An adaptive, preconditioned, electromechanical model for the simulation of cardiac arrhythmias

by

Nathan Robert Kirk

Submitted in accordance with the requirements for the degree of Doctor of Philosophy.

UNIVERSITY OF LEEDS

The University of Leeds
School of Computing

May 2012

The candidate confirms that the work submitted is his own and that the appropriate credit has been given where reference has been made to the work of others.

Abstract

In this thesis a coupled model of cardiac electromechanical activity is presented, using the finite element method to model both electrophysiology and mechanics within a deforming domain. The efficiency of the electrical model was improved using adaptive mesh refinement and the mechanical system performance was improved with the addition of preconditioning. Unstructured triangular meshes were used throughout.

The electrophysiology model uses the ten Tusscher-Panfilov 2006 detailed cellular model, and includes anisotropic diffusion, uses a semi-implicit time stepping scheme, stores data in an efficient sparse storage format and applies a Reverse Cuthill-McKee ordering algorithm to reduce the matrices' bandwidths. Linear elements were used to approximate the transmembrane voltage and spatial and temporal convergence tests were undertaken. Local mesh adaptivity is added to the electrical component of the model and improvements to the performance and efficiency gained by this technique were investigated. Two different monitor functions were utilised and these demonstrated that by targeting adaptive mesh refinement at the front of the electrical wave significant efficiency and performance benefits could be achieved.

The cardiac mechanical model is based on finite deformation elasticity theory, enforces the incompressibility of the tissue and incorporates anisotropic tension to simulate fibre orientation. This uses isoparametric quadratic elements for deformation, linear elements for pressure, was integrated with numerical quadrature and the resulting non-linear system solved with the iterative Newton method. Preconditioning was added to the mechanical component of the model and improvements in the performance of the solver due to this were investigated. An ILUT (Incomplete Lower Upper factorisation with drop Tolerance) preconditioner was implemented and this demonstrated performance improvements of up to 27 times on the meshes tested.

The resulting cardiac electromechanical solver was then used to consider how known changes in cardiac electrophysiology, which are manifest in end-stage heart disease, affect the stability of the electrical wave. Specifically, investigations were undertaken into the introduction of fibrotic regions (with different sizes and concentrations) and electrical remodelling caused by end-stage cardiac disease. These were modelled on both static and deforming domains to consider whether deformation can alter the stability of a spiral wave. These simulations demonstrated that fibrotic regions and tissue deformation can have significant disruptive effects on the stability of a re-entrant spiral wave and that remodelling the electrophysiology stabilises the wave.

Acknowledgements

I would like to thank my supervisors Dr Matthew Hubbard and Dr Christopher Goodyer for their guidance throughout this project. I have thoroughly enjoyed the experience and am very grateful for their support prior to, during, and at the completion of this thesis.

I would also like to thank all staff and students within the Scientific Computation research group, specifically Dr Mark Walkley, Professor Peter Jimack, Andrzej Warzynski, Keeran Brabazon, Dr Domokos Sarmany and Sarfraz, for all the help and advice they have given me over the last three and a half years. Additionally, the 'invincibles' computing football team has provided an excellent distraction to the challenges of academic work.

Most importantly, I would like to thank my wife, Mia, without her love, support and belief I would not have been able to undertake this project, and my children, Bradley, Gabrielle and Benjamin, who are a constant source of love and inspiration. Finally I would like to take this opportunity to thank my parents, for all their love and encouragement.

Declarations

Some parts of the work presented in this thesis have been published in the following article:

N. Kirk, A. P. Benson, C. E. Goodyer, and M. E. Hubbard, "Application of an efficient coupled electromechanical solver to investigate end stage human heart failure.", *Computing in Cardiology 2011*, 38:17–20, 2011.

Contents

List of Figures

List of Tables

Chapter 1

Introduction and thesis overview

1.1 Introduction

1.1.1 Cardiac function

The heart is essentially an electromechanical pump that has a repeating rhythmic action. It is the rapid propagation of electrical activation through cardiac tissue that initiates its contraction, and its electrical behaviour is influenced by the resulting mechanical activity. The heart is comprised of four chambers (see Figure 1.1), and these chambers work as two pairs, with the function of the left atrium and ventricle being to pump oxygenated blood around the body, and the function of the right atrium and ventricle to pump de-oxygenated blood to the lungs.

In a healthy heart operating normally, the electrical activity is governed by a 'pace-maker' node, called the sino-atrial node. This node starts an electrical wave which excites the surrounding cells and generates an action potential. An action potential is a self-regenerating wave of electrochemical activity that allows excitable cells to carry a signal over a distance, these are pulse-like waves of voltage that travel along several types of cell membranes such as cardiac muscle cells. The electrical wave travels from the sino-atrial node, in parallel around both atria, to the atrioventricular node, and then via the bundle of His and Purkinje fibres to excite the left and right ventricles.

As the electrical wave propagates through the atrial and ventricular muscle it causes it to contract, and it is this synchronised contraction that causes the pumping action of the

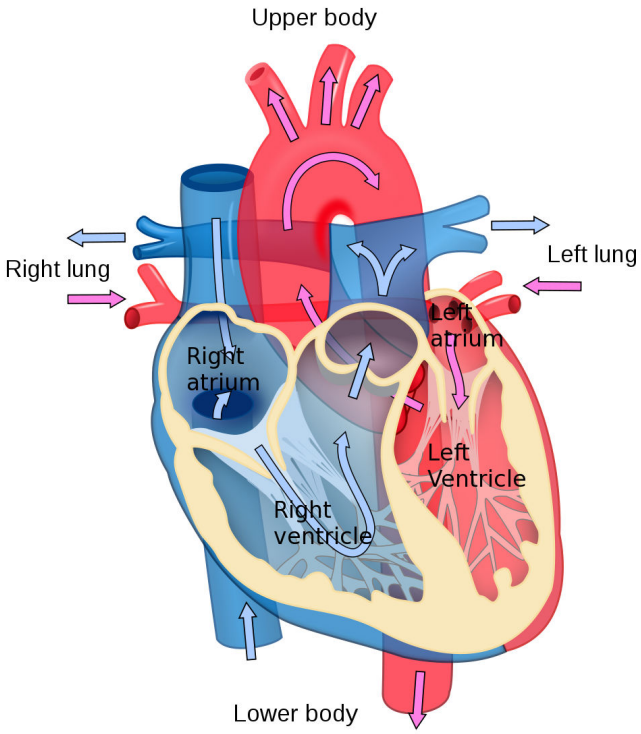


Figure 1.1: *Simplified diagram of the heart [128]*

heart. The atrioventricular node delays the electrical impulses (by having a slower conduction velocity) to ensure the atria have ejected their blood into the ventricles first before the ventricles contract. The electrical activity is caused by the movement of numerous ions from inside the cells to outside the cells and also between them. This movement of ions also causes the cells to contract.

The left ventricle pumps oxygenated blood around the body and is typically the main focus of simulation and modelling work. The left and right ventricles are not the same shape, the left ventricle is quite conical and its transverse cross-section is oval or nearly circular. The right ventricle wraps around the left ventricle and its transverse cross-section is more crescent shaped.

1.1.2 Motivation of research area

Cardiac and circulatory disease are the largest cause of death in the UK. In 2008, over 191,000 [97] died from heart and circulatory disease including 88,000 deaths from Coronary Heart Disease and a further 43,000 [97] from strokes. More than one in four deaths in men before the age of 75 and one in five deaths in women before the age of 75 are from

cardio-vascular disease. There are around 124,000 heart attacks in the United Kingdom every year.

The aim of computational simulations of the heart is to better understand how it functions, and provide the ability to model specific pathological conditions or medical intervention techniques. Computer models provide an efficient, and cost effective way of undertaking such research prior to clinical trials which are expensive and potentially a risk to the patients.

1.2 Introduction to cardiac modelling

1.2.1 Brief historical background

From the late nineteenth century it has been known that electrical excitation of muscle tissue causes it to contract. In terms of the mathematical and computational modelling of such activity the ground-breaking work was undertaken by Hodgkin and Huxley in 1952 [45]. The Hodgkin-Huxley model is a mathematical model that describes how action potentials are propagated in the squid giant axon, and for this work they were awarded the Nobel Prize in Physiology or Medicine in 1963.

Noble adapted the work of Hodgkin and Huxley to apply it to the Purkinje fibres of the heart [74]. Over the next 60 years the field of cardiac modelling has developed, with researchers across the world developing models for cardiac electrophysiology, cardiac mechanics, blood flow and also generating better representations of the cardiac muscle. The objective of this section is to provide a brief introduction to the various areas of mathematical and computational modelling of cardiac electro-mechanical activity. However, for a more comprehensive introduction to the modelling of cardiac functions please refer to [53, 81, 127].

1.2.2 Electrophysiology modelling overview

Figure 1.2 shows an example cardiac action potential and this involves a number of phases, numbered 1 to 5, as follows:

1. This is the resting potential and is the 'default' status of the cell, where the sodium ions, Na^+, are concentrated on the outside of the cell and the potassium ions K^+ are concentrated on the inside. The resting potential of a cell is negative, between -85mV to -95 mV.

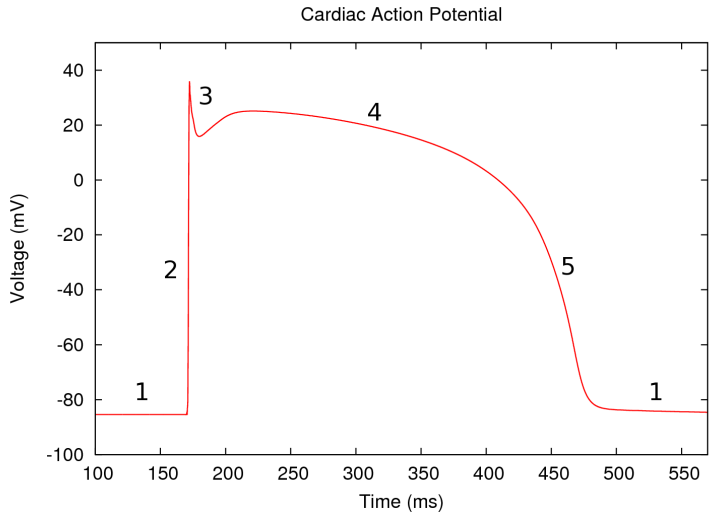


Figure 1.2: *An example plot of cardiac transmembrane voltage.*

2. The rapid depolarization phase is due to the opening of the fast Na^+ channels causing a rapid increase in the membrane conductance to Na^+ and thus a rapid influx of Na^+ ions into the cell. In Figure 1.2 this is the steep slope upwards.

3. Following the rapid depolarisation there is a small dip in net current caused by the inactivation of the fast Na^+ channels.

4. The plateau phase of the cardiac action potential is sustained by a balance between inward movement of calcium ions through calcium channels and outward movement of K^+ through rectifier potassium channels.

5. During the rapid repolarisation phase the Ca^{2+} channels close, while the slow delayed rectifier K^+ channels are still open. This ensures a net outward current, corresponding to negative change in membrane potential, thus allowing more types of K^+ channel to open. This net outward, positive current (equal to loss of positive charge from the cell) causes the cell to repolarize. In Figure 1.2 this is the downward slope back to the resting potential.

It should be noted that the above action potential has the 'spike and dome' profile, and this is seen in the epicardial (the outer layer of heart tissue) and the M cell types, but not in the endocardial cells (the innermost cells). This is due to sensitivity of the transient outward current (I_{to}) [43]. M cells, which are also known midmyocardial or Moe cells (named in memory of Gordon K. Moe [101]), are between the surface epicardial and endocardial layers.

As can be seen in the steps above, to fully model the cardiac action potential it is necessary to model the movement of various ions through the cell wall and the state of various ion pumps and exchangers. In [30] the transmembrane currents and other cellular ionic processes of 45 different electrophysiology models are compared. Clearly there many approaches used to accomplish electrophysiological modelling, and in [21], four groups of models are discussed, namely simplified two variable models, bio-physically detailed models (first and second generation) and reduced cardiac models. These techniques are discussed below.

1.2.3 Types of electrophysiology model

Simplified cell models do not seek to model all the ion currents within the cell, rather they use two variables to model the cell excitation and then subsequent recovery. These are derived from the Fitzhugh Nagumo model (FHN) [33] and take the form:

$$\frac{\partial V}{\partial t} = f(V,r), \qquad \frac{\partial r}{\partial t} = g(V,r), \tag{1.1}$$

where V is the activation variable and r is the recovery variable.

Many variants of the FHN model have been produced and one example is the Aliev and Panfilov model [2]. This was developed to model the transmembrane voltage within cardiac tissue and is a reaction-diffusion partial differential equation that takes the form:

$$\frac{\partial V}{\partial t} = \nabla \cdot (D\nabla V) - kV(V-a)(V-1) - rV + I_s, \tag{1.2}$$

$$\frac{\partial r}{\partial t} = (\varepsilon \frac{\mu_1 r}{\mu_2 + V})(-r - kV(V-b-1)), \tag{1.3}$$

where V is the transmembrane potential, r is the re-polarising current, and I_s is the current due to external stimulus. μ_1, μ_2, a, b, ε and k are parameters used to modify the behaviour of the wave, based on observation.

The two-variable models have a number of advantages, discussed in Clayton et al. [21], particularly the speed at which they can be solved and that the parameters are directly set to mimic a cardiac action potential. These types of models seek to provide a representation of transmembrane voltage, but do not attempt to accurately model activity within the cells. This means that "the behaviour and influence of specific ion channels and other intracellular processes on propagation" [21] cannot be modelled as they are not included. It is further stated in [21] that these models should be used to study the general effects of wave propagation or when results are needed in the shortest possible time.

Physically detailed cell models incorporate the movement of calcium, sodium and potassium ions, the functions of the sodium/potassium pumps and a number of ion "gates" that open at a particular time in the cycle. There are also different types of channels (long-lasting and transient) that respond differently to voltage changes across the cell membrane. These models provide a more accurate model of the electrophysiology.

The Beeler and Reuter model [8] includes descriptions of four currents and a simplified model of intracellular Ca^{2+}. The Luo Rudy 1 (LR1) [63] model additionally includes a voltage dependent background current and a time-dependent K^+ current and this is then developed further in [62]. These kinds of model are far more involved that the simplified models of action potential and at each time-step require a large number of ordinary differential equations to be solved. These equations are generally non-linear and stiff (a stiff equation is one where numerical methods for solving the equation may be unstable, unless the step size is taken to be extremely small). The changes occurring within a cell (for example the opening of Na^+ channels) occur very rapidly and therefore need very small time steps [21].

The second generation of physically detailed models have additional equations that balance the intracellular and extracellular ion concentrations [21]. The first of these models was the DiFrancesco-Noble model [26] of Purkinje cells and a number of other second generation models have been produced, based on various animal hearts including guinea pig, dog, rabbit and human. The ten Tusscher and Panfilov model is a second generation model of human tissue [108, 110] and is used within this thesis.

These models provide the necessary detail to simulate changes in individual ion currents that may be caused by cardiac disease. The second generation models are more computationally expensive to solve as they include more equations, which are needed to model ion channels, pumps, exchangers and intracellular ion concentrations [21].

Reduced cardiac models simplify the underlying cellular activity, and model the electrophysiology by modelling the ion currents through the cell membrane. The Fenton and Karma model [29] puts forward a three-variable model that tracks the total current through the cell membrane as the sum of a fast inward current, analogous to the sodium current, a slow outward current, analogous to the time-dependent potassium current, and a slow inward current, analogous to the calcium current. There are a number of extensions to this model, for example Cherry and Fenton in 2004 [17] added an extra variable to enable a "spike and dome action potential morphology to be reproduced" [21].

Fenton and Karma proposed their model because, even though the ionic models have become more complex, they do not assist in the resolution of all issues with the electrophysiological modelling, specifically extensive 3D simulations are difficult to carry out

and their inherent complexities make it difficult to isolate a subset of essential parameters.

1.2.4 Monodomain and bidomain models of electrophysiology

The monodomain approach assumes that cardiac tissue behaves as an excitable medium, with diffusion and local excitation of membrane voltage. It provides the simplest description of action potential propagation. In contrast to the monodomain model, the bidomain approach considers cardiac tissue as a two phase medium comprising intracellular and extracellular spaces. The earliest version of this was proposed in 1978 [115]. The transmembrane potential in the bidomain model is the difference between the intracellular Φ_i and extracellular Φ_e potentials:

$$V_m = \Phi_i - \Phi_e, \tag{1.4}$$

where V_m is the membrane voltage. The overall system of bidomain equations is given by the following [21, 42, 93]:

$$\nabla \cdot (\mathbf{G}_i + \mathbf{G}_e)\Phi_e = -\nabla \cdot (\mathbf{G}_i \nabla V_m), \tag{1.5}$$

$$\nabla \cdot (\mathbf{G}_i \nabla V_m) + \nabla \cdot (\mathbf{G}_i \nabla \Phi_e) = -S_v I_m, \tag{1.6}$$

$$I_m = \mathbf{C}_m \frac{dV_m}{dt} + I_{\text{ion}}, \tag{1.7}$$

where I_m is the current flow through the membrane, G are the conductivity tensors (the subscripts i and e relating to the intracellular and extracellular spaces), S_v is the surface to volume ratio of the cells, I_m is the membrane current, and $\mathbf{C}_m$ is the membrane capacitance.

Studies have been done to consider the differences between the two methods (monodomain and bidomain), for example [85]. In this paper the differences between the two models are investigated and it is demonstrated that for modelling propagating currents the monodomain model produces similar results to the bidomain model. The monodomain models cannot model realistic extracellular potentials by themselves, however additional calculations can be performed to model extracellular potentials from membrane currents. This method only works where there are no applied currents [85], and this paper concludes that the monodomain model is sufficient for propagation studies.

The bidomain is more computationally expensive, as the matrix of membrane capacitance, $\mathbf{C}_m$, plays a more significant role in the computation. $\mathbf{C}_m$ is not invertible [122] and this means that simpler explicit schemes cannot be used for integrating in time. This has

typically meant that simulations with bidomain models have been undertaken on smaller domains.

1.2.5 Introduction to cardiac mechanics

Within cardiac tissue, a cardiac myocyte is a specialized muscle cell that is composed of bundles of myofibrils. The myofibrils have distinct, repeating units, termed sarcomeres, which represent the basic contractile units of the myocyte. As the electrical wave passes through the cardiac tissue, the normal sarcomere length can change significantly (from a starting length of approximately 2.0 μm), with [65] noting this can be up to 13.9% (in canine epicardial cells). This contraction is typically modelled with finite deformation elasticity theory [47, 54, 69]. These models are based on general conservation principles of space, mass and momentum and these are explained further in Chapter 3.

Compared to the electrical modelling, which has been studied extensively, there is less published work in the area of cardiac contraction, and it is commented on in Fink et al. [31] that these models have generally been of a lower complexity. However, over recent years, detailed biophysical models of the inner workings of the sarcomere have emerged [14, 73] and more complex coupled electro-mechanical models have been developed [20, 54, 58].

In [69] it was demonstrated that the properties of re-entrant waves (see Section 1.3) are significantly altered by movement of the muscle cells, for example period of excitation and stability of rotation [69]. Their recommendation is that any model using a static undeforming geometry should be interpreted with caution.

In Section 1.2.2 the movement of ions within and between cells was discussed, and for the contraction of tissue calcium is considered to be the most important [9]. Cardiac excitation-contraction coupling is the process from electrical excitation of the myocyte to contraction of the tissue. The movement of Ca^{2+} is an essential part of cardiac electrophysiology and the myofilaments (a myofibril is the basic unit of a muscle) are directly affected by its behaviour to cause contraction.

Aspects of the cardiac modelling which make it difficult are the fact that the electrical propagation is affected by the size and location of the tissue, that the deformation of the cells affects cellular electrophysiology, and that the size and shape of the muscle is affected by the contractile forces generated by the electrical activity. This sets up a complex feedback system (known as mechano-electrical feedback) whereby each system is dependent on the other.

Stretch-activated channels (SACs) are ion channels which open in response to me-

chanical deformation of the cell. This process means the electrophysiological behaviour is altered as the cells deform, with the calcium transient being sensitive to the sarcomere length [55]. When SACs are being modelled it is necessary to use strong coupling of the mechanical and electrical systems. With strong coupling both the electrical and mechanical systems are solved together as one system of equations. Strongly coupled systems make it possible to incorporate specific biological effects of electromechanical coupling. For example, length-dependent calcium bindings or stretch-activated conductance channels. These features are present in some cellular and tissue models (for example [59,75,104]). The other approach used (for example in [69]) is 'weak coupling'. With weak coupling the electrical and mechanical systems are solved independently, and typically at different time intervals. An active tension [69, 82] is generated from the electrical system and this is used to provide a tensile force to the mechanical system.

Cardiac tissue is composed of fibres and sheets and the contraction is predominantly along the length of the fibres. In [71], Nickerson et al. developed a model of cardiac electromechanics and used it to investigate fibre length effects. This demonstrated that mechanical contraction produces a longitudinal and twisting deformation.

It is also now typically assumed that cardiac tissue is incompressible [54, 68, 82]. An incompressible material does not change volume (or area in two dimensions) when pressure is applied to it. Modelling cardiac tissue as incompressible can add complexity to the model as extra terms need to be included to enforce the incompressibility constraint, and it may be necessary to use higher order approximations for the deformation unknowns.

1.2.6 Recent cardiac mechanical models

In Section 1.2.5 the complexities of modelling cardiac contraction are briefly highlighted, with the need to consider the fibre anisotropy, mechanical-electric feedback, stretch activated channels and incompressibility amongst other things. Recent published research has attempted to overcome these obstacles and model specific electro-mechanical effects. In [20] mechanical deformation is used to induce electrical activity via the stretch-activated channels. It then considers how deformation can cause refractory areas that change the stability of the spiral wave.

In [54] a three-dimensional electro-mechanical model of the left ventricle of the human heart is presented. This uses a second generation detailed electrophysiology model [110] to model human cardiac cells and the mechanical activity is represented by the Niederer-Hunter-Smith [73] active tension model. These are modelled in a geometry that contains a detailed description of fibre orientation. Their findings were then compared

to experimental recordings, and the model used to study the effect of mechano-electrical feedback via the stretch-activated channels on the stability of re-entrant cardiac spiral waves. Their conclusions being that mechanically induced spiral wave break-up is closely associated with the deformation of the cardiac tissue and the presence of stretch activated channels. In [76] a coupled cardiac electro-mechanical model is produced and discusses the potential clinical application of such models.

1.3 Cardiac arrhythmias

Cardiac arrhythmia is the term used for a group of conditions in which there is abnormal electrical activity in the heart. Cardiac arrhythmias can form re-entrant waves, which can spiral within an entire chamber. These cause conditions known as atrial and ventricular tachycardia, which cause the chambers to contract out of sequence and the pumping action of the heart to be reduced. The chamber in which the tachycardia occurs beats too quickly and, in the case of ventricular tachycardia, has not been properly filled with blood before it expels it. Typical arrhythmias include:

- Bradycardias - a slow rhythm, less than 60 beats/minute.
- Tachycardias - any heart rate faster than 100 beats/minute.
- Re-entry - these arrhythmias occur when an electrical impulse recurrently travels in a tight circle within the heart, rather than moving from one end of the heart to the other and then stopping.
- Fibrillation - the electrical activity within an entire chamber becomes chaotic and the muscle twitches randomly rather than contracting in a coordinated fashion.

The key area for modelling is the analysis of the development of re-entrant electrical impulses and how these then break-up into chaotic patterns. It is this break-up that causes ventricular fibrillation and this is imminently life-threatening.

It is worth noting that every cardiac cell is able to transmit impulses in every direction, but will only do so once within a short period of time. Normally, the action potential impulse will spread through the heart quickly enough that each cell will only respond once. However, if conduction is abnormally slow in some areas, part of the impulse will arrive late and potentially be treated as a new impulse. Depending on the timing, this can produce a sustained arrhythmia.

1.3.1 Modelling arrhythmias

Computational models of cardiac electrophysiology are used to simulate the effects of known biological problems with the cardiac wave. Studying cardiac arrythmias numerically removes the inherent difficulties in undertaking physical experiments and provides useful tools for the biological and medical communities to consider intervention techniques.

Figure 1.3 illustrates how the two-dimensional representation of cardiac electrophysiology produced by computational models relates to biological conditions. The first row of images shows example electrocardiogram (ECG) output for three cardiac rhythms, namely normal sinus rhythm, tachycardia and fibrillation. The second row gives examples of how these are simulated in two dimensions and the third row shows examples of the action potential wave form caused by these rhythms.

In cardiac simulation re-entrant waves are self-sustaining waves which form a spiralling pattern within the domain. These are artificially instigated within the computational model. However due to their self-sustaining nature it is possible to create a stable spiral wave and then investigate its properties.

Ventricular fibrillation is a cardiac arrhythmia, whereby the re-entrant spiral waves seen in a tachycardia break-up to form a chaotic pattern. This arrhythmia has fatal consequences as the heart stops contracting rhythmically and quivers in a chaotic state. The break up of previously stable spiral waves has been considered in cardiac simulations (see [20, 110]), where amendments to the electrophysiology or mechanics are considered.

1.4 Cardiac geometry considerations

The heart is a complex organ in which the muscle tissue is composed of billions of individual cardiac myocytes (muscle cells), and these are structured in fibres and then fibres structured into sheets. The chambers of the heart are often simplified to smooth geometric shapes, however in reality there are various fibres, veins and other tissue within the chambers that makes their interior structure very complex. The images within [38] clearly illustrate the level of complexity present in a real heart.

In Kerckhoffs *et al.* [55], a number of anatomically detailed three-dimensional models of geometry and fibre orientation are discussed. For example, they reference Vetter and McCulloch [117] and their description of a rabbit heart. Similar work has been done for mouse, dog, pig, sheep and human hearts. These have been obtained from histological (under microscope) measurements or magnetic resonance imaging (MRI) scans [55]. The

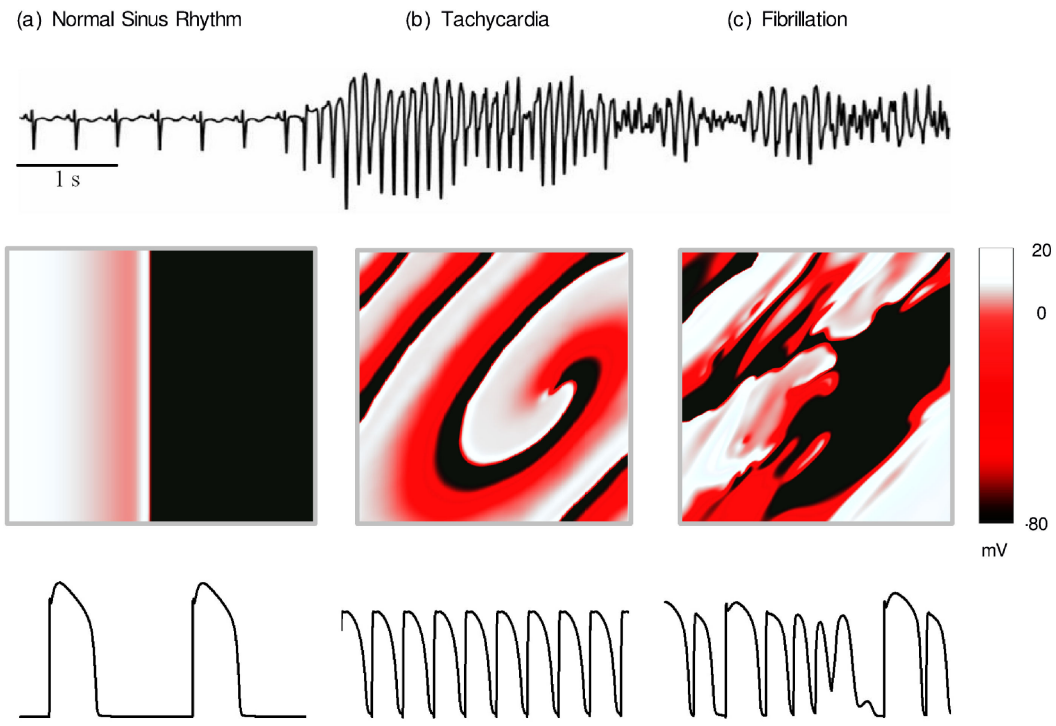


Figure 1.3: *Representation of cardiac arrhythmias. The first row of images shows example electrocardiogram (ECG) output for three cardiac rhythms, namely normal sinus rhythm, tachycardia and fibrillation. The second row gives examples of how these are simulated in two dimensions and the third row shows examples of the action potential wave form caused by these rhythms. ECG and APD wave images courtesy of Alan Benson.*

creation of accurate three-dimensional models and their conversion to meshes used in mathematical models is an area of research in its own right.

The challenge for modelling is that to accurately represent all the individual features of the heart requires computational grids/meshes with a very fine resolution. The finer the resolution the larger the system size and the more computing resources are required to solve the problem. There is inevitably a trade-off between these. When simulating cardiac activity it is also important to consider the particular effect being modelled or investigated. Simplified geometries and lower dimensions can be used to model specific effects, providing illustrative results, without the need to undertake prohibitively time-consuming simulations on an accurate three-dimensional geometry. Simplification of the geometry also makes the visualisation of results easier and this can assist in the identification of patterns within the results.

1.5 Numerical modelling

Early investigation into cardiac electrophysiology [74] considered the action potential and ionic currents on a single cell and over time more advanced single cell models were developed, for example [8, 26, 63]. These produce more detailed representations of the activity within a single cell with regard to the movement of ions, and their associated pumps and exchangers.

These single cell models allow variation in time and require a set of stiff ordinary differential equations to be solved. The numerical schemes used to approximate these models include explicit Euler, Runge-Kutta and implicit techniques [86]. A commonly-used approach was proposed by Rush and Larsen [94] which simplifies the system to one differential equation and uses a time-step adjustment strategy to increase efficiency.

Models to simulate the propagation of the electrical wave around the cardiac muscle, for example an entire ventricle, were developed in parallel with the single cell models. These simulations require the stiff ordinary differential equations that are required for the solution of the cellular activity to be embedded within a monodomain or bidomain representation of cardiac tissue.

These models vary in space as well as time, and the most common numerical method used to approximate these systems is the finite difference method. Finite difference methods have been widely used to simulate the propagation of the electrical wave over a domain and examples of these can be found in [18, 29, 81].

Modelling in both space and time is inherently more complex and the diffusion term that is present in both the monodomain and bidomain equations make these either parabolic

partial differential equations (monodomain) or a combination of elliptic and parabolic partial differential equations (bidomain). It is also important that the space step used for these methods is small enough to produce smoothly curved wavefronts. These wavefronts should not be not distorted by the modelling mesh. For example, Cherry and Fenton [17] use a space step of 0.25 mm, and this produces smooth spiral waves.

A technique for improving the performance of computations is operator splitting, where computation of the diffusion term and cellular models are separated [88]. Vigmond et al. [118] comment that operator splitting is often performed to separate the large non-linear system of ordinary differential equations. This then enables semi-implicit or fully implicit methods to be used for the remaining system of partial differential equations. The time-step for solving the cell model can also be made adaptive, with a shorter time step at the steep up-slope of the action potential, and this can result in improved performance of the solver.

In later work the finite element method (FEM) has been used to model the electrophysiology over a three-dimensional domain. Franzone et al. [34] use a parallel FEM solver with isoparametric trilinear finite elements in space and a semi-implicit adaptive method in time. Ying et al. [124] discretise the domain spatially with the continuous piecewise linear FEM and use an adaptive time-stepping strategy. The FEM techniques have the advantage that complex geometries are easier to represent (than with finite differences), however the FEM is more difficult to implement.

As described in Section 1.2.5, cardiac mechanics are modelled using finite deformation elasticity theory. This is typically modelled using the FEM as can be seen in [48,54,58,82,121]. This has been done on simplified two-dimensional tissue 'sheets' [69] and on more complex three-dimensional ventricular representations [54].

In recent years adaptive methods have been applied to the field of cardiac modelling, with methods for adapting both temporally and spatially being developed, using both finite differences and finite elements. An adaptive scheme for the bidomain model is proposed in [83], and this uses the finite element method and decomposes the domain, with each sub-domain then being independently refined. Further spatial adaptivity schemes have been developed for modelling cardiac electrophysiology [18, 19, 58, 122, 123], each of which aim to improve the efficiency of the solver whilst maintaining the solution accuracy.

In [114] both spatial and time adaptivity is applied, with an operator splitting technique used to decouple the diffusion and reaction terms. In [58] a strongly coupled electro-mechanical model is presented that includes techniques to efficiently approximate the electrical system over a larger time step and has features to speed up the mechanical aspects by modifying the Newton method.

1.5.1 Typical modelling limitations

The ultimate aim of cardiac modelling and simulation is to develop an accurate three-dimensional moving model of the human heart. Modelling cardiac behaviour completely requires the accurate representation of many physical attributes, for example a three-dimensional geometry with appropriate anisotropic tissue structure, a detailed electro-physiological model, an anisotropic tissue mechanics model incorporating fibre structure and mechano-electrical feedback. To date a complete coupled model of deforming tissue has not been produced as the computational complexity is too high.

