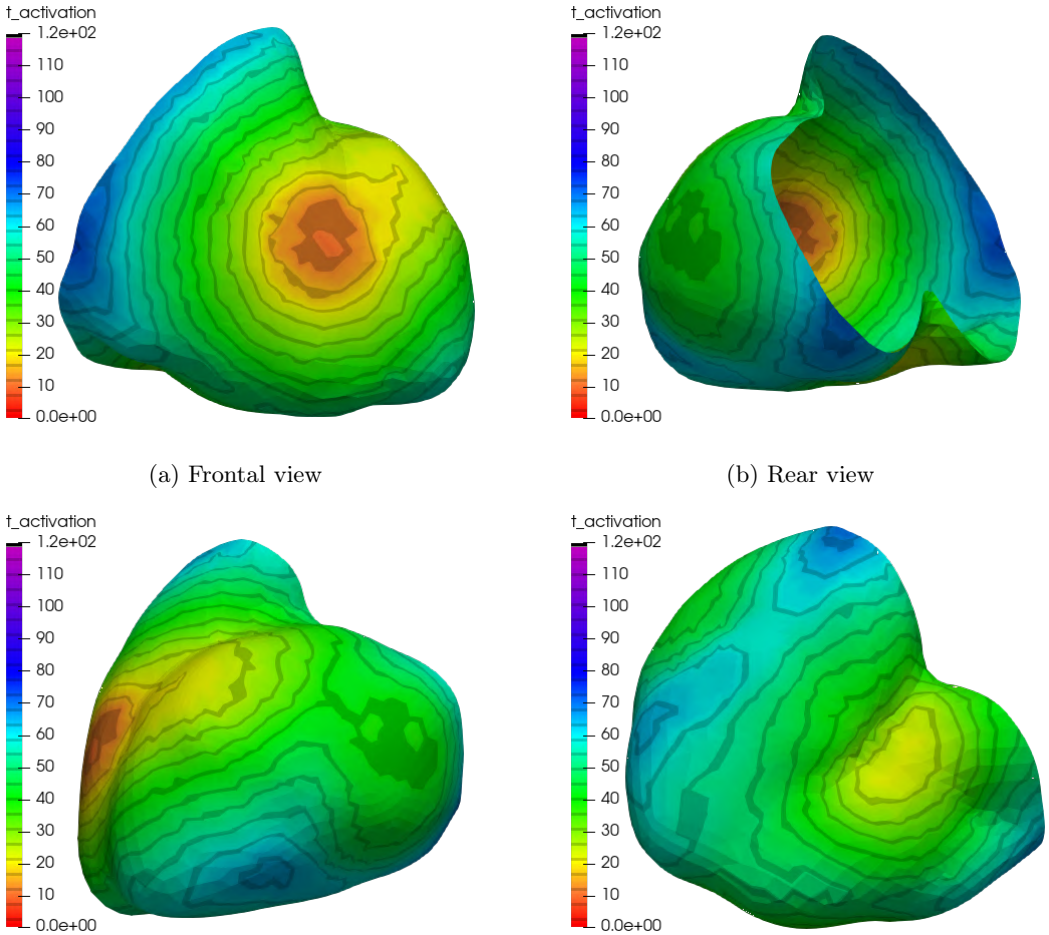
timings of the EBTs are in accordance with the findings of *Wyndham et al.* in [39], already described in a dedicated paragraph of Subsection 4.1.1. In particular, during the first 45 ms of the simulation, different areas of the endocardial and sub-endocardial layers have been excited, so as to reproduce a physiologically consistent EBT pattern; this has been done considering both healthy cases (following [39]) and cases of patients affected by either Left Bundle Branch Block (LBBB) (following [54]) or Right Bundle Branch Block (RBBB) (with an educated guess on EBT locations and timings, based on the findings of the previously cited articles).

Results

In the following we present the ECG signals and the epicardial activation maps obtained simulating the heart EP and the torso signal transmission in a healthy case and the case of a patient affected by LBBB.

The epicardial activation maps and the ECG signals obtained by simulating an **healthy case** can be found in Figures 4.11 and 4.12 respectively. Regarding the activation map, it is easy to recognize the two earliest EBTs on the anterior part of the right ventricle (around 10 ms) and of the left ventricle (around 25 ms). Also, later EBTs can be seen in the inferior part of the right ventricle, close to the base and paraseptally, around 30 ms and on the left ventricle free wall around 40 ms. The site of LEA is located at the base of the heart, anteriorly, near the acute margin of the right ventricle; it has been activated ≈ 75 ms after the QRS onset. All these observations are in accordance with the findings of *Wyndham et al.* in [39]. Concerning the ECG signals, instead, an overall qualitative similarity with the target healthy ECG described in Table 4.1 can be found; notice that, having simulated only the ventricular depolarization, only the QRS complex is visualized. For instance the S and R waves progression in the precordial leads is almost as expected (actually either in lead V_3 or in lead V_4 we should be able to observe both a R and a S wave, with comparable amplitudes, while here only a R wave can be observed; also the amplitude of the R wave in V_6 should be smaller than the one in V_5); the polarities of all the leads match our expectations; the duration of the QRS complex (≈ 75 ms) is short, but it falls in the physiological range. Readily, the simplifying assumptions both at modelistic level and concerning the geometries and the meshes prevent us from reconstructing realistic heart depolarization cycles and ECG signals; anyway, we managed to obtain physiologically and phenomenologically consistent results with a reduced computational effort (the whole simulation runs in ≈ 140 s on a laptop - for the specifics see Appendix C), which was our target.

The epicardial activation maps and the ECG signals obtained by simulating a **LBBB case** can be found in Figures 4.13 and 4.14 respectively. For what concerns the activation map, also in this case the results are in accordance with the findings of *Wyndham et al.* in [54]; in particular 2 EBTs can be visualized in the anterior paraseptal (at ≈ 15 ms) and inferior basal paraseptal (at ≈ 25 ms) right ventricle, while the depolarizing wavefront must pass through the septum from right to left to reach the left ventricle, whose conduction bundle is inactivated. The site of LEA is located on the left ventricle and more specifically in its basal lateral region; the signal reaches this area ≈ 110 ms after the QRS onset. Regarding the ECG signals, their similarity with real ones of patients affected by LBBB is lower compared to the one got in the healthy case; in particular, as reported for instance in [40], the signals of leads $V_1 - V_2 - V_3$ should show deep and steep Q waves (here they are not so deep and steep) and the duration of the QRS complex should exceed 120 ms. These major issues could be imputed, other than to the modelistic simplifications and to the coarseness of the employed computational meshes, also to a not so precise and reliable representation of the way the depolarization wavefront moves from the right ventricle to the left one if the left bundle branch is blocked. This is clearly a limitation of our simulation setting and further efforts can be made in this sense, for instance referring to the results presented by *Boulakia et al.* in [50].



(a) Frontal view

(b) Rear view

(c) Left Lateral view

(d) Inferior view

Figure 4.11: Epicardial activation maps obtained by simulating a healthy heart depolarization process from four different views

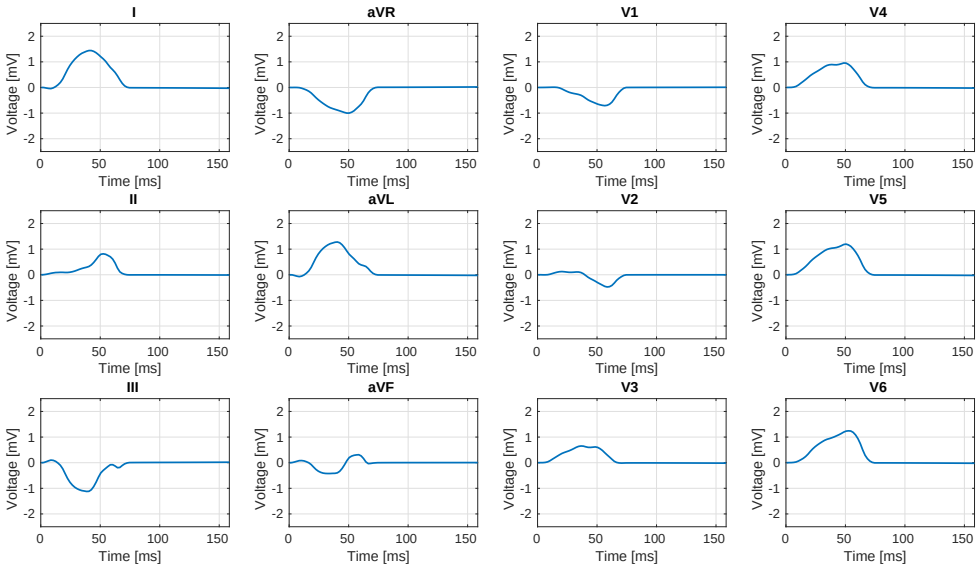
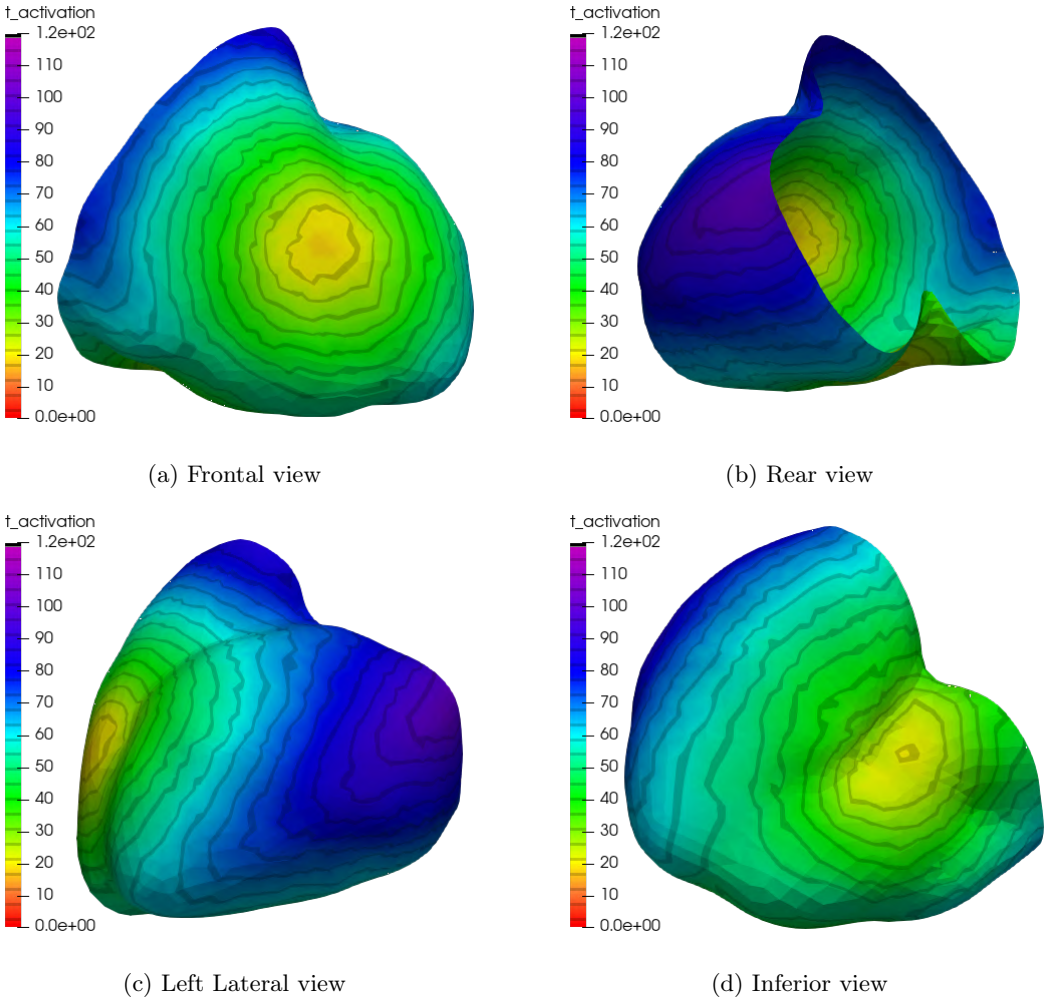


Figure 4.12: 12-lead ECG signals obtained simulating a healthy heart depolarization process



(a) Frontal view

(b) Rear view

(c) Left Lateral view

(d) Inferior view

Figure 4.13: Epicardial activation maps obtained by simulating a heart depolarization process affected by LBBB from four different views

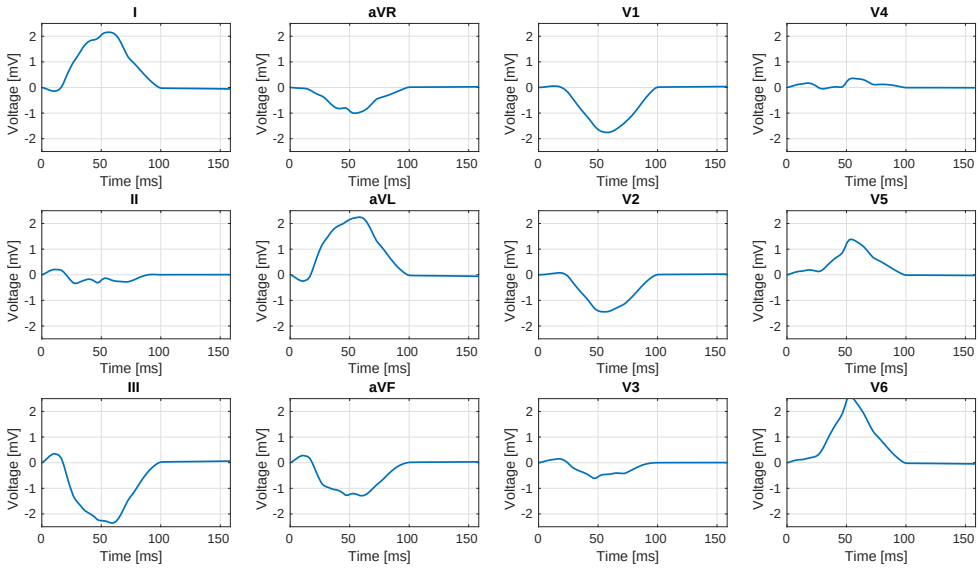


Figure 4.14: 12-lead ECG signals obtained simulating a heart depolarization process affected by LBBB

4.2.4 Model Order Reduction across Space and Time Dimensions

As anticipated in Chapter 1, the PDE-aware DL models that have been developed in this project feature a deterministic layer that solves the generalized Laplace equation (4.36) at the torso, being given as input the conducibility parameters (in case the employed torso geometry involves the presence of multiple organs) and the value of the extracellular potential at the ventricular epicardium. In doing that, it must be taken into account that the problem, at training stage, must be solved many times, precisely, $N_{\boldsymbol{\mu}} N_{epochs}$ times, being $N_{\boldsymbol{\mu}}$ the dimensionality of the training dataset and N_{epochs} the number of training epochs. Thus, it is unfeasible and inconvenient for the NN to solve the FOM problem, both in terms of memory occupation and of computational time. It emerges then the need of resorting to ROM techniques, that allow to compute a solution to the problem at hand which on the one side is less accurate than the one got with the FOM approximation, but, on the other side, brings to a dramatic reduction of the computational burden.

Application of the RB Method to the Generalized Laplace Equation at the Torso

The ROM technique chosen to handle the Forward Problem in the torso is the RB method, already introduced in Section 2.2. More in detail, the problem at hand is steady, but it has to be solved (independently) at many different time instants; thus it can be regarded as a time-dependent problem and we can leverage the concepts introduced in Subsection 2.2.3, concerning ROM techniques for unsteady parametrized PDEs. Since the different time instants are independent one from the other, we have decided not to resort to complex Space-Time-Reduced methods, such as the ST-LSPG one introduced in [30]; rather, we have employed the classical RB method at all the different time instants, thus ultimately performing a dimensionality reduction across the spatial dimension only.

The construction of the Reduced Order Model is quite straightforward and it follows the steps already discussed at the beginning of Subsection 2.2.3; for the sake of precision, we will briefly report them here and fit them to the problem at hand. So, suppose to consider the third order tensor $\mathcal{X}^t \in \mathbb{R}^{N_{\boldsymbol{\mu}} \times N_h^t \times N_t}$ storing the FOM solutions to the generalized Laplace equation at the torso at all the Space-Time DOFs and for $N_{\boldsymbol{\mu}}$ different parameter values. The basis in space, encoded by the matrix $\boldsymbol{V}_s^t \in \mathbb{R}^{N_h^t \times n_h^t}$, is computed by applying a truncated POD (with tolerance on the cumulative squared singular values $\epsilon_{POD}^{t,s}$ to be decided) to the mode-1 unfolding of $\mathcal{X}^t$, denoted as $\mathcal{X}_{(1)}^t \in \mathbb{R}^{N_h^t \times N_{\boldsymbol{\mu}} N_t}$. Figure 4.15 provides the visualization of some of the elements of such basis from an anterolateral point of view.

Once the basis has been computed and all quantities have been projected onto the dimensionality reduced space, Problem 4.7 can be written in reduced form as:

$$\tilde{\boldsymbol{A}}^t(\boldsymbol{\mu}) \tilde{\boldsymbol{u}}_{t_h}^{(l)} = \tilde{\boldsymbol{u}}_{e_h}^{\Omega_T{}^{(l)}}(\boldsymbol{\mu}) \qquad l \in \{0, \dots, N_t\} \tag{4.62}$$

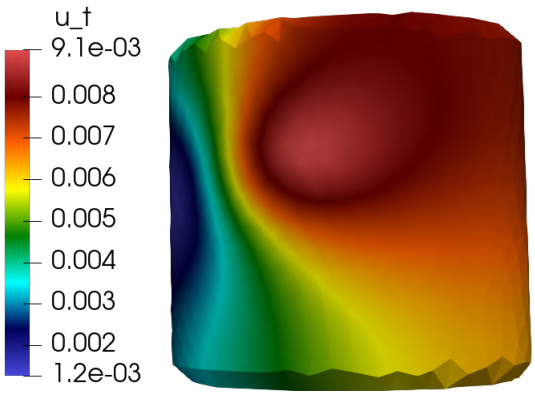
where:

$$\tilde{\boldsymbol{A}}^t(\boldsymbol{\mu}) =: \boldsymbol{V}_s^{t\,T} \boldsymbol{A}^t(\boldsymbol{\mu}) \boldsymbol{V}_s^t \qquad \in \mathbb{R}^{n_h^t \times n_h^t} \tag{4.63a}$$

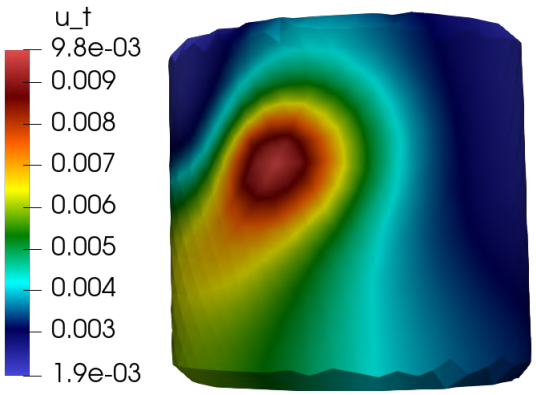
$$\tilde{\boldsymbol{u}}_{t_h}^{(l)} =: \boldsymbol{V}_s^{t\,T} \boldsymbol{u}_{t_h}^{(l)} \qquad \in \mathbb{R}^{n_h^t} \quad l \in \{0, \dots, N_t\} \tag{4.63b}$$

$$\tilde{\boldsymbol{u}}_{e_h}^{\Omega_T{}^{(l)}}(\boldsymbol{\mu}) =: -\boldsymbol{V}_s^{t\,T} \boldsymbol{A}^t(\boldsymbol{\mu}) \boldsymbol{u}_{e_h}^{\Omega_T{}^{(l)}}(\boldsymbol{\mu}) \qquad \in \mathbb{R}^{n_h^t} \quad l \in \{0, \dots, N_t\} \tag{4.63c}$$

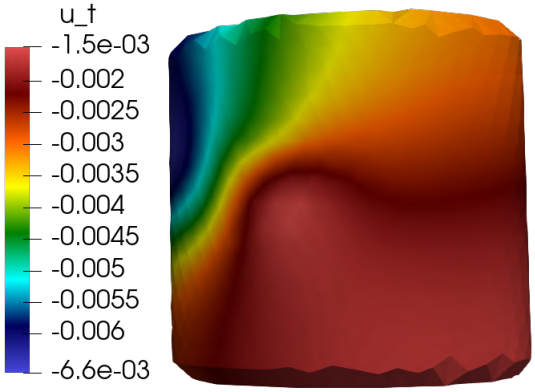
Now, two additional steps can be made. First, exploiting the formulation of the FOM torso problem (4.58), we can perform a dimensionality reduction step also on the epicardial extracellular potential. Indeed, let $\mathcal{X}^e \in \mathbb{R}^{N_{\boldsymbol{\mu}} \times N_h^e \times N_t}$ be defined as the third-order tensor storing the values of the epicardial extracellular potential $\boldsymbol{u}_{e_h}^{\Gamma_H}$ at all Space-Time DOFs of the epicardial surface and for $N_{\boldsymbol{\mu}}$ different parameter values; here N_h^e denotes the number of DOFs at the epicardial surface. Then a reduced basis in space, stored by the matrix $\boldsymbol{V}_s^e \in \mathbb{R}^{N_h^e \times n_h^e}$, can be computed by applying a truncated POD with tolerance $\epsilon_{POD}^{e,s}$ to the mode-1 unfolding of such tensor, i.e.



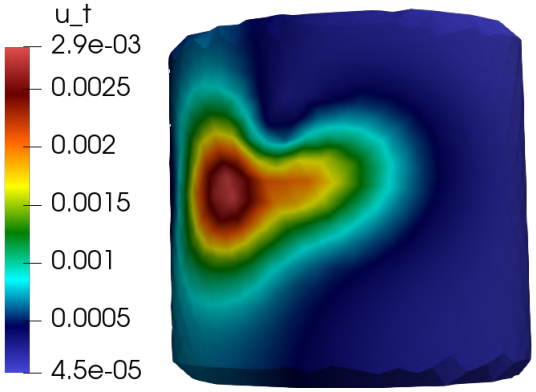
(a) Torso Basis Function 1



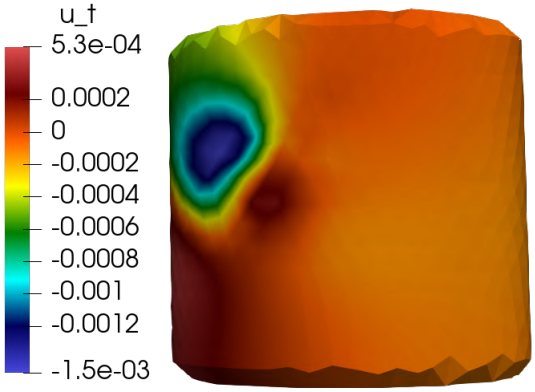
(b) Torso Basis Function 5



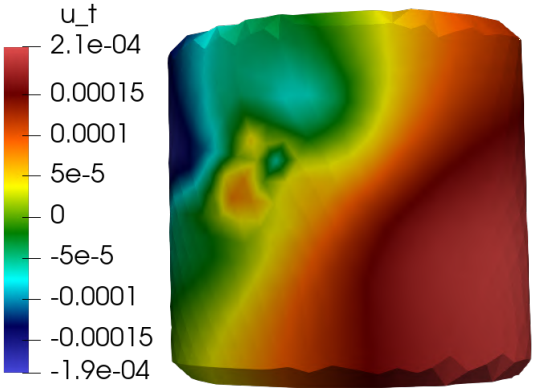
(c) Torso Basis Function 10



(d) Torso Basis Function 20



(e) Torso Basis Function 40



(f) Torso Basis Function 80

Figure 4.15: Visualization, from an anterolateral point of view, of some of the basis functions for the torso potential. A POD tolerance $\epsilon_{POD}^{t,s} = 10^{-3}$ has been chosen; randomized SVD algorithm has been used

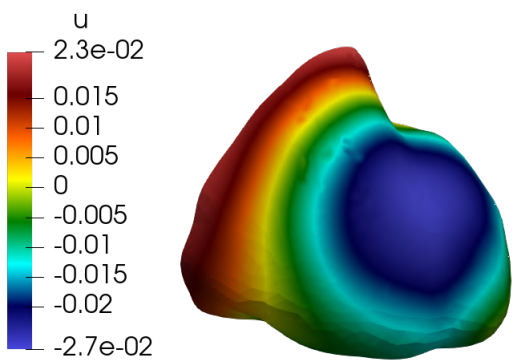
to $\mathcal{X}_{(1)}^e \in \mathbb{R}^{N_h^e \times N_\mu N_t}$. In the end, problem (4.58) can be written in reduced form as:

$$\tilde{\boldsymbol{A}}^t(\boldsymbol{\mu})\tilde{\boldsymbol{u}}_{t_h}^{(l)} = \tilde{\boldsymbol{A}}^{t,e}(\boldsymbol{\mu})(\boldsymbol{\mu})\tilde{\boldsymbol{u}}_{e_h}^{\Gamma_H{}^{(l)}}(\boldsymbol{\mu}) \qquad l \in \{0, \dots, N_t\} \tag{4.64}$$

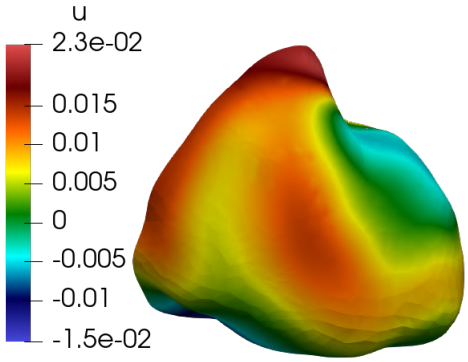
where $\tilde{\boldsymbol{A}}^t(\boldsymbol{\mu})$ and $\tilde{\boldsymbol{u}}_{t_h}^{(l)}$ are defined as in (4.63a) and (4.63b) respectively, while

$$\tilde{\boldsymbol{A}}^{t,e}(\boldsymbol{\mu}) =: \boldsymbol{V}_s^{t^T}\boldsymbol{A}^t(\boldsymbol{\mu})\boldsymbol{V}_s^e \qquad \in \mathbb{R}^{n_h^t \times n_h^e} \tag{4.65a}$$

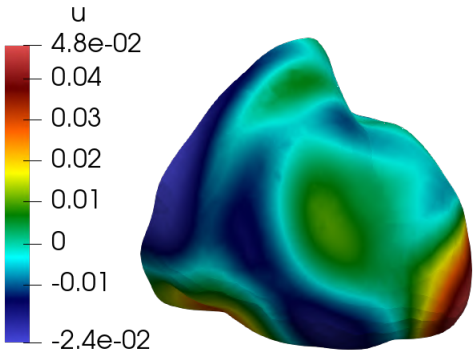
$$\tilde{\boldsymbol{u}}_{e_h}^{\Gamma_H{}^{(l)}}(\boldsymbol{\mu}) =: -\boldsymbol{V}_s^e \boldsymbol{u}_{e_h}^{\Gamma_H{}^{(l)}}(\boldsymbol{\mu}) \qquad \in \mathbb{R}^{n_h^e} \qquad l \in \{0, \dots, N_t\} \tag{4.65b}$$



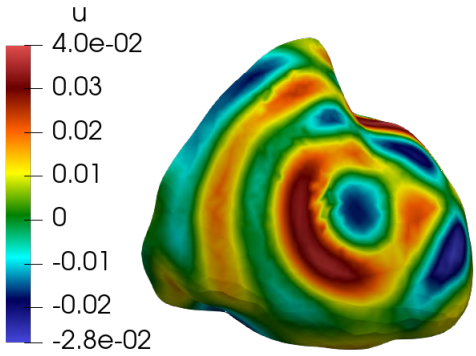
(a) Epicardial Basis Function 1



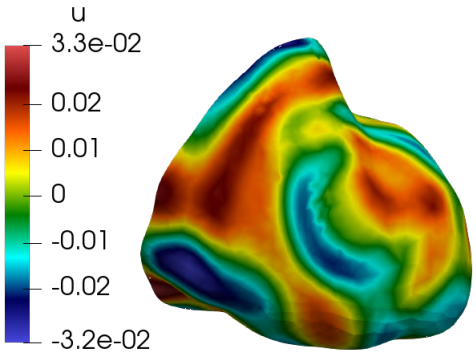
(b) Epicardial Basis Function 5



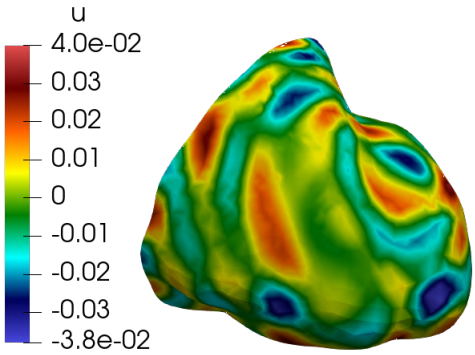
(c) Epicardial Basis Function 10



(d) Epicardial Basis Function 20



(e) Epicardial Basis Function 40



(f) Epicardial Basis Function 80

Figure 4.16: Visualization, from a frontal point of view, of some of the basis functions for the epicardial extracellular potential. A POD tolerance $\epsilon_{POD}^{e,s} = 10^{-2}$ has been chosen; randomized SVD algorithm has been used

A visualization, from a frontal point of view, of some of the basis functions for the epicardial extracellular potential, is offered by Figure 4.16. It can be easily noticed that the spatial frequency of the basis elements increases as their index increases; so the first basis functions, able to explain most of the problem dynamics and associated to the highest singular values, show simple and smooth spatial patterns, while the last ones exhibit much more complicated and irregular shapes.

The second step, instead, consists in taking advantage of the affine parametrization of the torso stiffness matrix $\boldsymbol{A}^t(\boldsymbol{\mu})$. At this point, is is worth remarking that the set of parameters that influence the torso stiffness matrix and the epicardial extracellular potential are disjoint; indeed the former depends on the values of the conducibilities in the torso, while the latter on the

set of parameters defined for the modeling of heart EP. Denoting these sets as $\mathcal{P}^t \subset \mathbb{R}^{N_p^t}$ and $\mathcal{P}^e \subset \mathbb{R}^{N_p^e}$ respectively, we can rewrite problem (4.64) as:

$$\tilde{\boldsymbol{A}}^t(\boldsymbol{\mu}^t)\tilde{\boldsymbol{u}}_{t_h}^{(l)} = \tilde{\boldsymbol{A}}^{t,e}(\boldsymbol{\mu}^t)\tilde{\boldsymbol{u}}_{e_h}^{\Gamma_H{}^{(l)}}(\boldsymbol{\mu}^e) \qquad l \in \{0,\dots,N_t\},\ \boldsymbol{\mu}^t \in \mathcal{P}^t,\ \boldsymbol{\mu}^e \in \mathcal{P}^e \tag{4.66}$$

We have assumed that the torso is made of different parts (like bones, lungs, ...) and that each of these parts is characterized by a different value of the conducibility; anyway we have also assumed the torso to behave like an isotropic conductor, so that the conducibility tensors can be reduced to non-negative scalar values. Thus, if we suppose the torso to be partitioned into N_p^t parts, each one equipped with its own conducibility parameter $\{\sigma_p^t\}_{p=1}^{N_p^t}$, then the torso stiffness matrix can be written as:

$$\boldsymbol{A}^t(\boldsymbol{\mu}^t) \equiv \boldsymbol{A}^t\big(\{\sigma_p^t\}_{p=1}^{N_p^t}\big) = \sum_{p=1}^{N_p^t} \sigma_p^t \boldsymbol{A}_p^t \tag{4.67}$$

where $\boldsymbol{A}_p^t$ is the stiffness matrix associated to the DOFs belonging to the part p of the torso and computed assuming a default conducibility of 1. Readily, this applies also to the reduced torso stiffness matrix, so that:

$$\tilde{\boldsymbol{A}}^t(\boldsymbol{\mu}^t) \equiv \tilde{\boldsymbol{A}}^t\big(\{\sigma_p^t\}_{p=1}^{N_p^t}\big) = \sum_{p=1}^{N_p^t} \sigma_p^t \tilde{\boldsymbol{A}}_p^t \tag{4.68}$$

where:

$$\tilde{\boldsymbol{A}}_p^t =: {\boldsymbol{V}_s^t}^T \boldsymbol{A}_p^t \boldsymbol{V}_s^t \qquad \in \mathbb{R}^{n_h^t \times n_h^t} \quad p \in \{1,\dots,N_p^t\} \tag{4.69}$$

Thus, as already explained in Subsection 2.2.2, all the affine components of the stiffness matrix can be assembled during the *offline phase* of the RB method, saving a significant amount of computational resources during the *online phase*.

In the end, the RB method applied to the generalized Laplace equation at the torso can be schematized as follows.

- **Offline Phase**
  1. The heart EP problem is solved $N_{\boldsymbol{\mu}}$ times, for different values of the parameters stored in $\boldsymbol{\mu}^e$. This allows to assemble the extracellular potential snapshots' tensor $\mathcal{X}^e$.
  2. The generalized Laplace equation at the torso is solved $N_{\boldsymbol{\mu}}$ times, for the $N_{\boldsymbol{\mu}}$ different values of the epicardial extracellular potential computed while solving the heart EP and for $N_{\boldsymbol{\mu}}$ different values of the conducibilities in the torso. This allows to assemble the torso potential snapshots' tensor $\mathcal{X}^t$.
  3. Truncated PODs applied to the mode-1 unfoldings of the snapshots' tensors $\mathcal{X}^t$ and $\mathcal{X}^e$ allow to compute the two Reduced Basis matrices $\boldsymbol{V}_s^t$ and $\boldsymbol{V}_s^e$. The tolerances are denoted as $\epsilon_{POD}^{t,s}$ and $\epsilon_{POD}^{e,s}$ respectively.
  4. The reduced affine components of the torso stiffness matrix are assembled, using (4.69).
- **Online Phase**
  1. Given the torso conducibilities $\boldsymbol{\mu}^t = \{\sigma_p^t\}_{p=1}^{N_p^t}$, the reduced torso stiffness matrix $\tilde{\boldsymbol{A}}^t(\boldsymbol{\mu}^t)$ is obtained as a linear combination.
  2. Given the epicardial extracellular potential at all time instants $\{\boldsymbol{u}_{e_h}^{\Gamma_H{}^{(l)}}(\boldsymbol{\mu}^e)\}_{l=0}^{N_t}$, its projection onto the epicardial Reduced Subspace $\{\tilde{\boldsymbol{u}}_{e_h}^{\Gamma_H{}^{(l)}}(\boldsymbol{\mu}^e)\}_{l=0}^{N_t}$ is computed
  3. The torso potential, projected onto the torso Reduced Subspace, is computed at all time instants by solving the set of linear systems (4.66)
  4. The FOM torso potential at all time instants is retrieved from the reduced one by computing $\boldsymbol{u}_{t_h}^{(l)} = \boldsymbol{V}_s^t \tilde{\boldsymbol{u}}_{t_h}^{(l)} \quad \forall\, l \in \{0,\dots,N_t\}$

Space-Time RB Approximation of the Epicardial Extracellular Potential

As it will be made clear in Chapter 5, the PDE-aware DL models implemented in this project aim at estimating the epicardial extracellular potential at all the Space-Time DOFs, starting from body surface potentials given as input. Performing such an estimation in the FOM setting is clearly unfeasible, since on the one side the NN would be required to learn a very high number of quantities (much higher than the ones given as input) and on the other side the complexity of the model would dramatically increase as the computational meshes get refined or as the order of the method is increased, which is not desired. A first fix to this problem is to perform a dimensionality reduction across the space dimension, employing the *RB* method as it has been described in the previous paragraph; in this way, the model would be required to learn $n_h^e N_t$ coefficients, rather than $N_h^e N_t$, thus reducing its complexity ($n_h^e \ll N_h^e$). Anyway, the number of trainable parameters still depends on the number of time instants N_t and a refinement of the discretization of the time domain will lead to a significant increase of the NN complexity.

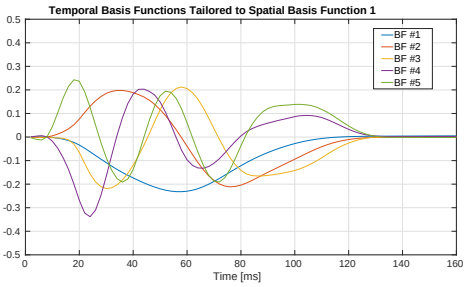
Such a problem can be then tackled by performing a Space-Time projection of the epicardial extracellular potential, using the techniques already described in Subsection 2.2.3. In particular, given the third-order tensor $\mathcal{X}^e$ storing the epicardial potential at all the FOM Space-Time DOFs, several truncated PODs are applied. A first one, with tolerance $\epsilon_{POD}^{e,s}$, acts on the mode-1 unfolding of the tensor $\mathcal{X}_{(1)}^e$ and it allows to compute the spatial basis $\boldsymbol{V}_s^e \in \mathbb{R}^{N_h^e \times n_h^e}$. Lately, n_h^e different PODs are applied to the projections of the snapshots' tensor onto the spaces spanned by the different spatial basis elements i.e. $\mathcal{X}^e(\boldsymbol{V}_{s_i}^t) =: \mathcal{X}^e \times_1 \boldsymbol{V}_{s_i}^t \in \mathbb{R}^{N_t \times N_\mu}, \quad i \in \{1, \dots, n_h^e\}$. These PODs allow to compute the temporal bases $\boldsymbol{V}_{t_i}^e \in \mathbb{R}^{N_t \times n_t^i} \quad \forall\, i \in \{1, \dots, n_h^e\}$. Concerning the tolerances, it is a common strategy to take them constant for all the n_h^e PODs; a drawback we noticed, anyway, is that, with this approach, we get lots of temporal basis functions associated to the least informative spatial basis functions and just a few tailored to the most relevant ones. Thus, in order to further reduce the overall number of Space-Time basis functions, we decided to pick the tolerances for the temporal PODs as follows:

$$\left(\epsilon_{POD}^{e,t}\right)_i = 10\ \hat{\epsilon}_{POD}^{e,t} \frac{\sum_{j=1}^{i} {\sigma_j}^{e,s}}{\sum_{j=1}^{n_h^e} {\sigma_j}^{e,s}} \qquad i \in \{1, \dots, n_h^e\} \tag{4.70}$$

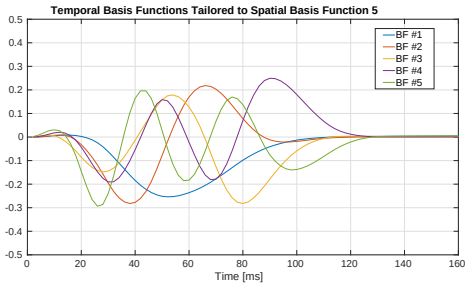
In this way, the first spatial basis function will be associated to a POD tolerance close to the reference one $\hat{\epsilon}_{POD}^{e,t}$, while the subsequent ones will be coupled to higher and higher tolerances (whose trend depends on the singular values, representative of the basis importance), up to a value equal to 10 times $\hat{\epsilon}_{POD}^{e,t}$. With such approach, the number of temporal basis functions associated to the different spatial basis ones is much more homogeneous and the loss in terms of accuracy is minimal, since only the least informative elements are excluded from the final spatio-temporal Reduced Basis. In the following, for simplicity, we will refer to $\hat{\epsilon}_{POD}^{e,t}$ as the temporal POD tolerance on the epicardial potential.

The overall Space-Time Reduced Basis is then made of $n_{st} =: \sum_{i=1}^{n_h^e} n_t^i$ elements of dimension $N_h^e \times N_t$, each one being defined as the outer product between an element of the Reduced Basis in space and an element of the associated Reduced Basis in time. For a more precise definition of all these concepts we refer again to Subsection 2.2.3. Figure 4.17 displays the temporal basis functions associated to the spatial ones (for the epicardial extracellular potential) represented in Figure 4.16. Notice how the high frequency component increases both as the index of the function itself increases (and this is a standard and expected trend) and also as the index of the associated basis function in space increases. This last observation justifies the choice of the "tailored" approach with respect to "non-tailored" one for the computation of the temporal basis.

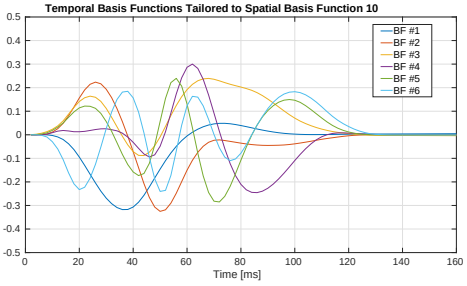
In the end, any epicardial extracellular potential $\{\boldsymbol{u}_{e_h}^{\Gamma_H\,(l)}(\boldsymbol{\mu}^e)\}_{l=0}^{N_t}$ can be expressed by means of only n_{st} coefficients, which are the ones that are ultimately estimated by our DL model. In this way, on the one side the NN complexity is significantly reduced (since $n_h^e \ll N_h^e$ and $n_t^i \ll N_t \quad \forall\, i \in \{1, \dots, n_h^e\}$ if sufficiently high POD tolerances are chosen) and on the other side the number of trainable parameters becomes much less sensitive to the level of refinement of the



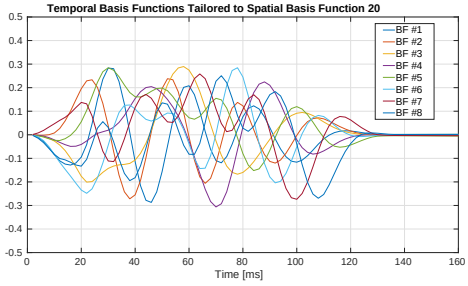
(a) Temporal Basis tailored to ϕ_1



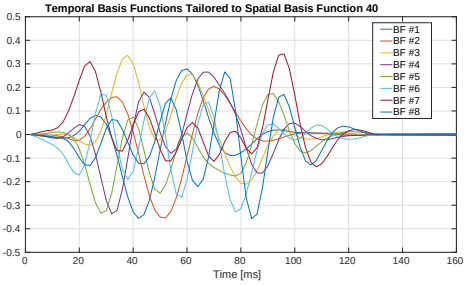
(b) Temporal Basis tailored to ϕ_5



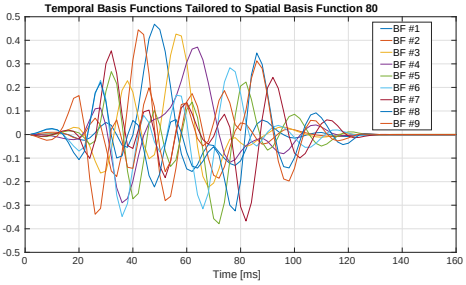
(c) Temporal Basis tailored to ϕ_{10}



(d) Temporal Basis tailored to ϕ_{20}



(e) Temporal Basis tailored to ϕ_{40}



(f) Temporal Basis tailored to ϕ_{80}

Figure 4.17: Visualization of the temporal reduced basis functions associated to the spatial basis functions of Figure 4.16 for the epicardial extracellular potential. A reference POD tolerance $\epsilon_{POD}^{e,t} = 5 \cdot 10^{-2}$ has been chosen

Space-Time discretization (since, for a sufficiently good level of refinement, it can be assumed independence between the number of FOM DOFs and the dimensionality of the Reduced Bases).

