FACULTY
OF SCIENCES

Supervised machine learning analysis of the different types of regular atrial tachycardia based on in-silico cardiac mapping data

Author:	Bjorge Meulemeester
Promoter:	Prof. Dr. N. Vandersickel
Copromoter:	Prof. Dr. Y. Saeys
Supervisors:	L. Lowie
	E. Van Nieuwenhuyse

In Partial Fulfilment of the Requirements for the Degree of Master of Science in Physics and Astronomy

August 2021

GHENT
UNIVERSITY

Thanks to

As my thesis is entering its final stages, the time has come to let my attention resurface to the land of the living and thank the people who have helped and guided me, directly or indirectly, throughout the process of writing, researching, coding, failure and success during this past year.

I'd like to start by thanking my promoter Nele, and my supervisor Lars. Your help went beyond what I expected from you. You were always reachable, always listening and pointing me in the right direction whenever I wandered too far off. You have helped me immensely. This work wouldn't even remotely look like what it looks like now without your help. You both provided me with a caring and supportive environment during what may be one of the most isolated years I have ever experienced. This is why I thank you, Nele, for supporting me and the other thesis students on an personal and emotional level, as well as professionally. Thank you Lars for your ever readiness to mentor me and explaining everything with patience and depth.

I'd love to thank my parents as well. Of course, I wouldn't be here without your financial support, but I'd like to highlight every other support I have received from you. I am undoubtedly making you both proud and I am thankful for never having to doubt that. Thank you for providing a home I could always fall back on. Thank you for the necessary occasional holiday, for cooking, for blasting 80's music in the kitchen with a bottle of wine, for coffee breaks, for sending me fireworks stickers in WhatsApp to celebrate the completion of this thesis, for calling every now and then just to check up on me. I really should pick up the phone more, but please don't quote me on this.

Thank you Lukas, for regularly yanking me away from my computer for banter and mischief, consciously or otherwise. Thank you Vincent for being the warm, considerate and remarkable person you are. Thank you Emma, for being by my side and asking me the questions I need to hear, among lots of others, and for your seemingly undying support. Thank you Pilar, for unwavering friendship and support, for our thesis meetups, for the memes. Thank you Emiel for providing me with excellent coffee. Thank you Peter and Waso for creating a warm crew where I was wholeheartedly welcomed and heard any time of the day. Thank you Bert, Ewoud, Casper, Bert, Femke and Ruben for the regular meetups where possible. All of you contributed to my wellbeing, inspiration and ambition. Your contribution to this thesis may be indirect, but not insignificant.

Contents

Abstract

English

This thesis aims to expand the existing machine learning analysis of cardiac data to atrial electrophysiological mapping data. It discusses a way of reconstructing simulatable meshes starting from in-vivo mapping data, including control over the thickness of myocardium, mesh coarseness and mesh conductivity regions. The meshes are consequently used for 2286 simulations of focal beats and 2097 simulations of reentries around the right pulmonary veins, left pulmonary veins and mitral valve. This expands the otherwise sparse mapping data availability. Additionally, it discusses the processing of mapping data to features. These are used by nine different machine learning models for two supervised machine learning classifications: distinguishing focals from anatomical reentry and inferring the location of anatomical reentry.

Nederlands

Deze thesis richt zich op het uitbreiden van machine learning op hartdata tot atriale elektrofysiologische mapping data. Het bespreekt een manier om simuleerbare meshes te reconstueren startende van in-vivo mapping data, met inclusie van controle over de dikte van het myocardium, mesh resolutie en geleidbaarheidsgebieden. De meshes worden vervolgens gebruikt voor 2286 simulaties van focale beats en 2097 simulaties van reentry rond de rechter longaderen, linker longaderen en de mitraalklep. Dit expandeert de anderzijds zeldzame beschikbaarheid van mapping data. Het bespreekt bovendien het verwerken van deze mapping data tot features. Deze worden gebruikt door negen verschillende machine learning modellen voor twee supervised machine learning classificatieproblemen: het onderscheiden van focale slagen en reentry, en de locatie van reentry classificeren.

1 Background: the heart

This section will give a general overview of the anatomy, functioning and electrical properties of the human heart and its implementation in electrophysiological simulations.

1.1 Anatomy and function of the heart

The heart is a hollow muscle, pumping and pressurising the circulatory system: a connected set of veins throughout the body, providing organs and tissue of oxygen, nutrients, hormones and blood cells. Two main structures can be identified in the heart: the atria (upper chambers) and the ventricles (bottom chambers). These are separated by valves that control the direction of blood flow. The valves between the atria and ventricles are called atrioventricular valves. There are two atrioventricular valves in the heart: one connecting the left atrium to the left ventricle and one connecting the right atrium to the right ventricle. The left and right ventricles are separated by the interventricular septum and the left and right atria are separated by the interatrial septum. The atria receive blood and pump it the blood to the ventricles via the atrioventricular valves. The left atrioventricular valve is called the mitral valve, while the right one is referred to as the tricuspid valve. These valves are a one-way gateway, making sure blood can pass from the atrium to the ventricle, but not vice versa. On top of two atrioventricular valves controlling the blood flow from the atria to the ventricles, two more valves can be identified, controlling blood flow from the ventricles to the arteries. These are the semilunar valves: the pulmonary valve controls the blood flow from the right ventricle to the pulmonary artery leading to the lungs, and the aortic valve does the same for the left ventricle and the aorta. The ventricles are larger structures than the atria, capable of exerting enough force to pump the blood to the rest of the body. The right atrium receives oxygen deprived blood via the vena cava and passes it to the right ventricle. The right ventricle pumps this oxygen deprived blood to the lungs where it is to be loaded with oxygen again. The left atrium

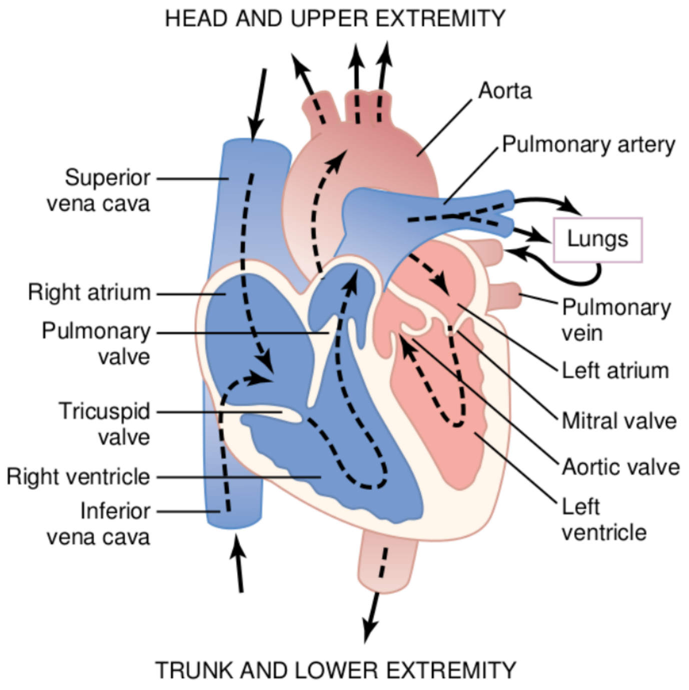


Figure 1.1: Blood flow through different chambers and valves in the heart. Figure taken from Guyton and Hall [1]

receives this oxygen rich blood via the pulmonary veins (literally “lung” veins) and passes it to the left ventricle, to be distributed to the rest of the body via the aorta. It is the contraction of the strong left ventricle that is responsible for our blood pressure. These structures and the direction of the blood flow are shown in Figure 1.1.

1.2 Electrical properties of the heart

The heart pumps blood by means of muscle contraction. This contraction is triggered by an electrical wave, starting in the sinoatrial node (SA node) in the right atrium. This electrical signal propagates through the muscle cells of the heart (myocardium) and triggers the contraction of those cells. This triggering of the myocardium is defined by a process of depolarisation; the electrical charge of the heart muscle cell (cardiomyocyte) rapidly increases in value.

While the ventricles and atria are connected by tissue at the mitral valve, they are not connected in terms of conductivity. The atria are electrically decou-

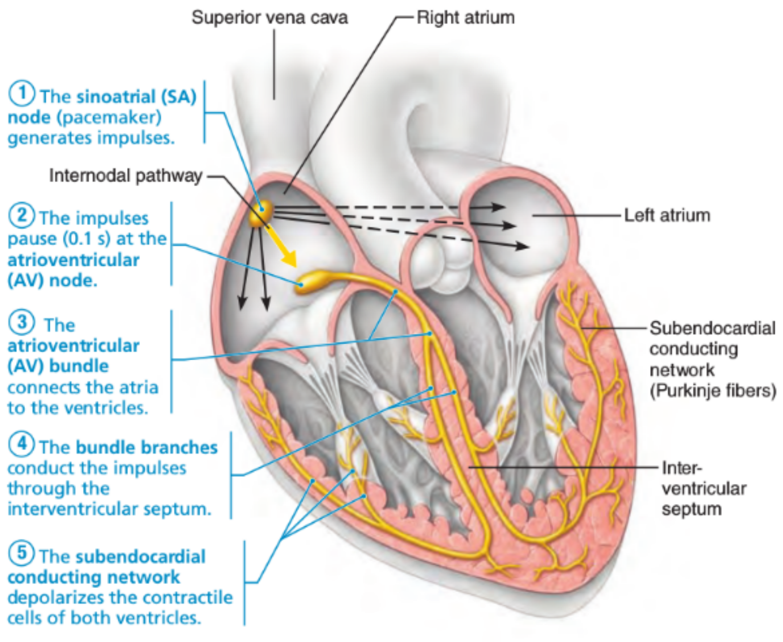

Superior vena cava
Right atrium
1 The sinoatrial (SA) node (pacemaker) generates impulses.
Internodal pathway
2 The impulses pause (0.1 s) at the atrioventricular (AV) node.
3 The atrioventricular (AV) bundle connects the atria to the ventricles.
4 The bundle branches conduct the impulses through the interventricular septum.
5 The subendocardial conducting network depolarizes the contractile cells of both ventricles.
Left atrium
Subendocardial conducting network (Purkinje fibers)
Inter-ventricular septum

Figure 1.2: The sequence of conduction throughout the heart. Figure taken from Marieb and Hoehn [2].

pled from the ventricles. The only gateway for electrical conduction is through a node between the atria and ventricles. This node is called the atrioventricular node (AV node). An electrical signal propagates more slowly through this node than through the atrial or ventricular cells, inducing a delay. This makes sure that the ventricles contract only after the atria contract. This is crucial, as this gives the atria time to push the blood to the ventricles while they are not yet contracting, and makes the ventricles contract only after they have received this blood. The sequence of the conduction of the electrical signal throughout the heart is show in Figure 1.2.

Generally speaking, two types of cells are present in the heart: heart muscle cells (cardiomyocytes) and pacemaker cells. The pacemaker cells can depolarise spontaneously, while the cardiomyocytes can only be depolarised by an external stimulus. The depolarisation of the heart cells is mediated by ionic currents through their cell membranes. These membranes separate the cell from the outside. This outside region is referred to as the bath region, as it is filled with ions that mediate the cellular depolarisation-repolarisation process. The heart cells are connected with this outside region through ionic channels. The ionic channels are proteins embedded in the membrane that can change their shape (open or closed), allowing only specific ions to enter from or exit to another heart cell; they are selectively permeable. Different ions are present in different concentrations, have different time scales and physiological effects. It's these different ion currents that mediate the depolarisation and replarisation of the cardiomyocytes and the pacemaker cells. The dynamics of the charged ions through the cell membrane induce a voltage potential across the cell membrane. This variation in transmembrane voltage gives rise to the so-called action potential (AP), as is shown in Figures 1.3 and 1.5. Let's take a closer look at the effects of the different transmembrane ion flows on both the cardiomyocytes and the pacemaker cells while taking a closer look at the shape of their action potential.

1.2.1 Electrical properties of the cardiomyocyte

This section will describe the electrochemical properties of the cardiomyocyte in terms of ion currents. Only the ion currents included by the Courtemanche ionic model [3] will be discussed. Ionic models will be discussed in Section 1.6.

The **sodium current** I_{Na} is a fast depolarising current and responsible for the fast upstroke of the myocardial action potential. External stimuli cause the Na^+ channels of cardiomyocytes to open very quickly, letting in Na^+ ions (phase 1). This phase ends with the inactivation of the Na^+ channels. The **transient outward potassium current** I_{to} is the main contributing current to the repolarisation after the sharp upstroke of the action potential.

The second phase of the polarising cycle is called the plateau phase and is defined by the intake of Ca^{2+} ions. These are the ions that initiate cardiac contraction. This intake of Ca^{2+} ions is made possible by the **L-type calcium current** I_{CaL}; the opening of the CaL channels allows for Ca^{2+} to flow into the cardiomyocyte. This initiates Ca-Induced-Ca-Release: a positive feedback system that releases enough Ca from the sarcoplasmatic reticulum, located in the bath region, into the cell. This releases enough calcium to make the muscle cell contract.

This phase happens in cooperation with the **ultrarapid delayed potassium rectifier current** I_{Kur} and the **slowly activating rectifier potassium current** I_{Ks}. Both of these currents are outward currents and counter the potential augmentation caused by the intake of Ca^{2+} ions, i.e. rectifying. The rapid current does most of the rectifying. Due to the slow timescale of I_{Ks}, it is only when I_{Kur} is blocked that this current will play a major role.

The third phase is the repolarisation phase. During this phase, the car-

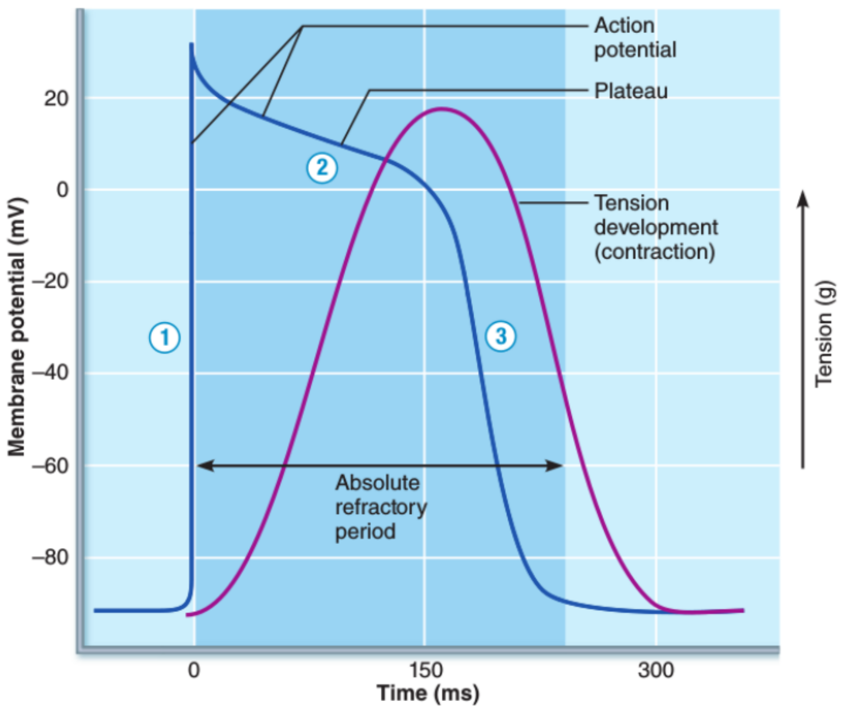


Figure 1.3: A typcal action potential of a cardiomyocyte (blue line) and the induced contraction (purple line). The myocardial action potential is characterised by the sharp upstroke (phase 1), a plateau phase (phase 2) and consequent repolarisation phase (phase 3). Figure taken from Marieb and Hoehn [2]

diomyocyte goes back to its resting potential. This phase is defined by the **potassium K1 current** I_{K1}. This is an inwardly pointed current with strong rectification properties. The K^+ ion channels are opened.

Other currents include I_{NaCa} and I_{NaK} for respectively the Na^+Ca^{2+} exchanger current and the Na^+K pump current which exchange Na^+ for respectively Ca^{2+} or K^+, or vice versa. The plateau current $I_{p,Ca}$ and the background currents $I_{b,Ca}$ and $I_{b,Na}$ are also included. An overview of these currents and their direction is given in Figure 1.4.

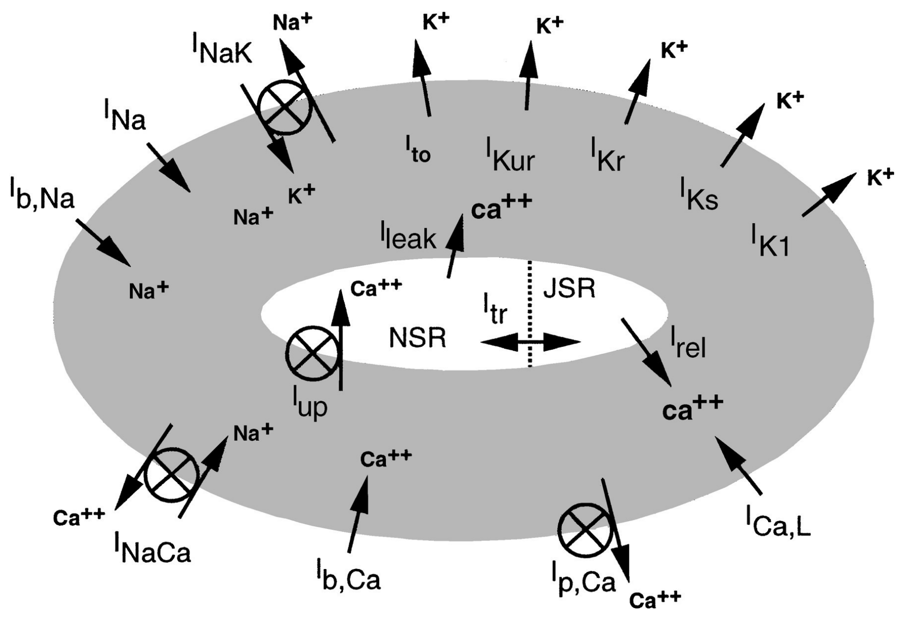


Figure 1.4: Schematic of a cardiomyocyte and its ionic currents. Figure taken from Courtemanche [3].

1.2.2 Electrical properties of the pacemaker cell

In contrast to cardiomyocytes, the pacemaker cells depolarise spontaneously. Their action potential has a different shape, as can be seen in Figure 1.5. These are the cells that are present in the sinoatrial node. These cell have no stable resting potential, but are in an eternal cycle of depolarising and repolarising. The core polarisation dynamics will be briefly discussed by considering three important ionic currents [4, 5] and the shape of the pacemaker action potential, shown in Figure 1.5.

The pacemaker action potential cycle is defined by three phases. In phase 1, the Na^+ channels open up and the K^+ close, allowing for the intake of Na^+ ions, giving rise to a slow depolarisation, in stark contrast to the sharp upstroke of the cardiomyocytes. Once the potential hits the threshold of about $-40\ mV$, Ca^{2+} flow increases substantially and phase 2 begins. This phase is defined by a sharp upstroke of the action potential. In phase 3, this upstroke is stopped by the opening of K^+ channels allowing the flow of rectifying K^+ currents, forcing K^+ ions out of the cell. Once repolarised, the K^+ channels close again, the Na^+ channels open up again, and the process is repeated. It is this cyclical process that determines the pacing of a normal heart rhythm.

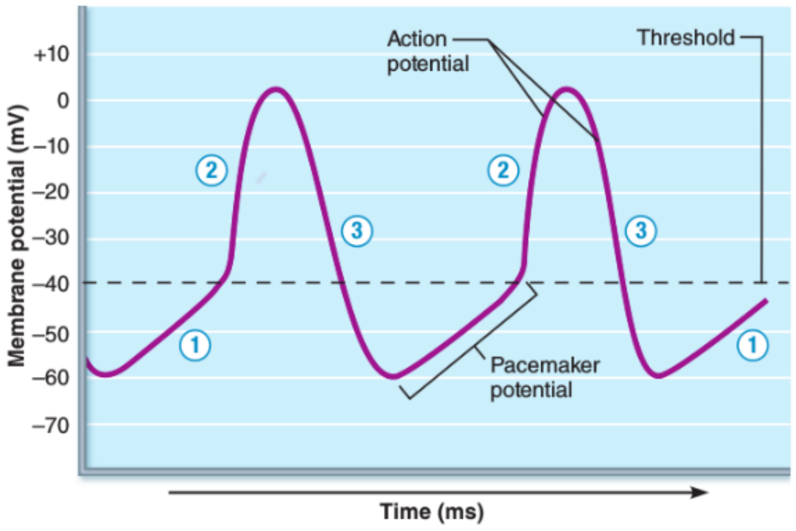


Figure 1.5: A typical action potential of a pacemaker cell, present in the SA node. The shape looks a bit like a sine wave, hence the name sinoatrial node. Figure taken from Marieb and Hoehn [2].

The cardiomyocytes and pacemaker cells are connected with each other through gap junctions: an intercellular tissue making electrochemical communication possible. This allows for larger scale tissue-level electrochemical dynamics to arise: the formation of a wave pattern called the depolarising wavefront.

1.3 Pattern formation

The depolarising wavefront is a trigger wave, meaning it brings a system from one state to another. In the case of cardiomyocytes, the trigger is identified as the moment of Na^+ ion intake, giving rise to the sharp upstroke of the action potential. This brings cardiac tissue from its resting state into an excited state. In this excited state, the cardiac tissue exchanges ions through the ionic channels in the cell membrane (as discussed in Section 1.2). After this excited state, the tissue enters a refractory period. During this period, the tissue cannot get depolarised again. This lasts until the cardiac cells are in their resting state again. This refractory period is associated with the plateau phase of the action potential. This plateau phase can be seen in Figure 1.3.

As cells cannot depolarise during their refractory period, the dynamics of the depolarising wavefront has some interesting properties. Colliding waves will annihilate each other, as each wavefront leaves a wake of tissue in the refractory state, unable to get excited again. This is shown schematically in 1D in Figure 1.6.

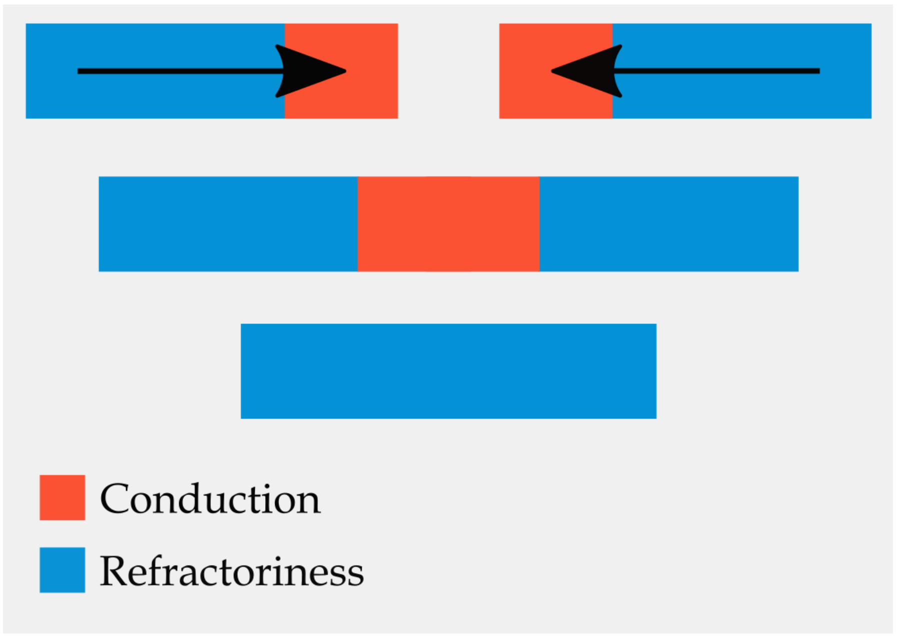


Figure 1.6: Two waves in 1D are about to collide (top). In the end (bottom), only tissue in the refractory period remains and no triggering wavefront is left.