The typical approach by computational modellers is to simplify certain aspects of the model and seek to produce results focused at one particular area of behaviour. Typical constraints and limitations applied to the models include:

- The geometry of the heart is modelled in two-dimensions.
- Three-dimensional models use simplified representations of the cardiac geometry, for example smooth truncated ellipsoids.
- The electrical diffusion is assumed to be isotropic rather than anisotropic.
- The monodomain model rather than the bidomain model are used.
- Certain aspects of mechano-electric feedback (e.g. stretch-activated channels) are ignored.
- Modelling one chamber (typically the left ventricle).
- Ignoring the effects of the fibre and sheet structure in the models.
- Various combinations of the above.

These techniques seek to make the problem being modelled tractable on available computing equipment.

Another issue with the building of a complete model is that as the complexity of the models increase so do the numbers of parameters. When combining multiple methods there may be parameter sensitivities that reduce the overall confidence in the results [55].

There are also many issues with the mathematical modelling of cardiac tissues and these are summarised in [122]. Firstly, certain modelling techniques, for example the finite difference method, struggle to model the complex cardiac geometry accurately. Secondly, to simulate the steep up-slope of the action potential the computational mesh needs to be very fine, if a fine mesh is used throughout the domain then this results in a very

large system size which is time consuming to solve and requires large amounts of memory to store. Numerical methods often have a time-step requirement that is tied to the spatial resolution, that is having a fine spatial resolution will often then mandate a small time-step. This increases the computational time required to model a period of real time.

1.6 Standardisation and re-usability

1.6.1 CellML

CellML is an XML standard that has been developed in the Bioengineering Institute at the University of Auckland [70]. It was designed as a means of storing and exchanging biological models. The mathematical equations in CellML are encoded using MathML (see http://www.w3.org/Math), which is a markup language designed to facilitate the use and presentation of mathematical content on the Web.

The interesting aspect of CellML is that a number of researchers in the field of cardiac simulation are publishing CellML versions of the their work. This may make it easier to reproduce their findings and then progress the research from there. One of the challenges of cardiac modelling is the ability for different research teams to use models developed by other teams, especially with regard to understanding the parameters and set-up requirements of a model and under what situations it is applicable [31].

It also provides a vehicle for corrections, for example the Nash and Panfilov paper [69] has an updated version at the CellML website (see http://www.cellml.org/models/nash_panfilov_2004_version01 as of June 2009).

1.6.2 Pre-developed software packages

There are a number of pre-developed modules or packages made available by the academic community or commercial developers to assist in aspects of cardiac modelling. Examples of these include:

- The Continuity project from UCSD (http://www.continuity.ucsd.edu). This is "a problem-solving environment for multi-scale modelling and data analysis in bioengineering and physiology, especially finite element modelling in cardiac biomechanics, biotransport and electrophysiology".

- Chaste (Cancer, Heart and Soft Tissue Environment). This is a general purpose simulation package aimed at multi-scale, computationally demanding problems arising in biology and physiology. See http://www.cs.ox.ac.uk/chaste/ for more details.

- COMSOL Multiphysics. This is more generalised engineering software and facilitates mathematical modelling of physical events. The software includes the ability to define a geometry, produce a mesh, define the problem mathematically and then solve and visualize the results. See http://www.uk.comsol.com/ for more details.

These packages aim to speed up the solution of various biological problems by providing a pre-built framework/toolkit to be developed on, for example in [20] the COMSOL package is used. These pre-developed packages are not used in this thesis, however are mentioned to illustrate the sophisticated tools available for biological modelling. For certain problems these packages may provide the functions necessary to complete the mathematical and computational modelling, thus enabling biologists to quickly simulate such problems without the need to develop their own modelling software.

1.7 Summary and thesis overview

1.7.1 Summary

This chapter has highlighted some of the complexity involved in modelling cardiac electromechanical activity. This is a vibrant research area and is one of the pioneers of the systems biology approach, whereby computational and mathematical models are used to better understand complex biological problems. This is illustrated by the cardiac physiome project [7,31] which is providing a means for disparate research teams to collaborate and share progress.

Modelling cardiac electrophysiology is a well studied area of research and there are many published mathematical models. Choosing the best model depends on the specific issues the research is looking to investigate and the computational resources available. Each of the published models has strengths and weaknesses and these are generally explained by the authors. There do, however, remain significant differences between certain models which were developed to explain similar phenomena [31] and so care needs to be taken when considering each model. It is also possible to utilise different electrical models once an overall framework is built. For example, in the electrophysiology modelling work undertaken in this thesis, the Barkley [6] model of a general excitable media was used initially, this was then replaced with the Aliev-Panfliov model [2] and then the ten Tusscher-Panfilov model [110] used.

The modelling of coupled deforming electro-mechanical models has progressed more slowly than the electrophysiological models. However over the last 4-5 years a number of complex coupled models [54, 58, 69] have been published and these seek to investigate

the effects that the movement of the domain (and the associated feedback processes) have on the electrical wave.

1.7.2 Overview of thesis

The objective of this thesis was twofold, first to build an efficient coupled electromechanical model with a deforming domain, and then to use this to model specific conditions prevalent in end-stage cardiac disease.

To undertake detailed simulations of cardiac activity requires the modelling of complex partial differential equations over a complex geometry. These computations can be very time-consuming, even in simplified geometries, and this thesis will consider ways to improve the speed of such simulations with an understanding of how this affects the accuracy of the solution. Efficiency improvements have been considered by the addition of preconditioning and locally adaptive mesh refinement.

End-stage disease conditions have been applied by changing the electrophysiological properties and by amending the underlying modelling domain (this is covered in detail in Chapter 7). To this end, techniques were used to provide the required complexity where needed (for example by using the ten Tusscher-Panfilov electrophysiological model) and simplifications elsewhere, for example using a two-dimensional domain. Emphasis will be placed on modelling the formation and dissipation of rotating spiral waves, that have been shown to be a key factor in many dangerous cardiac arrhythmias [84].

The techniques used and their outcomes are presented in detail in the following chapters.

Chapter 2

Cardiac electrophysiology

2.1 Introduction

The objective of this chapter is to describe the techniques used in this thesis to model the cardiac electrophysiology. The chosen mathematical models are described and then their approximation using the finite element method (FEM) is discussed. Practical issues such as, solving the resulting system of equations, how the progression of time is dealt with and how the modelling domain is discretised, are then discussed. Finally, the chapter presents results to demonstrate the convergence of the approximation and validate the model against other published work.

2.2 Governing equations

There are many published representations of cardiac electrophysiology, e.g. [2, 26, 33, 53, 62, 75, 108] and the review paper [21] has an excellent summary of the various techniques available. In this thesis the monodomain model is used, which is described by the following parabolic partial differential equation [53]:

$$\frac{\partial V}{\partial t} = -(I_{\text{ion}} + I_{\text{stim}}) + \nabla \cdot (\mathbf{D}\nabla V), \tag{2.1}$$

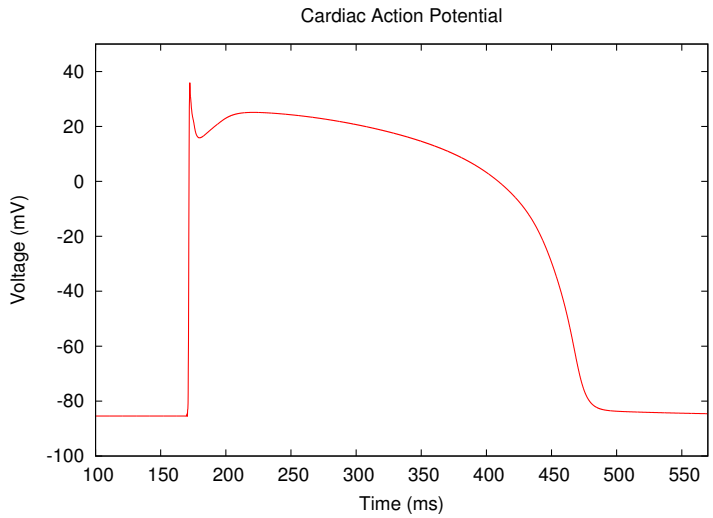


Figure 2.1: *An example plot of cardiac transmembrane voltage produced by the TP06 model*

where **D** is the diffusion tensor, V is the transmembrane potential, I_{ion} is the sum of all ionic currents, and I_{stim} is the externally applied transmembrane current.

2.2.1 Ionic current model

In this work I_{ion} is given by the ten Tusscher and Panfilov 2006 model [110] (referred to hereafter as TP06), which provides a detailed description of individual ionic currents and intracellular ion concentrations. This is a cellular model of the electrical wave propagation in human ventricular tissue based on experimental data. The TP06 model calculates most of the major ionic currents, and this enables the simulation of certain pathological conditions, for example end stage heart failure (as is modelled in Chapter 7). The model also includes calcium dynamics, which allow the realistic modelling of calcium transients and calcium current inactivation (also utilised further in Chapter 7).

The TP06 model provides a detailed approximation of the cardiac action potential wave form, as can be seen in Figure 2.1, and for reference purposes the TP06 model is included in the appendices (see Section C).

2.3 Modelling with the Finite Element Method

2.3.1 Weak form of governing equation

The electrophysiology is modelled with the parabolic reaction diffusion equation described in Equation (2.1). To model the unknown variables (in this case the transmembrane voltage V) over the domain the finite element method (FEM) is used. With the FEM, the domain is discretised into smaller 'elements' and the unknown variables approximated within these elements. The typical domain used in this thesis is a two-dimensional rectangle (with axes of X_1 and X_2) and this is divided into a mesh of unstructured triangles. The meshes used in this project are generated by the 'Triangle' [99, 100] mesh generation tool (see Section 4.1.1 for more details).

The FEM requires the governing equations to be in the 'weak' form (otherwise known as 'variational' form). The weak form of Equation (2.1) is obtained by first multiplying by a weight function (Ψ) and then integrating over the domain Ω, to give:

$$\int_\Omega \Psi \frac{\partial V}{\partial t} + \Psi(I_{\text{ion}} + I_{\text{stim}}) + \Psi \nabla \cdot (\mathbf{D}\nabla V) d\Omega = 0. \tag{2.2}$$

For the purposes of deriving the FEM model, the reaction term ($I_{\text{ion}} + I_{\text{stim}}$) will be referred to as f. If the diffusion is set to be isotropic then $\mathbf{D} = D\mathbf{I}$ and, by applying Green's identity, Equation (2.2) gives the weak form:

$$-\int_\Omega \Psi \frac{\partial V}{\partial t} d\Omega + D\int_\Omega \nabla\Psi \cdot \nabla V d\Omega - D\int_{\partial\Omega} \Psi \frac{\partial V}{\partial n} dS = \int_\Omega \Psi f d\Omega, \tag{2.3}$$

where $\partial\Omega$ is the boundary of the domain, $\frac{\partial V}{\partial n}$ is the derivative of V in the direction normal to the boundary. In this thesis no-flux Neumann boundary conditions are applied, in which the boundary term $\frac{\partial V}{\partial n}$ is a given function. With the no-flux condition $\frac{\partial V}{\partial n} = 0$, so no voltage is entering or leaving the domain.

By using

$$D\int_\Omega \nabla\Psi \cdot (\nabla V) d\Omega = \int_\Omega D\left(\frac{\partial\Psi}{\partial X_1}\frac{\partial V}{\partial X_1} + \frac{\partial\Psi}{\partial X_2}\frac{\partial V}{\partial X_2}\right) d\Omega, \tag{2.4}$$

and collecting the diffusion and time derivative terms on the same side, the reaction term on the other side and applying the no flux boundary condition, Equation (2.3) can be re-arranged to give:

$$\int_\Omega D\left(\frac{\partial\Psi}{\partial X_1}\frac{\partial V}{\partial X_1} + \frac{\partial\Psi}{\partial X_2}\frac{\partial V}{\partial X_2}\right) d\Omega - \int_\Omega \Psi \frac{\partial V}{\partial t} d\Omega = \int_\Omega \Psi f d\Omega. \tag{2.5}$$

This makes it simpler to see the stiffness matrix and mass matrix terms which will be needed later in this chapter.

2.3.2 Discretising the solution

When using the FEM, the domain is discretised into elements (in this thesis these are triangles) and the unknown variable (V) is approximated by piecewise polynomial functions within each element. To approximate the unknown values within an element, linear Lagrange basis functions, Ψ, are introduced as follows:

$$\Psi_l = \alpha_l + \beta_l X_1 + \gamma_l X_2 \tag{2.6}$$

where $l = 1,2,3$ for a triangular element. The variable V is replaced with a piecewise approximation over each element as follows:

$$V(X_1,X_2) \approx \hat{V}(X_1,X_2) = \sum_{l=1}^{3} V_l \Psi_l(X_1,X_2), \tag{2.7}$$

For a triangle with vertices $l = 1,2,3$ the partial derivatives of Ψ_l with respect to X_1 and X_2 are:

$$\frac{\partial \Psi_l}{\partial X_1} = \beta_l, \tag{2.8}$$

and

$$\frac{\partial \Psi_l}{\partial X_2} = \gamma_l. \tag{2.9}$$

The derivatives of the piecewise approximation to V become:

$$\frac{\partial \hat{V}}{\partial X_1} = \sum_{l=1}^{3} V_l \frac{\partial \Psi_l}{\partial X_1} = \sum_{l=1}^{3} \beta_l V_l, \tag{2.10}$$

and

$$\frac{\partial \hat{V}}{\partial X_2} = \sum_{l=1}^{3} V_l \frac{\partial \Psi_l}{\partial X_2} == \sum_{l=1}^{3} \gamma_l V_l. \tag{2.11}$$

For a triangle with successive vertices of i, j, k, the values of α, β and γ [125, page 127] are given by:

$$\alpha_i = \frac{X_1^j X_2^k - X_1^k X_2^j}{2\triangle^e}, \tag{2.12}$$

$$\beta_i = \frac{X_2^j - X_2^k}{2\triangle^e}, \tag{2.13}$$

$$\gamma_i^e = \frac{X_1^k - X_1^j}{2\triangle^e}, \tag{2.14}$$

where $\triangle^e$ is the area of the triangular element.

In this thesis the Galerkin method [89] is implemented, which uses the same basis functions as the weight functions used in the weak form of the governing equation (see Equation (2.2)).

2.3.3 Stiffness and mass matrices

The weak formulation of the governing Equation (2.5), can be arranged as follows:

$$\mathbf{KV} - \mathbf{M}\frac{\partial \mathbf{V}}{\partial t} = \mathbf{f}, \tag{2.15}$$

where

$$K_{lm}^e = \int_{\Omega^e} D\left(\frac{\partial \Psi_l}{\partial X_1}\frac{\partial \Psi_m}{\partial X_1} + \frac{\partial \Psi_l}{\partial X_2}\frac{\partial \Psi_m}{\partial X_2}\right) d\Omega, \tag{2.16}$$

and

$$M_{lm}^e = \int_{\Omega^e} \Psi_l \Psi_m d\Omega, \tag{2.17}$$

and

$$f_l^e = \int_{\Omega^e} \Psi_l f d\Omega, \tag{2.18}$$

and where $\mathbf{V}$ is a vector of the unknown voltage variable and Ω^e is an element in the FEM discretisation. In this notation the matrix $\mathbf{K}$ is known as the stiffness matrix and the matrix $\mathbf{M}$ is known as the mass matrix. As Neumann 'no flux' boundary conditions are used in this work, the boundary terms equate to zero and are excluded from these equations.

The FEM method approximates the unknown variable within an element, by first calculating K_{lm}^e, M_{lm}^e and f_l^e using Equations (2.16), (2.17) and (2.18). Equations (2.16), (2.17) produce temporary matrices which are known as the local matrices, with $l, m = 1, 2, 3$. The local stiffness matrix components are given by Equation (2.16) and for a three node triangular element with vertices i, j, k this becomes the following 3 x 3 matrix:

$$\int_{\Omega^e} D \begin{bmatrix} \left[\left(\frac{\partial \Psi_i}{\partial X_1}\right)^2 + \left(\frac{\partial \Psi_i}{\partial X_2}\right)^2\right] & \left[\left(\frac{\partial \Psi_i}{\partial X_1}\frac{\partial \Psi_j}{\partial X_1}\right) + \left(\frac{\partial \Psi_i}{\partial X_2}\frac{\partial \Psi_j}{\partial X_2}\right)\right] & \left[\frac{\partial \Psi_i}{\partial X_1}\frac{\partial \Psi_k}{\partial X_1} + \frac{\partial \Psi_i}{\partial X_2}\frac{\partial \Psi_k}{\partial X_2}\right] \\ \left[\left(\frac{\partial \Psi_i}{\partial X_1}\frac{\partial \Psi_j}{\partial X_1}\right) + \left(\frac{\partial \Psi_i}{\partial X_2}\frac{\partial \Psi_j}{\partial X_2}\right)\right] & \left[\left(\frac{\partial \Psi_j}{\partial X_1}\right)^2 + \left(\frac{\partial \Psi_j}{\partial X_2}\right)^2\right] & \left[\frac{\partial \Psi_j}{\partial X_1}\frac{\partial \Psi_k}{\partial X_1} + \frac{\partial \Psi_j}{\partial X_2}\frac{\partial \Psi_k}{\partial X_2}\right] \\ \left[\left(\frac{\partial \Psi_i}{\partial X_1}\frac{\partial \Psi_k}{\partial X_1}\right) + \left(\frac{\partial \Psi_i}{\partial X_2}\frac{\partial \Psi_k}{\partial X_2}\right)\right] & \left[\frac{\partial \Psi_j}{\partial X_1}\frac{\partial \Psi_k}{\partial X_1} + \frac{\partial \Psi_j}{\partial X_2}\frac{\partial \Psi_k}{\partial X_2}\right] & \left[\left(\frac{\partial \Psi_k}{\partial X_1}\right)^2 + \left(\frac{\partial \Psi_k}{\partial X_2}\right)^2\right] \end{bmatrix} d\Omega, \tag{2.19}$$

and when the individual derivatives of Ψ_i, Ψ_j and Ψ_k are calculated this becomes:

$$K^e = \int_{\Omega^e} D \begin{bmatrix} (\beta_i)^2 + (\gamma_i)^2 & \beta_i\beta_j + \gamma_i\gamma_j & \beta_i\beta_k + \gamma_i\gamma_k \\ \beta_i\beta_j + \gamma_i\gamma_j & (\beta_j)^2 + (\gamma_j)^2 & \beta_j\beta_k + \gamma_j\gamma_k \\ \beta_i\beta_k + \gamma_i\gamma_k & \beta_j\beta_k + \gamma_j\gamma_k & (\beta_k)^2 + (\gamma_k)^2 \end{bmatrix} d\Omega. \tag{2.20}$$

Equation (2.20) gives the nine values for the local stiffness matrix and each of these need integrating over the element. As the weight and basis functions are linear, the integral simply becomes a matter of multiplying each of these nine calculations by the area of the element triangle. As the values of α, β and γ can be calculated using Equations (2.12), (2.13), and (2.14), the nine values of the local stiffness matrix can be directly calculated.

The FEM uses the values from temporary local stiffness matrices to build a global stiffness matrix. The global stiffness matrix is an n x n matrix, where n is the number of unknowns in the system and stores the combined values from all the local matrices. For a problem with one unknown variable (in this case voltage) the number of unknowns is equal to the number of nodes in the mesh, and the global matrix has one row for each unknown. Each node in the mesh is given a 'node index', which is then used to position the corresponding row in the global matrix .

The global stiffness matrix is built by processing each element and calculating a local stiffness matrix comprised of 3 x 3 entries, and these are distributed to the row and column within the global matrix that correspond to the 'node index' of the nodes within the element. The 'node index' can be an arbitrary value, for example the order created by the mesh generator, however it is more efficient to use a node ordering algorithm such as Reverse Cuthill McKee (see Section 2.5.5 for more details).

The global mass matrix $\mathbf{M}$ is built in a similar fashion, with a local matrix being generated and the contributions being distributed to the global mass matrix. The local mass matrix components are given by Equation (2.17) and for a three node triangular element with vertices i, j, k this becomes the following 3 x 3 matrix:

$$\int_{\Omega^e} D \begin{bmatrix} (\Psi_i)^2 & (\Psi_i\Psi_j) & (\Psi_i\Psi_k) \\ (\Psi_j\Psi_i) & (\Psi_j)^2 & (\Psi_j\Psi_k) \\ (\Psi_k\Psi_i) & (\Psi_k\Psi_j) & (\Psi_k)^2 \end{bmatrix} d\Omega, \tag{2.21}$$

The integrals for the local mass matrices could be calculated by substituting Equations (2.6), (2.12), (2.13) and (2.14) into (2.21) and then using quadrature. Alternatively, using induction it is possible to show the following for linear (hat) functions, over a triangle

(Ω^e) with nodes i, j, k:

$$\int_{\Omega^e} \Psi_i^m \Psi_j^n \Psi_k^p d\Omega = \frac{2m!n!p!}{(m+n+p+2)!} \triangle^e, \tag{2.22}$$

where $\triangle^e$ is the area of the triangle and m, n, and p are positive integers. This then gives:

$$\int_{\Omega^e} \Psi_i \Psi_j d\Omega = \begin{cases} \frac{1}{12} \triangle^e & when \quad i \neq j \\ \frac{1}{6} \triangle^e & when \quad i = j \end{cases}, \tag{2.23}$$

where $i, j = 1, 2, 3$, and local mass matrix can be written in full as:

$$\frac{\triangle^e}{12} \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}. \tag{2.24}$$

Equation (2.24) provides the 9 values for the local mass matrix of a linear triangular element. A global mass matrix can then be produced by processing each element in turn, building a local matrix and then distributing the values of the local matrix to the correct row and column in the global mass matrix. This utilises the node order property of the node in the same way as the global stiffness matrix.

2.4 Modelling fibre orientation

2.4.1 Anisotropic diffusion

To simplify the the electrophysiological model the diffusion term from Equation (2.1) is often set as isotropic [69], however cardiac tissue is anisotropic. To model this anisotropy, by setting diffusion at different rates in different directions, a diffusion tensor is used in Equation (2.1). The diffusion term in Equation (2.1) is represented by $\nabla \cdot (\mathbf{D} \nabla V)$. For anisotropic diffusion in two dimensions, $\mathbf{D}$ is a 2 x 2 matrix with the cross terms included, so:

$$\mathbf{D}\nabla V = \begin{pmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{pmatrix} \begin{pmatrix} \frac{\partial V}{\partial X_1} \\ \frac{\partial V}{\partial X_2} \end{pmatrix} = \begin{pmatrix} D_{11} \frac{\partial V}{\partial X_1} + D_{12} \frac{\partial V}{\partial X_2} \\ D_{12} \frac{\partial V}{\partial X_1} + D_{22} \frac{\partial V}{\partial X_2} \end{pmatrix}. \tag{2.25}$$

For isotropic diffusion the entries for the local stiffness matrix are given in Equation (2.16). However for the anisotropic case the diffusion coefficient D is replaced with a

diffusion tensor $\mathbf{D}$ and this expands as follows:

$$\begin{aligned}\nabla\cdot(\mathbf{D}\nabla V) &= \begin{pmatrix} \frac{\partial V}{\partial X_1} \\ \frac{\partial V}{\partial X_2} \end{pmatrix} \cdot \begin{pmatrix} D_{11}\frac{\partial V}{\partial X_1} + D_{12}\frac{\partial V}{\partial X_2} \\ D_{21}\frac{\partial V}{\partial X_1} + D_{22}\frac{\partial V}{\partial X_2} \end{pmatrix} \\ &= D_{11}\frac{\partial V}{\partial X_1}\frac{\partial V}{\partial X_1} + \frac{\partial V}{\partial X_1}\frac{\partial V}{\partial X_2}(D_{12}+D_{21}) + D_{22}\frac{\partial V}{\partial X_2}\frac{\partial V}{\partial X_2}. \end{aligned} \tag{2.26}$$

This has two extra terms for each entry in the stiffness matrix (as compared to the isotropic case). The local stiffness matrix, K^e_{lm}, with $l,m = 1,2,3$, will have the following additional terms:

$$\frac{\partial \Psi_l}{\partial X_1}\frac{\partial \Psi_m}{\partial X_2}(D_{12}+D_{21}) = \beta_l \gamma_m (D_{21}+D_{12}) \tag{2.27}$$

Using Equation (2.27), the local stiffness matrix equation for anisotropic diffusion is given by:

$$K^e_{lm} = \int_{\Omega^e} \left(D_{11}\frac{\partial \Psi_l}{\partial X_1}\frac{\partial \Psi_m}{\partial X_1} + D_{22}\frac{\partial \Psi_l}{\partial X_2}\frac{\partial \Psi_m}{\partial X_2} + \frac{\partial \Psi_l}{\partial X_1}\frac{\partial \Psi_m}{\partial X_2}(D_{12}+D_{21}) \right), \tag{2.28}$$

and it is straightforward to calculate all nine terms for the local stiffness matrix. This also means that if $\mathbf{D} = D\mathbf{I}$ then this will model the isotropic case as previously defined.

2.5 Solving the system

2.5.1 Spatial discretisation

The simulations were undertaken on two-dimensional rectangular domains. These domains were discretised into genuinely unstructured triangular elements using the 'Triangle' mesh generation package [99, 100]. An example mesh is show in Figure 2.2.

2.5.2 Time discretisation

When the governing equation for the cardiac electrophysiology (as given by Equation (2.1)) is approximated using the FEM, a system of equations with the following form is produced:

$$\mathbf{M}\frac{\mathrm{d}\mathbf{V}}{\mathrm{dt}} + \mathbf{K}\mathbf{V} = (\mathbf{f}+\mathbf{Q}), \tag{2.29}$$

where $\mathbf{M}$ is a mass matrix, $\mathbf{K}$ is the stiffness matrix, $\mathbf{V}$ is the vector of unknown voltages, $\mathbf{f}$ is the reaction term (caused by the ionic current model) and $\mathbf{Q}$ is the boundary term.

This equation can be solved using an explicit forward Euler method, where the deriva-

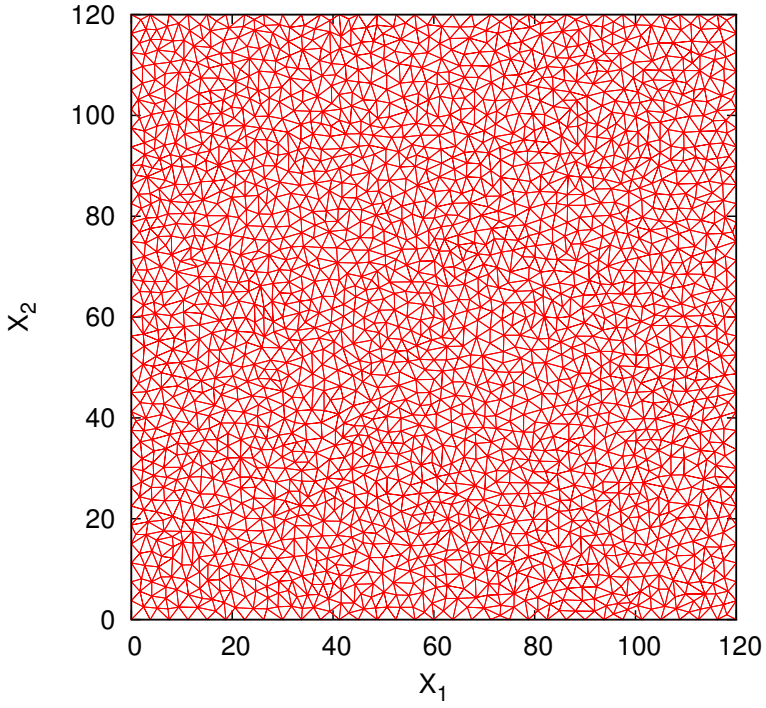


Figure 2.2: *An example unstructured mesh*

tive of $\mathbf{V}$ is approximated by:

$$\frac{\mathrm{d}\mathbf{V}}{\mathrm{dt}} \approx \frac{(\mathbf{V}_{n+1} - \mathbf{V}_n)}{dt}, \tag{2.30}$$

where dt is the time step and $\mathbf{V}_{n+1}$ is the unknown set of values of $\mathbf{V}_n$ at the the next time level $n+1$. With this technique Equation (2.29) can be re-arranged as follows:

$$\mathbf{M}\mathbf{V}_{n+1} = \mathbf{M}\mathbf{V}_n + (\mathbf{f} + \mathbf{Q} - \mathbf{K}\mathbf{V}_n)dt. \tag{2.31}$$

Equation (2.31) is of the general form $\mathbf{A}\mathbf{x} = \mathbf{b}$, where:

$$\begin{aligned} \mathbf{A} &= \mathbf{M} \\ \mathbf{x} &= \mathbf{V}_{n+1} \\ \mathbf{b} &= \mathbf{M}\mathbf{V}_n + (\mathbf{f} + \mathbf{Q} - \mathbf{K}\mathbf{V}_n)dt. \end{aligned} \tag{2.32}$$

and can be solved using a linear solver (in this thesis, the solver employed is GMRES as described in Section 2.5.4).

The stability of forward Euler places a constraint on dt. Specifically, for the diffusion part of the equation, the stability is governed by the ratio $r = D\frac{dt}{(dx)^2}$, where dx is the size

of the spatial discretisation. For the system to be stable this ratio must stay within the range of $0 < r \leq 0.5$ [103, Chapter 2]. This requirement means that each time the value of dx is halved, the value of dt needs to be quartered. To lift this restriction the theta method was implemented for the diffusion terms and this is discussed in Section 2.5.3 below.

2.5.3 Theta method

The forward Euler method described in Section 2.5.2 is an explicit method where $\mathbf{V}_{n+1}$ is calculated directly using known terms. The backward Euler method is an implicit method in which $\mathbf{V}_{n+1}$ is calculated by solving a system of equations. The theta method combines both implicit and explicit schemes and uses a constant, θ, to determine the contribution of the implicit and explicit techniques. Equation (2.15) has the right hand side term, $\mathbf{f}$, which relates to the reaction term caused by the ionic current model. In this thesis the reaction term ($\mathbf{f}$) is solved with an explicit forward Euler method, however the diffusion term is solved with the theta method. This technique is known as operator splitting. Equation (2.29) can then be reformatted with the theta method to give:

$$\mathbf{M}\frac{(\mathbf{V}_{n+1}-\mathbf{V}_n)}{dt}+\mathbf{K}(\theta\mathbf{V}_{n+1}+(1-\theta)\mathbf{V}_n)=(\mathbf{f}(\mathbf{V_n})+\mathbf{Q}), \tag{2.33}$$

which can be re-arranged to:

$$(\mathbf{M}+\mathbf{K}dt\,\theta)\mathbf{V}_{n+1}=(\mathbf{M}-\mathbf{K}dt(1-\theta))\mathbf{V}_n+dt(\mathbf{f}(\mathbf{V_n})+\mathbf{Q}). \tag{2.34}$$

If the value of θ is set to 0.5 this produces the Crank Nicolson method (which is unconditionally stable [111] for heat-diffusion equations and second order accurate in time), and Equation (2.34) becomes:

$$(\mathbf{M}+\frac{1}{2}\mathbf{K}dt)\mathbf{V}_{n+1}=(\mathbf{M}-\frac{1}{2}\mathbf{K}dt)\mathbf{V}_n+dt(\mathbf{f}(\mathbf{V_n})+\mathbf{Q}). \tag{2.35}$$

This provides an unconditionally stable scheme for time discretisation of the diffusion term which is utilised for the simulations undertaken in this thesis. In can be noted that, if Equation (2.35) is compared to the standard system of equations of the form $\mathbf{Ax}=\mathbf{b}$, then:

$$\begin{aligned}\mathbf{A} &= (\mathbf{M}+\frac{1}{2}\mathbf{K}dt)\\ \mathbf{x} &= \mathbf{V}_{n+1}\\ \mathbf{b} &= (\mathbf{M}-\frac{1}{2}\mathbf{K}dt)\mathbf{V}_n+dt(\mathbf{f}(\mathbf{V_n})+\mathbf{Q}).\end{aligned} \tag{2.36}$$

2.5.4 Solving with SPARSKIT GMRES solver

The governing Equation (2.1) is approximated using the FEM, resulting in the system of equations defined in Equation (2.15). The theta method is applied to form the system given in Equation (2.35). This is a linear system for the unknowns $\mathbf{V}_{n+1}$ and is of the general form $\mathbf{Ax} = \mathbf{b}$. As linear Lagrangian basis and weight functions are used, the FEM creates matrix rows where each row contains entries for the current node itself (on the diagonal) and then an entry for the other nodes this node is connected to. By the nature of the FEM discretisation this means that the matrix is very sparse, as a row will typically have fewer than ten entries (for a two-dimensional problem), however the number of columns in the matrix is equal to the number of unknowns. In the heart failure simulations undertaken in this thesis (see Chapter 7) the number of unknowns for the electrical system can be in excess of 300000. In such simulations only 0.0033% (approximately) of the matrix entries will be non-zero.