In this context, the computation of the solution to the generalized Laplace equation at the torso basically proceeds as described in the previous paragraph, requiring to solve the set of linear systems (4.66). The only difference is that now we are no longer given the FOM epicardial potential $\boldsymbol{u}_{e_h}^{\Gamma_H}(\boldsymbol{\mu}^e)$, but its projection onto the Space-Time Reduced Subspace $\hat{\boldsymbol{u}}_{e_h}^{\Gamma_H}(\boldsymbol{\mu}^e)$, which, exploiting orthonormality, is computed as:

$$\left(\hat{\boldsymbol{u}}_{e_h}^{\Gamma_H}(\boldsymbol{\mu}^e)\right)_k = \sum_{i=0}^{N_h^e} \sum_{j=0}^{N_t} \boldsymbol{\pi}_{ij}^k \left(\boldsymbol{u}_{e_h}^{\Gamma_H}(\boldsymbol{\mu}^e)\right)_{i,j} \qquad k \in \mathbb{N}(n_{st}) \tag{4.71}$$

where $\boldsymbol{\pi}_{ij}^k$ denotes the element in position (i,j) of the k-th Space-Time basis function. Thus, instead of computing the ROM-in-Space FOM-in-Time epicardial potential $\{\tilde{\boldsymbol{u}}_{e_h}^{\Gamma_H{}^{(l)}}(\boldsymbol{\mu}^e)\}_{l=0}^{N_t}$ by projecting the FOM one onto the Reduced subspace in space, that same quantity is derived by expanding in time the ROM-in-Space ROM-in-Time epicardial potential $\hat{\boldsymbol{u}}_{e_h}^{\Gamma_H}(\boldsymbol{\mu}^e)$; this is done with the formula

$$\left(\tilde{\boldsymbol{u}}^{\Gamma_H{}^{(l)}}(\boldsymbol{\mu}^e)\right)_i = \sum_{j=0}^{n_t^i} \boldsymbol{\psi}_j^{i^{(l)}} \hat{\boldsymbol{u}}_{\mathcal{F}(i,j)}^{\Gamma_H}(\boldsymbol{\mu}^e) \qquad i \in \{1,\dots,n_h^e\} \quad l \in \{0,\dots,N_t\} \tag{4.72}$$

Case Nb	$\epsilon_{\mathbf{POD}}^{\mathbf{t,s}}$	$\epsilon_{\mathbf{POD}}^{\mathbf{e,s}}$	$\hat{\epsilon}_{\mathbf{POD}}^{\mathbf{e,t}}$	n_h^t	n_h^e	n_{st}	err_{ECG}	err_{AM}
1	10^{-1}	10^{-1}	10^{-1}	36	80	259	$4.34 \cdot 10^{-2}$	$4.27 \cdot 10^{-2}$
2	10^{-1}	10^{-1}	10^{-2}	36	80	1368	$9.48 \cdot 10^{-3}$	$1.42 \cdot 10^{-2}$
3	10^{-1}	10^{-1}	10^{-3}	36	80	3303	$5.17 \cdot 10^{-3}$	$1.44 \cdot 10^{-2}$
4	10^{-1}	10^{-2}	10^{-1}	36	293	555	$4.34 \cdot 10^{-2}$	$3.79 \cdot 10^{-2}$
5	10^{-1}	10^{-2}	10^{-2}	36	293	7506	$9.36 \cdot 10^{-3}$	$5.28 \cdot 10^{-3}$
6	10^{-1}	10^{-2}	10^{-3}	36	293	14947	$4.98 \cdot 10^{-3}$	$3.45 \cdot 10^{-3}$
7	10^{-1}	10^{-3}	10^{-1}	36	320	581	$4.34 \cdot 10^{-2}$	$3.76 \cdot 10^{-2}$
8	10^{-1}	10^{-3}	10^{-2}	36	320	8465	$9.36 \cdot 10^{-3}$	$5.04 \cdot 10^{-3}$
9	10^{-1}	10^{-3}	10^{-3}	36	320	16526	$4.99 \cdot 10^{-3}$	$3.07 \cdot 10^{-3}$
10	10^{-2}	10^{-1}	10^{-1}	194	80	259	$4.34 \cdot 10^{-2}$	$4.27 \cdot 10^{-2}$
11	10^{-2}	10^{-1}	10^{-2}	194	80	1368	$7.72 \cdot 10^{-3}$	$1.42 \cdot 10^{-2}$
12	10^{-2}	10^{-1}	10^{-3}	194	80	3303	$2.47 \cdot 10^{-3}$	$1.44 \cdot 10^{-2}$
13	10^{-2}	10^{-2}	10^{-1}	194	293	555	$4.34 \cdot 10^{-2}$	$3.79 \cdot 10^{-2}$
14	10^{-2}	10^{-2}	10^{-2}	194	293	7506	$7.14 \cdot 10^{-3}$	$5.28 \cdot 10^{-3}$
15	10^{-2}	10^{-2}	10^{-3}	194	293	14947	$7.78 \cdot 10^{-4}$	$3.45 \cdot 10^{-3}$
16	10^{-2}	10^{-3}	10^{-1}	194	320	581	$4.34 \cdot 10^{-2}$	$3.76 \cdot 10^{-3}$
17	10^{-2}	10^{-3}	10^{-2}	194	320	8465	$7.14 \cdot 10^{-3}$	$5.04 \cdot 10^{-3}$
18	10^{-2}	10^{-3}	10^{-3}	194	320	16526	$7.72 \cdot 10^{-4}$	$3.07 \cdot 10^{-3}$
19	10^{-3}	10^{-1}	10^{-1}	316	80	259	$4.34 \cdot 10^{-2}$	$4.27 \cdot 10^{-2}$
20	10^{-3}	10^{-1}	10^{-2}	316	80	1368	$7.72 \cdot 10^{-3}$	$1.42 \cdot 10^{-2}$
21	10^{-3}	10^{-1}	10^{-3}	316	80	3303	$2.47 \cdot 10^{-3}$	$1.44 \cdot 10^{-2}$
22	10^{-3}	10^{-2}	10^{-1}	316	293	555	$4.34 \cdot 10^{-2}$	$3.79 \cdot 10^{-2}$
23	10^{-3}	10^{-2}	10^{-2}	316	293	7506	$7.14 \cdot 10^{-3}$	$5.28 \cdot 10^{-3}$
24	10^{-3}	10^{-2}	10^{-3}	316	293	14947	$7.33 \cdot 10^{-4}$	$3.45 \cdot 10^{-3}$
25	10^{-3}	10^{-3}	10^{-1}	316	320	581	$4.34 \cdot 10^{-2}$	$3.76 \cdot 10^{-2}$
26	10^{-3}	10^{-3}	10^{-2}	316	320	8465	$7.13 \cdot 10^{-3}$	$5.04 \cdot 10^{-3}$
27	10^{-3}	10^{-3}	10^{-3}	316	320	16526	$7.21 \cdot 10^{-4}$	$3.07 \cdot 10^{-3}$

Table 4.4: ECG and epicardial activation map errors for different values of the tolerances used in the truncated PODs. ECG errors (err_{ECG}) are computed as absolute l^1-norm errors (over all the signals and all the time instants); activation map errors (err_{AM}) are computed as relative l^1-norm errors. Errors are averaged over 25 different snapshots

being $\boldsymbol{\psi}_j^i$ the j-th basis function in time associated to the i-th basis function in space and

$$\mathcal{F} : \big(\mathbb{N}(n_h^e), \mathbb{N}(n_t^i)\big) \ni \quad (i,j) \rightarrow \sum_{k=1}^{i-1} n_t^k + j \quad \in \mathbb{N}(n_{st})$$

the index mapping from the Space and Time bases indexes to the Space-Time basis index.

Numerical Results

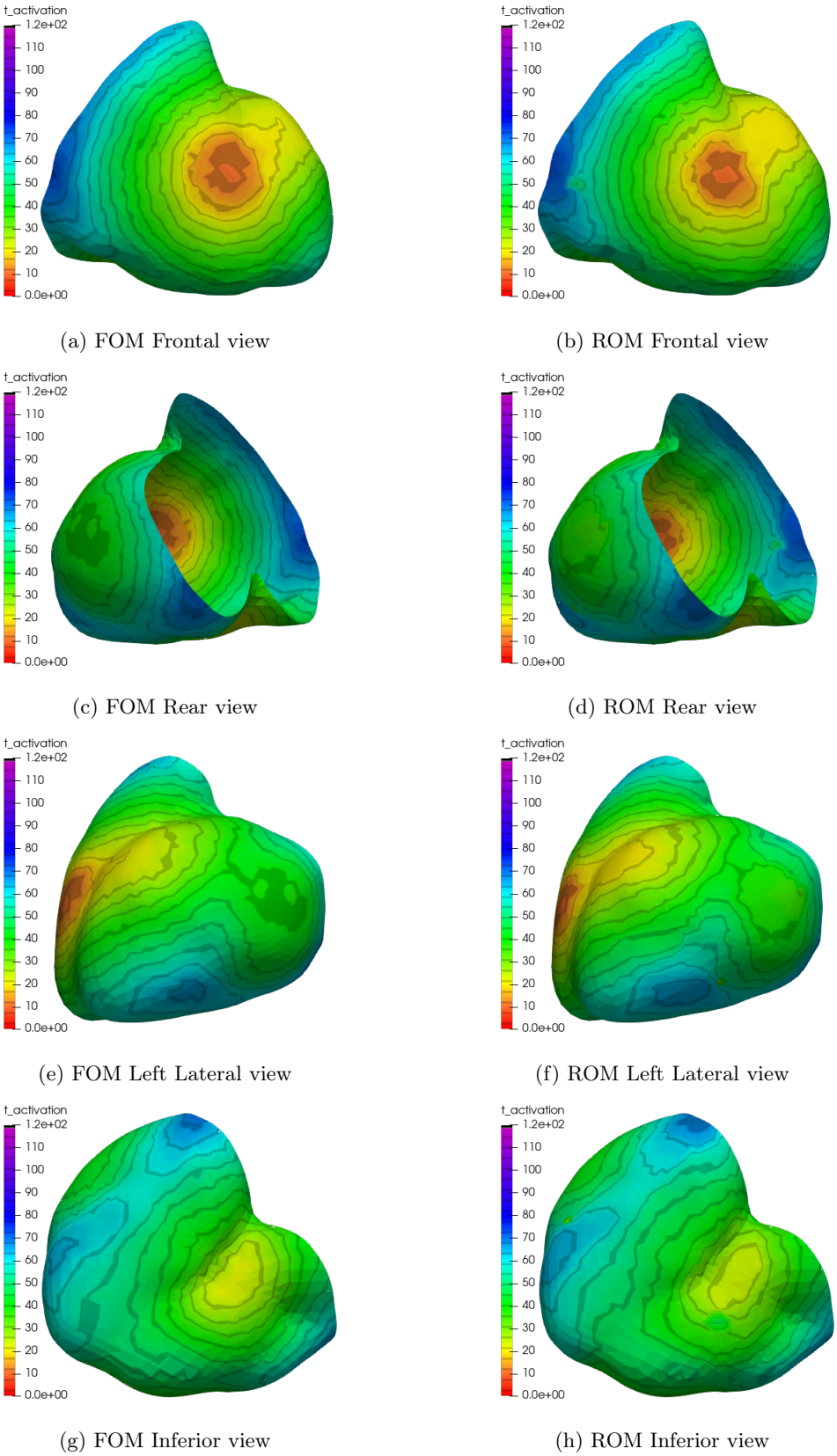
Here we present the results, in terms of epicardial activation maps and 12-lead ECG signals, obtained by employing the Space-Time ROM techniques described before. As a *caveat*, we recall that the torso geometry we adopted does not feature the presence of different organs, thus assuming a homogeneous and isotropic signal transmission; because of this, the torso stiffness matrix is actually parameter independent and all the parametric contribution comes from the epicardial boundary condition. Different Reduced Bases approximations have been tested, for different values of the tolerances used in the three truncated PODs; they have all been computed starting from the same dataset made of 400 solutions, that have been obtained solving the bidomain equations (coupled with the AP ionic model) and the generalized Laplace equation at the torso for different values of the model parameters and data. Additional details on the dataset specifics can be found in Subsubsection 5.2.3.1.

Table 4.4 reports the dimensionality of the Reduced Bases and the errors of the ROM approximation of ECG signals and epicardial activation maps with respect to the FOM one, obtained for 27 different values of the tolerances used in the three PODs. More specifically, values of 10^{-1}, 10^{-2} and 10^{-3} have been chosen for the tolerances of each of the PODs; errors have been computed as averaged over 25 different test cases; errors on ECG signals are l^1-norm absolute errors; errors on epicardial activation maps are l^1-norm relative errors. Trivially, the errors on both ECG signals and epicardial activation maps decrease as the tolerances of the PODs decrease; moreover the error on the epicardial activation maps is not influenced by the tolerance used in the torso POD, since the propagation of the signal in the torso does not influence the heart EP, under the isolated heart assumption.

Focusing on epicardial activation maps errors, we can observe that they are much higher (of order $\approx 1\% - 4\%$) if a tolerance of 10^{-1} is used either for the spatial POD or for the temporal one on the epicardial potential. A significant decreasing of the errors is instead observed if the tolerances are chosen to be equal to 10^{-2} or 10^{-3}; readily the best value is got if they are both equal to 10^{-3}, but that is just $\approx 0.2\%$ smaller than the one got with both tolerances equaling 10^{-2}. On the other side, the dimensionality of the Space-Time Reduced Basis equals 259 in case both tolerances equal 10^{-1}, passing to $7'506$ if they both equal 10^{-2} and to $16'526$ if they both equal 10^{-3}. From a heuristic point of view, the best choice seems then to choose both POD tolerances equal to 10^{-2}; indeed this allows to get small errors ($\approx 0.5\%$) at not too big computational cost. Our purpose, anyway, is to get a fairly good reconstruction of the epicardial activation maps with the smallest possible number of coefficients, since those will have to be estimated by our DL model; thus picking tolerances of 10^{-1} for the spatial POD and of $5 \cdot 10^{-2}$ for the temporal one, which gives 619 coefficients, appears to match our target. A qualitative comparison of the results can be performed by looking at Figure 4.18, where the FOM and the ROM (with the aforementioned POD tolerances) epicardial activation maps are reported, side-by-side, from four different perspectives.

A similar scenario can be observed also for what concerns 12-lead ECG signals, with the only major difference consisting in the fact that the errors are also influenced by the tolerance chosen for the POD on the body surface potentials; such dependency is anyway not too marked. In particular, if the epicardial POD tolerances are both chosen equal to 10^{-1}, the errors on ECG signals are constant at $4.34 \cdot 10^{-2}$, whichever the torso POD tolerance. Instead, if the epicardial POD tolerances are both equal to 10^{-2}, then ECG errors pass from $9.36 \cdot 10^{-3}$ for $\epsilon_{POD}^{t,s} = 10^{-1}$ to $7.14 \cdot 10^{-3}$ for $\epsilon_{POD}^{t,s} = 10^{-2}, 10^{-3}$. More in general, assuming epicardial POD tolerances to be low enough, it can be observed that ECG errors feature a more significant decrease when the body surface potential POD tolerance passes from 10^{-1} to 10^{-2}, while staying more or less stable when it passes from 10^{-2} to 10^{-3}. As a consequence, it seems natural to choose $\epsilon_{POD}^{t,s} = 10^{-2}$, which induces a dimensionality of 194 on the torso Reduced Basis. A qualitative comparison between the FOM and ROM ECG signals (with $\epsilon_{POD}^{t,s} = 10^{-2}$, $\epsilon_{POD}^{e,s} = 10^{-1}$ and $\epsilon_{POD}^{e,t} = 5 \cdot 10^{-2}$) can be performed looking at Figure 4.19; in the eyeball norm, the two curves are basically indistinguishable at all leads.

FOM approximation **ROM approximation**



(a) FOM Frontal view

(b) ROM Frontal view

(c) FOM Rear view

(d) ROM Rear view

(e) FOM Left Lateral view

(f) ROM Left Lateral view

(g) FOM Inferior view

(h) ROM Inferior view

Figure 4.18: Epicardial activation maps obtained with the FOM approximation (left column) and with the ROM approximation (right column), with tolerances of 10^{-1} for the spatial POD and of $5 \cdot 10^{-2}$ for the temporal ones.

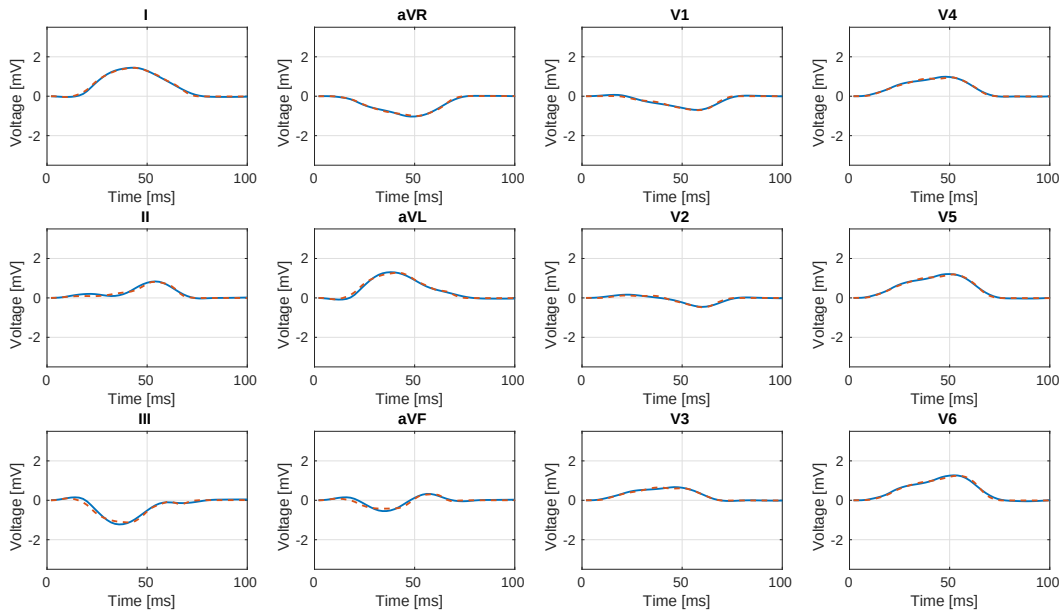


Figure 4.19: ECG signals obtained with the FOM approximation and with the ROM approximation of the generalized Laplace equation at the torso, with tolerances of 10^{-2} for the torso POD, 10^{-1} for the spatial PODs at the epicardium and $5 \cdot 10^{-2}$ for the temporal POD at the epicardium. Blue solid line represents the ECG got with the ROM approximation; red dashed line represents the ECG signal got with the FOM approximation

4.3 The Inverse Problem of Electrocardiography and ECGI

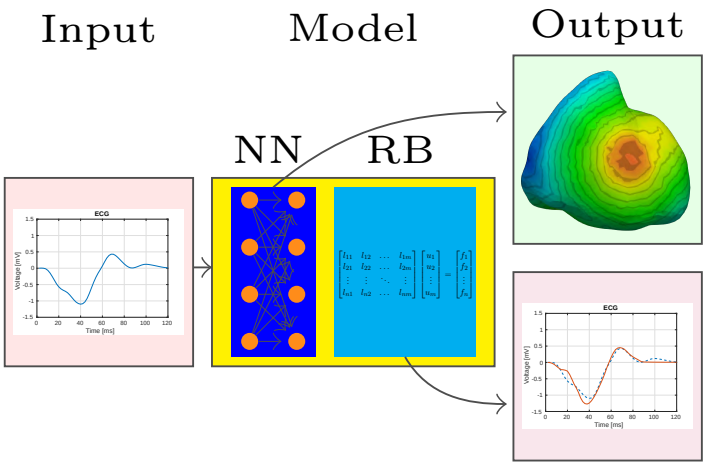


Figure 4.20: Basic structure of the developed PDE-aware DL model, where the model is highlighted

Over the last 50 years, several techniques aimed at recording the body surface potential maps (BSPMs) have spread out, since they have been proved to provide, in many heart diseases, relevant diagnostic information that could not be obtained with more standard recording techniques, as 12-lead ECG signals. Indeed, many features of the potential field on the chest are correlated to the shape and pattern of the depolarization wavefront and of the repolarization process. Anyway, the interpretation of BSPMs is a very difficult task, since they exhibit a highly changeable surface pattern and a high variability of the signal magnitude over the entire heartbeat. Thus, the problem of establishing a bidirectional link between the epicardial potential and the BSPMs has become central in the field of numerical approximation of heart EP and several techniques have been developed at this aim. In particular, the reconstruction of the epicardial potential from BSPMs is called *Potential-Based Inverse Problem* (or *Inverse Potential Problem*) of Electrocardiography and it is the subject of this Section.

As stated in [55], the Inverse Potential Problem is important since it serves as theoretical basis for ECGI, a novel imaging modality for non-invasive mapping of cardiac electrical activity. ECGI is able to reconstruct epicardial potentials, activation maps and recovery sequences on the epicardial surface from the knowledge of body-surface ECG signals and from ECG-gated thoracic CT scans. In terms of clinical applications, ECGI has undergone extensive validation tests on animals (mainly on canine hearts) [56, 57], before being actually applied to humans in various clinical settings [58–61]. An interesting and extensive analysis of the State-Of-Art of ECGI, focused on its validation in view of clinical applicability, is presented in [10].

This Section will be divided in two parts. In the first one (Subsection 4.3.1) the Potential-Based Inverse Problem of Electrocardiography is defined and its properties are briefly discussed; most of the material is taken from Chapter 6 of [41]. The second part (Subsection 4.3.2), instead, offers a brief overview on the most relevant and widely used methods to solve such a (ill-posed) problem.

4.3.1 Mathematical Definition of the Potential-Based Inverse Problem

The linkage the potential field on the thorax to the one at the epicardium can be expressed by means of two different problems, named as the Forward and the Inverse problem of Electrocardiography. The Forward problem reads as in (4.36), with boundary conditions given by (4.37a)-(4.37b); thus it is a simple generalized Laplace equation, where the epicardium acts as a "source" Dirichlet boundary condition and where the torso is approximated as an inhomogeneous isotropic volume conductor. Conversely, the (Potential-Based) Inverse Problem consists

in estimating the epicardial extracellular potential from the knowledge of BSPMs. Its solution would be remarkable from the clinical point of view, since, by measuring BSPMs in a fast and non-invasive way, it would be possible to reconstruct the epicardial potential activation maps, whose interpretation, in terms of the underlying cardiac events, is much easier and more directly readable.

Let us follow the notation adopted in Subsection 4.2; also define $\Sigma \subset \Gamma_B$ as the portion of the body surface where BSPMs are recorded (i.e. the region where the electrodes have been placed) and call $z(\boldsymbol{x})$ such recordings. Then, we have the following Cauchy problem:

$$\begin{cases} -div(D_T(\boldsymbol{x}) \bigtriangledown u_T(\boldsymbol{x})) = 0 & in\ \Omega_T \\ u_T(\boldsymbol{x}) = z(\boldsymbol{x}) & on\ \Sigma \\ \dfrac{\partial u_T(\boldsymbol{x})}{\partial \boldsymbol{n_B}} = 0 & on\ \Gamma_B \end{cases} \tag{4.73}$$

being $\boldsymbol{n_B}$ the outward unit normal vector to Γ_B. The Inverse Problem consists in estimating the potential $u_T(\boldsymbol{x})$ on the epicardial surface Γ_H. So, let $v \in L^2(\Gamma_H)$ define the state $y(\boldsymbol{x}; v) = y(v)$ as the unique solution in $H^{\frac{1}{2}}(\Omega_T)$ of

$$\begin{cases} -div(D_T(\boldsymbol{x}) \bigtriangledown y(v)) = 0 & in\ \Omega_T \\ y(v) = v & on\ \Gamma_H \\ \dfrac{\partial y(v)}{\partial \boldsymbol{n_B}} = 0 & on\ \Gamma_B \end{cases} \tag{4.74}$$

Let us also define the operator $A : L^2(\Gamma_H) \longrightarrow L^2(\Sigma)$ such that

$$Av = y(v)|_\Sigma \tag{4.75}$$

and the cost function

$$J(v) = \int_\Sigma |y(v) - z|^2 \, d\sigma = ||Av - z||^2_{L^2(\Sigma)} \tag{4.76}$$

for $z \in L^2(\Sigma)$. Then, the Inverse Problem can be written as:

$$find\ u_H \in L^2(\Gamma_H): \quad J(u_H) = \min_{v \in L^2(\Gamma_H)} J(v) \tag{4.77}$$

If $z \in A(L^2(\Gamma_H))$ (i.e. (4.73) admits a unique solution), then u_T would satisfy $Au_T = z$ and thus it would be the unique minimizer of $J(\cdot)$. The big issue is that the problem is ill-posed (in the sense of Hadamard) in usual Sobolev spaces, i.e. A admits an unbounded inverse operator in all the spaces H^s, $\forall\ s \in \mathbb{R}$ (see [62,63]). This implies that small perturbations (for instance due to measurement errors) in the observed surface potential $z(\boldsymbol{x})$ may lead to much larger variations of the reconstructed epicardial potential $u_T(\boldsymbol{x})|_{\Gamma_H}$. Thus, different techniques, somehow involving stabilization and regularization, have to be put in place; an overview on the most relevant ones is provided in the next Subsection.

4.3.2 Numerical Methods for the Inverse Potential Problem

In the following we provide a brief overview on the most widely used methods to regularize and stabilize the Inverse Potential Problem of Electrocardiography. We refer the interested reader to [64–66] for a much more detailed and extensive discussion on the theme, under the assumption that the FE method is used. Other than the presented approaches, it also deserves a mention the work of *Schuler et al.* [67]; indeed the authors make use of Space-Time model order reduction (MOR) (as we did) of body surface potentials to predict the epicardial one, in the context of an improved version of the classical Tikhonov regularization algorithm (see [68]).

1. Tikhonov Regularization

The most straightforward stabilization technique is given by *Tikhonov* regularization (see [69]), were suitable $L^2(\Gamma_H)$-norm smoothing constraints, justified by the physical problem, are added to the cost functional. The first applications of such technique in the context of the Inverse Potential Problem of Electrocardiography date back to the 80s with the works by *Colli-Franzone et al.* [70] and by *Rudy and Oster* [71]. So, problem (4.77) is stabilized by defining a regularized cost functional of the form

$$J_\lambda(v) = J(v) + \lambda \int_{\Gamma_H} |Bv|^2 \, d\sigma, \quad v \in U, \quad \lambda \in \mathbb{R}^+ \tag{4.78}$$

where B is the regularization operator and U is the space of admissible controls. Three main choices can be made in this sense and they are reported in Table 4.5.

Order \ Object	Regularization Operator	Space of Admissible Controls
Zero	$B = I\|_{\Gamma_H}$	$U = L^2(\Gamma_H)$
First	$B = \bigtriangledown\|_{\Gamma_H}$	$U = H^1(\Gamma_H)$
Second	$B = \triangle\|_{\Gamma_H}$	$U = H^2(\Gamma_H)$

Table 4.5: Regularization operators and spaces of admissible controls for the Tikhonov regularization approaches of zero, first and second order

The inverse problem (4.77) is then approximated by the following family of regularized problems:

$$find \; u_H^\lambda \in U: \quad J_\lambda(u_H^\lambda) = \min_{v \in U} J_\lambda(v) \tag{4.79}$$

In discrete form (for instance via the FE method), the solution to (4.79) can be written as:

$$\hat{\boldsymbol{v}} = \boldsymbol{A}^\dagger \boldsymbol{z} = (\boldsymbol{A}^T \boldsymbol{A} + \lambda^2 \boldsymbol{B}^T \boldsymbol{B})^{-1} \boldsymbol{A}^T \boldsymbol{z} \tag{4.80}$$

being $\boldsymbol{A}$ the discrete transfer matrix (i.e. the discrete counter-part of the operator A in (4.75)), $\boldsymbol{B}$ the discrete regularization operator and $\boldsymbol{z}$ the discrete vector of recorded body surface potentials. All details related to the numerical computation of $\boldsymbol{A}$ and $\boldsymbol{B}$ using a FE approximation can be found for instance in [65].

[72] contains a detailed proof of the well-posedness of the problem and of the convergence of $u_H^\lambda$to u_H as $\lambda \to 0^+$. Section 6.1 of [41] features a detailed analysis of the numerical approximation of the solution to (4.79).

2. Total Variation Regularization

In [55], *Ghosh and Rudy* have proposed to use L^1-norm penalties instead of the L^2-norm ones defined before in the context of Tikhonov regularization. In particular, they adopted L^1-norm regularization on the normal derivative of the epicardial potential (from which the name *Total Variation Regularization*), thus minimizing the following cost functional:

$$J_\lambda(v) = J(v) + \lambda \int_{\Gamma_H} |\bigtriangledown v \cdot \boldsymbol{n_H}| \, d\sigma, \quad v \in U; \quad \lambda \in \mathbb{R}^+ \tag{4.81}$$

being $\boldsymbol{n_H}$ the outward unit normal vector to Γ_H. The idea is that the L^2-norm regularization causes a too significant smoothing of the solution, which can be instead reduced by resorting to L^1-norm regularization; as a drawback, the resulting problem is non-linear and, thus, more difficult to handle numerically.

3. Generalized Singular Value Decomposition

Coming back to the Tikhonov regularization framework, we can write the problem in discrete form (for instance upon the usage of the FE method) as:

$$find\ \hat{\boldsymbol{v}} \in \mathbb{R}^{N_h^e}: \quad \hat{\boldsymbol{v}} = \min_{v \in \mathbb{R}^{N_h^e}} \left\{ ||\boldsymbol{A}\boldsymbol{v} - \boldsymbol{z}||^2 + \lambda^2 ||\boldsymbol{B}\boldsymbol{v}||^2 \right\} \tag{4.82}$$

being $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{z}$ defined as in (4.80).

If $\boldsymbol{B} = \boldsymbol{I}$, it is possible to take advantage of the SVD on the discrete transfer matrix $\boldsymbol{A}$ (i.e. $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}$) to write the solution as:

$$\hat{\boldsymbol{v}} = \boldsymbol{A}^{\dagger}\boldsymbol{z} = (\boldsymbol{A}^T\boldsymbol{A} + \lambda^2\boldsymbol{I})^{-1}\boldsymbol{A}^T\boldsymbol{z} = \sum_{i=1}^{N_h^e} \frac{\sigma_i^2}{\sigma_i^2 + \lambda^2} \frac{\boldsymbol{u}_i\boldsymbol{z}}{\sigma_i} \boldsymbol{v}_i \tag{4.83}$$

being $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ the i-th columns of the left and right eigenvectors matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ respectively and σ_i the i-th singular value of $\boldsymbol{A}$.

In case, instead, $\boldsymbol{B} \neq \boldsymbol{I}$, then the Generalized Singular Value Decomposition (GSVD) of the matricial pair $\{\boldsymbol{A}, \boldsymbol{B}\}$ has to be performed, i.e.

$$\boldsymbol{A} = \boldsymbol{P}\boldsymbol{C}\boldsymbol{K}^{-1}; \qquad \boldsymbol{B} = \boldsymbol{Q}\boldsymbol{S}\boldsymbol{K}^{-1} \tag{4.84}$$

where $\boldsymbol{P}$ and $\boldsymbol{Q}$ are $N_h^t \times N_h^e$ and $N_h^e \times N_h^e$ dimensional (respectively); $\boldsymbol{C}$ and $\boldsymbol{S}$ are the diagonal matrices storing the singular values $\{c_i\}_{i=1}^{N_h^e}$ and $\{s_i\}_{i=1}^{N_h^e}$ of $\boldsymbol{A}$ and $\boldsymbol{B}$ respectively (in increasing order in $\boldsymbol{C}$ and in decreasing order in $\boldsymbol{S}$); $\boldsymbol{K}$ is non-singular. Also, we define the generalized singular values of the pair $\{\boldsymbol{A}, \boldsymbol{B}\}$ as:

$$\bar{\lambda}_i = \frac{c_i}{s_i} \qquad i \in \{1, \dots, N_h^e\} \tag{4.85}$$

Then, the solution to (4.82) reads as:

$$\boldsymbol{v}^* = \boldsymbol{A}^{\dagger}\boldsymbol{z} = (\boldsymbol{A}^T\boldsymbol{A} + \lambda^2\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{A}^T\boldsymbol{z} = \sum_{i=1}^{N_h^e} \boldsymbol{\Phi}_i \frac{\boldsymbol{p}_i\boldsymbol{z}}{c_i} \boldsymbol{k}_i \tag{4.86}$$

being $\boldsymbol{p}_i$ and $\boldsymbol{k}_i$ the i-th columns of the matrices $\boldsymbol{P}$ and $\boldsymbol{K}$ respectively and $\boldsymbol{\Phi} \in \mathbb{R}^{N_h^e \times N_h^e}$ the diagonal matrix containing the filter factors, i.e.

$$\boldsymbol{\Phi}_{i,i} = \frac{\bar{\lambda}_i^2}{\bar{\lambda}_i^2 + \lambda^2} \tag{4.87}$$

Incidentally, notice that small singular values of the transfer matrix $\boldsymbol{A}$ tend to result in large expansion values for the epicardial potential; thus even small perturbations in $\boldsymbol{z}$ for modes with small singular values may result in large errors in the computation of $\hat{\boldsymbol{v}}$ or $\boldsymbol{v}^*$. Such problem can be prevented by considering only a subset of relevant singular values in the SVD (GSVD), thus resorting to a Truncated SVD (Truncated GSVD); the choice of the truncation tolerance (typically defined as in (2.11)) is then another hyperparameter that sums up to the regularization one λ.

4. Generalized Minimal Residual (GMRes)

A clear drawback of the previously reported methods is that their accuracy severely depends on the choice of the regularization parameter λ; several methods to determine a sub-optimal value of such parameter have been developed, but they are often non-consistent and furthermore they are quite much sensitive to the level of noise in the data. In [73], *Ramanathan et al.* propose

to fix this problem by employing the Generalized Minimal Residual (GMRes) method, which belongs to the class of Krylov subspace iterative methods and that does not involve any regularization hyperparameter.

Given $\boldsymbol{z}$ and $\boldsymbol{A}$, the n-dimensional Krylov subspace $K(n)$ is defined as:

$$K(n) = span\left\{\boldsymbol{z}, \boldsymbol{A}\boldsymbol{z}, \boldsymbol{A}^2\boldsymbol{z}, \ldots, \boldsymbol{A}^{n-1}\boldsymbol{z}\right\} \tag{4.88}$$

At the n-th iteration, the GMRes method approximates the inverse of the transfer matrix $\boldsymbol{A}$ by its projection $p_n(\boldsymbol{A})$ onto $K(n)$, for which an orthonormal basis is computed; then the epicardial potential at iteration n is reconstructed simply as:

$$\boldsymbol{v} = p_n(\boldsymbol{A})\boldsymbol{z} \tag{4.89}$$

For the sake of precision, we should say that the GMRes method only works on square matrices and $\boldsymbol{A}$ is not square in this case; thus a pre-multiplication by $\boldsymbol{A}^T$ is necessary, so that the matrix that is ultimately considered is $\boldsymbol{A}^T\boldsymbol{A} \in \mathbb{R}^{N_h^e \times N_h^e}$.

Numerical results have shown that GMRes performs comparably to (and in some cases even better than) Tikhonov regularization; anyway the number of iterations n should be kept low, since, as it gets larger, the Krylov subspace $K(n)$ approaches the whole space and noise components start to be included in the solution. So, in the end, also the application of GMRes to the problem at hand is not hyperparameter-free, since the value of n should be chosen properly to mimimize the contamination of amplified noise components.

Notice that all the reported methods attempt at overcoming the ill-posedness of the Inverse Potential Problem of Electrocardiography by means of various regularization techniques (addition of penalty terms, singular values truncation or reduction of the number of iterations). This aspect has significantly influenced the design of the developed PDE-aware DL model, which does exploit regularization in several ways; a much more detailed explanation on this aspect is provided in Chapter 5.

Chapter 5

PDE-aware Deep Learning models for Inverse Problems in Cardiac Electrophysiology

Chapter 5 is entirely devoted to the presentation of the PDE-aware DL models introduced in Chapter 1, combining all the elements presented in Chapters 2, 3 and 4. Such models, leveraging both physical knowledge and data availability, allow to estimate the epicardial depolarization dynamics, just being given as input measurements of the electric potential in a finite set of points for a discrete number of time instants (or quantities that can be derived from such measurements). The model inputs can be of different nature, since the model has been conceived as general; in the specific context of this project, body surface potentials have been used, thus placing the work in the framework of the Inverse Potential Problem of electrocardiography. The developed model has been given the name of Space-Time Reduced Basis Deep Neural Network (ST-RB-DNN).

The Chapter is organized as follows: Section 5.1 presents the general idea of the ST-RB-DNN model from a theoretical point of view, analyzing its goal (Subsection 5.1.1) and its general structure and architecture (Subsection 5.1.2). Section 5.2 discusses instead the application of ST-RB-DNNs to the Inverse Problem of electrocardiography; more specifically Subsections 5.2.1 and 5.2.2 present the two variants of the model, implemented dealing with either the raw signals in the form of time series or with their lowest frequency Discrete Fourier Transform (DFT) coefficients. Subsection 5.2.3 presents the numerical results obtained with the aforementioned models on two benchmark test cases. In particular, a best-model is firstly identified via a grid search process involving the most relevant hyperparameters, in a simplified simulation setting; the average l^1-norm relative error on the estimated epicardial activation maps is chosen as reference metric. Once the best model has been found, its dependency with respect to other minor hyperparameters is investigated, again on the same idealized test case. Finally, the best model performances are assessed in a more realistic scenario.

5.1 ST-RB-DNN models: a Methodological Analysis

The current section is devoted to a purely methodological analysis of the proposed ST-RB-DNN model, focusing on the motivations that have driven us towards its development (Subsection 5.1.1) and on the description of its general structure (Subsection 5.1.2).

5.1.1 Motivations

As discussed in Section 4.3, the development of a method able to provide a fast and accurate solution to the Inverse Potential Problem would be remarkable. Indeed, it would allow to ap-

proximate the epicardial electric potential from non-invasively collected data, whereas nowadays it can be only measured via intrusive procedures. Anyway, the ill-posedness of the Inverse Potential Problem makes it very sensitive to noise, so that finding out a method able to efficiently solve it is far from being trivial.

In Subsection 4.3.2 we have listed some of the approaches that have been developed to tackle the problem of interest. We have limited ourselves to the main ones, despite many "optimized" variants of those have been developed as well in recent years (e.g. [68]). It is interesting to notice that there is very little contamination by Machine Learning and Deep Learning techniques, since the vast majority of the proposed methods falls in the the frameworks of Optimization and Optimal Control Theory. At this point, it deserves a mention [74] by *Wang et al.*, where the authors build a NN which estimates the (sub-)optimal values of the regularization parameters involved in the *Alternating Direction Method of Multipliers* iterative optimization algorithm, managing to improve the overall performances. Also, several DL-based algorithms aimed at the classification of ECG signals or at the identification of Myocardial Infarction areas from body surface potentials (as [24, 75] for instance) have been constructed. Anyway, to the best of our knowledge, no DL-based algorithm specifically aimed at solving the Inverse Potential Problem has ever been implemented. The ST-RB-DNN model configures as a first attempt in this direction.

A crucial point that needs to be highlighted is the fact that the model we came up with is not just a classical DL model that, leveraging data abundance, tries to learn a mapping between the input and the output. Indeed, in the context of interest, we can exploit the knowledge of the physics of the problem at hand, i.e. the fact that the propagation of the signal from the epicardial surface through the human body can be modeled as a diffusion process, within an inhomogeneous isotropic volume conductor (see Subsection 4.2.1). Thus, the ST-RB-DNN model is a physically-aware DL model, inspired by the RB-DNNs introduced in [2], which leverages the awareness of some physical laws to seek for the solution to the Inverse Potential Problem in a lower-dimensional and physically-consistent manifold. The way this is achieved will be made clear in the following of the Section.

Another important element to be underlined is that the ST-RB-DNN model takes advantage also of the temporal dynamics of the input signals to extract information that can be useful in the reconstruction of the epicardial extracellular potential. Such an aspect is totally absent in the methods presented in Subsection 4.3.2, since they all try to perform a steady "reversion" of the epicardium-to-torso diffusion process of the electric potential, simultaneously applying it at all the time instants. This configures then as an additional novelty element of this work, despite some optimized versions of Tikhonov and Truncated GSVD algorithms exploiting time dynamics in the determination of the regularization parameter of the cost functional do exist (see [64]).

A final aspect that is worth a remark about is the fact that the ST-RB-DNN has been conceived as a very general model; indeed, leveraging both data abundance and problem physics, it should learn the mapping between some signals given as input and the epicardial extracellular potential. The nature of the input signals can thus be different and the inner architecture of the model can be adapted to it; readily, a *conditio sine qua non* to ensure a good model performance is that a mapping between the input and the output does exist, i.e. that the input signals contain enough information to make it possible to reconstruct the epicardial potential from those, with a reasonable level of precision. Given this, the input signals can be either BSPMs measured via the electrodes vests commonly used in ECGI applications or standard 12-lead ECG signals or even signals measured inside the torso via catheters. In the latter case, the ST-RB-DNN model is thus adapted to solve a Generalized Inverse Potential Problem, where the input data are collected from inside the torso and not on the body surface.

In conclusion, the aim of the current work is to exploit the capabilities of DL in the framework of the Inverse Potential Problem of Electrocardiography. As a result, we came up with a model that, leveraging both data abundance and the knowledge of the problem physics, manages to

learn the mapping between signals measured in a discrete set of points on the torso surface and the epicardial extracellular potential, which is additionally assumed to belong to a lower-dimensional and physically-consistent manifold, via a projection onto a Space-Time-Reduced subspace (see Subsection 4.2.4).

5.1.2 General Model Description

The general structure of the Space-Time Reduced Basis Deep Neural Network (ST-RB-DNN) model can be visualized in Figure 5.1. Before moving to a more detailed analysis of the different parts of the model, we give a macroscopic description of it, in order to identify its key functionality principles.

Core Features

As discussed in the previous Subsection, the aim of the ST-RB-DNN model is to estimate the epicardial extracellular potential, together with the values of some relevant scalar parameters (such as the torso conducibilities, see Subsection 4.2.4), by processing input quantities related to the measurement of the electric potential in a finite set of locations on the body surface for a discrete number of time instants (red in figure). The I/O mapping is approximated by the "Trainable NN" (blue in figure), which configures as a kind of black-box, whose architecture and design severely depend on the nature of the input. In particular, the fact that this part of the network estimates the epicardial potential from a contained number of signals collected on the body surface allows to visualize it as a (deep) *decoder*.

Whichever the structure of the "Trainable NN", its last layer is always a fully-connected one, that is split into two chunks: one is responsible for the estimation of the characteristic parameters (green in figure) and the other for the prediction of the epicardial extracellular potential (orange in figure). For what concerns the parameters, their estimation is not "problematic", meaning that their number is usually contained and so is their impact on the overall network complexity. Definitely not the same can be said regarding the epicardial potential: indeed the estimation of its FOM approximation would be unfeasible, since that would mean assembling a fully-connected layer made of $N_h^e N_t$ neurons, quantity that may easily reach the order of billions (the work by *Kutyniok at al.* [6] features a detailed theoretical analysis on this, that can be eventually adapted to the case of interest). So, it appears natural to resort to the ROM techniques discussed in Subsection 4.2.4: what is actually estimated by the model are the coefficients arising from the projection of the epicardial extracellular potential onto a Space-Time-Reduced Subspace. Furthermore, a basis for such subspace is computed from the same data that are used to assemble the training dataset (as suggested in [2]), thus lightening the overall computational burden. Incidentally, the choice of employing a Space-Time MOR, rather than keeping the FOM dimensionality in time and building a model to be applied iteratively, is justified by two elements. On the one side, the iterative application of ROM-in-Space FOM-in-Time RB-CNNs has lead to not-so-good results even for the much simpler thermal block problem (see Subsection 3.2.2); on the other side, a first attempt of developing a model of such kind in the context of the Inverse Potential Problem has been made, but numerical results have been discouraging, with errors dramatically growing after just few iterations.

Once the parameters and the (Space-Time-Reduced) epicardial potential have been estimated, the model does not just return them in output; rather, a deterministic *lambda-layer* (cyan in figure) takes them as input, reconstructs the ROM-in-Space FOM-in-Time epicardial potential from the ROM-in-Space ROM-in-Time one via (4.72) and solves a RB-projected generalized Laplace equation using (4.66) and taking advantage of the affine parametrization of the RB arrays (see (4.68)). Also, the computed reduced potentials on the torso are re-projected onto the FOM space in space (at least in a subset of DOFs of interest), in order to reconstruct signals that match the ones given as input (purple in figure). In a purely deterministic way, then, the quantities estimated by the "Trainable NN" (i.e. the Decoder) are employed to reconstruct the original input signals; in this way, the second part of the ST-RB-DNN model can be seen as a

(non-trainable) *encoder*.