When multiple sources repeatedly initiate a triggering wavefront, the fastest source will dominate the pacing and take over the slowest one. This is because the

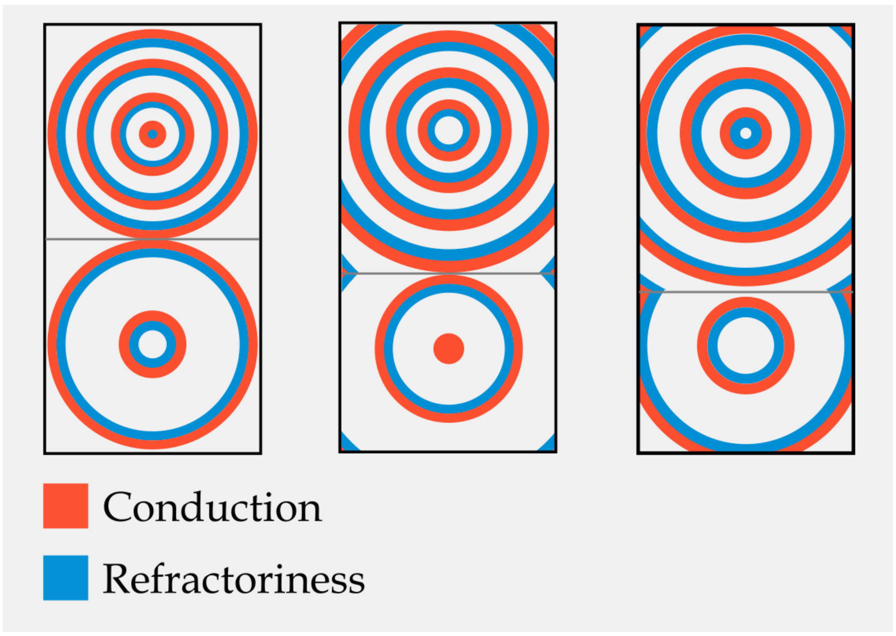


Figure 1.7: Competition between two sources of trigger waves. Figures represent different timeframes, increasing in time from left to right. The waves are annihilated upon collision. The collision region (horizontal grey line) shifts closer and closer to the centre of the slowest focal source as time passes until the slowest source is completely taken over.

collision region will shift more and more towards the location of the slowest source. This is shown schematically in 2D in Figure 1.7. These effects have important implications on the correct functioning of the heart.

1.3.1 Regular tachycardia: focal beats and reentry

Focal beats (or focals in short) refer to myocardial cells that depolarise spontaneously instead of needing an external stimulus. Depending on the rhythm of the focal beat and the SA node, it is possible for a focal beat to depolarise myocardium after the refractory period that was induced by the SA node. This triggers a depolarising wavefront, making the myocardium contract at a time that was not dictated by the SA node. This can make the heart beat out of sync. An example of this is shown on an ECG in Figure 1.10. If the focal beat has a minimum time interval that is quicker than the sinoatrial node, the focal beat may take over the SA node and become the pace dictator of the heart.

Depolarising waves can also create spirals. If there is some non-conductive

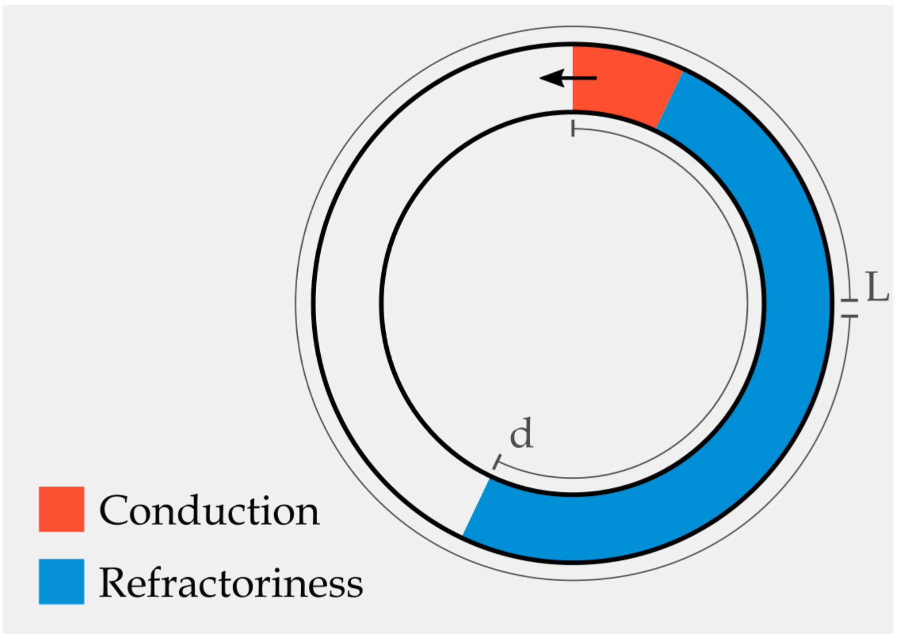


Figure 1.8: An example of anatomical reentry propagating counter-clockwise on a circle with circumference L. As long as the activation wavefront (red) meets excitable tissue, the wave will propagate. If $v \cdot t_R = d \geqslant L$, the wavefront will collide with refractoring non-excitable tissue and extinguish.

tissue with a circumference of L and has a refractory period of t_R, then a wave with velocity $v < L/t_R$ can propagate around this region without being annihalated. This is called **anatomical reentry**. An generalised example of this can be seen in Figure 1.8. Depending on the size of the non-conductive region at the center of the spiral wave, one can distinguish macro-reentry and localized reentry. This phenomenon can take place around scar tissue, veins or the mitral valve.

Reentrant mechanisms do not necessarily need a non-conductive region to propagate around; they can also rotate around a single point. This is called **functional reentry**. This point can be fixed in space, or move around the cardiac tissue. In the latter case, the movement of the point centre of the spiral is called meandering.

As the rotation period of reentries tends to be shorter than the rate of spontaneously depolarising cells, these pose a bigger threat of becoming the dominant source of activation and desynchronising the heart. Reentries are known to induce atrial fibrillation [6], i.e. an atrium that does not fully contract, but is instead defined by a chaotic depolarisation pattern.

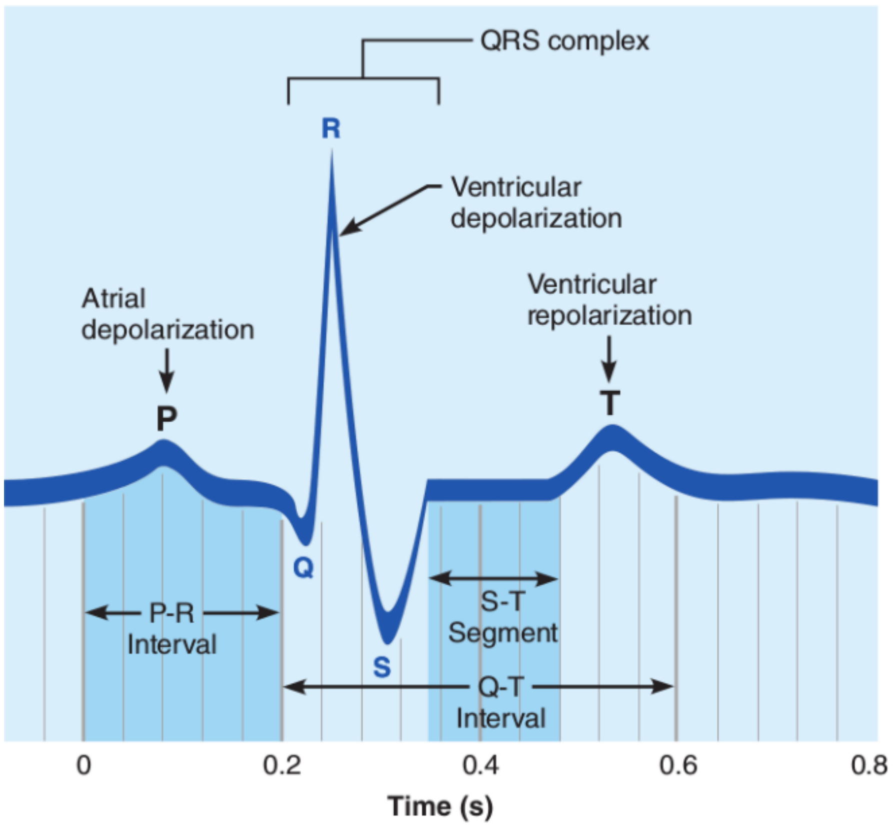


Figure 1.9: A typical shape of an ECG, with annotated sections PQRST. Figure taken from Marieb and Hoehn [2].

1.4 Detection of patterns in the clinical setting

1.4.1 ECGs

The depolarising wavefront as described in Section 1.2 can be measured by means of an electrocardiogram (ECG). This involves placing electrodes around the heart. These electrodes are capable of measuring the change of the electrical field in a single direction and can detect the changes in the electrical field generated by the depolarising wavefront of the heart [7].

Three main sections of an ECG can be identified, which are shown in Figure 1.9 [2]. The first section is called the P wave and it results from the atrial depolarisation. Secondly, the QRS complex is associated with the ventricular depolarisation and happens at the same time as atrial repolarisation. The third section is the T wave and results from ventricular repolarisation.

In a standard ECG, there are 10 electrodes (4 limb and 6 chest electrodes) that can be used to measure the change in electrical field across 12 angles (leads). These 12 leads are used to plot a composite ECG of the heart. This is a great way to extract data about the electrical properties of the heart; ECGs contain information about the rate, amplitude and general location of change of the electrical field. They are also relatively easily obtained and do not require an invasive procedure [7].

As an example, Figure 1.10 shows an excessive heartbeat as measured by such an ECG. Apart from the wrong timing of the excessive heartbeat, the shape of the ECG is also substantially different, indicating the corresponding change in electrical field i.e. the underlying mechanism also behaves differently. Also note the flatline where the normal sinus rhythm would have given a beat; the SA node cannot induce a depolarising wavefront as the heart is still in its refractory period after the excessive heartbeat.

Unfortunately, they are often insufficient for representing complex arrhythmias. They can only measure the rate of change of the electrical field in one direction at once. Due to the projection on this direction, a lot of interesting information is lost. They can only measure the net change in electrical field. ECGs of complex arrhythmias can have complicated shapes, and it is not always readily obvious how the resulting ECG relates to the underlying mechanisms. The analysis of an ECG is challenging as a perfect solution for accurate prediction is not available [8].

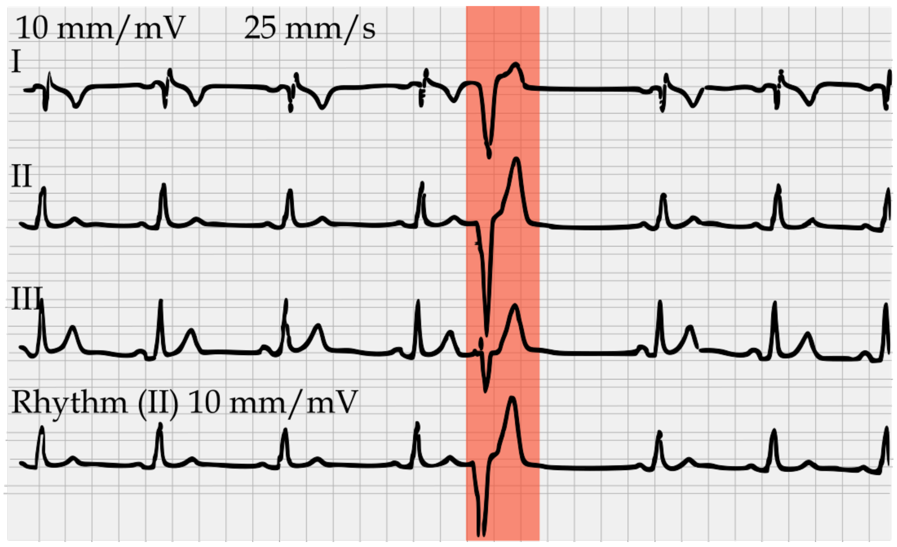


Figure 1.10: An electrocardiogram showing an excessive heartbeat (red). ECG taken from my heart. Each square represents 5 *mm*. The double lines are 1 *mm* apart.

1.4.2 Mapping data

For more complex arrhythmias, a second and invasive procedure is possible: mapping data. In order to acquire this data, one needs to measure the geometry of a heart chamber, in addition to the actual moment of depolarising of the heartcells. This is generally done by introducing a catheter via the groin, which follows the vein up to the inside of the right atrium. From there, the catheter can either map the right atrium, or access the right ventricle via the tricuspid valve. Accessing the left atrium can be done by drilling through the interatrial septum, and the left ventricle can be accessed via the mitral valve. The catheter is placed against the inside of the atrial or ventricular wall (the endocardium). In addition to registering the coordinates of the points against the wall, it also registers an electrogram of the passing depolarising wavefront [6], called intracardiac electrograms [9]. From this electrogram, the time at which a depolarising wavefront passes by can be extracted. This time is called local activation time (LAT). This way, the entire endocardium of a heart chamber can be mapped. The outside of a heart (epicardium) can be mapped as well. An example of the result of such mapping procedure is shown in Figure 1.11 [10], where the intracardiac electrograms are shown on the left hand side. The right hand side showcases the 12-lead electrocardiogram taken at the same time. The center shows a 3D atrium model with a colorscale referring to the LATs.

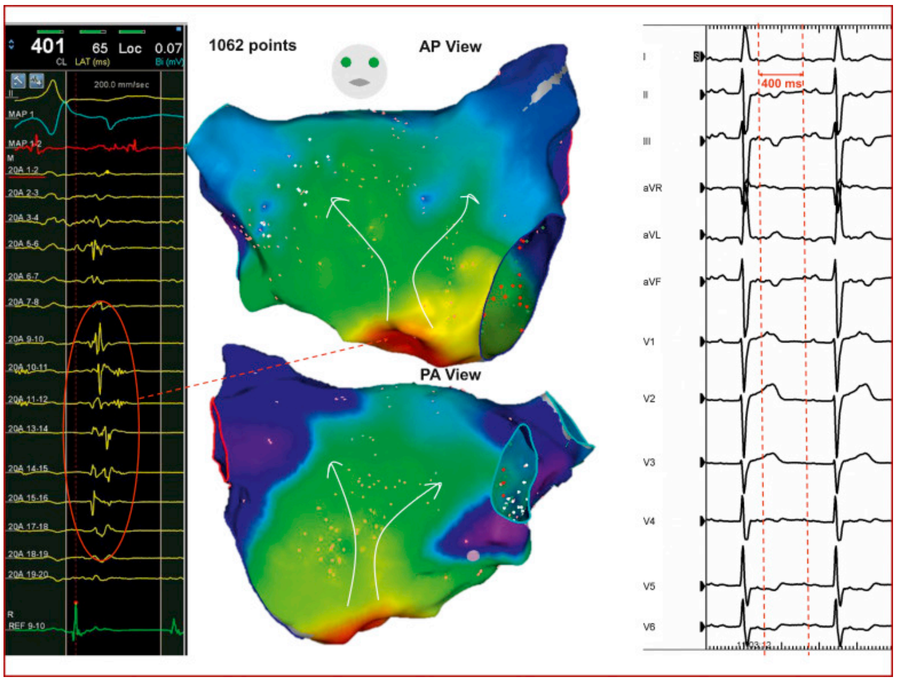


Figure 1.11: An example of mapping data. The intracardiac electrograms are shown on the left hand side. The right hand side showcases the 12-lead electrocardiogram. The center shows a 3D atrium model with a colorscale corresponding to the LATs. The white arrows denote the path of the depolarising wavefront. Figure taken from Bun et al. [10].

1.5 Mathematical representation of the cardiac cell

By identifying the ionic currents, the behaviour of the cardiomyocyte and pacemaker cell can now be described in terms of electrical currents. A mathematical description of the electrical activation of a nerve action potential was first proposed in 1952 with the development of the Hodgkin-Huxley model [11], describing ion currents through nerve cell membranes in mathematical terms, i.e. coupled ordinary differential equations. This model was consequently adapted to cardiac cells by Noble in 1962 [12] and further improved by the inclusion of more currents and a better description of the already included currents. A mathematical model that describes the ion flow through the cell membrane is called an ionic model. Let us discuss the basics of such an ionic model. The following information is also present in the OpenCARP [13] user manual. The base unit for for ionic models is the cellular membrane. Schematically, the electrical properties of a heart cell can be represented by a capacitor wired in parallel to a number of ionic transport mechanisms. In general, an ionic model is of the form

$$I_m = C_m \frac{dV_m}{dt} + \Sigma_\chi I_\chi \tag{1.1}$$

where V_m is the voltage across the cell membrane, I_m is the net current across the membrane, C_m is the capacitance of the lipid bilayer cell membrane and I_χ is a particular transport mechanism, either an ion channel, pump or exchanger. Additional equations can be added to account for additional effects (e.g. the Ca-induced Ca release from the sarcoplasmatic reticulum). Ion channels can be represented by a resistor in series with a battery. The battery represents the Nernst potential, which results from the electrical field due to the difference in concentration of some ion inside and outside the cell. This is the potential at which there is no net flow across a biological membrane for some ion species. Explicitly, this potential is:

$$V_N = \frac{RT}{zF} \ln \frac{[S]_i}{[S]_o} \tag{1.2}$$

, where R is the gas constant, T is the temperature, and z is the valence of the ion species S.

An equivalent electrical scheme for a single heart cell is given in Figure 1.12, where the ion currents are to be wired in parallel. C_m denotes the membrane capacitance, $\Sigma_\chi I_\chi$ the sum of all ionic currents (wired in parallel). $R_n(t, V)$ and V_n denote the non-linear voltage-gated resistance and Nernst potential respectively. These include the time dependency of the ion channel dynamics. R_L and V_L

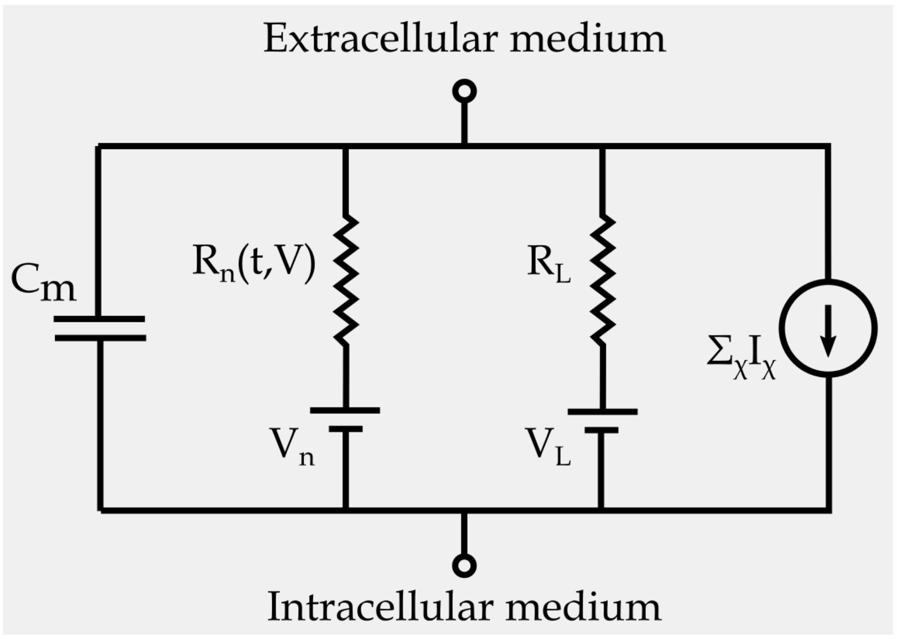


Figure 1.12: A generalised equivalent electrical scheme of a single heart cell membrane, where C_m denotes the membrane capacitance, $\Sigma_\chi I_\chi$ the sum of all ionic currents (wired in parallel), $R_n(t,V)$ and V_n the non-linear voltage-gated resistance and Nernst potential respectively. R_L and V_L denote the linear leakage resistance and leakage potential respectively.

denote the linear leakage resistance and potential: a constant leakage due to the difference in ion concentrations inside and outside of the cell. The electrochemical gradients driving the ion flow are represented by batteries V and the ion pumps and exchangers are represented by $\Sigma_\chi I_\chi$. The resistances are split into two categories. The non-linear resistances $R_n(t,V)$ represent all the resistances that have a voltage and/or time dependency. These include the time and voltage dependency of the voltage-gate ion channels. The linear resistance R_L represent the leakage current due to the difference of ion concentrations inside and outside the cell. R_L has no voltage or time dependency. The capacitance of the cell membrane is again denoted by C_m.

Even with the advancement and development of many more membrane-current describing mathematical models with varying complexity, a full mathematical description of the electrical properties of the heart (cardiac electrophysiology) remains a highly complex problem with an enormous amount of degrees of freedom. As such, relating real-life cardiac complications to the building blocks of these mathematical models is challenging. For this reason, they are commonly simulated.

1.6 Implementation of the cardiac cell in simulations

Simulations require the cardiac cells to be represented in a way that's simulatable. Representing each and every cardiac cell is not yet computationally feasible. Simulations are then commonly done by solving the transmembrane ion flow differential equations on a grid. This requires a finite element model: a representation of the heart that's broken up in little parts. Since the mesh finite elements are not equivalent anymore to actual cardiomyocytes, the Hodgkin-Huxley model needs some adaptations as well. A model that solves the equations for ionic currents in simulations are called ionic models. Models that attempt to discretely describe ion currents and processes within the cell are call biophysically detailed. Other models may use a set of ion currents that reproduce the same behaviour to a faithful degree. These currents are not actually biophysical, but can be considered to be an amalgamation of biophysical currents. The latter class of models is called phenomenologically detailed, as they can reproduce only the resulting phenomena of the ionic behaviour of the cell or tissue, and not the actual underlying discrete ion currents. A third class of models leans close to the phenomenologically detailed models in the sense that they can only reproduce resulting phenomena. They dictate transitions of the cells according to a set of rules instead of ion currents. Choosing the right model for the desired result is dependent on the behaviour that one wants to research. It is often a trade-off between computational efficiency and a detailed behaviour of the tissue.

1.6.1 Courtemanche ionic model

The ionic models can be implemented in varying ways and different ionic simulation models use different sets of ionic currents. Only the ionic currents used in the Courtemanche ionic model [3] are discussed in this thesis. The transmembrane currents included by this model are given in Equation 1.3.

$$\begin{aligned} I_{ion} = I_{Na} + I_{to} + I_{K1} + I_{Kur} + I_{Ks} + I_{Ca,L} \\ + I_{p,Ca} + I_{NaK} + I_{NaCa} + I_{b,Na} + I_{b,Ca} \end{aligned} \tag{1.3}$$

2 Background: machine learning

Machine learning is an umbrella term for a plethora of high-dimensional data analysis tools. It is related to deep learning (DL) and artificial intelligence (AI) as these too analyse data based on so-called features and are, due to their capabilities of analysing high-dimensional data, often applied in similar fields.

Machine learning distinguishes itself from AI and DL in the sense that it needs more direct architectural input. The input data needs to be processed a certain way, often dependent on the machine learning model, before an analysis can be done. This processing of data to make it an input for machine learning is called feature engineering and is, in the case of machine learning, done by human hand. Most machine learning models have parameters that need to be set and tweaked, depending on the data, precision and inquired predictive result.

Two main approaches of ML can be identified. On the one hand we have **supervised** machine learning, where the ground truth of some dataset is known, and the model uses this information during the training phase, i.e. adapting its parameters to fit the given data. The other category is called **unsupervised**. Here, the ground truth is not known and the outcome of a certain prediction cannot be cross-checked against this ground truth. The latter is useful for identifying inherent underlying structures in the dataset and is commonly used as an exploratory method, but can also have predictive value. A third approach consists of an in-between scenario, where some, but not all of the data is labelled. This is called semi-supervised machine learning. In this thesis, we will use supervised machine learning only.

2.1 Ground truth

It is hardly ever feasible to manually check every datasample and assign a ground truth. an automated method is needed for this thesis. To this end, Directed Graph Mapping (DGM) [14] was used to automatically detect locations of re-entry. DGM is a diagnostic tool using network theory on cardiac excitation to determine the source of a given arrhythmia. After processing the input and analyzing the resulting graph, it is capable of detecting reentry - macro, localized and functional - as well as focal activity. It should be noted that, when using this as a ground truth, one cannot measure the “true” accuracy of a machine learning model, but rather the level of correspondence with DGM. This is sufficient for now, as this thesis mostly aims to check whether or not it is feasible to analyse mapping data with machine learning. A robust analysis of its accuracy is beyond the scope of this work.