To efficiently store the matrix entries generated by the governing equations a sparse matrix storage format is employed. The format employed is the Compressed Sparse Row (CSR) format, whereby each matrix is represented by three vectors, namely a vector to store the non-zero values, a vector to store the column positions of the non-zero values and a vector to store the start positions of each row within the other two vectors.

SPARSKIT is a suite of software tools developed by Yousef Saad of University of Minnesota (see http://www-users.cs.umn.edu/ saad/software/SPARSKIT/) for the handling of sparse matrices. Included in SPARSKIT is an ILUT preconditioned GMRES linear solver (see Chapter 5 for more on preconditioning). The GMRES (Generalized minimal residual method [96]) is an iterative method for the solution of a system of linear equations.

The software developed in this thesis builds the required matrices (stored in the CSR sparse format) and vectors and then passes them over to SPARSKIT. In effect, the steps undertaken by SPARSKIT and its GMRES solver, for solving the electrical system, are treated as a “black box”.

2.5.5 Node reordering

The theta method described in Section 2.5.3 produces a system of equations of the general form $\mathbf{Ax} = \mathbf{b}$. The bandwidth of matrix $\mathbf{A}$ can have an effect on the performance when solving this system, with a lower bandwidth improving the performance. The bandwidth of a matrix is the maximum distance between diagonal entries and the column entries on the same row.

Cuthill and McKee [24] proposed a method for lowering the bandwidth of a matrix and this was refined by George and Liu in [35] by the reversal of the final order, thus creating the Reverse Cuthill McKee (RCM) method. The RCM method aims to reduce the bandwidth of the matrix by re-ordering the rows and columns of matrix **A**. This is done by 'walking' through the connections each node has to each other node and then ordering the nodes according to how they are connected to each other. An overview of the RCM method is given in Process 2.1.

Process 2.1 Overview of RCM Method

1: Firstly, it is necessary to store a set of node connectivities in the data structure. That is, each node in the mesh must have access to a list of the other nodes it is directly connected to. This is easily recorded when the edges of the elements are determined.
2: Prepare an empty queue Q and an empty result array R.
3: Select a node, N, from the mesh and place it in the first entry of Q
4: **repeat**
5: Take the first node (C) from Q and examine it.
6: If C hasn't previously been inserted in R, add C in the first free position of R.
7: Add to Q all the nodes connected to C that are not already in Q.
8: **until** Q is empty
9: Reverse the order of the entries in R.
10: Set the order (node index) of the node rows in the matrix to the order defined in R
Note: This algorithm slightly simplifies the classic RCM method, as it does not sort (by degree) the nodes connected to C, before they are added to the Q.

In this work, the RCM method is undertaken following each global mesh refinement or local adaptivity (see Section 4.1.1 and Chapter 6 for more information). This is because when new nodes are added or removed from a mesh the node connectivities change, and hence the node ordering produced by the RCM method needs re-calculating.

Table 2.1 illustrates the performance improvements achieved from using the RCM method for the electrical system solve. These tests undertaken on a 120 mm x 120 mm domain with varying mesh refinement and a time-step of $dt = 0.08$ ms. Figures 2.3 and 2.4 show the sparsity pattern for the matrix **A**, from Equation (2.36), with the RCM method disabled and enabled.

2.6 Electrical system results and validation

2.6.1 Stimulating the system

In this section the results of the electrical system will be considered and tests undertaken to investigate the convergence of the developed solver. However, firstly it is necessary to

Number Nodes	RCM On/Off	Ave. solve time	Percentage Saving
7927	Off	0.0478	-
7927	On	0.0375	27%
31453	Off	0.1967	-
31453	On	0.1588	24%
125305	Off	0.6741	-
125305	On	0.5043	34%

Table 2.1: *Performance improvement with RCM method. Tests undertaken on a 120 mm x 120 mm domain, with time-step* $dt = 0.08$ *ms.*

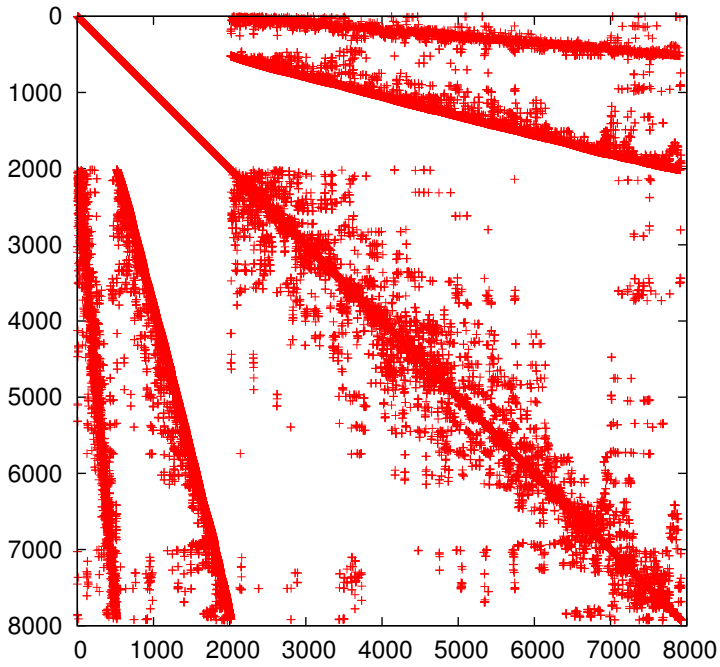


Figure 2.3: *Sparsity pattern for matrix* $\mathbf{A}$ *from Equation (2.36) with RCM Method disabled, with 7927 nodes.*

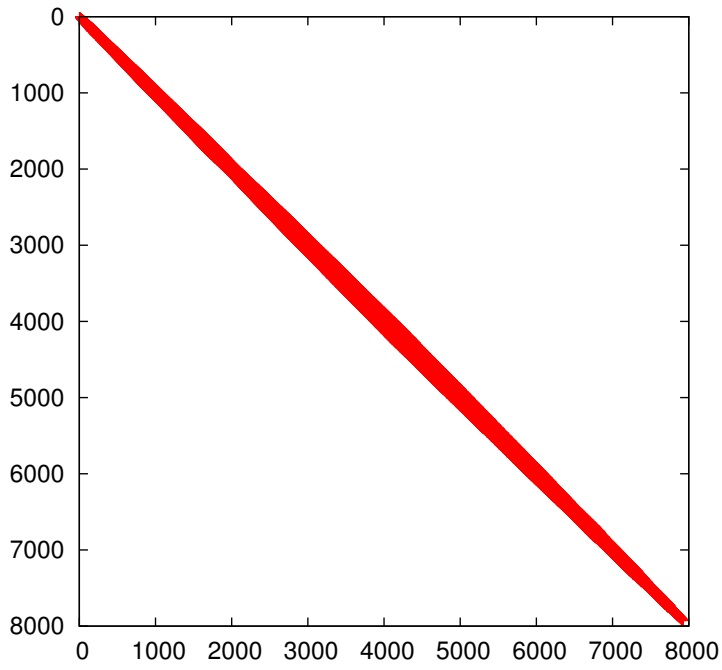


Figure 2.4: *Sparsity pattern for matrix* $\mathbf{A}$ *from Equation (2.36) with RCM Method enabled, with 7927 nodes*

understand how the electrical wave is instigated.

Equation (2.1) has a term for the stimulation of the system, I_{stim}. To start an electrical wave it is necessary to provide a stimulus to an area of the domain for a given period of time. In this thesis this is typically undertaken by adding a stimulus voltage of 52 mV to the value calculated within the TP06 ionic model for the transmembrane voltage, to an area at the left edge of the domain. For example, the stimulus may be applied where $X_1 < 5$ mm and $t < 1$ ms. Once the electrical wave has been formed it will progress across the domain without any further external stimulation.

The initial excitation of an area of elements in the domain means that if different mesh refinements are used, there will be slight variances in the starting area of the wave, due to the different triangle sizes and positions. These small changes in the starting conditions introduce errors when comparing the solution from different levels of mesh refinements.

To remove the variation in the element size (and hence the initial excitation area), it is possible to only stimulate the nodes on the left edge of the domain. However this results in the wave dying out for the higher levels of global mesh refinement. To prevent the wave dying out, the period of time the initial stimulus is applied can be increased, however this distorts the wave form, and so this technique was rejected.

To create a spiral wave in the domain, a plane wave is formed by stimulating the left edge of the domain ($X_1 = 0$) for $t < 1$ ms. The wave is allowed to progress across the domain and then after a given period, the transmembrane voltage for this wave is then split and 'clamped' for a further period of time. For example, between $t = 115$ ms and $t = 150$ ms, the voltage in the top half of the domain (where $X_2 > 60$ mm in a 120 $\times$ 120 mm domain) is artificially fixed to -86.2 mV. After $t = 150$ ms the 'clamping' is removed and the wave is allowed to progress as normal. This technique causes the wave to collapse back onto itself and form a self sustaining spiral.

2.6.2 Solution convergence with varying time steps

To consider the convergence of the electrical solver, tests were undertaken with the time-step varied. These tests were undertaken on a 1.8 mm x 30 mm domain, with 5113 nodes and 9728 elements, giving an average element area of 0.0056 mm^2 (with an approximate edge length of 0.1 mm). The diffusion was set to be isotropic with D=0.154. The left edge ($X_1 < 1$) of the domain was stimulated for $t < 2.56$ ms. The time step (dt) was progressively reduced from $dt = 0.64$ ms to $dt = 0.0025$ ms, and the time discretisation scheme described in Section 2.5.3 was undertaken. The solutions were compared to the result using the $dt = 0.0025$ ms solution to give an estimate of the error. The results from these tests can be seen in Table 2.2, Figure 2.5 and Figure 2.6. The wave speed was measured by recording the time that two nodes in the interior of the domain became excited. The distance between these nodes, in the X_1 direction, was calculated and this enabled wave speed in the X_1 direction to be determined.

These tests demonstrate that as the time step is reduced the measured wave speed converges linearly. The Crank Nicolson method is second order and this is used for the diffusion term, however the reaction term is solved with a first order explicit forward Euler method. The reaction term is non-linear and the value being measured is the speed of the wave, which is a derived term. Therefore a linear convergence rate is not unexpected. Also, as can be seen in Figure 2.5 the speed of the wave initially overshoots the correct value (for example at t=0.04 m/s) and then converges back to the correct solution. This explains why the t=0.08 m/s gives a very low error.

It can be seen that as the time step is reduced the wave speed increases and this is the opposite effect to that seen in the spatial convergence in the next section.

Time step(ms)	Speed (m/s)	Error
0.6400	0.5551	0.2890
0.3200	0.7174	0.1267
0.1600	0.8040	0.0401
0.0800	0.8440	0.0001
0.0400	0.8498	0.0056
0.0200	0.8469	0.0028
0.0100	0.8455	0.0013
0.0050	0.8447	0.0006
0.0025	0.8441	-

Table 2.2: *Electrical wave speed with varying time steps. Tests were undertaken on a 1.8 mm x 30 mm domain, with 5113 nodes and 9728 elements. The diffusion coefficient was set to be isotropic with D=0.154 and the left edge ($X_1 < 1$) of the domain was stimulated for $t < 2.56$ ms. The error is compared to the solution with time-step of 0.0025 ms*

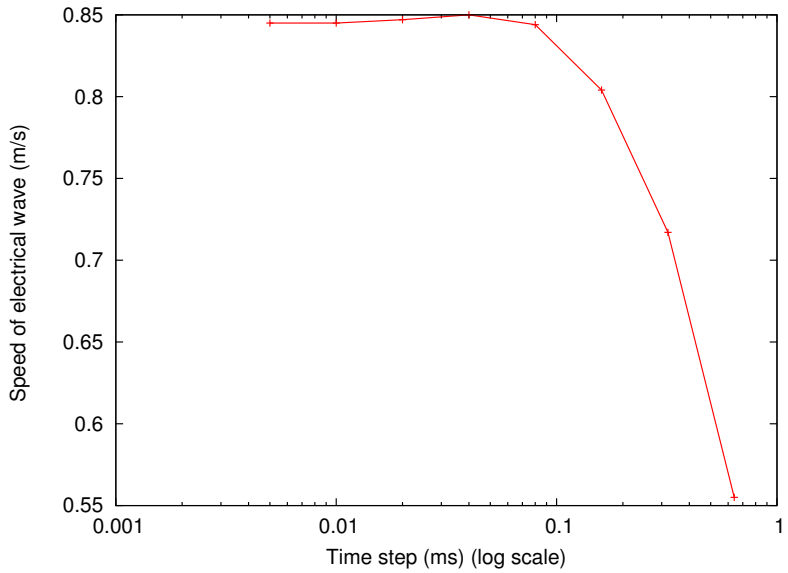


Figure 2.5: *Speed of electrical wave with varying time steps. Tests were undertaken on a 1.8 mm x 30 mm domain, with 5113 nodes and 9728 elements. The diffusion coefficient was set to be isotropic with D=0.154 and the left edge ($X_1 < 1$) of the domain was stimulated for $t < 2.56$ ms.*

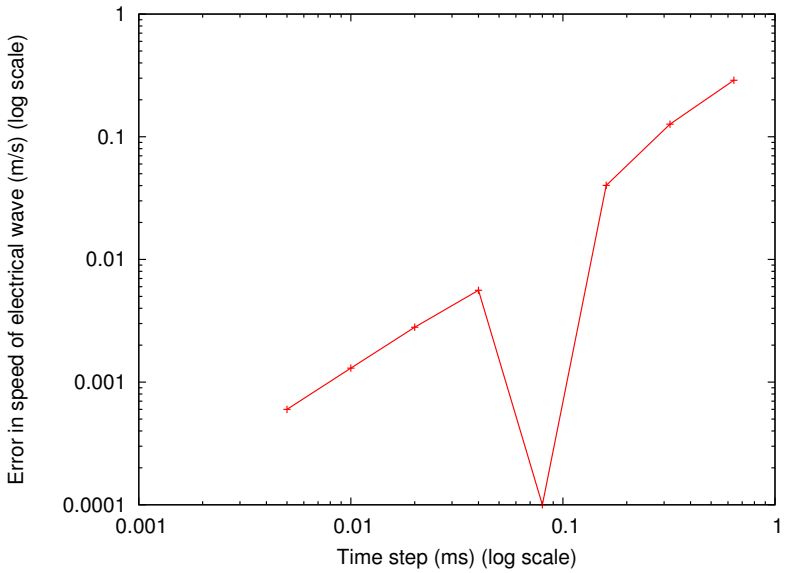


Figure 2.6: *Error in electrical wave speed with varying time steps. Tests were undertaken on a 1.8 mm x 30 mm domain, with 5113 nodes and 9728 elements. The diffusion coefficient was set to be isotropic with D=0.154 and the left edge ($X_1 < 1$) of the domain was stimulated for $t < 2.56$ ms. The error is compared to the solution with time-step of 0.0025 ms*

2.6.3 Solution convergence with varying spatial terms

In order to demonstrate the spatial convergence, tests were undertaken on a 1.8 mm x 30 mm domain, with a varying number of nodes and elements. The diffusion was set to be isotropic and the diffusion coefficient set to 0.154. In these tests the time step was fixed at $dt = 0.08$ ms in all simulations. The mesh data and results of the tests can be seen in Table 2.3, and these meshes were produced by progressively refining the coarsest mesh. The coarsest mesh has an average element area of 0.0888 mm^2, and an approximate edge length of 0.42 mm (and the subsequent meshes halve this length). A graph of the number of nodes against the wave speed can be seen in Figure 2.7. To consider the convergence of these results a test was undertaken on a mesh with 311337 nodes and 622598 elements. This gave an average element area of 0.0000867 mm^2 (with an approximate edge length of 0.013 mm). These were run with a time step of $dt = 0.005$ ms and used as a comparison to estimate errors. It generated a wave speed of 0.816 m/s. When considering these it is important to consider the average element mesh size. For the comparative solution the average element area is 0.000347 mm^2, which if used on a mesh of 120x120 mm (as for the heart failure simulations in Chapter 7) there would require 41498559 elements.

Similar tests were undertaken in which the time step and spatial step were varied

Nodes	Elements	Average Element Area (mm^2)	Time step (ms)	Speed (m/s)	Error
367	608	8.88×10^{-2}	0.080	1.0363	0.220177
1341	2432	2.22×10^{-2}	0.080	0.9055	0.089385
5113	9728	5.55×10^{-3}	0.080	0.8440	0.027927
19953	38912	1.39×10^{-3}	0.080	0.8217	0.005617
78817	155648	3.47×10^{-4}	0.080	0.8145	0.001559
311337	622592	8.67×10^{-5}	0.005	0.8161	-

Table 2.3: *Electrical wave speed with increasing mesh refinement, with error compared to solution with $dt = 0.005$ ms and 622592 elements. Tests were undertaken on a 1.8 mm x 30 mm domain, the diffusion was isotropic and D set to 0.154.*

Nodes	Elements	Average Element Area (mm^2)	Time step (ms)	Speed (m/s)	Error
367	608	8.88×10^{-2}	0.160	1.0138	0.197649
1341	2432	2.22×10^{-2}	0.080	0.9055	0.089385
5113	9728	5.55×10^{-3}	0.040	0.8479	0.031763
19953	38912	1.39×10^{-3}	0.020	0.8263	0.010167
78817	155648	3.47×10^{-4}	0.010	0.8145	0.001559
311337	622592	8.67×10^{-5}	0.005	0.8161	-

Table 2.4: *Electrical wave speed with increasing mesh refinement and halving time steps. Error compared to solution to with $dt = 0.005$ ms and 622592 elements. Tests were undertaken on a 1.8 mm $\times$ 30 mm domain, the diffusion was isotropic and D set to 0.154.*

together. In these tests as the number of elements was refined one level (hence creating 4 times as many elements), the time step was halved. As with the previous tests these were undertaken on a 1.8 mm x 30 mm domain, with a varying number of nodes and elements. The diffusion was set to isotropic and the diffusion coefficient set to 0.154. The data for the tests can be seen in Table 2.4.

Figure 2.7 shows the converging wave speed for the both the fixed time step tests and the varying time step tests, and these show the solution converging to a speed of 0.816 m/s. Figure 2.7 shows how the error reduces with the two sets of tests. These tables demonstrate that, with a fixed time step, as the spatial step is halved the error reduces by between 2.4 and 4.9 times, showing a convergence approaching quadratic.

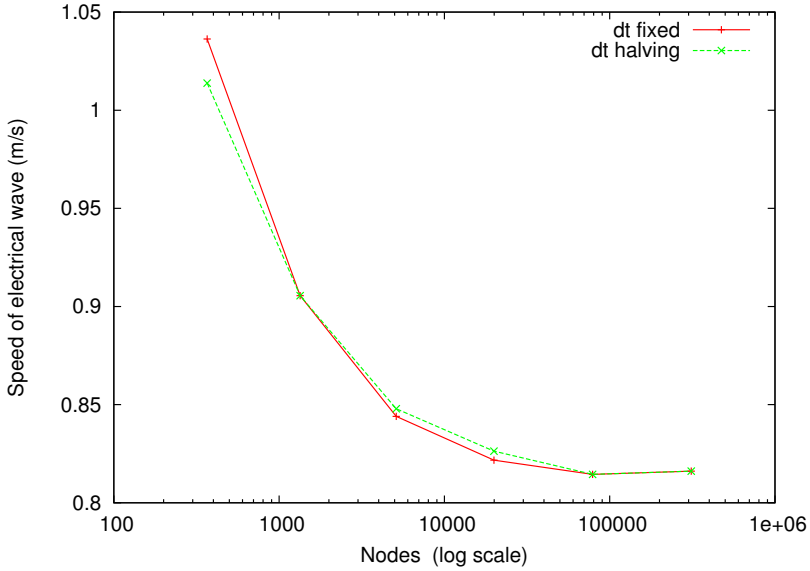


Figure 2.7: *Comparison of electrical wave speed across the domain with increasing mesh refinement and either fixed ($dt = 0.08$ ms) or halving time steps. Tests were undertaken on a 1.8 mm x 30 mm domain, the diffusion was isotropic and set to 0.154.*

2.6.4 Wave considerations

The profile of the cardiac action potential has a rapid change in voltage when the cell depolarizes. This causes a very steep (almost vertical) wave front as can be seen in Figure 2.8, where the wave is moving from left to right. The FEM (and other methods that involve discretising a continuum) approximates the wave across the domain, however to accurately model a rapid change in a variable you need to have a finely discretised domain. This can also be illustrated by considering the ability of a coarse mesh to properly resolve the features of the wave front. Figure 2.9 shows the wave front within a 30 mm $\times$ 30 mm domain, with 22016 elements (coarse) and 352256 elements (fine). As can be seen in this figure the coarse mesh cannot fully resolve the curve of the wave 'spike'.

It has been noted in previously published work (see [108, Table 2]) that the size of the spatial and temporal discretisation has an effect on the speed of the electrical wave across the domain and this has been further verified here.

2.7 Spiral wave comparison

In this section the electrical model is compared the results seen in [110]. The simulations are run on a square domain which occupies the region of $0 < X_1, X_2 < 120$ mm, divided

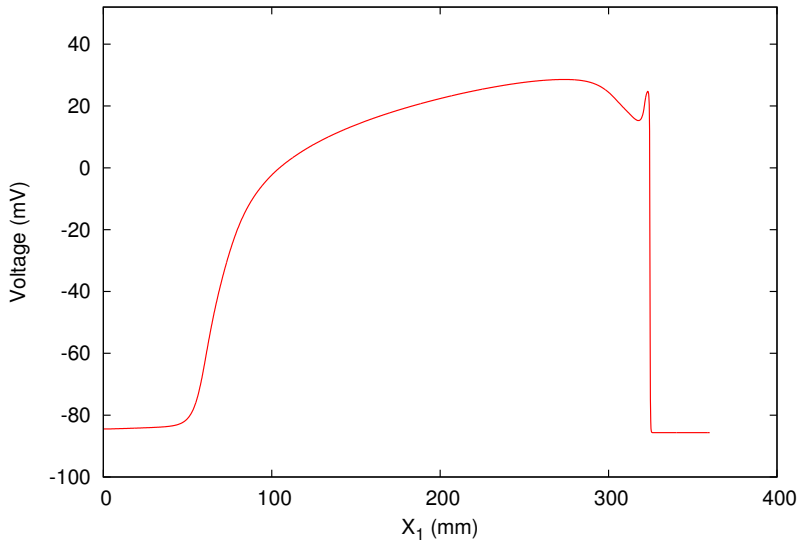


Figure 2.8: *Cardiac transmembrane voltage simulated with TP06 ionic current model*

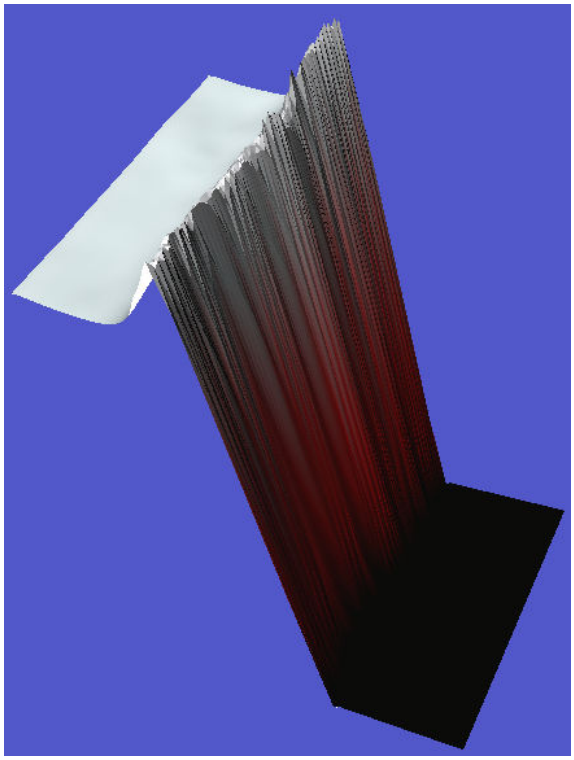
into an unstructured mesh of 634368 triangles, using 318065 nodes. This gives an approximate element edge length of 0.21 mm. The boundaries of the domain were set with Neumann no-flux boundary conditions. The electrical solve uses semi-implicit time discretisation (as described in Section 2.5.2 above), with a fixed time step (dt) of 0.08 ms. The diffusion tensor, $\mathbf{D}$, was set to

$$\mathbf{D} = \begin{bmatrix} 0.1540 & 0.01711 \\ 0.01711 & 0.1540 \end{bmatrix}, \tag{2.37}$$

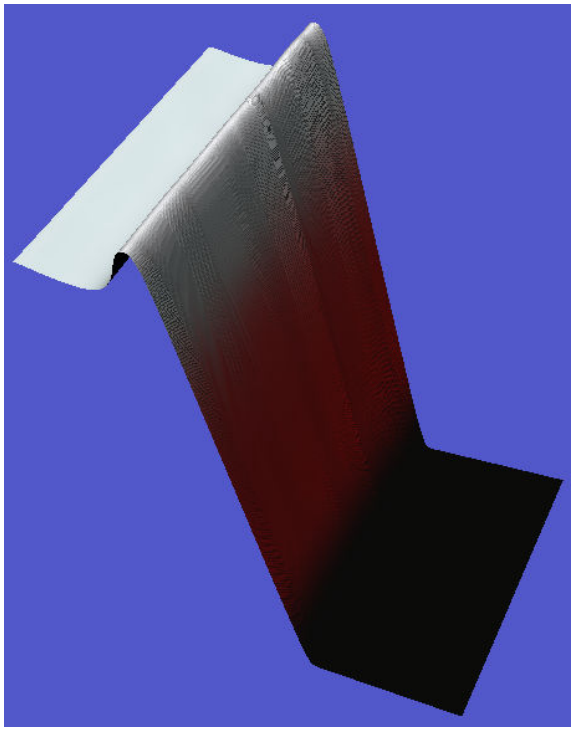
which gave conduction velocity of 0.68 m/s. As an anisotropic diffusion tensor $\mathbf{D}$ is used, the spiral wave formed should be elongated in the direction of the greater conduction speed, forming spirals which could be described as elliptical.

Simulations were undertaken using dynamic restitution slopes of 1.1, 1.4 and 1.8 (as defined in Table 2 of [110]). By changing the restitution slope the action potential profile is altered, with the higher valued slopes having a higher plateau and a longer wave length. The results of these tests can be seen in Figure 2.10, which displays the expected action potential profile for each of the three restitution curves for a given node in the mesh. That is, the transmembrane voltage for the restitution slopes of 1.4 and 1.8 have a higher value during the plateau phase and stay excited for longer than the 1.1 slope.

Tests were undertaken to produce a stable spiral wave with a restitution slope of 1.1. This was achieved by stimulating the left edge of the domain ($X_1 = 0$) for $t < 1$ ms, to form



(a) Coarse - 14952 elements



(b) Fine - 233472 elements

Figure 2.9: *Comparison of electrical wave front modelled on a coarse mesh and fine mesh, with average element areas of 0.0616 mm^2 (14592 elements, approximate edge length 0.35 mm) and 0.00385 mm^2 (233472 elements, approximate edge length 0.088 mm) respectively. Undertaken on 30 mm x 30 mm domain, at t=16 ms, with dt = 0.08. The colour scale for these images is given in Figure 2.11(c).*

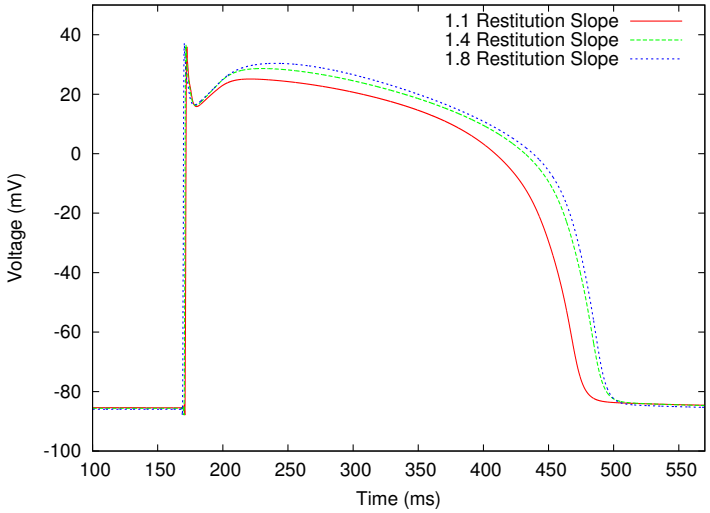


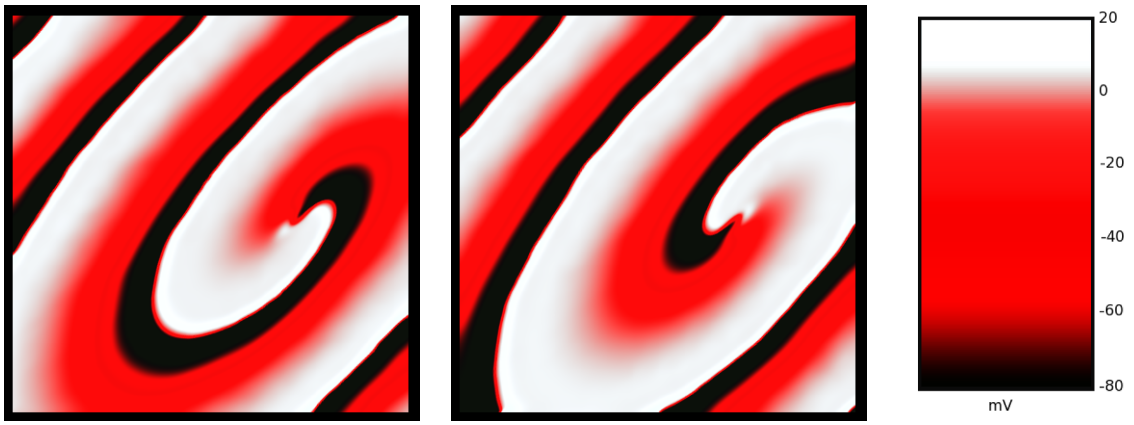
Figure 2.10: *Transmembrane voltage for a single node with restitution slopes of 1.1, 1.4 and 1.8.*

a plane wave and then splitting this wave at $t = 115$ ms. In the wave splitting technique the voltage in the lower half of the domain (perpendicular to the wave direction) is fixed at the resting potential for a period of time (in this case 35 ms). This resulted in a stable spiral wave over time and this can be seen in Figure 2.11. The colour scale for the output in this section is given in Figure 2.11(c).

The next set of validation tests were undertaken to consider the spiral wave stability for the three restitution slopes. The simulations were run for 5000 ms with the restitution slope set to 1.1, to ensure a stable spiral wave was formed, and then a new restitution slope of 1.4 or 1.8 introduced for a further 5000 ms. Figure 2.12 shows the results when the restitution slope was changed to 1.4. Figure 2.13 shows the results when the restitution slope was changed to 1.8. These results were then compared to the bottom three rows of the first column of Figure 7 of [110] and found to produce the same behaviour. Specifically, for restitution slopes of 1.1 and 1.4 a stable spiral wave is maintained over time, however for a restitution slope of 1.8 the spiral wave breaks up into a chaotic state.

2.8 Conclusion

This chapter has explained the process of taking a mathematical model of the cardiac electrophysiology and approximating it with the FEM. A number of topics have been covered in this process, including adding anisotropic diffusion, using a semi-implicit time stepping scheme, storing in an efficient sparse storage format, ordering the nodes to reduce

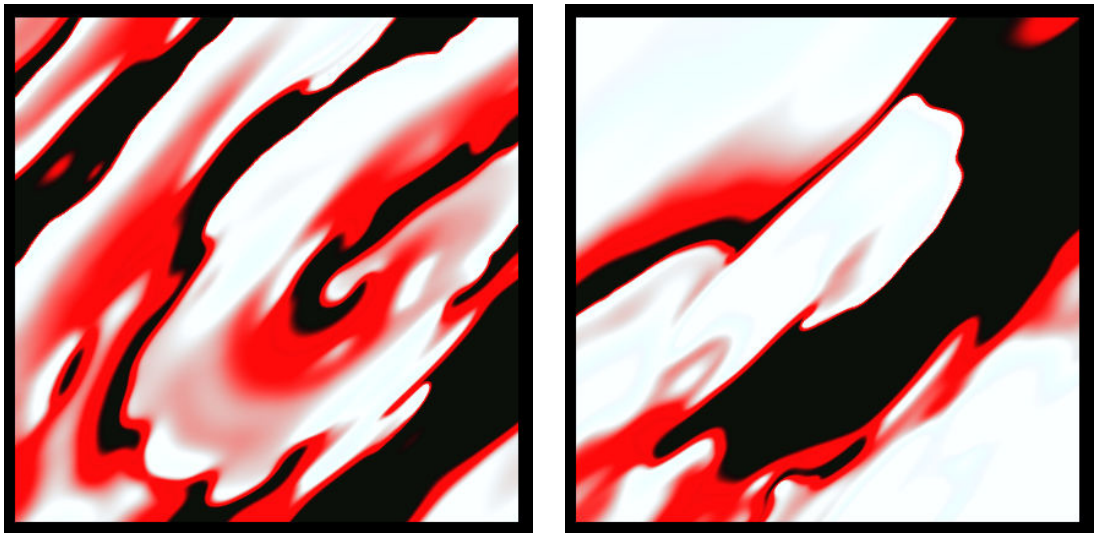


(a) t=6000 ms (b) t=8000 ms (c) Scale

Figure 2.11: *Stable spiral wave for TP06 model and restitution slope of 1.1. In a domain where* $0 < X_1, X_2 < 120$ *mm, divided into 634,368 elements and using 318,065 nodes. A fixed time step (dt) of* 0.08 *ms and the diffusion tensor,* $\mathbf{D}$*, set as per Equation (2.37).*

(a) t=6000 ms (b) t=8000 ms

Figure 2.12: *Stable spiral wave for TP06 model and restitution slope of 1.4. In a domain where* $0 < X_1, X_2 < 120$ *mm, divided into 634,368 elements and using 318,065 nodes. A fixed time step (dt) of* 0.08 *ms and the diffusion tensor,* $\mathbf{D}$*, set as per Equation (2.37).*



(a) t=6000 ms (b) t=8000 ms

Figure 2.13: *Break-up of spiral wave for TP06 model and restitution slope of 1.8. In a domain where* $0 < X_1, X_2 < 120$ *mm, divided into 634,368 elements and using 318,065 nodes. A fixed time step (dt) of* 0.08 *ms and the diffusion tensor,* $\mathbf{D}$*, set as per Equation (2.37).*

the matrices' bandwidths, and solving with an external sparse solver.