In principle, the ST-RB-DNN model can then made working as a deep *autoencoder*, made by a deep decoder followed by a non-trainable encoder; indeed it takes as input signals that can be measured in a simple and non-invasive way and it produces as output an estimate of that same signals. The output is assembled by solving the ROM-in-Space FOM-in-Time Forward Problem of Electrocardiography, upon having estimated the epicardial extracellular potential (projected onto a Space-Time-Reduced subspace) and some scalar parameters that may influence the characteristics of the signals. In actual practice, as it will be discussed in Subsection 5.2.3, involving only the reconstructed signals in the expression of the loss function results in a not precise estimate of the epicardial potential; thus we have decided to penalize also the Space-Time-Reduced epicardial potential and the parameters (with suitable weights to be tuned). In this way, the RB-solver layer configures more as a physically-aware regularization agent, which drives the predicted solutions to belong to a lower-dimensional and physically-consistent manifold. Thus, the expression of the loss functional is as follows:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\Theta}) =& \mathcal{L}_{sig}(\boldsymbol{\Theta}) + \mathcal{L}_{BC}(\boldsymbol{\Theta}) + \mathcal{L}_{\boldsymbol{\mu}}(\boldsymbol{\Theta}) \\ =& w_{sig} \sum_{i=1}^{n_{sig}} MSE\big(\boldsymbol{S}^r_{\#i}, \boldsymbol{S}^a_{\#i}(\boldsymbol{\Theta})\big) \; + w_{BC} \; MAE_{\boldsymbol{\sigma}}\big({\boldsymbol{u}_{\boldsymbol{e}ST}}^r, {\boldsymbol{u}_{\boldsymbol{e}ST}}^a(\boldsymbol{\Theta})\big) \; + w_{\boldsymbol{\mu}} \; MSE\big(\boldsymbol{\mu}^r, \boldsymbol{\mu}^a(\boldsymbol{\Theta})\big)\end{aligned} \tag{5.1}$$

The quantities with apex ^{r} denote target values, the quantities with apex ^{a} denote approximated values, $\boldsymbol{\Theta}$ represents the vector of the NN trainable parameters, $\boldsymbol{S}_{\#i}$ encodes the i^{th} input signal, from a total of n_{sig} signals, and w_{sig}, w_{BC} and $w_{\boldsymbol{\mu}}$ represents the loss weights of the signals, the Space-Time-Reduced epicardial potential and the characteristic parameters respectively. The choice of using the Mean Absolute Error (MAE) instead of the MSE for the loss term involving the Space-Time-Reduced epicardial potential aims at forcing to 0 the least relevant coefficients and it is justified by numerical experiments (see Subsection 5.2.3). Additionally, such MAE is not only weighted by the non-negative scalar w_{BC}, but also by the singular values associated to the different coefficients, so that:

$$MAE_{\boldsymbol{\sigma}}\big({\boldsymbol{u}_{\boldsymbol{e}ST}}^r, {\boldsymbol{u}_{\boldsymbol{e}ST}}^a(\boldsymbol{\Theta})\big) =: \frac{1}{b_s}\frac{1}{n_{st}} \sum_{i=1}^{b_s}\sum_{j=1}^{n_h^e}\sum_{k=1}^{n_t^j} \sqrt{\frac{\sigma_j^{e,s}\sigma_k^{e,t,j}}{\sigma_1^{e,s}\sigma_1^{e,t,1}}} \left|{\boldsymbol{u}_{\boldsymbol{e}ST}}^r{}_{\mathcal{F}(j,k)} - {\boldsymbol{u}_{\boldsymbol{e}ST}}^a{}_{\mathcal{F}(j,k)}\right| \tag{5.2}$$

where $\{\sigma_j^{e,s}\}_{j=1}^{n_h^e}$ are the singular values arising from the spatial POD applied to the epicardial potential tensor, $\{\sigma_k^{e,t,j}\}_{k=1}^{n_t^j}$ are the singular values coming from the temporal POD applied to the projection of the epicardial potential tensor onto the 1-dimensional subspace spanned by the j-th spatial basis element, b_s is the training batch size and $\mathcal{F}(\cdot,\cdot)$ is defined as in (2.36). In this way, the weight of the error on a Space-Time-Reduced epicardial potential coefficient in the loss gets lower and lower as its relevance decreases; this allows to obtain better estimates of the most relevant coefficients, easing the training process and ultimately improving the model performances. The choice of taking the square root of the singular values (rather than their plain values or their squares) is justified by numerical results.

In the end, two elements contribute to the physical awareness of the ST-RB-DNN model:

1. The estimation of the projection of the epicardial extracellular potential onto a Space-Time-Reduced subspace, which is generated (via PODs) from a tensor storing various solutions to the heart EP problem and which is also employed as training dataset. In this way, indeed, the estimated epicardial potential is forced to belong to a lower-dimensional manifold, which should configure as the one of physically-consistent solutions. This last consideration assumes the training dataset to be made of realistic (if not real) data and to be enough heterogeneous to represent the majority of the dynamics of the problem of interest.
2. The reconstruction in output of the signals given in input, achieved by solving the ROM-in-Space FOM-in-Time Forward Problem of Electrocardiography, within a deterministic

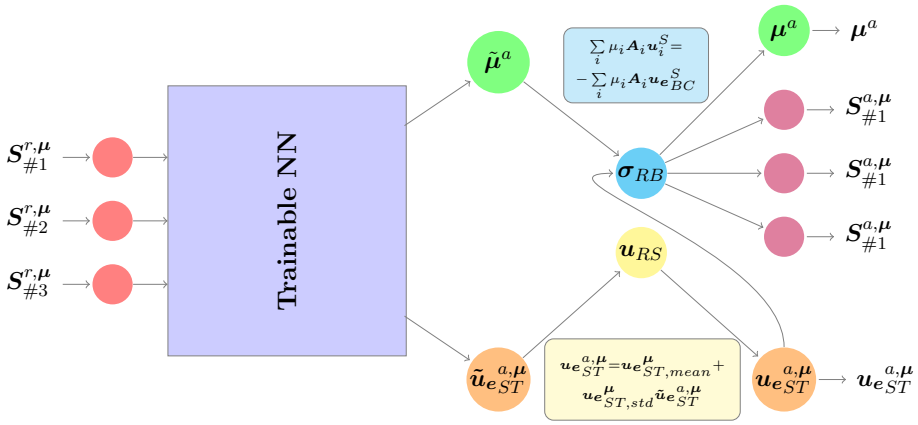


Figure 5.1: Scheme of the general architecture of the ST-RB-DNN model; the Input Layer is in red and it contains the signals to be processed; the trainable part of the NN is in blue and its actual architecture depends on the proper nature of the input signals; the parameter estimator layer is in green and it contains the normalized estimated values of some relevant scalar parameters; the epicardial extracellular potential estimator layer is in orange and it contains the values of the coefficients arising from the projection of the epicardial extracellular potential onto the dimensionality reduced subspace in both Space and Time dimensions; the epicardial extracellular potential rescaler layer is in yellow and it allows to rescale the quantities estimated by the first part of the NN to actually obtain the desired coefficients; the RB-solver layer is in cyan and it allows to make the NN aware of the physics of the phenomenon of interest, by solving a reduced-order problem that reconstructs the signals given as input; the signals reconstructor layer is in purple and it simply hosts the values of the signals reconstructed by the RB-solver layer, configuring as the autoencoding portion of the output

layer, from the estimated epicardial potential. Indeed, by involving in the loss a term which penalizes the signals' reconstruction (see (5.1)), the model is driven to estimate epicardial potentials which determine the onset of the desired signals. This further shrinks the space of admissible solutions to a lower-dimensional and physically-aware manifold.

Additional Details

In the following, additional details that appear to be relevant in the design of the ST-RB-DNN model are discussed. In particular, each part of the model is analyzed separately, following the color-coded notation of Figure 5.1.

- **Input Layer** (in red): the input layer contains the quantities to be processed by the network. In the general framework of inverse problems in cardiac EP, those are related to the measurement of the electric potential in a finite set of points for a discrete number of time instants. It is important here to clarify that the input of the NN does not have to be compulsorily made of the raw signals measured by the electrodes or by trivial linear combinations between those (as ECG signals, for instance). Rather, the training of the model may be eased if its input is made of quantities that are able to provide a better, richer and more robust encoding of all the information that are ultimately necessary to carry out the task at hand. In the field of ECGI, for instance, we may cite two different works [75, 76] in which DL models able to classify 12-lead ECG signals have been developed. They both aimed at performing the same task, but in [75] the logarithmic spectrum (upon application of a DFT) of the signals is processed by the network, while in [76] the NN input is derived from the raw signals, just by extracting the heartbeats and applying baseline normalization and band-pass filtering operations to reduce the effect of noise. In mathematical terms, the single input datapoint is denoted as $\boldsymbol{x}_{in} \in \mathbb{R}^{n_{sig} \times M}$, where M could represent either the number of time instants or the number of coefficients extracted by performing some transform on the original signal. The overall training tensor is then defined by $\mathcal{X}_{in}^{train} \in \mathbb{R}^{N_{train} \times n_{sig} \times M}$, while the test tensor analogously writes as $\mathcal{X}_{in}^{test} \in \mathbb{R}^{N_{test} \times n_{sig} \times M}$.

- **Trainable NN** (in blue): the input is then processed by a set of trainable layers, that as whole constitute the "trainable part" of the ST-RB-DNN model. This part of the network acts as a data-driven decoder and its architecture highly depends on the nature of the input. For instance, if the input is made of data organized in the form of time series, then it is common to insert, aside of classical fully-connected layers, also $1D$-convolutional layers that convolve the data along their temporal dimension, with the aim of extracting features related to the time-dynamics of the problem at hand (see Subsection 3.2.2). Conversely, if the input layer is assembled by applying suitable transforms to the original data that already take into account the time dynamics (as the Fourier Transform, for instance), then temporal convolutions are no longer useful and the trainable part of the model can consist just of a simple MLP.

  Two further aspects deserve a mention and they both go in the direction of preventing overfitting:

  1. In between fully-connected, convolutional or pooling layers, we have inserted also the so-called *dropout* layers, that allow to ignore a fraction p of network nodes at each forward and backward pass during the training process. This may be useful since neurons tend to develop some sort co-dependency between each other during the learning phase, which curbs their individual power and that easily leads to overfitting. By randomly excluding a portion of neurons (that changes at every pass), this undesired side effect is significantly reduced. Furthermore, the complexity of the NN to be trained is decreased by a fraction p; this leads to training epochs that are on the one side faster but on the other side less effective, so that a higher number of those is needed in order to reach convergence.
  2. A classical ML technique to reduce overfitting is to add to the loss function regularization terms, which are given by some norm of the vector storing the trainable parameters, pre-multiplied by a suitable regularization coefficient. Common choices on the type of norm are the (squared) l^2-norm (*Ridge Regularization* or *zero-order Tikhonov regularization*, see [69]) and the l^1-norm (*Lasso Regularization*, see [77]). The former is the most widely used, since its quadratic expression makes it suitable for optimizations tasks; the latter, instead, is typically adopted in contexts characterized by a high degree of sparsity, since it tends to shrink to 0 all the coefficients that appear to be non-relevant for the task at hand, while keeping the values of the others almost unchanged. In the case of ST-RB-DNN models, we have applied Ridge Regularization to all the weights and biases, thus ending up with a loss functional of the form
  $$\mathcal{L}(\boldsymbol{\Theta}) = \mathcal{L}_{sig}(\boldsymbol{\Theta}) + \mathcal{L}_{BC}(\boldsymbol{\Theta}) + \mathcal{L}_{\boldsymbol{\mu}}(\boldsymbol{\Theta}) + \lambda_r ||\boldsymbol{\Theta}||_2^2 \tag{5.3}$$
  with all the quantities defined as in (5.1) and $\lambda_r \in \mathbb{R}^+$ denoting the regularization coefficient, which configures an a hyperparameter of the model.

  In principle, any architecture able to handle the given input tensor can be put in place in this part; smart possibilities in the field of ECGI may be offered by the *ResNet* employed in [76] or by the Convolutional Recurrent Neural Network (CRNN) used in [75]. Whichever the architectural choices, the last layer of this black-box is always a fully-connected one, split in two separated chunks; such layer is responsible for the estimation of the physical quantities of interest i.e. the characteristic parameters and the Space-Time-Reduced epicardial potential.

- **Parameters' Estimator Layer** (in green): the parameters' estimator layer is one of the two chunks in which the last fully-connected layer of the Trainable NN is split into. As its name suggests, it is responsible for the estimation of the scalar parameters that happen to be relevant in reconstructing the original input signals, having knowledge of the epicardial extracellular potential. In the context of inverse problems in cardiac EP, the signals' reconstruction is carried out by solving a generalized Laplace equation, featuring the epicardial

potential as "inner" boundary condition and a portion of the human body surrounding the heart as computational domain. So the relevant parameters to be estimated are the electric conducibilities in the different regions of the body (see (4.66)).

Two aspects have to be underlined:

1. Whichever the way the training snapshots are computed, within the NN the potential in the body is recovered by solving a decoupled generalized Laplace equation, where the epicardial potential plays the role of Dirichlet boundary datum. Thus, in terms of conducibilities, what matters are their relative values and not their absolute ones. If the human body is partitioned into N_p^t parts, characterized by different conducibility values $\{\sigma_p^t\}_{p=1}^{N_p^t}$, then only $N_p^t - 1$ values have to be estimated, since one can be always normalized to a fixed value $\bar{\sigma} > 0$, typically chosen equal to 1. Readily, if the signal transmission in the body (or in a portion of it) is perfectly isotropic, then $N_p^t = 1$ and no parameters' estimation is needed.
2. In [2], the authors state that the performances of the implemented RB-DNN model are significantly better if the parameter values are normalized in the interval $[0; 1]$, by means of a simple *Min-Max* scaler. Indeed, the normalization allows to to use a sigmoid as activation function of the estimator layer (see (2.2)) and this happens to ease the model training. Borrowing this observation, we let the parameters' estimator layer compute the values $\tilde{\boldsymbol{\mu}}$ such that:

$$\boldsymbol{\mu} = \boldsymbol{\mu}_{min} + (\boldsymbol{\mu}_{max} - \boldsymbol{\mu}_{min})\tilde{\boldsymbol{\mu}} \tag{5.4}$$

with $\boldsymbol{\mu}_{min}$ and $\boldsymbol{\mu}_{max}$ being the vectors storing the lower and upper bounds for each of the $N_p^t - 1$ scalar parameters, respectively. The rescaling to the "original" values via (5.4) is done in the RB-solver layer, so that the target parameters, in their proper ranges, are returned in output.

- **Epicardial Potential Estimator Layer** (in orange): the second chunk of the last fully-connected layer of the "Trainable NN" part is responsible for the estimation of the epicardial potential.

As anticipated before, what is actually estimated is not the FOM approximation of such potential, since that would imply putting in place a gigantic $N_h^e N_t$-dimensional layer, but rather its projection onto a Space-Time-Reduced subspace. In this way, the number of quantities to be estimated reduces to $n_{st} \ll N_h^e N_t$, entailing in turn a significant lightening of the overall model complexity. Anyway, also performing a direct estimation of the coefficients arising from the spatio-temporal projection features a significant drawback: indeed, their values span over a broad range of orders of magnitude, which gets broader and broader as the values of the POD tolerances $\epsilon_{POD}^{e,s}$ and $\hat{\epsilon}_{POD}^{e,t}$ get lower. This issue could be fixed by resorting to an affine rescaling process, which is performed in the Rescaler Layer (yellow in Figure 5.1); so, the model estimates $\tilde{\boldsymbol{u}}_{\boldsymbol{e}_{ST}}^{a,\boldsymbol{\mu}}$, which relates to the actual value of the Space-Time-Reduced epicardial potential $\boldsymbol{u}_{\boldsymbol{e}_{ST}}^{a,\boldsymbol{\mu}}$ as follows:

$$\boldsymbol{u}_{\boldsymbol{e}_{ST}}^{a,\boldsymbol{\mu}} = \boldsymbol{u}_{\boldsymbol{e}_{ST}}^{mean} + \boldsymbol{u}_{\boldsymbol{e}_{ST}}^{std} \tilde{\boldsymbol{u}}_{\boldsymbol{e}_{ST}}^{a,\boldsymbol{\mu}} \tag{5.5}$$

with $\boldsymbol{u}_{\boldsymbol{e}_{ST}}^{mean}$ and $\boldsymbol{u}_{\boldsymbol{e}_{ST}}^{std}$ representing the additive and the multiplicative terms of the affine rescaler, which can be thought as a mean and as a standard deviation respectively. In simpler terms, the NN estimates how far (in terms of the epicardial potential) the current datapoint is from a target value, referred to as the "mean", with a scaling factor given by what is referred to as the "standard deviation". Several choices can be made for the values of these two quantities; for instance:

1. $\boldsymbol{u}_{\boldsymbol{e}_{ST}}^{mean}$ and $\boldsymbol{u}_{\boldsymbol{e}_{ST}}^{std}$ can be put equal to the mean and the standard deviation of the epicardial potentials in the training dataset.
2. $\boldsymbol{u}_{\boldsymbol{e}_{ST}}^{mean}$ can be set to $\mathbf{0}$ (totally inactivated heart) and $\boldsymbol{u}_{\boldsymbol{e}_{ST}}^{std}$ can be computed as the standard deviation of the epicardial potentials in the training dataset, assuming them to have $\mathbf{0}$ mean.

3. $\boldsymbol{u_e}_{ST}^{mean}$ can be selected to be equal to the Space-Time projection of the epicardial potential in a physiologically-activated heart and $\boldsymbol{u_e}_{ST}^{std}$ can be then computed as the standard deviation of the epicardial potentials in the training dataset, assuming them to have $\boldsymbol{u_e}_{ST}^{mean}$ as mean.

Other choices are obviously possible and they configure as another hyperparameter of the model. In any case, this strategy allows to homogenize the orders of magnitude of the quantities to be estimated by the model and this appeared to ease a lot the training process, preventing to get stuck in some local minima of the loss function in the parameters' space, after few epochs.

Despite having reduced significantly the complexity of this layer by resorting to spatio-temporal ROM techniques, still its dimensionality happens to be non-negligible. Thus, aside of the affine rescaling, other elements should be put in place to improve the model performances and the following considerations are worth a mention.

1. High-dimensional layers (as well as very deep networks) tend to be subject to the *vanishing gradient* problem, which consists in the gradient of the loss function with respect to the trainable parameters (or to a portion of those) to be so small that no significant improvements are achieved at optimization stage. The interested reader may refer to [78, 79] for more detailed information on the topic. Such a problem is very common with activation functions as the sigmoid (see (2.2)) or the hyperbolic tangent $Tanh$, whose derivatives tend to "saturate" if the absolute value of the input is too big; using $ReLU$ as activation function significantly prevents this kind of issue, since in such case the gradient saturates only in one direction. Anyway, in very big layers, especially when the weights and biases are initialized in a non-optimal way and/or the learning rate is set too high, also $ReLU$ activation function may fail (this is the so-called "Dying $ReLU$" problem), causing neurons to get completely stuck in a perpetually inactive state (see [80]). In such cases, it is possible to resort to optimized versions of $ReLU$, that try to circumvent this issue; in particular the most widely used functions are *Leaky-ReLu*

$$Leaky - Relu(x) = \begin{cases} x & if\ x > 0 \\ ax & if\ x \leq 0 \end{cases} \tag{5.6}$$

with $a \in \mathbb{R}$ typically chosen of the order of 10^{-1}, and *Exponential Linear Unit (ELU)*

$$ELU(x) = \begin{cases} x & if\ x > 0 \\ a(e^x - 1) & if\ x \leq 0 \end{cases} \tag{5.7}$$

with $a \in \mathbb{R}^+$ typically chosen of the order of 1. Figure 5.2 shows the plots of these functions; the scaling parameters have been set to the default values used by the *Tensorflow/Keras Python* package. Also, other workarounds to prevent the vanishing gradient problem are possible, as the usage of $ResNets$ or the choice of different weight initialization schemes; regarding the latter, numerical experiments have lead us to the choice of a random uniform weights initialization in the interval $[-0.05; 0.05]$, while we have not performed any kind of investigation on $ResNets$.
2. What matters in terms of network complexity is not only the dimensionality of the layer that predicts the Space-Time-Reduced epicardial potential, but also the one of the preceding layer: indeed the weights matrix has dimension $N_{pre} \times n_{st}$, being N_{pre} the number of neurons in such layer. Thus, a significant reduction of the complexity can be achieved either by placing a low number of nodes in the last-but-one layer of the "Trainable NN" or by inserting a $1D$ pooling layer between the last two fully-connected layers of such part. In the case of the ST-RB-DNN model, we have chosen the second alternative, inserting a $1D$ *max-pooling* layer, with pooling window of dimension 4.

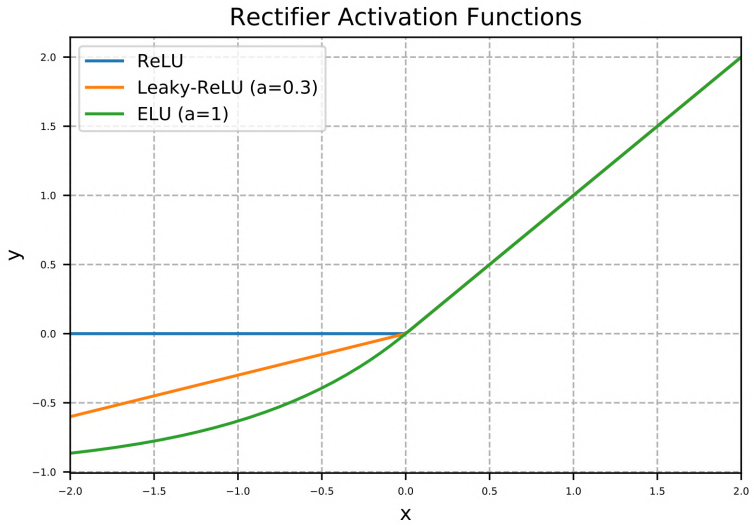


Figure 5.2: Plots of the rectifier activation functions ($ReLU$, $Leaky-ReLU$ and ELU)

Finally, the rescaled values of the Space-Time-Reduced epicardial potential are inserted into the output tensor $\mathcal{Y}_{out}^{train}$ by the Rescaler Layer.

- **RB-solver Layer** (in cyan): the RB-solver layer is a deterministic layer that is responsible for the reconstruction of the signals given as input to the model, starting from the values of the characteristic parameters and of the Space-Time-Reduced epicardial potential. Its presence is inspired by the RB-DNN models introduced in Chapter 3 and it acts as a sort of physically-aware regularization agent that prevents physically-meaningless solutions to be predicted.

  More in detail, the RB-solver layer assembles two out of the three portions of the output, namely:
  - The values of the characteristic parameters, which are rescaled to their proper ranges via (5.4).
  - The reconstruction of the signals, originally given as input to the NN, from the estimated values of the epicardial potential and of the characteristic parameters.

  While the first task is trivial, the second one is slightly more complicated, so it is worth describing how it is carried out more in detail. The following steps are performed:
  1. At each forward pass during the training, the RB-solver layer takes as input the normalized values of the parameters $\tilde{\boldsymbol{\mu}}^a \in \mathbb{R}^{bs \times N_p^t - 1}$ and the estimated rescaled Space-Time-Reduced epicardial potential $\boldsymbol{u_e}_{ST}^{a,\boldsymbol{\mu}} \in \mathbb{R}^{bs \times n_{st} \times 1}$. Here bs denotes the batch size used by the SGD algorithm. At testing stage, bs is substituted by N_{test}, dimensionality of the test dataset.
  2. The characteristic parameters are rescaled to their proper ranges via (5.4) and inserted in the output tensor.
  3. Taking advantage of the affine parametrization of the RB affine arrays (see (4.68)), the left-hand side and right-hand side stiffness matrices of (4.66) are assembled. Being both parameter-dependent, their dimensionalities are $bs \times n_h^t \times n_h^t$ and $bs \times n_h^t \times n_h^e$ respectively, where n_h^t and n_h^e are the dimensions of the Reduced Bases in space for the potential at the epicardium and at the torso respectively.
  4. The Space-Time-Reduced epicardial potentials $\boldsymbol{u_e}_{ST}^{a,\boldsymbol{\mu}} \in \mathbb{R}^{b_s \times n_{st} \times 1}$ are expanded along the temporal dimension only. Such operation has to be performed using tensorial computations, in order to be compliant with the algorithms used for the training. So,

two tensors have to be defined; on the one side, there must be a $3D$ tensor storing the Reduced Basis in time for the epicardial potential, i.e. $\mathcal{V}^e_{time} \in \mathbb{R}^{1\times n_{st}\times N_t}$ such that

$$\mathcal{V}^e_{time}[1,\cdot,\cdot] = \left[\begin{array}{ccc|c|ccc} \boldsymbol{\psi}^1_1 & \cdots & \boldsymbol{\psi}^{\mathbf{1}}_{\boldsymbol{n_t^1}} & \cdots & \boldsymbol{\psi}^{n^e_h}_1 & \cdots & \boldsymbol{\psi}^{n^e_h}_{n^{n^e_h}_t} \end{array}\right]^T \tag{5.8}$$

On the other side, a logical tensor allowing to sum up all the quantities related to the same spatial basis element must be defined; we call it $\mathcal{T}_{select} \in \mathbb{R}^{1\times n^e_h\times n_{st}}$ and its structure is as follows:

$$\mathcal{T}_{select}[1,\cdot,\cdot] = \begin{bmatrix} \mathbf{1}_{n^1_t} & \cdots & \cdots & \cdots \\ \cdots & \mathbf{1}_{n^2_t} & \cdots & \cdots \\ \vdots & \ddots & \ddots & \vdots \\ \cdots & \cdots & \cdots & \mathbf{1}_{n^{n^e_h}_t} \end{bmatrix} \tag{5.9}$$

being $\mathbf{1}_N$ a row vector of ones, of dimension N. The ROM-in-space FOM-in-time epicardial potential $\boldsymbol{u_e}^{a,\boldsymbol{\mu}}_S \in \mathbb{R}^{bs\times n^e_h\times N_t}$ is then recovered as:

$$\boldsymbol{u_e}^{a,\boldsymbol{\mu}}_S = \mathcal{T}_{select}\big(\boldsymbol{u_e}^{a,\boldsymbol{\mu}}_{ST}\cdot\mathcal{V}^e_{time}\big) \tag{5.10}$$

where $\cdot$ denotes the element-wise multiplication. Notice that the expansions of $\mathcal{V}^e_{time}$ and $\mathcal{T}_{select}$ in their first dimensions allow to compute all the desired potentials, with only one tensorial operation.

5. The ROM generalized Laplace equation (4.66) is solved for bsN_t times (i.e. for all the bs datapoints in the batch, for all the N_t time instants), so that the ROM-in-space FOM-in-time torso potential $\boldsymbol{u_t}^{a,\boldsymbol{\mu}}_S \in \mathbb{R}^{bs\times n^t_h\times N_t}$ is computed.
6. The last passage consists in the computation of signals that are analogous to the ones received as input by the NN and in their insertion inside the output layer (purple in Figure 5.1). The way this task is carried out readily depends on the nature of the input signals. A common passage is anyway given by the re-projection of the ROM-in-Space FOM-in-Time torso potential $\boldsymbol{u_t}^{a,\boldsymbol{\mu}}_S \in \mathbb{R}^{b_s\times n^t_h\times N_t}$ onto the FOM space in both Space and Time, at least in a subset of DOFs of interest. Calling $\mathcal{V}^t_{space} \in \mathbb{R}^{1\times N_{DOFs}\times n^t_h}$ the tensor storing the n^t_h elements of the Reduced Basis in space for the torso potential, evaluated at the desired N^t_{DOFs} DOFs, the desired potentials $\boldsymbol{u_t}^{a,\boldsymbol{\mu}} \in \mathbb{R}^{bs\times N^t_{DOFs}\times N_t}$ are computed as:

$$\boldsymbol{u_t}^{a,\boldsymbol{\mu}} = \mathcal{V}^t_{space}\boldsymbol{u_t}^{a,\boldsymbol{\mu}}_S \tag{5.11}$$

Notice that the expansion of $\mathcal{V}^t_{space}$ in its first dimension allows to compute all the desired potentials with only one tensorial operation.

- **Output layer and Loss**: the final output tensor, at each forward/backward pass during training, is stored in the tensor $\mathcal{Y}^{train}_{out} \in \mathbb{R}^{bs\times[(N^t_p-1)+(n_{sig}M)+(n_{st})]}$. It stores:
  - The $N^t_p - 1$ characteristic parameters, rescaled in their proper ranges via (5.4)
  - The n_{st} coefficients encoding the Space-Time Reduced epicardial potential
  - The reconstruction of the signals given as input, of dimension $n_{sig}M$, being M the dimensionality of a single signal.

The loss is constructed as in (5.1) - (5.3), with w_{sig} being normalized to 1 (unless it is set to 0); the choices of the optimization algorithm and of the learning rate configure as two key hyperparameters. As a baseline, the Adam optimizer (i.e. an optimized adaptive version of the SGD algorithm introduced in [14]) with a learning rate of 10^{-3} is used.

Remark: as the ST-RB-DNN model has been conceived as general, also the nature of its output can be modified and adapted to the case of interest. For instance, if one is interested in localizing the LEA areas of the left ventricle, he could include in the loss the FOM-in-Space ROM-in-Time epicardial potential, evaluated in a subset of sampled locations on

the left ventricle, instead of the ROM-in-Space ROM-in-Time one. Readily, this can be done only if suitable tensorial algorithms, able to turn the ROM-in-Space ROM-in-Time epicardial potential into the desired output, are implemented, either in the RB-solver layer or in the Rescaler layer.

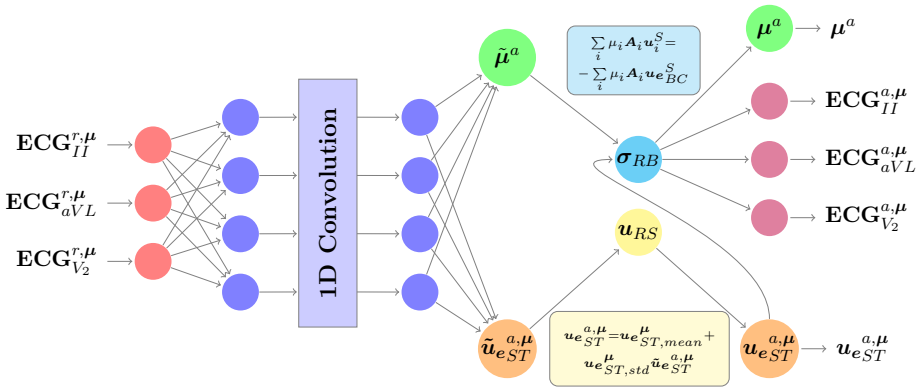


Figure 5.3: Scheme of the architecture of the ST-RB-DNN model with input signals organized as time series; the Input Layer is in red and it contains the time-series signals to be processed; the trainable part of the NN is in blue and its architecture is made of two sets of fully-connected layers, separated by some $1D$ convolutional layers; the parameter estimator layer is in green and it contains the normalized estimated values of some relevant scalar parameters; the epicardial extracellular potential estimator layer is in orange and it contains the values of the coefficients arising from the projection of the epicardial extracellular potential onto the dimensionality reduced subspace in both Space and Time dimensions; the epicardial extracellular potential rescaler layer is in yellow and it allows to rescale the quantities estimated by the first part of the NN, to actually obtain the desired coefficients; the RB-solver layer is in cyan and it allows to make the NN aware of the physics of the phenomenon of interest, by solving a dimensionality reduced problem that reconstructs the signals given as input; the signals reconstructor layer is in purple and it simply hosts the values of the signals reconstructed by the RB-solver layer, configuring as the autoencoding portion of the output

5.2 ST-RB-DNN models: Application to the Inverse Problem of Electrocardiography

This Section is devoted to the presentation of the ST-RB-DNN models that have been actually implemented and tested in the context of this thesis. In particular, we focused on the Inverse Problem of electrocardiography (see Section 4.3), thus trying to reconstruct the epicardial extracellular potential during ventricular depolarization, just being given as input measurements of the body surface potential in a discrete set of points. Two different models have been developed at this aim. The first one takes as input signals organized in the form of time series and it leverages temporal convolution to extract from those useful and representative features (Subsection 5.2.1); the second one, instead, processes the lowest-frequency coefficients arising from the application of a DFT to the original signals (Subsection 5.2.2). After having described the two models, highlighting their main specifics with respect to the general ST-RB-DNN model described in Subsection 5.1.2, numerical results achieved on two benchmark test cases (an idealized one and a slightly more realistic one) are presented (Subsection 5.2.3).

5.2.1 The Time-Series-Based ST-RB-DNN Model

The first ST-RB-DNN model we implemented is characterized by taking as input body surface potentials organized in the form of time series (as standard 12-lead ECG signals, for instance); its structure can be visualized in Figure 5.3.

The main specifics with respect to the general ST-RB-DNN model described in the previous Section are listed in the following.

- **Input Layer** (in red): the input is made of body surface potentials, or of linear combinations between those (as ECG signals), evaluated at some discrete time instants within a fixed observation window.

  The processing of body surface potentials is made difficult by the fact that they feature

high Signal-to-Noise Ratios (SNRs) (see [81, 82]). In the context of this project we only dealt with numerically simulated signals, to which we have added a noise contribution, in order to mimic a more "realistic" scenario. In particular, we added to all signals correlated white Gaussian noise, constructed in order to get an average SNR of $\approx 19\ dB$.

As suggested in [75, 76], at pre-processing stage it is useful to remove the noise from the signals as much as possible, since it eases the training process and it increases the representative power of DL models. Many works regarding the denoising of body surface potentials (and especially of ECGs) are available, as [81, 82] for instance. Since we did not handle real signals, just facing up to numerically simulated ones with artificially superimposed Gaussian noise, we did not actually have the need of resorting to such complicated filtering techniques; rather, we employed a simple low-pass Butterworth filter of order $n = 3$ (see [83]), whose gain function $G(\cdot)$ is given by

$$G^2(\omega) = |H(j\omega)|^2 = \frac{G_0^2}{1 + \left(\frac{j\omega}{j\omega_c}\right)^{2n}} \tag{5.12}$$

being ω the angular frequency (in $rad\ s^{-1}$), $H(\cdot)$ the filter's transfer function, n the order of the filter, ω_c the cutoff frequency and G_0 the DC gain. This allowed us to get rid of all the high-frequency noise and to improve the performances of the network. Readily, in realistic scenarios where real signals have to be processed, State-Of-Art denoising techniques should be put in place to ameliorate the model behavior.

Additionally, the signals given as input have been normalized in the interval $[-1; 1]$; this is a quite standard strategy in DL applications, since it happens to ease the training process. More specifically, n_{sig} different normalization constants have been computed as

$$\bar{\boldsymbol{S}} = \{\bar{s}_i\}_{i=1}^{n_{sig}} \quad such\ that \quad \bar{s}_i = 1.1||vec(\boldsymbol{S}_{\#i}^{train})||_\infty \tag{5.13}$$

being $vec(\boldsymbol{S}_{\#i}^{train}) \in \mathbb{R}^{N_t N_{train}}$ the vectorization of the matrix storing the i-th signal, for all the N_{train} training datapoints at all the N_t time instants.

- **Trainable NN**(in blue): several choices can be made concerning the architecture of the "Trainable NN". For instance, a *Resnet* similar to the one employed in [76] to perform ECG classification could be put in place, having it proved to be able to extract relevant features from complex and noisy input data, using a small number of hyperparameters and taking extreme advantage of temporal convolutions. This configures as a possible extension of the current project, but no efforts have been made in this direction.

  Conversely, the architecture chosen for the Trainable NN is similar to the one employed in [36], in the context of iterative RB-CNN models for unsteady parametrized PDEs. So, two sets of fully-connected layers are separated by $1D$ convolutional layers, that convolve the input signals over time, coupled with max-pooling layers. The roles of the three sets of layers are the following:

  - *Pre-Convolutional Fully-Connected Layers*: before the input data are fed to the $1D$ Convolutional Layers, they are processed by some fully-connected layers; their aim is to extract features on top of which the convolutional operations may exhibit better performances. It is relevant to notice that these layers only combine quantities related to the same time instant, using the same set of weights at all the time instants. In simpler terms, the original input signals are combined together in a non-linear way, so that signals characterized by a better encoding of the information that are relevant for the ultimate task at hand can be derived. These layers are intended not to weigh down the model complexity; for this reason either a single layer with no more than 128 neurons or two layers with no more than 64 neurons each have been considered. The activation function is *ReLU* for all neurons and Ridge Regularization is employed.

- *1D Convolutional Layers + Max-Pooling*: after the initial "pre-processing" stage, the resulting data are fed to $1D$ convolutional layers, that convolve them along the temporal dimension. The number of layers and the dimensionality of the convolutional kernel are two key hyperparameters of the model. Another important hyperparameter is given by the number of convolutional filters; indeed, as discussed in Subsection 2.1.2, the representative power of CNNs can be increased by training multiple convolutional kernels. In this way, different type of filters can be applied to the time-series data given as input, extracting a higher and hopefully richer amount of features. Also, each $1D$-convolutional layer is followed by a *Max-Pooling* layer, with pooling window of dimension 2×3 and with 2×2 stride; this means that, when passing from one pooling window to another, two steps are performed both along the temporal dimension and along the one of the different signals. As a result, with proper padding choices, the dimensionality of the input is halved along both dimensions at each loop, reducing in turn the number of trainable parameters of the model. Finally, all convolutional kernels are subject to Ridge Regularization, to prevent overfitting.

- *Post-Convolutional Fully-Connected Layers*: after the data have been convolved over time and subsampled via max-pooling operations, they are finally processed by a second set of fully-connected layers. Their aim is to extract relevant features, that allow to ultimately get a better estimation of both the $N_p^t - 1$ characteristic parameters and of the n_{st} "normalized" Space-Time-Reduced coefficients of the epicardial extracellular potential. Since the data given as input to these layers are not so high-dimensional (thanks to the *Max-Pooling* operations), it is possible to equip them with a higher number of neurons, without significantly impacting the model complexity. In our tests, we employed 3 to 7 layers, featuring a number of nodes that decreases exponentially from 2^{n+2} to 2^3, being n the number of layers. All neurons of all layers are $ReLU$-activated and Ridge Regularization is employed.

Additionally, *dropout* layers with $p = 0.20$ are placed both between the three aforementioned sets of layers and between each couple of convolutional and max-pooling layers. Also, recall that the last layer of the Trainable NN is always a fully-connected layer of dimension $(N_p^t - 1) + n_{st}$, which is responsible for the estimation of the characteristic parameters and of the "normalized" Space-Time-Reduced coefficients of the epicardial extracellular potential. The two estimator layers (green and orange in Figure 5.3) are analogous to the ones described for the general ST-RB-DNN model in Subsection 5.1.2.

- **RB-solver Layer** (in cyan): the RB-solver layer is equivalent to the one described in the previous Section, concerning the general ST-RB-DNN model. The only *caveat* is related to the output. Indeed, if the signals to be returned in output coincide with the values of the body surface potential at some DOFs, then nothing has to be changed. Conversely, it may happen that:
  - The output values do not coincide with the potential got in a subset of the mesh DOFs, either because the electrodes are located in places that do not host a mesh node or because the chosen FOM basis is not interpolatory. In such cases, spatial interpolation has to be performed.
  - The target signals are computed as combinations of the body surface potentials evaluated in a subset of the mesh DOFs, as in the case of ECG signals employing an interpolatory FOM basis (see Subsection 4.1.2). In such cases, the obtained signals have to be suitably combined in order to reconstruct the desired ones.

- **Output Layer and Loss**: the output and the loss function are analogous to the ones described for the general case. The only remark is that the value of M, representing the dimensionality of the signals, in this case equals N_t, i.e. the number of FOM temporal DOFs.

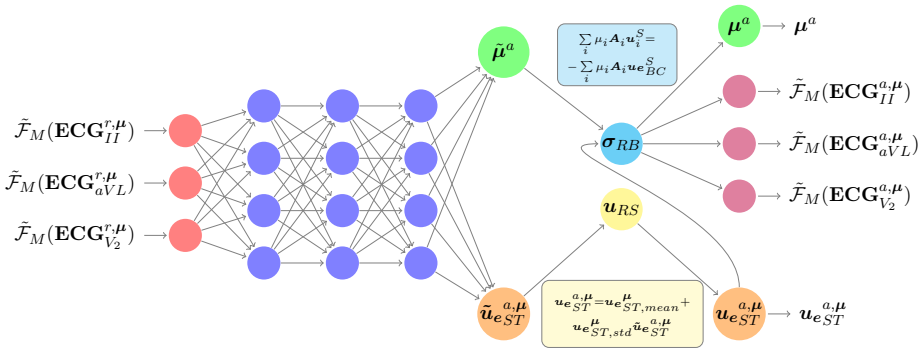


Figure 5.4: Scheme of the architecture of the ST-RB-DNN model with input signals expressed via their lowest-frequency DFT coefficients; the Input Layer is in red and it contains the lowest-frequency DFT coefficients of the signals to be processed; the trainable part of the NN is in blue and its architecture coincides with the one of a simple MLP; the parameter estimator layer is in green and it contains the normalized estimated values of some relevant scalar parameters; the epicardial extracellular potential estimator layer is in orange and it contains the values of the coefficients arising from the projection of the epicardial extracellular potential onto the dimensionality reduced subspace in both Space and Time dimensions; the epicardial extracellular potential rescaler layer is in yellow and it allows to rescale the quantities estimated by the first part of the NN to actually obtain the desired coefficients; the RB-solver layer is in cyan and it allows to make the NN aware of the physics of the phenomenon of interest, by solving a dimensionality reduced problem that reconstructs the signals, whose lowest-frequency DFT coefficients have been given as input; the signals reconstructor layer is in purple and it simply hosts the values of the lowest-frequency DFT coefficients of the signals reconstructed by the RB-solver layer, configuring as the autoencoding portion of the output

5.2.2 The DFT-Based ST-RB-DNN Model

The second ST-RB-DNN model we implemented is characterized by taking as input the lowest-frequency coefficients, arising from the application of a Discrete Fourier Transform (DFT) to the body surface potentials; its structure can be visualized in Figure 5.4.

Two main reasons have driven us towards the development of this model. On the one side, giving as input to the NN quantities that are already "aware" of the temporal dynamics of the problem at hand prevents from designing suitable layers (as the $1D$-Convolutional ones) to capture such dynamics; this results in the possibility of developing simpler architectures, easing and shortening the training, without (hopefully) sacrificing accuracy. On the other side, a drawback of the model that takes as input signals organized in the form of time series is that it can work only at a fixed acquisition frequency; in our test cases, for instance, we sampled the signals at 500 Hz, which is the standard ECG acquisition frequency in modern machines. Thus, if the time-series-based model has to be tested on signals sampled at 100 Hz, then those have to be interpolated in time before being used as input. Conversely, if a DFT is applied to the signals, the values of the resulting lowest-frequency coefficients feature low sensitivity with respect to the acquisition frequency; thus, using them as input, it allows to develop a model that works independently of the acquisition frequency, without any interpolation being required at pre-processing stage. Notice, as an important *caveat*, that the low sensitivity occurs only with respect to the lowest-frequency coefficients; higher-frequency ones are instead more sensitive to changes in the acquisition frequency and they should not be given as input to the model. Incidentally, such coefficients are typically not much informative, thus the model accuracy should not be affected too much by their removal from the input.

The main differences/specifics with respect to both the general ST-RB-DNN model and the time-series-based one are listed in the following.

- **Input Layer** (in red): the input is made by the first M coefficients, arising from the application of a $1D$ DFT to the original signals.