2.2 Features

Supervised machine learning is a type of data analysis that seeks to find correspondence between some data and some outcome. If this outcome is a continuous value, the machine learning problem is called a regression problem. If it’s a single value or label, it’s called a classification problem. Classification can be done for a single class (e.g. “is this an apple or not?”). This type of machine learning will be applied to distinguish focals from re-entries. Classification can also be done for multiple classes (e.g. “is this an apple, orange or banana?”). If some input can belong to only one of multiple classes, the machine learning problem is called a multi-class problem. On the other hand, if it can belong to multiple classes at once, it’s called a multi-label problem. Distinguishing different types of re-entry will be approached as a multi-label problem, as the labels will refer to the location of rotational activity, and there can be multiple locations of re-entrant mechanisms at the same time.

It is only when the data is used as an input for some machine learning model that the data is called a *feature* (or a set of features). It’s perfectly possible to use the data directly as a feature. In the case of analysing local activation time (LAT) data, this would mean using each $(3 + n)$-dimensional point (3 coordinates and n LATs) as a feature, and each point would have a label indicating whether it belongs to a simulation of focal beats or re-entries, or a label referring to the location of the re-entry.

The other way to create features is to actively design them. This involves creating a data representation to extract features from. A feature can be anything at all, and it tends to be beneficial to include as much of them as possible. They should, however, reflect the underlying class or value that the machine learning model will try to predict. This correspondence does not need to be one on one (if it was, there would be no need for machine learning), but some correlation or expected correlation might be of interest. The feature should encapsulate some piece of information about the class or value that's to be predicted. It is up to the machine learning model to interpret how exactly this information is encoded.

2.3 Accuracy

A point should be made about the concept of accuracy within machine learning. Measuring the predictive value of a model should be done with appropriate caution. It should also be known that there are multiple ways for measuring accuracy, and there is no one ideal way for all cases.

Let's first discuss the concept of a score. Scoring a model means assigning a number to the model that reflects how well it can predict some outcome. As this is in most cases the last step and, more importantly, the selection criterion for model building, this is no trivial matter. One might be tempted to simply use the accuracy as a score, i.e. the percentage of correct predictions. While this is certainly possible, it has some caveats. First of all, this does not reflect the success of a model in the case of imbalanced datasets. An imbalance refers to the ratio of different data entries.

One example that will shed light to the caveats of scoring a machine learning model is a notorious machine learning exercise "Machine Learning from Disaster" on Kaggle. In this project, one aims to predict who died on the Titanic based on a selection of features, such as cabin number, age, wealth, fare price etc... Only about 710 people survived the disaster, while over 1500 lost their lives. Simply assuming everybody has died already yields an accuracy of at least 68%. While this is a great score for how simple the model is, this is obviously not a very good model as it has no predictive value whatsoever.

In addition to data imbalance, it is also possible that in some cases one wants to avoid false positives or false negatives at all cost. Imagine an AI driven cancer detection software that has a 99% accuracy rate in correctly identifying cancer cells, but also falsely identifies healthy cells in 10% of the cases. This would give rise to a lot of people undergoing chemotherapy for no reason whatsoever. It is for these reasons one should be careful when choosing how to score a model.

ROC AUC score The scoring method in this thesis will be the Receiver Operating Curve (ROC) Area Under the Curve (AUC) score. A Receiver Operating curve is a way of visualising model accuracy while including information about false positives. It plots the False Positive Rate (FPR) against the True Positive Rate (TPR). The false positive rate is the ratio of entries not belonging to a class that are falsely classified as belonging to that class. The True Positive Rate is then the ratio of the correctly classified ones. The ROC curve plots these two values for different decision thresholds. What exactly a decision threshold entails depends on the machine learning model, but it reflects how strictly a model penalizes wrong classifications in contrast to how eager it is to include correct classifications. The derivative of the ROC curve is a measure for how much false positives a model has to allow in order to include true positives. A perfect model would have a 100% TPR without any false positives. The curve would go up vertically at 0 FPR, and allowing any false positives has no further influence on the TPR. In reality, some false positives tend to be present, and the ROC curve becomes a convex curve starting at $(0, 0)$ (no false positives, but also no true ones) and ending at $(1, 1)$ (accepting all entries as belonging to some class creates a 100% TPR, but also a 100% FPR). The area under the curve (AUC) is now a measure for how well a model deals with this trade-off. High TPR for low FPR gives rise to a AUC score close to 1. The baseline for this score is 0.5: randomly assigning labels according to their frequency in the input data will, on average, give one true positive per false positive, yielding a straight line from the origin to $(1, 1)$ and an area of 0.5. An example of this curve is given in Figure 2.1. The terms specificity and sensivity are defined as:

$$\begin{aligned} Specificity &= 1 - FPR \\ Sensitivity &= TPR \end{aligned} \tag{2.1}$$

Folding A way of measuring the accuracy of a model during the training, and thus figuring out if the model is doing a good job, is by cross-validating it against other data. In other words, once a model has some parameters, it adapts their data-dependent parameters (aka training) and tries to predict the outcome of some known data that was not used for the data-dependent parameter adaptation. This can be done multiple times, each time with another sample of data that will be kept at bay. This is known as K-Fold cross-validation. The folds refer to the amount of batches the data is split in. Each time, the model will adapt their data-dependent parameters to $K - n$ batches of data (the so-called "folds") and use n folds to cross-validate its accuracy. Eventually, every combination of $K - n$ training- and n cross-validation databatches is exhausted. The overall score of the model can now be calculated as the mean of the scores of each training/cross-validation batch permutation.

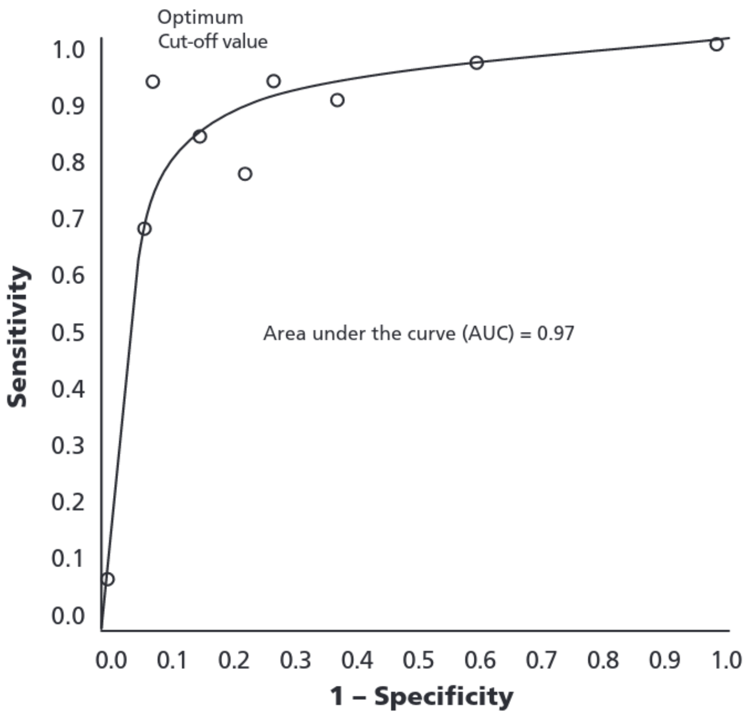


Figure 2.1: An example of a ROC curve. Figure taken from Fan [15].

Training and test scores All the aforementioned scoring methods are examples of **training scores**. Training scores involve assigning a score to outcome predictions based on input data that is also used to fit the model. As such, a training score is a measure for how well a model is fitted, but it should not be considered as an equivalent to its predictive value. Since the same data that is used to calculate the score as to fit the model, the training score carries a bias; the model is fitted exactly to maximise this score, but only for the considered input data. New data that was not used in the fitting process might have a vastly different score. Since the prediction of new input data is exactly what we want for a good machine learning model, the training score is not a good measure. If a model can distinguish two classes or values based on some feature, it is said that the model has "captured" the feature. Ideally, we want a model that represents the data well and also generalises to unseen samples. The training score gives no indication how well a model will generalise to unseen samples. In fact, a good training score can be a sign of a good model, or a sign of an over-fitted model. Over-fitted models capture noise or sample-specific quirks in the data as if they were features, and do a very good job at predicting only the specific dataset that was used to train on, but not necessarily other unseen samples. Finding the best possible model that is not overfitted is a bias-variance trade-off problem. Good data representation corresponds to a low bias, while a good generalisation corresponds to a low variance. Training models to a high degree and making them more complex might represent the data better (decreasing bias), but has a high risk of overfitting (increasing variance). Similarly, simpler models don't overfit (decreasing variance), but may underfit the training data, failing to capture important regularities (increasing bias).

It is for this reason that another type of score is used in machine learning: the **test score**. This score refers to the predictive success of a model on an unseen dataset. To this end, the input data is split into two parts. One part will be used to train on, the other will be kept at bay until after the model has been fully trained (unlike out-of-fold cross validation). The more data there is for testing the model, the more accurate one can test how well it generalises. On the other hand, the more data we keep for testing, the less data there is left for training the model. A typical ratio for training-test data is $80\% - 20\%$. A good way of visualizing this is by means of **learning curves**; these curves plot out the training and the test scores for different amount of training and test samples. If both the training and the test score keep going up with the inclusion of more and more data, it means that the model is still underfitted, and needs more data to capture the features better. On the other hand, if, at some point, the training score goes up, but the test score goes down, the model has been overfitted. In the end, it is the test score (also called cross-validation score, not to be confused with K-fold cross-validation) that gives the best idea of a model's success at capturing features and predicting outcomes.

2.4 Adapting models for multilabel classification

Some models have native support for multiple possible outcomes, such as the Random Forest model or Extra Trees. Most models, however, do not have this support. One way, and the way that was used in this thesis, is the One vs Rest classification method (also called One vs All). With this method, the model iterates over all n classes and tries to decide whether some input belongs to one class instead of the other $n-1$ classes. This approach is a way of reducing the multilabel problem to a single class classification problem. Each iteration, one class is treated as positive, while all the other classes are bundled and treated as negative. Each prediction comes with a probability score. The class corresponding to the largest probability score is considered to be the predictive class label. A model needs to be trained for each class in this case.

Another possible approach is the One vs One approach, where, instead of bundling all the other classes as negative, each possible class pair permutation is considered. This approach is beneficial for models that do not scale well for different sizes of datasets; the One vs Rest method bundles $n-1$ classes together to treat as one and rare classes will be underrepresented in this bundle. The downside is that this method requires $n(n-1)/2$ models to be trained, whereas the One vs Rest approach only requires n.

3 Introduction

Heart diseases are one of the leading causes of death in western countries. Due to the high complexity of the organ, a lot of the underlying mechanisms remain hard to describe. As a result, it can be very difficult to predict or cure propagation abnormalities, often requiring highly advanced tools and leading experts in the field to operate these and interpret the acquired data. This makes diagnosis and treatments expensive [16] and not always successful [17]. One popular treatment involves the ablation (burning) of cardiomyocytes [16, 18], making them unable to propagate a signal. Identifying the optimal ablation region remains a difficult problem.

The complexity of analysing cardiac arrhythmia mechanisms have attracted multiple approaches. One novel approach that has become increasingly popular and successful is the analysis of cardio-electrophysiological data through machine learning and artificial intelligence; two mathematical architectures that are highly efficient in encapsulating high-dimensional and large datasets. Depending on the exact data that was extracted from a heart, this approach has been used in the field of electrophysiology in many ways already. ECGs are a common datatype for machine learning input [19] and can e.g. predict tachycardia types [20] and atrial fibrillation before it starts [21], and has been used to find areas of rotors sustaining AF [22]. In addition to using ECGs as an input, CT images have been analysed with deep learning [23]. Machine learning has been applied for ion channel level analysis such as the influence of the SCN5A gene on the cardiac sodium channels [24] and identifying structure/function relationships in voltage-gated cardiac potassium channels [25] [26], as well as the analysis of computational electrophysiology [27, 28, 29, 30] and much more. Despite rapid advancements, much of the machine learning applications to cardiac electrophysiology is still in its nascent state when it comes to uncovering arrhythmia mechanisms [31] and has barely been used to analyse contact intracardiac electrograms during atrial fibrillation (with the purpose of guiding ablation procedures for example) [32]. They have not yet been used to analyse local activation times from cardiac mapping data at all.

This thesis will only consider the regular arrhythmia called atrial tachycardia. Tachycardia are defined as an unusual fast heart rate; faster than 100 beats per minute in rest for a normal adult person. We will only analyse tachycardia developed in the left atrium. These tachycardia can be categorised as either reentrant or focal in nature (see Section 1.3). These tachycardia are known to induce atrial flutter, even after catheter ablation [6].

The aim of this thesis is to expand the current applications of ML analysis of cardiac data to atrial mapping data too. In particular the local activation times as measured by this mapping data. The goal is to train a model that can accurately predict the location of reentrant mechanisms, which are known to sustain atrial fibrillation [33]. It will first explore the possibility to use supervised machine learning to distinguish between focal beats and re-entries based on in-silico mapping data. Following, it will explore the possibility to distinguish between three locations of anatomical re-entries in the left atrium: the left pulmonary veins, the right pulmonary veins and the mitral valve.

Due to the low availability of real patient data, the first part of the thesis will expand upon data multiplication. Based on in-vivo data of the atrial geometry and local activation times of 38 patients, a 3D mesh will be reconstructed per patient. Multiple ways of creating variations of the same reconstructed mesh will also be discussed. These mesh reconstructions will consequently be used in 200 electrophysiological simulations per patient model (100 focals and 100 reentries), each producing in silico local activation times data. From this data, several features will be considered and analysed with multiple machine learning models in order to gain insight on the applicability of the different models to different features.

The first machine learning problem is to distinguish focal beats vs re-entry. This will be analysed in two ways. First, by using the LAT data directly as an input for machine learning models. Secondly by creating a new data representation and extracting features from this representation. This is also how the analysis of the second distinction (different re-entry locations) will be done.

4 Methods

This section will provide a detailed rundown of the methods used to produce in-silico data and how this was used in machine learning analysis.

Section 4.1 will discuss the pipeline that reconstructs a Carto `.mesh` file to a mesh that can be used in such simulations. This will include a section on how a single mesh can be copied with variations in terms of conduction velocity. Section 4.2 will start by giving a quick overview of the OpenCARP simulation environment [13] in Section 4.2.1 and explain how the meshes need to be prepared in order to run these simulations in Section 4.2.3. Section 4.3 will expand on how the data produced in the simulations was extracted and processed. It will give an overview of the features that were constructed from this data, how they were calculated and the idea behind them. In order to keep a good overview, a flowchart of the sections is provided in Figure 4.1.

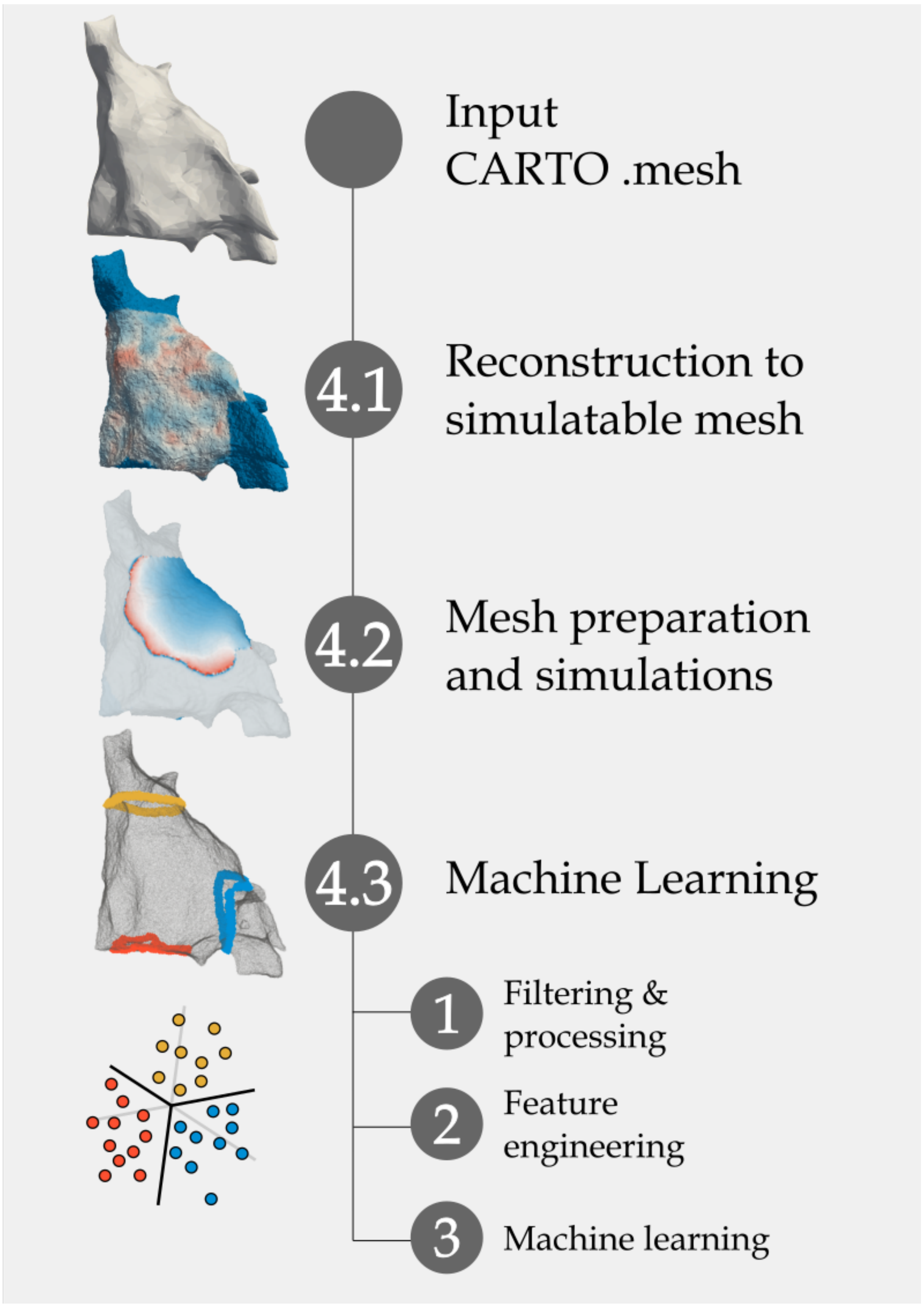


Figure 4.1: A flowchart of the subsections of Section 4: Methods, showing the progress from an input CARTO `.mesh` to a reconstructed mesh, to a simulation, and finally to machine learning features. Each numbered circle represents a subsection and is equipped with a figure representing the result of each subsection.

4.1 Mesh reconstruction from in-vivo mapping data

In order to run OpenCARP simulations, one needs a mesh that is in correspondence to the requirements of OpenCARP. A variety of wireframe, surface and volume meshes are supported for OpenCARP simulations: line meshes (linear and cubic), triangles, quadrilaterals, tetrahedrons, pyramids, prisms and hexahedrons.

The mapping data that's to be analysed are CARTO `.mesh` files provided by Biosense Webster Inc. and are triangular surface meshes. While these types of meshes are technically supported by OpenCARP and can be directly used for further simulations, two core issues arise in this case:

1. The resolution is too coarse for an activation front to propagate
2. This is a simplification of the real-life case that does not mirror the true case as well as it could, even losing some core dynamics such as transmural propagation and full 3D conduction anisotropy. Including this might be useful for future use.

The supplementary material provides a pipeline that discusses how to create a refined tetrahedral mesh with heterogeneous conduction regions that can be used for OpenCARP simulations. No support for conduction anisotropy has been implemented yet. The pipeline makes use of TetGen [34] to create a tetrahedron mesh from a bounding surface layer. The thickness of the mesh is kept thin to avoid transmural conduction [35] to play a significant role in the wave propagation dynamics or inducing endo- epicardial dissociation of electrical activity, which would add unnecessary extra complexity [36] to the simulations. DGM [14] is used to calculate conduction velocities based on in-vivo LATs and these are used to make variations of the original conduction velocity distribution. DGM is a diagnostic tool using network theory (Directed Graph) on cardiac excitation.

4.2 OpenCARP simulations on HPC

4.2.1 OpenCARP simulations specifications

The simulations will be run using the OpenCARP ecosystem [13]. OpenCARP is an open-source software system offering single cell as well as multiscale simulations from ion channel to organ level. This ecosystem contains several different programs and utilities that work together in order to prepare, calculate and analyse electrophysiological simulations. Below is a quick overview of the different programs and utilities.

OpenCARP This is the backbone of the ecosystem and consists of the simulator with all the different solvers, written in C++. Direct interaction with this simulator is possible by means of parameter files, specifying all the variables for a single simulation. Providing a parameter file for each simulation is tedious work if not for a parameter-generating script or some other automated interaction with the simulator.

CARPutils The python-based CARPutils framework enables the development and sharing of simulation pipelines, i.e. automating in-silico experiments including all modelling/simulation steps. This allows for easier simulation setup and automation, making it possible to run multiple simulations with different parameters.

Meshalyzer Meshalyzer is a graphical program that can display time dependent data on 3D finite elment meshes. It is developed to be compatible with the cardiac simulation environment OpenCARP. Meshalyzer can read in and visualize the binary `.igb` files; one of the outputs of OpenCARP simulations. It allows for the creation of videos of the simulation outputs.

Meshtool Meshtool is a command-line tool written in C++ that supports the OpenCARP file formats. It can do a selection of mesh operations, such as imporving the quality, extracting submeshes and converting from and to other mesh file formats such as the `.vtk` format. The latter option has been used frequently to allow for mesh operations in other programs (such as ParaView [37] or MeshLab [38]) that do not support the OpenCARP file formats.

Simulations will be run with the parameters as described in Table 4.1.

4.2.2 The HPC infrastructure

Simulations will be run on the victiny cluster of the High Performance Computing core facilites (HPC) of the Flemish Supercomuter Centre (VSC). They will be run in seven batches total. Each simulation will be run using 36 cores. The binary `vm.igb` output files will be saved for visual inspection where necessary and the `depol-thresh.dat` output files will be saved for the machine learning analysis (infra).

4.2.3 Simulation prequisites

Per mesh, 100 focals and 100 reentries will be simulated. Simulations will be run by selecting a stimulus region and, in the case of simulating a reentry, a block region. This stimulus will be propagated using the Courtemanche ionic model [3], resulting in a depolarising wavefront. Anatomical re-entry will be induced around one of three anatomical structures in the atrium: the mitral valve, the left pulmonary veins and the right pulmonary veins.

A stimulus is an external source that triggers depolarisation in cardiomyocytes. A block is a selection of cells that do not depolarise and thus do not allow wave propagation for a limited period of time (around 200 ms). In addition to being non-conductive, a region that is selected as a block is clamped on the resting potential of -81.2 mV. When timed correctly, this should prevent a depolarising wave from propagating in one direction, but not the other. If the block disappears by the time the wave propagated around some anatomical region and arrived at the same location again, it allows the depolarising wave to propagate further. To induce macro re-entry, this block region should connect two anatomical objects so that the passage of a wavefront between these anatomical regions becomes impossible. The block should be present longer than the APD of adjacent cells, so that they can't induce further propagation when the block disappears. The block should also not linger around for too long, however. When the depolarising wave has made its way around the atrium, it should be able to continue its way around. As such, the block should have a slightly shorter period than the cycle length of the depolarising wavefront around the anatomical region of interest.

The OpenCARP simulation environment can read in locations of stimuli and blocks by means of a `.vtx` file, of which an example is shown in Table 4.2.