Spatial and temporal convergence tests have been undertaken to demonstrate the convergence of the solution. Simulations have been undertaken to validate the qualitative behaviour of the solver again other published work [110], specifically the stability of the spiral waves under varying restitution slopes. Figure 2.14 shows a spiral wave formed using the solver and this figure has the transmembrane voltage plotted on the z-axis to illustrate the features of the wave as it spirals.

The electrophysiological solver forms a basis for the production of a coupled electromechanical solver when combined with the cardiac mechanical system discussed in Chapter 3 and the coupling techniques described in Chapter 4. Anisotropic diffusion has been implemented and this enables the simulation of fibre orientation within the domain.

The convergence testing undertaken in this chapter highlights the need to model the electrophysiology on a finite element mesh with a fine resolution. As the element sizes decrease the speed of the electrical wave converges, however convergence is not reached until the average element areas are beneath $0.0014\,\text{mm}^2$ (approximate edge length of 0.05 mm). The qualitative behaviour of the electrical wave can produce the expected results with a coarser mesh than this, however the wave speed may not be close to the converged value. Due to the requirements for a very refined mesh, the electrical system

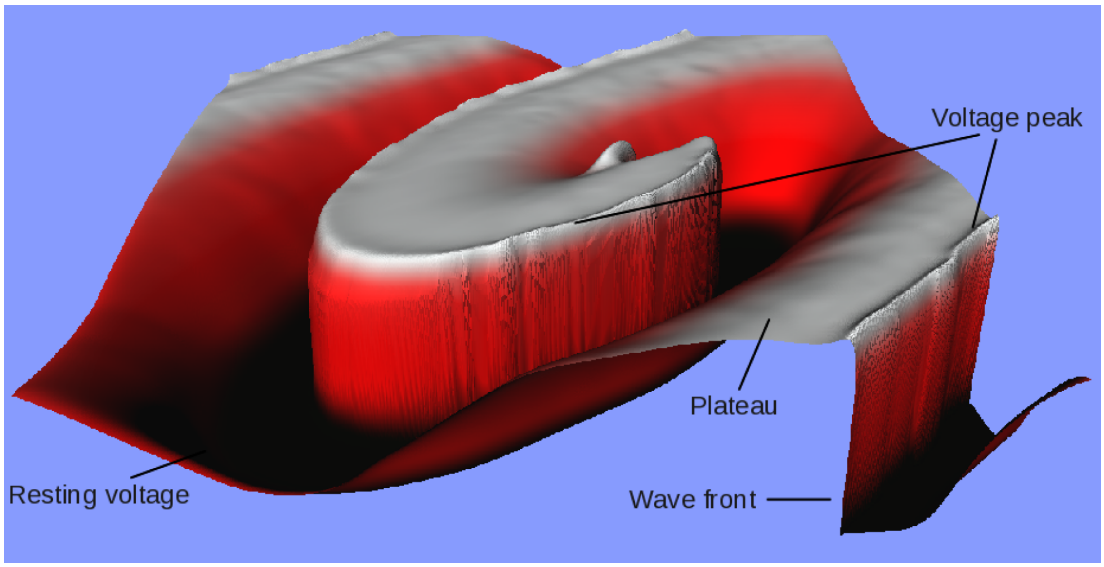


Figure 2.14: *An example 3D spiral wave with the voltage plotted on the z-axis. In a domain where* $0 < X_1, X_2 < 120$ *mm, divided into 634,368 elements and using 318,065 nodes. A fixed time step (dt) of* 0.08 *ms and the diffusion tensor,* $\mathbf{D}$*, set as per Equation (2.37).*

may benefit from an adaptive mesh refinement strategy (AMR) and this is discussed in detail in Chapter 6.

Finally, the TP06 ionic current model is sufficiently complex to allow changes in biological features to be modelled and this enables the heart failure simulations covered in Chapter 7.

Chapter 3

Cardiac mechanics

3.1 Introduction

This chapter describes the mathematical and computational models of cardiac mechanical activity employed in this thesis. Specifically, the chapter describes a mathematical model of cardiac tissue deformation, how this is approximated by the finite element method, the details of formulating a model with incompressible tissue, and how the resulting complex non-linear system of equations is solved.

For the mechanical model to deform it requires an electrical wave to stimulate the contraction. This electrical model is described in Chapter 2, and the coupling of the electrical activity to the mechanical deformation is covered in Chapter 4.

To model the contraction, deformation and accompanying forces it is necessary to understand stress mechanics and finite deformation elasticity theory. This is covered in detail in Chapter 12 of Computational Biology of the Heart [81]. The sections below summarise how to develop a mathematical model of cardiac deformation and the computational methods used to solve this.

3.2 Cardiac modelling concepts

3.2.1 Kinematics

In the mechanical model it is important to understand the relationship between points in the undeformed reference state and their deformed state (during contraction). To this end, the deformed coordinates (x_i) are written as functions of the reference coordinates, i.e. in three space dimensions:

$$\begin{aligned} x_1 &= f_1(X_1, X_2, X_3) \\ x_2 &= f_2(X_1, X_2, X_3) \\ x_3 &= f_3(X_1, X_2, X_3). \end{aligned} \tag{3.1}$$

Equation (3.1) is defined in [47], which further states that the kinematics of the deforming body are defined by the relationship between the reference coordinates (X_M) and the deformed coordinates (x_i), by a deformation gradient tensor $\mathbf{F}$. The components of $\mathbf{F}$ are defined by

$$F_M^i = \frac{\partial x_i}{\partial X_M}, \tag{3.2}$$

for $i, M = 1, 2, 3$.

3.2.2 Strain and stress

Strain is defined as the measure of deformation of a body and, in continuum mechanics, stress is a measure of the internal forces acting within a deformable body. To relate stress and strain to each other a constitutive law is needed and Section 3.2.3 describes a cardiac tissue constitutive model. The forces created by the changing ion concentrations caused by the electrical activity are modelled as a stress and the constitutive law for that material then relates this to the strain in the material. The strain is then used to calculate the deformations within the body.

The deformation gradient tensor $\mathbf{F}$ for two dimensions is:

$$\mathbf{F} = \begin{pmatrix} \frac{\partial x_1}{\partial X_1} & \frac{\partial x_1}{\partial X_2} & 0 \\ \frac{\partial x_2}{\partial X_1} & \frac{\partial x_2}{\partial X_2} & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{3.3}$$

$\mathbf{F}$ is also used to calculate the Cauchy-Green right deformation tensor $\mathbf{C}$. Cauchy-Green deformation tensors are used in continuum mechanics, and if the material is thought to

be composed of infinitesimally thin fibres then this tensor quantifies the squared length of these fibres in the deformed configuration. The invariants of $\mathbf{C}$ are often used in the calculation of strain energy density functions, and this is covered further in Section 3.2.3. The Cauchy-Green right deformation tensor is defined as $\mathbf{C} = \mathbf{F}^T\mathbf{F}$, and in the two dimensional case this becomes:

$$\begin{aligned}\mathbf{C} &= \begin{pmatrix} \frac{\partial x_1}{\partial X_1} & \frac{\partial x_2}{\partial X_1} & 0 \\ \frac{\partial x_1}{\partial X_2} & \frac{\partial x_2}{\partial X_2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{\partial x_1}{\partial X_1} & \frac{\partial x_1}{\partial X_2} & 0 \\ \frac{\partial x_2}{\partial X_1} & \frac{\partial x_2}{\partial X_2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} \frac{\partial x_1}{\partial X_1}\frac{\partial x_1}{\partial X_1} + \frac{\partial x_2}{\partial X_1}\frac{\partial x_2}{\partial X_1}, & \frac{\partial x_1}{\partial X_1}\frac{\partial x_1}{\partial X_2} + \frac{\partial x_2}{\partial X_1}\frac{\partial x_2}{\partial X_2} & 0 \\ \frac{\partial x_1}{\partial X_1}\frac{\partial x_1}{\partial X_2} + \frac{\partial x_2}{\partial X_1}\frac{\partial x_2}{\partial X_2}, & \frac{\partial x_1}{\partial X_2}\frac{\partial x_1}{\partial X_2} + \frac{\partial x_2}{\partial X_2}\frac{\partial x_2}{\partial X_2} & 0 \\ 0 & 0 & 1 \end{pmatrix}. \end{aligned} \tag{3.4}$$

The Lagrangian Green strain tensor is given by:

$$\mathbf{E} = \frac{1}{2}(\mathbf{C} - \mathbf{I}), \tag{3.5}$$

and this is a measure of how much the Cauchy-Green tensor differs from the identity matrix. It is used in the derivation of the Second Piola Kirchhoff tensor described in Section 3.2.4.

3.2.3 Constitutive equations and strain energy function

Constitutive equations provide a means of relating stress and strain, so that the contractile forces instigated by the electrical activity can be converted into a measure of deformation of the muscle, that is, the strain of the muscle. Typically, constitutive equations are derived from experimental observation.

In describing their constitutive law, which is used in this thesis, Hunter, Nash and Sands in [81], define two rules governing non-linear elastic materials. These relate to rigid motion and material invariance. By postulating these two governing rules and introducing a scalar strain energy function W, this is dependent only on the Cauchy-Green deformation tensor $\mathbf{C}$, or Lagrangian Green strain tensor $\mathbf{E}$ [39].

In common with [69], a strain energy function (W) which describes certain types of rubbers and silicone gels, known as Mooney-Rivlin materials, is used in this thesis. A Mooney-Rivlin solid is a hyper-elastic material model where the strain energy density function, W, is a linear combination of two invariants of the Cauchy-Green deformation tensor. The model was proposed by Melvin Mooney and Ronald Rivlin [66, 91]. The

expansion of the function W is covered in detail in Section 3.2.7 below. It is noted in [81], that "myocardial tissue has several features which make the task of constitutive law formulation difficult. First, in common with all soft biological tissues, the stress-strain behaviour is highly non-linear and quite anisotropic". Also in [69] they comment that "experimental evidence has shown that cardiac tissue exhibits different response along various material axes". This is referenced from Smaill and Hunter [36]. Using an isotropic strain energy function (W) simplifies the model.

Other approaches to defining the strain energy function are used, by others. Notably the "pole zero" strain energy function is commonly used [68, 71] in modelling cardiac mechanics. The pole zero strain energy function is derived from the observation that the stress-strain behaviour along one axis is very nearly independent of the degree of lateral stretch [102]. The strain energy function can then be expressed in terms of the stretch along each of the material axes. Another approach, as used in [54], is to assume that the passive myocardium can be modelled as an incompressible hyperelastic material, as described in [41], with a transversely isotropic mechanical response. This technique uses strains in the fibre, transverse (within the wall-plane perpendicular to the fibre direction) and radial (perpendicular to the wall-plane) directions.

To simulate fibre orientation it was required to be able to model tension acting anisotropically. Rather than introducing a fully anisotropic strain energy function it was decided to use the isotropic Mooney-Rivlin strain-energy function defined in Equation (3.14), but set the resultant force to act parallel to the theoretical fibre direction (this is covered in Section 3.2.5).

3.2.4 Second Piola-Kirchhoff tensor

For cardiac tissue, the Second Piola-Kirchhoff stress tensor, which expresses the stress relative to the reference configuration, is comprised of an elastic component and a biochemically generated component [82]. The elastic component is analogous to the passive stress that the tissue is under when no external forces are applied. The biochemically generated component is caused by the tension induced by the electrical wave passing through the tissue. The generation of this tension is described in more detail in Section 4.2.

The Piola-Kirchhoff stress tensors are used for finite deformations and they express the stress relative to the reference coordinates X_m. This is in contrast to the Cauchy stress tensor which expresses the stress relative to the present configuration. The Second Piola Kirchhoff stress tensor is given by:

$$T_{MN} = T_{MN}^{elast} + T_{MN}^{bioch}, \tag{3.6}$$

where $N, M = 1, 2$ (for two dimensions) and the elastic components of T_{MN} are given by:

$$T_{MN}^{elast} = \frac{1}{2}\left(\frac{\partial W}{\partial E_{MN}} + \frac{\partial W}{\partial E_{NM}}\right) - pC_{MN}^{-1}, \tag{3.7}$$

where W is a suitable strain energy function, $\mathbf{E}$ the Lagrangian Green's strain tensor (given in Equation (3.5)), $\mathbf{C}$ is the Cauchy-Green deformation tensor (given in Equation (3.4)) and p (which is referred to as the pressure) is a Lagrange multiplier used to enforce incompressibility. In common with other authors [47,68] the tissue is set as incompressible. Incompressibility is enforced by setting the determinant of the deformation gradient tensor to be equal to 1 (as per [82]):

$$det(\mathbf{F}) = 1. \tag{3.8}$$

The T_{MN}^{bioch} components of T_{MN} are given by:

$$T_{MN}^{bioch} = T_a C_{MN}^{-1}, \tag{3.9}$$

where T_a is the active tension generated from the electrical system. This is explained further in Equations (4.2) and (4.3) in Chapter 4.

3.2.5 Modelling fibre orientation

Cardiac tissue is composed of fibres and sheets and the electrical wave travels faster in parallel to the sheets than transversely. It has also been shown experimentally [102] that the tensile force generated along the cardiac fibres is much greater than the tension across them.

In Section 2.4 the ability to model anisotropic diffusion in the electrical model was introduced to simulate the faster wave speed along the fibres. Although the strain energy function used is isotropic, the active tension component can be set to act in only one direction, as described in [82], and this ensures the force acts along the direction of fibre orientation [68, 120].

The active tension component is set using the technique described in [82] and results in:

$$T_{MN}^{bioch} = \frac{T_a}{C_{11}}\delta_{M1}\delta_{N1}, \tag{3.10}$$

where $\delta_{M1}\delta_{N1}$ are Kronecker delta terms, which ensure the force acts in parallel to the X_1 axis. The generation of the active tension, T_a, is described in more detail in Section 4.2.

To simulate a fibre orientation at an angle θ to the domain, the domain was rotated

by this angle. This provides an anisotropic tension diagonal to the domain, and the anisotropic diffusion in the electrical system can be set to act in the same direction, hence providing a means of simulating the fibre orientation.

3.2.6 Stress Equilibrium

Finite deformation elasticity is a subject within the theory of continuum mechanics, and the equations that govern finite deformations are derived from the conservation of linear momentum following Newton's laws of motion [64]. The stress-equilibrium equation used is defined in terms of the Second Piola-Kirchhoff tensor, T_{MN}, and this models the material independently of rigid-body motion [64,69]. In [69] it is stated that the governing equations expressed in terms of Second Piola-Kirchhoff stress components can be reduced to:

$$\frac{\partial}{\partial X_M}(T_{MN}F_N^j) = 0, \tag{3.11}$$

where there is static equilibrium in the absence of other forces and $M,N,j = 1,2,3$ in three-dimensions. The tensor notation in Equation (3.11) (and used throughout the thesis) is the summation notation, where the subscript MN indicates the summation of the individual terms. So in two-dimensions, for example, $M,N,j = 1,2$ and Equation (3.11) will expand to the summation of four terms when $j = 1$ and four terms when $j = 2$. When $j = 1$ this would be:

$$\frac{\partial}{\partial X_M}(T_{MN}F_N^1) = \frac{\partial}{\partial X_1}(T_{11}F_1^1) + \frac{\partial}{\partial X_1}(T_{12}F_2^1) + \frac{\partial}{\partial X_2}(T_{21}F_1^1) + \frac{\partial}{\partial X_2}(T_{22}F_2^1). \tag{3.12}$$

Equation (3.11) enables the calculation of the current position of an arbitrary point within the tissue, based on the stress applied to the system via the Second Piola-Kirchhoff tensor. This provides a means of calculating the deformation of the cardiac tissue depending on the current tension generated from the electrical system (see Section 4.2 for more details).

The following sections describe the expansion of the terms of the stress equilibrium equation into a format that can be solved using a numerical technique.

3.2.7 Expanding the strain energy function

To solve the system given by Equations (3.11) and (3.8) it is necessary to calculate the Second Piola-Kirchhoff tensor. Firstly this needs to be transformed into the X_1, X_2 coor-

dinate system. The Second Piola-Kirchhoff tensor is given as:

$$T_{MN} = \frac{1}{2}\left(\frac{\partial W}{\partial E_{MN}} + \frac{\partial W}{\partial E_{NM}}\right) - pC_{MN}^{-1} + T_a C_{MN}^{-1}, \tag{3.13}$$

where E_{MN} are the components of the Lagrangian Green's strain tensor. The strain energy function, W, for an incompressible Mooney-Rivlin material is given by:

$$W(I_1, I_2) = c_1(I_1 - 3) + c_2(I_2 - 3), \tag{3.14}$$

where I_1 and I_2 are the first and the second invariants of the Cauchy-Green deformation tensor and c_1 and c_2 are material constants determined from experimentation.

There are three important invariants of $\mathbf{C}$ which remain unchanged under coordinate rotation at a given deformation and these are given by:

$$I_1 = tr\mathbf{C} \tag{3.15}$$

where tr is the notation for the 'trace' (the sum of the diagonal elements) of the tensor $\mathbf{C}$,

$$I_2 = \frac{1}{2}\left[(tr\mathbf{C})^2 - tr\mathbf{C}^2\right], \tag{3.16}$$

$$I_3 = det(\mathbf{C}), \tag{3.17}$$

and for an incompressible material $I_3 = 1$. In the calculation of $\mathbf{C}$ the three-dimensional form is used as this ensures the correct version of I_1 and I_2 are calculated.

Equation (3.5) can be re-arranged to put $\mathbf{C}$ in terms of $\mathbf{E}$, i.e.,

$$\mathbf{C} = 2\mathbf{E} + \mathbf{I}. \tag{3.18}$$

This enables W to be expressed in terms of $\mathbf{E}$, and the partial derivatives of W stated with respect to $\mathbf{E}$. First the trace of $\mathbf{C}$ is:

$$I_1 = tr(\mathbf{C}) = C_{11} + C_{22} + 1, \tag{3.19}$$

and given that

$$\mathbf{C}^2 = \begin{pmatrix} C_{11}^2 + C_{12}C_{21} & C_{11}C_{12} + C_{12}C_{22} & 0 \\ C_{11}C_{21} + C_{21}C_{22} & C_{12}C_{21} + C_{22}^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{3.20}$$

it follows that:

$$I_2 = \frac{1}{2}\left[(tr\mathbf{C})^2 - tr(\mathbf{C}^2)\right] = C_{11}C_{22} - C_{12}C_{21} + C_{11} + C_{22}. \tag{3.21}$$

Hence:

$$W = c_1(C_{11} + C_{22} + 1 - 3) + c_2(C_{11}C_{22} - C_{12}C_{21} + C_{11} + C_{22} - 3). \tag{3.22}$$

W needs to be expressed in terms of E, to enable it to be differentiated with respect to E, i.e.

$$W = 2c_1(E_{11} + E_{22}) + 4c_2[E_{11}E_{22} - E_{12}E_{21} + E_{11} + E_{22}] \tag{3.23}$$

This gives:

$$\frac{\partial W}{\partial E_{11}} = 2c_1 + 4c_2E_{22} + 4c_2, \tag{3.24}$$

$$\frac{\partial W}{\partial E_{22}} = 2c_1 + 4c_2E_{11} + 4c_2, \tag{3.25}$$

$$\frac{\partial W}{\partial E_{12}} = -4c_2E_{21}, \tag{3.26}$$

and

$$\frac{\partial W}{\partial E_{21}} = -4c_2E_{12}. \tag{3.27}$$

Using the partial derivatives of W from Equations (3.24), (3.25), (3.26) and (3.27) and then substituting the components of $\mathbf{C}$ into $\mathbf{E}$, it is possible to express the values of T_{MN} in terms of derivatives of the form $\frac{\partial x_i}{\partial X_M}$:

$$T_{11} = 2c_1 + 2c_2\left(\left(\frac{\partial x_1}{\partial X_2}\right)^2 + \left(\frac{\partial x_2}{\partial X_2}\right)^2\right) + 2c_2, \tag{3.28}$$

$$T_{22} = 2c_1 + 2c_2\left(\left(\frac{\partial x_1}{\partial X_1}\right)^2 + \left(\frac{\partial x_2}{\partial X_1}\right)^2\right) + 2c_2, \tag{3.29}$$

and the cross terms are:

$$T_{12} = T_{21} = -2c_2\left(\frac{\partial x_1}{\partial X_1}\frac{\partial x_1}{\partial X_2} + \frac{\partial x_2}{\partial X_1}\frac{\partial x_2}{\partial X_2}\right). \tag{3.30}$$

Equations (3.28), (3.29) and (3.30) provide the four terms of the Second Piola-Kirchhoff tensor in two-dimensions in terms of the reference (X_M) and deformed (x_i) coordinates.

3.2.8 Stress equilibrium equation in weak form

To solve with the finite element method, the governing equation (see Equation (3.11)) needs to be put into its weak form, giving:

$$\int_{\partial\Omega}\left(T_{MN}F_N^j\Psi n_j\right)\mathrm{d}S-\int_{\Omega}\left(T_{MN}F_N^j\frac{\partial\Psi}{\partial X_M}\right)\mathrm{d}\Omega=0, \tag{3.31}$$

where $j=1,2$ (for two-dimensional problems), Ψ is the weight function, Ω is the domain being modelled, n_j is a normal (in the X_j direction) to the domain with a boundary S. The first term on the left hand side of Equation (3.31) relates to the boundary term and as no forces are applied to the boundary this is set to zero. In this thesis the simulations are undertaken in two-dimensions, and this means that $T_{3N}=T_{M3}=0$.

Including both the biochemical and elastic components (from Equations (3.6) and (3.7)), and setting the tension to act parallel to the X_1 direction (from Equation (3.10)), the Second Piola-Kirchhoff tensor (T_{MN}) is:

$$T_{MN}=\frac{1}{2}\left(\frac{\partial W}{\partial E_{MN}}+\frac{\partial W}{\partial E_{NM}}\right)-pC_{MN}^{-1}+\frac{T_a}{C_{11}}\delta_{M1}\delta_{N1}. \tag{3.32}$$

Equation (3.32) can be expanded using Equations (3.28), (3.29) and (3.30) and then Equation (3.31) becomes:

$$\begin{aligned}\int_{\Omega}\Bigg(&\left(2c_1+2c_2\left(\frac{\partial x_1}{\partial X_2}\frac{\partial x_1}{\partial X_2}+\frac{\partial x_2}{\partial X_2}\frac{\partial x_2}{\partial X_2}\right)+2c_2-pC_{11}^{-1}+\frac{T_a}{C_{11}}\right)\left(\frac{\partial x_j}{\partial X_1}\right)\left(\frac{\partial\Psi_n}{\partial X_1}\right)+\\&\left(-2c_2\left(\frac{\partial x_1}{\partial X_1}\frac{\partial x_1}{\partial X_2}+\frac{\partial x_2}{\partial X_1}\frac{\partial x_2}{\partial X_2}\right)-pC_{12}^{-1}\right)\left(\frac{\partial x_j}{\partial X_1}\right)\left(\frac{\partial\Psi_n}{\partial X_2}\right)+\\&\left(-2c_2\left(\frac{\partial x_1}{\partial X_1}\frac{\partial x_1}{\partial X_2}+\frac{\partial x_2}{\partial X_1}\frac{\partial x_2}{\partial X_2}\right)-pC_{21}^{-1}\right)\left(\frac{\partial x_j}{\partial X_2}\right)\left(\frac{\partial\Psi_n}{\partial X_1}\right)+\\&\left(2c_1+2c_2\left(\frac{\partial x_1}{\partial X_1}\frac{\partial x_1}{\partial X_1}+\frac{\partial x_2}{\partial X_1}\frac{\partial x_2}{\partial X_1}\right)+2c_2-pC_{22}^{-1}\right)\left(\frac{\partial x_j}{\partial X_2}\right)\left(\frac{\partial\Psi_n}{\partial X_2}\right)\mathrm{d}\Omega\\&=0,\end{aligned} \tag{3.33}$$

where $j=1,2$ and this relates to building the terms that are used to solve in the X_1 and X_2 directions respectively and $n=1,2,3$ for a linear triangular element, and

$$C_{11}=\left(\frac{\partial x_1}{\partial X_1}\right)^2+\left(\frac{\partial x_2}{\partial X_1}\right)^2, \tag{3.34}$$

and where

$$\mathbf{C}^{-1} = \frac{1}{det(\mathbf{C})} \begin{pmatrix} \frac{\partial x_1}{\partial X_2}\frac{\partial x_1}{\partial X_2} + \frac{\partial x_2}{\partial X_2}\frac{\partial x_2}{\partial X_2} & -(\frac{\partial x_1}{\partial X_1}\frac{\partial x_1}{\partial X_2} + \frac{\partial x_2}{\partial X_1}\frac{\partial x_2}{\partial X_2}) & 0 \\ -(\frac{\partial x_1}{\partial X_1}\frac{\partial x_1}{\partial X_2} + \frac{\partial x_2}{\partial X_1}\frac{\partial x_2}{\partial X_2}) & \frac{\partial x_1}{\partial X_1}\frac{\partial x_1}{\partial X_1} + \frac{\partial x_2}{\partial X_1}\frac{\partial x_2}{\partial X_1} & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{3.35}$$

3.2.9 Determinant of deformation gradient tensor

By including a pressure term (p) in the Second Piola-Kirchhoff tensor there is an extra unknown variable in the system. The additional equation, Equation (3.8), added to enforce incompressibility, provides the extra constraint needed to solve this pressure. This extra equation does not have a pressure term in it, but is defined by the deformation gradient. The deformation gradient tensor $\mathbf{F}$ is:

$$F = \begin{pmatrix} \frac{\partial x_1}{\partial X_1} & \frac{\partial x_1}{\partial X_2} & 0 \\ \frac{\partial x_2}{\partial X_1} & \frac{\partial x_2}{\partial X_2} & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{3.36}$$

and for an incompressible material $det(\mathbf{F}) = 1$

$$det(F) = \frac{\partial x_1}{\partial X_1}\frac{\partial x_2}{\partial X_2} - \frac{\partial x_1}{\partial X_2}\frac{\partial x_2}{\partial X_1} = 1 \tag{3.37}$$

To approximate this with the finite element method the weak form of Equation (3.37) is needed. This is produced by taking everything to one side, multiplying by a weight function (Ψ) and integrating, thereby giving:

$$\int_\Omega \left(\frac{\partial x_1}{\partial X_1}\frac{\partial x_2}{\partial X_2} - \frac{\partial x_1}{\partial X_2}\frac{\partial x_2}{\partial X_1} - 1 \right) \Psi \mathrm{d}\Omega = 0, \tag{3.38}$$

where Ω is the domain and Ψ is a weight function. Equation (3.38) provides the remaining equation needed to be able to solve for the pressure unknown.

3.3 Approximating with the Finite Element Method

3.3.1 Discretising the domain

To approximate the unknown values, in this case the deformed coordinates x_1, x_2 and the pressure (p) over the domain, the FEM is used. With the FEM, the domain is discretised into smaller 'elements' and the unknown variables approximated within these elements.

The domain used in this thesis is a two-dimensional rectangle and this is divided into a mesh of unstructured triangles. This has been covered in more detail in Section 2.3.

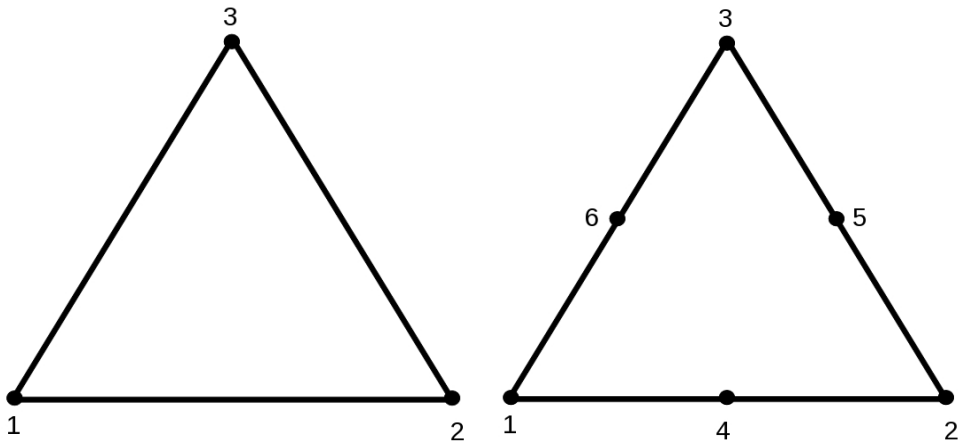
Equations (3.33) and (3.38) are the weak form of the governing equations which are needed by the finite element method. It is necessary to approximate the unknowns within each element using a basis function. It has been highlighted in [40, page 535], that there are stability issues when using the same basis functions for pressure as deformation. This is covered in detail in Section 3.13 of [40]. To resolve this, a scheme consisting of a combination of quadratic basis functions for deformations and linear basis functions for pressures [40] has been used in this thesis.

As with the electrical system, the Galerkin method [89] is implemented. In this method the same functions are used for the weight functions, introduced in the weak form of the governing equations, and the basis functions, needed for discretising the unknown values within an element. The basis and weight functions for the mechanical system are discussed in more detail below.

3.3.2 Quadratic deformation and linear pressure elements

The objective of this section is to document how to implement quadratic basis functions into the finite element approximations for the mechanical solve. The use of triangular elements with quadratic basis functions is further explained in Zienkiewicz and Taylor [126], page 179 onwards. When using quadratic elements, each triangle has 6 nodes, the three corner nodes and the three mid-points on each edge, see image (b) of Figure 3.1. For the mechanical system the deformations (in two dimensions) have two unknowns at each node, for the deformation in the X_1 and X_2 directions.

Quadratic elements are a higher order approximation to the unknown values over the element, and use a quadratic function to undertake the approximation. As with linear elements, the solution to the function is approximated at each of the element nodes, so in quadratic elements this means there are 6 points of approximation. There is a deformation equation for the unknowns in each direction at each node, so in a quadratic triangular element there are twelve deformation unknowns (and twelve associated equations in the system). The pressure unknowns are solved with linear elements, so there are three pressure unknowns for each triangular element also. This gives 15 unknowns for each element, with 15 associated equations in the system. It can be noted that in a mesh of elements, a node is typically present in more than one element and so the overall system size is considerably smaller than 15 times the number of elements. For a given set of basis function Ψ_i, the piecewise approximations to the unknowns for each element are



(a) Linear (b) Quadratic

Figure 3.1: *Triangular elements. Image (a) showing a three-node triangle used for linear approximation and Image (b) showing a six-node triangle used for quadratic approximation.*

given by:

$$x_1 = \sum_{i=1}^{6} \Psi_i x_1^i, \tag{3.39}$$

$$x_2 = \sum_{i=1}^{6} \Psi_i x_2^i, \tag{3.40}$$

$$p = \sum_{i=1}^{3} \Psi_i p^i. \tag{3.41}$$

3.3.3 Isoparametric elements

As described in Section 3.3.2, to maintain the stability of the mechanical solve, one technique is to use quadratic basis functions for the FEM approximations of the deformation and linear basis functions to model the pressures [40]. This requires the development of quadratic elements within the finite element software developed for this thesis.