Regarding the DFT and the Fast Fourier Transform (FFT) algorithm employed for its efficient computation, we refer the reader to Appendix B.2 and to [84] for further details and information. Here, it is enough to say that the $1D$ DFT is the discrete counterpart of the uni-variate Fourier transform and that it turns a sequence of N complex numbers $\{x_n\}_{n=0}^{N-1}$ into another sequence of N complex numbers $\{X_k\}_{k=0}^{N-1}$ such that:

$$\begin{aligned} X_k &= \sum_{n=0}^{N-1} x_n \, e^{-\frac{2\pi i}{N} kn} \\ &= \sum_{n=0}^{N-1} x_n \left[\cos\left(\frac{2\pi}{N} kn\right) - i \sin\left(\frac{2\pi}{N} kn\right) \right] \end{aligned} \tag{5.14}$$

It is a linear transform and an important property is that the DFT of real-valued signals is Hermitian, i.e. the component at frequency f_k is the complex conjugate of the component at frequency $-f_k$. This implies that, out of the N entries of X_k, only $\left\lfloor \frac{N}{2} \right\rfloor + 1$ actually carry relevant information, where the 1 refers to the zero-frequency component. Thus, being N_t the length of the measured signals, the dimension of the resulting DFT is at most $\bar{M} =: \left\lfloor \frac{N_t}{2} \right\rfloor + 1$; the length of the signals given as input to the model is then $M \leq \bar{M}$, upon having excluded a certain fraction of coefficients, related to the highest-frequency modes. The choice of M is a hyperparameter of the model.

Despite NNs can be trained also with complex-valued inputs and outputs, the vast majority of the models processes real-valued data; indeed this approach appears to be more robust and also less subject to implementation bugs of the employed DL packages. Because of this, the DFT-based ST-RB-DNN model is not given as input M complex-valued coefficients, but $2M$ real-valued ones, obtained just by splitting the real and the imaginary parts. More specifically, the input is organized in such a way that its first M entries are the real parts of the M complex-valued DFT coefficients, while the the last M ones are their imaginary parts; clearly this is just a matter of choice and the performances of the model should not be affected by changes in this sense.

Finally, it is worth observing that, as the RB coefficients, also the DFT ones span over a broad range of orders of magnitude, which gets broader and broader as the number of considered modes gets larger. Such a behavior has been found to hinder the learning process, since performances of DL models are enhanced if the input/output datapoints are of the same order of magnitude and hopefully even normalized either in $[0; 1]$ or in $[-1; 1]$. An adjustment in this direction, that proved to give better estimation results, has been obtained by applying the following bi-symmetric logarithmic transform to the real and imaginary parts of the chosen DFT coefficients

$$\hat{x} = sign(x) \log_{10}\left(1 + \left|\frac{x}{C}\right|\right) \tag{5.15}$$

with $C = \dfrac{1}{\ln(10)}$, so that a unity slope is achieved in the region near the origin. This transform has been taken from [85], it is diplayed in Figure 5.5 (in red) and its inverse, power transform reads as:

$$x = sign(\hat{x}) C(-1 + 10^{|\hat{x}|}) \tag{5.16}$$

- **Trainable NN** (in blue): since the NN is fed with values that already take into account the time dynamics of the problem at hand, there is no need of constructing layers aimed at capturing such dynamics. Thus, the "Trainable NN" in this case simply consists of a MLP, with several fully-connected layers flanked one after the other.

  More precisely, such layers are divided into two subsets. A first set of layers combines only the DFT coefficients (or better, their real and imaginary parts) relative to the same

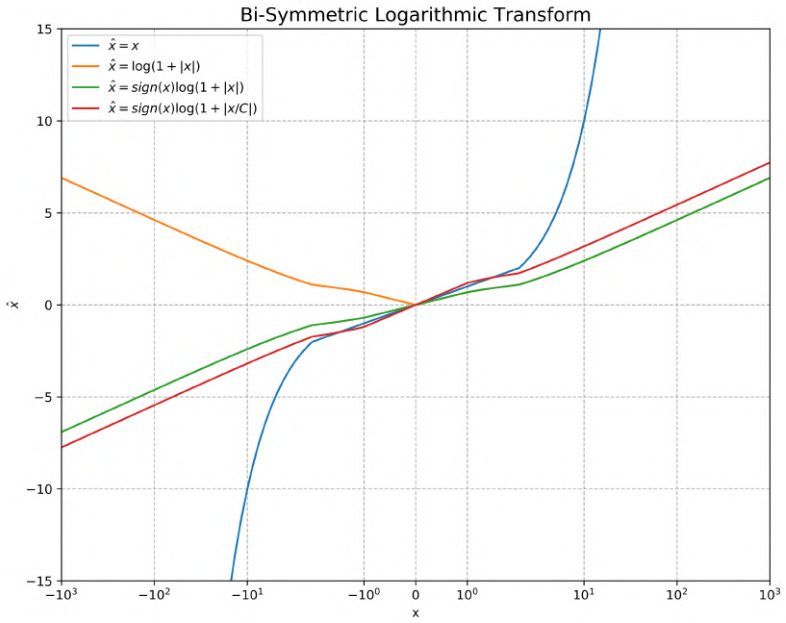


Figure 5.5: Plot of the bi-symmetric logarithmic transform used to scale the DFT coefficients. In particular the plot features a logarithmic scale along the x-axis and it displays: the identity transform (in blue), a non-symmetric logarithmic transform (in orange), a bi-symmetric logarithmic transform (in green) and the final scaled bi-symmetric logarithmic transform (in red)

signal, without merging information coming from different signals and using the same set of weights for all the signals; these are called *pre-flattening fully-connected layers*. Upon the application of a Flatten layer, that reduces the data dimensionality from being $3D$ to $2D$, a second set of fully-connected layers combines without distinction all the available information, with the aim of ultimately performing the best possible prediction of both the characteristic parameters and the "normalized" Space-Time-Reduced epicardial potential; these are called *post-flattening fully-connected layers*. Also, *Dropout* layers with $p = 0.20$ are placed both between the two aforementioned sets of fully-connected layers and between the post-flattening fully-connected layers and the RB-solver one. All layers are subject to Ridge Regularization.

In terms of complexity, we may expect this model to be simpler than the one taking time-series data as input; indeed, on the one side the dimensionality of the input is smaller (since $2M$ should be quite much lower than N_t, if a many high-frequency DFT coefficients are discarded). On the other side, both pre-convolutional and $1D$ convolutional layers are no longer needed, thus leading to a reduction of the number of trainable parameters. Beware, anyway, that this is just a heuristic hypothesis: indeed it is possible that the DFT-based ST-RB-DNN model requires much more fully-connected layers than the time-series-based one in order to reach a comparable degree of accuracy, thus ultimately resulting as or more complex.

- **RB-solver layer** (in cyan): the major difference of the RB-solver layer with respect to the one of the time-series-based ST-RB-DNN model consists in the fact that the lowest-frequency DFT coefficients of the target signals have to be computed and stored in the output tensor. Thus, after having derived such signals (proceeding as described in Subsection 5.2.1), a DFT is applied to those (via a tensorial version of the FFT algorithm), the M lowest-frequency coefficients are extracted, their real and imaginary parts are split

and the bi-symmetric logarithmic transform (5.15) is applied, in order to homogenize their range of values.

- **Output Layer and Loss**: in terms of output and loss function, the only difference is that the log-scaled real and imaginary parts of the M lowest-frequency DFT coefficients have to be returned and penalized, instead of the plain time-series signals.

5.2.3 Numerical Results

This Subsection is devoted to the presentation of the numerical results got on two benchmark test cases with the ST-RB-DNN models described before.

5.2.3.1 Datasets construction

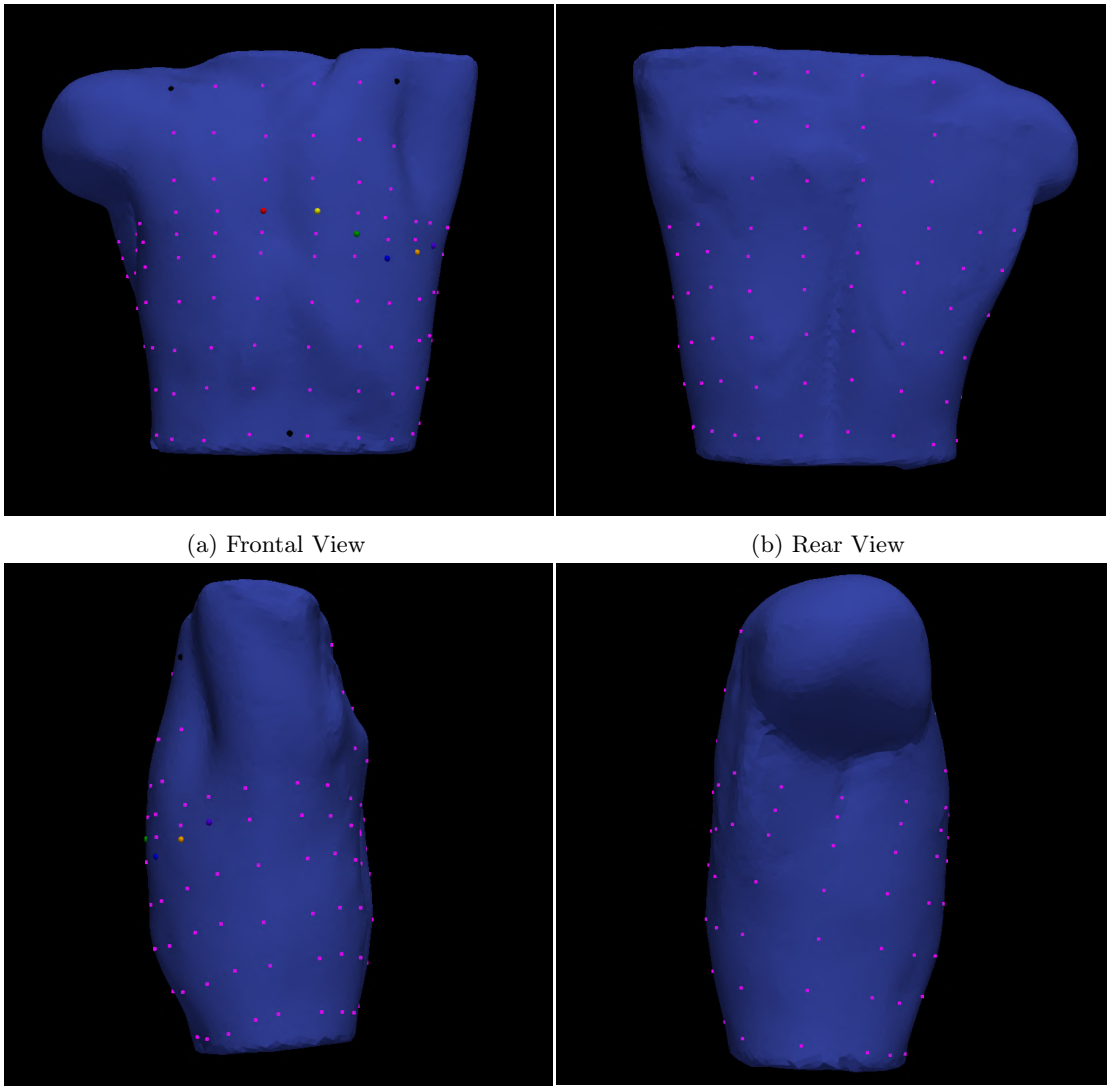
As discussed in Chapter 4, the data used for the training and testing of the ST-RB-DNN model have been generated artificially; indeed DL models need a quite massive amount of data to be trained with success, while the procedures to acquire epicardial potentials are costly and invasive.

For the first test case, the numerical setup is analogous to the one described in Subsection 4.2.3. We approximated the heart EP by means of the bidomain equations, coupled with the phenomenological AP ionic model (see Problem 4.6), on a fixed reference bi-ventricular geometry (see Figure 4.8); we reconstructed the 12-lead ECG signals by solving a generalized Laplace equation (see Problem 4.7) in an idealized geometry of the human torso (see Figure 4.9), approximated as a homogeneous and isotropic volume conductor.

The second test case has been carried out in a slightly more realistic setting. In particular, the approximation of the heart EP has been performed as in the first test case, solving the same set of equations (see Problem 4.6) and employing the same geometry and computational mesh (see Figure 4.8). Conversely, the generalized Laplace equation to estimate the potential field in the torso (see Problem 4.7) has been solved in a more realistic geometry taken from [86] (see Figure 5.6), on top of which a computational mesh made of $498'992$ tetrahedral cells, which result in $94'976$ vertices, has been assembled. Furthermore, the input dataset has not been constructed from 12-lead ECG signals, but from 158 leads, which are generated from the values of the body surface potential recorded by 155 different electrodes, placed on the human torso. The goal was to reproduce the data, collected via electrodes vests, that are employed in modern ECGI applications. Notice that the geometry of Figure 5.6 does not feature the presence of multiple organs; thus the human torso is still approximated as a homogeneous and isotropic volume conductor.

The last step in order to generate a dataset that can be used for the training/testing of the proposed ST-RB-DNN models is the one of adding *variability*. Indeed, on the one side the training datapoints must differ one from the other, so that the model can learn from them the widest possible amount of dynamics and conditions; on the other side, also the testing datapoints should show differences both within each other and with respect to the training ones, so that a proper assessment of the performances of the model can be made. It is then clear that the behavior of the model highly relies on the way the dataset is constructed. In our case, we are aware of the fact that the simplifying modeling assumptions we made (in both test cases) prevent us from performing realistic simulations and thus from employing our model with success on real data. Despite that, anyway, we tried to enrich the data generation process with several sources of randomness, in order to construct a sufficiently variable dataset, able to challenge the capabilities of the developed physics-aware DL model.

Since the torso problem consists simply in a generalized Laplace equation in a homogeneous and isotropic volume conductor, no randomness has been imposed at such stage. Actually, apart from considering a non-homogeneous case where some variability on the different conducibilities could be imposed, an interesting possibility would be to take into account the position of the heart in the thorax, performing different simulations for different values of the cardiac axis. This



(a) Frontal View (b) Rear View

(c) Left Lateral View (d) Right Lateral View

Figure 5.6: View of the human torso geometry employed in the second test case from four different perspectives (frontal, rear, left lateral and right lateral). The black spheres (frontal view) denote the positions of the three reference electrodes, which are analogous to the ones used by the standard 12-lead ECG system. The colored spheres (frontal and left lateral views) denote the positions of the 12-lead ECG electrodes, which are part of the dataset; the colors follow the *American Heart Association (AHA)* color-coding system. The lilac squares denote the positions of the electrodes that, together with the 12-lead ECG ones, allow to assemble the dataset

configures as a possible update of the project, but no investigation has been made in this sense. Incidentally, notice that the choice of a homogeneous torso implies that no parameters' estimator layer is present in the developed models.

Regarding the heart EP, instead, several sources of randomness have been added. Since the geometry and the computational mesh of the heart is the same in the two test cases, the same choices have been made. Tables 5.1 and 5.2 report the values of the parameters used in the AP model and in the bidomain system of equations, respectively. Notice that only ϵ_0 has been chosen random, following a normal distribution, for what concerns the ionic model. Conversely all the heart conducibilities (longitudinal and transversal to the fibers direction, intracellular and extracellular) have been sampled from a uniform distribution, since they appeared to massively influence the problem dynamics both in terms of epicardial activation maps and, in turn, of body surface potentials.

$\mathbf{K}$	$\mathbf{a}$	$\epsilon_\mathbf{0}$	$\mu_\mathbf{1}$	$\mu_\mathbf{2}$	$\mathbf{b}$	$\mathbf{V_{min}}$	$\mathbf{V_{max}}$
8	$1.5 \cdot 10^{-1}$	$\mathcal{N}(6.5, 1.0) \cdot 10^{-3}$	$1.0 \cdot 10^{-1}$	$3.0 \cdot 10^{-1}$	$1.5 \cdot 10^{-1}$	$-80\ mV$	$20\ mV$

Table 5.1: Values of the parameters used in the AP ionic model to construct the training/testing datasets. $\mathcal{N}(\mu, \sigma)$ denotes a normal distribution with mean equal to μ and standard deviation equal to σ.

$\sigma_{\mathbf{l_i}}[S\ cm^{-1}]$	$\sigma_{\mathbf{t_i}}[S\ cm^{-1}]$	$\sigma_{\mathbf{l_e}}[S\ cm^{-1}]$	$\sigma_{\mathbf{t_e}}[S\ cm^{-1}]$
$\mathcal{U}(4.5, 7.5) \cdot 10^{-4}$	$\mathcal{U}(0.75, 1.05) \cdot 10^{-4}$	$\mathcal{U}(1.80, 2.70) \cdot 10^{-3}$	$\mathcal{U}(1.80, 2.70) \cdot 10^{-4}$

Table 5.2: Values of other random parameters used in the numerical approximation of the bidomain equations to construct the training/testing datasets. $\mathcal{N}(\mu, \sigma)$ denotes a normal distribution with mean equal to μ and standard deviation equal to σ. $\mathcal{U}(a, b)$ denotes a uniform distribution over the interval $[a; b]$.

Nb.	**Position**	**Pr(Healthy)**	**Pr(LBBB)**	**Pr(RBBB)**	**Times[ms]**	**Incompatible**
1.	ALV paraseptal	0.70	0.00	1.00	$\mathcal{U}(0, 10)$	//
2.	ALV apex	0.30	0.00	0.40	$\mathcal{U}(0, 20)$	3
3.	ALV free wall	0.30	0.00	0.40	$\mathcal{U}(10, 30)$	2
4.	ILV apex	0.30	0.00	0.40	$\mathcal{U}(0, 20)$	//
5.	ILV base	0.30	0.00	0.40	$\mathcal{U}(0, 20)$	6
6.	ILV base lateral	0.20	0.00	0.30	$\mathcal{U}(0, 20)$	5
7.	ARV paraseptal	1.00	1.00	0.00	$\mathcal{U}(0, 15)$	//
8.	IRV base	0.40	0.20	0.00	$\mathcal{U}(5, 25)$	9
9.	IRV base lateral	0.50	0.20	0.00	$\mathcal{U}(5, 25)$	8
10.	IRV apex	0.10	0.40	0.00	$\mathcal{U}(5, 25)$	//

Table 5.3: Positions, sampling probabilities, activation times and incompatibilities of the 10 selected initial stimulation areas for both test cases. The column named "Incompatible" lists the indices of the areas that cannot be stimulated together with the considered one. $\mathcal{U}(a, b)$ denotes a uniform distribution in the interval $[a; b]$.

Another crucial source of randomness is given by the initial activation pattern that, as already said in Subsection 4.2.3, is inspired by the works of *Wyndham et al.* on EBTs localization in both healthy [39] and LBBB-affected [54] patients. In particular, a thin endocardial and sub-endocardial layer is activated, for a duration of 5 ms, in 10 possible different ventricular regions, localized on both the left and the right ventricle. If the patient is healthy, then all these 10 regions could possibly activate (at different times), with the constraint that no more than 5 EBTs should appear; if, instead, the patient is affected by LBBB (RBBB), then only the regions located on the right (left) ventricle can be involved in the stimulation protocol and no more than 3 EBTs should be observed. The "choice" between healthy, LBBB and RBBB cases is governed by a discrete uniform random variable, so that the dataset is equi-partitioned with respect to the three categories. Furthermore, the position of the point around which the initial stimulation is applied can vary within a sphere of 2 mm radius, obeying to a $3D$ uniform distribution, and the time instants at which the different regions are activated are sampled from a uniform distribution, with ranges having been defined according to [39, 54]. Table 5.3 summarizes the positions of the initial activation areas, their probabilities of sampling in the healthy, LBBB and RBBB cases and the ranges of time instants at which stimulation could occur. Also, eventual incompatibilities between different activation regions are reported.

Ultimately, the two datasets have been obtained by solving the FOM problem for $N_{\boldsymbol{\mu}} = 400$ and $N_{\boldsymbol{\mu}} = 180$ different values of the aforementioned parameters, respectively. Training and Test datasets have been separated via a $90\% - 10\%$ splitting and the validation datasets have been assembled by picking the 10% of the training datapoints. Furthermore, the training and validation datasets have been subject to on-the-fly data augmentation, by superimposing a correlated white Gaussian noise, so that an average SNR of $\approx 19\ dB$ is achieved.

5.2.3.2 Test Case 1: The Time-Series-Based ST-RB-DNN Model

In the following, the discussion of the results achieved with the application of the time-series-based ST-RB-DNN model to the first benchmark case is provided.

Grid Search

As a starting point, a grid search involving the most relevant model hyperparameters is performed; more specifically the considered hyperparameters are the number of neurons in the pre-convolutional layers, the number of convolutional filters, the dimension of the $1D$ convolutional kernel, the number of neurons in the post-convolutional layers and the weight of the Space-Time-Reduced epicardial potential in the loss functional. A total of 288 models has been tested. All the other model hyperparameters have been kept fixed to values that proved to give satisfactory results in some preliminary numerical tests; in particular:

- **POD tolerances**: the tolerances of the PODs on both the torso (in space) and the epicardial (in space and in time) potential are two key hyperparameters; the dependency of the model performances on those will be investigated lately. For the first tests, we have kept the tolerances fixed at $\epsilon_{POD}^{t,s} = 10^{-3}$, $\epsilon_{POD}^{e,s} = 10^{-1}$ and $\hat{\epsilon}_{POD}^{e,t} = 5 \cdot 10^{-2}$; with such choices, the spatial Reduced Basis for the torso potential is made of 316 elements, while the spatio-temporal Reduced Basis for the epicardial one has a cardinality of 619. The relative l^1-norm error on epicardial activation maps is equal to $3.98 \cdot 10^{-2}$, while the one on ECG signals is of $4.34 \cdot 10^{-2}$; errors are computed as averaged over 25 different snapshots.
- **Optimizer**: the choice of the optimizer (i.e. the method used to minimize the loss functional) is crucial in DL models. We chose the Nesterov adaptive moment estimation (Nadam) algorithm, a variant of the classical Adam algorithm, where Nesterov momentum is incorporated; all details can be found in [87].
- **Learning Rate**: also the choice of the learning rate is very important in order to properly train a NN and to ensure that it converges adequately. Too small learning rates may indeed cause the training to get stuck in a fixed point of the parameters' space, while too large ones may prevent the model to converge towards its minima. As a first guess, we chose the learning rate $\nu = 10^{-3}$.
- **Regularization**: the value of the regularization parameter can be important in ensuring a good training, which does not suffer of overfitting and that features a remarkable generalization power. As anticipated in Section 5.1, we chose to apply Ridge Regularization to all the fully-connected and $1D$ convolutional layers; we set the regularization parameter to 10^{-7} in the fully-connected layers and to 10^{-4} in the convolutional ones.
- **Activation Functions**: another key hyperparameter is the choice of the activation functions. We decided to employ the *ReLU* function to all the layers, but to the one responsible for the estimation of the epicardial potential, since we experienced the so-called "Dying *ReLU*" problem. Such layer has been then activated with the *Scaled Exponential Linear Unit (SELU)* i.e. a scaled version of *ELU* (see (5.7)) with scale factor $s > 1$. In particular we chose $a = 1.67326324$ and $s = 1.05070098$, being the default values provided by the *Tensorflow/Keras Python* package.

Other hyperparameters, as the value of the dropout fraction, have been kept fixed to the values reported in the previous Subsection. All models have been trained for a maximum of 100 epochs; training has been stopped if the loss on the validation dataset did not show any improvement for 20 consecutive epochs; the learning rate has been reduced by a factor of 4, for no more than 4 times and up to a minimum value of 10^{-6}, if the validation loss did not show any improvement for 10 consecutive epochs. Tables 5.4, 5.5, 5.6 report the results of the grid search, in terms of l^1-norm relative errors on the epicardial activation maps, for three values of the weight of the Space-Time-Reduced epicardial potential in the loss functional. **All errors are computed with respect to the optimal Space-Time ROM approximation**. Figures 5.7-5.9 and 5.8-5.10 display the epicardial activation maps and the ECG signals reconstructed by the best model (in terms of activation maps average l^1-norm relative error) for two different test datapoints.

Dense \ 1D Conv.	(5) - 15	(5) - 25	(10) - 15	(10) - 25	(5,5) - 15	(5,5) - 25	(10,10) - 15	(10,10) - 25
(32) - (64,32,16,8)	9.95e-2	1.58e-1	1.26e-1	1.11e-1	1.64e-1	1.57e-1	1.36e-1	1.30e-1
(32) - (128,64,32,16,8)	1.13e-1	1.07e-1	1.46e-1	9.04e-2	1.40e-1	1.75e-1	1.46e-1	1.02e-1
(32) - (256,128,64,32,16,8)	1.04e-1	1.14e-1	1.21e-1	1.02e-1	1.37e-1	1.05e-1	1.42e-1	1.26e-1
(64) - (64,32,16,8)	1.29e-1	9.39e-2	1.13e-1	1.22e-1	1.11e-1	1.41e-1	1.25e-1	1.26e-1
(64) - (128,64,32,16,8)	1.16e-1	1.14e-1	1.18e-1	1.09e-1	1.29e-1	1.33e-1	1.03e-1	1.38e-1
(64) - (256,128,64,32,16,8)	1.05e-1	1.02e-1	1.03e-1	8.10e-2	1.22e-1	1.25e-1	1.11e-1	1.07e-1
(32,32) - (64,32,16,8)	1.06e-1	1.30e-1	1.27e-1	1.06e-1	1.36e-1	1.16e-1	1.29e-1	1.29e-1
(32,32) - (128,64,32,16,8)	1.00e-1	9.89e-2	1.19e-1	1.02e-1	1.30e-1	1.25e-1	1.34e-1	1.41e-1
(32,32) - (256,128,64,32,16,8)	1.08e-1	1.18e-1	1.03e-1	1.27e-1	1.38e-1	1.40e-1	1.15e-1	1.16e-1
(64,64) - (64,32,16,8)	1.06e-1	9.38e-2	8.89e-2	9.43e-2	1.06e-1	1.31e-1	1.26e-1	1.28e-1
(64,64) - (128,64,32,16,8)	9.42e-2	1.01e-1	9.30e-1	9.80e-1	1.27e-1	8.29e-1	1.30e-1	1.22e-1
(64,64) - (256,128,64,32,16,8)	1.04e-1	9.90e-2	8.11e-2	8.34e-2	1.05e-1	1.16e-1	1.54e-1	1.18e-1

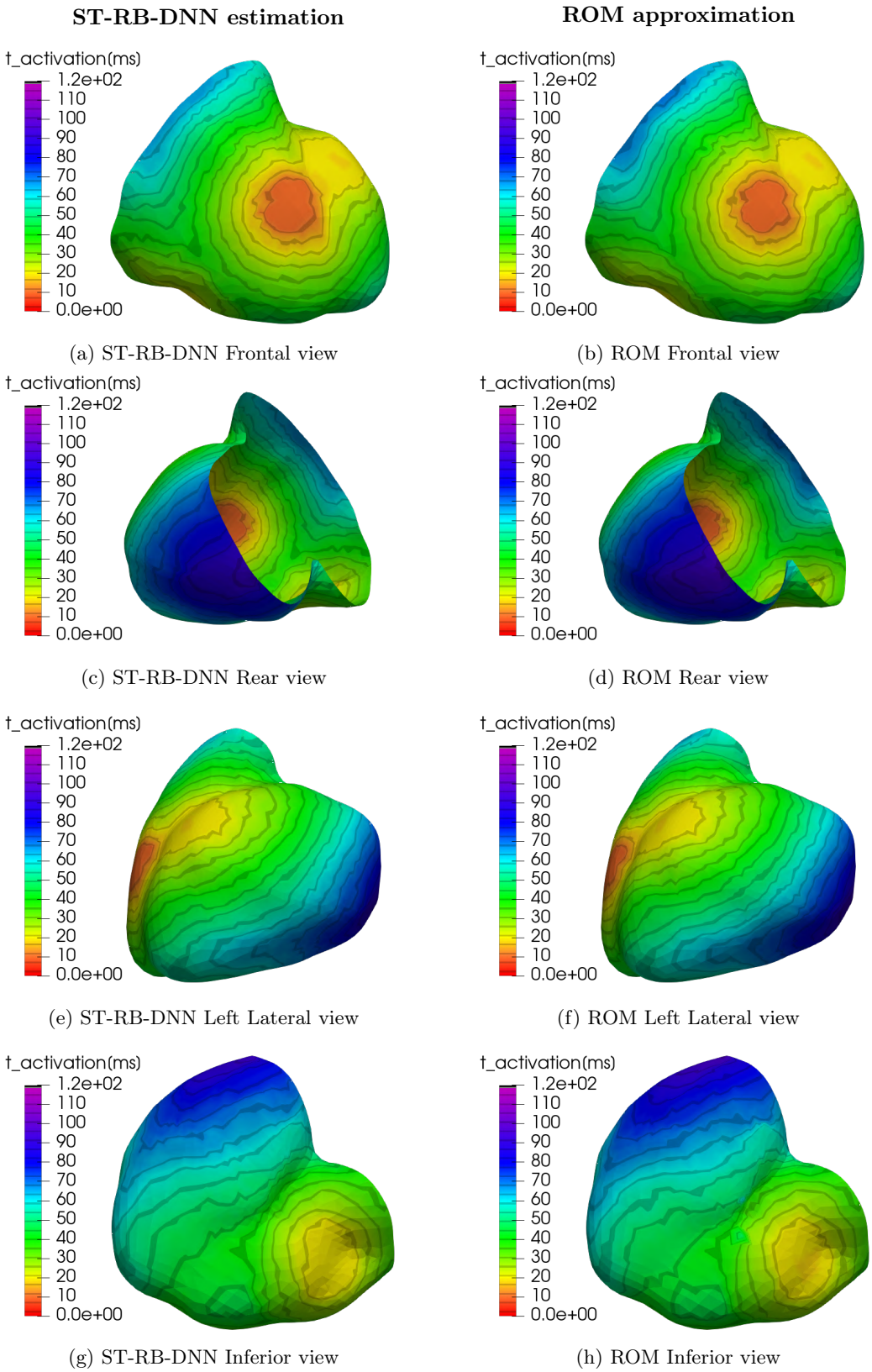
Table 5.4: Average relative errors in l^1-norm on the epicardial activation maps of the test dataset with the time-series-based ST-RB-DNN model, using epicardial potential **loss weight equal to 100**. The green cell displays the best model; the red cell displays the worst model; the yellow cells display the best 5 models (except from the very best one). Rows label are of the form *Pre-Layers* - *Post-Layers*, where the first entry defines the layers of the pre-convolutional fully-connected block and the second one the layers of the post-convolutional fully-connected block. Columns labels are of the form N_F - K_{dim}, being N_F the number of convolutional filters and K_{dim} the dimension of the $1D$ convolutional kernel.

Dense \ 1D Conv.	(5) - 15	(5) - 25	(10) - 15	(10) - 25	(5,5) - 15	(5,5) - 25	(10,10) - 15	(10,10) - 25
(32) - (64,32,16,8)	9.31e-2	8.90e-2	1.00e-1	1.01e-1	1.21e-1	1.07e-1	1.25e-1	1.02e-1
(32) - (128,64,32,16,8)	1.13e-1	7.83e-2	1.06e-1	9.89e-2	1.09e-2	1.04e-1	9.23e-2	1.12e-1
(32) - (256,128,64,32,16,8)	8.89e-2	9.32e-2	9.56e-2	9.49e-2	1.08e-1	1.17e-1	1.08e-1	1.00e-1
(64) - (64,32,16,8)	8.24e-2	9.14e-2	7.58e-2	9.54e-2	9.74e-1	9.08e-2	9.72e-2	1.08e-1
(64) - (128,64,32,16,8)	8.50e-2	7.48e-2	7.89e-2	9.06e-2	9.93e-2	8.46e-2	1.15e-1	8.71e-2
(64) - (256,128,64,32,16,8)	1.01e-1	7.37e-2	7.75e-2	6.26e-2	9.23e-2	8.50e-2	9.73e-2	7.41e-2
(32,32) - (64,32,16,8)	1.14e-1	9.19e-2	9.21e-2	8.38e-2	1.10e-1	1.08e-1	9.98e-2	1.10e-1
(32,32) - (128,64,32,16,8)	9.34e-2	7.99e-2	8.12e-2	8.83e-2	1.14e-1	8.32e-2	1.10e-1	1.02e-1
(32,32) - (256,128,64,32,16,8)	9.76e-2	7.81e-2	8.03e-2	7.45e-2	7.75e-2	9.47e-2	1.14e-1	8.47e-2
(64,64) - (64,32,16,8)	9.62e-2	7.74e-2	1.01e-1	6.91e-2	1.06e-1	8.34e-2	8.30e-2	9.33e-2
(64,64) - (128,64,32,16,8)	8.28e-2	8.21e-2	8.04e-2	8.37e-2	9.60e-2	1.11e-1	8.81e-2	9.22e-2
(64,64) - (256,128,64,32,16,8)	8.10e-2	7.29e-2	7.62e-2	6.36e-2	9.72e-2	8.39e-2	9.27e-2	7.32e-2

Table 5.5: Average relative errors in l^1-norm on the epicardial activation maps of the test dataset with the time-series-based ST-RB-DNN model, using epicardial potential **loss weight equal to 500**. The green cell displays the best model; the red cell displays the worst model; the yellow cells display the best 5 models (except from the very best one). Rows label are of the form *Pre-Layers* - *Post-Layers*, where the first entry defines the layers of the pre-convolutional fully-connected block and the second one the layers of the post-convolutional fully-connected block. Columns labels are of the form N_F - K_{dim}, being N_F the number of convolutional filters and K_{dim} the dimension of the $1D$ convolutional kernel.

Dense \ 1D Conv.	(5) - 15	(5) - 25	(10) - 15	(10) - 25	(5,5) - 15	(5,5) - 25	(10,10) - 15	(10,10) - 25
(32) - (64,32,16,8)	9.21e-2	8.87e-2	8.43e-2	9.84e-2	1.35e-1	9.18e-2	9.57e-2	1.04e-1
(32) - (128,64,32,16,8)	9.84e-2	7.18e-2	7.67e-2	8.78e-2	9.13e-2	1.02e-1	1.42e-1	1.10e-1
(32) - (256,128,64,32,16,8)	9.99e-2	6.70e-2	8.93e-2	8.31e-2	9.06e-2	1.25e-1	8.65e-2	9.35e-2
(64) - (64,32,16,8)	9.35e-2	9.70e-2	7.47e-2	8.88e-2	1.04e-1	1.21e-1	1.04e-1	9.97e-2
(64) - (128,64,32,16,8)	8.73e-2	8.85e-2	9.21e-2	7.28e-2	1.24e-1	1.13e-1	7.71e-2	1.08e-1
(64) - (256,128,64,32,16,8)	9.20e-2	6.44e-2	7.06e-2	8.57e-2	8.52e-2	8.83e-2	1.04e-1	8.86e-2
(32,32) - (64,32,16,8)	8.74e-2	7.42e-2	7.16e-2	9.72e-2	1.02e-1	1.03e-1	1.26e-1	1.16e-1
(32,32) - (128,64,32,16,8)	9.82e-2	7.42e-2	7.93e-2	8.27e-2	9.22e-2	8.45e-2	9.15e-2	1.21e-1
(32,32) - (256,128,64,32,16,8)	8.83e-2	8.07e-2	7.92e-2	9.51e-2	1.15e-1	1.08e-1	1.11e-1	1.12e-1
(64,64) - (64,32,16,8)	9.12e-2	9.40e-2	7.56e-2	8.53e-2	8.06e-2	9.56e-2	1.08e-1	8.81e-2
(64,64) - (128,64,32,16,8)	8.39e-2	6.76e-2	9.22e-2	6.78e-2	9.90e-2	7.97e-2	7.60e-2	6.94e-2
(64,64) - (256,128,64,32,16,8)	8.03e-2	6.66e-2	6.96e-2	7.33e-2	9.59e-2	8.80e-2	9.88e-2	7.77e-2

Table 5.6: Average relative errors in l^1-norm on the epicardial activation maps of the test dataset with the time-series-based ST-RB-DNN model, using epicardial potential **loss weight equal to 1000**. The green cell displays the best model; the red cell displays the worst model; the yellow cells display the best 5 models (except from the very best one). Rows label are of the form *Pre-Layers* - *Post-Layers*, where the first entry defines the layers of the pre-convolutional fully-connected block and the second one the layers of the post-convolutional fully-connected block. Columns labels are of the form N_F - K_{dim}, being N_F the number of convolutional filters and K_{dim} the dimension of the $1D$ convolutional kernel.



(a) ST-RB-DNN Frontal view

(b) ROM Frontal view

(c) ST-RB-DNN Rear view

(d) ROM Rear view

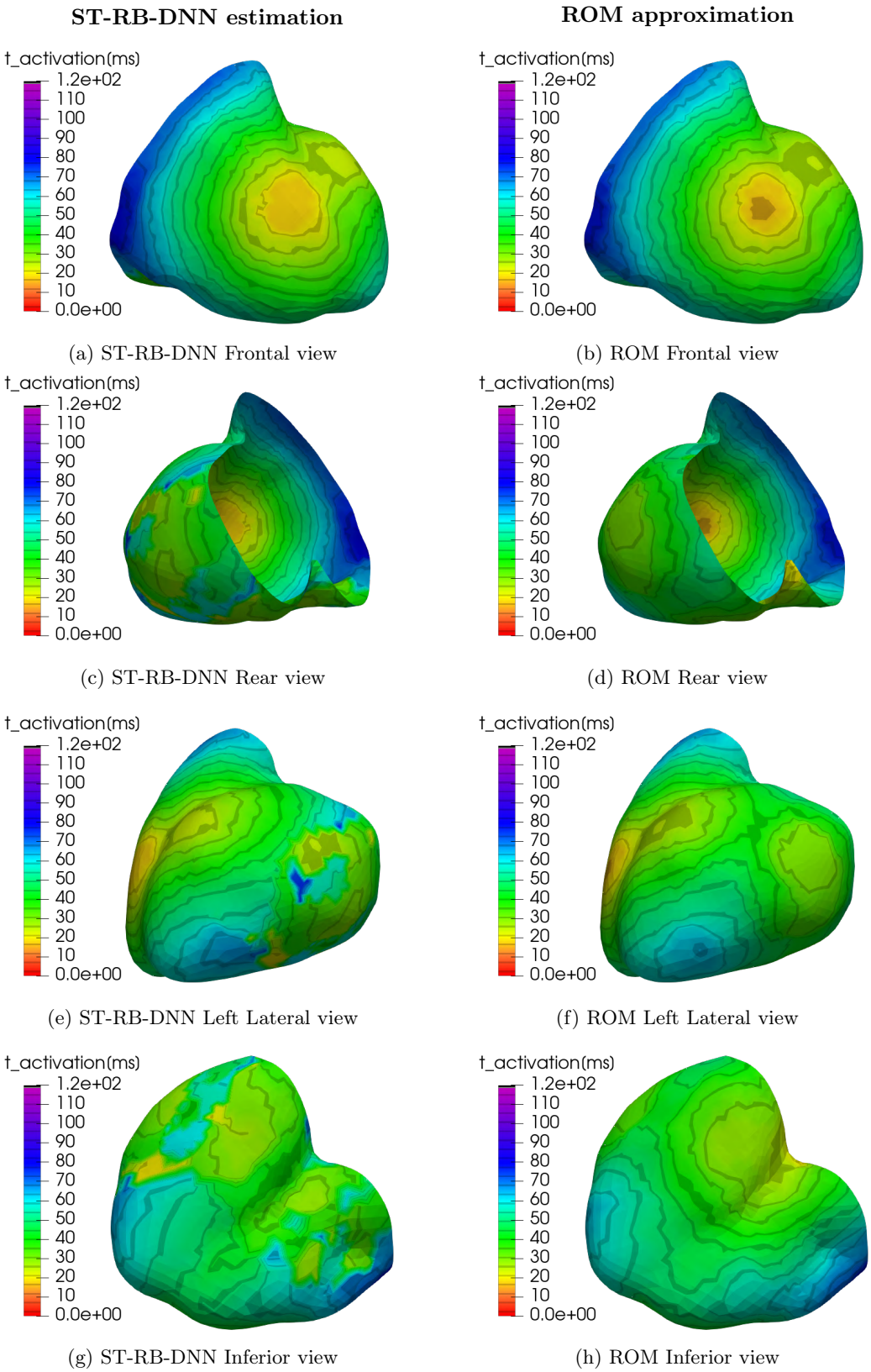
(e) ST-RB-DNN Left Lateral view

(f) ROM Left Lateral view

(g) ST-RB-DNN Inferior view

(h) ROM Inferior view

Figure 5.7: Epicardial activation maps obtained, on test datapoint 1, with the best time-series-based ST-RB-DNN model (left column) and with Space-Time ROM reconstruction (right column) from four different perspectives (frontal, rear, left-lateral and inferior). The l^1-norm relative error equals $2.49 \cdot 10^{-2}$.



(a) ST-RB-DNN Frontal view

(b) ROM Frontal view

(c) ST-RB-DNN Rear view

(d) ROM Rear view

(e) ST-RB-DNN Left Lateral view

(f) ROM Left Lateral view

(g) ST-RB-DNN Inferior view

(h) ROM Inferior view

Figure 5.8: Epicardial activation maps obtained, on test datapoint 2, with the best time-series-based ST-RB-DNN model (left column) and with Space-Time ROM reconstruction (right column) from four different perspectives (frontal, rear, left-lateral and inferior). The l^1-norm relative error equals $6.39 \cdot 10^{-2}$.

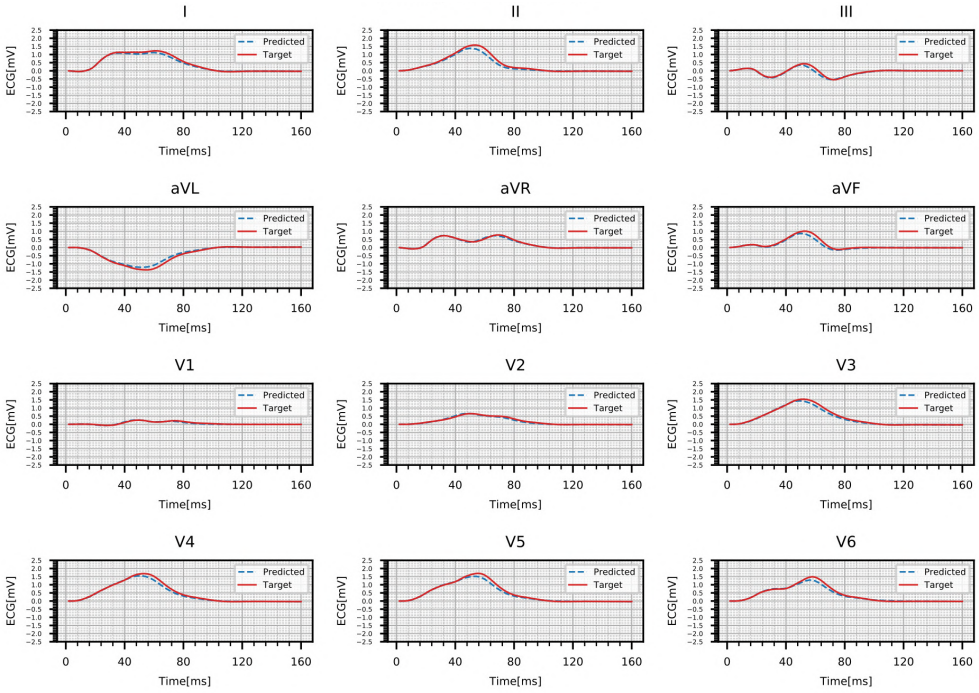


Figure 5.9: ECG signals obtained, on test datapoint 1, with the best time-series-based ST-RB-DNN model and with Space-Time ROM reconstruction. The Red solid line represents the ECG signal got with the ROM approximation; the Blue dashed line represents the ECG signal got with the best time-series-based ST-RB-DNN model. The l^1-norm absolute error (averaged on all the leads) equals $2.81 \cdot 10^{-2}\ mV$.