Meaning	Parameter name	Value
Simulation time resolution (ms)	dt	20
Time between spatial output (ms)	spacedt	5
End time of simulation (ms)	tend	2000
Experiment label	experiment	0
Use parabolic solver	parab_solve	1
	mass_lumping	1
Amount of stimuli	num_stim	1
	stimulus[0].vtx_fcn	1
	stimulus[0].start	1000
	stimulus[0].strength	80
	stimulus[0].duration	0.1
	stimulus[0].stimtype	0
	stimulus[0].x0	0
	stimulus[0].xd	1
	stimulus[0].y0	0
	stimulus[0].yd	1
	stimulus[0].z0	0
	stimulus[0].zd	1
	stimulus[0].vtx_file	stim
N conduction regions	num_gregions	1
	gregion[0].g_il	0.7787
Conduction velocity scaling in intracellular (i) or extracellular (e) region for longitudinal (l), transversal (t) or normal (n) direction	gregion[0].g_it	0.7787
	gregion[0].g_in	0.7787
	gregion[0].g_el	2.7970
	gregion[0].g_et	2.7970
	gregion[0].g_en	2.7970
	gregion[0].num_IDs	1
	gregion[0].ID[0]	0
N physical regions	num_phys_regions	1
	phys_region[0].num_IDs	1
	phys_region[0].ID[0]	0
	num_LATs	1
	lats[0].all	1
	lats[0].measurand	0
	lats[0].threshold	-75
	lats[0].ID	depol
	lats[0].method	1
	num_tsav	1
	tsav[0]	2000
	write_statef	backup

Table 4.1: Parameters used in OpenCARP simulations. Greyed out parameters were (re)defined in the CARPutils Python framework.

Format	Example
n_indices	106
”intra” or ”extra”	intra
index_1	0
index_2	6
...	...
index_n	539

Table 4.2: `.vtx` file format

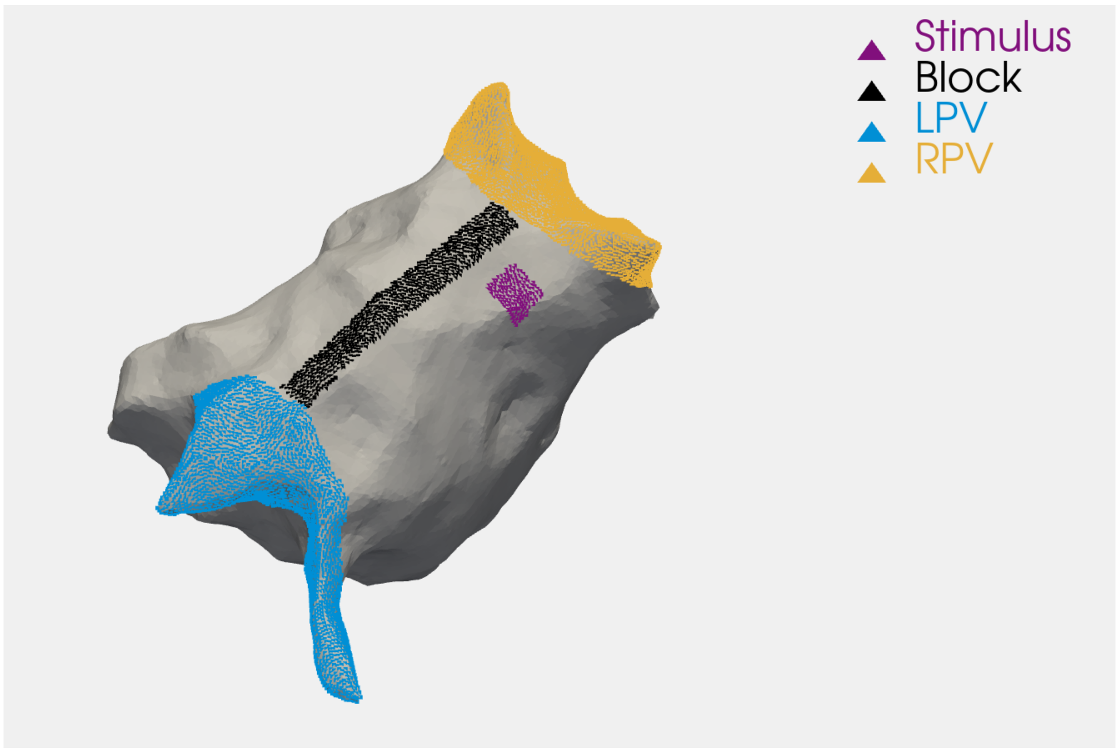


Figure 4.2: An example of a stimulus (purple) and block (black) on the atrial roof. The stimulus will induce a depolarising wavefront. This wavefront will be unable to pass through the block as long as this block is present. The block has to disappear before the wavefront has propagated around the right pulmonary veins (yellow), but has to linger long enough so the stimulus can’t induce propagation in the clockwise direction.

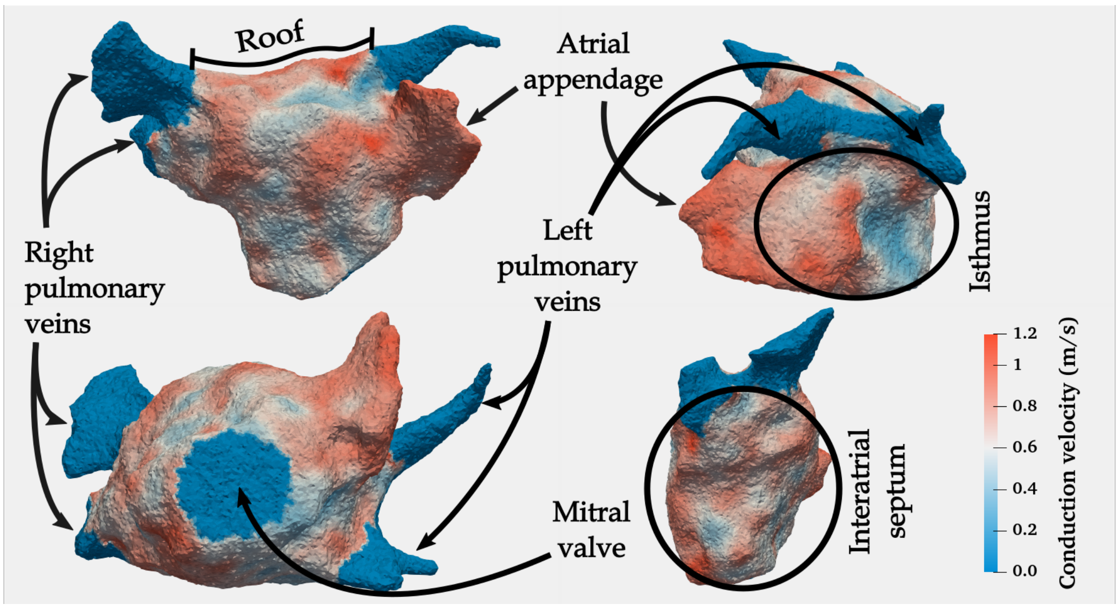


Figure 4.3: A reconstructed 3D model of an atrium showing the names of various anatomical regions.

Parameter	Original	Modifier	Result
g_CaL	0.1238	$\times 0.25$	0.03095
g_Kr	0.0294	$\times 4$	0.1176

Table 4.3: Courtemanche ionic model parameter modifications

This selection procedure was done with Paraview. To this end, each mesh point was equipped with their index as scalar data. This scalar data is remembered by Paraview throughout mesh operations. After selecting a stimulus region on a mesh, the point indices were exported and converted to `.vtx`. The same procedure can be done for the selection of block regions. For each mesh, 10 stimuli were selected throughout the atrial myocardium: two at the isthmus, four on the roof and four on the interatrial septum. These locations are shown in Figure 4.3. If the septum or the isthmus of the atrium was too small for these selections, a location as closeby as possible was stimulated instead. In combination with a mesh, each stimulus region can now be used for a simulation of focal beats. Additionally, 10 block regions were also selected in correspondence to the stimuli locations, such that combination of stimulus, block and their corresponding mesh can be used for a reentry simulation.

In order to ensure that the circumference of the atrial structure L is larger than the action potential duration times the conduction velocity $L > v \cdot APD$, two parameters of the Courtemanche ionic model were modified. These are shown in Table 4.3 and shorten the APD.

4.3 Mapping data processing and feature engineering

The following section will focus on how the LAT data as generated by the OpenCARP simulations will be filtered, processed and manipulated into features for supervised machine learning.

4.3.1 Generating a ground truth

Ground truths for the focal and reentry classification problem was already present by preparing the simulations and were easily extracted. A ground truth for the reentry location was generated by making use of DGM. DGM can determine the source of a given arrhythmia. It is capable of detecting reentry - macro, localized and functional - as well as focal activity.

4.3.2 Processing and filtering in-silico cardiac mapping data

For each simulation, the coordinates and corresponding LATs were extracted. The mesh coordinates were rotated to align with some reference mesh and represented in both cartesian coordinates (x, y, z) and angular coordinates (θ, ϕ).

For further processing steps, the cycle length needs to be calculated. To this end, the angular coordinates (θ, ϕ) of the activated points were sorted by their respective activation time. This time-ordered data was passed through a low-pass filter and a sine function was fitted to this low-pass filtered data. The frequency of the fitted sine function is now equal to the cycle length. A plot of the time-ordered coordinates with their low-pass filtered variants and a fitted sine function is shown in Figure 4.5, for both a full simulation and a subsampled simulation with 2000 LAT points. Filtering was done using the `scipy` module `signal.butter()` with parameters as shown in Table 4.4.

As we only want to analyse stable simulations, we need to check whether or not the simulations contain a stable wavefront during the last cycle length. To this end, the time intervals of each node of the simulation were checked and compared

	Full simulation	Subsampled simulation (2000 points)
Critical frequency	3.3e-4	3.3e-3
Order	1	
btype	'low'	
analog	False	

Table 4.4: Parameters used in the Butterworth low-pass filtering of time-ordered angular coordinates.

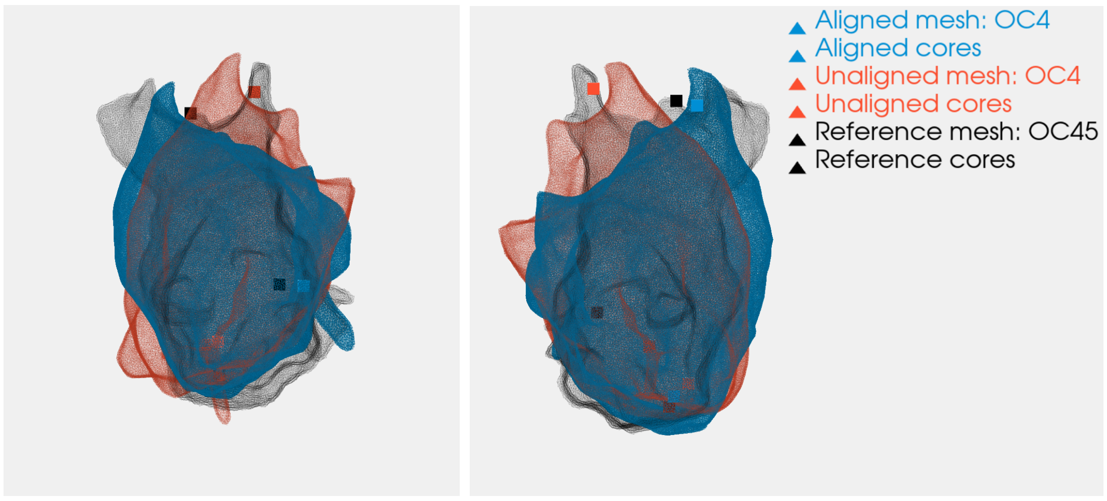


Figure 4.4: Example of mesh alignment. The "cores" refer to the centers of the RPV, LPV and MV.

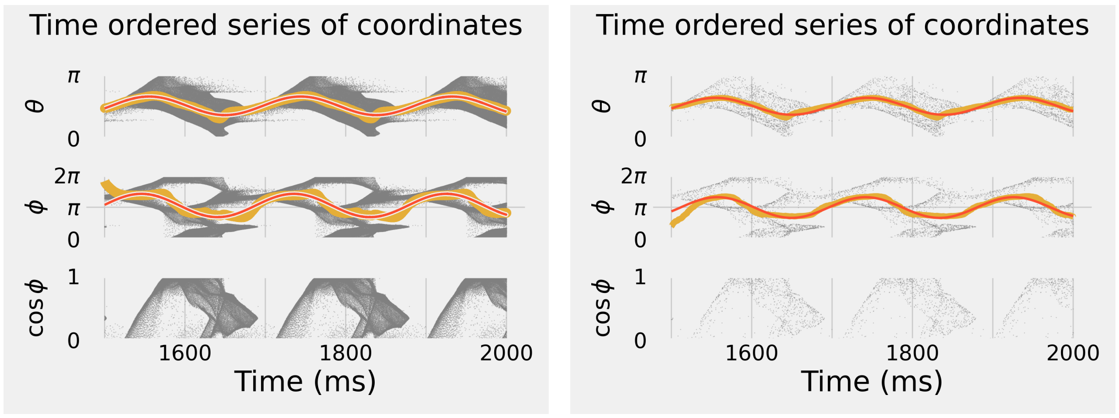
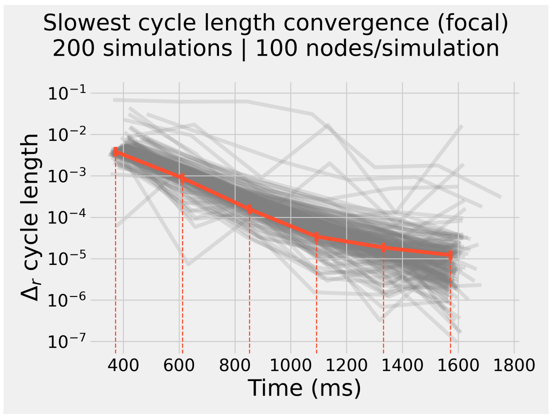


Figure 4.5: Time-ordered coordinates of a full simulation (left, 55105 points) and a subsampled simulation (right, 2000 points). The low-pass filtered coordinate series is show in yellow. The fitted sine is shown in red.



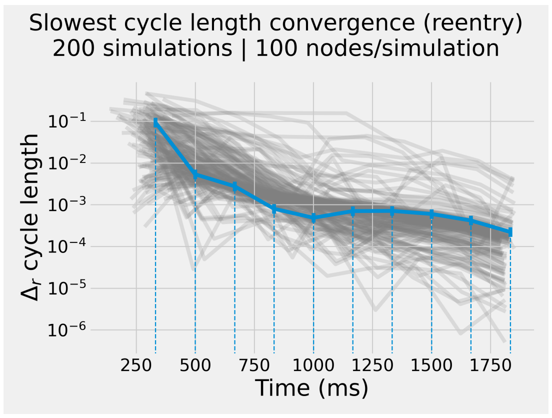


Figure 4.6: Convergence of 100 slowest converging nodes for 700 randomly selected simulations: 200 focals and 500 re-entries. At the end, each simulation's slowest converging nodes are within 1% relative difference to their last LAT interval.

to their last time interval, corresponding to the last cycle length. Figure 4.6 shows the 100 slowest converging nodes for 700 randomly selected simulations. If no activation was found during the last cycle length, the simulation was considered to be failed and was left omitted for further analysis. If a simulation had nodes that were not within 1% relative difference to the last measured time interval, they were considered non-converged and left omitted for further analysis. Only the last cycle length of each remaining simulation was used in further analysis. As real-life mapping data only has a couple of thousand datapoints at best, each simulation was subsampled to 2000 points to more closely resemble clinical settings.

4.3.3 Feature engineering from processed cardiac mapping data

The previous section provided us with processed cardiac mapping data. The current section will discuss how to extract features from this processed data.

The first method, as discussed in Section 2.2, feeds the LAT datapoints directly as features to a machine learning model, without any feature engineering steps.

The second method involves the engineering of features. These features will be subdivided into two main types: global and regional features. Global features refer to features that reflect information about the entire myocardium, rather than some statistic around or referring to a specific anatomical region. The regional features can again be subdivided into two main types: features referring to the **amount** of activation and features referring to the **location** of activation.

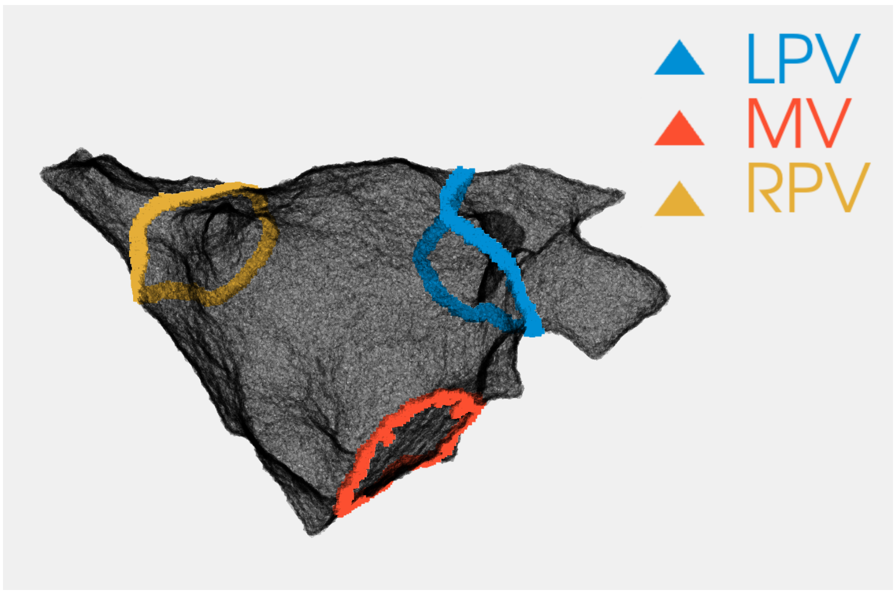


Figure 4.7: Example of a selection of the myocardium around the three anatomical regions of interest: the left pulmonary veins (blue, LPV), the right pulmonary veins (yellow, RPV) and the mitral valve (red, MV). The anatomical regions are non-conductive and will not have any registered LATs. The selected myocardium does have registered activation.

Global features Only one global feature was implemented: the silent time. This is the amount of time that no activation is measured across the entire myocardium. Since reentries loop around some anatomical object or region, there is sustained activity and the corresponding silent time is expected to be 0 ms. This feature alone should suffice for a machine learning model to distinguish between focal beats and reentries. In fact, this feature is so straightforward that machine learning shouldn't even be necessary for the focal beats vs reentries classification problem. It was implemented anyways as an initial explorative test to see if the data processing worked and the machine learning could interpret the resulting features.

Regional features These features reflect some information about the myocardium around anatomical objects of interest, i.e. the left pulmonary veins, the right pulmonary veins and the mitral valve. These regional features will be of value when developing the machine learning models distinguishing the locations of re-entrant mechanisms. For these features, the neighbouring myocardium was extracted from the three anatomical regions of interest: left pulmonary veins, right pulmonary veins and mitral valve. The neighbouring myocardium is highlighted in Figure

4.7. Only registered LATs in these regions will be used for further feature engineering.

The first type of regional features convey information about the **amount** of activation around some anatomical object. If an activation wavefront loops around some anatomical object, then there should be sustained activity around this anatomical object. The amount of LAT registrations per time unit (activation rate) should never be 0 Hz. Beware that an anatomical object where there is no rotational activity around it can still have an activation rate that does not drop to zero, as it's perfectly possible that a wavefront enters the neighbouring myocardium on one side when the wavefront of the previous cycle has exited on the other. Additionally, it's also possible that the time resolution of the LATs is lower than the time resolution used to calculate the activation rate; this results in empty time intervals due to a resolution mismatch, rather than an actual period of no activation. To avoid confusion with the silent time, the lack of activation around an anatomical object will be called regional quiet time. The regional quiet time is now calculated as the sum of all the time intervals that did not register an LAT. This method different from the global silent time, as the latter calculates the longest time interval without activation, and not the sum of all time intervals without activation. This feature needs a somewhat coarse time resolution. Taking the resolution (i.e. bin width) too fine will give rise to a lot of quiet time intervals, even for regions with sustained activation. This is due to the fact that these regions only contain about $100 - 200$ points in the case of subsampled simulations. The activation rate around some region with $N = 200$ points for a simulation with a cycle length of about 200 ms is expected to be

$$\langle \dot{N}_{act} \rangle = \frac{N}{CL} = \frac{200}{200\ ms} = 1\ kHz \tag{4.1}$$

, or about one registered LAT per ms. If we consider this activation rate to represent a Nyquist sampling rate of the activation wavefront signal, the Nyquist theorem tells us that this can reconstruct a signal with a time resolution/bin width of at best 2 ms. This is, however, an ideal case. Since the points around the anatomical object are not perfectly evenly sampled, and the conduction velocity is not constant, the activation rate cannot be considered a Nyquist sampling rate and this value has a considerable error. If we want to surely find a series of non-empty time intervals for the regions where there is sustained activity, the time resolution needs to be about 10 times as large. This is a ballpark value at best and no rigorous research was done to find the optimal time resolution for a given sample rate i.e. activation rate. A time resolution this large may fail to capture regional quiet

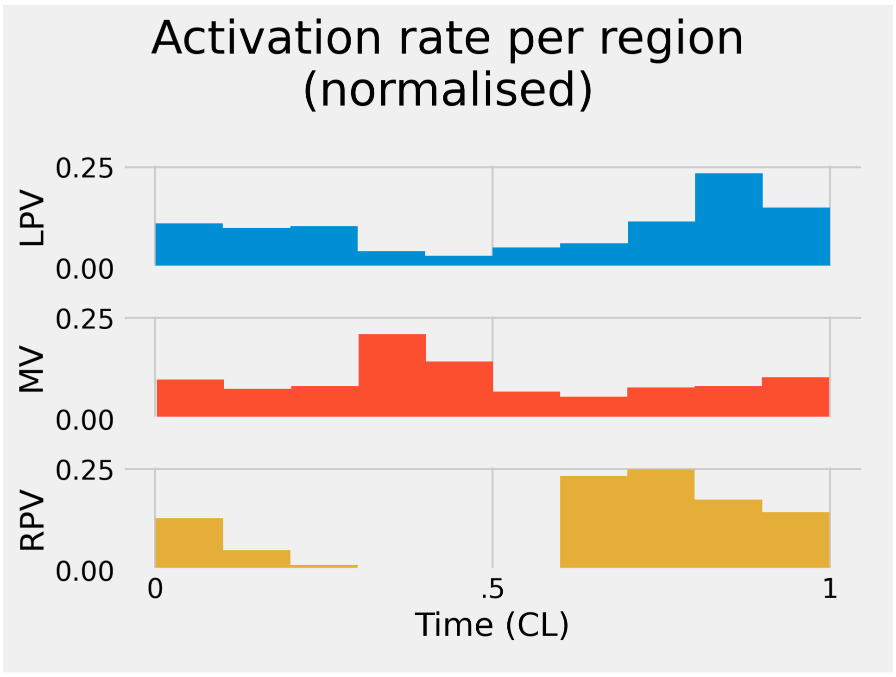


Figure 4.8: Example of the activation rate around anatomical regions for a simulation with 2000 points.

times around anatomical objects with no sustained rotational activity around it. To this end, another feature was created: the standard deviation of the activation rate.

The second type of regional features will reflect information about the **location** of the depolarising wavefront with respect to the anatomical regions. A wavefront that loops around an anatomical object can only be at one side of the anatomical object at the same time. When a wavefront passes some anatomical object at two sides at once, it surely cannot be the location of an anatomical reentry. To convey this in a feature, the average angle of activated points around each anatomical object was calculated per time interval. This results in a smooth continuous line for anatomical objects with rotational activity, but not for regions where the wavefront passes around the object at two sides at once. For the latter case, we expect the mean angles to be scattered around the mean at random. This feature is, similarly as before, dependent on the chosen time resolution, as well as possible regional quiet time. To combat empty time intervals due to the time resolution and point density mismatch, the coordinate values of empty time intervals were filled with artificial points, monotonically increasing/decreasing between the values of neighbouring non-empty time intervals. Passing these time-ordered average coordinates through a low-pass filter provides a baseline on their variation. The difference between the actual coordinates and the low-pass filtered ones is now a measure for how smoothly these coordinates vary over time. Denoting low-pass filtered coordinates with a tilde $(\widetilde{\theta}, \widetilde{\phi})$, the error is calculated as:

$$error = \sqrt{\frac{1}{N}\sum_{i}^{N}(\theta_i - \widetilde{\theta}_i)^2} \tag{4.2}$$

Filling in empty time intervals with artificial coordinates will also fill in the time intervals that were empty due to regional quiet time rather than a resolution mismatch. As this has very little influence on the value of the feature (the difference between a monotonically increasing set of values and their low-pass filtered variants is low to zero) and the information about the amount of activation is already encoded in the previously discussed features, this was left as it is.