To improve the flexibility of the software it was decided to implement isoparametric elements. Isoparametric elements use a local reference coordinate system and implement a mapping process to specify the relationship to the original undeformed coordinate system. The local coordinate system can be a simple element, for example a right angled triangle. This simplifies the implementation of the quadratic functions and also provides a more modular approach, whereby it is easier to substitute other higher order functions

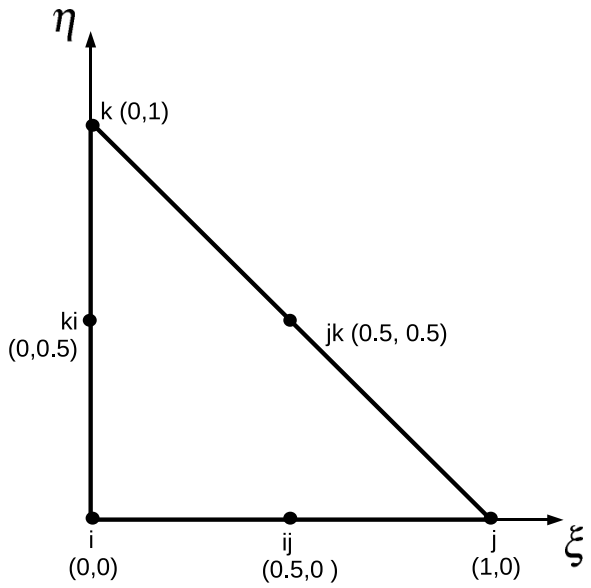


Figure 3.2: *Reference (local) triangular element*

if required later. Using quadratic isoparametric elements also makes it possible to model elements with curved boundaries [32].

In isoparametric elements, shape functions are used to specify the relation between the global (X_1, X_2) and local (ξ, η) coordinate systems. Shape functions are defined for an idealized mapped element, as per Figure 3.2. The coordinate transformation is therefore:

$$X_1 = \sum_{i=1}^{6} \Psi_i \xi_i, \tag{3.42}$$

$$X_2 = \sum_{i=1}^{6} \Psi_i \eta_i, \tag{3.43}$$

where Ψ_i are the shape functions (given in Equations (3.44) to (3.49)) for the six-node triangular element.

3.3.4 Functions for mapping local coordinates

For quadratic elements, 6 node triangles are needed (as shown in Figure 3.2) and shape functions are required for each node. The functions used for the triangle in Figure 3.2 are

the Lagrange basis functions:

$$\Psi_i = 2(1-\xi-\eta)(\frac{1}{2}-\xi-\eta) \tag{3.44}$$

$$\Psi_{ij} = 4\xi(1-\xi-\eta), \tag{3.45}$$

$$\Psi_j = 2\xi(\xi-\frac{1}{2}), \tag{3.46}$$

$$\Psi_{ki} = 4\eta(1-\xi-\eta), \tag{3.47}$$

$$\Psi_{jk} = 4\xi\eta, \tag{3.48}$$

$$\Psi_k = 2\eta(\eta-\frac{1}{2}). \tag{3.49}$$

The derivatives of these with respect to ξ are:

$$\frac{\partial\Psi_i}{\partial\xi} = -3+4\xi+4\eta, \tag{3.50}$$

$$\frac{\partial\Psi_{ij}}{\partial\xi} = 4-8\xi-4\eta, \tag{3.51}$$

$$\frac{\partial\Psi_j}{\partial\xi} = -1+4\xi, \tag{3.52}$$

$$\frac{\partial\Psi_{ki}}{\partial\xi} = -4\eta, \tag{3.53}$$

$$\frac{\partial\Psi_{jk}}{\partial\xi} = 4\eta, \tag{3.54}$$

$$\frac{\partial\Psi_k}{\partial\xi} = 0, \tag{3.55}$$

and with respect to η these are:

$$\frac{\partial\Psi_i}{\partial\eta} = -3+4\xi+4\eta, \tag{3.56}$$

$$\frac{\partial\Psi_{ij}}{\partial\eta} = -4\xi, \tag{3.57}$$

$$\frac{\partial\Psi_j}{\partial\eta} = 0, \tag{3.58}$$

$$\frac{\partial\Psi_{ki}}{\partial\eta} = 4-4\xi-8\eta, \tag{3.59}$$

$$\frac{\partial \Psi_{jk}}{\partial \eta} = 4\xi, \tag{3.60}$$

$$\frac{\partial \Psi_{k}}{\partial \eta} = -1 + 4\eta. \tag{3.61}$$

Since the coordinates are now functions of η and ξ, the deformation must be determined by differentiation using the chain rule. To obtain the derivatives of shape functions expressed in local element coordinates with respect to the global physical coordinates (X_1, X_2), first the chain rule is used:

$$\begin{pmatrix} \frac{\partial \Psi_i}{\partial \xi} \\ \frac{\partial \Psi_i}{\partial \eta} \end{pmatrix} = \begin{pmatrix} \frac{\partial \Psi_i}{\partial X_1}\frac{\partial X_1}{\partial \xi} + \frac{\partial \Psi_i}{\partial X_2}\frac{\partial X_2}{\partial \xi} \\ \frac{\partial \Psi_i}{\partial X_1}\frac{\partial X_1}{\partial \eta} + \frac{\partial \Psi_i}{\partial X_2}\frac{\partial X_2}{\partial \eta} \end{pmatrix}. \tag{3.62}$$

This can be rearranged into the Jacobian ($\mathbf{J}^e$) multiplied by a vector of derivatives as follows:

$$\begin{pmatrix} \frac{\partial \Psi_i}{\partial \xi} \\ \frac{\partial \Psi_i}{\partial \eta} \end{pmatrix} = \begin{pmatrix} \frac{\partial X_1}{\partial \xi} & \frac{\partial X_2}{\partial \xi} \\ \frac{\partial X_1}{\partial \eta} & \frac{\partial X_2}{\partial \eta} \end{pmatrix} \begin{pmatrix} \frac{\partial \Psi_i}{\partial X_1} \\ \frac{\partial \Psi_i}{\partial X_2} \end{pmatrix}, \tag{3.63}$$

where the Jacobian, $\mathbf{J}^e$, is the 2x2 matrix on the right hand side of Equation (3.63). The required derivatives can then be obtained by rearranging Equation (3.63), to give:

$$\begin{pmatrix} \frac{\partial \Psi_i}{\partial X_1} \\ \frac{\partial \Psi_i}{\partial X_2} \end{pmatrix} = (\mathbf{J}^e)^{-1} \begin{pmatrix} \frac{\partial \Psi_i}{\partial \xi} \\ \frac{\partial \Psi_i}{\partial \eta} \end{pmatrix} \tag{3.64}$$

From Equations (3.42) and (3.43), the derivative of the global coordinates, with respect to the local coordinates, can be calculated, giving

$$\mathbf{J}^e = \begin{pmatrix} \sum_{i=1}^{6} \frac{\partial \Psi_i}{\partial \xi} X_1^i & \sum_{i=1}^{6} \frac{\partial \Psi_i}{\partial \xi} X_2^i \\ \sum_{i=1}^{6} \frac{\partial \Psi_i}{\partial \eta} X_1^i & \sum_{i=1}^{6} \frac{\partial \Psi_i}{\partial \eta} X_2^i \end{pmatrix}. \tag{3.65}$$

By utilising Equations (3.44) to (3.65) it is possible to map the global coordinate system (X_1, X_2) to the local coordinate system (ξ, η).

3.3.5 Quadrature and integration

The use of quadratic elements means that the integrals in Equations (3.33) and (3.38) cannot be simply calculated by multiplying by the area of the element, rather these are calculated with numerical quadrature. In this section the left hand sides of Equations (3.33) and (3.38) equations will be referred to as $f(X_1, X_2)$.

Quadrature calculates a numerical estimate of an integral by selecting a number of

n	W_i	Coord ξ_i	Coord η_i
3	0.333333333333333	0.666666666666667	0.166666666666667
		0.166666666666667	0.166666666666667
		0.166666666666667	0.666666666666667

Table 3.1: *A 3-Point quadrature rule for a triangle*

abscissae at which to evaluate the function being integrated, these function evaluations are then each multiplied by a weight value and summed together. The quadrature approximation for a triangle is:

$$\int\int_{\Omega^e} f(X_1, X_2)\mathrm{d}x\mathrm{d}y \approx \triangle^e \sum_{i=1}^{n} W_i f(X_1^i, X_2^i), \tag{3.66}$$

where X_1^i and X_2^i are the evaluation points for the function, n is the number of points evaluated over, Ω^e is the triangular element, $\triangle^e$ is the area of the triangular element and W_i are the weighting values. This is an approximation to the exact integral, and by using the correct number and position of evaluation points the approximation error can be controlled.

As described in Section 3.3.3, an isoparametric mapping is used, which involves using a reference (local coordinates) triangle and then mapping this to the real coordinates (global coordinates) using a Jacobian transformation.

$$\int\int_{\Omega^e} f(\xi, \eta)\mathrm{d}\xi\mathrm{d}\eta \approx \triangle^e \sum_{i=1}^{n} W_i f(\xi_i, \eta_i)\, det(\mathbf{J^e}), \tag{3.67}$$

where ξ, η are the local coordinates, $\triangle^e$ is the area of the reference triangle ($\triangle^e = 0.5$ for the reference triangle used) and $det(\mathbf{J^e})$ is the transformation Jacobian as defined in Equation (3.65).

The quadrature rule used is defined in Table 3.1 [23], and this has a degree of precision of 2 which produces exact results for quadratic polynomials.

3.4 Solving the mechanical system of equations

3.4.1 Newton method for a system of equations

Due to the highly non-linear nature of Equations (3.33) and (3.38) it was decided to use the Newton iterative method to solve them. The Newton method determines the solutions

of a system of n non-linear equations which are given by:

$$\begin{aligned} f_1(x_1, x_2, \cdots, x_n) &= 0, \\ f_2(x_1, x_2, \cdots, x_n) &= 0, \\ \vdots \quad & \quad \vdots \\ f_n(x_1, x_2, \cdots, x_n) &= 0. \end{aligned} \tag{3.68}$$

This is seeking to solve n non-linear equations in n unknowns. The notation can be simplified by defining a vector valued function **f(x)** as follows:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_n(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} f_1(x_1, x_2, \cdots, x_n) \\ f_2(x_1, x_2, \cdots, x_n) \\ \vdots \quad \vdots \\ f_n(x_1, x_2, \cdots, x_n) \end{bmatrix}, \tag{3.69}$$

so the system of Equations (3.68) can be written as:

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}, \tag{3.70}$$

where $\mathbf{0}$ denotes the zero vector, $\mathbf{0} = [0, 0, 0 \cdots, 0]^T$. Assuming that a solution ($\mathbf{x} = [x_1, x_2, \cdots, x_n]^T$) exists for the system then the method iteratively finds a set of vectors $\mathbf{x}_m$, such that

$$\lim_{m \to \infty} \mathbf{x}_m = \mathbf{x} \tag{3.71}$$

The Newton method for a system of equations is given by [80]:

$$\mathbf{J}_{f(x)}(\mathbf{x}_m)\delta_m = -\mathbf{f}(\mathbf{x}_m), \tag{3.72}$$

where $\delta_m = (\mathbf{x}_{m+1} - \mathbf{x}_m)$ and the Jacobian of $\mathbf{f}(\mathbf{x})$ is defined as:

$$\mathbf{J}_{f(\mathbf{x})} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}) & \frac{\partial f_1}{\partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f_1}{\partial x_n}(\mathbf{x}) \\ \frac{\partial f_2}{\partial x_1}(\mathbf{x}) & \frac{\partial f_2}{\partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f_2}{\partial x_n}(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1}(\mathbf{x}) & \frac{\partial f_n}{\partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f_n}{\partial x_n}(\mathbf{x}) \end{bmatrix}. \tag{3.73}$$

The linear system (3.72) is known as the inner solve and has the unknowns δ_m. The solution of this linear equation system is then used to calculate the next value of $\mathbf{x}_m$,

where:

$$\mathbf{x}_{m+1} = \mathbf{x}_m + \delta_m, \tag{3.74}$$

and this is known as the outer iteration. The process starts with an initial value $\mathbf{x}_0$, and progresses until a given convergence tolerance is reached, i.e. $|\mathbf{f}(\mathbf{x})| < tol$.

Process 3.1 Newton Method

1: Set initial estimate $\mathbf{x}_0$.
2: Set a convergence tolerance, i.e. $|\mathbf{f}(\mathbf{x})| < tol$
3: Repeat until the convergence tolerance is met:
4: Compute the Jacobian $\mathbf{J}_{f(x)}$
5: Solve the linear system $\mathbf{J}_{f(x)}\delta_m = -\mathbf{f}(\mathbf{x})$ for the unknown vector δ_m
6: Set $\mathbf{x}_{m+1} = \mathbf{x}_m + \delta_m$

In Equation (3.72) the solution is expected to provide a small change of $\mathbf{x}_m$ and therefore an initial guess of zero is used for the vector δ . To improve the performance of the iteration given in Equation (3.72), the linear system can be preconditioned, as discussed further in Chapter 5.

The outer Newton iteration, given in Equation (3.74), needs to be provided with an initial estimate for the $\mathbf{x}_m$ vector. This vector is a combination of the deformation and pressure unknowns. For the deformation unknowns, the first time the system is solved the initial estimate is the undeformed coordinate positions. For the second solve, the first solution of deformed coordinates is used, and from then on a linear extrapolation of the last two solutions for deformed coordinates are used. For the pressure unknowns, the first time the system is solved the initial estimate is a given constant, for the second solve, the first solution of the pressures is used, and from then on a linear extrapolation of the last two solutions for the pressures are used. The mechanical system is not time dependent, if the active tension changes rapidly, the mechanical solver needs to be undertaken frequently to ensure the initial estimate does not become too inaccurate. If the initial estimate becomes too inaccurate the method may fail to converge.

3.4.2 Building the function vector

The Newton method seeks to solve an equation of the form $\mathbf{f}(\mathbf{x}) = \mathbf{0}$, and Equations (3.33) and (3.38) provide a set of n equations for the n unknowns of the system. The central process in building the function vector, $f(X_1, X_2)$, is the quadrature loop, described by Equation (D.34). By undertaking the quadrature loop for each element in the mesh, the contributions to the $\mathbf{f}$ vector are calculated. Each row of the function vector corresponds to an unknown in the system, and as described previously, each corner node of a triangular

element has two deformation unknowns and one pressure unknown. Each mid-point node has two deformation unknowns. As each element is processed in the quadrature loop, the values calculated are added to the row of the vector that corresponds to that nodal unknown.

Within **f**, the first set of entries contain the deformations in the X_1 and X_2 directions, these relate to the unknowns x_1 and x_2. The bottom entries of the function vector are used for the pressure unknowns. The node numbering for the mechanical nodes (and hence the order of the function vector) is determined by the RCM method as described in Section 2.5.5. Once the quadratic nodes have been built from the original linear triangles (described in Section 4.1.1 and Process 4.1), the RCM method is undertaken to re-order the nodes.

With an incompressible material the deformations are zero with no external force applied and so there are no boundary contributions from Equation (3.33). However to prevent spurious rotation and translation, and preserve uniqueness of the solution, it is necessary to fix a number of nodal coordinates within the system. The technique used typically is to fix the node closest to the centre of the domain in both directions and fix a neighbouring node (with a similar X_2 position) in the X_1 direction. This removes three unknowns from the system. It is possible to remove these unknowns from the system and restructure the vector of unknowns, however this is a complex process that is time consuming. To resolve this, the solution adopted is to include the fixed nodes in the solution, but set their values to the known undeformed locations.

3.4.3 Solving with KINSOL

To solve the non-linear mechanical system the third party tool, KINSOL is employed. This is part of the SUNDIALS (SUite of Nonlinear and DIfferential/ALgebraic equation Solvers) suite of programs managed by the Lawrence Livermore National Laboratory (see https://computation.llnl.gov/casc/sundials/main.html) and provides a suite of iterative solvers.

KINSOL requires an equation of the form:

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}, \tag{3.75}$$

and that a user defined software procedure is provided that calculates **f**. The function vector $\mathbf{f}(\mathbf{x}_m)$, built as described in Section 3.4.2, is used to generate this software procedure. At the basic level, it is only necessary to provide KINSOL with a function to calculate the function vector **f(x)** and an initial estimate vector. KINSOL then undertakes all the actions

to perform the Newton iterations. Using this technique provides a matrix-free approach to solving the mechanical system.

In the non-preconditioned (see Chapter 5) simulations, the sparse GMRES [96] solver function (KINSPGMR) was used. When right preconditioning is required it is also necessary to provide KINSOL with a preconditioning set-up function and an inner solve function. The set-up function calculates the numerical Jacobian and then passes it to the ILUT preconditioner in SPARSKIT. The solve function then utilises the GMRES solver provided by SPARSKIT. So when the preconditioned solve is undertaken only the outer Newton iterations are undertaken by KINSOL, the preconditioning and inner solve are undertaken by SPARSKIT and the numerical Jacobian is built within the software developed for this thesis. By the end of the thesis KINSOL was only used as a skeleton framework and the majority of its functions had been replaced.

The generalized minimal residual method (GMRES [96]) is an Krylov subspace iterative method for the solution of a system of linear equations. The combination of the Newton iteration for the outer solve and the GMRES solver for the inner solve results in these solvers being referred to as 'Newton-Krylov'.

3.5 Conclusions

In this chapter a number of concepts have been covered, including modelling the cardiac tissue mathematically, approximating this with the FEM, using isoparametric quadratic elements, integrating with quadrature and solving with the Newton method. By combining these techniques mathematical equations of cardiac tissue deformation can be built and numerical techniques used to approximate these equations, thereby providing a means of simulating the deformation of cardiac tissue.

The governing equations for the mechanical system are given in Equations (3.33) and (3.38). These are approximated using the FEM, over an unstructured mesh of triangular elements, and this process results in a system of n equations for the n unknowns in the system. The unknowns relating to the deformations in the X_1 and X_2 directions and the pressure p at each node in the mesh. A matrix-free solver has been implemented that removes the need to algebraically build the FEM stiffness matrix, rather a function vector is calculated. This function vector is then used within an iterative Newton technique to solve the system.

The cardiac tissue is assumed to be incompressible, and forces are applied to the system via the active tension variable (T_a), present in the Second Piola-Kirchhoff tensor, and generated from the electrical system. Therefore for deformations to be produced in

the mechanical model it needs to be coupled with the electrical system. This is covered in detail in Chapter 4, which also shows the tests undertaken to consider the convergence and validation of the mechanical deformation. The mechanical system is solved using the FEM over unstructured triangular elements and this facilitates its coupling with the electrical system.

By setting the tissue as incompressible, an extra unknown variable (for pressure) was added and this resulted in the combination of linear and quadratic elements used in the FEM. This adds complexity to the model, however in the cardiac modelling literature incompressibility is now widely enforced.

The use of quadratic elements with isoparametric mappings enables the production of curved boundary edges, which are more biologically realistic. Isoparametric elements also allow easier implementation of higher order basis functions if required in the future.

The Mooney-Rivlin strain energy function, used in this thesis, is isotropic, however by applying the tensile forces in one direction, fibre orientation can be simulated, and this technique is used in the heart failure simulations in Chapter 7.

Chapter 4

Coupled electromechanical model

4.1 Introduction

To simulate cardiac activity it is necessary to model the electrical wave as it passes through the tissue and the resulting deformation from the contraction caused by changes in the cellular ion concentrations. The objective of this chapter is to describe how the model of electrophysiology described in Chapter 2 and the model of cardiac mechanics described in Chapter 3 are coupled together. This produces a model in which the domain deforms as the electrical wave passes through it.

As discussed previously in this thesis, the electrophysiology and mechanics are approximated using the finite element method, using two-dimensional meshes. The electrical system varies in time and space and hence each mesh node in the electrical system needs to store its position in the coordinate space. The mechanical system and electrical system share common mesh nodes and this allows changes to the mechanical system (i.e. the deformation of the domain) to be passed back easily to the electrical system.

4.1.1 Mesh generation

In this work, it is a requirement that the mesh supports both the electrical and mechanical systems. The mechanical system (see Chapter 3) is highly non-linear and the time to solve this is much greater than the electrical system (see Section 5.1) on the same mesh. The electrical system needs a fine mesh to properly resolve the electrical wave front, and

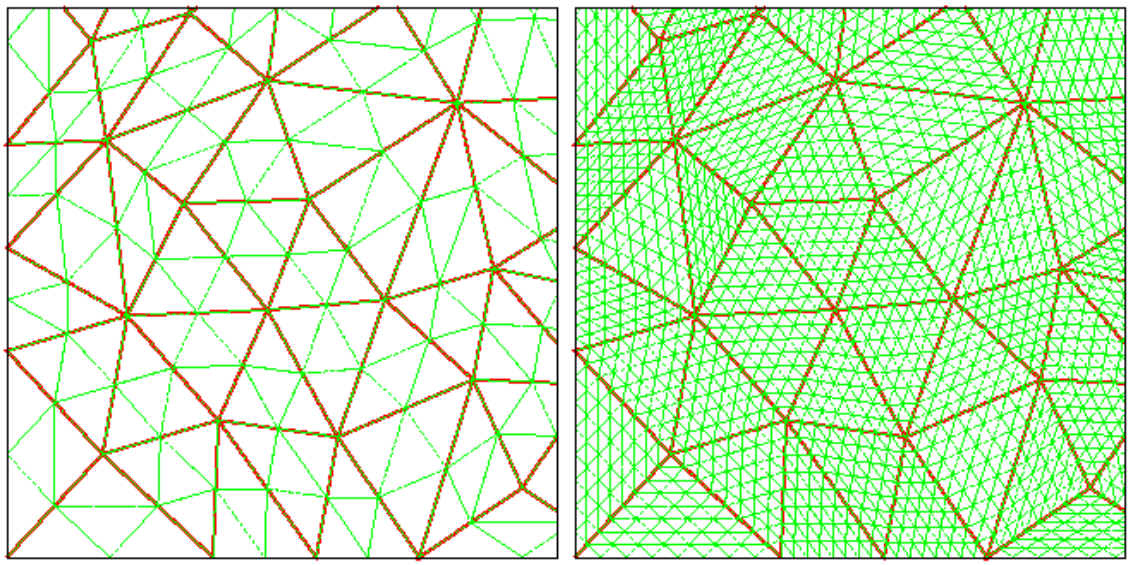
due to the performance of the mechanical system it is impractical to use the same level of refinement for both systems. However the electrical and mechanical meshes need to operate together so that information can be passed easily between the two systems in the coupling process (see Section 4.2). The mechanical system uses quadratic basis/weight functions and this requires each triangle to have six nodes (see Section 3.3.2), with an extra node being added at the mid-point of each of the triangle's edges. To facilitate these objectives an embedded hierarchy of meshes is implemented using the strategy described in Process 4.1.

Process 4.1 Overview of mesh refinement technique

1: An initial mesh (Mesh0), created using the 'Triangle' [99, 100] mesh generation software, is loaded.
2: A procedure runs to find the mid-point of the edges of all the triangles of Mesh0
3: New nodes are created at these midpoints
4: The three new nodes are added to Mesh0 to create the extra nodes for the quadratic elements
5: A new mesh is created, MeshE, with four new elements created from every triangle in Mesh0 using the midpoint nodes as the new vertices.
6: The set of edges for the elements MeshE are determined.
7: A max refinement (MR) parameter is specified
8: **while** NumRefines $\leq$ MR **do**
9: Find midpoint of edges in elements in MeshE
10: Create new nodes at these midpoints (unless previously created)
11: Divide each element into four new elements using the new midpoint nodes
12: Add the new elements to MeshE
13: Re-calculate the edges of the elements of MeshE
14: NumRefines = NumRefines + 1
15: **end while**

The technique described in Process 4.1 provides a flexible solution whereby the refinement of the mechanical mesh can be easily altered, whilst maintaining the required level of fine refinement for the electrical system. Figure 4.1 includes examples of the first and third levels of refinement. The first refinement produces both the extra nodes needed for the quadratic elements in the mechanical system and a new, more refined mesh, for the electrical system. The second and subsequent refinements then further refine the electrical mesh only.

By using Process 4.1 it ensures that all nodes in the mechanical system are present in the electrical system and the mid-point nodes also become the vertices of a more refined triangle in the electrical mesh. Figure 4.2 shows a very simple mesh that contains 9 nodes. With this mesh the mechanical system would have two 6-node elements, the



(a) First refinement (b) Third refinement

Figure 4.1: *Refinement of triangular elements in a subsection of a mesh*

first element is comprised of nodes {1,5,2,6,4,7} and the second element is comprised of nodes {2,9,3,8,4,6}. The electrical system would have eight 3-node elements comprised of nodes {1,5,7}, {5,2,6}, {5,6,7}, {6,4,7}, {2,9,6}, {9,3,8}, {9,8,6} and {8,4,6}.

By defining a mesh that has common nodes in both the electrical and mechanical system it is straightforward to update both the mechanical system with changes to the transmembrane voltage and the electrical system with changes to the coordinate locations of the nodes.

4.2 Coupling the electrical and mechanical systems

As described in Chapter 3, the mechanical system being solved is derived from Equations (3.7) and (3.8). Equation (3.7) is repeated here again for convenience:

$$T^{MN} = \frac{1}{2}\left(\frac{\partial W}{\partial E_{MN}} + \frac{\partial W}{\partial E_{NM}}\right) - pC^{-1}_{MN} + T_a C^{-1}_{MN}, \tag{4.1}$$

where T^{MN} is the Second Piola Kirchhoff tensor, W is the scalar strain energy function, E the Lagrangian Green strain tensor, p is a Lagrange multiplier, T_a is the active tension generated by the electrical system, and C is Green's strain tensor.

The active tension component (T_a) of Equation (4.1) is determined using the voltage generated by the electrical system and it is this variable that is used to couple the electrical

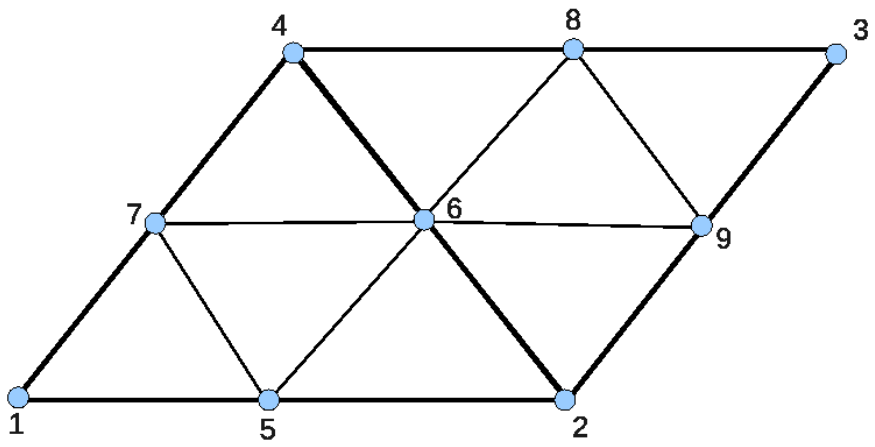


Figure 4.2: *Example simple mesh with 9 nodes, which can be used to form 2 quadratic elements and/or 8 linear elements.*

and mechanical models. An active tension is generated for each node in the electrical mesh (see Section 4.2.2) and as the electrical and mechanical nodes are common to both meshes, there is a tension value at each node in the mechanical mesh. After a given number of time steps of the electrical system, the mechanical solve is undertaken and a new tissue deformation is produced. The coordinates of the deformed tissue are then passed back to the common nodes in the electrical model.

The electrical system requires a fine resolution to properly represent the steep wave front of a cardiac action potential. The mechanical system is highly non-linear and hence takes a relatively long time to solve (see Chapter 5), and so the electrical mesh is refined further than the mechanical mesh. That is, there are more elements (and nodes) in the electrical system than in the mechanical system. This means that when deformation is calculated in the mechanical system and passed back to the electrical system there will be nodes in the electrical system that do not have a corresponding entry in the mechanical system. The deformations for these nodes are interpolated from the nodes that are in common.

4.2.1 Weak and strong coupling

The method described in Section 4.2 above is known as 'weak coupling' since the two systems are solved independently. Another method used in this research area is to 'strongly couple' the electromechanical models (for example [58,71,72]), in which both the electrical and mechanical models are solved simultaneously. Strongly coupled systems make it

possible to incorporate other biological effects of electromechanical coupling. For example, length dependent calcium binding or stretch activated conductance channels. These features are present in some cellular and tissue models (for example [59, 75, 104]), however for the simulations undertaken in this thesis these deformation dependent features are not utilised and so a strongly coupled model is not mandated.

4.2.2 Modelling active tension

The biochemical component of the Second Piola-Kirchhoff tensor (Equation (4.1)) contains the active tension variable (T_a). Equations (22c) and (23) from [69] (subsequently updated at www.cellml.org website) provide a phenomenological description of the tension within cardiac tissue and these are reproduced here, with the rate of change of the active tension given by:

$$\frac{\partial T_a}{\partial t} = \varepsilon(V)(K_{T_a}V - T_a), \tag{4.2}$$

where K_{T_a} controls the amplitude of the active tension (T_a), V is the transmembrane voltage, and the function $\varepsilon(V)$ is defined by:

$$\varepsilon(V) = 10\varepsilon_0 \text{ for } V < 0.005, \qquad \varepsilon(V) = \varepsilon_0 \text{ for } V \geq 0.005, \tag{4.3}$$

and where the initial value for the active tension is 0 at all nodes.

In [69] the electrical model used is the Aliev and Panfilov model [2]. This electrical model produces a normalised transmembrane voltage which ranges from $0 \leq V \leq 1$. The ten Tusscher–Panfilov electrical model [110] used in this thesis, produces transmembrane voltage in a biologically realistic range of $-86.2mV \leq V \leq 30mV$. To account for this, the transmembrane voltage calculated from the electrical model is scaled to $0 \leq V \leq 1$ before it is used as an input to Equations (4.2) and (4.3).

4.2.3 Transient modelling considerations

In this thesis weak coupling is implemented and hence the electrical and mechanical systems need to feedback to each other after a certain period of time.

In [82] the mechanical time-step is considered and tests were undertaken on range of values from 0.01 to 1.28 ms, with the conclusion being that a 'sensible' choice of time step for the mechanical component is 1 ms. In the simulations in Chapter 7 the mechanical time step employed was typically 0.8 ms, and this was undertaken after ten time steps of the electrical solver running at a time step of 0.08 ms.

It is noted in [82] that large time steps do not necessarily lead to a faster overall solve time. As the time-step of the mechanical solver is increased the initial estimate used in the Newton iterative solver (see Section 3.4.1) becomes progressively more inaccurate. This leads to more Newton iterations which are time consuming. In this thesis it was also noticed that if the initial estimate becomes too inaccurate the Newton solver can fail to converge. To enable large tensions early on in simulations, a technique has been developed that progressively increases the tension with the electrical system disabled. This enables the production of a good initial estimate for larger deformations.

4.3 Validation and convergence

4.3.1 Convergence of mechanical solver

To consider the convergence of the mechanical system a number of tests were undertaken with varying levels of mesh refinement. A mesh with a coarse level of refinement with 96 mechanical nodes (these are the quadratic nodes from 6 node triangles) was created and this was refined to form 5 subsequent embedded meshes. These meshes were each 120 mm $\times$ 120 mm and were fixed along their left edge (where $X_1 = 0$). Providing a consistent, repeatable test for the mechanical deformation requires that the variability of the electrical wave speed (see Chapter 2) be removed. This was achieved by setting an extremely high value the active tension maximum parameter ($K_{T_a} = 4000$) KPa and running the mechanical solve for only one time step. This ensures that electrical wave has not moved and that the active tension force is in the same place (along the left edge) in all tests.

Figure 4.3 shows the deformed boundary of the domain for three of the meshes tested and, as can be seen from this figure, the results are all similar (at this scale). The main variation occurs close to the left edge corners, where the coarse meshes cannot reproduce the curve of the boundary. This can be seen clearly in Figure 4.4. These results also show that in the top right hand corner (see Figure 4.5), the difference between the meshes is very small. There is less of a curve to the domain in this area and hence a coarse mesh with fewer triangles can still approximate this quite well. The coarse meshes become less accurate in areas where a finer curve needs to be reproduced.