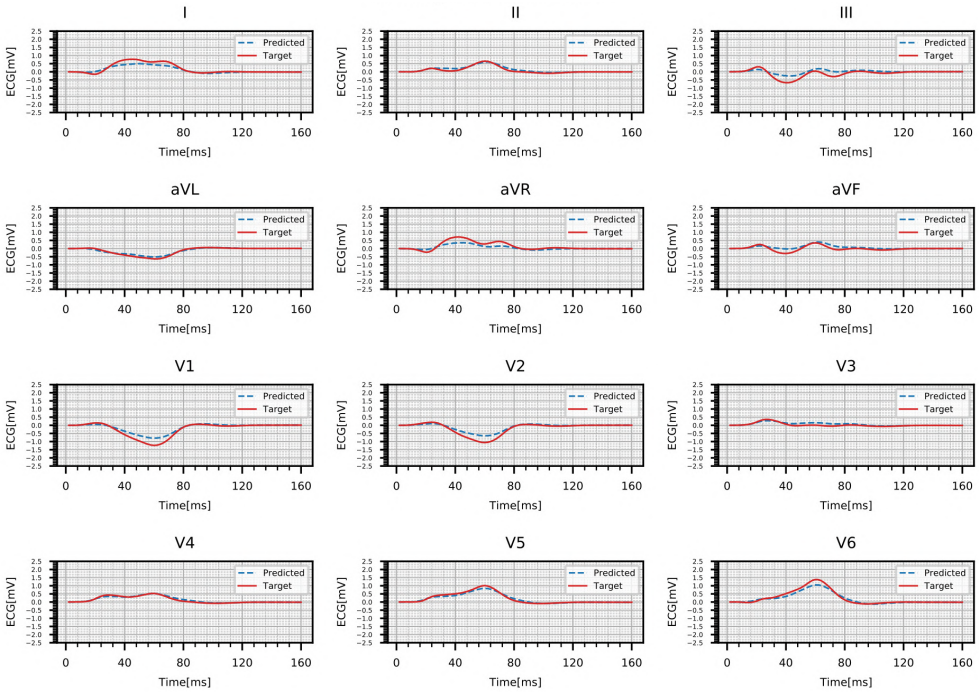


Figure 5.10: ECG signals obtained, on test datapoint 2, with the best time-series-based ST-RB-DNN model and with Space-Time ROM reconstruction. The Red solid line represents the ECG signal got with the ROM approximation; the Blue dashed line represents the ECG signal got with the best time-series-based ST-RB-DNN model. The l^1-norm absolute error (averaged on all the leads) equals $4.83 \cdot 10^{-2}\ mV$.

Looking at the results, the following considerations can be made:

- At first, we analyze the behavior with respect to the weight of the Space-Time-Reduced epicardial potential in the loss functional. At this aim it is worth recalling that the loss functional is built as in (5.3), with w_{sig} being kept fixed at 1 and with the regularization term deriving from the Ridge Regularization applied to all fully-connected and $1D$-convolutional layers. It can be clearly seen that the errors are higher if $w_{BC} = 100$, while they get lower if $w_{BC} = 500$ or if $w_{BC} = 1000$. To see it from a different perspective, choosing $w_{BC} = 100$ determines a $80\% - 20\%$ split of the loss (discarding the regularization term contribution) into the MAE on the Space-Time-Reduced epicardial potential and the MSE on the ECG signals, while with $w_{BC} = 500$ the splitting is of the order of $90\% - 10\%$ and with $w_{BC} = 1000$ it is of the order of $99\% - 1\%$. Comparing the results got with weights of 500 and 1000, no significant differences can be observed, with the average activation maps errors being quite similar. Actually, the best model is got with $w_{BC} = 500$ and it features an error that is 0.18% lower than the one achieved by the best model with $w_{BC} = 1000$; this is an indication of the physically-aware regularization properties of the RB-solver layer, since increasing its "contribution" allows, to some extent, to improve the model performances.
- Concerning the hyperparameters related to the fully-connected layers preceding and following the $1D$ convolutional ones, it can be noticed that, whichever the value of w_{BC}, the best results have been achieved with a single layer made of 64 neurons before the convolutions and 6 layers, with a number of nodes exponentially decreasing from 2^8 to 2^3, after the convolutions. This outcome may suggest that even better results can be achieved placing more neurons in a unique pre-convolutional layer and/or increasing the number of layers following the convolutional one; no further investigations have been made in this sense. Results concerning the usage of more than 8 nodes in the last post-convolutional layer (which could induce a dramatic increase of the number of trainable parameters) will be instead discussed in the following of the report.
- Finally, if we focus on the model performances with respect to the hyperparameters relative to the $1D$-convolutional layers (i.e. the number of channels and the dimension of the kernel), we notice that the best model always features a unique convolutional layer, with kernel of dimension 25, whichever the value of w_{BC}. The number of channels, instead, seems to exert a smaller influence, being the errors quite similar between the cases with $N_F = 5$ and $N_F = 10$. More in detail, models trained with lower values of w_{BC} show better performances if $N_F = 10$, while, for $w_{BC} = 1000$, choosing $N_F = 5$ leads to lower errors. Also in this case, the outcome may suggest that better results can be achieved taking even larger convolutional kernels, but no investigations have been made in this sense.
- The model exhibiting the lowest errors on the epicardial activation maps is also the best one in terms of l^1-norm absolute errors on the ECG signals, featuring an average error of $4.87 \cdot 10^{-2}\ mV$. This observation allows to say that increasing the contribution of the signals in the expression of the loss functional does not necessarily bring to a better reconstruction of those; rather, a more careful calibration of the loss contributions enables the optimization algorithm to perform a smarter minimization, ultimately attaining better results both in terms of activation maps and of ECG signals.
- **Test Datapoint 1**: numerical results got by the best model on the first test datapoint are remarkably good, both in terms of activation map (Figure 5.7) and of ECG signals reconstruction (Figure 5.9). Indeed, on the one side all the 3 EBTs (paraseptal both at ALV and ARV, inferior, lateral and close to the base at ILV) have been identified correctly, both in terms of positions and timings; the l^1-norm relative error equals $2.49 \cdot 10^{-2}$ in this case. On the other side, the traces of the predicted ECG signals are close (at least in the eyeball norm) to the target ones at all leads; the l^1-norm absolute error (averaged on all the leads) equals $2.81 \cdot 10^{-2}\ mV$.
- **Test Datapoint 2**: the results got on the second test datapoint are worse, both in terms of epicardial activation map (Figure 5.8) and of reconstructed ECG signals (Figure 5.10). Regarding the activation map, the errors are concentrated in the inferior part of the heart,

more on the left than on the right ventricle, and on the lateral side of the left ventricle; the l^1-norm relative error is $6.39 \cdot 10^{-2}$. Such an outcome can be imputed to the fact that, reconstructing the dynamics of the predicted epicardial potential, these regions are crossed by more than one depolarization wavefront. Indeed errors in the reconstruction of the epicardial potential coefficients, together with the intrinsic oscillatory nature of the elements of the Space-Time Reduced Basis, may reflect in the estimation of "artificial" wavefronts. If the errors are small, then other wavefronts than the "target" one are negligible and do not leave any trace on the activation map. Conversely, if errors are larger, then it may happen that one of the "artificial" wavefronts induces, in some points, a time derivative larger than the one due to the passage of the "target" wavefront; this, in turn, leads to the generation of a "noisy" activation map. Aside of the pure NN design and training, several post-processing routines can be thought in order to reduce such disturbing effects; for instance, suitable filters could be applied to ROM-in-Space FOM-in-Time epicardial potential or more clever and robust methods to compute the activation map can be developed, rather than just associating each DOF to the time instant at which the local time derivative is maximal. Also in terms of ECG signals the worsening of the results is evident, with more significant estimation errors occurring especially at leads III, aVF (which look at the inferior part of the heart) and aVR (which monitors the lateral aspect of the left ventricle); the l^1-norm absolute error (averaged on all the leads) equals $4.83 \cdot 10^{-2}\ mV$. Incidentally, despite being affected by clear errors, the shape of the target activation map can be qualitatively inferred from the estimated one; the physically-aware regularization achieved via the RB-solver layer may play a central role in this.

Effect of other model hyperparameters

In the following, we provide a brief analysis on the dependency of the time-series-based ST-RB-DNN model performances on other hyperparameters. In particular, we always start from the "best" model architecture derived before and we proceed by one hyperparameter at a time, in order to illustrate its effect.

1. **Epicardial Potential POD Tolerances**

$\hat{\epsilon}^{e,t}_{POD}$ \ $\epsilon^{e,s}_{POD}$	$\mathbf{10^{-1}}$	$\mathbf{5 \cdot 10^{-2}}$	$\mathbf{10^{-2}}$
$\mathbf{10^{-1}}$	1.05e-1	9.89e-2	9.93e-2
$\mathbf{5 \cdot 10^{-2}}$	6.26e-2	7.30e-2	8.69e-2
$\mathbf{10^{-2}}$	8.25e-2	7.64e-2	1.18e-1

Table 5.7: Average activation maps relative errors in l^1-norm on the test dataset for different values of the spatial and temporal POD tolerances on the epicardial potential in the time-series based ST-RB-DNN model (with its "best" architecture)

$\hat{\epsilon}^{e,t}_{POD}$ \ $\epsilon^{e,s}_{POD}$	$\mathbf{10^{-1}}$	$\mathbf{5 \cdot 10^{-2}}$	$\mathbf{10^{-2}}$
$\mathbf{10^{-1}}$	181′285	210′336	242′940
$\mathbf{5 \cdot 10^{-2}}$	256′525	389′031	726′566
$\mathbf{10^{-2}}$	413′066	749′347	1′696′953

Table 5.8: Number of trainable parameters of the time-series-based ST-RB-DNN model (with its "best" architecture) for different values of the POD tolerances

$\hat{\epsilon}^{e,t}_{POD}$ \ $\epsilon^{e,s}_{POD}$	$\mathbf{10^{-1}}$	$\mathbf{5 \cdot 10^{-2}}$	$\mathbf{10^{-2}}$
$\mathbf{10^{-1}}$	259	398	554
$\mathbf{5 \cdot 10^{-2}}$	619	1253	2868
$\mathbf{10^{-2}}$	1368	2977	7511

Table 5.9: Dimensionality of the Space-Time Reduced Basis for the epicardial potential for different Space-Time POD tolerances

Table 5.7 summarizes the average activation maps l^1-norm relative errors on the test dataset for different POD tolerances both in space and in time. Tables 5.8 and 5.9 show the number of trainable parameters and the dimensionalities of the Space-Time Reduced Basis, respectively, for the same models.

The best result is achieved for $\epsilon^{e,s}_{POD} = 10^{-1}$ and $\hat{\epsilon}^{e,t}_{POD} = 5 \cdot 10^{-2}$, which leads to a model that has to estimate $n_{st} = 619$ coefficients by training the values of $256'525$ parameters. Such a choice seems to offer an optimal compromise; indeed, on the one side in models that have to estimate a lower number of coefficients, the impact of an estimation error is higher, since it is more difficult to "compensate" it. On the other side, instead, models featuring a higher value of n_{st} are more complex and thus more difficult to train, ultimately leading to worse estimates of the activation maps.

2. **Optimizer**

SGD	**AGD**	**SGD Nesterov**	**Adam**	**RMSProp**	**Nadam**	**L-BFGS**
3.86e-1	2.41e-1	2.56e-1	7.01e-2	6.56e-2	6.26e-2	5.33e-1

Table 5.10: Average activation maps relative errors in l^1-norm on the test dataset using different optimizers in the time-series based ST-RB-DNN model (with its "best" architecture)

Table 5.10 reports the average activation maps relative errors in l^1-norm on the test dataset for the best model, trained using different optimizers. The best result is got with the Nadam optimizer, i.e. an improved version of the more classical Adam optimizer (see [14]), which incorporates Nesterov momentum, introduced by *T.Dozat* in [87]. Good results are also achieved with the RMSProp algorithm [15], while the standard SGD algorithm and its variants with either plain momentum (i.e. Accelerated Gradient Descent (AGD)) or Nesterov momentum did not perform well. Additionally, a training attempt has been made with the Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) full-batch algorithm, as it has been employed by *Raissi et al.* with PINNs in [1]; anyway significantly worse estimation results have been achieved.

3. **Learning Rate**

$\mathbf{10^{-4}}$	$\mathbf{5 \cdot 10^{-4}}$	$\mathbf{10^{-3}}$	$\mathbf{5 \cdot 10^{-3}}$	$\mathbf{10^{-2}}$
1.13e-1	6.69e-2	6.26e-2	6.71e-2	8.90e-2

Table 5.11: Average activation maps relative errors in l^1-norm on the tets dataset using different learning rates in the time-series based ST-RB-DNN model (with its "best" architecture)

Table 5.11 reports the average activation maps relative errors in l^1-norm on the test dataset for different values of the learning rate ν. Recall that the learning rate is reduced by a factor of 4 on plateaus of the validation loss, for no more than 4 times; no exponential decay has instead been taken into account.

The best result has been obtained with $\nu = 10^{-3}$; also results with $\nu = 5 \cdot 10^{-3}$ and $\nu = 5 \cdot 10^{-4}$ are below the threshold of 7% and, thus, good. If, instead, the value of ν is taken too low (i.e. 10^{-4}) the training struggles to converge and it gets stuck in some less optimal local minima of the loss in the parameters' space. Conversely, if ν is chosen too high (i.e. 10^{-2}), then the optimization algorithm, which is Nadam in this case, is not able to well identify the positions of the loss functional minima and it is ultimately less precise.

4. **Regularization Parameter**

0	$\mathbf{10^{-6}}$	$\mathbf{10^{-5}}$	$\mathbf{10^{-4}}$	$\mathbf{10^{-3}}$	$\mathbf{10^{-2}}$	$\mathbf{10^{-1}}$
6.90e-2	7.51e-2	7.02e-2	6.26e-2	7.09e-2	8.52e-2	1.05e-1

Table 5.12: Average activation maps relative errors in l^1-norm on thetest dataset for different values of the regularization parameter in the time-series based ST-RB-DNN model (with its "best" architecture)

Table 5.12 reports the average activation maps l^1-norm relative errors on the test dataset for different values of the regularization parameter λ_r (see (5.3)). Recall that all fully-connected and $1D$-convolutional layers are subject to *Ridge Regularization* (i.e. l^2-norm penalty of weights and biases). Also, the reported parameter refers to the regularization level of the convolutional layers only, since in fully-connected ones a downscale of 10^{-3} is performed.

The best model is got for $\lambda_r = 10^{-4}$, which means that the regularization level is 10^{-4} on the entries of the convolutional kernels and of 10^{-7} on the weights and biases characterizing the fully-connected layers. In general, good results are achieved for all the values of λ_r smaller than or equal to 10^{-3}, while for $\lambda_r = 10^{-2}, 10^{-1}$ model performances are worse, producing an average error even higher than 10% in the latter case.

5. **Activation Function of the Epicardial Potential Estimator Layer**

ReLU	***Leaky-ReLU***	***ELU***	***SELU***	***Linear***
3.85e-1	7.36e-2	9.13e-2	6.26e-2	6.65e-2

Table 5.13: Average activation maps relative errors in l^1-norm on the test dataset for different activation functions of the epicardial potential estimator layer in the time-series based ST-RB-DNN model (with its "best" architecture)

Table 5.13 shows the average activation maps l^1-norm relative errors on the test dataset, obtained employing different activation functions in the epicardial potential estimator layer, i.e. the high-dimensional fully-connected layer responsible for the estimation of the Space-Time-Reduced epicardial extracellular potential.

The best results have been achieved employing the *Scaled Exponential Linear Unit (SELU)* function, i.e. a scaled version of *ELU* (see (5.7)) with scale factor $s > 1$. Also results achieved with *Leaky-ReLU* (see (5.6)) are good and the ones got with the simplest linear activation function (i.e. $a(x) = x$) are even better. Conversely, the outcome with *ELU* is worse, with an error almost reaching 10%. Finally, employing the "standard" *ReLU* as activation function, the majority of the Space-Time-Reduced epicardial potential entries are predicted to 0, as a consequence of the "Dying ReLU" problem discussed in Subsection 5.1.2; the relative error, then, explodes up to 38.5%.

6. **Number of Neurons Prior to the Epicardial Potential Estimator Layer**

	4	**8**	**16**	**32**	**64**
Error	9.85e-2	6.26e-2	6.20e-2	5.87e-2	6.79e-2
# **Trainables**	192′185	256′525	385′141	642′117	1′155′055

Table 5.14: Average activation maps relative errors in l^1-norm on the test dataset and model complexity for different numbers of neurons in the last post-convolutional fully connected layer, in the time-series based ST-RB-DNN model (with its "best" arcitecture)

Table 5.14 contains the average activation maps l^1-norm relative errors on the test dataset and the number of trainable parameters, corresponding to a different number of neurons in the last post-convolutional fully-connected layer. More specifically, the post-convolutional fully-connected layers have been designed to feature a number of neurons that decays exponentially, with a factor of 2, from a starting value of 256; thus if, for instance, we indicate that the number of neurons in the last layer is 32, then it means that we have considered 4 layers made of $256, 128, 64, 32$ neurons respectively.

First of all, it can be noticed how big is the impact of the considered hyperparameter on the overall model complexity, with the number of trainable parameters increasing from just 192′185 if 4 neurons are considered to 1′155′055 if 64 neurons are instead used. This is due

to two reasons. On the one side, the dimension of the subsequent fully-connected layer (i.e. the epicardial potential estimator layer) is always high and equal to $n_{st} = 619$ in the current case. On the other side, the number of neurons in the last layer preceding the epicardial potential estimation is not actually equal to the values reported in the Table 5.14, but to those values multiplied by $\left\lceil \frac{N_t - N_{mp} + 1}{L_{mp}} \right\rceil$, being L_{mp} the stride and N_{mp} the dimension of the window of the $1D$ *Max-Pooling* layer, that acts along the data temporal axis. In the case of interest, $N_t = 80$, $N_{mp} = 4$ and $L_{mp} = 3$, so the last layer is made of $26N$ neurons, being N the value inserted in Table 5.14. So, doubling the number of neurons (from N to $2N$) in the considered layer (still following the exponential-decaying design introduced before), the number of trainable parameters increases of $\Delta_N = \left\lceil \frac{N_t - N_{mp} + 1}{L_{mp}} \right\rceil N n_{st} - N - 2N^2$, where the first positive term considers the "new" weights related to the epicardial potential estimator layer, while the two negative ones account for the decrease in trainable parameters due to the removal of one post-convolutional layer.

Looking at the activation maps average relative errors, we can recognize a monotonically decreasing trend from $N = 4$ to $N = 32$, for which the best result of $5.87 \cdot 10^{-2}$ is achieved. Anyway, if $N = 64$, then the model becomes very complex and also the vast majority of its trainable parameters is localized in a single couple of fully-connected layers; as a result, it is more difficult for the NN to learn and it is much more likely for it to perform overfitting, ultimately producing worse estimates. Despite the best result is achieved for $N = 32$, the best compromise between model complexity and model performances is (heuristically) got for $N = 8$; indeed, with respect to the best case, the model complexity is more than halved, while model performances are improved just of 6.64%.

7. **Loss Composition**

ECG \ $\boldsymbol{u_{EST}}$	$\boldsymbol{MAE_\sigma}$	$\boldsymbol{MSE_\sigma}$
$\boldsymbol{MAE}$	7.16e-2	6.84e-2
$\boldsymbol{MSE}$	6.26e-2	6.81e-2

Table 5.15: Average activation maps relative errors in l^1-norm on the test dataset for different loss compositions in the time-series based ST-RB-DNN model (with its "best" arcitecture). In particular MAE and MSE on both ECG signals and Space-Time-Reduced epicardial potential have been considered, with a rescaling driven by the singular values (see (5.2)) on the latter

Table 5.15 reports the average activation maps l^1-norm relative errors on the test dataset, achieved for different compositions of the loss functional (5.1). In particular, the MAE and the MSE on both the ECG signals and the Space-Time-Reduced epicardial potential have been considered, with a rescaling driven by the singular values (see (5.2)) in the latter case. The weight w_{BC} of the epicardial potential has been re-calibrated every time, so that the contribution of ECG signals in the loss is of the order of 5%.

The best result has been achieved using the MSE on the ECG signals and the (scaled) MAE on the Space-Time-Reduced epicardial potential. Anyway, also the results of the other tested combinations are close, meaning that the choice of the composition of the loss functional does not seem to play a major role.

Autoencoder Model

As last test on the time-series-based ST-RB-DNN model in the framework of the first benchmark case, we have evaluated its performances by setting $w_{BC} = 0$ in the loss functional (5.1). This implies that, apart from regularization, the loss is built just as the MSE on the reconstructed ECG signals, so that the model attempts at estimating the epicardial extracellular potential acting as a pure autoencoder. The clear advantage of this approach is that the model can be

trained being completely unaware of the values of the epicardial potential.

Results, anyway, are significantly worse than the ones presented before; Figures 5.11 and 5.12 report the estimated activation map and the reconstructed ECG signals respectively, for the first test datapoint. While the errors on the ECG signals (measured as averaged in absolute l^1-norm) are comparable with the ones made by the "best" model ($6.06 \cdot 10^{-2}$ mV vs. $4.87 \cdot 10^{-2}$ mV), the estimated epicardial activation maps feature an average error (relative, in l^1-norm) of 40.63% and even qualitatively it is very difficult to reconstruct the propagation pattern of the depolarizing wavefront from it. More in detail, the predicted potential field exhibits a much higher variability both in space and in time with respect to the target one, with several small EBTs, rather than few larger ones, arising over the epicardial surface. Thus, the ST-RB-DNN model has not learned the actual epicardial activation pattern, but an alternative (and more complex) one, which anyway ends up bringing to a very similar outcome in terms of 12-lead ECG signals. The ill-posedness of the Inverse Problem of electrocardiography and the limited amount of information on the epicardial potential carried by the 12-lead ECG signals surely play a key role in this.

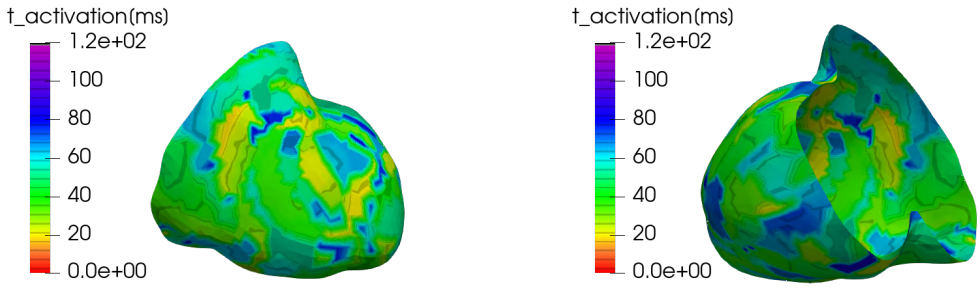
Several ways of reducing such errors could be put in place; in the context of this project we have taken two of them into account. The first one is quite straightforward and it is inspired by the classical Tikhonov regularization technique described in Subsection 4.3.2; it consists in inserting into the loss functional a novel regularization term, which explicitly penalizes the l^2-norm of the epicardial potential (zero-order) or the one of its spatial derivatives (first-order or second-order), at all time instants. In our case, anyway, what is estimated by the model is not the FOM approximation of the epicardial potential field, but its projection onto a Space-Time-Reduced subspace, computed starting from a set of FOM snapshots. Also, we would like not to expand the solution to the FOM space, in order to limit the memory occupation and the training time. Thus, as a very basic attempt, we simply inserted in the loss a term that penalizes the l^2-norm of the Space-Time-Reduced epicardial potential coefficients and that sums up, with a given weight, to the regularization terms relative to the model trainable parameters.

In particular, two different tests have been performed. In the first one, the plain l^2-norm of the Space-Time-Reduced epicardial potential has been penalized, with a loss weight of 10^{-4}; the average activation maps error decreased to 35.72%, but yet qualitatively very small improvements could be observed. In the second one, the loss has been instead enriched with the weighted l^2-norm of the epicardial potential coefficients, with weights given by $\{1 - w_i\}_{i=1}^{n_{st}}$, being

$$w_i = \sqrt{\frac{\sigma_j^{e,s}\sigma_k^{e,t,j}}{\sigma_1^{e,s}\sigma_1^{e,t,1}}} \quad such\ that:\ \mathcal{F}(k,j) = i \tag{5.17}$$

defined as in (5.2). The loss weight has been kept fixed to 10^{-4}. The idea is that, in this way, the coefficients associated to the least important basis elements are more penalized, hopefully mitigating the spurious and undesired oscillating components, that lead to such imprecise activation maps, as the one of Figure 5.11. Anyway, also in this case no significant improvements could be appreciated, with an average error on activation maps equal to 33.21%.

The second approach, instead, focuses on the fact that 12-lead ECG signals carry only a limited amount of information regarding the epicardial potential field, which is instead well encoded by BSPMs, measured via suitable electrodes vests (see Section 4.3). In particular, all recent relevant works in the framework of ECGI have been carried out starting from the measurement of the body surface potential at least in 100 different locations over the human torso, rather than just in 9, as we have done so far. Thus, we have decided to evaluate the performances of the ST-RB-DNN model on a second benchmark case, employing a more realistic geometry of the human torso (see Figure 5.6) and taking the body surface potential measured at 155 different locations as input. In this context, we have tested both the best time-series-based ST-RB-DNN model (i.e. the one derived via the previous grid search process) and the autoencoder one; the obtained results will be presented later, in Subsubsection 5.2.3.4.



(a) Frontal view

(b) Rear view

(c) Left Lateral view

(d) Inferior view

Figure 5.11: Epicardial activation maps obtained, on test datapoint 1, with the time-series-based ST-RB-DNN model trained as an autoencoder (i.e. $w_{BC} = 0$ in (5.1)) from four different perspectives (frontal, rear, left-lateral and inferior). The l^1-norm relative error equals $4.28 \cdot 10^{-1}$.

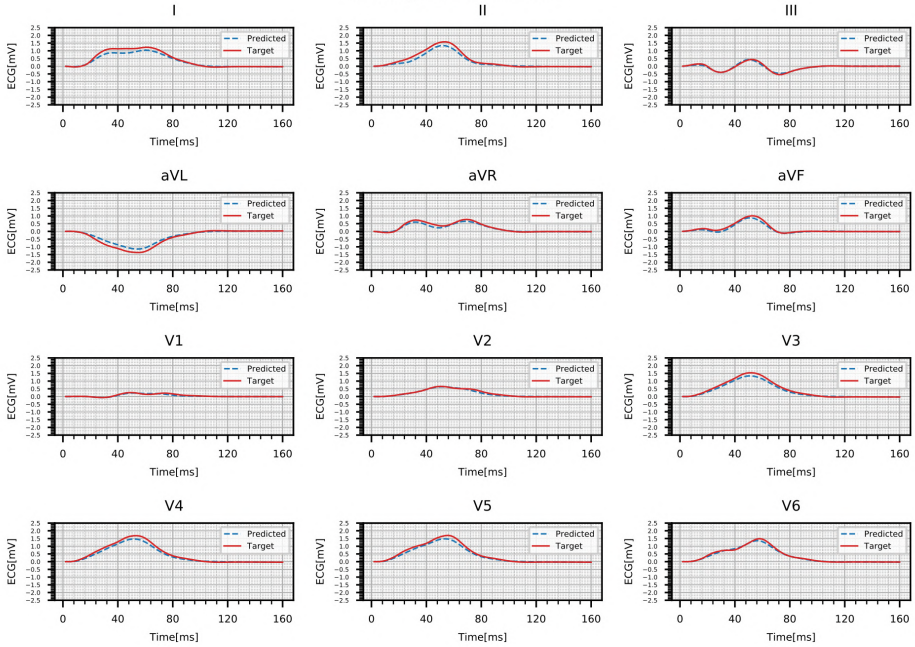


Figure 5.12: ECG signals obtained, on test datapoint 1, with the time-series-based ST-RB-DNN model trained as an autoencoder (i.e. $w_{BC} = 0$ in (5.1)). The Red solid line represents the ECG signal got with the ROM approximation; the Blue dashed line represents the ECG signal got with the considered time-series-based ST-RB-DNN model. The l^1-norm absolute error (averaged on all the leads) equals $5.60 \cdot 10^{-2}$ mV.

5.2.3.3 Test Case 1: The DFT-Based ST-RB-DNN Model

After having analyzed the performances of the time-series-based model, here we focus on the ones of the model which takes as input the lowest-frequency coefficients, arising from the application of a DFT to the body surface signals. All the analysis is carried out in the framework of the first idealized test case.

Grid Search

As before, we first perform a grid search involving the most relevant model hyperparameters; in this case, the considered hyperparameters are the number of neurons in the pre-flattening layers, the number of neurons in the post-flattening layers, the number of considered DFT coefficients and the weight of the Space-Time-Reduced epicardial potential in the loss functional w_{BC}. A total of 240 models has been tested. All the other model hyperparameters (POD tolerances, optimizer, learning rate, regularization parameters, dropout fraction, activation functions, training epochs, early stopping condition) have been kept fixed to the same values we adopted in the analysis of the time-series-based model, since those proved to give satisfactory results in some preliminary numerical tests.

Tables 5.16 - 5.17 - 5.18 report the results of the grid search, in terms of average l^1-norm relative errors on the epicardial activation maps, for three different values of the weight of the Space-Time-Reduced epicardial potential in the loss functional. **All errors are computed with respect to the optimal Space-Time ROM approximation.** Figures 5.13- 5.15 and 5.14-5.16 display the epicardial activation maps and the ECG signals reconstructed by the best model (in terms of activation maps average l^1-norm relative error) for the same test cases considered in the analysis of the time-series-based model.

DFT modes / **Dense**	9	17	25	33
(32) - (256,256,128,64,32,16)	1.69e-1	1.67e-1	1.59e-1	1.55e-1
(32) - (256,256,128,64,32)	1.54e-1	1.49e-1	1.49e-1	1.19e-1
(32) - (256,256,128,64)	9.63e-2	9.24e-2	9.65e-2	1.07e-1
(32) - (256,256,128)	8.05e-2	7.84e-2	7.94e-2	9.00e-2
(32) - (256,256)	8.20e-2	9.83e-2	8.47e-2	8.43e-2
(64) - (256,256,128,64,32,16)	1.68e-1	1.56e-1	1.75e-1	1.68e-1
(64) - (256,256,128,64,32)	1.28e-1	1.19e-1	1.19e-1	1.16e-1
(64) - (256,256,128,64)	1.08e-1	9.00e-2	8.81e-2	9.52e-2
(64) - (256,256,128)	8.19e-2	9.18e-2	7.52e-2	8.05e-2
(64) - (256,256)	8.03e-2	7.77e-2	8.39e-2	8.00e-2
(32,32) - (256,256,128,64,32,16)	1.79e-1	1.79e-1	1.60e-1	1.66e-1
(32,32) - (256,256,128,64,32)	1.38e-1	1.34e-1	1.26e-1	1.33e-1
(32,32) - (256,256,128,64)	1.06e-1	9.46e-2	8.36e-2	1.05e-1
(32,32) - (256,256,128)	9.37e-2	8.41e-2	8.74e-2	8.24e-2
(32,32) - (256,256)	8.26e-2	8.12e-2	8.56e-2	9.25e-2
(64,64) - (256,256,128,64,32,16)	1.79e-1	1.41e-1	1.60e-1	1.51e-1
(64,64) - (256,256,128,64,32)	1.26e-1	1.25e-1	1.06e-1	1.32e-1
(64,64) - (256,256,128,64)	9.19e-2	9.43e-2	8.34e-2	8.36e-2
(64,64) - (256,256,128)	8.09e-2	8.44e-2	7.70e-2	8.12e-2
(64,64) - (256,256)	7.06e-2	6.81e-2	7.46e-2	7.15e-2

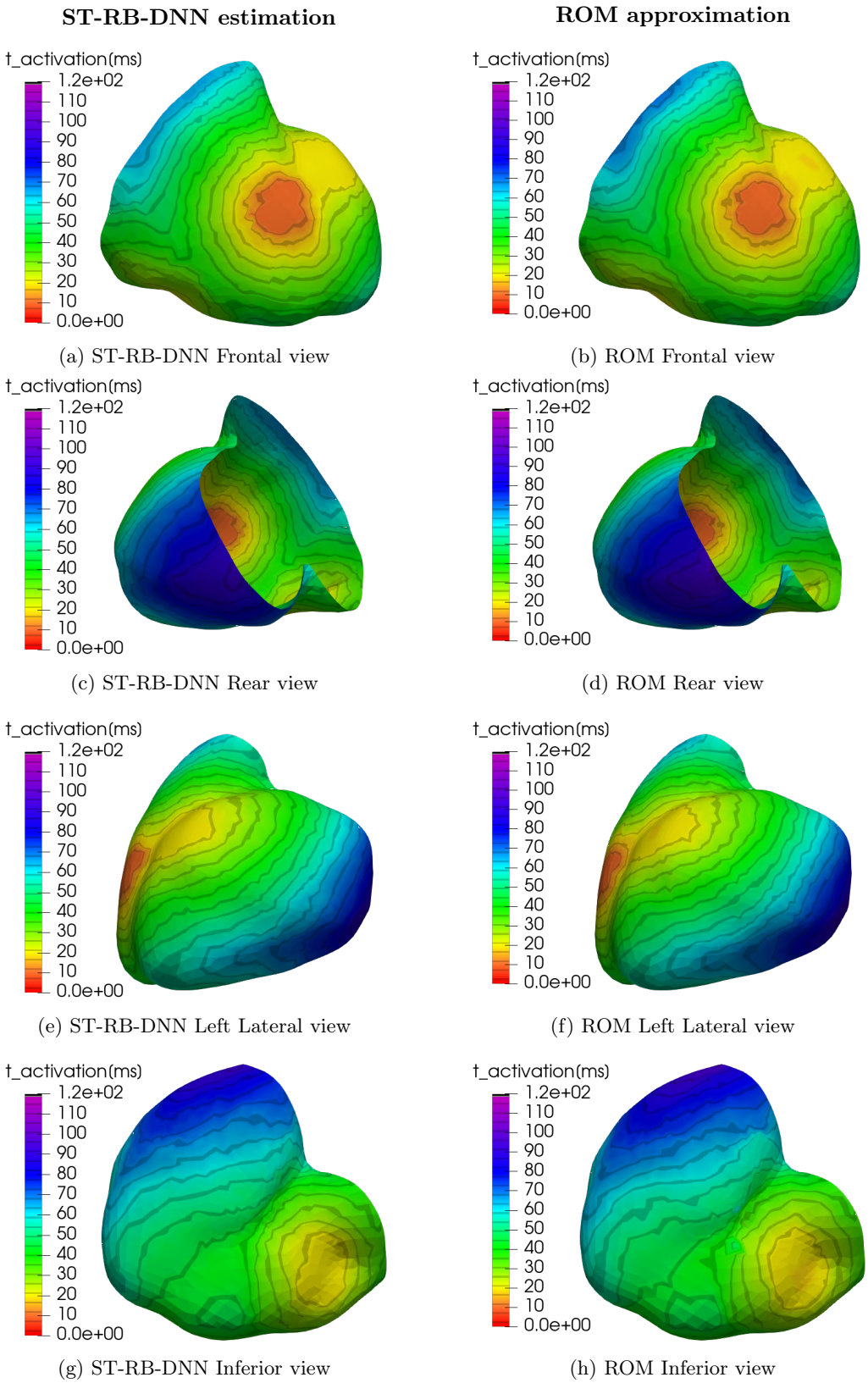
Table 5.16: Average relative errors in l^1-norm on the epicardial activation maps of the test dataset with the DFT-based ST-RB-DNN model, using epicardial potential **loss weight equal to 5**. The green cell displays the best model; the red cell displays the worst model; the yellow cells display the best 5 models (except from the very best one). Rows labels are of the form *Pre-Layers - Post-Layers*, where the first entry defines the layers of the pre-flattening fully-connected block and the second one the layers of the post-flattening fully-connected block. Columns labels represent the number of DFT coefficients given as input to the model.

Dense \ DFT modes	9	17	25	33
(32) - (256,256,128,64,32,16)	1.42e-1	1.55e-1	1.41e-1	1.53e-1
(32) - (256,256,128,64,32)	1.21e-1	1.28e-1	1.24e-1	1.21e-1
(32) - (256,256,128,64)	8.27e-2	8.87e-2	8.65e-2	9.70e-2
(32) - (256,256,128)	8.35e-2	8.49e-2	8.04e-2	8.51e-2
(32) - (256,256)	7.23e-2	7.36e-2	7.23e-2	7.78e-2
(64) - (256,256,128,64,32,16)	1.50e-1	1.43e-1	1.50e-1	1.66e-1
(64) - (256,256,128,64,32)	1.12e-1	1.09e-1	1.18e-1	1.08e-1
(64) - (256,256,128,64)	9.12e-2	7.94e-2	9.55e-2	8.61e-2
(64) - (256,256,128)	7.64e-2	8.89e-2	8.22e-2	7.20e-2
(64) - (256,256)	7.25e-2	7.15e-2	6.62e-2	6.78e-2
(32,32) - (256,256,128,64,32,16)	1.62e-1	1.64e-1	1.65e-1	1.54e-1
(32,32) - (256,256,128,64,32)	1.36e-1	1.41e-1	1.28e-1	1.26e-1
(32,32) - (256,256,128,64)	9.16e-2	8.69e-2	8.31e-2	9.59e-2
(32,32) - (256,256,128)	8.26e-2	8.27e-2	8.45e-2	7.53e-2
(32,32) - (256,256)	6.89e-2	7.26e-2	7.27e-2	7.89e-2
(64,64) - (256,256,128,64,32,16)	1.49e-1	1.42e-1	1.35e-1	1.62e-1
(64,64) - (256,256,128,64,32)	1.31e-1	1.03e-1	1.21e-1	1.06e-1
(64,64) - (256,256,128,64)	8.27e-2	7.81e-2	8.40e-2	8.00e-2
(64,64) - (256,256,128)	7.43e-2	7.07e-2	7.22e-2	7.83e-2
(64,64) - (256,256)	5.75e-2	6.00e-2	6.81e-2	6.18e-2

Table 5.17: Average relative errors in l^1-norm on the epicardial activation maps of the test dataset with the DFT-based ST-RB-DNN model, using epicardial potential **loss weight equal to 10**. The green cell displays the best model; the red cell displays the worst model; the yellow cells display the best 5 models (except from the very best one). Rows labels are of the form *Pre-Layers* - *Post-Layers*, where the first entry defines the layers of the pre-flattening fully-connected block and the second one the layers of the post-flattening fully-connected block. Columns labels represent the number of DFT coefficients given as input to the model.

Dense \ DFT modes	9	17	25	33
(32) - (256,256,128,64,32,16)	1.35e-1	1.27e-1	1.30e-1	1.46e-1
(32) - (256,256,128,64,32)	1.02e-1	1.08e-1	1.17e-1	1.05e-1
(32) - (256,256,128,64)	8.98e-2	7.84e-2	1.03e-1	9.15e-2
(32) - (256,256,128)	7.01e-2	7.68e-2	7.43e-2	7.35e-2
(32) - (256,256)	6.50e-2	7.36e-2	7.01e-2	6.85e-2
(64) - (256,256,128,64,32,16)	1.23e-1	1.35e-1	1.25e-1	1.17e-1
(64) - (256,256,128,64,32)	1.00e-1	1.06e-1	9.14e-2	1.06e-1
(64) - (256,256,128,64)	8.14e-2	9.08e-2	8.10e-2	8.84e-2
(64) - (256,256,128)	7.04e-2	6.84e-2	7.80e-2	6.80e-2
(64) - (256,256)	7.18e-2	6.81e-2	6.29e-2	6.48e-2
(32,32) - (256,256,128,64,32,16)	1.37e-1	1.43e-1	1.24e-1	1.10e-1
(32,32) - (256,256,128,64,32)	9.73e-2	1.10e-1	1.08e-1	1.07e-1
(32,32) - (256,256,128,64)	7.11e-2	1.07e-1	8.46e-2	7.50e-2
(32,32) - (256,256,128)	7.71e-2	7.50e-2	7.99e-2	7.36e-2
(32,32) - (256,256)	6.27e-2	5.54e-2	6.14e-2	7.70e-2
(64,64) - (256,256,128,64,32,16)	1.25e-1	1.48e-1	1.45e-1	1.50e-1
(64,64) - (256,256,128,64,32)	9.83e-2	1.04e-1	1.02e-1	1.02e-1
(64,64) - (256,256,128,64)	8.06e-2	7.74e-2	7.62e-2	8.32e-2
(64,64) - (256,256,128)	7.38e-2	6.86e-2	7.07e-2	6.78e-2
(64,64) - (256,256)	4.72e-2	5.55e-2	6.86e-2	4.92e-2

Table 5.18: Average relative errors in l^1-norm on the epicardial activation maps of the test dataset with the DFT-based ST-RB-DNN model, using epicardial potential **loss weight equal to 50**. The green cell displays the best model; the red cell displays the worst model; the yellow cells display the best 5 models (except from the very best one). Rows labels are of the form *Pre-Layers* - *Post-Layers*, where the first entry defines the layers of the pre-flattening fully-connected block and the second one the layers of the post-flattening fully-connected block. Columns labels represent the number of DFT coefficients given as input to the model.



(a) ST-RB-DNN Frontal view

(b) ROM Frontal view

(c) ST-RB-DNN Rear view

(d) ROM Rear view

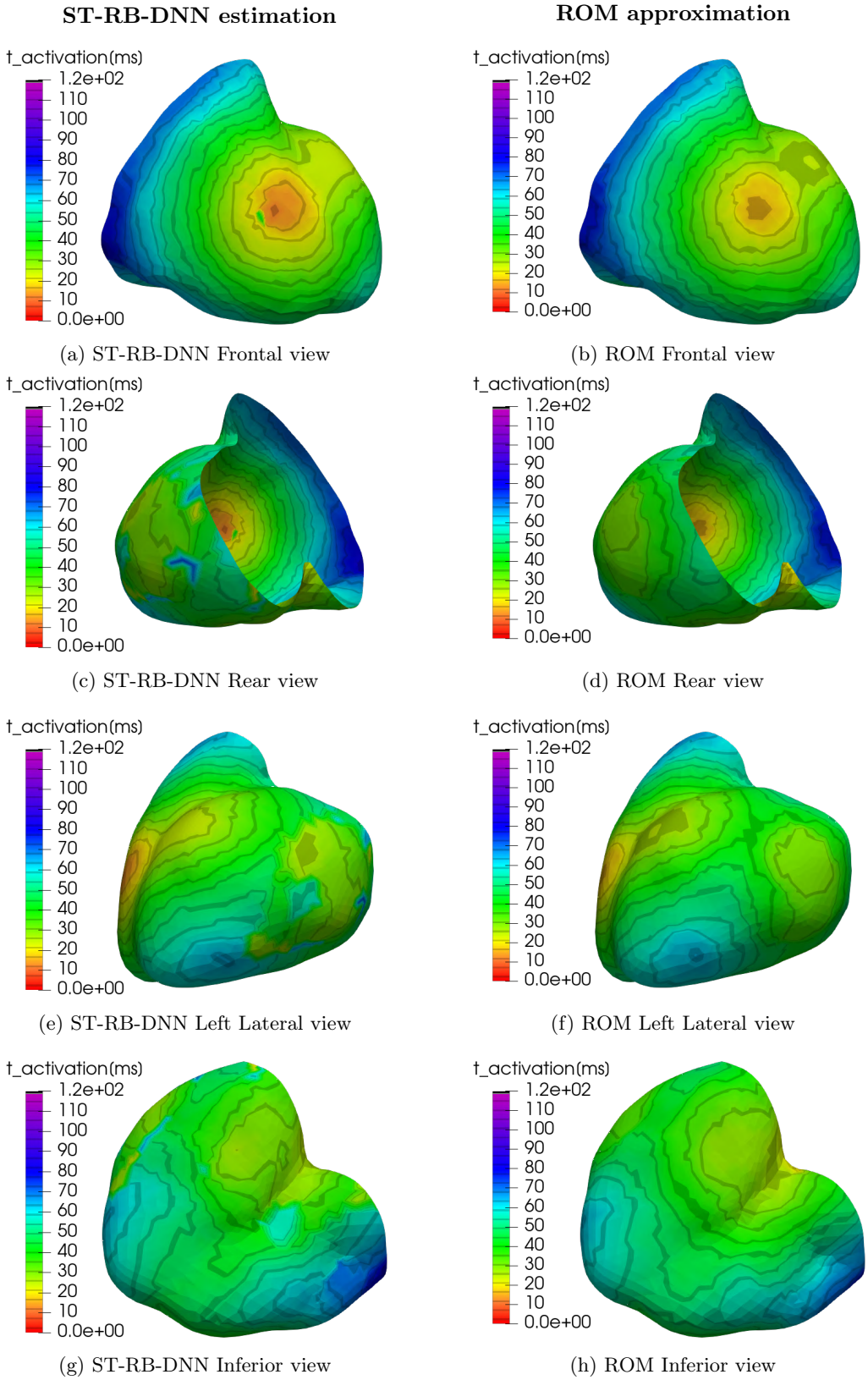
(e) ST-RB-DNN Left Lateral view

(f) ROM Left Lateral view

(g) ST-RB-DNN Inferior view

(h) ROM Inferior view

Figure 5.13: Epicardial activation maps obtained, on test datapoint 1, with the best DFT-based ST-RB-DNN model (left column) and with Space-Time ROM reconstruction (right column) from four different perspectives (frontal, rear, left-lateral and inferior). The l^1-norm relative error equals $2.49 \cdot 10^{-2}$.