An example of the angles and their low-pass filtered variants is shown in Figure 4.9. In this figure, it's visible how the ϕ coordinates of the time-ordered activated points around the right pulmonary veins do not follow their low-pass filtered variants. This regions probably does not have any rotational activity around it. This is better visible in Figure 4.10, where it's visible how the activation wavefront passes the RPV on both sides at the same time.

In addition to the error rate, the variance of the time-ordered mean of coordinates was also added as a feature.

4.3.4 Machine Learning

Given the features as described in Section 4.3.3, a total of nine different machine learning models were trained and compared:

- K Nearest Neighbors
- AdaBoost
- Gradient Boost
- Support Vector Classifier
- Bernouilli Naive Bayes
- Complement Naive Bayes
- Multinomial Naive Bayes

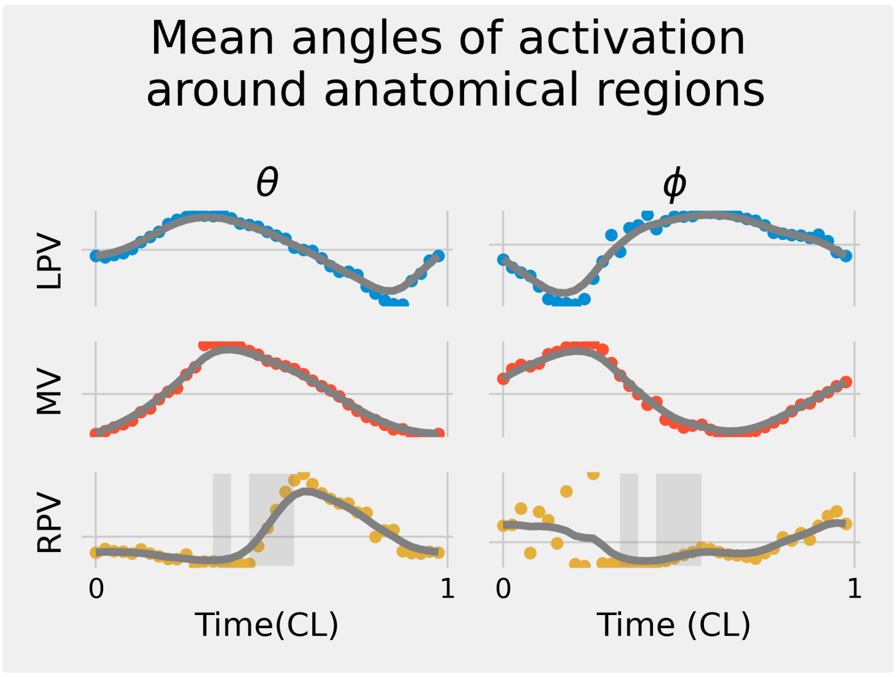


Figure 4.9: Example of a time-series of coordinates and their low-pass filtered variants. Grey areas indicate artificially filled `NaN` values.

- Random Forest
- Extra Trees

To this end, a parameter grid was created for each model. Using `sklearn.model_selection.GridSearchCV()`, the per-model parameters yielding the best score were saved, along with the corresponding score. Fitting the models was done by splitting the training data in 5 folds, training on 4 of them and using the last one for cross-validation. Scoring was done by calculating the mean ROC AUC out-of-fold score.

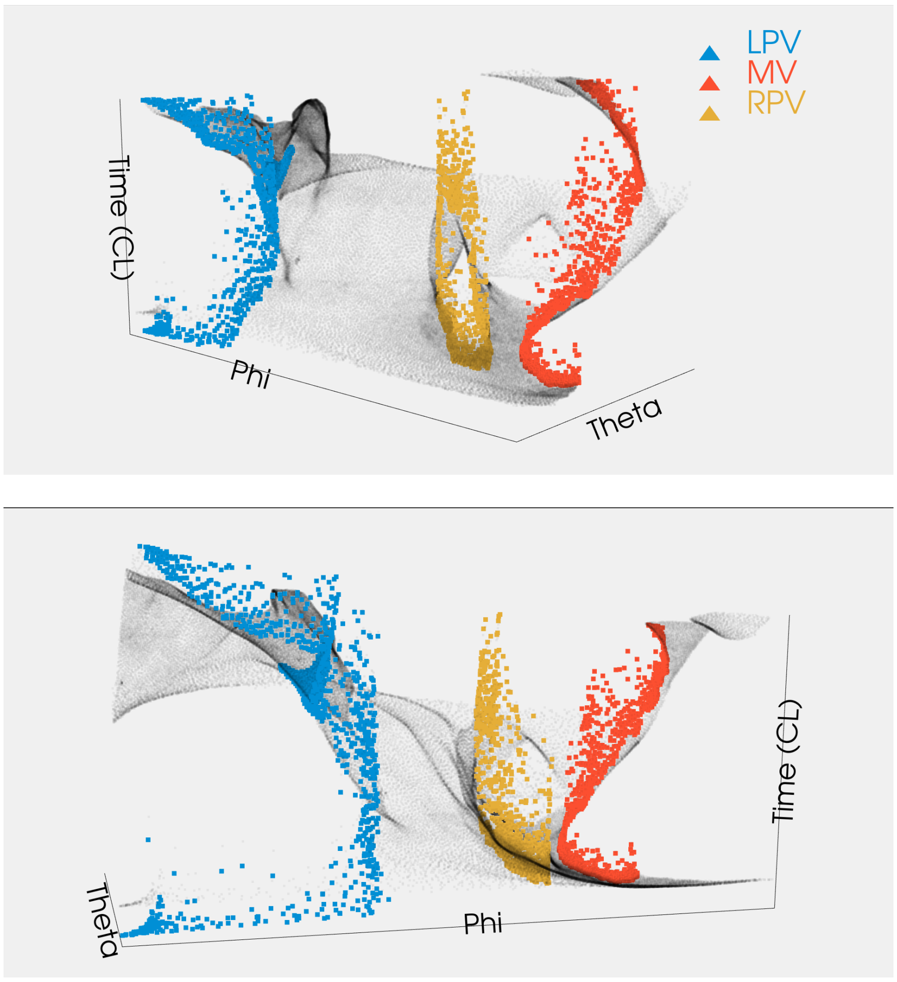


Figure 4.10: Two plots of the (θ, ϕ, t) coordinates of activated points from the same simulation as Figure 4.9. It is clear here how there is rotational activity around the left pulmonary veins (blue spiral) and mitral valve (red spiral), but not around the right pulmonary veins (yellow cylinder): the activation wavefront passes on both sides at the same time.

5 Results

5.1 Mesh reconstruction

The mesh refinement procedure can adapt the mesh edges to a desired edgelength. This can be seen in Figure 5.1, showing a boxplot of the mesh edgelength distribution at each iteration step, as well as the values for the maximum allowed edgelength (α) and minimum allowed edgelength (tolerance). For each iteration step, the length of the mesh edges are distributed closer and closer within the desired range. At the last iteration step, the amount of edges that are not within the desired range is marginal and their lengths are close to the desired range.

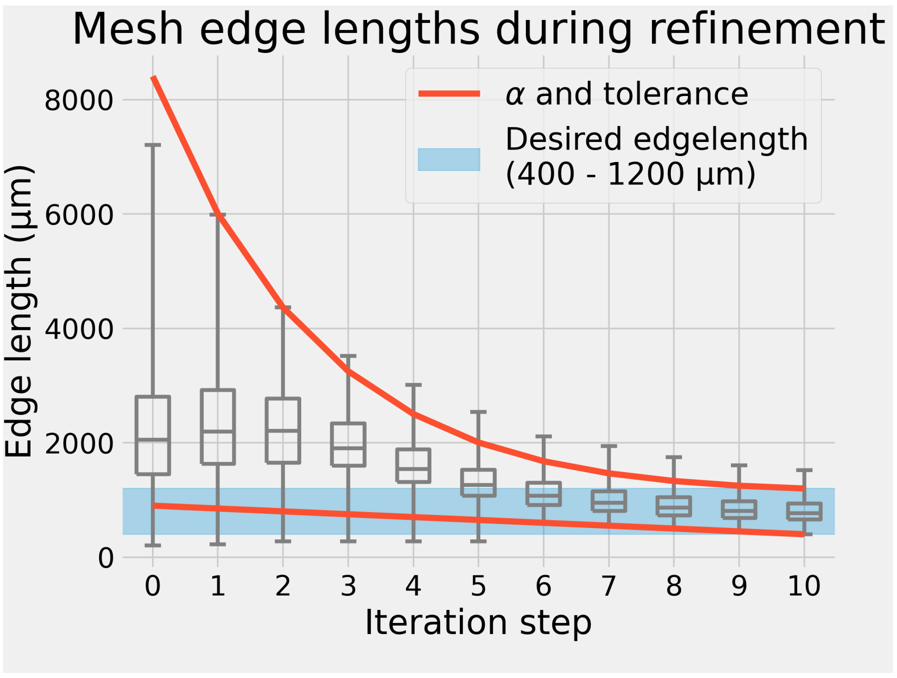


Figure 5.1: A mesh refinement procedure. The longest allowed edge (α) follows the distribution $\left(1-\frac{n}{N}\right) \cdot max(original\ edges) \cdot e^{-3n/N} + UL$, while the minimum allowed edge length (tolerance) decreases linearly as $500\mu m \cdot \left(1-\frac{n}{N}\right) + LL$, where n is the current iteration step and N is the total amount of iteration steps. UL and LL are the desired upper and lower limit respectively, which shown in blue (here: $400-1200\ \mu m$).

The reconstruction procedure can successfully make variations of these conduction regions based on the existing conduction velocity distribution throughout the mesh. It can automatically assign non-conductive regions based on tags in the CARTO `.mesh` file. They provide an efficient way of expanding sparse data. It is unclear how realistic these conduction velocity distributions are.

13 meshes did not complete the process of tetrahedralisation, of which 4 could potentially be re-run with different edgelength intervals or iteration steps. 9 failed due to unknown reasons. Possibly, the latter 9 meshes were not manifold and/or watertight. No further time was spent on trying to fix these meshes. 25 other anatomical 3D models were successfully recreated and passed on to further preprocessing steps.

5.2 Simulations

25 meshes were used in a total of 5000 simulations. 617 out of 5000 simulations did not run correctly: 214 focals and 403 re-entries. This was due to:

- Faulty selection of a stimulus or block region, preventing propagation.
- Lack of stimulus propagation due to a mesh resolution that's too low or a bad quality mesh.
- A self-extinguishing re-entry

5.3 Features

The distributions of regional features are shown in Figures 5.3-5.8, where the columns refer to the label of the simulation and the rows refer to the location at which the feature was calculated.The means are shown as a vertical lines. Keep in mind that, since this is a multilabel problem and simulations can have multiple labels at once, there is overlap between the columns. Simply because a simulation is e.g. in the LPV column does not mean it can't be in another column i.e. have another label too.

Figures 5.3 and 5.4 show that the standard deviation of the activation rate around anatomical regions and the regional quiet time are indicative of the location of reentry, as their mean is visibly lower for the feature distributions where the region of the feature corresponds to the region label. The other features show

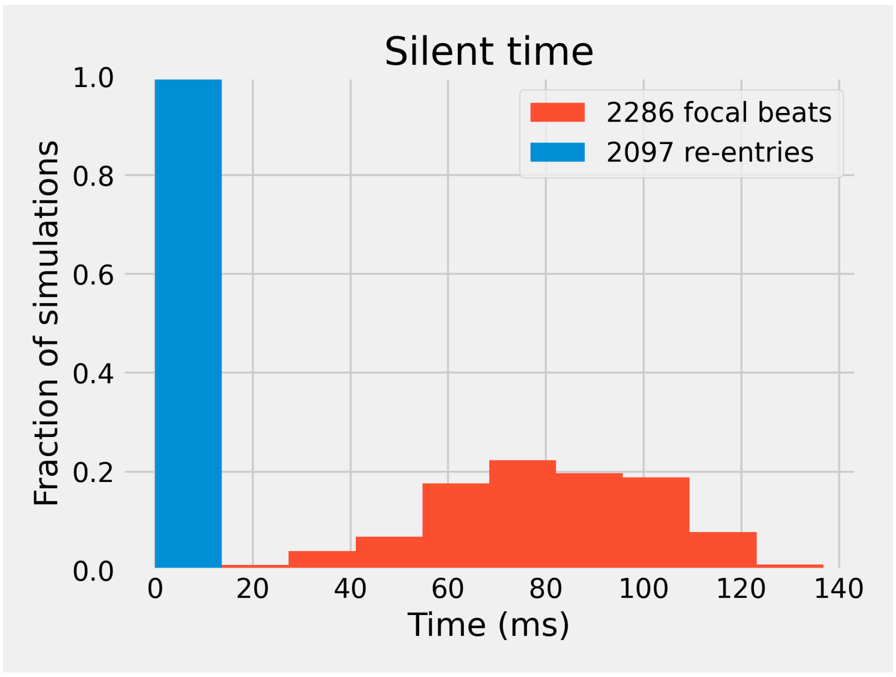


Figure 5.2: Silent time of simulations.

no obvious differences. Figure 5.9 shows that the regional quiet time and the standard deviation of the activation rate around the same anatomical region have a correlation of .8 for the left pulmonary veins and the mitral valve and .9 for the right pulmonary veins. This correlation is very high and implies that one of either feature is sufficient to encapsulate the underlying information. The silent time is also heavily correlated with the regional quiet time, with a value of .8 for all anatomical regions. Some correlation is to be expected, as a silent time will always give rise to a regional quiet time as well. Other features are not significantly correlated.

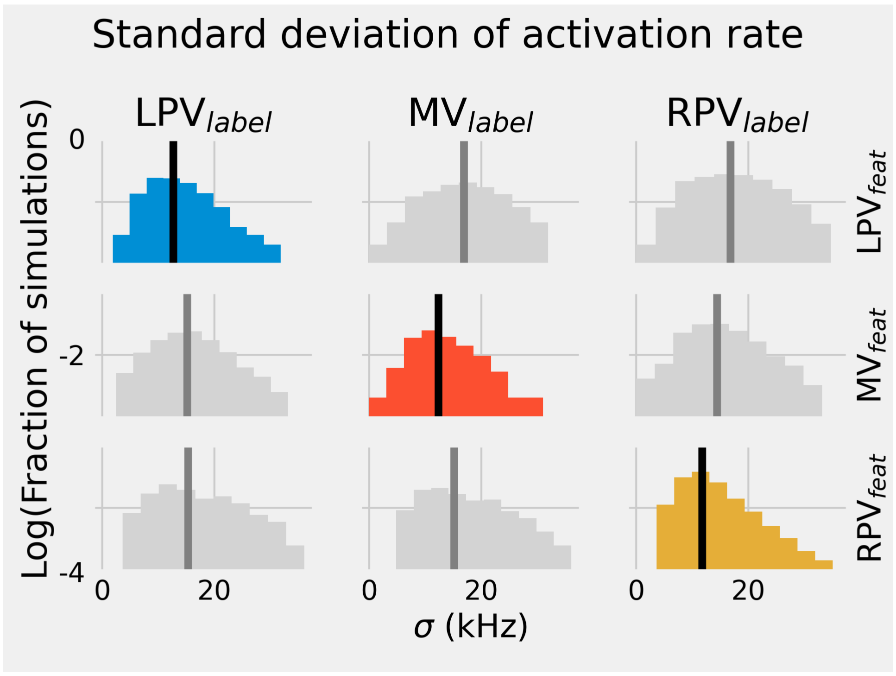


Figure 5.3: Standard deviation of activation rate $\frac{dN_{act}}{dt}$

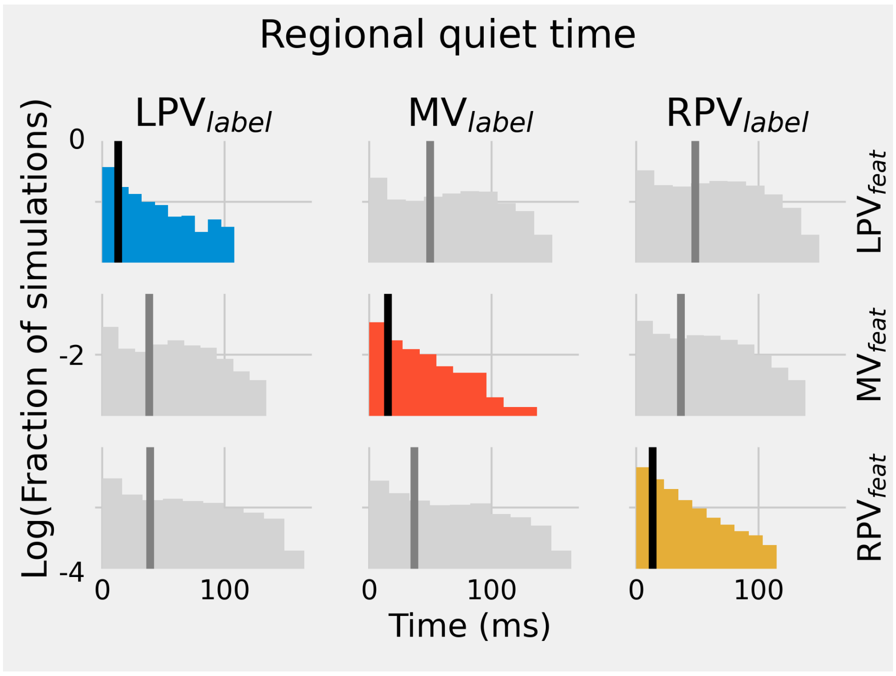


Figure 5.4: Regional quiet time per anatomical region

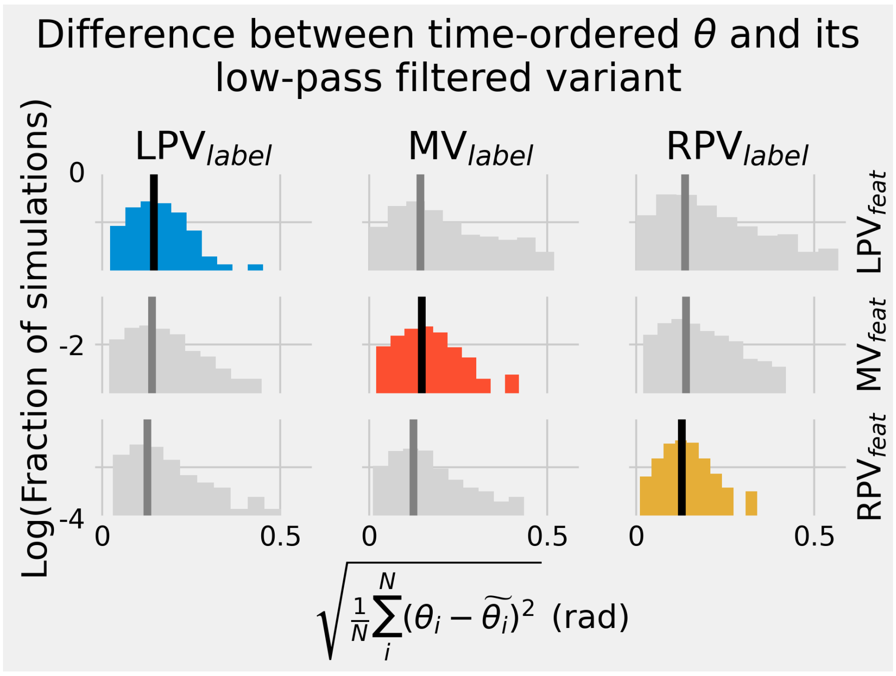


Figure 5.5: Error rate between the means of binned time-ordered theta coordinates and their low-pass filtered variant

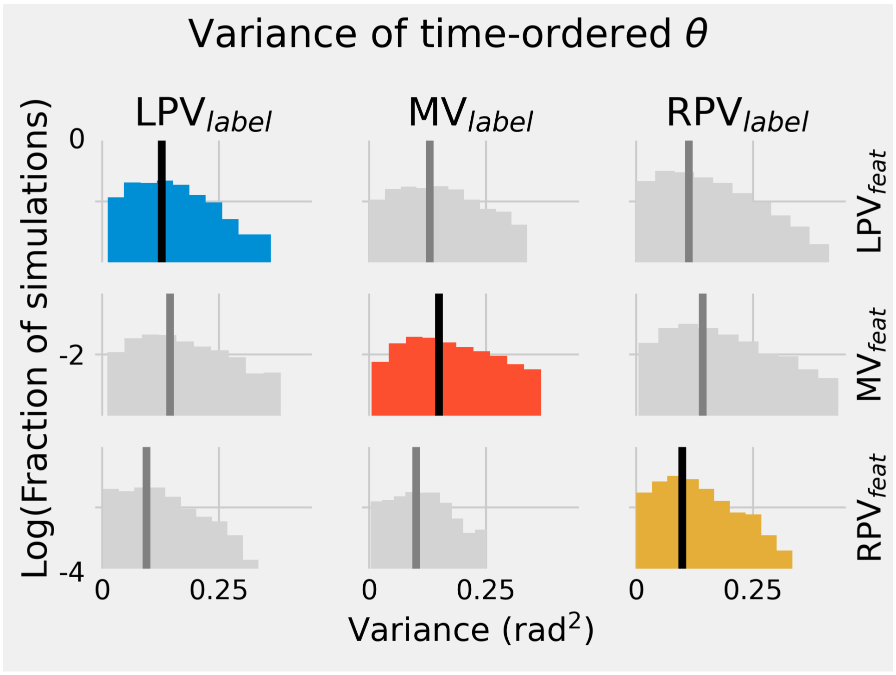


Figure 5.6: Variance of the means of binned time-ordered theta coordinates

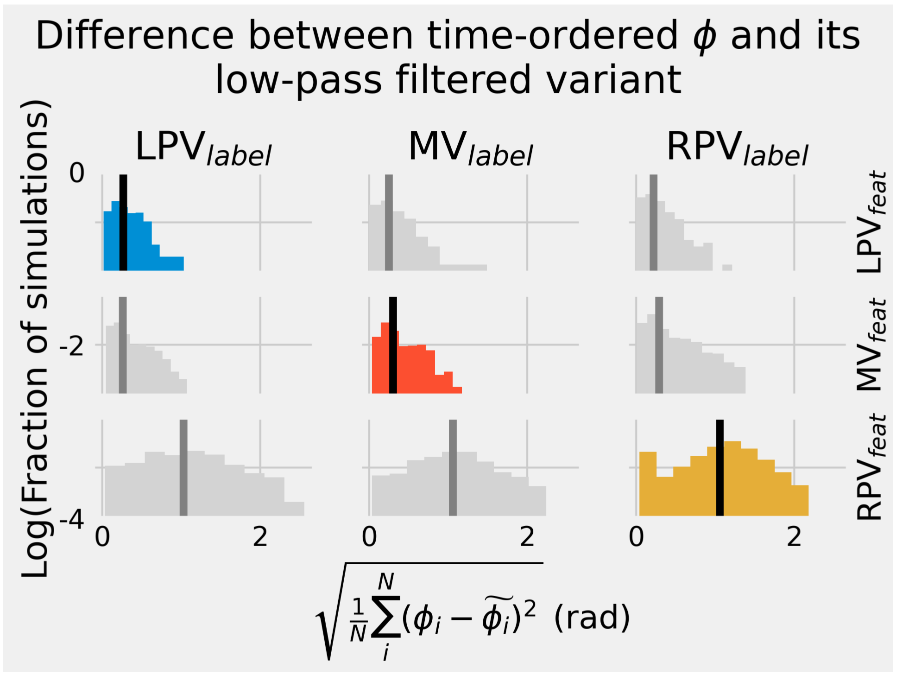


Figure 5.7: Error rate between the means of binned time-ordered phi coordinates and their low-pass filtered variant