To consider the convergence of the solution as the meshes were refined a further refined embedded mesh was created with with 84385 nodes. The mechanical solve was undertaken and the solution to this used as a benchmark to compare the convergence of the other meshes. As the meshes are all embedded within the original 41 element mesh,

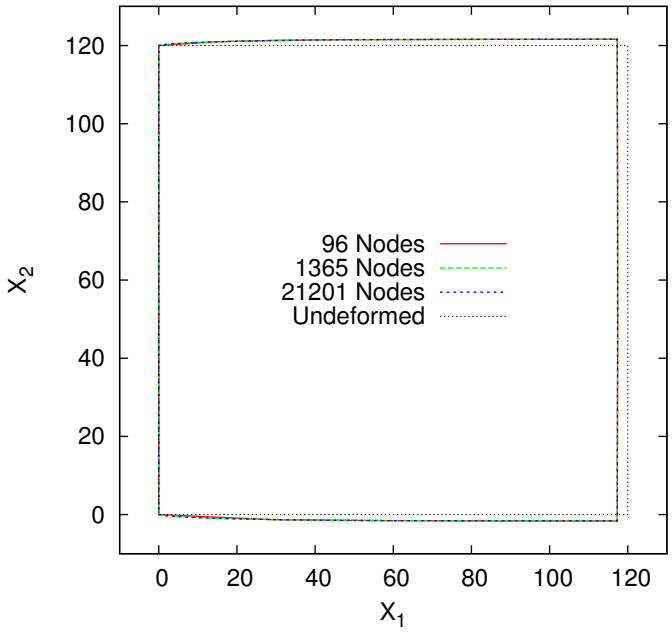


Figure 4.3: *Mechanical deformation convergence of domain edge. Tests undertaken on a 120 mm × 120 mm domain with varying mesh refinement. Meshes fixed along edge where $X_1 = 0$, and the active tension maximum parameter (K_{T_a}) set to 4000 KPa and one mechanical solve undertaken.*

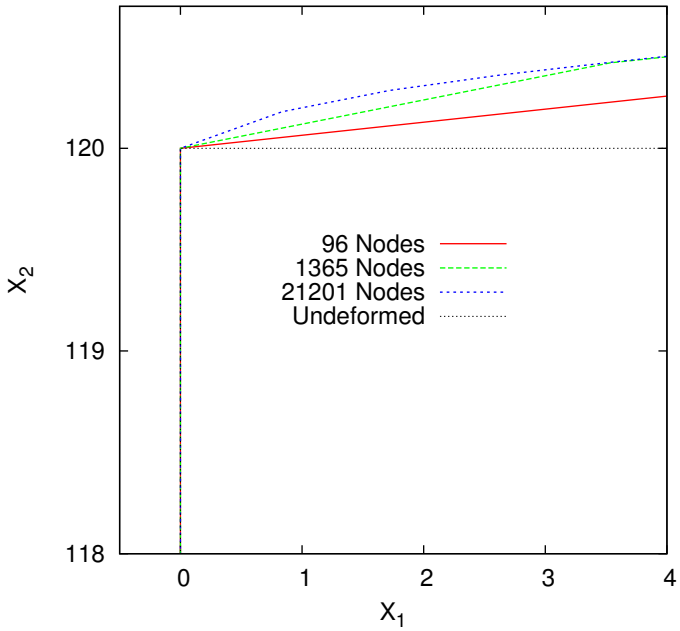


Figure 4.4: *Mechanical deformation convergence–domain edge zoom top left. Tests undertaken on a 120 mm × 120 mm domain with varying mesh refinement. Meshes fixed along edge where $X_1 = 0$, and the active tension maximum parameter (K_{T_a}) set to 4000 KPa and one mechanical solve undertaken.*

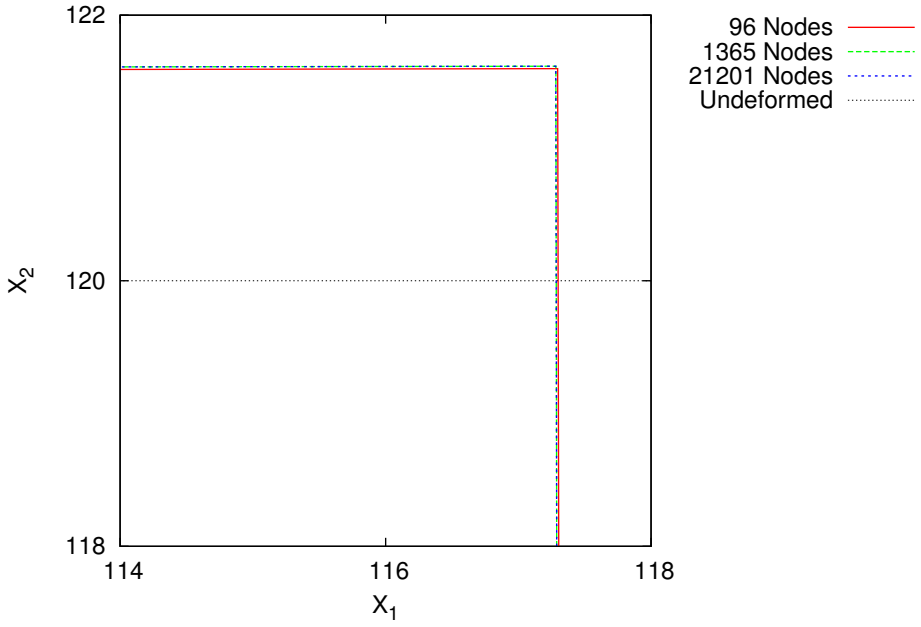


Figure 4.5: *Mechanical deformation convergence–domain edge zoom top right. Tests undertaken on a 120 mm × 120 mm domain with varying mesh refinement. Meshes fixed along edge where $X_1 = 0$, and the active tension maximum parameter (K_{T_a}) set to 4000 KPa and one mechanical solve undertaken.*

Quadratic Nodes	Elements	RMS Error Norm X_1	RMS Error Norm X_2	Ratio of X_1 convergence
96	41	0.013080	0.010603	-
355	164	0.004959	0.002804	2.64
1365	656	0.001501	0.000235	3.30
5353	2624	0.000436	0.000045	3.44
21201	10496	0.000147	0.000031	2.96
84385	41984	-	-	-

Table 4.1: *Mechanical convergence in X_1 and X_2 directions, RMS error compared to solution with 84385 nodes.*

they all share 96 nodes (the original quadratic nodes). These nodes were used to take a root mean squared (RMS) error norm of the permutations in the X_1 and X_2 directions. The results can be seen in Table 4.1, and these are individually plotted using log scales in Figure 4.6.

The error convergence in Table 4.1 considers the position of the 96 common nodes between the meshes and these are both internal and on the domain boundary. This table shows that as the spatial step is halved, the error in the X_1 direction reduces by up to 3.44 times, hence giving a convergence that is approaching quadratic.

Figures 4.7 and 4.8 illustrate the displacement of the elements in the top left hand corner of the domain as the mesh is refined. These images show that as the number of nodes in the mesh increases, a smoother curve can be represented on the domain edge.

4.3.2 Validating the mechanical model

In order to validate the mechanical model, the results from the coupled solver were compared to the results in [82]. The left edge (where $X_1 = 0$) of the domain was fixed in space and stimulated to form a plane wave. The results from this are displayed in Figure 4.9.

These results provide deformations quantitatively similar to Figure 1 of [82] and with a similar deformation profile. Specifically in both simulations, as the electrical wave moves from left to right across the domain, the top and bottom edges of the domain extend outwards and the right edge of the domain is moved to the left. The right edge of the domain forms a slight arc from top to bottom in both simulations. It should be noted that [82] uses the pole-zero strain energy function, rather than the Mooney-Rivlin strain energy function used in this thesis.

The simulations in Figure 4.9 were undertaken on a coarse mechanical mesh with 3198 degrees of freedom in the mechanical system and 28609 nodes in the electrical

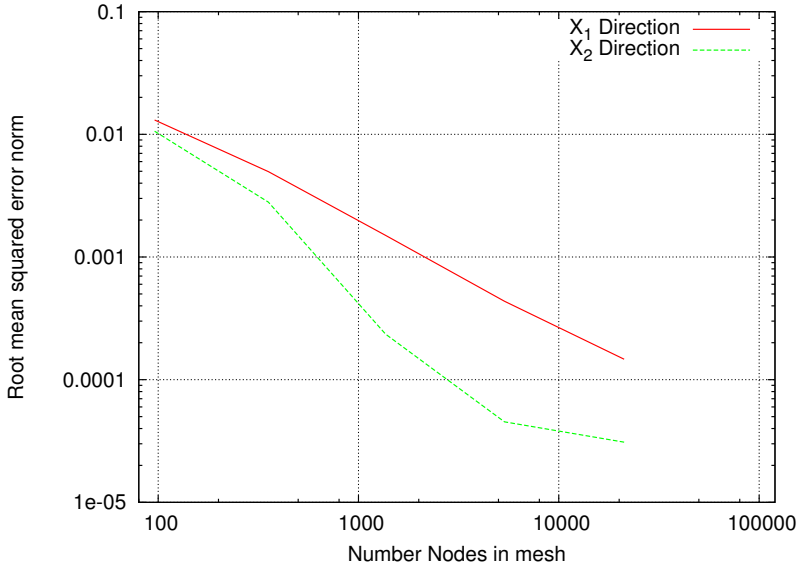


Figure 4.6: *Mechanical deformation convergence, with varying mesh resolutions, X_1 and X_2 RMS error of the 96 coincident points compared with a mesh with 84385 nodes. Tests undertaken on a 120 mm $\times$ 120 mm domain with varying mesh refinement. Meshes fixed along edge where $X_1 = 0$, and K_{T_a}=4000 KPa. One mechanical solve undertaken.*

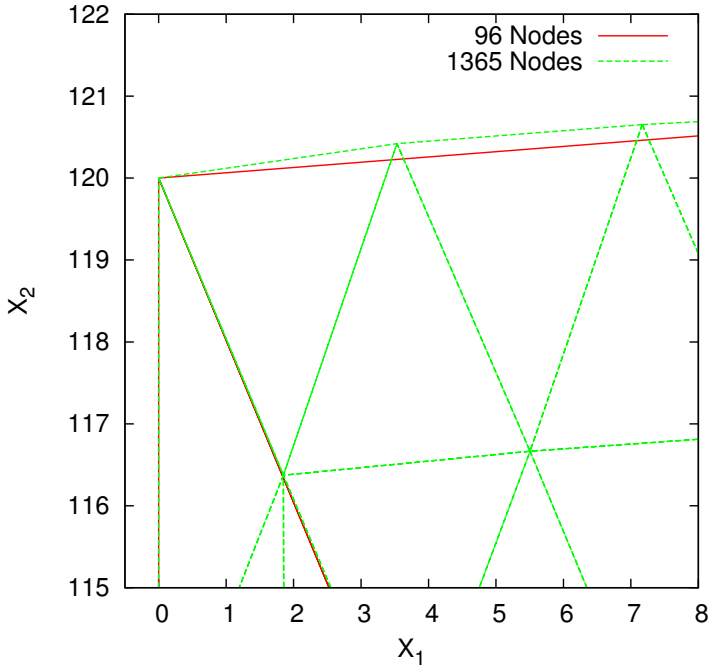


Figure 4.7: *Mechanical deformation showing element deformation, with initial mesh nodes increasing from 96 to 136. Tests undertaken on a 120 mm $\times$ 120 mm domain with varying mesh refinement. Meshes fixed along edge where $X_1 = 0$, and K_{T_a}= 4000 KPa. One mechanical solve undertaken.*

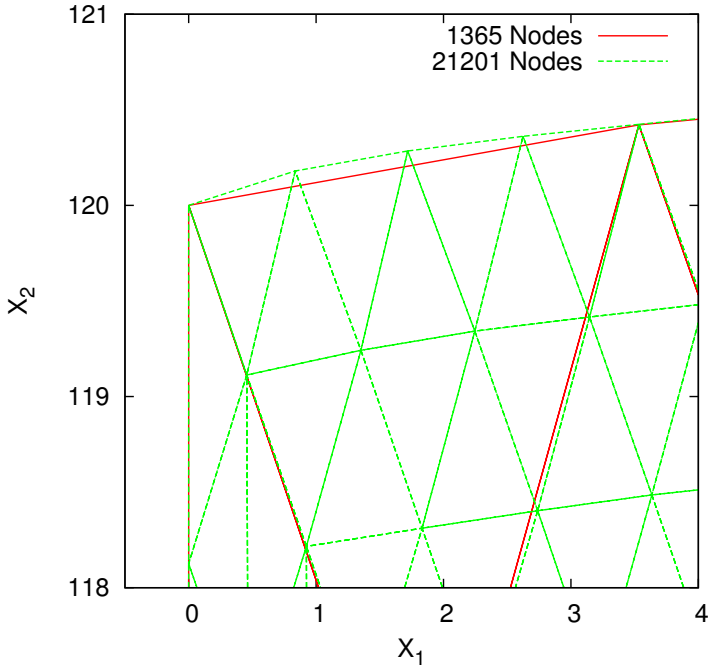
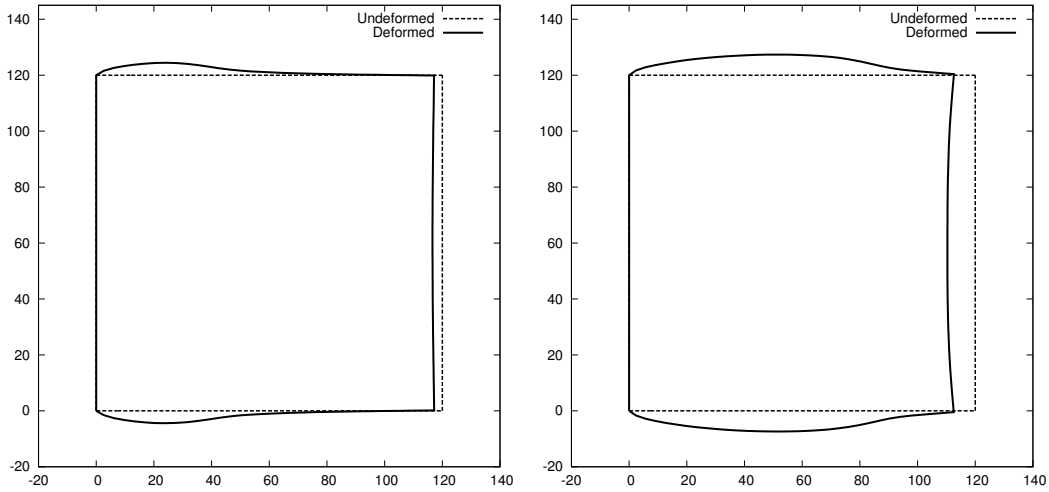


Figure 4.8: *Mechanical deformation showing element deformation, with initial number of nodes in mesh increasing from 1365 to 21201. Tests undertaken on a 120 mm* $\times$ *120 mm domain with varying mesh refinement. Meshes fixed along edge where* $X_1 = 0$*, and* K_{T_a}*=4000 KPa. One mechanical solve undertaken.*



(a) t=50ms (b) t=100ms

(c) t=150ms (d) t=200ms

Figure 4.9: *Mechanical deformation caused by an electrical wave, with left edge $X_1 = 0$ fixed and stimulated to form a plane wave. This is for comparison with Figure 1 of [82].*

system. In these simulations the electrical wave speed was 0.92 m/s. It has been observed previously (see Chapter 2) that the electrical wave is not fully resolved on coarse meshes and these results further demonstrated a correlation between electrical wave speed and mesh resolution. The increased wave speed is present in Figure 4.9.

4.4 Deformation and spiral wave stability

Tests were undertaken to determine the effects that a coupled deforming domain would have on the stability of the electrical spiral wave. This work is also presented in [57]. The simulations were run on a 120 mm $\times$ 120 mm domain. For the electrical simulations the domain is divided into 634368 unstructured triangular elements using 318065 nodes. This gives an approximate element edge of 0.21 mm. For the mechanical simulations a mesh of 2478 unstructured triangular elements and 5067 nodes is used. These nodes are common to the electrical mesh. To prevent spurious rotation and translation, and hence preserve uniqueness of the solution, the node closest to the centre of the domain is fixed in both directions and a neighbouring node at the same vertical height is fixed in the X_2 direction.

In [110] the stability of the electrical spiral wave is investigated with varying restitution slopes and sodium dynamics. The objective in this section was to consider the results presented in Figure 7 of [110] and determine whether a coupled system would behave similarly. In these tests the sodium dynamics (I_{NA}) were set to the 'standard' value [110] and all simulations were initialised by running the simulations until $t = 5000$ ms with a restitution slope of 1.1. This is to ensure a stable spiral wave has formed. After $t = 5000$ ms the simulation settings were amended to introduce the conditions being tested.

Figure 4.10 shows the results of the solver with electromechanical coupling disabled and enabled. Both these simulations were undertaken for 10000 ms (from t=0 ms) with a restitution slope of 1.1. In the deforming domain for this restitution slope the spiral wave is similar to that formed in the static domain, and both spiral waves remain stable over the time period. The anisotropic diffusion set diagonally across the domain (from the origin to $X_1 = 120$mm, $X_2 = 120$mm) can be seen and the active tension is also set to act in this direction. This causes the tissue to contract along this direction and, as incompressibility is enforced, this causes the domain to expand in the direction of the diagonal perpendicular to this.

Once the spiral wave is established the deformation within the system forms a regular pattern, resulting in a 'kite-shaped' shaped domain. However, as the spiral wave rotates, the domain also moves around the central fixed point. Figure 4.11 illustrates this at three

(a) Static Domain (b) Deforming Domain

Figure 4.10: *Static and coupled electromechanical simulation of spiral wave with restitution slope of 1.1, at time t=10000 ms. In a 120 mm × 120 mm domain, divided into 634368 elements and using 318065 nodes. A fixed time step (dt) of* 0.08 *ms was taken and the diffusion tensor,* $\mathbf{D}$*, set as per Equation (2.37).*

different points in time (6000 ms, 8000 ms and 10000 ms). Figure 4.12 illustrates a spiral cardiac wave with a restitution slope of 1.8, for both a static and a deforming domain. In these simulations the restitution slope was set to 1.1 initially and then, after t=5000 ms, changed to 1.8. In both the static and deforming domains the introduction of the 1.8 restitution slope caused the spiral wave to break up very quickly. The images in Figure 4.12 are at t=6000 ms.

Figure 4.13 illustrates the effects the deforming tissue has on a spiral wave with a restitution slope of 1.4. It can be clearly seen that the deforming domain contributes the the spiral wave break-up. On the static domain the spiral wave is stable, however on the deforming domain the wave has started to become chaotic.

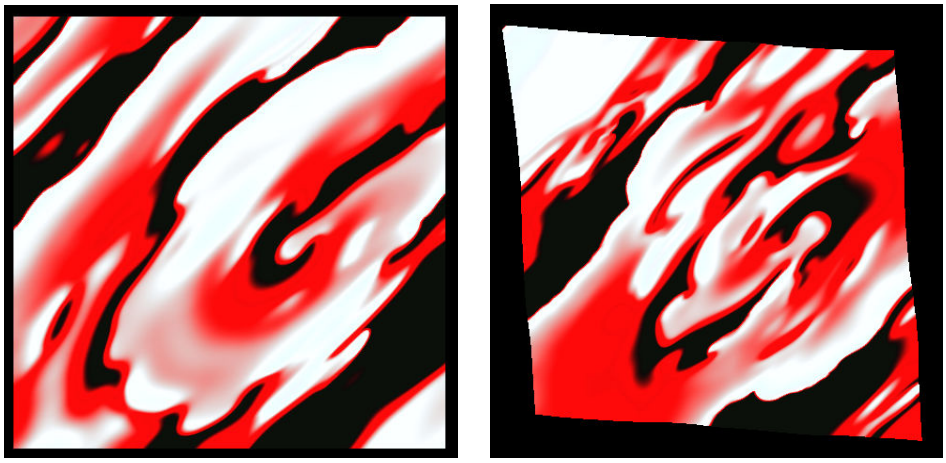
4.5 Conclusions

The electrical and mechanical systems were coupled together using the active tension variable (T_a), the evolution of which is governed by Equations (4.2) and (4.3), which use a phenomenological approach taken from [69].

The approach was validated qualitatively against the results of [82] and demonstrates that comparable deformations can be produced. Convergence tests were undertaken to consider mesh size and this demonstrated that node displacements converge, at a rate that is approaching quadratic, as the element size decreases.

(a) Time = 6000 ms (b) Time = 8000 ms (c) Time = 10000 ms

Figure 4.11: *Coupled electromechanical simulation of spiral wave with restitution slope of 1.1, at times t=6000 ms, t=8000 ms t=10000 ms. In a 120 mm × 120 mm domain, divided into 634368 elements and using 318065 nodes. A fixed time step (dt) of* 0.08 *ms was taken and the diffusion tensor,* $\mathbf{D}$*, set as per Equation (2.37).*



(a) Static Domain (b) Deforming Domain

Figure 4.12: *Static and coupled electromechanical simulation of spiral wave with restitution slope of 1.8, at time t=6000 ms. In a 120 mm × 120 mm domain, divided into 634368 elements and using 318065 nodes. A fixed time step (dt) of* 0.08 *ms was taken and the diffusion tensor,* $\mathbf{D}$*, set as per Equation (2.37).*

(a) Static Domain (b) Deforming Domain

Figure 4.13: *Static and coupled electromechanical simulation of spiral wave with restitution slope of 1.4, at time t=8000 ms. In a 120 mm × 120 mm domain, divided into 634368 elements and using 318065 nodes. A fixed time-step of* 0.08 *ms was taken and the diffusion tensor,* $\mathbf{D}$*, set as per Equation (2.37).*

The coupled solver was used to simulate changes in an electrical wave with varying restitution slopes of 1.1, 1.4 and 1.8 (as per [110]). These tests show that the deformation of the domain can affect the stability of an electrical spiral wave, with a previously stable wave breaking up into chaotic patterns. This illustrates that the modelling of cardiac electrophysiology on a static domain may not accurately represent the wave form as the deformation can contribute to spiral wave break-up.

Chapter 5

Preconditioning the mechanical system

5.1 Introduction

In Chapter 4 the electrical and mechanical systems were coupled together to produce a deforming model of cardiac tissue. The electrical system is presented in Chapter 2 and the mechanical system is presented in Chapter 3. When running the coupled solver the time to solve the mechanical system was far greater than the time to solve the electrical system. The objective of this chapter is to present the relative performance of the two components and to consider how preconditioning the mechanical solve can improve the overall solution time. The chapter puts forward results from preconditioning the system, with an initial set of preconditioning parameters and then investigates if these can be improved further.

Also it should be noted that throughout this chapter the performance comparisons were undertaken from the start of a simulation (where $t = 0\,\text{ms}$) onwards. In the ordinary differential equation (see Equation (4.2)), that is solved for the active tension variable (T_a), the initial value of T_a is zero and then it builds up over each time step (and is bounded by the maximum active tension parameter (K_{Ta})). This means that at the start of the simulations there is a relatively low amount of tension in the system, which results in smaller deformations, which in turn are easier for the Newton iteration to solve. The simulations in Chapter 7 are run for long time period to remove this issue.

Test no.	Mechanical Unknowns	Mechanical Solve Time (s)	Electrical Solve Time (s)	Performance Ratio
N1	1573	7.09	0.0064	111
N2	2355	15.74	0.0091	1739
N3	3652	34.29	0.0139	2461
N4	11429	279.27	0.0431	6480

Table 5.1: *Relative performance of mechanical and electrical solvers. Tests undertaken on a 120 mm $\times$ 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{Ta}) set to 4.79 kPa.*

5.1.1 Comparison of electrical and mechanical solvers

On average a non-preconditioned mechanical solve was taking up to 6400 times longer than a single electrical solve (see Table 5.1), on the same mesh. These tests were run on basic computing equipment (see B.1), and so the actual performance times are not of interest here, however the way in which the performance of the overall coupled solver is dominated by the mechanical solver component was an issue.

Another concern was that the mechanical solver does not scale linearly. In Table 5.1 the change from 3652 mechanical unknowns, in test N3, to 11429, in test N4, is an increase of 3.1. However the corresponding increase in mechanical solve time is 8.1. This would be problematic in using the solver for mechanical meshes of a finer resolution. The electrical system does, however scale linearly, with the time to solve the N4 test with 11429 nodes being approximately 7 times longer than the N1 case with 1573 nodes.

The performance of the mechanical solve at these levels prohibits running simulations on workstation level computing equipment. To undertake the heart failure simulations (see Chapter 7) there were in excess of 30000 mechanical solves undertaken (there was one mechanical solve undertaken for 10 electrical solves). For a mesh with 11429 unknowns this would take in excess of 96 days with the CPU time required in Table 5.1. Therefore a means of improving the performance of the mechanical solver was required.

It should be noted that this was not the main focus of the research, and so a simple solution was sought to enable simulations to be undertaken utilising days of workstation computing resource (rather than weeks or months).

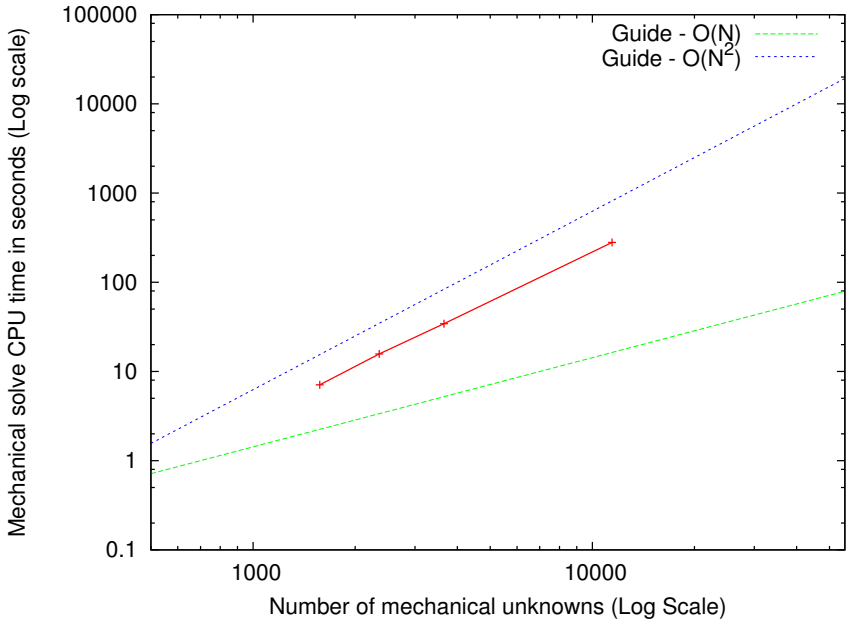


Figure 5.1: *Scaling of non-preconditioned mechanical solver. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{Ta}) set to 4.79 kPa.*

5.1.2 Recap of Newton method

The FEM approximation of the equations governing the mechanical system (as described in Chapter 3), results in a system of non-linear equations of the form:

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}, \tag{5.1}$$

where $\mathbf{f}$ is a system of n non-linear equations, $\mathbf{x}$ is a vector of the n unknowns and $\mathbf{0}$ is a vector of zeros of length n. The system of equations generated by the cardiac mechanics are highly non-linear and for this reason an iterative Newton-Krylov method is employed to solve them (as described in Chapter 3, Section 3.4.1).

Applying the Newton method for a system of equations (as described in Section 3.4.1) results in an equation for the vector of iterative updates, δ, given by:

$$\mathbf{J}_{\mathbf{f}(\mathbf{x}_m)}\delta = -\mathbf{f}(\mathbf{x}_m), \tag{5.2}$$

where $\delta = (\mathbf{x}_{m+1} - \mathbf{x}_m)$ and is a vector of n unknowns, $\mathbf{J}_{\mathbf{f}(\mathbf{x}_m)}$ is the Jacobian matrix calculated from the function vector $\mathbf{f}(\mathbf{x}_m)$ and m is the current iteration number.

An overview of the Newton Method is described in Process 3.1, with the solution of Equation (5.2) being used to calculate the next value of $\mathbf{x}_m$ (that is, $\mathbf{x}_{m+1} = \mathbf{x}_m + \delta$) used in the main 'outer' Newton iteration. Equation (5.2) is known as the 'inner solve', and we solve this linear system using the GMRES method [96] provided within SPARSKIT. To improve the performance of the GMRES solver we can apply a preconditioner [95, Chapter 10].

5.2 Preconditioning

The objective of preconditioning is to transform a system of equations into one with the same solution but which is easier to solve with an iterative technique [95]. The inner solve as described in Equation (5.2) contains the matrix $\mathbf{J}_{\mathbf{f}(\mathbf{x}_m)}$ and the format of this matrix has an effect on the performance of the solver. Preconditioning techniques modify the system prior to attempting the solve.

The system being solved is described in Equation (5.2). With right preconditioning this is rewritten as:

$$\mathbf{J}_{\mathbf{f}(\mathbf{x_m})}\mathbf{P}^{-1}\mathbf{P}\delta = -\mathbf{f}(\mathbf{x}_m), \tag{5.3}$$

where $\mathbf{P}$ is an n x n preconditioner matrix and $\mathbf{P}^{-1}$ is its inverse. Equation (5.3) is divided into two component parts and these are solved as two different systems, namely:

$$\mathbf{J}_{\mathbf{f}(\mathbf{x_m})}\mathbf{P}^{-1}\mathbf{y} = -\mathbf{f}(\mathbf{x_m}) \tag{5.4}$$

and

$$\delta = \mathbf{P}^{-1}\mathbf{y} \tag{5.5}$$

where $\mathbf{y}$ is a temporary vector of length n.

It was decided to firstly consider an ILUT preconditioner [95] and determine whether this could provide the required performance improvements.

5.2.1 ILUT preconditioning

LU factorization is a technique in which a matrix is factorized into a product of a lower triangular matrix and an upper triangular matrix. Incomplete LU factorisation uses an approximation to the full LU factorisation. ILUT factorisation is a version of ILU factorisation in which only non-diagonal matrix entries above a tolerance (hence the T) are kept. The abbreviation ILUT is derived from an **I**ncomplete **L**ower **U**pper factorisation with a drop **T**olerance. The method also includes a further parameter representing the number

of extra 'fill-in' (subsequently referred to as '*nfil*' in this thesis) entries on a row of the matrix. Following the factorisation and application of the drop tolerance, only the largest *nfil* non-diagonal values in a row are kept. For example, if you have set *nfil* to 20, the 20 largest entries on each row of the factorised LU matrices (after the drop tolerance has been applied) will be kept and the remaining entries set to zero.

One of the advantages of ILUT preconditioning is that the overhead of building the **L** and **U** matrices can be controlled. By adjusting the *drop tolerance* and *nfil* parameters the number of row entries within the **L** and **U** matrices can be altered and hence the effort required to build them is reduced. The computational penalty of reducing the number of row entries in the **L** and **U** matrices, is that it lowers the accuracy of the approximation that the **LU** factorisation has to $\mathbf{J}_{\mathbf{f}(\mathbf{x}_m)}$. This may require the Newton solver to undertake more inner iterations to reach a solution within the required tolerance. Therefore setting these parameters is a balance between memory, **LU** build time and solve performance, and this is discussed further in Section 5.3.1.

5.2.2 Building the Jacobian and solving the system

To undertake the preconditioning it is necessary to calculate an n x n Jacobian matrix of the function vector $\mathbf{f}(\mathbf{x_m})$. The numerical Jacobian is calculated using the finite difference method with the process described in Process 5.1.

Process 5.1 Jacobian Build

1: Calculate the values of the function vector $\mathbf{f}(\mathbf{x}_m)$ for each value of $\mathbf{x}_m$.
2: Define a vector ε of length n (where n is the number of unknowns), where the i^{th} entry has a small value ($\varepsilon_0 = 1.0 \times 10^{-11}$) and all other entries are zero.
3: **For** $i = 0$ to n
4: Calculate the finite difference approximation for the i^{th} column of the Jacobian

$$\frac{\mathbf{f}(\mathbf{x}_m + \varepsilon_i) - \mathbf{f}(\mathbf{x_m})}{\varepsilon_0} \tag{5.6}$$

Note: When x_i is perturbed (which corresponds to an unknown relating to a node in the mesh), it is only necessary to recalculate the function vector, $\mathbf{f_i}$, for the elements of the mesh that node x_i is a member of. The values of $\mathbf{f_i}$ will not change for nodes not connected to the node being perturbed. This also means that when undertaking the finite difference calculation it is only necessary to process the rows of $\mathbf{f_i}$ that have been changed on the i^{th} step.

To solve the non-linear mechanical system KINSOL is used, as described in Section 3.1. When right preconditioning is required it is necessary to provide KINSOL with

a preconditioning 'set-up' function and a 'solve' function. The set-up function calculates the numerical Jacobian and then passes it to the ILUT preconditioner in SPARSKIT. The solve function then utilises the ILUT preconditioned GMRES solver provided by SPARSKIT. To interact with SPARSKIT it is necessary to store the Jacobian in a sparse matrix format (see Section 2.5.4). Following the addition of preconditioning most of the functions provided by KINSOL have been replaced, however the KINSOL code was kept as a skeleton and still provided the outer Newton iteration.

KINSOL is relatively complex to interact with so, for this reason, the steps necessary to undertake a solve with right preconditioning in KINSOL are described in Process A.1 in Appendix A.