(a) ST-RB-DNN Frontal view

(b) ROM Frontal view

(c) ST-RB-DNN Rear view

(d) ROM Rear view

(e) ST-RB-DNN Left Lateral view

(f) ROM Left Lateral view

(g) ST-RB-DNN Inferior view

(h) ROM Inferior view

Figure 5.14: Epicardial activation maps obtained, on test datapoint 2, with the best DFT-based ST-RB-DNN model (left column) and with Space-Time ROM reconstruction (right column) from four different perspectives (frontal, rear, left-lateral and inferior). The l^1-norm relative error equals $5.02 \cdot 10^{-2}$.

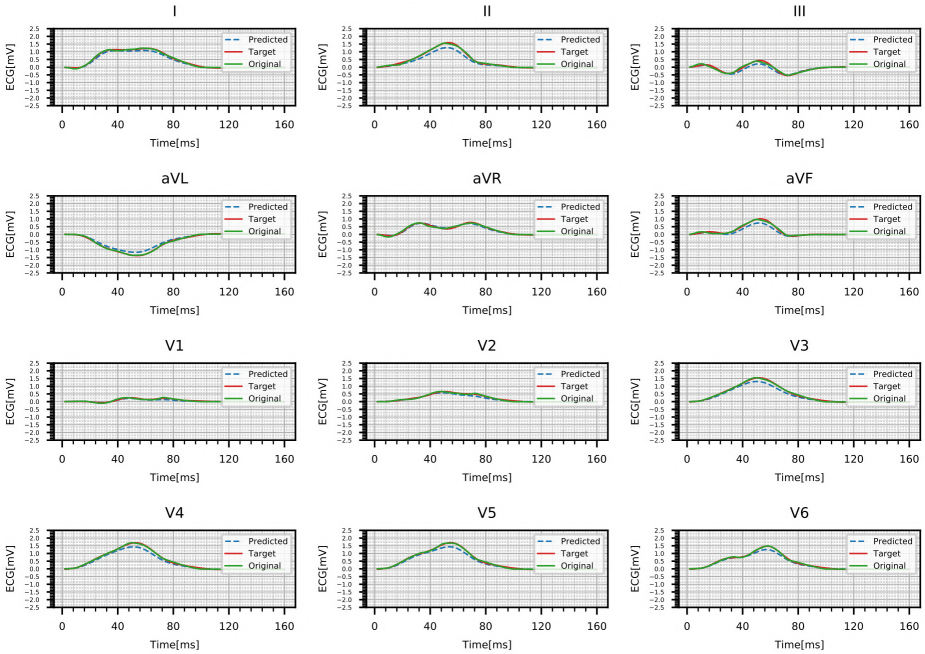


Figure 5.15: ECG signals obtained, on test datapoint 1, with the best DFT-based ST-RB-DNN model and with Space-Time ROM reconstruction. The Green solid line represents the ECG signal got with the ROM approximation; the Red solid line represents the ROM ECG signal, reconstructed via the selected DFT coefficients through an Inverse DFT; the Blue dashed line represents the ECG signal got with the best DFT-based ST-RB-DNN model. The l^1-norm absolute error (averaged on all the leads) equals $5.29 \cdot 10^{-2}\ mV$.

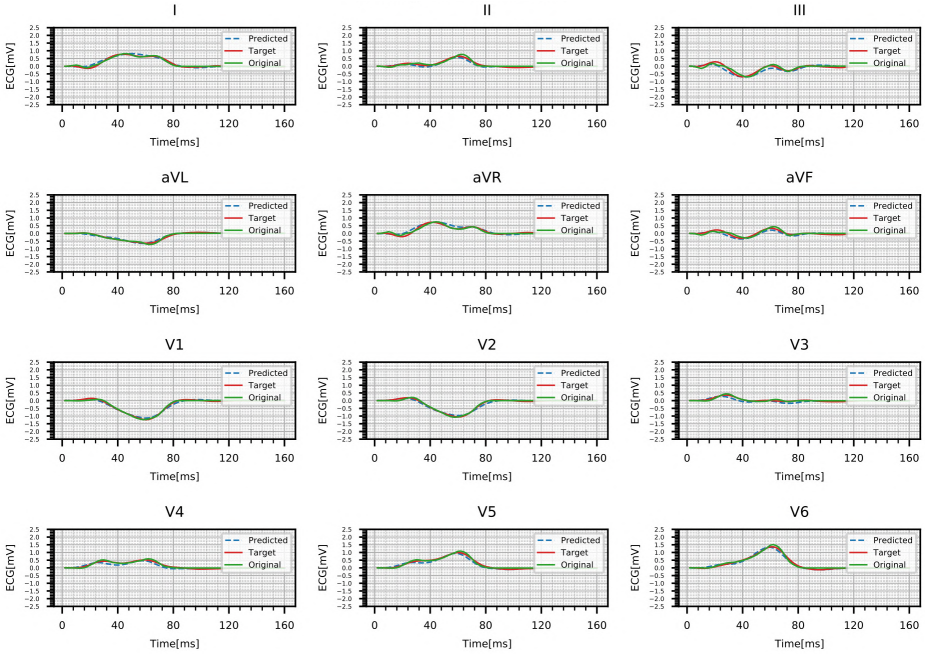


Figure 5.16: ECG signals obtained, on test datapoint 2, with the best DFT-based ST-RB-DNN model and with Space-Time ROM reconstruction.The Green solid line represents the ECG signal got with the ROM approximation; the Red solid line represents the ROM ECG signal, reconstructed via the selected DFT coefficients through an Inverse DFT; the Blue dashed line represents the ECG signal got with the best DFT-based ST-RB-DNN model. The l^1-norm absolute error (averaged on all the leads) equals $4.47 \cdot 10^{-2}\ mV$.

Looking at the results, the following considerations can be made:

- The first important observation is that the performances of the DFT-based ST-RB-DNN model are better than the ones of the time-series-based one; indeed, the best DFT-based model features a relative error on activation maps of 4.75%, which is significantly lower (-1.51%) than the one of best time-series based model (6.26%). Additionally, the complexities of the two best models are very similar, since the best time-series-based model has $256'525$ trainable parameters, while the best DFT-based one has $267'371$. This suggests that equipping the ST-RB-DNN model with information that already take into account the time evolution of the available signals makes it easier for it to learn the desired I/O mapping, allowing to get better results.
- If we focus on the trend with respect to the weight w_{BC} of the Space-Time-Reduced epicardial potential in the loss functional, we can observe that results tend to get better as such weight increases. Indeed, if $w_{BC} = 5$ (which leads to a loss split of approximately $60\% - 40\%$ between epicardial potential and ECG errors) then only one model features a relative error on activation maps that is lower than 7%, while if $w_{BC} = 10$ (loss split $\approx 75\% - 25\%$) the best model presents an error lower than 6% and if $w_{BC} = 50$ (loss split $\approx 95\% - 5\%$) the error goes even below the threshold of 5%. This outcome may imply that the contribution of the RB-solver layer is useless; in order to clarify this, we have made further tests, considering the best model architecture (Table 5.18), but increasing the weight w_{BC} to 100 and to 500. In such cases, the loss split is extremely unbalanced towards the epicardial potential, but the average relative errors on the activation maps we got are respectively $5.06 \cdot 10^{-2}$ and $5.89 \cdot 10^{-2}$. Thus, if the contribution of the RB-solver layer is very small, the model loses part of its physical awareness and its performances get worse.
- Concerning the hyperparameters related to the architecture of the MLP, a dramatic dependency with respect to the number of neurons in the last layer can be observed. Indeed the average activation maps errors, whichever the value of the loss weight and the number of input DFT coefficients, drop from values always above 10% in case the last layer is made of 16 nodes to values at most equaling 8.20% if it consists of 256 neurons. The justification for this is analogous to the one given while considering the time-series-based model. For fixed POD tolerances, the dimensionality of the last fully-connected layer is the parameter that mostly affects the network complexity; thus, by increasing it of just few units, we end up getting a much bigger and more powerful model that, if trained properly in order to avoid overfitting, could deliver better results. Starting from the best model architecture (Table 5.18), we have made other tests increasing the number of neurons in this last layer, up to 512; the resulting activation maps errors have always been of the order of 5%, thus telling that placing more than 256 neurons does not induce major improvements, while it severely affects model complexity. It may be also possible that a lower number of neurons (but higher than 128) is able to guarantee errors around the 5% threshold; no investigations have been made in this sense. Aside of this major effect, it can also be noticed that models tend to exhibit better performances as their complexities get higher. Indeed, keeping fixed the architecture of the post-flattening layers, the lowest errors are often got by placing two layers made of 64 neurons each in the pre-flattening block, which is the most complex scenario that has been taken into consideration in this grid search. Incidentally, the three best models (for the three different loss weights) all feature a pre-flattening block having structure $(64, 64)$ and a post-flattening block having structure $(256, 256)$.
- Finally, the investigation of the model performances with respect to the number of DFT coefficients (both given as input and returned in output) reveals that no dramatic differences are appreciable. More in detail, it can be observed that less complex models (i.e. models with few neurons in the last fully-connected layer) tend to have better performances if more DFT coefficients are involved, while more complex models deliver better results if just few DFT coefficients are considered. This can be explained by the fact that, if the model is too simple, then it benefits of being given more information, since its ability of extracting useful ones is somehow limited. Conversely, if the model is more complex, then

it manages to pull out of the input features that are useful to accomplish the ultimate predictive task at hand; equipping it with the knowledge of higher frequency components with low informative content only has a disturbing effect, hindering and slowing down the training process.

- **Test Datapoint 1**: numerical results achieved by the best DFT-based model on the first test datapoint are good, both in terms of epicardial activation maps (Figure 5.13) and of reconstructed ECG signals (Figure 5.15). In particular, the error on the epicardial activation map (relative in l^1-norm) is identical to the one got with the best time-series based model (i.e. $2.49 \cdot 10^{-2}$) and all the three EBTs are identified correctly, both in terms of positions and timings. Similarly, the predicted ECG traces are close, at all leads, to the target ones (red in Figure 5.15), that have been constructed by applying an Inverse DFT to the set of the 9 lowest-frequency coefficients considered in the model; the absolute l^1-norm error (averaged on all the leads) equals $5.29 \cdot 10^{-2}$ mV.
- **Test Datapoint 2**: as with the time-series based ST-RB-DNN model, also with the DFT-based one the results on the second test datapoint are worse, both in terms of epicardial activation map (Figure 5.14) and of reconstructed ECG signals (Figure 5.16). Focusing on the activation map, the l^1-norm relative error is lower than the one got with the best time-series based model ($5.02 \cdot 10^{-2}$ vs. $6.39 \cdot 10^{-2}$), indicating that the DFT-based models appear to have, globally, better performances. Also visually (Figures 5.8 and 5.14), it can be inferred a smaller prediction error; indeed, while the best time-series-based model fails in estimating the propagation of the depolarization wavefront in the inferior part of both ventricles and on the lateral margin of the left one, the DFT-based model only suffers of bad accuracy in the latter area. More specifically, both the ARV paraseptal and the ILV paraseptal basal EBTs are affected by the presence of small "noisy" regions around them, but the prediction errors are only evident on the lateral margin of the left ventricle, since the EBT occurring at around 30 ms is not precisely identified. This leak of precision in the EBT identification induces spurious oscillations in the predicted epicardial potential; those, in turn, give rise to "artificial" wavefronts that lead to time derivatives higher that the one due to the passage of the "target" wavefront, ultimately leading to "noise" in the activation map. As proposed in the analysis of the performances of the time-series-based model, either suitable filtering techniques to be applied to the estimated epicardial potential or more complex algorithms to derive the activation map could be developed in order the reduce these disturbing effects and to improve the overall model outcome. Regarding ECG signals, no significant reconstruction errors can be actually observed; anyway, as for the time-series-based model, the most problematic leads are III, aVF and aVR, since they "look" at the inferior and lateral areas of the left ventricle, that appear to be the most problematic ones to reconstruct. The absolute l^1-norm error (averaged on all the leads) equals $4.47 \cdot 10^{-2}$ mV

Effect of other model hyperparameters

In the following, we provide a brief analysis on the dependency of the DFT-based ST-RB-DNN model performances on other hyperparameters. We always start from the "best" model architecture derived before and we proceed by one hyperparameter at a time, in order to illustrate its effect.

1. **Epicardial Potential POD Tolerances**

$\hat{\epsilon}^{e,t}_{POD}$ \ $\epsilon^{e,s}_{POD}$	$\mathbf{10^{-1}}$	$\mathbf{5 \cdot 10^{-2}}$	$\mathbf{10^{-2}}$
$\mathbf{10^{-1}}$	9.17e-2	8.95e-2	9.23e-2
$\mathbf{5 \cdot 10^{-2}}$	4.72e-2	6.43e-2	6.08e-2
$\mathbf{10^{-2}}$	5.15e-2	7.11e-2	7.43e-2

Table 5.19: Average activation maps relative errors in l^1-norm on the test dataset for different values of the spatial and temporal POD tolerances on the epicardial potential in the DFT-based ST-RB-DNN model (with its "best" architecture)

$\hat{\epsilon}^{e,t}_{POD}$ \ $\epsilon^{e,s}_{POD}$	$\mathbf{10^{-1}}$	$\mathbf{5 \cdot 10^{-2}}$	$\mathbf{10^{-2}}$
$\mathbf{10^{-1}}$	174′851	210′574	259′666
$\mathbf{5 \cdot 10^{-2}}$	267′371	430′309	854′364
$\mathbf{10^{-2}}$	459′864	873′377	2′038′615

Table 5.20: Number of trainable parameters of the DFT-based ST-RB-DNN model (with its "best" architecture) for different values of the POD tolerances

Table 5.19 summarizes the average activation maps l^1-norm relative errors on the test dataset for different POD tolerances both in space and in time. Table 5.20 shows the number of trainable parameters of the considered models. The dimensionalities of the Space-Time Reduced Bases computed for the different POD tolerances can be found in Table 5.9.

As for the time-series-based model, the best result is achieved for $\epsilon^{e,s}_{POD} = 10^{-1}$ and $\hat{\epsilon}^{e,t}_{POD} = 5 \cdot 10^{-2}$, which leads to a Reduced Basis of dimension 619 and to a model featuring 267′371 trainable parameters. Again, on the one side models obtained at high POD tolerances offer worse performances compared to the ones of the best model (in terms of epicardial activation maps reconstruction), because there is less margin to compensate an estimation error on a relevant coefficient via estimation errors on the least relevant ones. On the other side, models characterized by low POD tolerances have to estimate a lot of coefficients and thus they feature a great number of hyperparameters; this entails that the training process is more complicated and, in turn, that the activation maps estimation is not optimal.

2. **Optimizer**

SGD	**AGD**	**SGD Nesterov**	**Adam**	**RMSProp**	**Nadam**	**L-BFGS**
5.54e-1	4.62e-1	4.42e-1	5.79e-2	5.73e-2	4.72e-2	2.55e-1

Table 5.21: Average activation maps relative errors in l^1-norm on the test dataset using different optimizers in the DFT-based ST-RB-DNN model (with its "best" architecture)

Table 5.21 reports the average activation maps l^1-norm relative errors in on the test dataset for the best model, trained using different optimizers.

As in the time-series-based case, the Nadam optimizer is the one that guarantees the best results in terms of activation maps errors. In general, all adaptive optimization algorithms (i.e. Adam, Nadam and RMSProp) exhibit good performances. Conversely, the standard SGD algorithm and its accelerated versions (i.e. AGD and SGD with Nesterov momentum) experience struggle in converging, as well as the full-batch L-BFGS algorithm.

3. **Learning Rate**

$\mathbf{10^{-4}}$	$\mathbf{5 \cdot 10^{-4}}$	$\mathbf{10^{-3}}$	$\mathbf{5 \cdot 10^{-3}}$	$\mathbf{10^{-2}}$	$\mathbf{5 \cdot 10^{-2}}$
1.16e-1	6.45e-2	4.72e-2	5.86e-2	5.67e-2	1.30e-1

Table 5.22: Average activation maps relative errors in l^1-norm on the test dataset using different learning rates in the DFT-based ST-RB-DNN model (with its "best" architecture)

Table 5.22 reports the average activation maps l^1-norm relative errors on the test dataset for different values of the learning rate ν. Recall that the learning rate is reduced by a factor of 4 on plateaus of the validation loss, for no more than 4 times; no exponential decay has instead been taken into account.

The best result has been achieved at the "intermediate" value of $\nu = 10^{-3}$. At lower learning rates (and especially for $\nu = 10^{-4}$) the training happens to get stuck in some shallow local minima of the loss functional in the trainable parameters' space, being unable perform a good optimization. Conversely, results at higher learning rates (up to $\nu = 10^{-2}$)

are still good, with the error on activation maps increasing just of $\approx 1\%$; the reduction of ν on plateaus of the validation loss surely helps in this sense. For $\nu = 5 \cdot 10^{-2}$, instead, the error is much larger, since the length of the step done by the Nadam optimization algorithm, despite being adaptive, is yet too big to properly detect the minima of the loss and also its reduction on plateaus of the validation loss is not effective.

4. **Regularization Parameter**

$\mathbf{0}$	$\mathbf{10^{-9}}$	$\mathbf{10^{-8}}$	$\mathbf{10^{-7}}$	$\mathbf{10^{-6}}$	$\mathbf{10^{-5}}$	$\mathbf{10^{-4}}$
4.50e-2	5.47e-2	5.03e-2	4.72e-2	6.61e-2	9.59e-2	1.99e-1

Table 5.23: Average activation maps relative errors in l^1-norm on the test dataset for different values of the regularization parameter in the DFT-based ST-RB-DNN model (with its "best" architecture)

Table 5.23 reports the average activation maps l^1-norm relative errors on the test dataset for different values of the regularization parameter λ_r (see (5.3)).

The best result is achieved without any regularization, i.e. setting $\lambda_r = 0$; in general, models featuring with low values of λ_r have good performances, with errors of the order of $4 - 5\%$. As the regularization parameter increases, instead, results tend to get worse and worse, with the average error on activation maps reaching 19.9% for $\lambda_r = 10^{-4}$. This is due to the fact that the contribution of the regularization term in the loss functional becomes too big, so that the model "cares" more at reducing the magnitudes of its weights and biases than at producing an output which is close to the target one.

5. **Activation Function of the Epicardial Potential Estimator Layer**

ReLU	***Leaky-ReLU***	***ELU***	***SELU***	***Linear***
3.27e-1	7.27e-2	6.45e-2	4.72e-2	6.04e-2

Table 5.24: Average activation maps relative errors in l^1-norm on the test dataset for different activation functions of the epicardial potential estimator layer in the DFT-based ST-RB-DNN model (with its "best" architecture)

Table 5.24 shows the average activation maps l^1-norm relative errors on the test dataset, obtained employing different activation functions in the epicardial potential estimator layer, i.e. the high-dimensional fully-connected layer responsible for the estimation of the Space-Time-Reduced epicardial extracellular potential.

As for the time-series-based model, the best result is got employing the *Scaled Exponential Linear Unit (SELU)* activation function, i.e. a scaled version of *ELU* (see (5.7)) with scale factor $s > 1$; the values of the parameters a and s have been set to the default ones of the *Tensorflow/Keras Python* package. Again, results got with improved versions of *ReLU* (as *ELU* and *Leaky-ReLU*) or even with a simple linear activation function, are good ($\approx 6 - 7\%$), since those are all able to circumvent the "Dying ReLU" problem and to allow the optimization algorithm to properly converge. Conversely, the usage of the standard *ReLU* activation function does not allow to achieve the desired convergence, with the vast majority of the Space-Time-Reduced epicardial potential coefficients being estimated as 0.

6. **Loss Composition**

$\boldsymbol{\mathcal{F}(ECG)}$ \ $\boldsymbol{u_{eST}}$	$\boldsymbol{MAE_\sigma}$	$\boldsymbol{MSE_\sigma}$
$\boldsymbol{MAE}$	5.45e-2	6.80e-2
$\boldsymbol{MSE}$	4.72e-2	7.26e-2

Table 5.25: Average activation maps relative errors in l^1-norm on the test dataset for different loss compositions in the DFT-based ST-RB-DNN model (with its "best" architecture). In particular MAE and MSE on both the DFT coefficients of ECG signals and Space-Time-Reduced epicardial potential have been considered, with a rescaling driven by the singular values (see (5.2)) on the latter

Table 5.25 reports the average activation maps l^1-norm relative errors on the test dataset, achieved for different compositions of the loss functional (5.1). In particular, the MAE and the MSE on both the DFT coefficients of ECG signals and the Space-Time-Reduced epicardial potential have been considered, with a rescaling driven by the singular values (see (5.2)) in the latter case. The weight w_{BC} of the epicardial potential has been re-calibrated every time, so that the contribution of ECG signals in the loss is of the order of 5%.

Results show that the best model is obtained using the MSE on the DFT coefficients of the 12-lead ECG signals and the MAE (weighted by the singular values - see (5.2)) on the Space-Time-Reduced epicardial potential coefficients. Also, it appears that using the (weighted) MAE, rather than the (weighted) MSE, on the epicardial potential gives better results, with average relative errors being $\approx 2\%$ lower. The choice of the loss metric on ECG signals, instead, seems to play a minor role.

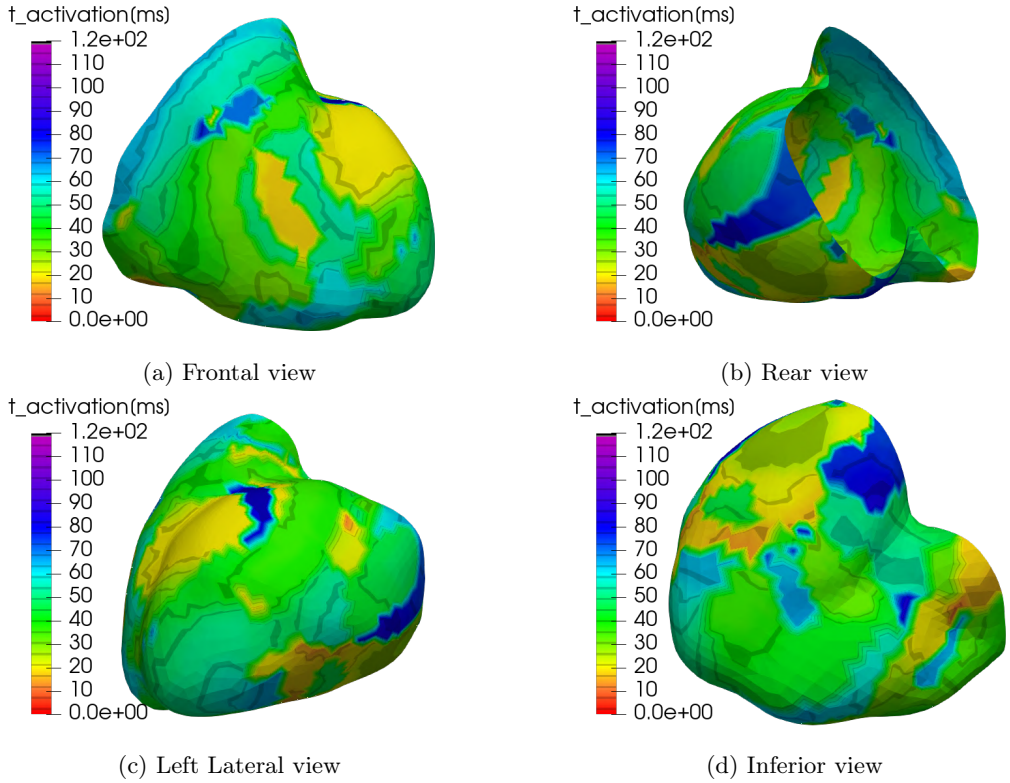
Autoencoder Model

The last test to be performed on the DFT-based ST-RB-DNN model is the analysis of its behavior in case it is trained as an autoencoder, i.e. if w_{BC} is set to 0 in (5.1).

Results are shown in Figures 5.17 and 5.18, which display the estimated epicardial activation map and the reconstructed ECG signals (respectively), for the first test datapoint. As in the time-series-based case, the outcome is not optimal. Indeed, on the one side the average errors on the reconstructed ECG signals are comparable to the ones done by the best DFT-based model ($6.42 \cdot 10^{-2}\ mV$ vs. $5.90 \cdot 10^{-2}\ mV$), entailing that the autoencoding process is carried out with discrete success. On the other side, anyway, the reconstruction of the epicardial activation maps is much worse; the relative l^1-norm average error on the test dataset equals 36.33% (on test datapoint 1 it is 39.66%) and even qualitatively it is difficult to infer from it the propagation pattern of the depolarization wavefront. The reasons that lead to such a bad estimation are basically analogous to the ones discussed for the time-series-based ST-RB-DNN model in the previous Subsubsection.

As in the time-series-based case, we have tried to reduce the epicardial potential estimation errors by inserting in the loss functional a regularization term, inspired by the Tikhonov regularization technique (see Subsection 4.3.2), which penalizes the l^2-norm of the Space-Time-Reduced epicardial potential coefficients. Actually, two tests have been executed. In the first one, the plain l^2-norm of the Space-Time-Reduced epicardial potential is penalized, with a loss weight of 10^{-4}; the average error on epicardial activation maps decreases down to 32.58%, but no major qualitative improvements can be noticed. The second test, instead, features a loss with a penalization of a weighted l^2-norm of the Space-Time-Reduced epicardial potential, with weights expressed as in (5.17) and with a loss weight again set to 10^{-4}. Anyway, results are again discouraging, with the average error on activation maps (38.23%) that increases with respect to the previous tests. Clearly, several other attempts could be made, at least for different values of the loss weight; anyway, in view of the bad results we got, we preferred not to investigate these regularization strategies any longer.

Rather, as discussed also in the analysis of the time-series-based model, we preferred to carry out additional tests in a different simulation setting, i.e. employing a more realistic geometry of the human torso (see Figure 5.6) and taking body surface potentials measured by 155 different electrodes, instead of the 12-lead ECG signals, as input. In this context, we have tested both the best DFT-based ST-RB-DNN model (derived via the grid search process) and the autoencoder one; the obtained results will be presented later, in Subsubsection 5.2.3.4.



(a) Frontal view

(b) Rear view

(c) Left Lateral view

(d) Inferior view

Figure 5.17: Epicardial activation maps obtained, on test datapoint 1, with the DFT-based ST-RB-DNN model trained as an autoencoder (i.e. $w_{BC} = 0$ in (5.1)) from four different perspectives (frontal, rear, left-lateral and inferior). The l^1-norm relative error equals $3.97 \cdot 10^{-1}$

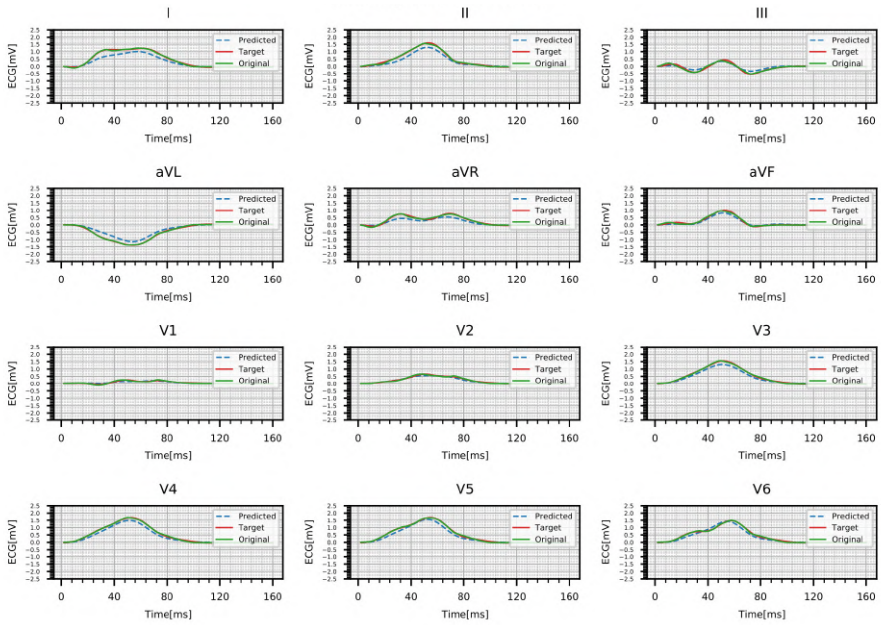


Figure 5.18: ECG signals obtained, on test case 1, with the DFT-based ST-RB-DNN model trained as an autoencoder (i.e. $w_{BC} = 0$ in (5.1)). The Green solid line represents the ECG signal got with the ROM approximation; the Red solid line represents the ROM ECG signal, reconstructed via the selected DFT coefficients through an Inverse DFT; the Blue dashed line represents the ECG signal got with the DFT-based ST-RB-DNN model. The l^1-norm absolute error (averaged on all the leads) equals $7.56 \cdot 10^{-2}$ mV

5.2.3.4 Test Case 2: Evaluation of the Best Models Performances

As anticipated in Subsubsection 5.2.3.1, the second test case features three main differences with respect to the first one:

1. The usage of a realistic geometry for the human torso (taken from [86] - see Figure 5.6), equipped with a more refined computational mesh made of $498'992$ tetrahedral cells
2. The construction of the I/O dataset from 158 leads (instead of just 12), which derive from recordings of the body surface potential made by 155 different electrodes, placed over the torso
3. The presence of just $N_{\boldsymbol{\mu}} = 180$ datapoints in the overall dataset (instead of 400). This is due to the much heavier computational effort required by the FOM simulations and it is expected to have a non-negligible impact on the model performances.

The simulation framework of the second test case represents a small step in the direction of considering more realistic scenarios where to evaluate the model performances. Anyway, we have still worked under several simplifying hypothesis (as the approximation of the torso as a homogeneous isotropic volume conductor, for instance), so that further efforts should be made in order to employ the model on clinical data.

While in the first test case we investigated the dependency of the implemented ST-RB-DNN models with respect to their main hyperparameters, in the second one we carried out just few different trainings. Specifically, we limited ourselves at evaluating the performances of the best time-series-based and DFT-based models, as identified via the grid search processes described in the two previous Subsubsections. For both models, three different "scenarios" have been considered and compared:

1. **Standard scenario**: as in the first test case, the model takes 12-lead ECG signals as input and the loss functional (besides of regularization terms) is built as the weighted average between the MSE on the reconstructed signals and the MAE on the Space-Time-Reduced epicardial extracellular potential.
2. **Augmented input scenario**: the model takes as input 158 signals, derived from measurements of the body surface potential made by 155 electrodes placed over the human torso, rather than 12-lead ECG signals. The loss is constructed as the usual weighted average, with weights tuned in order to get a $95\% - 5\%$ split between the MAE on the Space-Time-Reduced epicardial potential and the MSE on the torso signals.
3. **Augmented input & Autoencoder scenario**: the model takes as input 158 signals, derived from measurements of the body surface potential made by 155 electrodes placed over the human torso, and the loss functional is constructed just as the MSE on the reconstructed signals.

Remark: Since we have maintained the same specifics of the best models identified in the previous Subsubsections, we have also chosen the same tolerances for the PODs, i.e. $\epsilon_{POD}^{t,s} = 10^{-3}$, $\epsilon_{POD}^{e,s} = 10^{-1}$ and $\hat{\epsilon}_{POD}^{e,t} = 5 \cdot 10^{-2}$. On the considered dataset, made of $N_{\boldsymbol{\mu}} = 180$ datapoints, this has led to the generation of a Reduced Basis in Space on the torso potentials of dimension 80 and to a Reduced Basis in Space-Time on the epicardial potential of dimension 434 (68 basis functions in space, $\approx 6 - 7$ basis functions in time for each basis function in space).

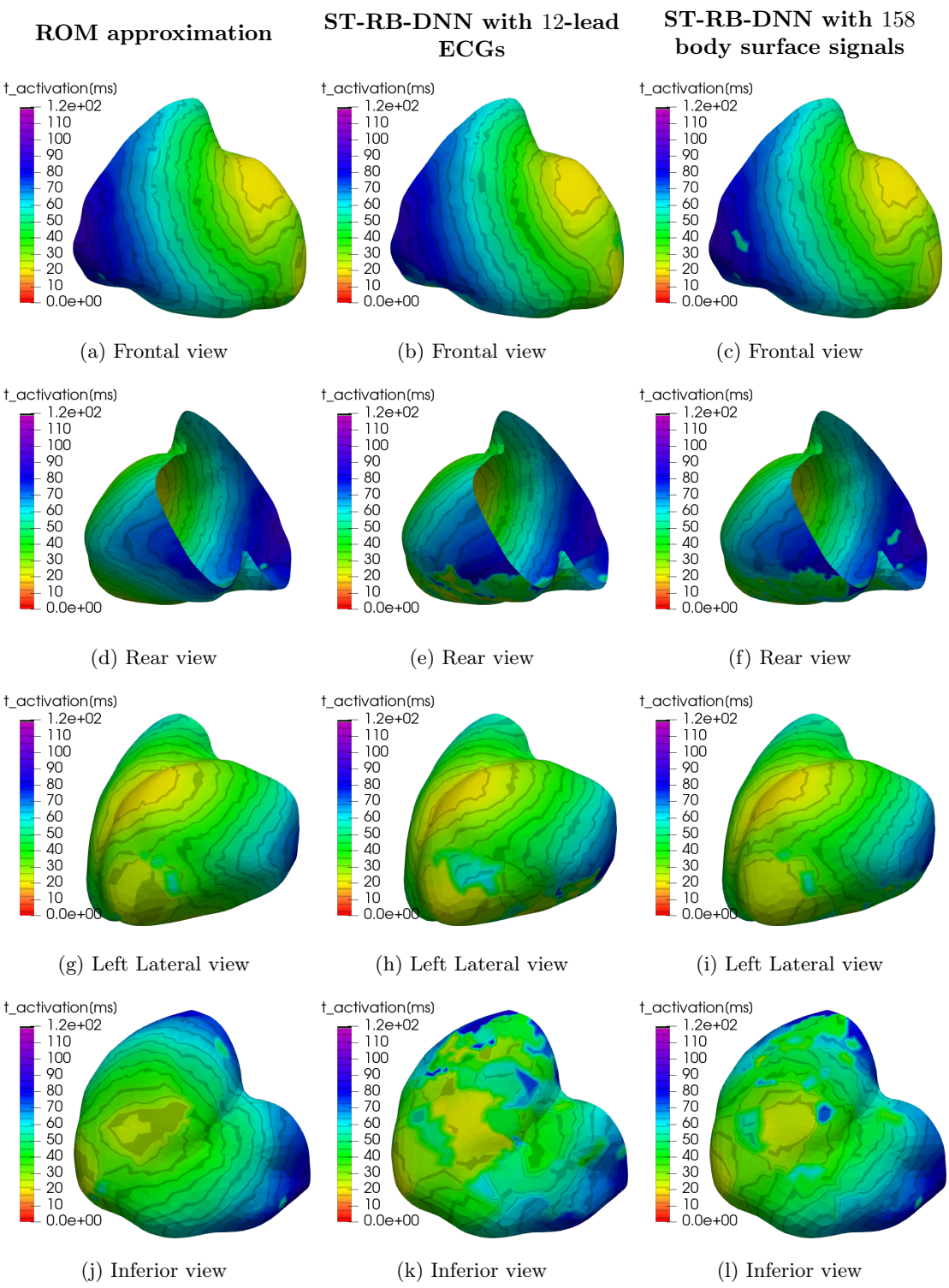
All errors are computed with respect to the optimal Space-Time ROM approximation

The Time-Series-Based ST-RB-DNN Model

Figure 5.19 shows the epicardial activation maps for the 9^{th} datapoint of the test dataset, which have been either obtained via the Space-Time ROM approximation or estimated by the (best) time-series-based ST-RB-DNN model. The model performances have been analyzed both if the input is generated from 12-lead ECG signals or from 158 signals, recorded by 155 electrodes placed over the torso. Figures 5.20 and 5.21 display the 12-lead ECG signals reconstructed by the two considered time-series-based ST-RB-DNN models, compared to the target ones obtained via the ROM approximation. Looking at the results, the following considerations can be made:

- The performances of the time-series-based ST-RB-DNN model which is given as input 158 signals are better than the ones of the model processing 12-lead ECGs. Indeed, the average activation maps l^1-norm relative error on the test dataset equals $6.73 \cdot 10^{-2}$ for the former and $7.91 \cdot 10^{-2}$ for the latter. This is expected, since BSPMs recorded via electrodes vests carry much more information on the epicardial potential field than 12-lead ECG signals
- The errors on the activation maps are higher if compared to the ones got in the first test case, even if 158 signals are given as input to the model. Despite the increased complexity of the FOM simulations may play a role, the main reason for this is that the dataset is made of less than half datapoints (180 vs 400). This should highlight how fundamental is, in DL applications, to have at disposal a big amount of training data, able to represent the vast majority of the possible dynamics that could occur in the system.
- The number of trainable parameters in the two models are similar ($228'040$ vs $217'860$). This is an important aspect to be underlined, as the model is designed to keep its complexity almost independent of the dimensionality of the datapoints to be processed.
- Looking at the epicardial activation maps of Figure 5.19, we can recognize how the performances of the model receiving in input 158 signals are better than the ones of the model which processes only 12 signals. Quantitatively, the l^1-norm relative error of the former equals $5.70 \cdot 10^{-2}$, while the one of the latter is $1.03 \cdot 10^{-1}$. In particular, a better estimation is evident in the inferior part of both ventricles (with the only EBT being localized in much more precise way both in space and in time) and at the apical region of the left ventricle (where the EBT is predicted in a less "noisy" way).
- Looking at the ECG signals of Figures 5.20 and 5.21, not so significant differences can be appreciated. Quantitatively, the average l^1-norm absolute reconstruction error on the signals equals $4.05 \cdot 10^{-2}\ mV$ for the model which is provided with 158 signals and $5.64 \cdot 10^{-2}\ mV$ for the one processing 12-lead ECGs; for the considered test datapoint the errors are $5.78 \cdot 10^{-2}\ mV$ and $8.27 \cdot 10^{-2}\ mV$ respectively. In both cases, the most "problematic" leads are V_1, V_2 and V_3 and the "new" model does not precisely estimate also III and aVR. Its error is anyway lower since it is averaged over all the 158 leads and not only over the 12 ones displayed here, whose approximation appears to be slightly more complicated.

Finally, we have trained the (best) time-series-based ST-RB-DNN model as an autoencoder which takes as input BSPMs. This trial configures as an upgrade of the ones discussed in the first test case: indeed the physically-aware deep autoencoder is expected to work better if it is provided with data that carry more information on the epicardial potential field than 12-lead ECG signals. Anyway, results are again discouraging. Despite the error on the reconstructed signals is close to the one done by the "best" model ($5.86 \cdot 10^{-2}$), the estimated potential fields feature several depolarizing wavefronts, that collide one with the other and who give rise to "noisy" epicardial activation maps. The average activation maps l^1-norm relative error on the test dataset equals $3.22 \cdot 10^{-1}$. Surely the reduced amount of data has an influence, but the good estimation of the torso signals suggests the problem being that different epicardial potential fields manage to give rise to very similar signals in the torso (i.e. ill-posed nature of the Inverse Problem of Electrocardiography). Thus, other than increasing the dataset dimensionality, more advanced strategies should be put in place to train the model as an autoencoder; in particular the manifold where to seek for the epicardial potential should be further restricted, somehow preventing solutions as the one of Figure 5.11 to be predicted.



(a) Frontal view (b) Frontal view (c) Frontal view

(d) Rear view (e) Rear view (f) Rear view

(g) Left Lateral view (h) Left Lateral view (i) Left Lateral view

(j) Inferior view (k) Inferior view (l) Inferior view

Figure 5.19: Epicardial activation maps obtained, on test datapoint 9 of the second test case, with the Space-Time ROM approximation (left column) and the time-series-based ST-RB-DNN model, being given as input both 12-lead ECG signals (central column) and signals recorded by 155 different electrodes placed on the torso (right column). The l^1-norm relative error equals $1.03 \cdot 10^{-1}$ for the former and $5.70 \cdot 10^{-2}$ for the latter.

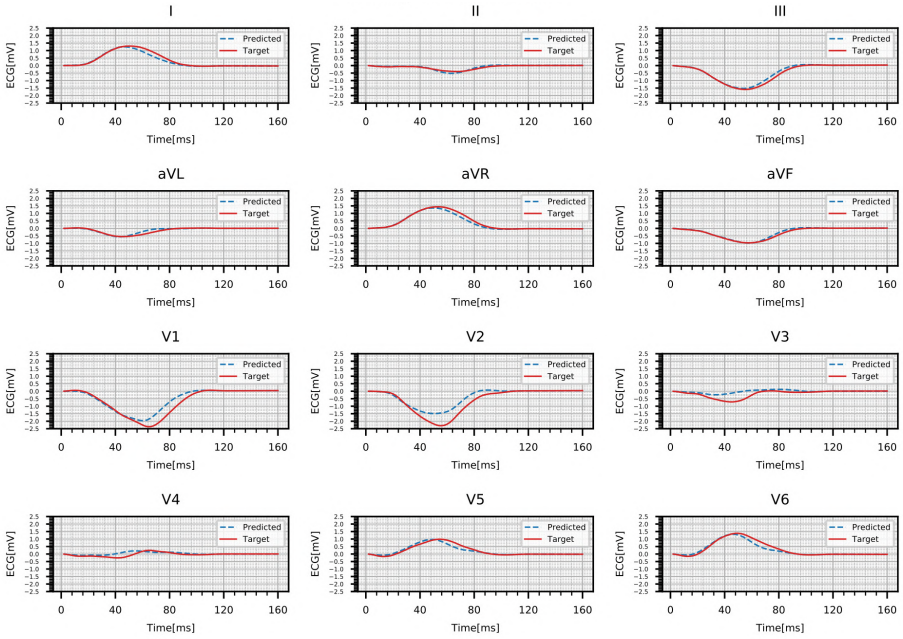


Figure 5.20: ECG signals obtained, on test datapoint 9 of the second test case, with the time-series-based ST-RB-DNN model, being given as input 12-lead ECG signals, and with Space-Time ROM reconstruction. The Red solid line represents the ECG signal got with the ROM approximation; the Blue dashed line represents the ECG signal got with the considered ST-RB-DNN model. The l^1 norm absolute error (averaged on all the leads) equals $8.27 \cdot 10^{-2}\ mV$.