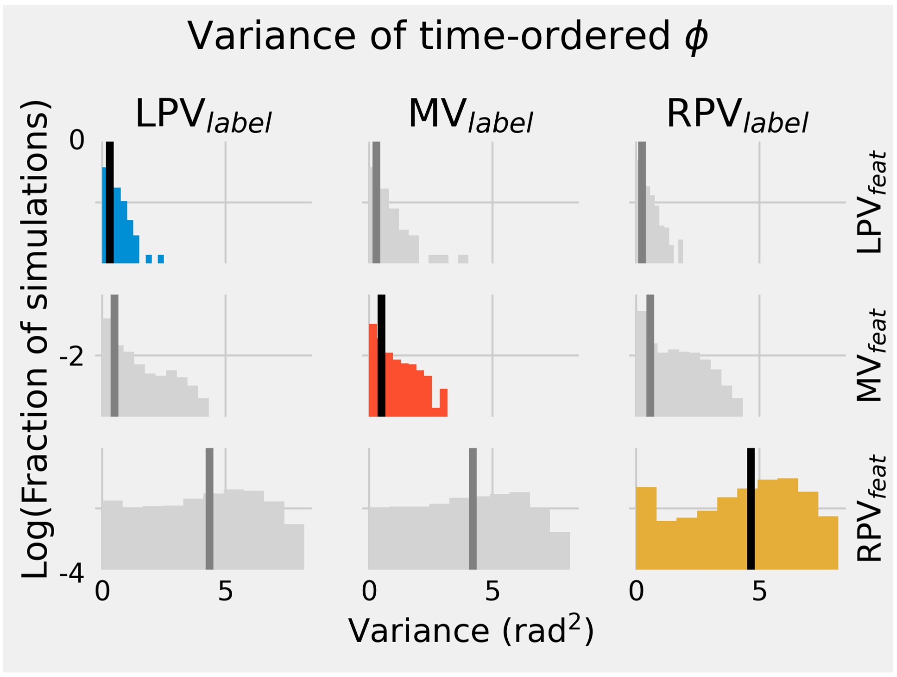
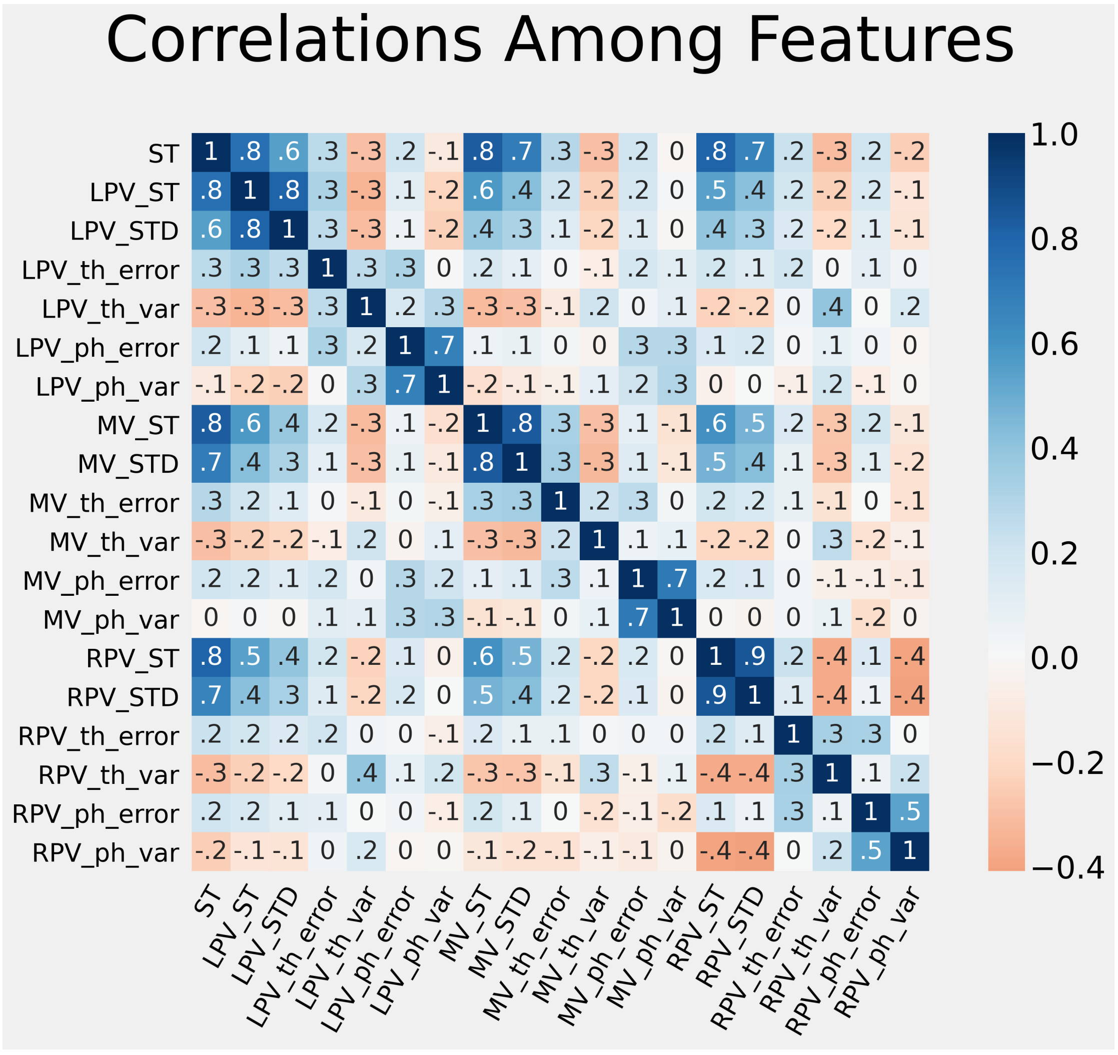


Figure 5.8: Error rate between the means of binned time-ordered phi coordinates and their low-pass filtered variant



	ST	LPV_ST	LPV_STD	LPV_th_error	LPV_th_var	LPV_ph_error	LPV_ph_var	MV_ST	MV_STD	MV_th_error	MV_th_var	MV_ph_error	MV_ph_var	RPV_ST	RPV_STD	RPV_th_error	RPV_th_var	RPV_ph_error	RPV_ph_var
ST	1	.8	.6	.3	-.3	.2	-.1	.8	.7	.3	-.3	.2	0	.8	.7	.2	-.3	.2	-.2
LPV_ST	.8	1	.8	.3	-.3	.1	-.2	.6	.4	.2	-.2	.2	0	.5	.4	.2	-.2	.2	-.1
LPV_STD	.6	.8	1	.3	-.3	.1	-.2	.4	.3	.1	-.2	.1	0	.4	.3	.2	-.2	.1	-.1
LPV_th_error	.3	.3	.3	1	.3	.3	0	.2	.1	0	-.1	.2	.1	.2	.1	.2	0	.1	0
LPV_th_var	-.3	-.3	-.3	.3	1	.2	.3	-.3	-.3	-.1	.2	0	.1	-.2	-.2	0	.4	0	.2
LPV_ph_error	.2	.1	.1	.3	.2	1	.7	.1	.1	0	0	.3	.3	.1	.2	0	.1	0	0
LPV_ph_var	-.1	-.2	-.2	0	.3	.7	1	-.2	-.1	-.1	.1	.2	.3	0	0	-.1	.2	-.1	0
MV_ST	.8	.6	.4	.2	-.3	.1	-.2	1	.8	.3	-.3	.1	-.1	.6	.5	.2	-.3	.2	-.1
MV_STD	.7	.4	.3	.1	-.3	.1	-.1	.8	1	.3	-.3	.1	-.1	.5	.4	.1	-.3	.1	-.2
MV_th_error	.3	.2	.1	0	-.1	0	-.1	.3	.3	1	.2	.3	0	.2	.2	.1	-.1	0	-.1
MV_th_var	-.3	-.2	-.2	-.1	.2	0	.1	-.3	-.3	.2	1	.1	.1	-.2	-.2	0	.3	-.2	-.1
MV_ph_error	.2	.2	.1	.2	0	.3	.2	.1	.1	.3	.1	1	.7	.2	.1	0	-.1	-.1	-.1
MV_ph_var	0	0	0	.1	.1	.3	.3	-.1	-.1	0	.1	.7	1	0	0	0	.1	-.2	0
RPV_ST	.8	.5	.4	.2	-.2	.1	0	.6	.5	.2	-.2	.2	0	1	.9	.2	-.4	.1	-.4
RPV_STD	.7	.4	.3	.1	-.2	.2	0	.5	.4	.2	-.2	.1	0	.9	1	.1	-.4	.1	-.4
RPV_th_error	.2	.2	.2	.2	0	0	-.1	.2	.1	.1	0	0	0	.2	.1	1	.3	.3	0
RPV_th_var	-.3	-.2	-.2	0	.4	.1	.2	-.3	-.3	-.1	.3	-.1	.1	-.4	-.4	.3	1	.1	.2
RPV_ph_error	.2	.2	.1	.1	0	0	-.1	.2	.1	0	-.2	-.1	-.2	.1	.1	.3	.1	1	.5
RPV_ph_var	-.2	-.1	-.1	0	.2	0	0	-.1	-.2	-.1	-.1	-.1	0	-.4	-.4	0	.2	.5	1

Figure 5.9: Correlations between features

5.4 Machine Learning: focal vs reentry classification

5.4.1 LATs as point features

Using the LAT points as features gave no satisfactory result. They will not be further discussed.

5.4.2 Engineered features

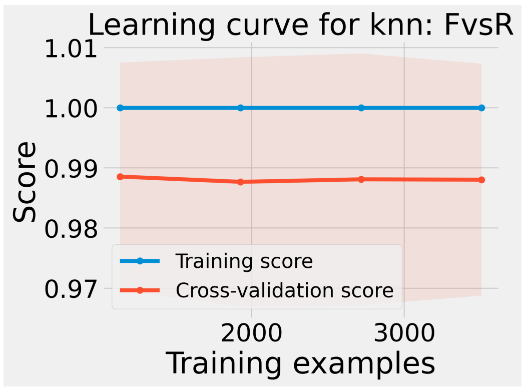
Table 5.1 shows us the ROC AUC scores, specificities and sensitivities of all the models. For the sake of reproducibility, the model parameters yielding these scores are given in the supplementary material in Table 7.2.

Each score is the mean of five out-of-fold predictions. The scores show that machine learning is very capable of distinguishing focal beats versus reentries. This is no surprise, as the silent time alone already provides this information, as is obvious in Figure 5.2. Only the model Complement Naive Bayes (CNB) has a mean ROC AUC score below 0.995. CNB also has a perfect sensitivity score, indicating that it can correctly classify all focals, but also classifies reentries as focals. Support Vector Machine (SVM) is the most successful model with a perfect score.

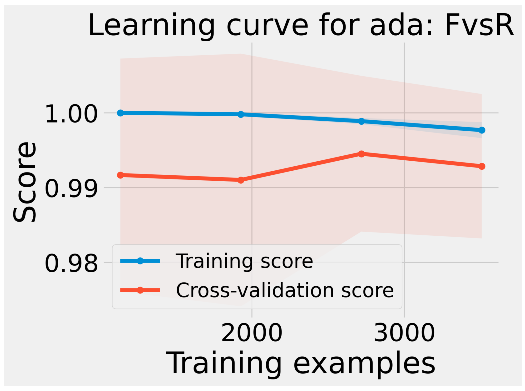
The learning curves in Figures 5.10-5.14 do not show a lot of valuable information for the successful models as the y-axis has a very small range. This implies that the models can find the difference between focal beats and reentries very early on, and adding more data does not impact the mean ROC AUC score a lot. Two learning curves should be discussed however. The models with a low score i.e. Multinomial Naive Bayes (MNB) and Complement Naive Bayes (CNB) are different. Both the training and cross-validation score of MNB in drop after about 2500 training samples, as can be seen in Figure 5.13a. This implies that the model fails to fit to more input samples than 2500, or that the parameters are not optimal for the given amount of samples. The training and cross-validation score of CNB seem to be on a rise in Figure 5.12b. This implies that either the model is still underfitted for the given amount of samples and will fit better given more input data, or that the parameters are not optimal for the given amount of samples.

Model	Mean ROC AUC	Mean specificity	Mean sensitivity
SVM:	**1.00000**	**1.00000**	0.98074
Random forest:	0.99929	0.99739	0.98906
Extra trees:	0.99929	0.99869	0.98818
AdaBoost:	0.99929	0.99695	0.98906
Gradient Boost:	0.99929	0.99913	0.98468
KNN:	0.99842	0.99696	0.97943
MNB:	0.99619	0.98929	0.99519
BNB:	0.99562	0.99095	0.98643
CNB:	0.78391	0.73702	**1.00000**
Model Mean:	0.97459	0.96738	0.98809

Table 5.1: Mean out-of fold cross-validation scores and average out-of-fold specificity & sensitivity for the label ‘focal’. Models are sorted by their mean ROC AUC score. Highest scores per column are shown in bold.

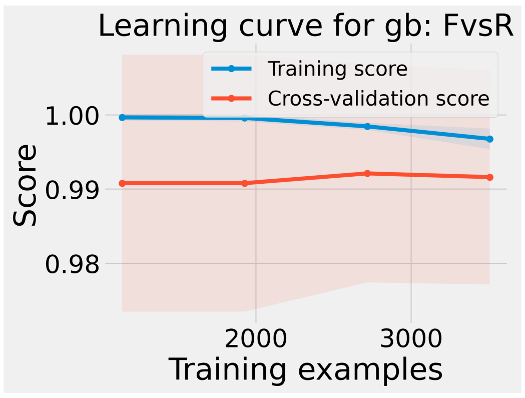


(a) Learning curve for KNN (focals vs reentries)

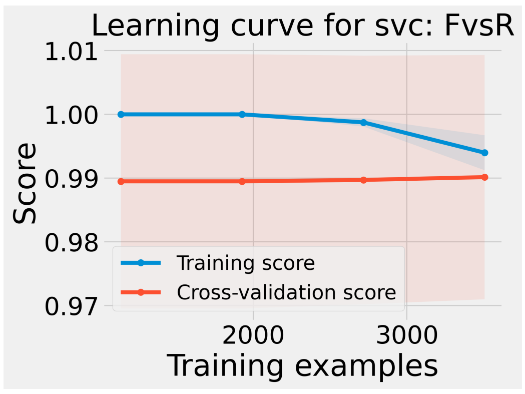


(b) Learning curve for AdaBoost (focals vs reentries)

Figure 5.10

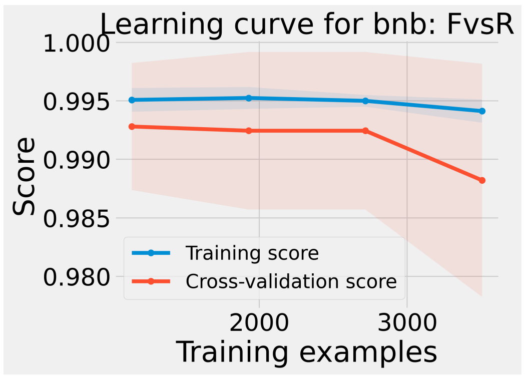


(a) Learning curve for Gradient Boost (focals vs reentries)

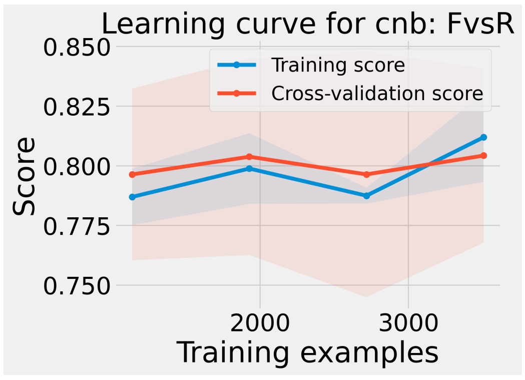


(b) Learning curve for Support Vector Classifier (focals vs reentries)

Figure 5.11

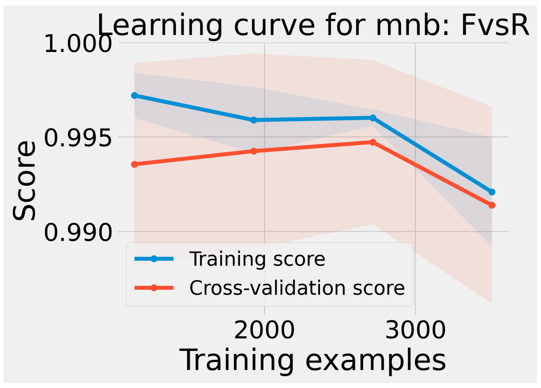


(a) Learning curve for Bernouilli Naive Bayes (focals vs reentries)

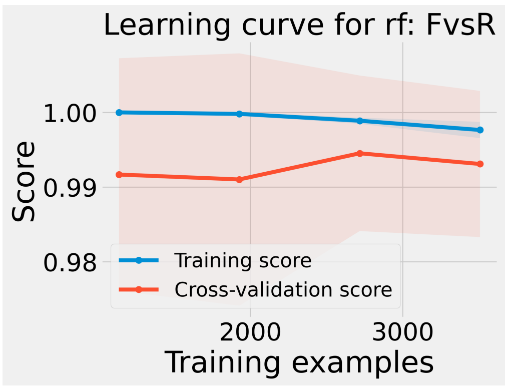


(b) Learning curve for Complement Naive Bayes (focals vs reentries)

Figure 5.12



(a) Learning curve for Multinomial Naive Bayes (focals vs reentries)



(b) Learning curve for Random Forest (focals vs reentries)

Figure 5.13

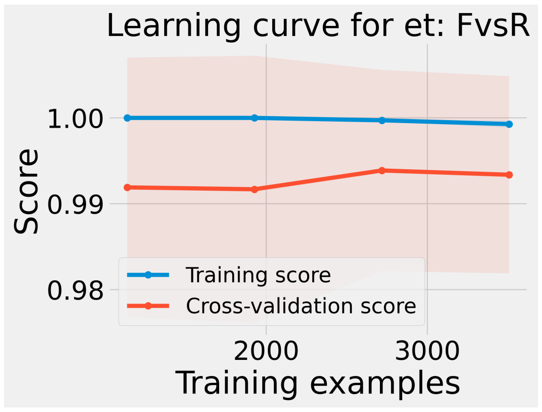


Figure 5.14: Learning curve for Extra Trees (focals vs reentries)

5.5 Machine Learning: reentry location classification

Table 5.2 shows the out-of-fold averaged ROC AUC scores of the models. The specificity and sensitivity have been averaged over the folds, as well as the labels. Means taken over the cross-validation folds will be denoted with a subscript ‘F’. Means taken over the labels will be denoted with a subscript ‘L’. An overview of the per-label sensitivity and specificity is given in Table 5.3. For the sake of reproducibility, the model parameters yielding these scores are given in the supplementary material in Table 7.3.

It is visible from both the ROC AUC scores and the sensitivities and specificities how machine learning is capable of inferring the location of reentries, albeit with lower accuracy than the focal vs reentry classification problem. The Complement Naive Bayes (CNB) classifier shows the best ROC AUC score at 0.7630. 7 out of 9 models have a ROC AUC score above 0.71. Extra Trees has the best sensitivity at 0.85560 and the second best specificity at 0.82002. Gradient boost has the best specificity at 0.84726 and second highest sensitivity at 0.8245. The specificity is for most models comparable to its sensitivity, with the exception of CNB, Multinomial Naive Bayes (MNB) and Bernouilli Naive Bayes (BNB). Similar scores for specificity and sensitivity imply that false classifications aren’t skewed to be predominantly positively or negatively assigned. CNB and MNB have a notable higher sensitivity than specificity: a difference of 0.13417 and 0.13018 respectively. This implies that these models have a lower false negative rate than a false positive rate. For BNB, the opposite holds true. Looking at Table 5.3, some specificities and sensitivities yield even higher scores, with a specificity of 0.91841 for the classification of MV reentries with Extra Trees, and a sensitivity of 0.96970 for the classification of RPV with Extra Trees. The mean specificities and sensitivities are dragged down by having lower scores for other labels. Looking at the low model-average sensitivity for the classification of MV reentries, it’s visible how the mitral valve label has a surpsisingly low sensitivity for most models, with values 0.35810 for Random Forest, 0.31895 for BNB and even 0.29694 for Support Vector Classifier. This implies that the features do not easily lend themselves to capturing mitral valve reentries for these models, yielding a lot of false negatives.

With the exception of the learning curves of Gradient Boost (Figure 5.16a), Multinomial Naive Bayes (Figure 5.18a) and Extra Trees (Figure 5.19), all learning curves have a strictly rising cross-validation score with the inclusion of more training data. This implies that all these models are still underfitted. It would be informative to fit these models to even more data to see how long the trends hold.

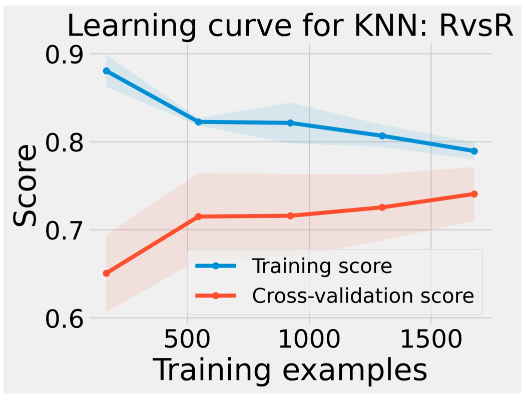
Model	Mean$_{F,L}$ ROC AUC	Mean$_{F,L}$ specificity	Mean$_{F,L}$ sensitivity
CNB:	**0.76300**	0.70729	0.84146
MNB:	0.76254	0.70806	0.83824
MLKNN	0.74438	0.74683	0.78704
Extra trees:	0.74284	0.82002	**0.85560**
SVM:	0.73884	0.77982	0.72024
Gradient Boost:	0.73552	**0.84726**	0.84254
AdaBoost:	0.71953	0.79352	0.78589
Random forest:	0.69608	0.74032	0.72791
BNB:	0.65798	0.73216	0.61261
Model Mean:	0.72897	0.76392	0.77906

Table 5.2: Mean out-of fold cross-validation scores for the reentry multilabel classification problem and average out-of-fold specificity & sensitivity, averaged again over the labels. Models are sorted by their mean ROC AUC score. Highest scores per column are shown in bold.

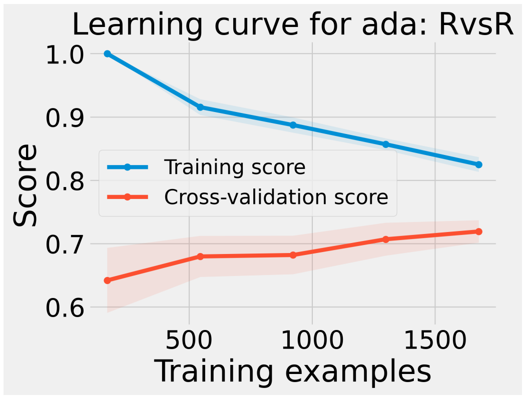
Extra Trees, Gradient Boost and Bernouilli Naive Bayes seem to be overfitted, but it is not fully clear and they too could use more data to gain a better view over the trend of the cross-validation score.

Model	$\text{Mean}_{F,L}$ ROC AUC	Mean_F specificity LPV	RPV	MV	Mean_F sensitivity LPV	RPV	MV
CNB:	**0.76300**	0.78110	**0.74335**	0.59788	0.86374	0.88822	**0.77384**
MNB:	0.76254	0.78618	0.74239	0.60188	0.86259	0.89226	0.76701
MLKNN:	0.74438	0.76272	0.67506	0.79198	0.86607	0.96768	0.54500
Extra trees:	0.74284	0.86430	0.69984	**0.91841**	0.88685	**0.97912**	0.69758
SVM:	0.73884	0.76603	0.68077	0.89725	**0.89259**	0.97643	0.29694
Gradient Boost:	0.73552	**0.89938**	0.73776	0.89309	0.88914	0.96566	0.71520
AdaBoost:	0.71953	0.84357	0.67965	0.85698	0.84295	0.96162	0.55589
Random forest:	0.69608	0.74258	0.57778	0.87478	0.85329	0.96970	0.35810
BNB:	0.65798	0.87158	0.51199	0.83526	0.60167	0.89360	0.31895
Model Mean:	0.73857	0.82832	0.72095	0.83411	0.89222	0.96251	0.62564

Table 5.3: Average out-of fold cross-validation scores for the reentry multilabel classification problem and average out-of-fold specificity & sensitivity for each label. Highest scores per column are shown in bold.