5.3 Testing preconditioned performance

To test the performance of the mechanical solver with the preconditioner enabled, a set of tests were undertaken as described in Table 5.2. The tests in Table 5.2 were all carried out

Test No.	Preconditioner (on/off)	Number unknowns	Mechanical Solve Time (s)	Electrical Solve Time (s)	Performance Improvement Ratio
N1	off	1573	7.09	0.0064	
P1	on	1573	0.75	0.0061	9.436
N2	off	2355	15.74	0.0091	
P2	on	2355	1.67	0.0091	9.442
N3	off	3652	34.29	0.0139	
P3	on	3652	2.95	0.0138	11.616
N4	off	11429	279.27	0.0432	
P4	on	11429	14.43	0.0435	19.353

Table 5.2: *Testing the preconditioned system. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa. Drop tolerance was 0.0001, maximum row fill-in 296 and Jacobian re-use 5.*

on the same computer (see Section B.1). The electrical solve times are shown as a means of verifying the reliability of the tests. As no changes were made to the electrical solver these times should remain constant even though the tests were carried out at different times. The preconditioned solver has three parameters, namely the *drop tolerance*, the maximum fill-in parameter (*nfil)* and the Jacobian re-use parameter. These parameters are

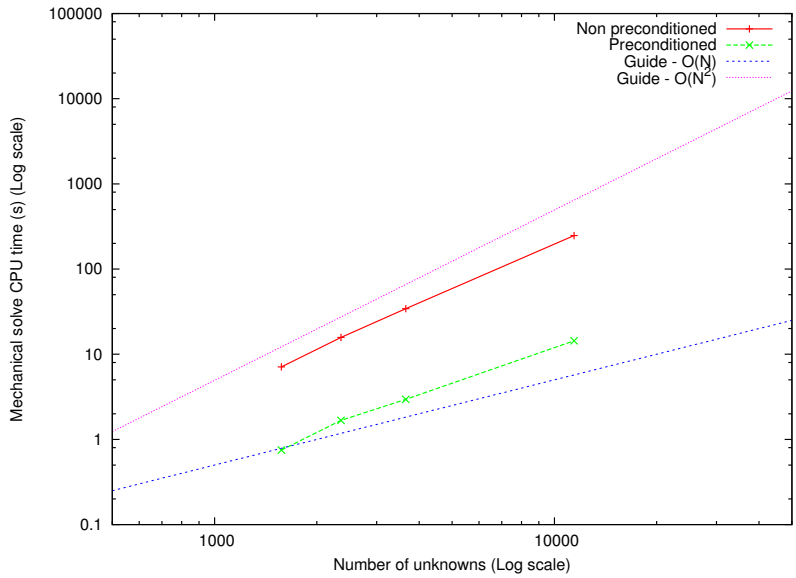


Figure 5.2: *Mechanical solve performance with and without preconditioning. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{Ta}) set to 4.79 kTa. For preconditioned solve the drop tolerance was 0.0001, maximum row fill-in 296 and Jacobian re-use 5.*

discussed in detail below and further tests were undertaken to consider the best values for these parameters. However, for the initial tests described in Table 5.2, these were set to general values which, from experience, would provide reasonably good results. Specifically for all the preconditioned solves, the *drop tolerance* was 0.0001, the fill-in parameter (*nfil)* was set to 296 and the Jacobian re-use parameter was set to 5.

The mechanical solve times from Table 5.2 are plotted in Figure 5.2 on a log scale. As can be seen from Table 5.2, the implementation of an ILUT preconditioner has significant performance improvements over a non-preconditioned system. For a system with 11429 mechanical unknowns the preconditioned solver outperforms the non-preconditioned solver by over 19 times. The results also show that the benefits of preconditioning increase with the system size, for a system with 1573 mechanical unknowns the performance improvement is 9.436 times. For the preconditioned solve, the change in system size from 3652 to 11429 is an increase of 3.1, and the increase mechanical solve time increasing by a factor 4.9, however the non-preconditioned solve increases by approximately 8.1 times. Figure 5.2 illustrates that, although the preconditioned solver is significantly faster for these meshes, the scaling of both solvers is between O(n) and O(n^2), with the preconditioned solver scaling slightly better.

Test No.	Preconditioner (on/off)	Unknowns	Average Outer Iterations	Average Inner Iterations per outer solve	Total Inner Iterations
NI1	off	1573	7.55	300.29	70268
PI1	on	1573	9.1	8.07	2276
NI2	off	2355	9.16	385.94	109606
PI2	on	2355	10.29	10.92	3485
NI3	off	3652	11.58	435.65	156399
PI3	on	3652	9.61	13.35	3977
NI4	off	5889	16.97	490.10	257791
PI4	on	5889	11.55	17.54	6281
NI5	off	11429	29.19	535.16	484318
PI5	on	11429	10.94	24.02	8143

Table 5.3: *Testing the preconditioned system. Number of iterations for 31 mechanical solves. Tests undertaken on a 120 mm $\times$ 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa.*

Table 5.3 shows the average number of outer solves for each mechanical solve and the average number of inner solvers for each of these outer solves. The performance improvements by the preconditioner can be seen by the large reduction in the average number of inner solves per outer solve, with the total number of inner iterations reducing from 484318 (in test NI4) to 8143 (in test PI4). Also as the mesh refines and there are more unknowns the number of outer solves required increases significantly for the non-preconditioned solve.

Within the solver that has been have developed, there are three parameters which modify the behaviour of the ILUT precondioner, namely the *drop tolerance*, the row fill-in (*nfil*) and the Jacobian/Preconditioner re-use. Altering these parameters can affect the preconditioned system performance and further investigation was undertaken to quantify the improvements possible.

5.3.1 Varying preconditioner drop tolerance

During the ILUT process row entries that are smaller than a pre-defined tolerance are removed ('dropped') from the resulting matrix by setting them to zero. Changing the *drop tolerance* setting therefore changes the number of non-zero entries within the LU approximation. The smaller the value of the *drop tolerance* the more row entries are kept and hence the more accurately the LU factorisation approximates the original matrix (in

Test No.	Number unknowns	Drop Tolerance	Mechanical solve time (s)	Average inner iterations	Average outer iterations
P-D1	11429	0.00005	21.62	20.40000	5.83
P-D2	11429	0.00010	14.43	24.12684	5.65
P-D3	11429	0.00050	11.72	42.00299	5.57
P-D4	11429	0.00100	12.18	54.31322	5.80
P-D5	11429	0.00500	34.31	178.16519	5.65

Table 5.4: *Performance with varying drop tolerances. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa. The maximum row fill-in was 296 and Jacobian re-use 5.*

this case $\mathbf{J}_{\mathbf{f}(\mathbf{x}_m)}$). A smaller *drop tolerance* produces a progressively more accurate LU factorisation, however it requires more time and memory to build.

To consider the effect that amending the *drop tolerance* had on the overall performance the tests in Table 5.4 were undertaken. In these tests the Jacobian/Preconditioner re-use parameter was set to 5 and *nfil* was set to 296 (both of these parameters are discussed further below). For the given problem, Table 5.4 shows that setting a drop tolerance of 0.0005 produced the best performance in terms of solve time. It also illustrates that an incorrectly set drop tolerance can have a significant effect on the performance, with a drop tolerance of 0.00005 performing almost twice as slowly, and a tolerance of 0.005 performing nearly three times as slowly.

The best performing *drop tolerance* is neither the smallest nor largest, and this is due to the time required to undertake the ILUT factorisation. Figure 5.3 shows how varying the *drop tolerance* affects the number of iterations required for each inner solve. If we set a very small drop tolerance the inner solves require fewer iterations to converge, however this has the cost of a much longer build time for the preconditioner. Building the preconditioner is a time consuming exercise and the time this takes is directly controlled by the size of the drop tolerance. The smaller the drop tolerance the longer it takes to build the preconditioner (see Figure 5.5). To maximise performance, the best result is achieved by not setting the drop tolerance too low, as the performance benefits of fewer inner solve iterations (each of which takes a relatively small amount of computational time) are far outweighed by the CPU time required to build the preconditioner.

Figure 5.5 also shows the time to build the Jacobian and this remains constant throughout these tests, as the Jacobian build time relates to the size of the system, not the *drop*

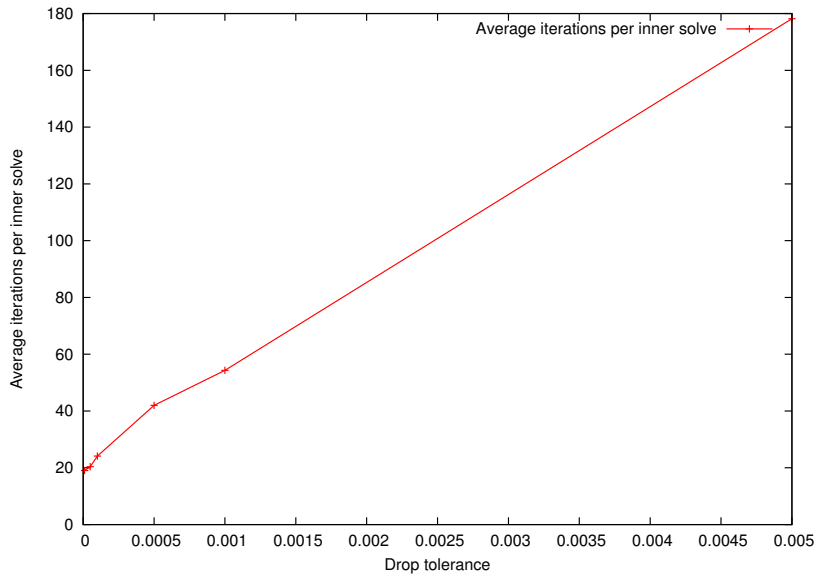


Figure 5.3: *Average inner iterations with varying drop tolerance. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa. The maximum row fill-in was 296 and Jacobian re-use 5.*

tolerance. However it does illustrate that as the *drop tolerance* becomes larger the time to build the Jacobian becomes progressively more significant, and hence an efficient Jacobian build algorithm is essential.

The objective of the preconditioner is to improve the performance of the inner solve (see Equation (5.2)), however we also measured the number of outer solves required by the Newton solver. Figure 5.4 shows that the number of outer solves remains approximately constant with a varying drop tolerance.

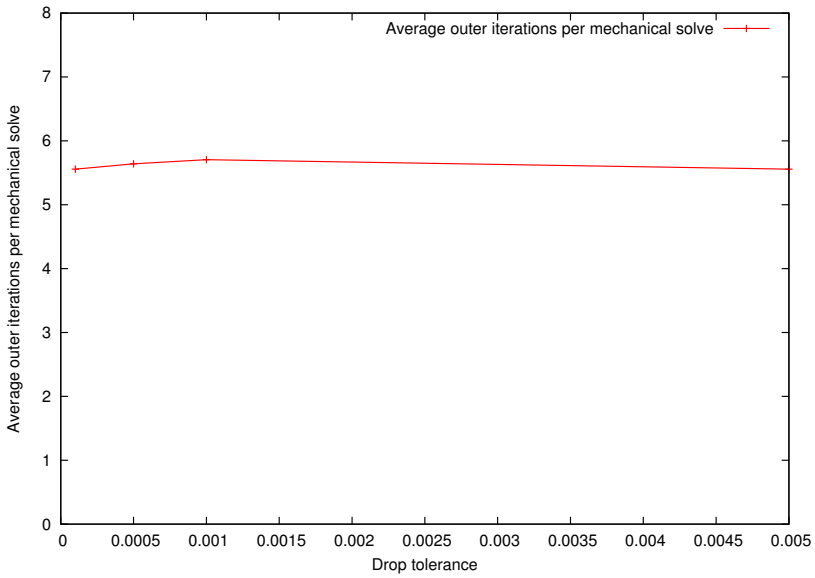


Figure 5.4: *Outer iterations with varied drop tolerance. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa. The maximum row fill-in was 296 and Jacobian re-use 5.*

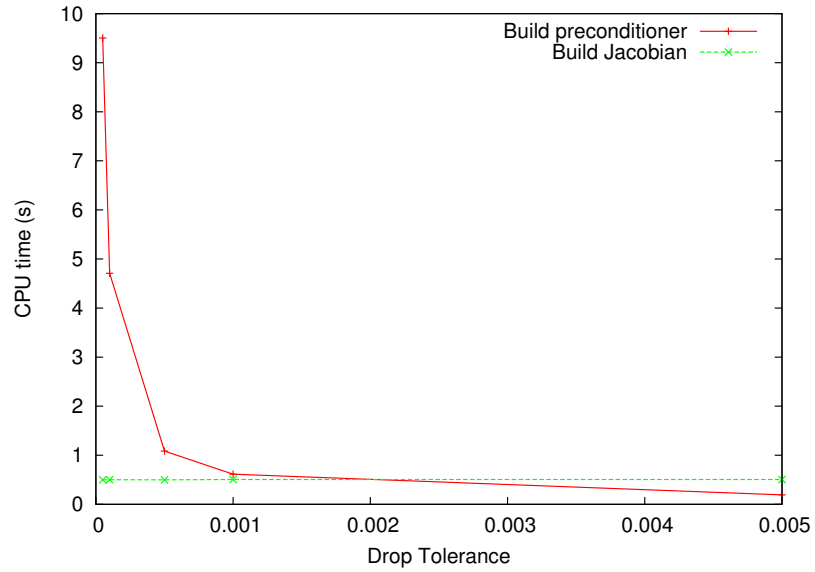


Figure 5.5: *CPU time to build Jacobian and preconditioner. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa. The maximum row fill-in was 296 and Jacobian re-use 5.*

5.3.2 Modifying the maximum row fill-in

The next parameter to consider was the ILU matrix row fill-in (*nfil*) parameter. This sets the maximum number of entries on any row in the LU factorisation and is applied after the drop tolerance parameter. That is, any values lower than the drop tolerance are removed (set to zero) and then the largest *nfil* of these are kept. The remaining entries are then set to zero (and effectively discarded as they are not held in the sparse structure). To consider this variable the tests in Table 5.5 were undertaken. In these tests the Jacobian re-use parameter was set to 5 and the *drop tolerance* was set to 0.0005.

These results show that varying the *nfil* parameter can alter the performance of the solver. If *nfil* is too low, the performance of the solver degrades. If you consider the ILUT process, the *drop tolerance* takes precedence over the row fill-in and so there will come a point where no further row fill-ins are required. That is, once the value of *nfil* is sufficient for the entries that the drop tolerance determines, having a larger *nfil* has no further effect, and this is demonstrated in Figure 5.6 and Figure 5.7.

Test No.	Number unknowns	Row fill-in	Mechanical Solve Time (s)
P-F1	11429	20	18.60
P-F2	11429	40	12.22
P-F3	11429	80	11.82
P-F4	11429	160	11.60
P-F5	11429	240	11.47
P-F6	11429	400	11.38
P-F7	11429	592	11.51

Table 5.5: *Modifying matrix row fill-in maximum parameter. Tests undertaken on a 120 mm $\times$ 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa. The drop tolerance was 0.0005 and Jacobian re-use 5.*

Figure 5.7 shows the effect varying *nfil* has on the number of inner solve iterations. We can see that using a larger *nfil* initially provides benefits in lowering the number of inner solve iterations, however beyond a point this provides no further benefit, again showing that beyond a point further increases to $nfil$ have no effect as extra entries are not needed for the specific *drop tolerance* applied.

Further tests were undertaken with 11429 unknowns, including testing with a low drop tolerance (0.0001) and a high *nfil* parameter of 1200. With these parameters the average inner iterations drops to 11.95 per outer solve. Although this did not improve the

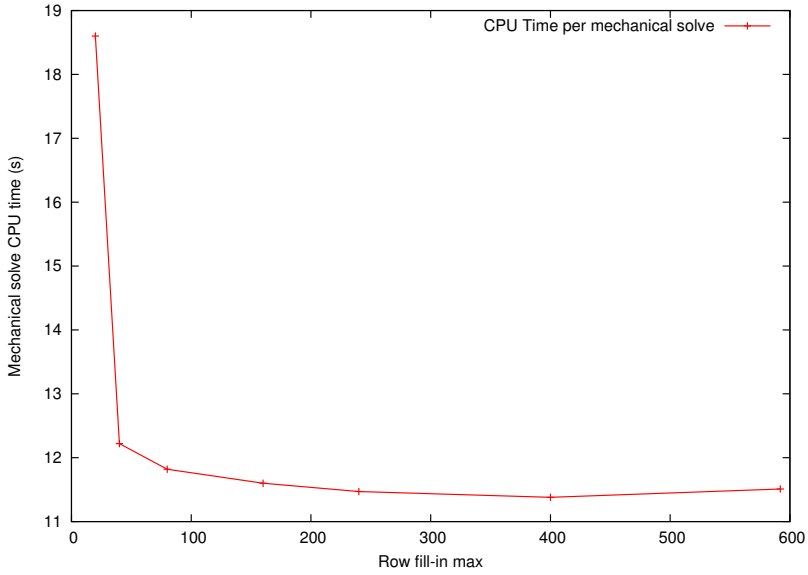


Figure 5.6: *Solver performance with varying row fill-in maximum. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa. The drop tolerance was 0.0005 and Jacobian re-use 5*

performance of the solver, it demonstrated that with an accurate ILUT preconditioner the number of inner iterations for a system with a higher number of unknowns is comparable to that from a smaller system (for example test PI2 in Table 5.3).

5.3.3 Re-use of numerical Jacobian

The inner system (see Equation (5.2)) uses the numerical Jacobian that is generated as per Process 5.1. As this is, by definition, an approximation to the true system Jacobian, this approximation can be used more than once. By re-using the Jacobian the solve technique being undertaken is the Modified Newton method. Each time it is re-used the approximation becomes less accurate and so the solver will take more iterations to solve, however this might be outweighed by the performance savings gained by not recalculating the Jacobian. If the Jacobian is not re-calculated there is no point in building the ILUT factorisation either, as the lower and upper triangular matrices are factorised from the Jacobian. Figure 5.5 illustrates the CPU time involved in these two tasks and so it was felt further benefits could be obtained by not rebuilding these for every outer iteration.

Within the software a parameter has been added that controls how frequently the Jacobian is recalculated and the ILUT factorisation is then only undertaken when the Jacobian

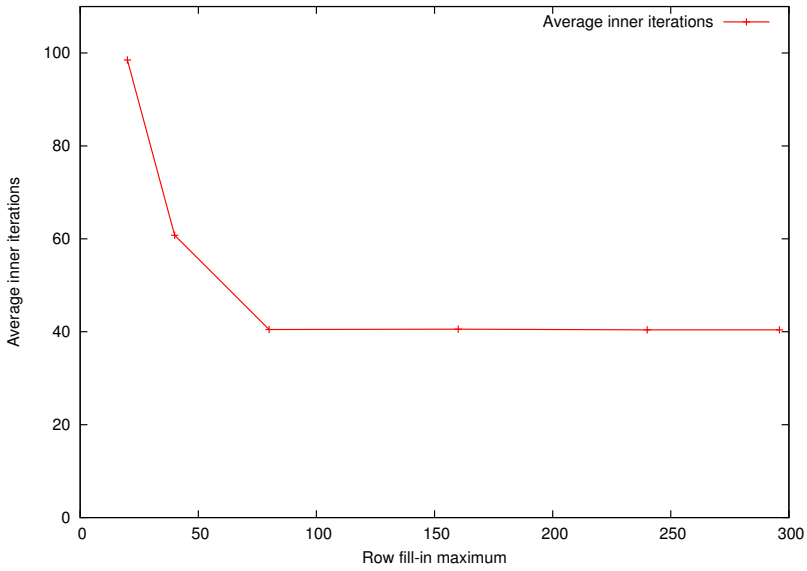


Figure 5.7: *Average iterations per inner solve with varying row fill-in. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa. The drop tolerance was 0.0005 and Jacobian re-use 5*

is re-built.

The tests in Table 5.6 were undertaken to see how re-using the Jacobian would affect the overall mechanical solve performance. In these tests the *nfil* parameter was set to 296 and the *drop tolerance* was set to 0.0001. In Table 5.6 we can see that modifying the Jacobian re-use parameter can have a significant effect on solve performance time. By re-using the Jacobian we can improve the performance by a factor of almost 2.

By not re-building the Jacobian on every outer iteration it would be expected that the number of outer iterations required would increase. This can be seen in Table 5.6, which shows that the average outer iterations increases as we increase the Jacobian re-use. There comes a point when the number of outer solves is less than the Jacobian re-use parameter and any extra increases in the Jacobain re-use have no further effect. It should be noted that the preconditioning work was undertaken mid-way through the thesis and recent publications (for example [58], which uses a technique from [60]) utilise a more sophisticated algorithm for Jacobian re-use based on monitoring the number of iterations taken.

Test No.	Number unknowns	Jacobian Re-Use	Mechanical solve time (s)
P-J1	11429	1	24.81
P-J2	11429	2	18.92
P-J3	11429	5	14.43
P-J4	11429	10	13.13
P-J5	11429	15	13.31
P-J6	11429	20	13.20

Table 5.6: *Re-using the numerical Jacobian. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa. The drop tolerance of 0.0001, row $nfil$ set to 296 and Jacobian re-use 5*

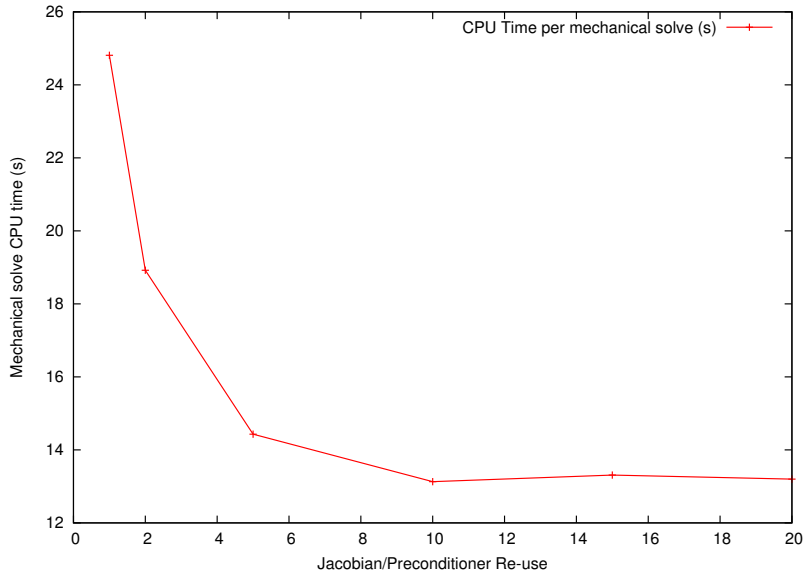


Figure 5.8: *Solve performance with varying Jacobian/Preconditioning re-use. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa. The drop tolerance was 0.0001, row $nfil$ set to 296 and Jacobian re-use 5*

Test No.	Number unknowns	Jacobian Re-Use	Drop Tolerance	Row Fill-in	Mechanical Solve Time (s)
P-C1	11429	5	0.0005	400	10.003

Table 5.7: *Combining preconditioning parameter changes. Tests undertaken on a 120 mm × 120 mm domain, with the central point fixed in both directions and an adjoining point fixed in one direction. The left edge of the domain was stimulated for $t < 1$ ms and the active tension maximum (K_{T_a}) set to 4.79 kPa.*

5.3.4 Combining parameter changes

In the above tests only one parameter was modified at a time, and by combining changes further improvements in performance can be achieved. Table 5.7 has the results from combining the best features of the previous tests.

In Figure 5.9 a selection of the above tests results are displayed, showing that even the poorly performing tests with preconditioning (e.g. P-D5 uses a higher than optimal *drop tolerance*) still outperforms the non-preconditioned test (N4) by over ten times. If we refine the parameters for the preconditioner we can improve performance further and the best performing test in the examples (P-C1) outperforms the non-preconditioned solve by 27 times. This also shows that by optimising the preconditioning parameters, in this instance, the mechanical solve scaled linearly. The mechanical solve time for the change in number of unknowns from 3652 to 11429 increased by 3.39.

5.4 Conclusions

The tests in Section 5.3 demonstrate that implementing an ILUT preconditioner within the mechanical solver has significant performance benefits. In the best performing example the preconditioned system outperformed the non-preconditioned system by over 27 times. To model cardiac tissue (especially the electrophysiology) it is necessary to use meshes with small element sizes and this in turn puts a constraint on the time step that you can use to move forward in time. In the simulations of end-stage heart failure in Chapter 7 we have undertaken in excess of 30000 mechanical solves (and ten times as many electrical solves), and the versions of these simulations with tissue deformation enabled took over 7 days to undertake on ARC1 (see Appendix B.3) with preconditioning enabled. It would be impractical to leave these tests running for over 180 days using a non-preconditioned solver.

Similarly the demands on the iterative solver vary with the tension in the system, and

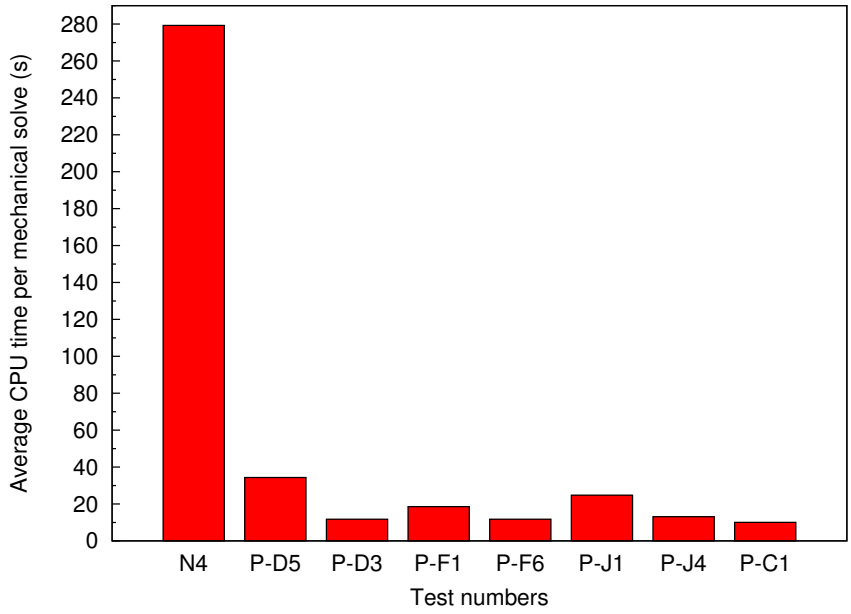


Figure 5.9: *Mechanical solve times, comparing non-preconditioned results to various preconditioned results taken from the tables in this section.*

the deformation in the system varies over time. Whilst running the heart failure simulations there were instances where the mechanical solver failed to converge. A 'checkpointing' mechanism was added to the software to periodically store the values of all the system variables. This meant that the situations where the system failed to converge, the preconditioning parameters could be amended (for example lowering the *drop tolerance*) and then the solver restarted from the last saved checkpoint. This enables an aggressive set of parameters to be used to maximise initial solver performance which can then be refined if required, without the need to restart the simulations from the beginning.

There is a significant computational resource required to undertake the ILUT factorisation and this increases as the *drop tolerance* is reduced and the row fill-in maximum is increased. The lower the drop tolerance and the higher the fill-in the more memory is needed to hold the ILUT matrices and the longer it takes to build them. However the more accurate the ILUT matrices the fewer iterations are required to undertake the inner solve. Also, if these become too inaccurate the system will fail to converge. Each inner solve does not take a relatively long time and so it is more efficient (in general) to have more inner solves and less time building the ILUT preconditioner.

Building the numerical Jacobian and undertaking ILUT factorisation is computationally costly and it was found that by re-using this Jacobian the overall solve performance could be improved.

All the tests in Table 5.3 were undertaken on a lightly stressed system, whilst the active tension is still building early in the simulations. It should be noted that as the active tension builds over time (up to the maximum set by the K_{T_a} parameter in the active tension equation (see Equation (4.2) or (7.2)), the stress also builds over time and this puts more load on the iterative solver and requires further iterations to solve. In the simulations undertaken in this thesis, a greater relative improvement by using the preconditioner was noticed as the simulations progressed.

Similar tests to the ones in Tables 5.4, 5.5 and 5.6 were undertaken with 3652 mechanical unknowns. For this system size the optimal *drop tolerance* was 0.001 and the optimal *nfil* was 40. This shows that there is no 'one' optimal value for these parameters, rather they vary with the size of the system being solved, however the optimal *drop tolerance* reduces as the system size increases.

At the completion of this section of the project it was felt that ILUT preconditioning provided significant performance benefits to the solver that enabled the heart failure simulations to be undertaken in several days, rather than months. It has been demonstrated that ILUT preconditioning provides significant performance improvements to the non-linear system of equations generated by the cardiac mechanical system.

Chapter 6

Mesh adaptivity

6.1 Introduction

In Chapter 2 the need for the electrophysiology to be simulated on a fine finite element mesh was discussed in detail. The steepness of the cardiac wave front (see Figure 2.10) requires small mesh elements to properly resolve the wave front. However using smaller elements increases the computational time and resources required. In Pathmanathan and Whiteley [82] they comment that for coupled mechano-electric simulations to be undertaken on realistic geometry then adaptive techniques will need to be employed.

Adaptive mesh refinement strategies seek to minimise the number of nodes (and elements) in a mesh, whilst still providing an accurate solution to the given problem. It has been discussed in Chapter 2 that the electrical wave requires a fine mesh to properly resolve it, however the gradient of the transmembrane voltage at the front of the wave (during de-polarisation) is much steeper than during the wave plateau and re-polarisation phases.

This chapter seeks to investigate whether an adaptive mesh refinement (AMR) strategy can be implemented that provides the refinement needed to model the wave front with a lower refinement elsewhere in the mesh. The objective will be to determine whether the whole wave needs to be finely adapted or whether refinement around the front of the wave is sufficient to maintain an acceptable level of accuracy, and then assess the potential efficiency benefits of this approach. It should be noted that in certain pathological

conditions, e.g. tachycardia, spirals waves are formed and this causes more of the domain to be excited. How this impacts on the efficiency of an adaptive solution this will also be considered.

Furthermore, in this chapter we will present results of the adaptivity on a deforming domain, where the adapted electrical system is coupled to the deforming mechanical system.

6.2 Local mesh adaptivity introduction

In Section 4.1.1 and in the previous chapters of this thesis, mesh refinement has been discussed, and the term 'mesh refinement' has been used to describe the global refinement of all the elements in a given mesh. Local adaptive mesh refinement (AMR) seeks to have varying levels of mesh refinement in different places within the mesh, to provide a more accurate approximation in areas where it is needed.

Adaptive methods have been widely used in engineering problems since their introduction in the 1970s [4, 12], and these are generally grouped into three strategies, described below:

- h-refinement is the addition/removal of extra elements where they are needed by sub-dividing or merging existing elements [22]. In the case of subdivision, an element is divided into a number of child elements. In the case of merging an element, existing child elements are merged back into their parent element.

- p-refinement is where the improvement in accuracy is achieved by increasing the order of the approximation functions used by the method, for example the weight/basis functions used by the finite element method [78, 119]. With p-refinement the number of degrees of freedom in an element increases and hence when this is undertaken the system size increases. Local p-refinement only increases the order of the approximation functions on certain elements [16].

- r-refinement is the movement of mesh nodes to areas of the domain where they are needed, without adding extra nodes to the mesh [5, 46]. With this technique the system size remains constant, however by moving nodes to areas of highest error the aim is to minimise the global error.

For the adaptivity undertaken in this thesis h-refinement is utilised. This method was chosen as it provides the flexibility of adding extra refinement where it is needed in the mesh, and because the number of refined elements to properly capture an electrical wave

is difficult to pre-determine. For example, the amount of the domain excited can vary greatly between line wave and spiral wave simulations. This would make r-refinement difficult to implement efficiently.

Modelling cardiac electrophysiology is a time dependent problem and this means that the mesh needs to be adapted multiple times as the electrical wave moves within the domain. Undertaking local adaptivity on time dependent problems is more complex than for a steady state problem and requires techniques to build the mesh repeatedly and also to re-calculate the finite element matrices.

6.2.1 Monitor functions

In this thesis h-refinement is used and the concept of adding extra elements 'where they are needed' needs to be clearly defined. This phrase is used to describe the areas of the mesh where local adaptivity can improve the accuracy of the solution. In an AMR system these areas are targeted to increase the refinement in areas which need more resolution or to decrease the refinement in areas which do not need extra resolution. The aim is that a specified level of accuracy can be achieved with lower resources.

With AMR, the selection of elements to be refined is made by finding the regions where there is the most activity or where the largest errors are incurred. In the case of modelling cardiac electrophysiology this is at the front of the wave, where there is a very steep slope from the resting to excited state. As the electrical system is time dependent it is necessary to have a flexible AMR process that will continually refine and de-refine the mesh as the wave moves across the domain. Another challenge for a time dependent problem with a moving wave is to ensure the mesh is refined 'in front' of the wave. The mesh needs to be refined before the wave crosses it, otherwise the wave front will be modelled initially on a coarse mesh.

Adaptivity has been widely employed in the field of engineering and there are several techniques for designing a function to select the elements to refine and these are discussed in [15]. One approach is to consider the activity of the variables within the domain and refine elements based on changes to these variables. For example Oden et al. [79] use the gradient of the density (for incompressible flow problems) to determine adaptivity. In [13], the local Mach number is used when considering results corresponding to flows around aerofoils.

A more sophisticated approach than relying on the change in solution variables is to develop a posteriori error estimates. A posteriori error estimation is a technique often used to determine areas to refine and a discussion of this is in [77, 116]. This requires error

estimates to be calculated across the mesh before the adaptivity can be undertaken. To calculate the error estimates a set of equations will need to be solved and this incurs extra work. Also in [123] they comment that calculating a posteriori error estimates requires problem dependent analysis and this may be difficult for monodomain equations with complex membrane dynamics.