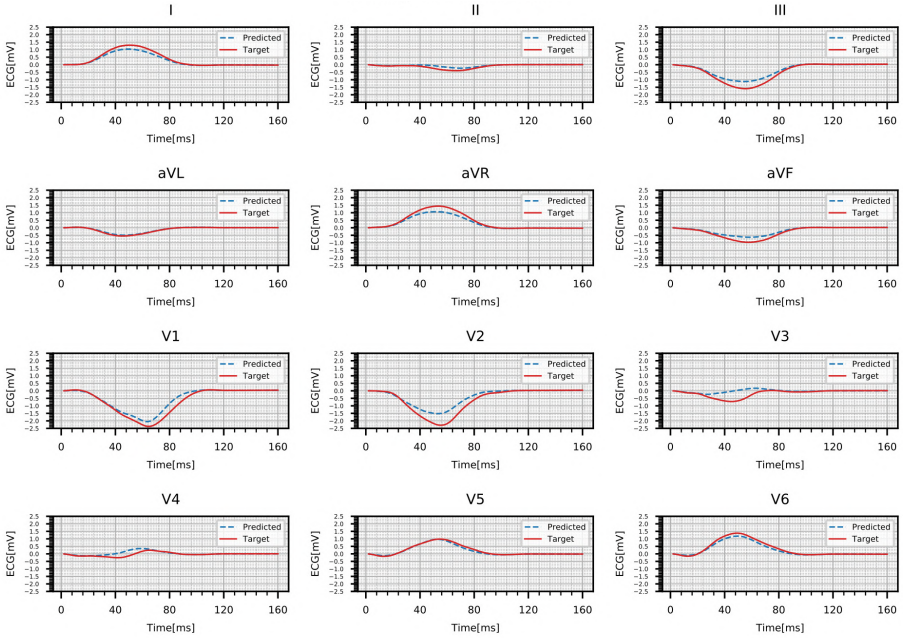


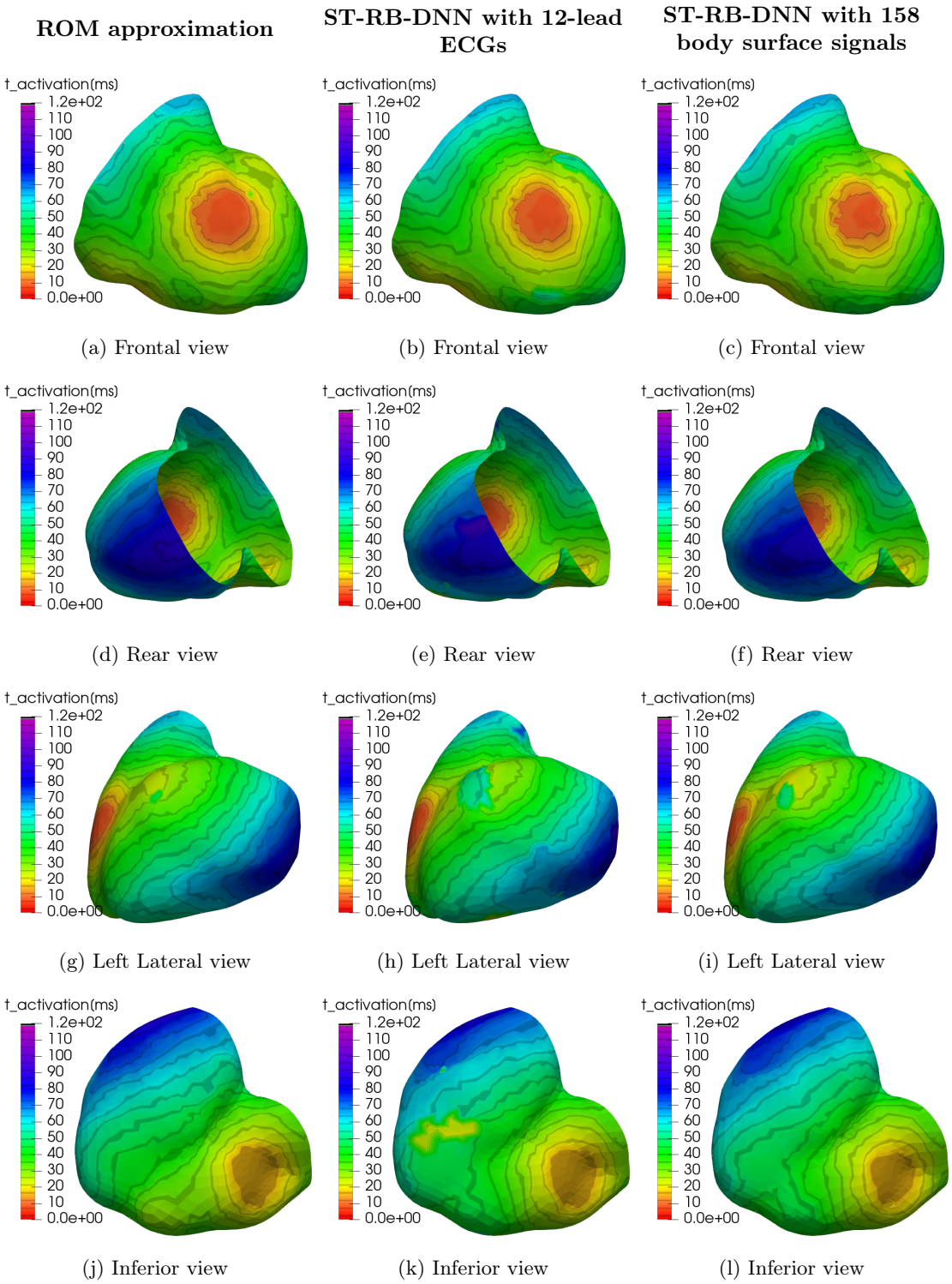
Figure 5.21: ECG signals obtained, on test datapoint 9 of the second test case, with the time-series-based ST-RB-DNN model, being given as input 158 signals recorded by 155 electrodes placed over the human torso, and with Space-Time ROM reconstruction.The Red solid line represents the ECG signal got with the ROM approximation; the Blue dashed line represents the ECG signal got with the considered ST-RB-DNN model. The l^1 norm absolute error (averaged on all the leads) equals $5.78 \cdot 10^{-2}\ mV$.

The DFT-Based ST-RB-DNN Model

Figure 5.22 shows the epicardial activation maps for the 2^{nd} datapoint of the test dataset, which have been either obtained via the Space-Time ROM approximation or estimated by the (best) DFT-based ST-RB-DNN model. The model performances have been analyzed both if the input is generated from 12-lead ECG signals or from 158 signals recorded by 155 electrodes placed over the torso. Figures 5.23 and 5.24 display the 12-lead ECG signals reconstructed by the two considered DFT-based ST-RB-DNN models, compared to the target ones obtained via the Space-Time ROM approximation. Looking at the results, the following considerations can be made:

- The performances of the DFT-based ST-RB-DNN model whose input is generated from the signals recorded by 155 electrodes are better than the ones of the model which processes the lowest-frequency DFT coefficients of 12-lead ECG signals. In particular, the average activation maps l^1-norm relative error on the test dataset of the former is $7.24 \cdot 10^{-2}$, while the one of the latter is $8.02 \cdot 10^{-2}$. As discussed before, the trend is expected as BSPMs are more informative than 12-lead ECG signals in terms of epicardial potential fields.
- The performances of both DFT-based models are worse than the ones of the corresponding time-series-based ones. This is opposite to what happened in the first test case and it suggests that the choice between one model or the other may severely depend on the dataset at disposal (both in terms of the type of data that it stores and of its dimensionality).
- Also the DFT-based ST-RB-DNN model has been designed to have a complexity which is almost independent of the dimensionality of the input datapoints. Indeed, the number of trainable parameters of the two considered models are $219'826$ (if 12-lead ECGs are processed) and $263'862$ (if 158 signals are considered).
- Looking at the epicardial activation maps of Figure 5.22, it is possible to recognize the best performances of the model processing 158 signals with respect to the ones of the model receiving 12-lead ECGs in input. Quantitatively, the l^1-norm relative error of the former is $5.18 \cdot 10^{-2}$, while the one of the latter equals $6.95 \cdot 10^{-2}$. Qualitatively, instead, a better estimation can be observed on the left ventricle, especially at the free wall, in the anterior paraseptal area (where the EBT is better localized) and in the inferior basal region (where the model processing 12-lead ECGs predicts an "artificial" elongated EBT).
- Focusing on signals reconstruction, again the model processing 158 signals features lower l^1-norm absolute errors, both averaged on the test dataset ($4.50 \cdot 10^{-2}\ mV$ vs $6.21 \cdot 10^{-2}\ mV$) and for the test datapoint considered in Figures 5.23 and 5.24 ($4.09 \cdot 10^{-2}\ mV$ vs $6.26 \cdot 10^{-2}\ mV$). Qualitatively, very small differences can be anyway appreciated, with both models being able to well reconstruct the limb leads, but experiencing more difficulties on the chest ones.

Finally, also in this case we have tried to train the model as a deep autoencoder which takes as input the lowest-frequency DFT coefficients of 158 leads, computed from the signals recorded by 155 electrodes placed over the torso. As we could expect, results have been discouraging, showing a good reconstruction of the signals on the torso (average l^1-norm absolute error: $4.27 \cdot 10^{-2}$), but a very bad estimation of the epicardial potential fields (average l^1-norm relative error: $2.13 \cdot 10^{-1}$). The same considerations made for the time-series-based model do hold and the need of putting in place more sophisticated techniques to regularize the predicted epicardial potential fields does emerge.



(a) Frontal view (b) Frontal view (c) Frontal view

(d) Rear view (e) Rear view (f) Rear view

(g) Left Lateral view (h) Left Lateral view (i) Left Lateral view

(j) Inferior view (k) Inferior view (l) Inferior view

Figure 5.22: Epicardial activation maps obtained, on test datapoint 2 of the second test case, with the Space-Time ROM approximation (left column) and the DFT-based ST-RB-DNN model, whose input is generated both from both 12-lead ECG signals (central column) and from signals recorded by 155 different electrodes placed on the torso (right column). The l^1-norm relative error equals $6.95 \cdot 10^{-2}$ for the former and $5.18 \cdot 10^{-2}$ for the latter.

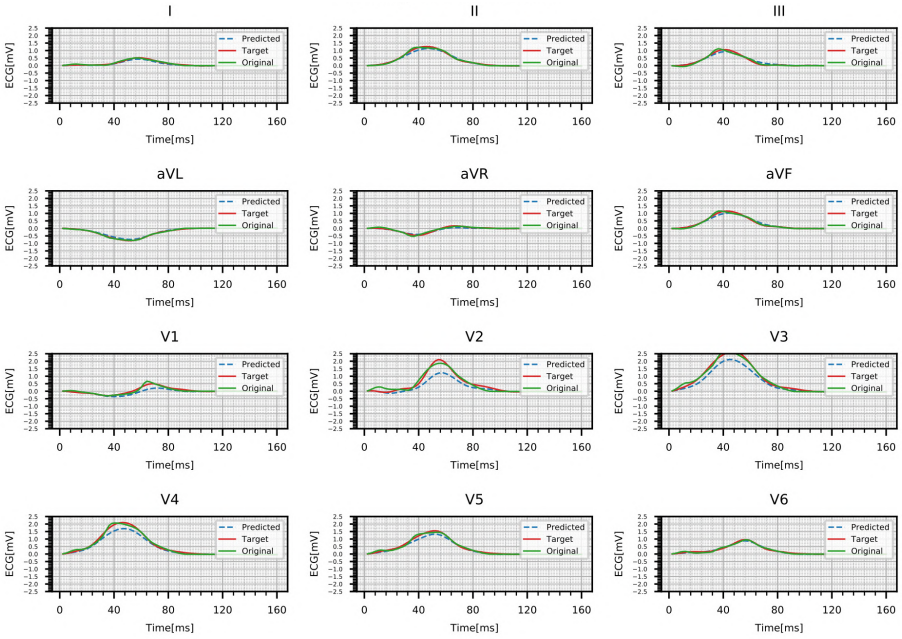


Figure 5.23: ECG signals obtained, on test datapoint 2 of the second test case, with the DFT-based ST-RB-DNN model, whose input is generated from 12-lead ECG signals, and with Space-Time ROM reconstruction. The Green solid line represents the ECG signal got with the ROM approximation; the Red solid line represents the ROM ECG signal, reconstructed via the selected DFT coefficients through an Inverse DFT; the Blue dashed line represents the ECG signal got with the considered DFT-based ST-RB-DNN model. The l^1-norm absolute error (averaged on all the leads) equals $6.26 \cdot 10^{-2}\ mV$.

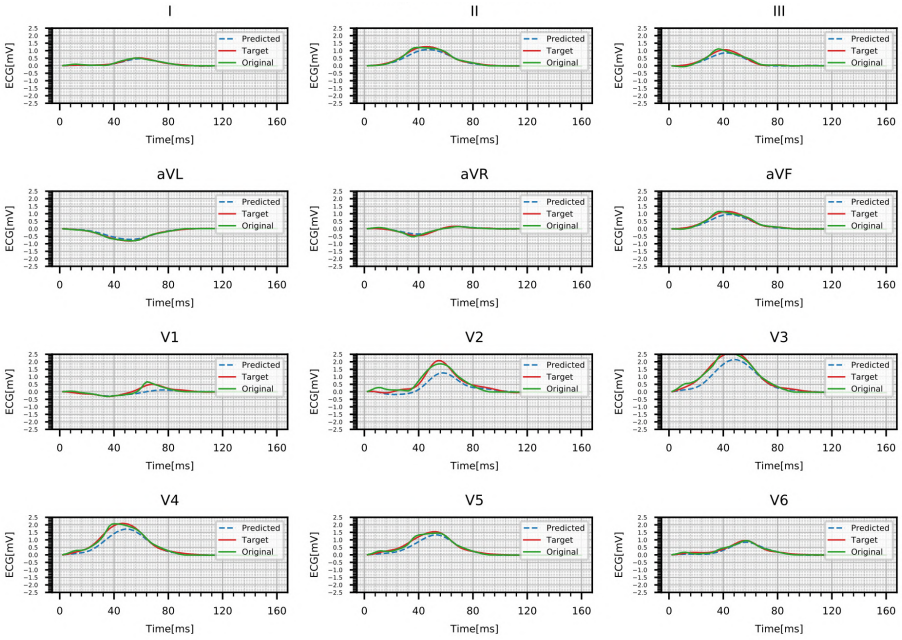


Figure 5.24: ECG signals obtained, on test datapoint 2 of the second test case, with the time-series-based ST-RB-DNN model, whose input is generated from 158 signals recorded by 155 electrodes placed over the human torso, and with Space-Time ROM reconstruction. The Green solid line represents the ECG signal got with the ROM approximation; the Red solid line represents the ROM ECG signal, reconstructed via the selected DFT coefficients through an Inverse DFT; the Blue dashed line represents the ECG signal got with the considered DFT-based ST-RB-DNN model. The l^1-norm absolute error (averaged on all the leads) equals $4.09 \cdot 10^{-2}\ mV$.

Chapter 6

Conclusion

The last chapter is devoted to a final overview of the project, summarizing its main motivations, the methods that have been put in place and the most relevant findings and results. Additionally, the major limitations are underlined and some possible ways of overcoming those are proposed.

In detail, Section 6.1 features the conclusive summary of the project, while Section 6.2 discusses its main limitations and proposes some possible further developments.

6.1 A Conclusive Summary

In this project we tried to develop a PDE-aware DL model, able to provide physically-consistent and data-driven solutions to Inverse Problems in cardiac electrophysiology.

Two kinds of motivations guided us. On the one side, there is the clinical need of estimating, via simple and non-invasive procedures, data that nowadays can be collected only via intrusive and expensive measurements. On the other side, instead, there is the mathematical ambition of coming up with DL models that manage to learn relevant mappings even from a small or imprecise amount of data, by exploiting the knowledge of some of the physical laws underlying the phenomenon of interest. Indeed, as said alongside the thesis, the fields of Deep Learning and of Numerical Approximation of PDEs/ODEs have undergone an extensive development in the last 50 years; anyway, a clean separation margin between those has been maintained, at least until the last decade. In recent years, indeed, some works aimed at developing DL models that manage to ease and speed up their training, by leveraging the knowledge of the physics of the problem at hand via PDEs, started to appear (e.g. [1,2]). Such models proved to be successful in the so-called *small data regime*, i.e. in situations where the amount of data at disposal is either limited or imprecise, but an accurate description of some of the physical laws characterizing the problem is present. The Inverse Potential Problem of Electrocardiography is a context of such kind; thus making a first attempt in this direction appeared both challenging and reasonable.

After having compensated the leak of data by numerically approximating the heart EP and the Forward Problem of Electrocardiography (see Section 4.2), we developed a physical-aware DL model for Inverse Problems in cardiac EP, called ST-RB-DNN. The DL part is due to the fact that the model learns, via classical DL paradigms and techniques, the epicardial extracellular potential, from signals measured in a finite number of points on the human torso; this constitutes the "Trainable NN" portion of the model, whose actual architecture depends on the nature of the signals to be processed. The physical-awareness, instead, is achieved in two ways:

1. Estimating the epicardial potential as projected onto a Space-Time-Reduced subspace, generated via PODs from solutions to the heart EP problem
2. Reconstructing in output (and penalizing in the loss functional) the signals given as input, by solving the ROM-in-Space FOM-in-Time Forward Problem of Electrocardiography, inside a deterministic layer, using the predicted epicardial potential as Dirichlet boundary

datum

Both elements allow to shrink the space of admissible solutions to a lower-dimensional and physically-consistent manifold, easing and speeding up the overall training process.

In the numerical tests we conducted, we considered two different models:

1. The time-series-based ST-RB-DNN model, which processes raw signals, organized in the form of time series, by leveraging temporal convolution
2. The DFT-based ST-RB-DNN model, which processes the M lowest-frequency coefficients, arising from the application of a DFT to the original input signals. M is a model hyper-parameter

Also, we tested the aforementioned models in two different settings: a simplified one, employing an idealized geometry and a coarse mesh for the human torso and providing 12-lead ECG signals in input, and a slightly more realistic one, where we used a human-shaped geometry and a refined mesh for the torso and we provided the model with 158 signals, generated from the measurements of 155 different electrodes.

The reduced computational effort required by the FOM simulations, the generation of the Reduced Bases and the model training in the first test case allowed us to carry out an extensive cost-benefit analysis. For both the time-series-based model and the DFT-based one, we performed several trainings, for different values of the most important hyperparameters. Ultimately, we derived a set of optimal hyperparameters, which allowed us to identify the "best models". They both exhibited good performances, attaining l^1-norm relative errors of the order of $4-6\%$ on the activation maps of the test dataset and featuring low complexities (which allowed them to be trained in $\approx 30-45\ min$ in a computational environment as the one described in Appendix C). Actually, the best DFT-based model managed to attain an average l^1-norm relative error on the activation maps equal to $4.72 \cdot 10^{-2}$, while the best time-series-based one only reached an error of $6.26 \cdot 10^{-2}$. Finally, we tried to train the models (with their "best" architecture) as pure autoencoders, removing the (weighted) MAE on the Space-Time-Reduced epicardial potentials from the loss. Results have been anyway discouraging, as both models managed to well reconstruct the input signals, but they estimated "noisy" and "non-physical" activation maps, characterized by the presence of many depolarizing wavefronts, colliding one with the other.

The increased computational burden characterizing the second test case (especially in terms of FOM simulations and of Reduced Bases generation) forced us generate a smaller dataset and to perform fewer trainings of the ST-RB-DNN models. In particular, we have fixed the architectures of both the time-series-based model and the DFT-based one, employing the optimal sets of hyperparameters identified in the first test case, and we compared the performances of the models in two scenarios. In the first one, they receive in input 12-lead ECGs, while in the second one they process 158 signals, that are generated from the measurement of the electric potential in 155 different locations on the body surface. Results clearly revealed that providing the model with more informative data on the epicardial potential field allows to improve its performances; in both cases, indeed, the l^1-norm relative error on the activation maps decreased when passing from 12 input signals to 158 (-1.18% and -0.78% for the time-series-based model and the DFT-based one, respectively). Errors, anyway, were higher if compared to the ones got in the first test case; this is mainly due to the much reduced dimensionality of the input dataset. Finally, also in this case we tried to train the models as pure autoencoders, with the hope that providing in input more informative data would allow to better reconstruct the epicardial potential field, without involving it in the loss. Anyway, results have been similar to the ones got in the first test case, thus suggesting more sophisticated techniques to be put in place to regularize the estimated Space-Time-Reduced epicardial potential and to further shrink the physically-consistent manifold where the model seeks for a solution to the Inverse Problem.

6.2 Limitations and Further Developments

As pointed out throughout the report, the current project proposes as a first methodological attempt of tackling the Inverse Potential Problem of Electrocardiography, both taking advantage of DL techniques and of the knowledge of the problem physics, expressed by means of suitable PDEs/ODEs. In view of this, it is reasonable that it features some limitations, which prevent it from being used in the actual clinical setting on real data.

In the following, the main limitations of the project are listed and discussed; also, some ways of overcoming those, configuring as possible further developments, are presented.

- **Simplified Dataset**: the training of the ST-RB-DNN model, unless it is designed as a pure autoencoder, requires lots of epicardial activation maps to be available. As discussed in Chapters 4 and 5, nowadays this is possible only via invasive measurement techniques and it is then unfeasible to build a dataset of the dimensions required by standard DL applications. Because of this, we have been forced to train and test our models with "artificially generated data", i.e. by numerically approximating the heart electrophysiology (via the bidomain equations, coupled with the AP ionic model) and the Forward Problem of Electrocardiography (via a generalized Laplace equation) - see Problem 4.1. Furthermore, since the project configures as a methodological proof-of-concept, we decided to restrict ourselves to simplified settings. Thus, we employed coarse meshes (apart from the torso one in the second test case), made just of some thousands of DOFs both for the heart and for the human torso (see Figures 4.8 and 4.9) and we neglected the presence of conducibility inhomogeneities, for instance due to the presence of different organs (see Subsection 4.2.3). Despite having made some efforts in reproducing signals exhibiting the expected polarities and amplitudes (see Subsection 4.1.2 and, in particular, Table 4.1 for some reference values in this sense), the results we obtained are far from being realistic. As a consequence, the ST-RB-DNN models we presented cannot be employed with success on real data. In view of a possible clinical application, it is then evident the need for the development of a fast and accurate solver (even on relatively coarse meshes) for both the Heart Electrophysiology and the Forward Problem of Electrocardiography; in this way, a variety of realistic scenarios could be numerically reproduced in a reasonable amount of time and the ST-RB-DNN model could learn from data which are similar to the ones it would process in the clinical setting. This is not a direct development of the current project, but it rather wants to underline how much the presented models can benefit from advances in the field of numerical approximation of the Heart Electrophysiology and of the Forward Problem of Electrocardiography.

- **Reference Geometry**: another important limitation lies in the fact that all the data the ST-RB-DNN model has been trained and tested on have been generated (in both test cases) on the same, reference, geometry. In general, it could be possible to derive a parametrization of the Forward Problem with respect to the heart and the torso geometries; in this case, anyway, such geometric parameters would have to be estimated by the model, significantly increasing the level of complexity of the task to be performed. Thus, we may assume at first to consider fixed geometries for the heart and the torso. Even in this simplified setting, it is broadly known that the position and the rotation of the heart inside the torso can vary a lot, both from patient to patient and, for the same patient, from heartbeat to heartbeat, exerting a non-negligible impact on the measured BSPMs. Several ways of taking these effects into account could be figured out, readily all subject to the fact that the data used for the training are obtained for different positions and rotations of the heart inside the torso. For instance, we could take advantage of already developed methods to track the position of the heart from body surface potentials (as the one presented in [61]). Once the heart position is (approximately) known, we could then provide it as input to a ST-RB-DNN model together with the body surface potentials. The model would be analogous to the ones described in Chapter 5, but for the fact that the information on the heart position are used within the RB-solver layer to properly

assemble the Forward Problem to be solved; in this sense, deriving a parametrization of such problem with respect to the heart position would be extremely useful. A one-step alternative could instead consist in performing the estimation of the heart position within the ST-RB-DNN model, together with the one of the epicardial potential and, if any, of the torso conducibility parameters. The implementation of the RB-solver layer would be the same as in the previous approach, but in this case the model would end up being more complex and, thus, more difficult to train and more likely to fail in convergence. Among the two, then, the first approach seems to be (potentially) the best one, provided that the selected heart position tracking algorithm is accurate enough.

- **Unrealistic Noise**: another key aspect that differentiates clinically measured body surface potentials from numerically approximated ones is the presence of noise. As briefly discussed in Subsubsection 5.2.3.1, we have augmented the available dataset by superimposing some correlated white Gaussian noise, constructed in order to get an average SNR of $\approx 19\ dB$. Anyway, this is a very simplistic approach, real body surface potentials feature the presence of different sources of noise, which are due to various aspects as muscular contraction/relaxation, breathing or measurement instruments and procedures. Because of this, several *ad hoc* filtering techniques have been developed, in order to reduce as much as possible the effect of noise components and to visualize the "clean" signals due to the heart electrical activity only (see [81, 82]). Thus, provided that a fast and accurate solver of the Heart EP and of the Forward Problem of Electrocardiography is developed and that the heart position is somehow taken into account in the model, a subsequent update would be to enrich the numerically simulated signals with realistic noise components.

- **Missing Atria**: another limitation, related to the numerical approximation of the heart EP, is the fact that we have never considered the presence of the atria, just focusing on the two ventricles. Overcoming this issue is absolutely feasible, since it would suffice to include the atria in the numerical simulations, as done for instance by *Schenone et al.* in [88], and it would allow to potentially detect pathological conditions due to atrial defects. Anyway, numerical methods to solve the Inverse Potential Problem perform way better on the ventricles than on the atria since, being thicker and wider, they induce larger potentials on the body surface, so that even small variations could be detected. As a starting point, then, it is preferable to focus on the ventricles only; the inclusion of the atria configures as a more advanced development.

- **Limited Design Research**: an additional limitation, which is anyway common to all works in the field of AI and DL, lies in the fact that only a limited amount of NN architectures has been taken into account in the project. In particular, the presented models are all based on the design choices made in [36] in the context of RB-CNNs for unsteady parametrized PDEs (see Subsection 3.2.2). During the first stages of development, some tests have been done also considering *ResNets* and CRNNs, as those have been employed in [76] and in [75] respectively, in the context of ECG heartbeat classification; anyway, no significant improvements have been observed, so no further efforts in this sense have been made. In general, the field of NNs offers plenty of possibilities in terms of architectural design and of optimization algorithms, so additional efforts in this sense are very likely to result in the development of better models.

Acknowledgments

This thesis was developed in the framework of the *iHeart* project (grant agreement n. 740132, *iHEART - An Integrated Heart Model for the simulation of the cardiac function*, P.I. Prof. A. Quarteroni).

Appendices

Appendix A

The Heart Anatomy and the Cardiovascular System

This Appendix Chapter provides an overview on the anatomy of the heart and on the cardiovascular system. As in Section 4.1, the vast majority of the information is taken from [37].

In particular, Section A.1 features a description of the anatomy of the human heart; Section A.2 focuses on the other elements of the cardiovascular system (i.e. the blood vessels and the blood); Section A.3 illustrates the main aspects of the operation of the cardiovascular system.

A.1 The Heart Anatomy

The **cardiovascular system** is made of three main elements: the **heart** - a muscular organ that pumps blood throughout the body; the **blood vessels** - ducts through which blood circulates; the **blood** - a fluid that, traveling along the blood vessels, brings nourishment to the cells and takes waste products away from them.

The heart is a muscle that generates the force which is needed to pump the blood into the blood vessels, so that it can reach all the cells of the human body, bringing substances as oxygen and nutrients and taking away the waste products as carbon dioxide. It is located in the middle of the *chest cavity*, just above the diaphragm, a muscle that separates the chest cavity from the abdominal cavity. Its dimensions are of the order of the ones of a fist and its weight is around 300-350 g in males and 250-300 g in females.

The anatomy of the heart can be visualized in Figure A.1. The heart is made of four chambers: the two superior chambers, called **atria**, receive the blood that comes back to the heart from the veins, while the two inferior chambers, called **ventricles**, take the blood from the atria and generate the pressure that is needed to push it into the arteries. Other than a superior/inferior division, the heart can be also seen as divided into a left and a right part, typically called the **left heart** (made of the left atrium and the left ventricle) and the **right heart** (made of the right atrium and the right ventricle). The left heart and the right heart are divided by the **septum**, which prevents the blood of the one side to mix up with the one of the other side; in particular, the portion of septum which separates the atria is called *interatrial septum*, while the one that separates the ventricles is called *interventricular septum*.

The four cardiac chambers are involved in phases of contraction and relaxation, that are continuously repeated and that constitute the **cardiac cycles**. During such cycles, it is crucial that the atrial contraction happens before the ventricular one, so that the blood flow proceeds in a unique direction, driven by the pressure gradients; thus it is necessary to prevent the blood from flowing in the opposite direction and this function is accomplished by the **cardiac valves**. More precisely, the four cardiac valves allow the blood to circulate according to a unique specific

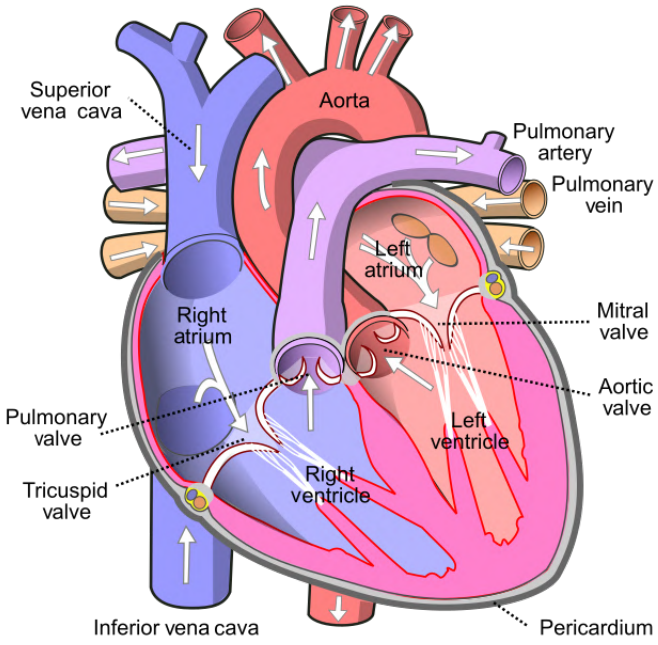


Figure A.1: Schematic view of the anatomy of the human heart.
Image by Wapcaplet - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=830253

direction both within the heart itself and between the heart and the arteries directly connected to it (the *aorta* and the *pulmonary arteries*). The atria and the ventricles of both the left and the right side are separated by the **AV valves**, which allow the blood to flow from the atria to the ventricles only. The left AV valve is also called **bicuspid valve** or **mitral valve**, being it made by two flaps (called *leaflets* or *cusps*); the right AV valve, instead, is called **tricuspid valve**, since it is made of three different flaps. The valves that regulate the flow between the ventricles and the arteries are instead called **SL valves**, due to their "half-moon" shape; in particular the **aortic valve** is located at the opening between the left ventricle and the aorta, while the **pulmonary valve** is located at the opening between the right ventricle and the pulmonary trunk.

The heart wall is made of three different layers: the external layer, called **epicardium**, is made of conjunctive tissue; the intermediate layer, called **myocardium**, is made of cardiac muscular tissue; the internal layer, called **endocardium**, is made of epithelial cells. Additionally, the heart is surrounded by a membranous double-walled sac, called **pericardium**; it contains the *pericardial fluid*, that provides lubrication to the heart during its beats.

A.2 Other Elements of the Cardiovascular System

Other than the heart, the circulatory system is made of two additional elements: the blood vessels and the blood.

The blood vessels form a closed system of ducts that allows to transport the blood from the heart to the different organs and from the different organs back to the heart; this system is commonly referred to as the **vascular system**. The blood which leaves the heart is carried to the different organs and tissues of the human body via big vessels, called **arteries**; these vessels tend to sequentially branch out, becoming more and more numerous and tiny as their distance from the heart increases. The smallest arteries, called *arterioles*, branch out in vessels which are even smaller, called **capillaries**; from the capillaries, the blood comes back to the heart, flowing

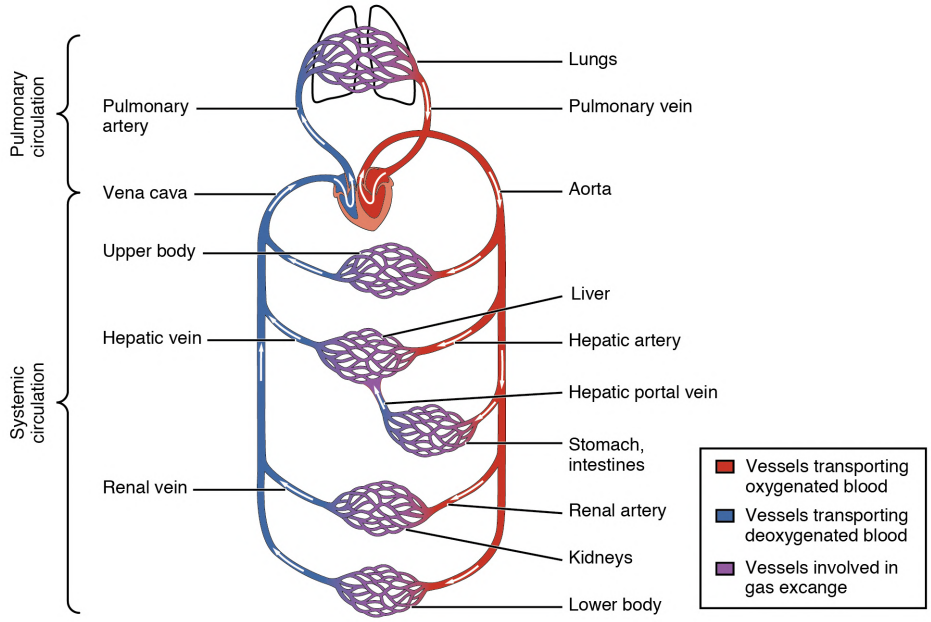


Figure A.2: Schematic view of the cardiovascular system.
Image by OpenStax College - Anatomy & Physiology, Connexions Web site. http://cnx.org/content/col11496/1.6/, Jun 19, 2013., CC BY 3.0, https://commons.wikimedia.org/w/index.php?curid=30148241

in vessels which get bigger and bigger - the **venules** and then the **veins**.

The blood is a fluid, despite almost half of its volume is made of cells. The biggest cells are called **erythrocytes** (or **red blood cells**), which contain *hemoglobin*, a protein that allows to transport the oxygen. The other cells are the **leukocytes** (or **white blood cells**), which allow the organism to defend itself against the aggression of microorganisms. Additionally, the blood contains the **platelets**, fragments of cells which play a key role in blood coagulation. The liquid portion of the blood is called **plasma** and it is made of water in which proteins, electrolytes and other solutes are dissolved.

A.3 Blood Flow through the Heart and the Blood Vessels

A schematic representation of the circulatory system can be found in Figure A.2. The circulatory system consists of two different circuits: the **pulmonary circulation**, made by the pulmonary blood vessels and by the vessels that connect the lungs to the heart, and the **systemic circulation**, that contains all the blood vessels directed from/to the other parts of the body to/from the heart. The systemic circulation has to supply with oxygen and nourishment all the organs, including also the lungs (via the *bronchial arteries*) and the heart itself (via the *coronary arteries*). These two circulations get blood from different parts of the heart; indeed the right heart provides blood to the pulmonary circulation, while the left heart supplies the systemic one. It is worth noticing that the blood of the one side of the heart must not mix up with the one from the other side; because of this, the heart consists of two distinct pumps, yet located within the same organ.

Both the pulmonary and the systemic circulation are equipped with dense nets of capillaries, known as *capillary beds*, where the exchange of nutrients and gases (oxygen and carbon dioxide) physically happens. In particular, at the pulmonary capillaries, the oxygen coming from the air present in the lungs is transferred to the blood, while the carbon dioxide leaves the blood and goes into the lungs; thus, when the blood leaves the pulmonary capillaries, it is relatively rich in oxygen and it is then called *oxygenated blood* (or *red blood*). Conversely, at the capillary beds of the systemic circulation the cells consume oxygen and produce carbon dioxide; thus, when the

blood flows in such vessels, the oxygen leaves it and carbon dioxide enters it. The blood leaving the systemic capillaries is then poor in oxygen, thus being referred to as *deoxygenated blood* (or *blue blood*).

Let's now follow the blood flow through the systemic and pulmonary circulations, supposing to start from the left ventricle.

1. The left ventricle, contracting, pumps the oxygenated blood into the **aorta** (passing through the aortic valve); its branches then carry the blood up to the capillary beds of all the organs and tissues of the human body.
2. The blood gets deoxygenated at the systemic capillary beds and it goes back to the heart via the **caval veins**, two big veins that bring the blood to the right atrium. More specifically, the *superior caval vein* carries the blood from the parts of the body above the diaphragm, while the *inferior caval vein* carries the blood from the parts of the body below the diaphragm.
3. Upon a contraction of the right atrium, the blood passes through the tricuspid valve and it reaches the right ventricle.
4. The right ventricle, contracting, pumps the blood into the pulmonary trunk (passing through the pulmonary valve), which branches out immediately into the **pulmonary arteries**; such arteries carry the deoxygenated blood to the lungs and they are the only arteries which contain poorly oxygenated blood.
5. The blood gets oxygenated at the pulmonary capillary beds and then, through the **pulmonary veins**, reaches the left atrium. Such veins are the only ones containing blood that has a high quantitative of oxygen.
6. Finally, upon a contraction of the left atrium, the blood reaches again the left ventricle, passing through the mitral valve.

It is interesting to notice how the organization of the blood flow through the systemic and pulmonary circulations reflects into the anatomy of the heart; indeed the thickness of the wall (i.e. its muscular mass) of each heart chamber is directly proportional to the distance that the blood contained in such chamber has to cover. For instance, the atria only have to pump blood into the ventricles, through the AV valves, thus atrial walls are quite slim; conversely, the blood pumped by the ventricles has to reach all the organs and tissues of the body, thus ventricles must have a much higher muscular mass than the atria, which results in having much thicker walls. Moreover, the left ventricle has to push the oxygenated blood to all the body, from the brain to the feet, while the right ventricle only has to push the deoxygenated blood to the lungs, which are pretty close to it; this reflects in the left ventricle having a muscular mass which is significantly bigger than the one of the right ventricle. This plays an important role in electrocardiography (see in Subsections 4.1.1 and 4.1.2).

Finally, it is worth observing that the blood flow through the systemic and pulmonary circulations does not happen only *in series*, but also *in parallel*. Indeed it is true that, on the one side, the blood has to go through both the systemic and the pulmonary circulations before getting back to the starting point (i.e. *in series*), but it is also true, on the other side, that the systemic and the pulmonary circulations happen simultaneously (i.e. *in parallel*). The right heart pumps deoxygenated blood to the lungs at the same time as the left heart pumps oxygenated blood to all the organs and tissues; because of this, the left and the right heart contract and relax almost simultaneously, so that we can speak about **systole** (i.e. ventricular contraction) and **diastole** (i.e. ventricular relaxation), without making a distinction between the left and the right halves of the heart. Also, the systemic circulation in itself features an organization *in parallel*, as all the organs are nourished simultaneously by different arteries. This has two main advantages: on the one side every organ receives fully oxygenated blood, that has not been already depleted in oxygen by another one. On the other side, the blood flux towards each organ can be tuned almost independently and adapted to the constant variations of the metabolic demands; at any time it is then possible to increase the blood flow towards the most active organs and to reduce the one towards the least active ones.

Appendix B

Additional Numerical Methods

B.1 POD, SVD and Randomized SVD

Proper Orthogonal Decomposition

Proper Orthogonal Decomposition (POD), also known as *Principal Component Analysis (PCA)* or *Karhunen-Loéve Transform (KLT)* is a statistical procedure, originally introduced by *K. Pearson* in [89], which aims at obtaining low-dimensional approximate descriptions of high-dimensional processes. A general overview can be found in [90]; here we will just summarize the most relevant aspects, which proved to be useful in carrying out the current project. In mathematical terms, if a process is described by the function $\boldsymbol{f} \in V$ such that $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^M$, the POD aims at identifying $n \ll N$ functions $\{\boldsymbol{\phi}_k\}_{k=1}^n,\ \boldsymbol{\phi}_k : \mathbb{R}^N \to \mathbb{R}^M$ such that

$$\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\mu}) \approx \sum_{k=1}^{n} a_k(\boldsymbol{\mu})\boldsymbol{\phi}_k(\boldsymbol{x}) \tag{B.1}$$

with the reasonable expectation that the approximation becomes exact (almost everywhere) as n approaches infinity. Here $\boldsymbol{\mu} \in \mathcal{P}$ denotes a vector of parameters that influence the expression of $\boldsymbol{f}$. Furthermore, the name POD comes from the fact that the functions $\{\boldsymbol{\phi}_k\}_{k=1}^n$ are chosen to be orthonormal, i.e.

$$\langle \boldsymbol{\phi}_{k_1}, \boldsymbol{\phi}_{k_2} \rangle = \delta_{k_1 k_2} \tag{B.2}$$

where δ_{ij} is the Kronecker Delta function and $\langle \cdot, \cdot \rangle$ is an inner product of the considered vector space V. Under (B.2), the expansion coefficients $\{a_k(\boldsymbol{\mu})\}_{k=1}^n$ can be computed as

$$a_k(\boldsymbol{\mu}) = \langle \boldsymbol{f}(\boldsymbol{\mu}), \boldsymbol{\phi}_k \rangle \qquad \forall\ k \in \{1, \dots, n\} \tag{B.3}$$

Singular Value Decomposition

The *Singular Value Decomposition (SVD)* configures as the discrete version of the POD. Indeed consider a system (in a steady state for the sake of simplicity) where we take measurements of N different quantities for M different times (i.e. for M different values of the parameters characterizing the system). Statistically, we can say that we collect M different realizations of the random vector $\boldsymbol{a} \in \mathbb{R}^N$ and we can arrange all the data in a matrix $\boldsymbol{A} \in \mathbb{R}^{N \times M}$. Then, we can compute the SVD of such matrix, which writes as

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{B.4}$$

where $\boldsymbol{U} \in \mathbb{R}^{N \times N}$ and $\boldsymbol{V} \in \mathbb{R}^{M \times M}$ are orthogonal matrices and $\boldsymbol{\Sigma} \in \mathbb{R}^{N \times M}$ is a diagonal matrix, whose diagonal elements $\{\sigma_i\}_{i=1}^{\min(N,M)}$, stored in decreasing order, are called singular values of $\boldsymbol{A}$. $\boldsymbol{U}$ is also the matrix of the eigenvectors of the symmetric matrix $\boldsymbol{A}\boldsymbol{A}^T = \boldsymbol{U}\boldsymbol{\Sigma}^2\boldsymbol{U}^T \in \mathbb{R}^{N \times N}$, while $\boldsymbol{V}$ is the matrix of the eigenvectors of $\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Sigma}^2\boldsymbol{V}^T \in \mathbb{R}^{M \times M}$; $\boldsymbol{\Sigma}^2$ denotes the diagonal

matrix storing the squared singular values of $\boldsymbol{A}$. The rank of $\boldsymbol{A}$ is equal to the number r of non-zero singular values and its SVD can be also rewritten in compact form as

$$\boldsymbol{A} = \tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}}\tilde{\boldsymbol{V}}^T \tag{B.5}$$

being $\tilde{\boldsymbol{\Sigma}}$ the $r \times r$ diagonal matrix storing the non-zero singular values $\{\sigma_i\}_{i=1}^r$; $\boldsymbol{U} \in \mathbb{R}^{N\times r}$ and $\boldsymbol{V} \in \mathbb{R}^{r\times M}$ the truncated matrices of the left and right eigenvectors.

It is evident that (B.4) is the discrete version of (B.1); indeed we can write (B.4) (and also (B.5)) as:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{Q} = \sum_{k=1}^{N} \boldsymbol{u}_k \boldsymbol{q}_k^T \tag{B.6}$$

being $\boldsymbol{u}_k$ the k-th column of $\boldsymbol{U}$ and $\boldsymbol{q}_k$ the k-th row of the matrix $\boldsymbol{Q} = \boldsymbol{\Sigma}\boldsymbol{V}^T$. Then $\boldsymbol{A}$ is equivalent to $\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\mu})$, $\{\boldsymbol{q}_k\}_{k=1}^N$ to $\{a_k(\boldsymbol{\mu})\}_{k=1}^n$ and $\{\boldsymbol{u}_k\}_{k=1}^N$ to $\{\boldsymbol{\phi}_k(\boldsymbol{x})\}_{k=1}^n$.