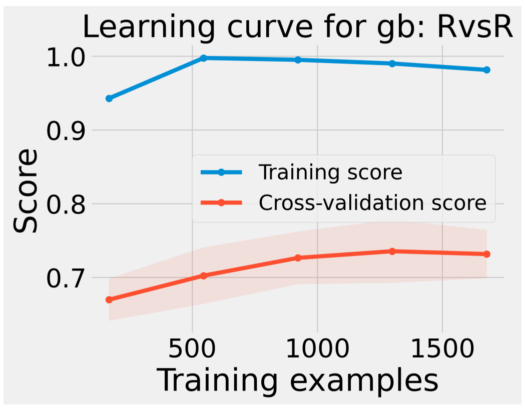


(a) Learning curve for KNN (reentry location)

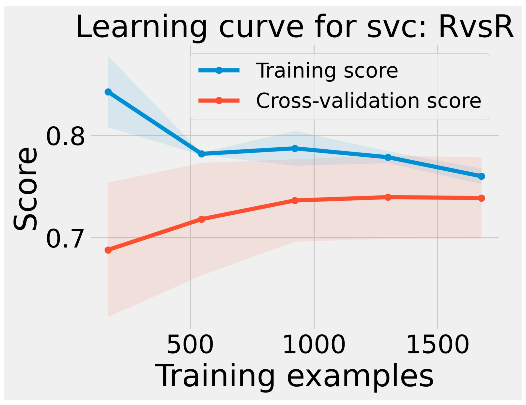


(b) Learning curve for AdaBoost (reentry location)

Figure 5.15

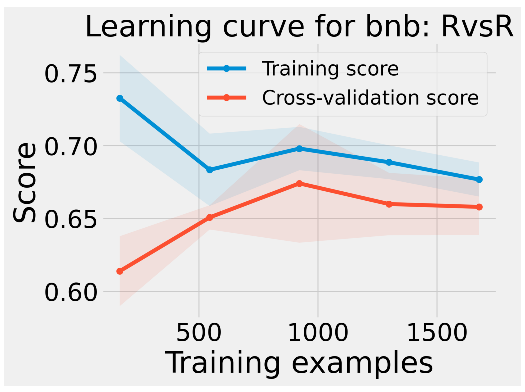


(a) Learning curve for Gradient Boost (reentry location)

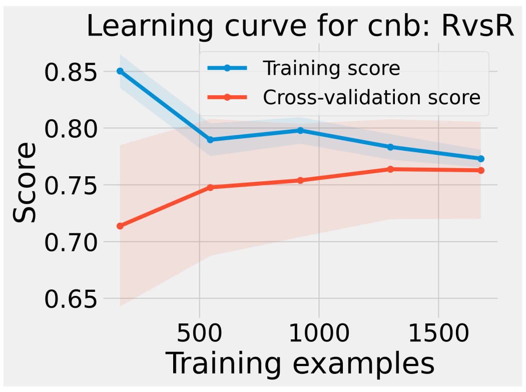


(b) Learning curve for Support Vector Classifier (reentry location)

Figure 5.16

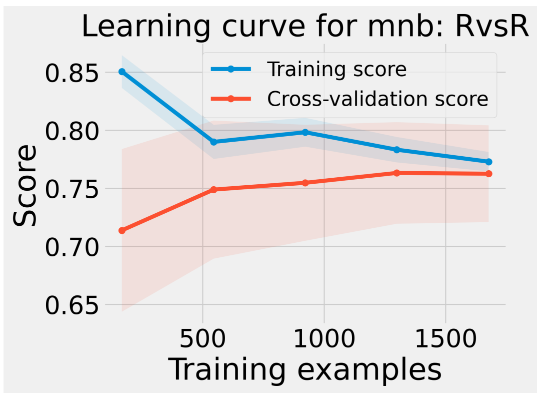


(a) Learning curve for Bernouilli Naive Bayes (reentry location)

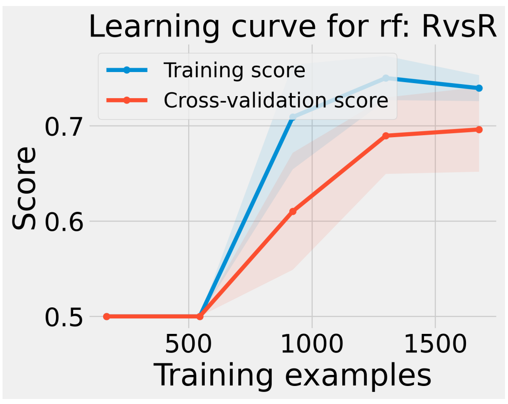


(b) Learning curve for Complement Naive Bayes (reentry location)

Figure 5.17



(a) Learning curve for Multinomial Naive Bayes (reentry location)



(b) Learning curve for Random Forest (reentry location)

Figure 5.18

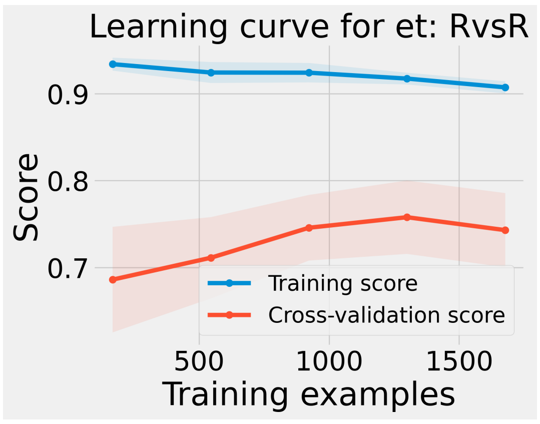


Figure 5.19: Learning curve for Extra Trees (reentry location)

With eight models having ROC AUC scores between 0.995 and 1.0, supervised machine learning is very successful in distinguishing focal beats from reentry, indicating that cardiac mapping data can lend itself to machine learning analysis. It is moderately successful in inferring the location of reentry, with seven models having mean out-of-fold ROC AUC scores between 0.71 and 0.76. The latter classification problem needs more samples in order to properly fit the models to the data.

6 Discussion

6.1 Mesh reconstruction

The mesh reconstruction pipeline as discussed in Section 4.1 can reconstruct a CARTO `.mesh` into a qualitative simulatable mesh with control over edge lengths and conduction regions.

This pipeline can be useful for any electrophysiological simulations on CARTO data. The pipeline can be expanded to other data types as well, as long as they contain point and triangle information. The automatic assignment of conduction regions and tagged physical regions to the output mesh only works for CARTO `.mesh` files and will not be possible for other data types, but this can be implemented if necessary.

6.1.1 Limitations

The pipeline is sensitive for input meshes that have defects. Some defects are automatically fixed throughout the pipeline, but most are not. Whether or not the mesh survives the reconstruction process depends on the refinement parameters as well as these mesh defects. Sometimes meshes would fail the reconstruction process, but succeed with slightly different allowed edge length or iteration step parameters. Out of 38 input meshes, 13 failed. Of these thirteen, about four could be rerun with other parameters and succeed. The other 9 failed due to mesh defects or other unknown reasons.

Since CARTO data carries no information about fiber directions, this was not implemented in the pipeline. However, the inclusion of fiber directions is important for realistic simulations [39]. This is a major shortcoming.

The availability of atrium models remains sparse. 25 atrium meshes is not enough to fully encapsulate the variance of atrium shapes that are present in the population.

6.1.2 Further development

The mesh reconstruction pipeline could use a mesh defect fixing step for a more reliable reconstruction process, as well as support for other datatypes. Preferably datatypes with fiber direction information. Fiber directions can also be inferred after the reconstruction process based on histology maps, rule-based methods or methods that use morphological information of the atrium as well as a local solution of Laplace's equations [40, 41, 42].

It would be beneficial to include more atrium meshes. Tweaking the shape of the current meshes is also an option. To this end, one could make variations in shape of a given atrium mesh by varying spatial operations such as stretching along some axis.

6.2 Machine Learning

Machine learning is very capable of discerning focal beats from reentry, as is visible in the high scores in Table 5.1. Complement Naive Bayes has the lowest ROC AUCscore for this classification problem at 0.78391. The other 8 models have a near perfect ROC AUC score.

It can also distinguish between different locations of anatomical reentry with moderate succes, as is reflected by the scores in Table 5.2. 6 out of 9 models have a ROC AUC score above 0.73. Gradient Boost and Extra Trees provide the highest specificities and sensitivities pairs at $(0.84726, 0.84254)$ and $(0.82002, 0.85560)$ respectively.

The machine learning methods as discussed in this work provide the initial steps towards a new field of analytical applications, more closely correlated to the wave dynamics and anatomical properties of the heart. An analysis of this kind has not been done before and can combine existing state of the art machine learning analysis techniques to existing datasets, possibly providing new and insightful information. It also highlights the capabilities of DGM [14] for generating ground truth labels and conduction velocities, making a supervised machine learning analysis of this scale possible.

6.2.1 Limitations

The ground truth is generated by DGM. The accuracy of a machine learning model based on this ground truth is a correspondence rate to DGM, and can only ever get as accurate as DGM.

Only 2097 simulations of reentries could be used for machine learning analysis. This is not enough for most models to fully fit to the data given the feature set. With the exception of Gradient Boost, Multinomial Naive Bayes and Extra Trees, all models are underfitted. More data needs to be generated to fully explore the capabilities of these models.

During feature engineering process, the scope was limited to designing features with a presupposed direct link to the underlying dynamics of reentries, such as regional quiet time, mean angular deviation from the low-pass filtered angles etc. This does not truly exploit the full capabilities and strengths of machine learning. It would be beneficial to add much more features that do not necessarily contain obvious information about the underlying dynamics. One of the strengths of machine learning is to find connections where we can not. It is thus not necessary to limit the features to those with presupposed obvious links to underlying dynamics.

All features are dependent on a certain time resolution or bin width. The chosen values might not be optimal. They might not even be sufficient in some cases. The time resolutions for these features were educated guesses, but unexplored ones. This is also applicable to the choice of the amount of points in the bands around anatomical regions.

All features referring to the location of activity, as discussed in Section 4.3.3, were represented in angular coordinates (θ, ϕ). As some features are dependent on distances (the error rate with the low-pass filtered coordinates, see Figure 4.9) or spatial point density (the variance of the coordinates, see Figure 5.6 and 5.8), these distances and densities would have to be scaled with the spherical Jacobian to give accurate values. This was not implemented. In hindsight, the spherical coordinate representation was unnecessary, save for the fact that this makes it possible to make 3D plots of (θ, ϕ, t) space like Figure 4.10. All these coordinates and corresponding features can be perfectly well represented in (x, y, z) space. In addition to providing two extra regional features (the error rate with the low-pass filtered time-ordered coordinate and the variance of this time-ordered coordinate), there would also be no need to scale these with a Jacobian.

The error rate between the time-ordered spherical coordinates and their low-pass filtered variants might not be the best way to represent a wavefront passing on both sides of an anatomical object. Averaging out the angles that are measured in the same time interval loses a lot of valuable information. A better way might be to not average these coordinates at all, but simply time-order them as is, and count the amount of zero crossings.

The regional quiet time is calculated as the ratio of empty time intervals around some anatomical object to the cycle length. A better implementation that better represents the amount of time without activation around an anatomical object, and one that is less susceptible to the choice of an appropriate time resolution, is to find the longest amount of time some anatomical object's neighbouring myocardium goes without activation. For regions with sustained activity, this will be the time between the two slowest consequent LATs. For regions without sustained activation, this can be much larger than this slowest consequent LAT interval if the depolarising wave is at some point not propagating in the neighbouring myocardium.

Each simulation of focal beats was run with a stimulus pacing interval of exactly 240 ms. This might skew the accuracy of the machine learning to seem more accurate than it really is. It might be beneficial to rerun the simulations of focal beats with varying pacing intervals and to analyse these instead.

The LAT datapoints taken from the simulations were subsampled to 2000 points. This sampling was done homogeneously. This does not reflect the case of a clinical setting, where there can be significant variance in the point density.

The machine learning was only done on simulations. Even with the implementation of varying conduction velocities on the simulation mesh and usage of precise ionic models, these simulations are and idealised case of the clinical setting and lack a lot of noise that in-vivo data has.

6.2.2 Further development

More data needs to be generated. On the one hand to see if the success for discerning focals from reentries holds true for more data, and on the other hand to fully explore the capabilities of the models inferring the reentry location. As each simulation was subsampled to 2000 points, and obvious way to increase the amount of input data is to sample the same simulation multiple times, each time with different points. This will represent the same simulation in multiple ways, effectively expanding the same data. However, this method will not add to a variation in conduction regions, nor the atrium anatomy, which is an important influence on the susceptibility of the atrium to atrial fibrillation [43].

The current features can be developed further to more accurately convey the intended information. For example, using the amount of zero crossings in the time-ordered mean angular coordinates (Figure 4.9) might better convey the fact that a wavefront passes an anatomical object on both sides better than the currently implemented error rate. Other features are often dependent on a certain time resolution or bin width. In addition to polishing the existing features, more features altogether should be beneficial to the machine learning.

More research should be done on finding appropriate time resolutions for these features, as well as the influence of the size of the myocardium used for feature engineering, as shown in Figure 4.7, on the machine learning results.

It would be interesting to see the effect of heterogeneous sampling of simulation data on the machine learning accuracy, as well as the addition of noise to the simulation data.

7 Supplementary material

7.1 Mesh reconstruction pipeline

7.1.1 CARTO `.mesh` to `.csv`

The first step in the mesh preparation sequence consists of extracting the point and triangle information from the provided `.mesh` files. These contain the coordinates of the points and the point indices that define each triangle respectively. Both are written out to `.csv` files, as well as all other sections of the `.mesh` file. Some `.mesh` files provide extra information regarding the anatomy of the left atrium, such region tags to define left and right pulmonary veins as well as the mitral valve. This info can later on be used to automatically detect and assign non-conductive regions, but will not be kept during the tetrahedralisation process. Other info defined in the `.mesh` files will be ignored.

7.1.2 Vertices and triangles `.csv` to double layered surface mesh

The second step towards adding thickness to the surface mesh is to calculate two bounding layers with a certain distance inbetween (i.e. the epi- and endocardium) based on the provided surface mesh. The original input surface mesh will be discarded and only the two bounding layers will be saved.

There are multiple ways one can go about this calculation, such as spherical expansion and Poisson surface reconstruction. Reconciling accuracy and computational ease, the method used in this thesis is as follows: for each point in the mesh, all triangles including that point were extracted. The normals of these triangles were information already present in the CARTO .mesh file, but can also be easily calculated. Along the direction of the average of these normals, a point was added .4 mm to the outside of the mesh and .1 mm to the inside. The sum of these two values at .5 mm will from now on be referred to as the “thickness”

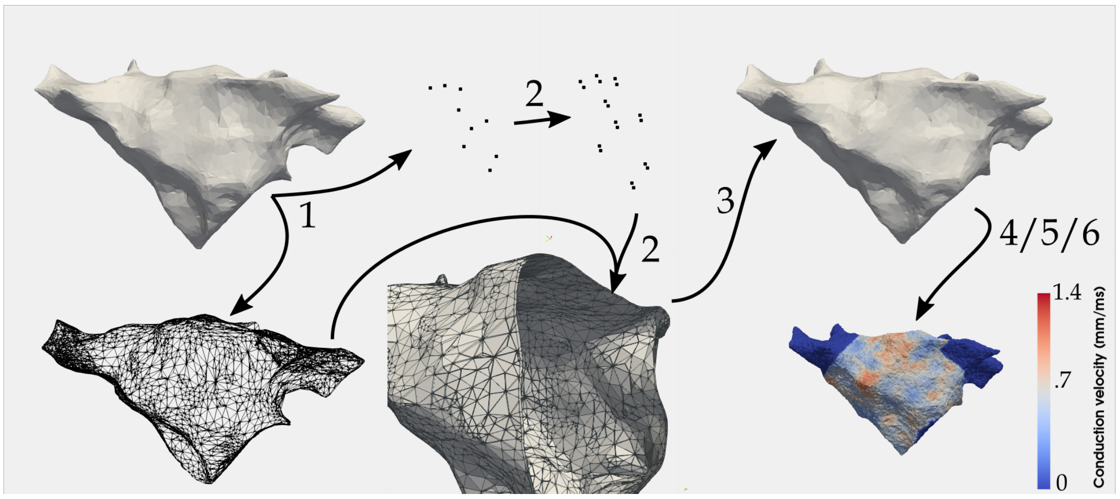


Figure 7.1: A flowchart of the mesh reconstruction process. The arrows are numbered in correspondence to the subsections of Section 4.1.

of the mesh. This thickness is significantly thinner than an actual atrium, which ranges from 2.5 *mm* to 6.5 *mm* [44]. The thickness was kept thin to avoid transmural conduction [35] to play a significant role in the wave propagation dynamics and inducing endo- epicardial dissociation of electrical activity, which would add unnecessary extra complexity [36]. This way, the thin mesh essentially functions as a two-dimensional surface for wave propagation.

As long as the points do not cross over each other during this mesh thickening, the triangle information of the original mesh can simply be recycled and applied to this new point cloud mesh. It should be noted that this method can go wrong if the length of the curvature radius in any location of the mesh is comparable to the thickness of the mesh. As such, points may cross over one another when expanding in a concave region and a self-intersection is created, as can be seen in Figure 7.2. Consequently, the triangle info of the initial mesh becomes inapplicable to the newly created, self-intersecting mesh. This can be avoided by using qualitative input meshes, choosing a thinner thickness, or using mesh cleaning software as a postprocessing tool to fix any holes, self-intersections or non-manifoldness that might have made their way into the mesh, such as PyVista's [45] wrapper for MeshFix [46], MeshLab [47] or any apt meshing software one might be fond of.

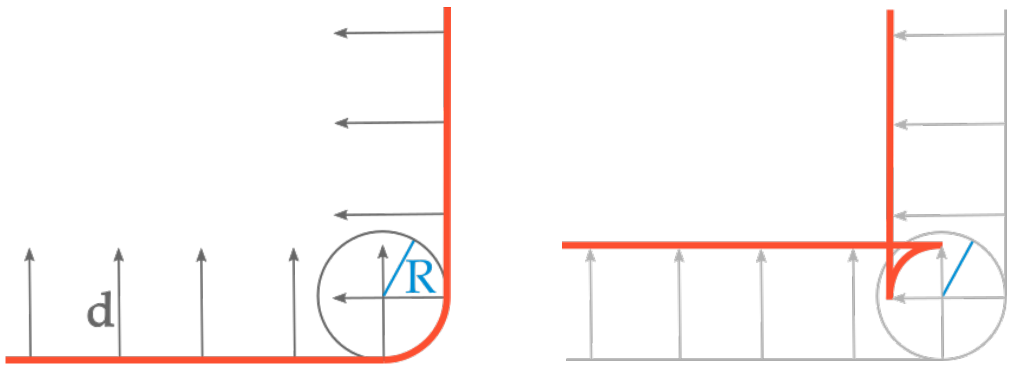


Figure 7.2: A second mesh layer (right, red) is added to a pre-existing mesh (left, red), separated by a distance d (thickness), where $d > R$ and R is the curvature radius (blue) of a concave corner in the mesh. This creates a self-intersection.

7.1.3 Refinement

The third step involves refining the mesh to a desired coarseness. The mesh now consists of two triangular surface meshes separated by a distance d. These will further on be used to define boundary surfaces of tetrahedrons. As such, they have to be refined before the actual tetrahedralisation takes place. The Python-integrated C++ library PyMesh combines already existing and state-of-the-art meshing libraries, and as such provides two useful functions for this: `split_long_edges()` and `collapse_short_edges()`. Calling these two functions iteratively in alternating fashion, each time with a stricter constraint on the minimum and maximum allowed edgelength, makes the edge lengths converge to a single value. This makes the mesh more homogeneous with respect to edge length. This procedure also improves the quality of the mesh, which becomes especially important when tetrahedralizing the mesh and propagating a stimulus on the final result. The maximum and minimum allowed edgelengths are called α and tolerance respectively. They follow the iteration step dependent distribution:

$$\begin{aligned} \alpha &= \left(1 - \frac{n}{N}\right) \cdot max(original\ edges) \cdot e^{-3n/N} + UL \\ tolerance &= 500\mu m \cdot \left(1 - \frac{n}{N}\right) + LL \end{aligned} \tag{7.1}$$

, where UL and LL are the desired upper limit and lower limit respectively on the mesh edge lengths, n is the current iteration step and N is the total amount of iteration steps.. This procedure should be continued at least until the shortest edge length is larger than the resolution of the simulation software and the longest edge length is shorter than the maximum allowed resolution of the simulation software. For the meshes used in this thesis, an edge range of $700 - 1100m$ was chosen.

Switch	Value	Meaning
-p		Tetrahedralizes input surface mesh
-Y		Preserves input faces as boundaries
-k		Outputs mesh to .vtk file
-N		Suppresses creation of .node file
-E		Suppresses creation of .ele file
-F		Suppresses creation of .face file
-q	2.5/20	minRadiusEdgeRatio/minDihAngle - Refines mesh
-a	2e+08	Volume constraint in µm^3
		Optional:
-O	0-10/0-7/0-∞	Optimization level:
		flip level/local optimisation/max iterations
-V		Verbose output

Table 7.1: TetGen switches

This proved to be a small enough resolution for accurate wavefront propagation while being coarse enough to save on valuable CPU time, as was tested by running simulations. During this process, one can try to connect the two meshes in places where they are not closed. This will be necessary for meshes where e.g. the mitral valve is an open hole, or where the mesh has holes that were not fixed by any mesh fixing software. If the mesh is not closed, the upcoming tetrahedralization process will unavoidably fail.

7.1.4 Tetrahedralisation

The fourth step and the essence of the process is the actual tetrahedralization of the mesh. At this point, the pipeline has provided us two refined triangular meshes that act as epi- and endocardium surface layer, but the cells are still triangles and the inbetween space of the two layers is empty. In order to tetrahedralize the mesh, Hang Si's TetGen [34] library was used. To this end, a `.smesh` file was written, which allows for the definition of holes, i.e. the entire hollow inside of the atrium. Only with the use of `.smesh` files can such a hole be defined. The Python wrappers for TetGen provided by either PyVista, PyMesh or PyTetGen do not provide this feature.

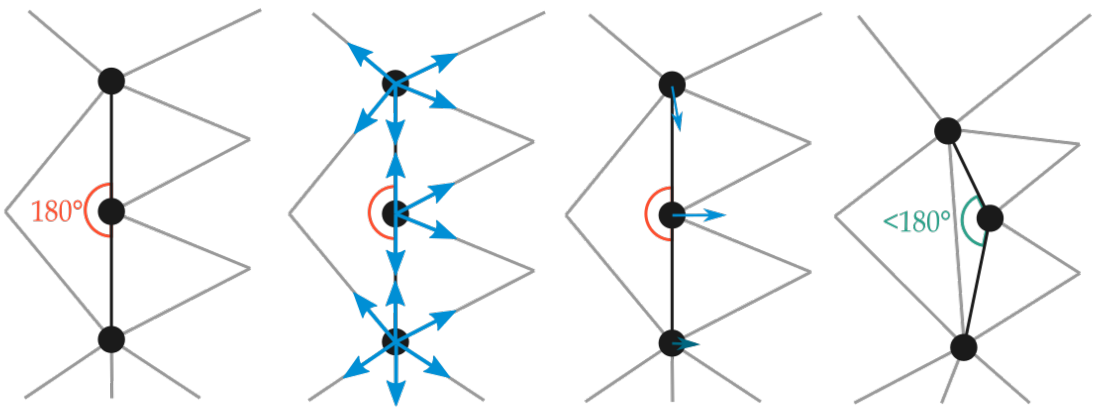


Figure 7.3: Schematic of fixing a collinearity. From left to right: the collinear edgepair, the connections of the points that define the collinear edgepair, the average directions of these connections, the mesh with the collinear points shifted along these average directions.

TetGen, when used as a bash command, uses so-called switches. These act as parameters for the tetrahedralisation process, defining constraints on tetrahedron quality, tetrahedron volume, refinement etc. Reading in a `.smesh` file and creating a tetrahedron mesh with quality and volume constraints gives rise to the following bash command

```
tetgen -pYkNEFq1.5/20 -a2.0e+08 <meshname>.smesh        (7.2)
```

The switches used in the command, along with their value and explanation are shown in table 7.1, with some optional switches as well. The `.smesh` file format is explained in the TetGen documentation, as well as the usage of command-line switches. This command gives a 3D tetrahedral mesh as output in the form of a `.vtk` file.

During this process, some errors can be brought to the attention of the user by TetGen. The most common error for this process is:

```
Collinearity found!
Cannot further reduce collinear tolerance (179.9 degrees).
Aborting.
```

This means that the mesh contains three points whose connecting lines have a smallest angle of over 179.9 degrees. As such, TetGen cannot build a tetrahedron on the triangular surface that contains these two lines. In order to avoid this, the error output of the command was caught. This error output contains the indices of the points defining the collinear line segments. The collinearity can be fixed by shifting the points away from each other. This has to be done in the right direction, as shifting them the wrong direction will intersect the edge connecting the outer two vertices, giving rise to an intersection error. Getting all the connections for each of these points, calculating the average direction along these connections and moving the point along this average direction will always move the points away from each other without creating a self-intersection, as long as the points are shifted over a distance that's smaller than the distance to its closest neighbour. This can easily be seen in the schematic Figure 7.3. The only time when this method fails is when all three average directions are exactly equal within machine precision; a negligible situation.

Other errors were not fixed. These include self-intersections, overlapping edge pairs and overlapping vertices. Changing the desired edge length interval or the amount of conversion steps in the refinement phase usually fixed these errors.

7.1.5 Conversion

The fifth step towards a simulatable mesh is to convert this `.vtk` file to something openCARP can work with. To this end, Meshtool was used to convert the meshes to `carp_txt` format.

7.1.6 Cleaning

As a sixth step, meshtool is again used to clean the mesh. This cleaning algorithm involves shifting vertices in a non self-intersecting way such that the quality of the corresponding tetrahedra improves. Defining the quality of a tetrahedron can be done in multiple ways. The metric used here is the volume metric, which aims to maximise the ratio Q between the tetrahedron's volume and the volume it would have if all the edges were as long as the average edge length:

$$Q = \frac{6\sqrt{2}V}{\overline{a}^3} \quad , \ Q \, \epsilon \, [0, 1] \tag{7.3}$$

, where $\overline{a}$ is the average edge length of the tetrahedron. The closer this ratio Q is to 1, the more qualitative the tetrahedron is.

This cleaned mesh is then converted back to `.vtk` format for further steps. During this process, a second `.vtk` file is also written out, one where the point index is added as scalar data to the mesh points. This is because Paraview was used to select anatomical regions, stimuli and blocks on the mesh (as was discussed in Section 4.2.3) and Paraview does not keep track of the original point index during certain operations.

7.1.7 Applying conduction velocities and scar tissue

DGM [14] was used to calculate an estimated conduction velocity for each node that registered an LAT during the in-vivo cardiac mapping of the left atrium. PyVista was then used to interpolate this data on the refined tetrahedral VTK mesh and to convert the point data to cell data.

Since the square of this conduction velocity is proportional to the conductivity in each cell, the squared value of this cell data can be used to define a per-cell mesh conductivity scaling factor. OpenCARP uses this in the form of an additional `.txt` file. Each line of this file contains a scale factor between 0 and 1. The lines indices correspond to the indexing of the mesh cells. OpenCARP can scale the conductivity during the simulation making use of this file.

Similarly to scaling the conduction velocity, one can also select a region and scale the conduction velocity to zero. As far as propagation dynamics go, this effectively acts as scar tissue. If wanted, it's even possible to write a small procedure that randomly selects one or more nodes, get the neighbouring nodes and scaling the conduction velocity of these nodes down to zero. This way, one can generate a selection of meshes with random scar tissues of random sizes.

It should be noted that scaling down the conduction velocity is not truly the same as creating scar tissue. In order to properly create a scar tissue for OpenCARP simulations, one needs a so-called adjustment file that describes whether or not some cell has functioning ionic channels for a certain species of ions. This can be implemented in a similar fashion as the previously mentioned scale factor file. For this thesis however, no scar tissue was defined on the meshes.

7.1.8 Conduction velocity variation

In addition to a reconstruction of the conduction velocities based on the LATs, nine more variations were made. Variations were made by iterating over all the points of the mesh, fetching the k closest neighbours, fitting a normal distribution to the CV distribution of these neighbours and drawing a random new conduction

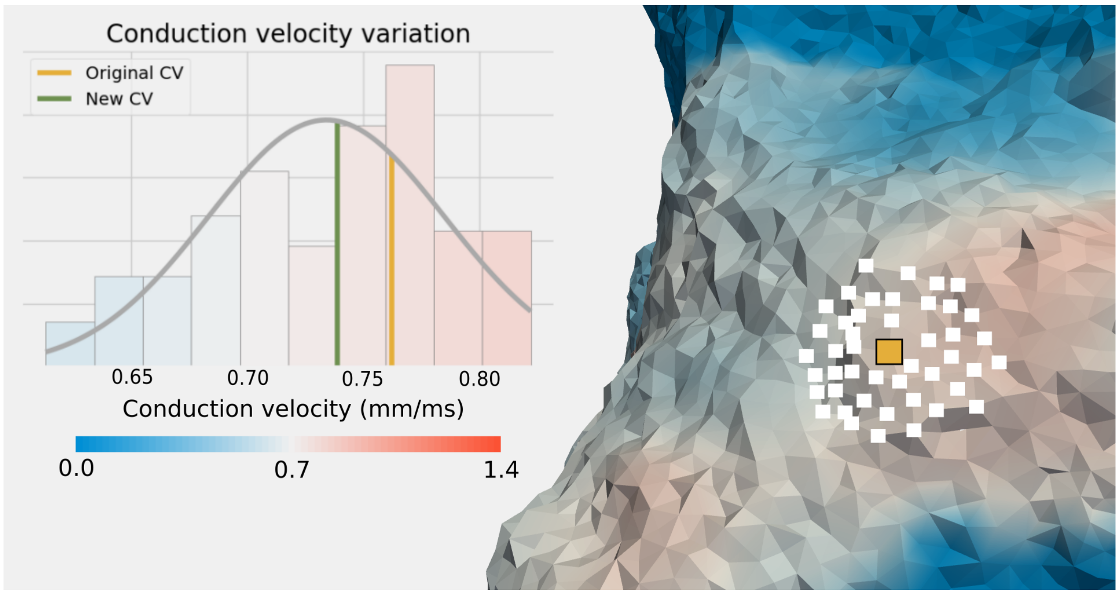


Figure 7.4: Example of making variations of a existing conduction velocity distribution. For each point (yellow), k neighbors (white, here $k = 100$, not all visible) are used to create a local conduction velocity distribution. A normal distribution (grey) is fitted to this local CV distribution and a new CV (green) is randomly drawn from this distribution.

velocity from this distribution. These steps are shown in Figure 7.4. These new conduction velocities were again interpolated on the mesh for a new conduction velocity distribution. This method prevents extreme outliers due to the interpolation process and keeps large regions of comparable conduction velocities intact due to the Gaussian sampling distribution. The similarity of CV distribution this method provides is dependent on the amount of neighbours k and the interpolation smoothing parameters, as well as some randomness.

The `carp_txt` mesh files and corresponding scale factor `.txt` files together define a reconstructed (variation of a) patient-specific simulatable atrium model.

7.2 Machine learning

7.2.1 Focal vs reentry parameters

Model	ROC AUC	Parameters
SVC:	1.0	{’C’: 0.5, ’gamma’: ’scale’, ’kernel’: ’rbf’}
KNN:	0.9994	{‘algorithm’: ‘ball_tree’, ‘leaf_size’: 1, ‘n_neighbors’: 1, ‘p’: 2, ‘weights’: ‘uniform’}
Random forest:	0.9993	{’criterion’: ’gini’, ’max_depth’: 5, ’max_features’: None, ’min_samples_leaf’: 40, ’n_estimators’: 20, ’n_jobs’: -1, ’warm_start’: True}
Extra trees:	0.9993	{’criterion’: ’gini’, ’max_depth’: 5, ’max_features’: None, ’min_samples_leaf’: 1, ’n_estimators’: 70, ’n_jobs’: -1, ’warm_start’: True}
AdaBoost:	0.9993	{’learning_rate’: 0.0005, ’n_estimators’: 100}
Gradient Boost:	0.9993	{’learning_rate’: 0.005, ’loss’: ’exponential’, ’max_depth’: 1, ’max_features’: None, ’min_samples_leaf’: 1, ’n_estimators’: 80, ’subsample’: 0.02, ’warm_start’: True}
BNB	0.9956	{’alpha’: 0.5, ’binarize’: 0}
CNB:	0.7839	{’alpha’: 0.01, ’norm’: 1}
MNB:	0.5526	{’alpha’: 0.001}

Table 7.2: Parameters used for each machine learning model for the focal vs reentry classification problem, resulting in the corresponding ROC AUC score.

7.2.2 Reentry location parameters and scores

Model	ROC AUC	Parameters
MLKNN	0.7494	{‘k’: 65, ‘s’: 0}
CNB:	0.7473	{‘alpha’: 0.005, ‘norm’: 0}
Extra Trees:	0.7428	{‘criterion’: ‘gini’, ‘max_depth’: 20, ‘max_features’: None, ‘min_samples_leaf’: 7, ‘n_estimators’: 200, ‘n_jobs’: -1, ‘warm_start’: True}
SVC:	0.7427	{‘C’: 0.2, ‘gamma’: ‘scale’, ‘kernel’: ‘rbf’}
Gradient Boost:	0.7372	{‘learning_rate’: 0.07, ‘loss’: ‘deviance’, ‘max_depth’: 5, ‘max_features’: None, ‘min_samples_leaf’: 50, ‘n_estimators’: 200, ‘subsample’: 0.9, ‘warm_start’: True}
AdaBoost:	0.7195	{‘learning_rate’: 0.05, ‘n_estimators’: 500}
Random Forest:	0.6961	{‘criterion’: ‘entropy’, ‘max_depth’: 2, ‘max_features’: None, ‘min_samples_leaf’: 200, ‘n_estimators’: 50, ‘n_jobs’: -1, ‘warm_start’: True}
BNB:	0.6580	{‘alpha’: 0.01, ‘binarize’: 0}
MNB:	0.5526	{‘alpha’: 0.001}

Table 7.3: Parameters used in each machine learning model for the multiblabel reentry location classification problem, resulting in the corresponding ROC AUC score.

Bibliography

[1] John E. Hall. *Guyton and Hall textbook of medical physiology.* Elsevier, 13 edition, 2016.

[2] Elaine Nicpon Marieb and Katja Hoen. *Human anatomy & physiology.* Pearson Education, 2013.

[3] M. Courtemanche, R. J. Ramirez, and S. Nattel. Ionic mechanisms underlying human atrial action potential properties: insights from a mathematical model. *Am J Physiol*, 275(1):H301–321, 07 1998.

[4] S. Dokos, B. Celler, and N. Lovell. Ion currents underlying sinoatrial node pacemaker activity: a new single cell mathematical model. *J Theor Biol*, 181(3):245–272, Aug 1996.

[5] Y. Kurata, I. Hisatome, S. Imanishi, and T. Shibamoto. Dynamical description of sinoatrial node pacemaking: improved mathematical model for primary pacemaker cell. *Am J Physiol Heart Circ Physiol*, 283(5):H2074–2101, Nov 2002.

[6] G. D. Veenhuyzen, S. Knecht, M. D. O'Neill, D. Phil, M. Wright, I. Nault, R. Weerasooriya, W. Rukshen, S. Miyazaki, F. Sacher, M. Hocini, P. Jaïs, and M. Haïssaguerre. Atrial tachycardias encountered during and after catheter ablation for atrial fibrillation: part I: classification, incidence, management. *Pacing Clin Electrophysiol*, 32(3):393–398, Mar 2009.

[7] U. R. Anoop and K. Verma. The ECG made easy for the dental practitioner. *Indian J Dent Res*, 25(3):386–389, 2014.

[8] A. K. Sangaiah, M. Arumugam, and G. B. Bian. An intelligent learning approach for improving ECG signal classification and arrhythmia analysis. *Artif Intell Med*, 103:101788, 03 2020.

[9] Gerald A. Serwer and Ira Shetty. 18 - pediatric pacing and defibrillator use. In Kenneth A. Ellenbogen, G. Neal Kay, Chu-Pak Lau, and Bruce L. Wilkoff, editors, *Clinical Cardiac Pacing, Defibrillation and Resynchronization Therapy*

(Fourth Edition), pages 393–427. W.B. Saunders, Philadelpia, fourth edition edition, 2011.

[10] Sok-Sithikun Bun, Tahar Delassi, Decebal Gabriel Latcu, Mohammed El Jamili, Anis Ayari, Abdelkarim Errahmouni, Benjamin Berte, and Nadir Saoudi. A comparison between multipolar mapping and conventional mapping of atrial tachycardias in the context of atrial fibrillation ablation. *Archives of Cardiovascular Diseases*, 111(1):33–40, 2018.

[11] A. L. HODGKIN and A. F. HUXLEY. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J Physiol*, 117(4):500–544, Aug 1952.

[12] D. NOBLE. A modification of the Hodgkin–Huxley equations applicable to Purkinje fibre action and pace-maker potentials. *J Physiol*, 160:317–352, Feb 1962.

[13] G. Plank, A. Loewe, A. Neic, C. Augustin, Y. L. Huang, M. A. F. Gsell, E. Karabelas, M. Nothstein, A. J. Prassl, J. Sánchez, G. Seemann, and E. J. Vigmond. The openCARP simulation environment for cardiac electrophysiology. *Comput Methods Programs Biomed*, 208:106223, Jun 2021.

[14] Nele Vandersickel, Enid Van Nieuwenhuyse, Nico Van Cleemput, Jan Goedgebeur, Milad El Haddad, Jan De Neve, Anthony Demolder, Teresa Strisciuglio, Mattias Duytschaever, and Alexander V. Panfilov. Directed networks as a novel way to describe and analyze cardiac excitation: Directed graph mapping. *Frontiers in Physiology*, 10:1138, 2019.

[15] Alison Stewart. Slco1b1 polymorphisms and statin-induced myopathy. *PLoS currents*, 5, December 2013.

[16] J. E. Poole and D. Wilber. Outcomes That Matter: Assessing Benefits of Atrial Fibrillation Catheter Ablation. *JACC Clin Electrophysiol*, 5(3):340–342, 03 2019.

[17] J. E. Poole and D. Wilber. Outcomes That Matter: Assessing Benefits of Atrial Fibrillation Catheter Ablation. *JACC Clin Electrophysiol*, 5(3):340–342, 03 2019.

[18] A. J. Camm, G. Y. Lip, R. De Caterina, I. Savelieva, D. Atar, S. H. Hohnloser, G. Hindricks, P. Kirchhof, J. J. Bax, H. Baumgartner, C. Ceconi, V. Dean, C. Deaton, R. Fagard, C. Funck-Bretano, D. Hasdai, A. Hoes, P. Kirchhof, J. Knuuti, P. Kolh, T. McDonagh, C. Moulin, B. A. Popescu,

Ž. Reiner, U. Sechtem, P. A. Sirnes, M. Tendera, A. Torbicki, A. Vahanian, S. Windecker, P. Vardas, N. Al-Attar, O. Alfierei, A. Angelini, C. Blömstrom-Lundqvist, P. Colonna, J. De Sutter, S. Ernst, A. Goette, B. Gorenek, R. Hatala, H. Heidbüchel, M. Heldal, S. Dalby Kristensen, P. Kolh, J. Y. Le Heuzey, H. Mavrakis, L. Mont, P. Perrone Filardi, P. Ponikowski, B. Prendergast, F. H. Rutten, U. Schotten, I. C. Van Gelder, and F. W. Verheugt. 2012 focused update of the ESC Guidelines for the management of atrial fibrillation: an update of the 2010 ESC Guidelines for the management of atrial fibrillation. Developed with the special contribution of the European Heart Rhythm Association. *Eur Heart J*, 33(21):2719–2747, Nov 2012.

[19] Igor Matias, Nuno Garcia, Sandeep Pirbhulal, Virginie Felizardo, Nuno Pombo, Henriques Zacarias, Miguel Sousa, and Eftim Zdravevski. Prediction of atrial fibrillation using artificial intelligence on electrocardiograms: A systematic review. *Computer Science Review*, 39:100334, 2021.

[20] G Luongo, S Schuler, M W Rivolta, O Doessel, R Sassi, and A Loewe. 236Automatic classification of 20 different types of atrial tachycardia using 12-lead ECG signals. *EP Europace*, 22(Supplement 1), 06 2020. euaa162.048.

[21] Julie K. Shade, Ashish N. Doshi, Eric Sung, Dan M. Popescu, Anum S. Minhas, Nisha A. Gilotra, Konstantinos N. Aronis, Allison G. Hays, and Natalia A. Trayanova. Covid-heart: Development and validation of a multivariable model for real-time prediction of cardiovascular complications in hospitalized patients with covid-19. *medRxiv*, 2021.

[22] Giorgio Luongo, Luca Azzolin, Massimo W Rivolta, Tiago P Almeida, Juan Pablo Martínez, Diogo C Soriano, Olaf Dössel, Roberto Sassi, Pablo Laguna, and Axel Loewe. Machine learning to find areas of rotors sustaining atrial fibrillation from the ecg. In *2020 Computing in Cardiology*, pages 1–4, 2020.

[23] Buntheng Ly, Sonny Finsterbach, Marta Nuñez-Garcia, Hubert Cochet, and Maxime Sermesant. Scar-related ventricular arrhythmia prediction from imaging using explainable deep learning. In Daniel B. Ennis, Luigi E. Perotti, and Vicky Y. Wang, editors, *Functional Imaging and Modeling of the Heart*, pages 461–470, Cham, 2021. Springer International Publishing.

[24] M. Clerx, J. Heijman, P. Collins, and P. G. A. Volders. Predicting changes to INa from missense mutations in human SCN5A. *Sci Rep*, 8(1):12797, 08 2018.

[25] B. Li and W. J. Gallin. Computational identification of residues that modulate voltage sensitivity of voltage-gated potassium channels. *BMC Struct Biol*, 5:16, Aug 2005.

[26] S. Ramasubramanian and Y. Rudy. The Structural Basis of IKs Ion-Channel Activation: Mechanistic Insights from Molecular Simulations. *Biophys J*, 114(11):2584–2594, 06 2018.

[27] B. A. Lawson, K. Burrage, P. Burrage, C. C. Drovandi, and A. Bueno-Orovio. Slow Recovery of Excitability Increases Ventricular Fibrillation Risk as Identified by Emulation. *Front Physiol*, 9:1114, 2018.

[28] R. Grosu, E. Bartocci, F. Corradini, E. Entcheva, S. A. Smolka, and A. Wasilewska. Learning and detecting emergent behavior in networks of cardiac myocytes. In Magnus Egerstedt and Bud Mishra, editors, *Hybrid Systems: Computation and Control*, pages 229–243, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[29] Mahesh Kumar Mulimani, Jaya Kumar Alageshan, and Rahul Pandit. Deep-learning-assisted detection and termination of spiral and broken-spiral waves in mathematical models for cardiac tissue. *Physical Review Research*, 2(2), May 2020.

[30] Cesare Corrado, Steven Williams, Caroline Roney, Gernot Plank, Mark O'Neill, and Steven Niederer. Using machine learning to identify local cellular properties that support re-entrant activation in patient-specific models of atrial fibrillation. *EP Europace*, 23(Supplement_1):i12–i20, 01 2021.

[31] N. A. Trayanova, D. M. Popescu, and J. K. Shade. Machine Learning in Arrhythmia and Electrophysiology. *Circ Res*, 128(4):544–566, Feb 2021.

[32] Ana Maria Sánchez de la Nava, Felipe Atienza, Javier Bermejo, and Francisco Fernández-Avilés. Artificial intelligence for a personalized diagnosis and treatment of atrial fibrillation. *American Journal of Physiology-Heart and Circulatory Physiology*, 2021.

[33] Cesare Corrado, Steven Williams, Caroline Roney, Gernot Plank, Mark O'Neill, and Steven Niederer. Using machine learning to identify local cellular properties that support re-entrant activation in patient-specific models of atrial fibrillation. *EP Europace*, 2021.

[34] Hang Si. Tetgen, a delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Softw.*, 41(2), February 2015.

[35] K. K. Aras, N. R. Faye, B. Cathey, and I. R. Efimov. Critical Volume of Human Myocardium Necessary to Maintain Ventricular Fibrillation. *Circ Arrhythm Electrophysiol*, 11(11):e006692, 11 2018.

[36] S. Verheule, J. Eckstein, D. Linz, B. Maesen, E. Bidar, A. Gharaviri, and U. Schotten. Role of endo-epicardial dissociation of electrical activity and transmural conduction in the development of persistent atrial fibrillation. *Prog Biophys Mol Biol*, 115(2-3):173–185, Aug 2014.

[37] Charles D. Hansen and Chris R. Johnson. *The visualization handbook*. Elsevier-Butterworth Heinemann, 2005.

[38] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. MeshLab: an Open-Source Mesh Processing Tool. In Vittorio Scarano, Rosario De Chiara, and Ugo Erra, editors, *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008.

[39] Martin W. Krueger, Viktor Schmidt, Catalina Tobón, Frank M. Weber, Cristian Lorenz, David U. J. Keller, Hans Barschdorf, Michael Burdumy, Peter Neher, Gernot Plank, Kawal Rhode, Gunnar Seemann, Damien Sanchez-Quintana, Javier Saiz, Reza Razavi, and Olaf Dössel. Modeling atrial fiber orientation in patient-specific geometries: A semi-automatic rule-based approach. In Dimitris N. Metaxas and Leon Axel, editors, *Functional Imaging and Modeling of the Heart*, pages 223–232, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[40] T. E. Fastl, C. Tobon-Gomez, A. Crozier, J. Whitaker, R. Rajani, K. P. McCarthy, D. Sanchez-Quintana, S. Y. Ho, M. D. O'Neill, G. Plank, M. J. Bishop, and S. A. Niederer. Personalized computational modeling of left atrial geometry and transmural myofiber architecture. *Med Image Anal*, 47:180–190, 07 2018.

[41] Byounghyun Lim, Minki Hwang, Jun-Seop Song, Ah-Jin Ryu, Boyoung Joung, Eun Bo Shim, Hyungon Ryu, and Hui-Nam Pak. Effectiveness of atrial fibrillation rotor ablation is dependent on conduction velocity: An in-silico 3-dimensional modeling study. *PLOS ONE*, 12(12):1–16, 12 2017.

[42] T. E. Fastl, C. Tobon-Gomez, A. Crozier, J. Whitaker, R. Rajani, K. P. McCarthy, D. Sanchez-Quintana, S. Y. Ho, M. D. O'Neill, G. Plank, M. J. Bishop, and S. A. Niederer. Personalized computational modeling of left atrial

geometry and transmural myofiber architecture. *Med Image Anal*, 47:180–190, 07 2018.

[43] D. Cozma, B. A. Popescu, D. Lighezan, P. Lucian, C. Mornos, C. Ginghina, and S. I. Dragulescu. Left atrial remodeling: assessment of size and shape to detect vulnerability to atrial fibrillation. *Pacing Clin Electrophysiol*, 30 Suppl 1:S147–150, Jan 2007.

[44] S. Y. Ho, J. A. Cabrera, and D. Sanchez-Quintana. Left atrial anatomy revisited. *Circ Arrhythm Electrophysiol*, 5(1):220–228, Feb 2012.

[45] C. Bane Sullivan and Alexander A. Kaszynski. Pyvista: 3d plotting and mesh analysis through a streamlined interface for the visualization toolkit (vtk). *Journal of Open Source Software*, 4(37):1450, 2019.

[46] Marco Attene. A lightweight approach to repairing digitized polygon meshes. *The Visual Computer*, 26:1393–1406, 2010.

[47] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. MeshLab: an Open-Source Mesh Processing Tool. In Vittorio Scarano, Rosario De Chiara, and Ugo Erra, editors, *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008.