Notice that, in this case, the decomposition is exact; to get an approximated one only a subset of singular values/vectors $n < r$ should be considered. In fact, it can be proved that the matrix

$$\boldsymbol{A}_n = \boldsymbol{U}\boldsymbol{\Sigma}_n\boldsymbol{V}^T \tag{B.7}$$

being $\boldsymbol{\Sigma}_n$ obtained from $\boldsymbol{\Sigma}$ by setting $\sigma_{n+1} = \sigma_{n+2} = \cdots = \sigma_r = 0$, is the best rank-$n$ approximation of $\boldsymbol{A}$, i.e.

$$\boldsymbol{A}_n = \underset{\boldsymbol{B}\in\mathbb{R}^{N\times M}:\ rank(B)=n}{\operatorname{argmin}} ||\boldsymbol{A} - \boldsymbol{B}||_F \tag{B.8}$$

where $||\cdot||_F$ denotes the *Fröbenius* norm. Additionally, the approximation error is defined through the "extra" singular values as

$$||\boldsymbol{A} - \boldsymbol{A}_n||_F^2 = \sum_{k=n+1}^{N} \sigma_i^2 \tag{B.9}$$

Thus, the first $n < r \leq N$ columns of matrix $\boldsymbol{U}$ define an optimal orthonormal basis to approximate the data stored in $\boldsymbol{A}$; such columns are called *Proper Orthogonal Modes*. This result is known as the *Eckart-Young theorem* and the proof can be found directly in the seminal paper [91].

The optimality properties of the SVD can be deduced also considering that $\boldsymbol{A}\boldsymbol{A}^T$ is proportional to the empirical sample covariance matrix of the random vector $\boldsymbol{a} \in \mathbb{R}^N$, of which M realizations are stored column-wise in matrix $\boldsymbol{A}$. Indeed, it has been proved (see again [89, 90]) that the eigenvectors of the empirical covariance matrix $Cov(\boldsymbol{A})$ (and thus of $\boldsymbol{A}\boldsymbol{A}^T$) coincide with the *Principal Axes*, defined as solutions of the following problem.

Problem 1: *Find* $\boldsymbol{e_1}, \boldsymbol{e_2}, \ldots, \boldsymbol{e_N}$ *such that*

$$Var[\boldsymbol{e_i}^T\boldsymbol{A}] = \sup_{\boldsymbol{e}\in\mathbb{R}^N:||\boldsymbol{e}||=1} Var[\boldsymbol{e}^T\boldsymbol{A}] \tag{B.10}$$

and such that $Cov[\boldsymbol{e_i}^T\boldsymbol{A}, \boldsymbol{e_j}^T\boldsymbol{A}] = 0 \quad \forall\ i, j \in \{1, \ldots, N\},\ i \neq j$

The so-called *Principal Components* of $\boldsymbol{A}$ are then derived just by projecting it onto the *Principal Axes* $\{\boldsymbol{e}_i\}_{i=1}^N$. Since $\boldsymbol{U}$ stores, column-wise, the eigenvectors of $\boldsymbol{A}\boldsymbol{A}^T$, it ends up that the *Proper Orthogonal Modes* and the *Principal Axes* of $\boldsymbol{A}$ are equivalent, thus featuring the same optimality properties.

Randomized Singular Value Decomposition

Coming to the numerical approaches, the SVD of a matrix $\boldsymbol{A} \in \mathbb{R}^{N\times M}$ is typically computed via a two-steps procedure and this is done also in *MATLAB*, which in turn employs a *LAPACK* implementation (see [92]). In the first step, the matrix $\boldsymbol{A}$ is reduced to a bi-diagonal matrix $\bar{\boldsymbol{A}}$ at a cost of $\mathcal{O}(\max(N,M)^2)$ flops. The second step consists then in the computation of the SVD of $\bar{\boldsymbol{A}}$ and it can be done only via iterative methods. In the *LAPACK* routine used by *MATLAB*, the method proposed by *C.F. Van Loan & G.H. Golub* in [25], based on solving a sequence of 2×2 SVD problems, is employed and, if the computation is stopped once machine precision is reached, it costs only $\mathcal{O}(M)$ flops. Thus, the overall computational cost of the *MATLAB* SVD routine is $\mathcal{O}(\max(N,M)^2)$ flops and it is due to the initial "reduction" step. Also, if the rank, say k, of the desired matrix approximation is chosen *a priori*, there exist algorithms based on rank-revealing QR factorization which feature a computational cost of $\mathcal{O}(NMk)$ flops (see [93]).

In many DL applications, SVD has to be applied to very large and dense matrices, sometimes so large that they cannot even fit in memory. During the last decade, optimized techniques to compute low-rank matrix approximations have been developed, leveraging randomness as a powerful tool in this sense. In particular, in the context of this project we have employed the *Randomized SVD* introduced by *Halko et al.* in [26], which allows to efficiently compute an accurate approximation of the best rank-$2k$ approximation of a matrix, being k a target number of singular vectors. As classical algorithms, also this one features a two-steps procedure; considering a matrix $\boldsymbol{A} \in \mathbb{R}^{N\times M}$ it works as follows.

- **Step 1**: *Random sampling is employed to identify a subspace which captures most of the action of the matrix.* In particular, a matrix $\boldsymbol{Q}$ such that $\boldsymbol{A} \approx \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{A}$ is derived. The steps of the algorithm are:
  1. Generate an $M \times 2k$ random Gaussian test matrix $\boldsymbol{\Omega}$
  2. Form $\boldsymbol{Y} = \left(\boldsymbol{A}\boldsymbol{A}^T\right)^q \boldsymbol{A}\boldsymbol{\Omega} \in \mathbb{R}^{N\times 2k}$, with $q = 1$ or $q = 2$. The pre-multiplication by $\left(\boldsymbol{A}\boldsymbol{A}^T\right)^q$ allows to circumvent problems arising from a slow decay of the singular values of $\boldsymbol{A}$, which is common in many applications.
  3. Construct the matrix $\boldsymbol{Q} \in \mathbb{R}^{N\times 2k}$, such that its columns form an orthonormal basis for the range of the random matrix $\boldsymbol{Y}$
- **Step 2:** *The SVD of the projection of the original matrix onto the reduced subspace spanned by the columns of $\boldsymbol{Q}$ is computed.* The steps of the algorithm are:
  1. Form $\boldsymbol{B} = \boldsymbol{Q}^T\boldsymbol{A} \in \mathbb{R}^{2k\times M}$
  2. Compute the SVD of $\boldsymbol{B}$ with classical algorithms, i.e. $\boldsymbol{B} = \tilde{\boldsymbol{U}}\boldsymbol{\Sigma}\boldsymbol{V}^T$
  3. Set $\boldsymbol{U} = \boldsymbol{Q}\tilde{\boldsymbol{U}} \in \mathbb{R}^{N\times 2k}$ and assemble the desired rank-$2k$ approximation of $\boldsymbol{A}$ as $\boldsymbol{A}_{2k} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$.

A theoretical result proved in [26] tells that

$$\mathbb{E}\left(||\boldsymbol{A} - \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T||\right) \leq \left[1 + 4\sqrt{\frac{2\min(N,M)}{k-1}}\right]^{1/(2q+1)} \sigma_{k+1} \tag{B.11}$$

where $\mathbb{E}$ denotes the expectation with respect to the Gaussian random test matrix $\boldsymbol{\Omega}$, $2 \leq k \leq \frac{1}{2}\min(N,M)$ and σ_{k+1} is the $(k+1)$-th singular value of $\boldsymbol{A}$. Additionally, if the approximate SVD is truncated, retaining only the first k singular values and vectors, then the error bound becomes

$$\mathbb{E}\left(||\boldsymbol{A} - \boldsymbol{U}\boldsymbol{\Sigma}_k\boldsymbol{V}^T||\right) \leq \sigma_{k+1} + \left[1 + 4\sqrt{\frac{2\min(N,M)}{k-1}}\right]^{1/(2q+1)} \sigma_{k+1} \tag{B.12}$$

So, we only pay no more than an additive term equal to the $(k+1)$-th singular value of $\boldsymbol{A}$ if we perform the truncation step.

Numerical results on finding the k dominant components of a $N \times M$ dense matrix (which fits in memory) showed that the proposed algorithm features a computational complexity of $\mathcal{O}(MN \log(k))$ flops, which improves the one of $\mathcal{O}(NMk)$ flops of classical deterministic algorithms.

All additional details, proofs and specifics of the presented algorithm can be found in [26].

B.2 Discrete Fourier Transform and Fast Fourier Transform Algorithms

Fourier Transform

In [94], *J. Fourier* showed, in the context of heat transfer, that some functions could be written as an infinite sum of harmonics. This lead to the definition of the *Fourier Transform*, a mathematical transform that maps a function into its complex-valued constituent frequencies. In mathematical terms, the Fourier Transform $\mathcal{F}$ of a real-valued function $f : \mathbb{R}^n \to \mathbb{C}$ such that $f \in L^1(\mathbb{R}^n)$ (so, f must be Lebesgue integrable in $\mathbb{R}^n$) is defined as

$$\mathcal{F}(f)(\boldsymbol{\xi}) =: \hat{f}(\boldsymbol{\xi}) = \int_{\mathbb{R}^n} f(\boldsymbol{x}) e^{-2\pi i \langle \boldsymbol{x}, \boldsymbol{\xi} \rangle} \, d\boldsymbol{x} \tag{B.13}$$

where $\langle \cdot, \cdot \rangle$ represents the standard inner product in $\mathbb{R}^n$. Also, it is possible to define an *Inverse Fourier Transform* $\mathcal{F}^{-1}$ of a function $\hat{f} : \mathbb{R}^n \to \mathbb{C}$ as

$$\mathcal{F}^{-1}(\hat{f})(\boldsymbol{x}) = f(\boldsymbol{x}) = \left(\frac{1}{2\pi}\right)^n \int_{\mathbb{R}^n} \hat{f}(\boldsymbol{\xi}) e^{2\pi i \langle \boldsymbol{\xi}, \boldsymbol{x} \rangle} \, d\boldsymbol{\xi} \tag{B.14}$$

The domain of the function f to be transformed is typically referred to as the time domain, since a classical application of the Fourier Transform is on functions of time (in case $n = 1$); the domain of the transformed function $\hat{f}$ is instead called the frequency domain (being it "inverse" to the time domain) or the Fourier domain.

Given $f, g \in L^1(\mathbb{R}^n)$, the most important properties of the Fourier Transform are:

- **Linearity**: $\mathcal{F}(af(\boldsymbol{x}) + bg(\boldsymbol{x})) = a\hat{f}(\boldsymbol{\xi}) + b\hat{g}(\boldsymbol{\xi}) \quad \forall\, a, b \in \mathbb{C}$
- **Translation**: $\mathcal{F}(f(\boldsymbol{x} - \boldsymbol{x_0})) = e^{-2\pi i \langle \boldsymbol{x_0}, \boldsymbol{\xi} \rangle} \hat{f}(\boldsymbol{\xi}) \quad \forall\, \boldsymbol{x_0} \in \mathbb{R}^n$
- **Modulation**: $\mathcal{F}(e^{2\pi i \langle \boldsymbol{x}, \boldsymbol{\xi_0} \rangle} f(\boldsymbol{x})) = \hat{f}(\boldsymbol{\xi} - \boldsymbol{\xi_0}) \quad \forall\, \boldsymbol{\xi_0} \in \mathbb{R}^n$
- **Scalar Scaling**: $\mathcal{F}(f(a\boldsymbol{x})) = \dfrac{1}{|a|} \hat{f}\left(\dfrac{\boldsymbol{\xi}}{a}\right) \quad \forall\, a \in \mathbb{R} : a \neq 0$
- **Scaling**: $\mathcal{F}(f(\boldsymbol{A}\boldsymbol{x})) = |det(\boldsymbol{A})|^{-1} \hat{f}(\boldsymbol{A}^{-1}\boldsymbol{\xi}) \quad \forall\, \boldsymbol{A} \in \mathbb{R}^{n \times n} : det(\boldsymbol{A}) \neq 0$
- **Conjugation**: $h(\boldsymbol{x}) = \overline{f(\boldsymbol{x})} \implies \hat{h}(\boldsymbol{\xi}) = \overline{\hat{f}(-\boldsymbol{\xi})}$
- **Convolution Theorem**: $\mathcal{F}(f(\boldsymbol{x}) * g(\boldsymbol{x})) = \hat{f}(\boldsymbol{\xi})\hat{g}(\boldsymbol{\xi})$ and $\mathcal{F}(f(\boldsymbol{x})g(\boldsymbol{x})) = \hat{f}(\boldsymbol{\xi}) * \hat{g}(\boldsymbol{\xi})$
- **Parseval's Theorem**: $\int_{\mathbb{R}^n} f(\boldsymbol{x}) g(\boldsymbol{x}) \, d\boldsymbol{x} = \int_{\mathbb{R}^n} \hat{f}(\boldsymbol{\xi}) \overline{\hat{g}(\boldsymbol{\xi})} \, d\boldsymbol{\xi}$
- **Plancherel's Theorem**: $\int_{\mathbb{R}^n} |f(\boldsymbol{x})|^2 \, d\boldsymbol{x} = \int_{\mathbb{R}^n} |\hat{f}(\boldsymbol{\xi})|^2 \, d\boldsymbol{\xi}$
- **Differentiation**: $\mathcal{F}\left(\dfrac{d^n}{d\boldsymbol{x}_k^n} f(\boldsymbol{x})\right) = (2\pi i \boldsymbol{\xi}_k)^n \hat{f}(\boldsymbol{\xi})$ and $\dfrac{d^n \hat{f}(\boldsymbol{\xi})}{d\boldsymbol{\xi}_k^n} = \mathcal{F}\left((-2\pi i \boldsymbol{x}_k)^n f(\boldsymbol{x})\right)$

Notice that Plancherel's Theorem allows to identify the Fourier Transform as an isometry between $L^2(\mathbb{R}^n)$ and itself, with respect to the norm of $L^2(\mathbb{R}^n)$. Because of its many interesting properties, the Fourier Transform has become a very important tool in a variety of fields, from the analysis of PDEs and ODEs to signal/image processing and quantum mechanics. Additional information about the Fourier Transform, its properties and its applications can be found in [95].

Discrete Fourier Transform

The *Discrete Fourier Transform (DFT)* is the discrete counterpart of the Fourier Transform and it acts on finite-dimensional sets made of equally-spaced samples of a complex-valued function. In the one-dimensional case, it turns a sequence of N complex numbers $\boldsymbol{x} = \{x_0, x_1, \ldots, x_{N-1}\}$ into another sequence of complex numbers $\boldsymbol{X} = \{X_0, X_1, \ldots, X_{N-1}\}$ as follows:

$$\begin{aligned} X_k &= \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \\ &= \sum_{n=0}^{N-1} x_n \left[\cos\left(\frac{2\pi}{N} kn\right) - i \sin\left(\frac{2\pi}{N} kn\right)\right] \end{aligned} \tag{B.15}$$

$\forall\ k \in 0, \ldots, N-1$. Actually the equation can be evaluated also outside the domain $k \in \{0, \ldots N-1\}$ and the resulting sequence is N-periodic, i.e. $X_{k+cN} = X_k \quad \forall\ c \in \mathbb{Z}$, because of the N-periodicity of all the complex exponential terms. As for the Continuous Fourier Transform, also the discrete one admits an inverse transform which reads as:

$$\begin{aligned} x_n &= \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{\frac{2\pi i}{N} kn} \\ &= \frac{1}{N} \sum_{k=0}^{N-1} X_k \left[\cos\left(\frac{2\pi}{N} kn\right) + i \sin\left(\frac{2\pi}{N} kn\right) \right] \end{aligned} \tag{B.16}$$

An important observation is that the DFT is actually nothing more than a "smart" linear application from $\mathbb{C}^n$ to $\mathbb{C}^n$. Indeed it can be fully encoded in the so-called *DFT matrix*, which reads as follows:

$$\boldsymbol{F} = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ 1 & e^{-\frac{2\pi i}{N} \cdot 1 \cdot 1} & \ldots & e^{-\frac{2\pi i}{N} \cdot 1 \cdot (N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{-\frac{2\pi i}{N} \cdot (N-1) \cdot 1} & \ldots & e^{-\frac{2\pi i}{N} \cdot (N-1) \cdot (N-1)} \end{bmatrix} \tag{B.17}$$

such that $\boldsymbol{X} = \boldsymbol{F}\boldsymbol{x}$ and $\boldsymbol{x} = \frac{1}{N}\boldsymbol{F}^*\boldsymbol{X}$, being $\boldsymbol{F}^{-1} = \frac{1}{N}\boldsymbol{F}^*$ the inverse DFT matrix ($\boldsymbol{F}^*$ denotes the Hermitian transpose of $\boldsymbol{F}$). Moreover, the columns of $\boldsymbol{F}$ form an orthogonal basis for $\mathbb{C}^N$ and they all have norm equal to $\sqrt{N}$; thus, by dividing $\boldsymbol{F}$ by $\sqrt{N}$, we get a unitary matrix $\boldsymbol{U} = \frac{1}{\sqrt{N}}\boldsymbol{F}$ that defines the so-called *Unitary DFT*. In particular, it holds that:

$$\begin{aligned} \boldsymbol{X}^u &= \boldsymbol{U}\boldsymbol{x} \\ \boldsymbol{x} &= \boldsymbol{U}^{-1}\boldsymbol{X}^u = \boldsymbol{U}^*\boldsymbol{X}^u \end{aligned} \tag{B.18}$$

where $\boldsymbol{U}^*$ denotes the Hermitian transpose of $\boldsymbol{U}$ and $\boldsymbol{X}^u$ the Unitary DFT of $\boldsymbol{x}$.

Clearly, it is possible to extend the DFT to handle multidimensional inputs, thus defining the Multidimensional DFT. So, if $\{x_{n_1,n_2,\ldots,n_d}\}$ is a d-dimensional array storing $\prod_{l=1}^{d} N_l$ equally-spaced samples of a function $f : \mathbb{R}^d \to \mathbb{C}$, then the Multidimensional DFT of it is defined as

$$X_{k_1,k_2,\ldots,k_d} = \sum_{n_1=0}^{N_1-1} \left(e^{-\frac{2\pi i}{N_1} n_1 k_1} \sum_{n_2=0}^{N_2-1} \left(e^{-\frac{2\pi i}{N_2} n_2 k_2} \ldots \sum_{n_d=0}^{N_d-1} e^{-\frac{2\pi i}{N_d} k_d n_d} x_{n_1,n_2,\ldots,n_d} \right) \right) \tag{B.19}$$

where $\{X_{k_1,k_2,\ldots,k_d}\}$ is the d-dimensional array storing the $\prod_{l=1}^{d} N_l$ complex values of the Multidimensional DFT of $\{x_{n_1,n_2,\ldots,n_d}\}$. In compact vectorial form, (B.19) can be rewritten as:

$$\boldsymbol{X_k} = \sum_{\boldsymbol{n=0}}^{\boldsymbol{N-1}} e^{-2\pi i \langle \boldsymbol{k}, \boldsymbol{n/N} \rangle} \boldsymbol{x_n} \tag{B.20}$$

where $\boldsymbol{n} = (n_1, n_2, \ldots, n_d)$, $\boldsymbol{k} = (k_1, k_2, \ldots, k_d)$ and $\boldsymbol{N} = (N_1, N_2, \ldots, N_d)$. Analogously, the inverse Multidimensional DFT writes as:

$$\boldsymbol{x_n} = \frac{1}{\prod_{l=1}^{d} N_l} \sum_{\boldsymbol{k=0}}^{\boldsymbol{N-1}} e^{2\pi i \langle \boldsymbol{n}, \boldsymbol{k/N} \rangle} \boldsymbol{X_k} \tag{B.21}$$

Finally, sticking to the same notation, the Unitary Multidimensional DFT writes as

$$\boldsymbol{X}^u_{\boldsymbol{k}} = \frac{1}{\prod_{l=1}^{d} \sqrt{N_l}} \sum_{\boldsymbol{n=0}}^{\boldsymbol{N-1}} e^{-2\pi i \langle \boldsymbol{k}, \boldsymbol{n/N} \rangle} \boldsymbol{x_n} \tag{B.22}$$

The DFT inherits all the properties listed before for the Continuous Fourier Transform, upon an adaptation to the discrete domain. Among those, the (Discrete) Convolution Theorem makes it an extremely useful tool in the field of image and signal processing (since all basic filtering techniques are based on convolutions and, thus, can be easily carried out in the Fourier domain), while properties linked to Differentiation are widely used in the numerical approximation of PDEs and ODEs. Additionally, if the entries of the input $\boldsymbol{x}$ are all real, then its DFT is symmetric, i.e.

$$X_{k_1,k_2,\ldots,k_d} = X^*_{-k_1\%N_1,-k2\%N_2,\ldots,-k_d\%N_d} \tag{B.23}$$

$\forall\, k_l \in \{0,\ldots N_l-1\}, \quad \forall\, l \in \{1,\ldots,d\}$, where * denotes complex conjugation and $\%$ the modulo operation.

Fast Fourier Transform Algorithms

As seen before, the $1D$ DFT consists in a matrix-vector product between a $N \times N$ matrix and a N-dimensional vector; thus its "natural" computational cost is $\mathcal{O}(N^2)$ flops. Anyway, starting from the 60s, several algorithms able to compute the DFT of a finite dimensional array in just $\mathcal{O}(N\log(N))$ flops have been developed. All such algorithms go under the name of *Fast Fourier Transform (FFT) algorithms* and they all exhibit the same complexity score, despite no proof that a lower one is impossible has ever been performed.

In the following we will limit ourselves in presenting the first and by far the most commonly used FFT algorithm, which is the Cooley-Tukey algorithm, introduced by *J.W. Cooley & J.W. Tukey* in [96] (1965), despite having it already been developed by *C.F. Gauss* around 1805 in [97]. A more detailed and exhaustive presentation of other relevant FFT algorithms (as Prime-Factor FFT, Bruun's FFT, Rader's FFT, Bluestein's FFT, Goertzel's FFT and others) can be found in the review work [84] by *P. Duhamel & M. Vetterli*.

So, the most basic FFT algorithm is the *radix-2 decimation in time (DIT) Cooley-Tukey* algorithm and it has been introduced in [96]. From a general point of view, it is a divide-and-conquer algorithm that, recursively, breaks a DFT of size N into two DFTs (from which the name "radix-2") of size $N/2$. Readily, this is possible only if N is a power of 2 but, since the number of sample points can be typically chosen freely in the vast majority of applications, this is not a major issue. More in detail, the radix-2 DIT algorithm rearranges the DFT of a N-dimensional vector $\boldsymbol{x}$ into two parts, one related to the even-numbered indices and one related to the odd-numbered ones; thus

$$X_k = \sum_{m=0}^{N/2-1} x_{2m} e^{-\frac{2\pi i}{N}(2m)k} + \sum_{m=0}^{N/2-1} x_{2m+1} e^{-\frac{2\pi i}{N}(2m+1)k} \tag{B.24}$$

Then, (B.24) can be rewritten as

$$X_k = \sum_{m=0}^{N/2-1} x_{2m} e^{-\frac{2\pi i}{N/2}mk} + e^{-\frac{2\pi i}{N}k} \sum_{m=0}^{N/2-1} x_{2m+1} e^{-\frac{2\pi i}{N/2}mk} = E_k + e^{-\frac{2\pi i}{N}k} O_k \tag{B.25}$$

being E_k and O_k the DFTs on the even-numbered and odd-numbered input indices, respectively. Thanks to the $\dfrac{N}{2}$-periodicity of the complex exponential term, the following equalities hold

$$\begin{aligned} X_k &= E_k + e^{-\frac{2\pi i}{N}k} O_k \\ X_{k+\frac{N}{2}} &= E_k - e^{-\frac{2\pi i}{N}k} O_k \end{aligned} \tag{B.26}$$

Thus, the DFT of length N can be expressed by means of two DFTs of size $\dfrac{N}{2}$ and, by applying this trick recursively and by re-using results of intermediate computations, the algorithm ultimately reaches a complexity score of $\mathcal{O}(N\log(N))$ flops.

Several variations of the radix-2 DIT Cooley-Tukey algorithm do exist.

- First of all, despite the algorithm has been conceived recursively, the vast majority of its implementations perform a rearranging to avoid explicit recursion, getting non-negligible computational gains.
- Secondly, the general Cooley-Tukey algorithm recursively re-expresses a DFT of composite size $N = N_1 N2$ as:
  1. Perform N_1 DFTs of size N_2
  2. Multiply by the complex exponential terms
  3. Perform N_2 DFTs of size N_1

  This way of proceeding basically amounts at reshaping the $1D$ input vector of size $N = N_1 N_2$ into a $2D$ matrix of size $N_1 \times N_2$ and then at sequentially performing a row-wise and a column-wise DFT to such matrix, with an intermediate element-wise multiplication step. Typically, either N_1 or N_2 is chosen small and it is called the radix; if N_1 is the radix, the algorithm is called DIT, while if N_2 is the radix is called decimation in frequency (DIF). In this way, the Cooley-Tukey algorithm can be generalized to any not-prime input length N. In case the length of the signals is, instead, a prime number, different algorithms have to be employed, as the Rader's FFT (see [98]).
- The fact that the Cooley-Tukey algorithm proceeds recursively by breaking a DFT into smaller ones allows it to be potentially combined with other FFT algorithms.
- Finally, if the input data is real-valued, so that its DFT satisfies the symmetry condition (B.23), additional computational resources can be saved. Some specific and extremely efficient FFT algorithms have been designed at this aim, but simply removing the useless parts of the computation with the Cooley-Tukey algorithm already allows to approximately halve the overall computational burden.

The final step to be performed is the extension of the FFT algorithms to the multidimensional case. Such passage, anyway, is quite straightforward, since the d-dimensional DFT (see (B.19)) is nothing more than the composition of a sequence of d sets of $1D$ DFTs, performed along one dimension at a time. Thus, employing any of the available FFT algorithms and performing the transformation over one dimension at a time, we get an overall computational complexity of $\mathcal{O}(N \log(N))$ flops, being $N = \prod_{l=1}^{d} N_l$ the total number of datapoints of the input. This way of proceeding is known as the *row-column multidimensional FFT algorithm.*

Appendix C

Computational Environment

The majority of the numerical simulations has been carried out on a *Lenovo ThinkPad T490s* mounting *Ubuntu 18.04.4 LTS*, with 16 GB RAM and an *Intel i7-8565U* processor with 4 cores at 1.80 GHz.

The numerical simulations needed to assemble the dataset for the second test case were instead executed on the *IHeart* cluster (*Lenovo SR950* 8x24-Core *Intel Xeon Platinum 8160*, 2.10 GHz and 1.7 TB RAM) at *MOX, Dipartimento di Matematica, Politecnico di Milano*.

The following programming languages, packages, libraries and programs have been used:

- In *Python* (version 3.7.4) [99] programming language, the main used packages are:
  - *NumPy* (version 1.17.2) [100], the fundamental package for scientific computing in *Python*
  - *SciPy* (version 1.4.1) [101], a *Python* library for scientific computing and technical computing
  - *Tensorflow* (version 2.2.0) [102], a system for large-scale machine learning in *Python*
  - *Keras* (version 2.3.1) [103], a *Python* open-source neural-network library
- In *MATLAB* (version R2019b) [104] programming language, we employed the *redbKIT* library [105], which handles reduced-order modeling of parametrized PDEs. It has been developed at Ecole Polytechnique Fédérale de Lausanne (EPFL), it is currently maintained by *Federico Negri* and it is distributed under BSD 2-clause license.
- *ParaView* (version 5.8.0) [106], an open-source, multi-platform data analysis and visualization application, has been employed for results visualization purposes.

Bibliography

[1] 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," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[2] N. Dal Santo, S. Deparis, and L. Pegolotti, "Data driven approximation of parametrized PDEs by Reduced Basis and Neural Networks," *Journal of Computational Physics*, p. 109550, 2020.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[4] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[5] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *ieee Computational intelligenCe magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[6] G. Kutyniok, P. Petersen, M. Raslan, and R. Schneider, "A theoretical analysis of deep neural networks and parametric PDEs," *arXiv preprint arXiv:1904.00377*, 2019.

[7] K. Lee and K. T. Carlberg, "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders," *Journal of Computational Physics*, vol. 404, p. 108973, 2020.

[8] J. S. Hesthaven and S. Ubbiali, "Non-intrusive reduced order modeling of nonlinear problems using neural networks," *Journal of Computational Physics*, vol. 363, pp. 55–78, 2018.

[9] Q. Wang, J. S. Hesthaven, and D. Ray, "Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem," *Journal of computational physics*, vol. 384, pp. 289–307, 2019.

[10] M. Cluitmans, D. H. Brooks, R. MacLeod, O. Dössel, M. S. Guillem, P. M. van Dam, J. Svehlikova, B. He, J. Sapp, L. Wang, *et al.*, "Validation and opportunities of electrocardiographic imaging: from technical achievements to clinical applications," *Frontiers in physiology*, vol. 9, p. 1305, 2018.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT press, 2016.

[12] A. Quarteroni, A. Manzoni, and F. Negri, *Reduced Basis methods for Partial Differential Equations: an introduction*, vol. 92. Springer, 2015.

[13] J. S. Hesthaven, G. Rozza, B. Stamm, *et al.*, *Certified Reduced Basis methods for parametrized Partial Differential Equations*, vol. 590. Springer, 2016.

[14] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[15] T. Tieleman and G. Hinton, "Lecture 6.5-RMSProp: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[16] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[17] K. Fukushima, "Neocognitron: a hierarchical neural network capable of visual pattern recognition," *Neural networks*, vol. 1, no. 2, pp. 119–130, 1988.

[18] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, 2018.

[19] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in neural information processing systems*, pp. 396–404, 1990.

[20] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[22] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[24] P. Rajpurkar, A. Y. Hannun, M. Haghpanahi, C. Bourn, and A. Y. Ng, "Cardiologist-level arrhythmia detection with convolutional neural networks," *arXiv preprint arXiv:1707.01836*, 2017.

[25] C. F. Van Loan and G. H. Golub, *Matrix computations*. Johns Hopkins University Press Baltimore, 1983.

[26] N. Halko, P.-G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM review*, vol. 53, no. 2, pp. 217–288, 2011.

[27] P. Binev, A. Cohen, W. Dahmen, R. DeVore, G. Petrova, and P. Wojtaszczyk, "Convergence rates for greedy algorithms in Reduced Basis methods," *SIAM journal on mathematical analysis*, vol. 43, no. 3, pp. 1457–1472, 2011.

[28] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera, "An 'empirical interpolation'method: application to efficient Reduced Basis discretization of Partial Differential Equations," *Comptes Rendus Mathematique*, vol. 339, no. 9, pp. 667–672, 2004.

[29] S. Chaturantabut and D. C. Sorensen, "Nonlinear model reduction via discrete empirical interpolation," *SIAM Journal on Scientific Computing*, vol. 32, no. 5, pp. 2737–2764, 2010.

[30] Y. Choi and K. Carlberg, "Space-Time Least-Squares Petrov-Galerkin projection for nonlinear model reduction," *SIAM Journal on Scientific Computing*, vol. 41, no. 1, pp. A26–A58, 2019.

[31] K. Hornik, M. Stinchcombe, H. White, *et al.*, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[32] E. Weinan, J. Han, and Q. Li, "A mean-field optimal control formulation of deep learning," *Research in the Mathematical Sciences*, vol. 6, no. 1, p. 10, 2019.

[33] P. Petersen, M. Raslan, and F. Voigtlaender, "Topological properties of the set of functions generated by neural networks of fixed size," *arXiv preprint arXiv:1806.08459*, 2018.

[34] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: a survey," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 5595–5637, 2017.

[35] R. Tenderini, "PDE-aware machine learning for parametrized parabolic PDEs." EPFL Semester Project, unpublished, 2019.

[36] R. Tenderini, "PDE-aware machine learning for parametrized parabolic PDEs." Politectico di Milano, Advanced Programming for Scientific Computing project, unpublished, 2020.

[37] C. Stanfield, *Fisiologia*. Edises, 2012.

[38] D. Durrer, R. T. Van Dam, G. Freud, M. Janse, F. Meijler, and R. Arzbaecher, "Total excitation of the isolated human heart," *Circulation*, vol. 41, no. 6, pp. 899–912, 1970.

[39] C. R. Wyndham, M. K. Meeran, T. Smith, A. Saxena, R. M. Engelman, S. Levitsky, and K. Rosen, "Epicardial activation of the intact human heart without conduction defect," *Circulation*, vol. 59, no. 1, pp. 161–168, 1979.

[40] Rawshani Research AB, "Clinical ECG interpretation." https://ecgwaves.com/lesson/introduction-to-ecg-interpretation-2/. Accessed: 2020-06-16.

[41] P. C. Franzone, L. F. Pavarino, and S. Scacchi, *Mathematical cardiac electrophysiology*, vol. 13. Springer, 2014.

[42] J. Sundnes, G. T. Lines, X. Cai, B. F. Nielsen, K.-A. Mardal, and A. Tveito, *Computing the electrical activity in the heart*, vol. 1. Springer Science & Business Media, 2007.

[43] W. K. Panofsky and M. Phillips, *Classical electricity and magnetism*. Courier Corporation, 2005.

[44] L. Tung, *A bi-domain model for describing ischemic myocardial DC potentials*. PhD thesis, Massachusetts Institute of Technology, 1978.

[45] P. C. Franzone, L. Guerri, M. Pennacchio, and B. Taccardi, "Spread of excitation in 3-D models of the anisotropic cardiac tissue," *Mathematical biosciences*, vol. 147, no. 2, pp. 131–171, 1998.

[46] B. J. Roth, "Action potential propagation in a thick strand of cardiac muscle," *Circulation research*, vol. 68, no. 1, pp. 162–173, 1991.

[47] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology*, vol. 117, no. 4, p. 500, 1952.

[48] R. R. Aliev and A. V. Panfilov, "A simple two-variable model of cardiac excitation," *Chaos, Solitons & Fractals*, vol. 7, no. 3, pp. 293–301, 1996.

[49] P. Colli Franzone and L. F. Pavarino, "A parallel solver for reaction-diffusion systems in computational electrocardiology," *Mathematical models and methods in applied sciences*, vol. 14, no. 06, pp. 883–911, 2004.

[50] M. Boulakia, S. Cazeau, M. A. Fernández, J.-F. Gerbeau, and N. Zemzemi, "Mathematical modeling of electrocardiograms: a numerical study," *Annals of biomedical engineering*, vol. 38, no. 3, pp. 1071–1097, 2010.

[51] S. Pagani, A. Manzoni, and A. Quarteroni, "Numerical approximation of parametrized problems in cardiac electrophysiology by a local Reduced Basis method," *Computer Methods in Applied Mechanics and Engineering*, vol. 340, pp. 530–558, 2018.

[52] M. Boulakia, M. A. Fernández, J.-F. Gerbeau, and N. Zemzemi, "A coupled system of PDEs and ODEs arising in electrocardiograms modeling," *Applied Mathematics Research eXpress*, vol. 2008, 2008.

[53] H. A. Van der Vorst, "Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems," *SIAM Journal on scientific and Statistical Computing*, vol. 13, no. 2, pp. 631–644, 1992.

[54] C. Wyndham, T. Smith, M. K. Meeran, R. Mammana, S. Levitsky, and K. Rosen, "Epicardial activation in patients with left bundle branch block," *Circulation*, vol. 61, no. 4, pp. 696–703, 1980.

[55] S. Ghosh and Y. Rudy, "Application of l1-norm regularization to epicardial potential solution of the inverse electrocardiography problem," *Annals of biomedical engineering*, vol. 37, no. 5, pp. 902–912, 2009.

[56] B. Messinger-Rapport and Y. Rudy, "Noninvasive recovery of epicardial potentials in a realistic heart-torso geometry. Normal sinus rhythm," *Circulation research*, vol. 66, no. 4, pp. 1023–1039, 1990.

[57] H. S. Oster, B. Taccardi, R. L. Lux, P. R. Ershler, and Y. Rudy, "Noninvasive electrocardiographic imaging: reconstruction of epicardial potentials, electrograms, and isochrones and localization of single and multiple electrocardiac events," *Circulation*, vol. 96, no. 3, pp. 1012–1024, 1997.

[58] S. Ghosh, E. K. Rhee, J. N. Avari, P. K. Woodard, and Y. Rudy, "Cardiac memory in WPW patients: noninvasive imaging of activation and repolarization before and after catheter ablation," *Circulation*, vol. 118, no. 9, p. 907, 2008.

[59] Y. Wang, P. S. Cuculich, P. K. Woodard, B. D. Lindsay, and Y. Rudy, "Focal atrial tachycardia after pulmonary vein isolation: noninvasive mapping with electrocardiographic imaging (ECGI)," *Heart Rhythm*, vol. 4, no. 8, pp. 1081–1084, 2007.

[60] A. Intini, R. N. Goldstein, P. Jia, C. Ramanathan, K. Ryu, B. Giannattasio, R. Gilkeson, B. S. Stambler, P. Brugada, W. G. Stevenson, *et al.*, "Electrocardiographic imaging (ECGI), a novel diagnostic modality used for mapping of focal left ventricular tachycardia in a young athlete," *Heart Rhythm*, vol. 2, no. 11, pp. 1250–1252, 2005.

[61] J. Coll-Font and D. H. Brooks, "Tracking the position of the heart from body surface potential maps and electrograms," *Frontiers in physiology*, vol. 9, p. 1727, 2018.

[62] N. N. Tarkhanov, *The Cauchy problem for solutions of elliptic equations*, vol. 7. Vch Pub, 1995.

[63] L. E. Payne, *Improperly posed problems in Partial Differential Equations*. SIAM, 1975.

[64] A. Karoui, L. Bear, P. Migerditichan, and N. Zemzemi, "Evaluation of fifteen algorithms for the resolution of the electrocardiography imaging inverse problem using ex-vivo and in-silico data," *Frontiers in Physiology*, vol. 9, p. 1708, 2018.

[65] D. Wang, R. M. Kirby, and C. R. Johnson, "Resolution strategies for the finite-element-based solution of the ECG inverse problem," *IEEE Transactions on biomedical engineering*, vol. 57, no. 2, pp. 220–237, 2009.

[66] N. Zemzemi, C. Dobrzynski, L. Bear, M. Potse, C. Dallet, Y. Coudière, R. Dubois, and J. Duchateau, "Effect of the torso conductivity heterogeneities on the ECGI inverse problem solution," in *2015 Computing in Cardiology Conference (CinC)*, pp. 233–236, IEEE, 2015.

[67] S. Schuler, A. Wachter, and O. Dössel, "Electrocardiographic imaging using a spatio-temporal basis of body surface potentials—application to atrial ectopic activity," *Frontiers in physiology*, vol. 9, p. 1126, 2018.

[68] F. Greensite and G. Huiskamp, "An improved method for estimating epicardial potentials from the body surface," *IEEE Transactions on Biomedical Engineering*, vol. 45, no. 1, pp. 98–104, 1998.

[69] A. N. Tikhonov and V. Y. Arsenin, "Solutions of ill-posed problems," *New York*, pp. 1–30, 1977.

[70] P. Colli-Franzone, L. Guerri, S. Tentoni, C. Viganotti, S. Baruffi, S. Spaggiari, and B. Taccardi, "A mathematical procedure for solving the inverse potential problem of electrocardiography. Analysis of the time-space accuracy from in vitro experimental data," *Mathematical Biosciences*, vol. 77, no. 1-2, pp. 353–396, 1985.

[71] Y. Rudy and B. Messinger-Rapport, "The inverse problem in electrocardiography: solutions in terms of epicardial potentials," *Critical reviews in biomedical engineering*, vol. 16, no. 3, pp. 215–268, 1988.

[72] J. L. Lions, "Optimal control of systems governed by Partial Differential Equations problèmes aux limites," 1971.

[73] C. Ramanathan, P. Jia, R. Ghanem, D. Calvetti, and Y. Rudy, "Noninvasive Electrocardiographic Imaging (ECGI): application of the Generalized Minimal Residual (GMRes) method," *Annals of biomedical engineering*, vol. 31, no. 8, pp. 981–994, 2003.

[74] L. Wang, W. Wu, Y. Chen, and C. Liu, "An ADMM-net solution to inverse problem of electrocardiology," in *2018 5th International Conference on Systems and Informatics (ICSAI)*, pp. 565–569, IEEE, 2018.

[75] M. Zihlmann, D. Perekrestenko, and M. Tschannen, "Convolutional recurrent neural networks for electrocardiogram classification," in *2017 Computing in Cardiology (CinC)*, pp. 1–4, IEEE, 2017.

[76] M. Kachuee, S. Fazeli, and M. Sarrafzadeh, "ECG heartbeat classification: a deep transferable representation," in *2018 IEEE International Conference on Healthcare Informatics (ICHI)*, pp. 443–444, IEEE, 2018.

[77] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[78] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, *et al.*, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," 2001.

[79] G. B. Goh, N. O. Hodas, and A. Vishnu, "Deep learning for computational chemistry," *Journal of computational chemistry*, vol. 38, no. 16, pp. 1291–1307, 2017.

[80] L. Lu, Y. Shin, Y. Su, and G. E. Karniadakis, "Dying relu and initialization: Theory and numerical examples," *arXiv preprint arXiv:1903.06733*, 2019.

[81] R. Sameni, M. B. Shamsollahi, C. Jutten, and G. D. Clifford, "A nonlinear bayesian filtering framework for ECG denoising," *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 12, pp. 2172–2185, 2007.

[82] O. Sayadi and M. B. Shamsollahi, "ECG denoising and compression using a modified extended Kalman filter structure," *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 9, pp. 2240–2248, 2008.

[83] S. Butterworth *et al.*, "On the theory of filter amplifiers," *Wireless Engineer*, vol. 7, no. 6, pp. 536–541, 1930.

[84] P. Duhamel and M. Vetterli, "Fast Fourier transforms: a tutorial review and a state of the art," *Signal Processing (Elsevier)*, vol. 19, no. ARTICLE, pp. 259–299, 1990.

[85] J. B. W. Webber, "A bi-symmetric log transformation for wide-range data," *Measurement Science and Technology*, vol. 24, no. 2, p. 027001, 2012.

[86] A. Ferrer, R. Sebastián, D. Sanchez-Quintana, J. F. Rodriguez, E. J. Godoy, L. Martinez, and J. Saiz, "Detailed anatomical and electrophysiological models of human atria and torso for the simulation of atrial activation," *PloS one*, vol. 10, no. 11, p. e0141573, 2015.

[87] T. Dozat, "Incorporating Nesterov momentum into Adam," 2016.

[88] E. Schenone, A. Collin, and J.-F. Gerbeau, "Numerical simulation of electrocardiograms for full cardiac cycles in healthy and pathological conditions," *International journal for numerical methods in biomedical engineering*, vol. 32, no. 5, p. e02744, 2016.

[89] K. Pearson, "LIII. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[90] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[91] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936.

[92] S. Blackford, "LAPACK Documentation - Singular Value Decomposition (SVD)." https://www.netlib.org/lapack/lug/node32.html. Accessed: 2020-08-03.

[93] H. Cheng, Z. Gimbutas, P.-G. Martinsson, and V. Rokhlin, "On the compression of low rank matrices," *SIAM Journal on Scientific Computing*, vol. 26, no. 4, pp. 1389–1404, 2005.

[94] J. B. J. baron Fourier, *Théorie analytique de la chaleur*. F. Didot, 1822.

[95] R. N. Bracewell and R. N. Bracewell, *The Fourier transform and its applications*, vol. 31999. McGraw-Hill New York, 1986.

[96] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Mathematics of computation*, vol. 19, no. 90, pp. 297–301, 1965.

[97] C. Gauss, "Theoria interpolationis methodo nova tractata Werke band 3, 265–327," *Göttingen: Königliche Gesellschaft der Wissenschaften*, 1886.

[98] C. M. Rader, "Discrete Fourier transforms when the number of data samples is prime," *Proceedings of the IEEE*, vol. 56, no. 6, pp. 1107–1108, 1968.

[99] G. Van Rossum and F. L. Drake Jr, *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[100] T. E. Oliphant, *A guide to NumPy*, vol. 1. Trelgol Publishing USA, 2006.

[101] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, 2020.

[102] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: a system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

[103] F. Chollet *et al.*, "Keras," 2015.

[104] MATLAB, *version 9.7.0.1190202 (R2019b)*. Natick, Massachusetts: The MathWorks Inc., 2019.

[105] F. Negri at EPFL, "redbKIT." https://github.com/redbKIT/redbKIT, 2017. Accessed: 2020-08-04.

[106] U. Ayachit, *The Paraview guide: a parallel visualization application*. Kitware, Inc., 2015.