The monitor functions used in this thesis are defined in Process 6.1 and Process 6.2 and are discussed later. These functions monitor the voltage changes in the domain to determine where to adapt the mesh. By utilising this approach it is relatively straightforward to add techniques to consider different adaptivity for the front and back of the wave and also whether the entire excited region needs adapting. Also, there is minimal computational cost of calculating these functions, as compared to more sophisticated error estimates.

6.3 Methods used

6.3.1 Data structure considerations

Developing algorithms to manage a locally adapting mesh, especially with a time-dependent problem, has a number of challenges not encountered in fixed meshes, or where the refinement is global and known in advance. In systems with a known size the data structures can be built at the commencement of the software and memory allocated accordingly. Programming languages may provide facilities to re-allocate memory or re-size arrays and these tools can be used for data structures that need to increase in size. These facilities will typically assist in the addition of entries at the end of the existing data structure. However, with a locally adapting mesh it is necessary to be able to remove data from any point in the existing data set and/or add new data to the end of the data set. The removal of arbitrary (that is, not predetermined) data from a large set of data causes problems for traditional array data structures, in that it is inefficient to leave gaps in the data and re-ordering a huge data set because a few items have been removed is inefficient.

A number of techniques are typically used to solve this problem including linked lists, tree structures (for example quad-trees [56]) and block structures (for example [107]). In this thesis, the solution used is a ‘linked-list’ methodology. In a FEM system the main data structures represent the nodes, elements, mesh and edges. By using linked-lists for these it is easy to remove individual items of data (for example one node), without having to re-order the remaining data set. With a linked-list each item of the data structure is declared individually and hence can be destroyed (and memory released) individually.

Each item in the data structure holds a link to the previous item and the next item (or null pointers for the first and last items). The disadvantage of a linked list data structure is that you cannot directly access a specific data item, rather you need to parse the data set from the beginning until the item is found. For systems using the FEM this disadvantage is reduced by the fact that most operations are required on all elements or all nodes. It is not necessary to locate an individual node and process it outside of one of the main system functions which loop all entries.

To mitigate the effort required in parsing through a linked-list it is recommended to hold pointers to any associated data for that item. For example an element would hold a pointer to the nodes it is constituted from. In an array-based system such associations can be achieved by holding the appropriate array index numbers (rather than a pointer). By using such pointers it is efficient to access the data associated with any linked list item directly (by following the pointer), rather than having to loop through a list to retrieve it. The effort required to re-develop the required data structures from an array-based system is not trivial.

6.3.2 Determining elements to refine and de-refine

In this thesis locally targeted h-refinement is used and with this it is necessary to target elements to refine. It is also essential to ensure that the wave front stays within the refined area of the mesh. If the wave front is simulated in a coarse area of the mesh then it will not be properly resolved and the wave speed will alter. This means that in determining the elements to be refined two processes need to be undertaken, one to select elements for refinement and one to project this refinement to the surrounding area to ensure the wave front stays within the adapted region.

Prior to starting a simulation it is necessary to ensure the initial stimulated area of the wave is within the refined region. Therefore a portion of the domain needs to be adapted prior to the commencement of the electrical solve. It should be noted that the initial refinement needs to be repeated for the maximum number of refinements allowed, and this ensures the initial wave is contained by elements at the finest resolution.

Once the initial adaptivity is completed, and the system starts to solve, further elements need to be targeted for refinement or de-refinement as the wave progresses. Two different techniques were implemented and these are summarised in Process 6.1 and Process 6.2. These algorithms are controlled by a number of thresholds and parameters and these are stated in Table 6.1. Figure 6.1 illustrates a single element, with voltages (V_1, V_2, V_3) that is referenced in these algorithms.

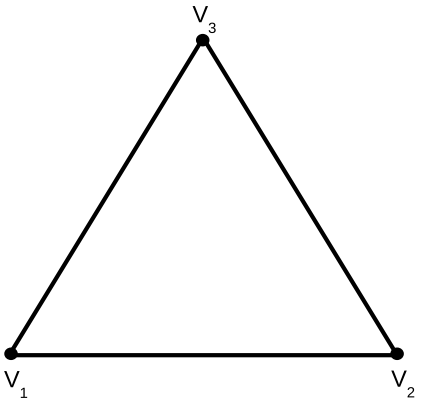


Figure 6.1: *Example triangle with voltages at the three corner nodes*

The mesh needs to be refined in front of the wave and to achieve this a secondary process is run that considers the neighbouring elements of the ones that have been flagged for refinement. A process is undertaken which determines which other elements are near the elements to be refined in the primary process and flags these for refinement. This process is included in Process 6.1 and Process 6.2. The distance of projection is controlled by a system parameter (P_{dist}) and the smaller this is made the fewer elements are refined on each refinement cycle. However if this distance is made too small the wave may 'break out' of the refined region, before the mesh is adapted again, and therefore it is essential that the distance parameter is related to the speed of the wave and the size of the elements.

Finally, there is also a parameter (A_{minE}), to determine whether elements within the excited region of the domain (i.e. elements where the voltage is greater than the resting potential), should be allowed to de-refine to the coarsest level. This parameter enables the entire excited area of the domain to be kept at a given minimum level of refinement.

Process 6.1 was implemented initially, and has a simple refinement test that compares voltage within an element. The same test is used throughout the mesh. The de-refinement process simply de-refines elements after a period of time. This monitor function provided a good technique to test the AMR software and also provided useful insight into the areas of the mesh that require most refinement. This is discussed in more detail below.

Process 6.2 applies a more sophisticated monitor function that allows distinction between rising and falling voltages. This effectively allows the front and back of the wave to be treated differently. The de-refinement process is also more robust and compares the voltages within an element. Using the time to control de-refinement, as in Process 6.1 is a crude technique that also needs optimising for different problems.

Name	Description	Units	Where used
A_{max}	Maximum number of levels of refinement	None	All
V_{tolR0}	Threshold for determining when to refine elements	mV	Refinement, Process 6.1
V_{tolR1}	Threshold for determining when to refine elements when voltage is rising	mV	Refinement, Process 6.2
V_{tolR2}	Threshold for determining when to refine elements when voltage is falling	mV	Refinement, Process 6.2
P_{dist}	Projection buffer distance to determine which additional elements to refine	mm	Refinement, Algorithms 6.1 and 6.2
A_{minE}	Minimum level of refinement for excited region	None	De-Refinement
V_{tolE}	Voltage threshold above which an element is deemed to be excited	mV	De-Refinement
T_{tolD}	Time threshold for determining when to de-refine elements	ms	De-Refinement - Process 6.1
V_{tolD}	Voltage threshold for determining when to de-refine elements	mV	De-Refinement - Process 6.2

Table 6.1: *Parameters and tolerances used in monitor functions for refinement and de-refinement*

Process 6.1 Monitor Function - simple

1: Loop over all elements in the mesh (E_i)
2: Refinement level of E_i is referred to as E_i^R
3: If $E_i^R = A_{max}$ then ignore this element
4: Else if $|V_1 - V_2| > V_{tolR0}$ OR $|V_2 - V_3| > V_{tolR0}$ OR $|V_3 - V_1| > V_{tolR0}$ then
5: Mark the element for refinement
6: Flag the refinement as 'primary'
7: Record the time refined in E_i^{RTime}
8: Loop over all active elements in the mesh (E_i)
9: Get the centre point of E_i, referred to as E_iC
10: Loop over the elements marked for primary refinement (E_p).
11: Get the centre point of E_p, referred to as E_pC
12: Calculate the distance between E_pC and E_iC, referred to as $Dist$
13: If $|Dist| < P_{dist}$ then flag E_i for refinement
14: Loop over all active elements in the mesh (E_i)
15: If E_i is flagged for refinement then ignore this element
16: Else if $E_i^R = 0$ then ignore this element
17: Else if the voltage at $E_i > V_{TolE}$ AND E_i refinement level equals A_{minE} then ignore
18: Else if $(E_i^{RTime} - T_{tolD}) >$ Current time, then flag for de-refinement

Process 6.2 Monitor Function - second

Loop over all elements in the mesh (E_i)
  Refinement level of E_i is referred to as E_i^R
  If $E_i^R = A_{max}$ then ignore this element
  Compare voltage in E_i with voltage at last time step
  If voltage is rising then
    If $|V_1 - V_2| > V_{tolR1}$ OR $|V_2 - V_3| > V_{tolR1}$ OR $|V_3 - V_1| > V_{tolR1}$ then
      Mark the element for refinement
      Flag the refinement as 'primary'
  Else if voltage is falling then
    If $|V_1 - V_2| > V_{tolR2}$ OR $|V_2 - V_3| > V_{tolR2}$ OR $|V_3 - V_1| > V_{tolR2}$ then
      Mark the element for refinement
      Flag the refinement as 'primary'
Loop over all active elements in the mesh (E_i)
  Get the centre point of E_i, referred to as E_iC
  Loop over the elements marked for primary refinement (E_p).
    Get the centre point of E_p, referred to as E_pC
    Calculate the distance between E_pC and E_iC, referred to as $Dist$
    If $|Dist| < P_{dist}$ then flag E_i for refinement
Loop over all active elements in the mesh (E_i)
  If E_i is flagged for refinement then ignore this element
  Else if $E_i^R = 0$ then ignore this element
  Else if the voltage at $E_i > V_{TolE}$ AND $E_i^R = A_{minE}$ then ignore this element
  Else if $|V_1 - V_2| < V_{tolD}$ AND $|V_2 - V_3| < V_{tolD}$ AND $|V_3 - V_1| < V_{tolD}$, then
    Flag for de-refinement

The results from these techniques are discussed in Sections 6.4.4, 6.4.5 and 6.4.6.

6.3.3 Checking the validity of the mesh

The processes described in Section 6.3.2 'flag' elements to be refined or de-refined. The refinement is not actually undertaken during that process, as there may be refinements that are flagged that cause issues with the validity of the mesh. A number of rules are applied to the refinement process as follows:

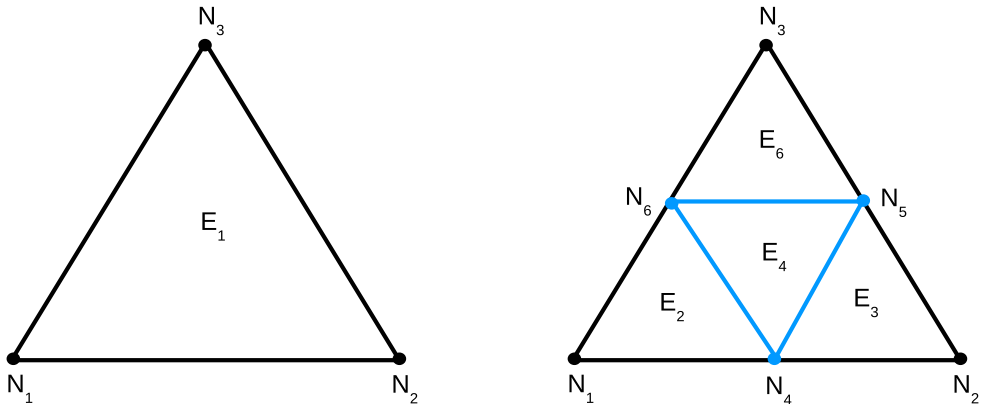
- No element can be de-refined further than its original starting level. This means that the original mesh provides the set of coarsest mesh elements.
- Maximum refinement (A_{max}). A maximum refinement threshold is set (for example 5 refinements) and elements cannot be refined further than this.
- Neighbour refinement. Two neighbouring elements can only differ in one level of refinement. For example, a level 5 element cannot be next to a level 3 element.
- Island refinement. An element should not be left as an island, whereby all its neighbouring elements are refined to a higher level than it is.
- De-refine together. When an element is marked for de-refinement, all its 'sibling' elements need to be de-refined with it.

The process of 'flagging' elements creates a proposed new mesh. The proposed new mesh is checked against the rules defined above and extra elements flagged for refinement or elements are blocked from de-refinement. This results in another new proposed mesh and the process starts again. This process continues until there are no elements flagged for refinement or de-refinement, and hence all the rules have been met.

6.3.4 Refining and de-refining the mesh

With h-refinement, selected elements are divided into new smaller elements. The approach used in this thesis is to divide each element into four new elements [61]. This is described further in this section.

Following the selection of elements to be refined and the mesh validation, the proposed new mesh is then actually refined into a new working mesh. Elements flagged for refinement are split into 4 child elements. The process of adapting the element illustrated in image (a) of Figure 6.2, is achieved as follows:



(a) Before refinement (b) After refinement

Figure 6.2: *Adapting a trianglular element in four new (child) elements*

- Three new nodes are created (N_4, N_5, N_6) at the mid-points of the element's edges. For example, N_4 is created at the mid-point between N_1 and N_2.
- At N_4, N_5, N_6, the values of the TP06 state variables, voltage and active tension, are set to the averages of the two nodes they are bisecting.
- Four new elements (E_2, E_3, E_4, E_5) are created by joining the new nodes together.
- The new elements and nodes can be seen in (b) of Figure 6.2.
- The original element, E_1, is marked as "in-active" and excluded from the FEM processes, until it is marked as "active" again during a de-refinement.

The de-refinement process takes four sibling elements (e.g. E_2, E_3, E_4, E_5) and removes them from the mesh. The process of de-refining the element illustrated in image (b) of Figure 6.2, is achieved as follows:

- Loop around the elements marked for de-refinement
- If E_2 is flagged for de-refinement, ensure that the three sibling elements, where E_3, E_4, E_5 are the siblings of E_2, are ALL flagged for de-refinement during the same adaptivity process.
- If not then do not proceed with the de-refinement.
- Remove the elements E_2, E_3, E_4, E_5 and nodes N_4, N_5, N_6 from the data structures and free the memory used.

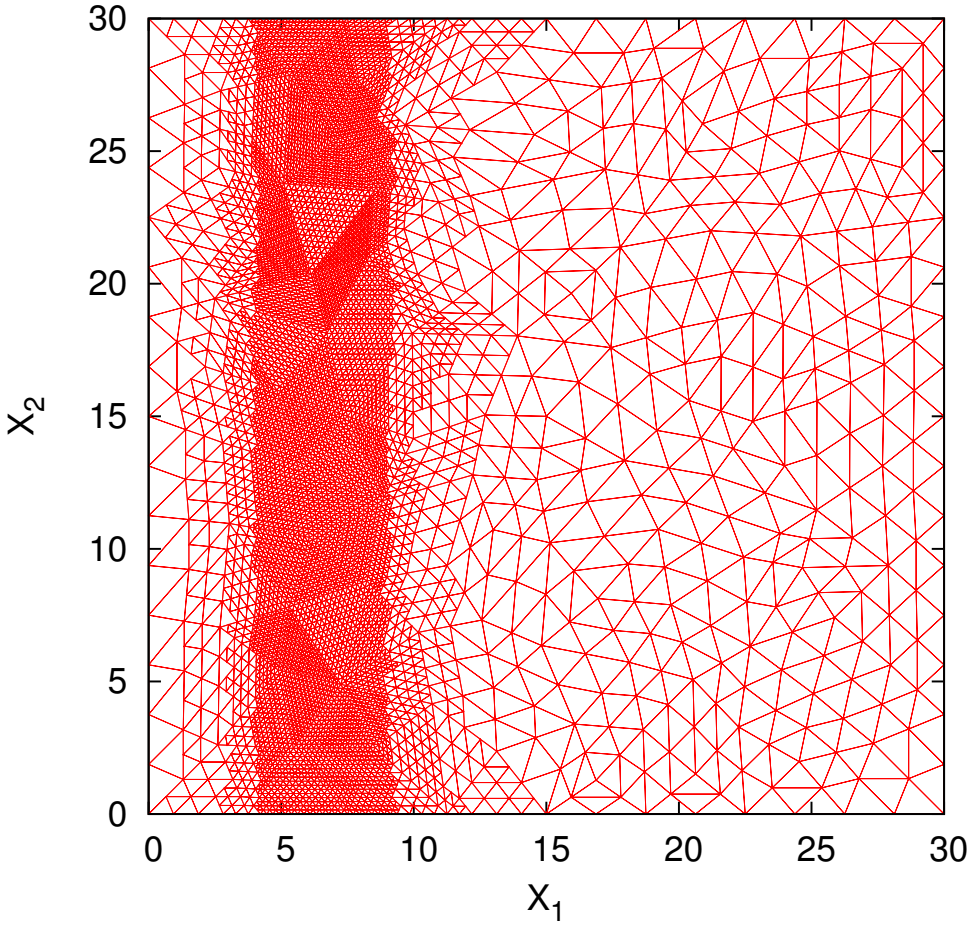


Figure 6.3: *Example AMR mesh with four levels of refinement*

- Flag the original parent element, E_1, as active again.
- This will result in returning to the element in image (a) of Figure 6.2.

An example of an adaptively refined mesh is given in Figure 6.3. This illustrates a mesh with four different levels of refinement in different areas of the mesh, and shows the mesh refining in a band to coincide with the electrical wave front.

6.3.5 Finite element method implications

The FEM operates on a given mesh and the key matrices (the global stiffness and global mass matrix - see Section 2.3.3) are spatially dependent. This means that when the mesh is refined they need to be rebuilt.

Following a refinement process the system size may have changed, which means that the data structures may need more memory allocating. To do this efficiently requires the

data structures to be easily increased or decreased in size and this has been discussed in Section 6.3.1. It also means that the sparsity pattern of the system will have changed and will need rebuilding.

In a globally refined mesh all interior nodes are common to a number of elements. However, following the refinement process in an AMR system, some triangle vertices will not be shared with other neighbouring elements, as certain elements will be more refined than their neighbours.

6.3.6 Hanging nodes

When a mesh is locally adapted it is an inevitable consequence that there will be regions of finer resolution next to regions of coarser resolution. Along this internal area of transition some nodes will only be in elements on one side of the threshold (the refined elements), and not be part of elements on the other side (see Figures 6.4 and 6.5). These nodes are known as hanging nodes. There are a number of techniques for dealing with

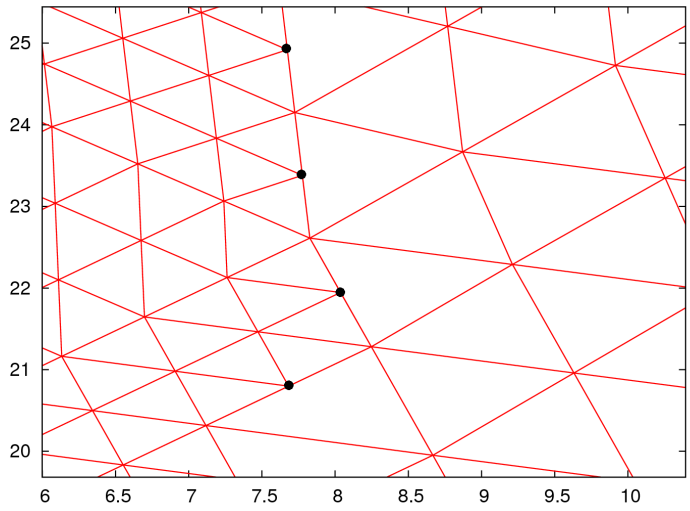


Figure 6.4: *An example of a section of a mesh with hanging nodes marked with black circles.*

hanging nodes, the most common being to add extra 'green' elements. These new elements are formed by the bisection of an element from the hanging node to the opposite vertex. For example, in Figure 6.5, two new triangles would be created by joining node 7 and node 4. There are a number of management issues with green elements, including their removal before the next adaptivity process [106, Section 3.3]. In this thesis it was decided to use a technique where the hanging nodes are dealt with algebraically. In [90],

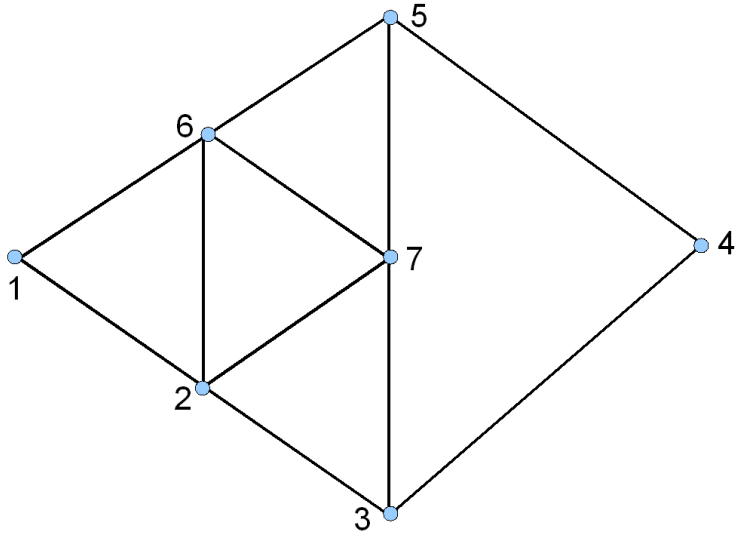


Figure 6.5: *Simple mesh with a hanging node*

a method is described whereby the solution at the hanging node is interpolated from the two neighbouring nodes. This is also used in [50, Chapter 4].

For the FEM to remain consistent the stiffness and mass matrices need to be built for all nodes (including hanging ones). The system is then re-arranged as per the theta method (see Section 2.5.3) to provide the system that requires solving. It is at this point that the hanging nodes are removed from the solve, and their values interpolated. However as these rows will have contributions from their neighbouring nodes, these contributions need to be distributed before they are removed from the solve, and this is described in Equation (6.1) and Equation (6.2). The hanging node rows are not actually removed from the system, rather their rows are set so that the value of the solution will equal the average value of the two adjacent nodes on its 'hanging edge', where a hanging edge is an edge on the transition between a two areas of different levels of refinement.

Figure 6.5 shows a simple example mesh with a hanging node. The FEM builds contributions to the global matrices and right hand side based on the connectivities of the nodes. The FEM global matrix and right hand side that would be built for such a mesh are given by:

$$\mathbf{U} = \begin{pmatrix} U_{11} & U_{12} & & & & U_{16} & \\ U_{21} & U_{22} & U_{23} & & & U_{26} & U_{27} \\ & U_{32} & U_{33} & U_{34} & U_{35} & & U_{37} \\ & & U_{43} & U_{44} & U_{45} & & \\ & & U_{53} & U_{54} & U_{55} & U_{56} & U_{57} \\ U_{61} & U_{62} & & & U_{65} & U_{66} & U_{67} \\ & U_{72} & U_{73} & & U_{75} & U_{76} & U_{77} \end{pmatrix}, \quad \mathbf{R} = \begin{pmatrix} R_1 \\ R_2 \\ R_3 \\ R_4 \\ R_5 \\ R_6 \\ R_7 \end{pmatrix}, \tag{6.1}$$

where U_{ij}, for $i, j = 1,..7$, are the entries of the global FEM matrices (these are the terms from the **A** matrix of Equation (2.36)) and R_i are the right hand side entries (these are the terms from **b** vector of Equation (2.36)). The technique applied in this thesis is to build the matrix as per Equation (6.1), however prior to solving the system interpolate the values of the hanging node, by setting the hanging node to equal the average of the solution value at the two adjacent nodes of the hanging edge. The other entries on the hanging node row of the matrix are distributed equally to the connected end node rows (of the hanging node edge). With this modification Equation 6.1 becomes:

$$\mathbf{U} = \begin{pmatrix} U_{11} & U_{12} & & & & U_{16} & \\ U_{21} & U_{22} & U_{23} & & & U_{26} & U_{27} \\ & U_{32}+\frac{U_{72}}{2} & U_{33}+\frac{U_{73}}{2} & U_{34} & U_{35}+\frac{U_{75}}{2} & \frac{U_{76}}{2} & U_{37}+\frac{U_{77}}{2} \\ & & U_{43} & U_{44} & U_{45} & & \\ & \frac{U_{72}}{2} & U_{53}+\frac{U_{73}}{2} & U_{54} & U_{55}+\frac{U_{75}}{2} & U_{56}+\frac{U_{76}}{2} & U_{57}+\frac{U_{77}}{2} \\ U_{61} & U_{62} & & & U_{65} & U_{66} & U_{67} \\ & 0 & -0.5 & & -0.5 & 0 & 1 \end{pmatrix}, \tag{6.2}$$

$$\mathbf{R} = \begin{pmatrix} R_1 \\ R_2 \\ R_3+\frac{R_7}{2} \\ R_4 \\ R_5+\frac{R_7}{2} \\ R_6 \\ 0 \end{pmatrix}. \tag{6.3}$$

The AMR process with FEM is complex, with many steps, and these are described in Process 6.3.

Process 6.3 Overview of refinement process with hanging nodes

1: Find elements that meet the criteria for refinement or de-refinement
2: **repeat**
3: Check the validity of the proposed mesh
4: Mark any extra elements for refinement or de-refinement
5: **until** No changes in this loop or maximum counter reached
6: Undertake the mesh refinement
7: Create required new nodes and elements
8: Set new node variables to the average of the two nodes they bisect
9: Remove elements marked for de-refinement and free their memory
10: Find all edges for the new mesh, recalculate areas and run RCM method
11: Determine hanging nodes by:
12: Looping through each edge
13: Find non-boundary edges that only appear in one element
14: Mark the central node on such an edge as a hanging node, N_h.
15: Store link to the the other two nodes on that edge within N_h.
16: Free all existing matrices and rhs vectors and create new memory allocations
17: Create a sparsity pattern and copy as needed
18: Build stiffness matrix, mass matrix and right hand side vector
19: Adjust the system and RHS as per the Theta method, to create a system of the form **Ax=b** as per Equation (2.36)
20: Process the hanging rows in the **A** matrix and vector **b** as per Equation (6.2) and Equation (6.3)
21: Note that if the appropriate columns do not exist in the sparse representation of **A**, they must be added to the sparsity pattern
22: The resulting **Ax=b** system can now be solved

6.4 Testing the locally adaptive system

The tests were undertaken using the electrical system as described in Chapter 2. This uses the TP06 ionic current model and is solved with the FEM. To test the locally adaptive system a 360 mm$\times$3.75 mm domain was used and a stimulus applied to a region near the left edge (where $X_1 < 1$) of the domain. The domain is sufficiently long to contain a full wave and the values of the transmembrane voltage at various nodes in the mesh were recorded at given times.

It should be noted that for the tests in this chapter the restitution slope of the electrical wave was set to 1.4. This was not chosen to demonstrate any electrophysiology property, but it does have an effect on the wave profiles seen in this chapter.

6.4.1 Errors from changing the refinement level

Ideally the voltage produced using an adapted mesh would be compared to a globally refined mesh with a very fine refinement and a convergence study undertaken. However comparing solutions from higher and lower refinement levels has the issue of the speed of the wave to contend with. The wave speed changes with the mesh refinement and a small change in wave speed changes the location of the wave front. This causes a large difference in the voltage at a given node. This can be see in Table 6.2. These results were produced on a 360 mm$\times$3.75 mm domain, with the time step halving with each refinement. The original coarse mesh had 130 nodes and 128 elements, and a single global refinement undertaken in all cases to produce the quadratic nodes for the mechanical system, giving a mesh with 387 nodes and 512 elements (this has an average element area of 0.165 mm^2 and an approximate edge length of 0.57 mm). Diffusion was set to be isotropic with a coefficient of $D = 0.154$. The area defined by $X_1 < 1$ mm was stimulated for $t < 1$ ms. The global refinement level then varied for a further 1 to 5 refinements. The solution for the voltage at the original 130 nodes was compared to the 5th refinement level, and the results are presented in Table 6.2. The RMS errors displayed in Table 6.2 are relatively large, considering that the voltage range for the system is from -86.2 mV to approximately 30 mV. Figure 6.6 demonstrates this graphically, with the wave position in each of the five examples being at a different position.

Furthermore, and as previously discussed in Section 2.6.1, the initial excitation of the left edge elements of the domain mean that there are slight variances in the starting point of the wave, due to the different triangle sizes and positions. These small changes in the starting conditions and the differences in wave speed mean RMS errors when comparing solutions of different refinement levels are large. However it is clear that as the mesh

Refinements	Nodes	Elements	Ave. Element Area	Time step	RMS Error
1	1285	2048	0.65918	0.32	55.3
2	4617	8192	0.16479	0.16	50.45
3	17425	32768	0.04120	0.08	44.45
4	67617	131072	0.01030	0.04	23.84
5	266305	524288	0.00257	0.02	-

Table 6.2: *RMS Error comparing the voltage at 130 common nodes, to the solution with 5 refinements. Tests undertaken on a 360 mm×3.75 mm domain, Diffusion was set to isotropic with a coefficient of D = 0.154 and* $X_1 < 1$ *mm was stimulated for* $t < 1$ *ms.*

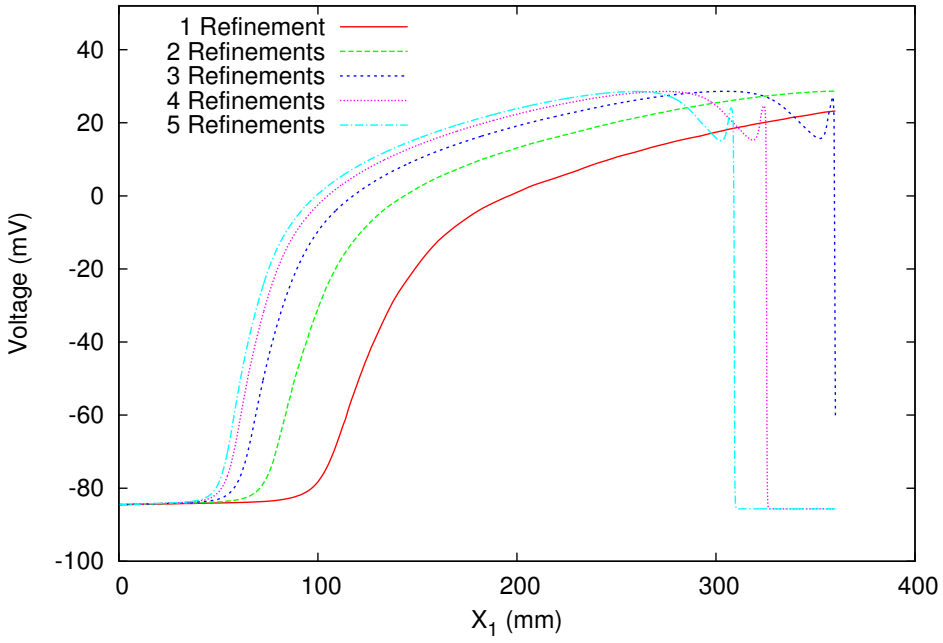


Figure 6.6: *Voltage at* X_2*=0, for 360 mm×3.75 mm domain with 5 different global refinement levels*

refinement increases the solution is converging and hence the RMS errors are seeming to drop linearly between refinements 3,4 and 5.

6.4.2 Validating the AMR software

To determine whether the AMR system could produce a close match to the globally refined system, tests were undertaken with the "buffer projection" (P_{dist}) of the local refinement set to be larger than the domain size. This means that as soon as local refinement is instigated, the entire mesh is "locally" adapted. This provides a simple means of producing a global refinement using the local adaptive mechanisms. This was tested on a 30 mm×30 mm domain, with 1 level of adaptivity and P_{dist} =50 mm, hence all the AMR mesh is refined from the outset. Tests were undertaken with a level 1 global refinement and compared to an AMR mesh with a maximum of 1 level of adaptivity. The voltages were compared for the common nodes in both meshes (i.e. the nodes before the local adaptivity) and the results were the same (to machine precision).

The AMR system has a feature to pre-adapt the stimulated area before the simulation starts. This feature ensures that where AMR is set to several levels, the area of initial excitation can be pre-adapted to the maximum level before the simulation starts. With this feature enabled the above tests were also undertaken with 2 and 3 levels of refinement. The AMR and globally refined solutions again produced results with no measurable differences (to machine precision).

6.4.3 Testing the wave front

Further tests were undertaken using AMR, with the monitor described in Process 6.1, to determine a projection parameter, P_{dist}, that could produce a good comparison to the globally refined system. The globally refined solution was found for refinements of 2 and 3 levels and the solution at the common nodes (i.e. coarse mesh) was compared to the local adapted solution with varying P_{dist}. The RMS error was calculated between the solution on the AMR mesh and on the globally refined mesh. The maximum adaptivity level, A_{max}, of the AMR mesh was set to the number of refinements in the globally refined mesh. The results of these tests can be seen in Table 6.3 and illustrate that if the P_{dist} used by the AMR system is large enough then the AMR solution is identical (to machine precision) to the globally refined solution. The average element area of the two meshes in the Table 6.3 were 0.0475 mm^2 and 0.0119 mm^2, with approximate edge lengths of 0.31 mm and 0.15 mm respectively. It should be noted that these measurements were

A_{max}	Average Refined Element Area	P_{dist} = 10 mm, RMS Error	P_{dist} = 5 mm, RMS Error	P_{dist} = 2.5 mm, RMS Error	P_{dist} = 1.125 mm, RMS Error	Global RMS Error
2	0.0475	0.00×10^{-16}	0.00×10^{-16}	7.02×10^{-6}	2.79×10^{-4}	50.45
3	0.0119	0.00×10^{-16}	0.00×10^{-16}	0.00×10^{-16}	4.56×10^{-5}	44.45

Table 6.3: *RMS error for voltage compared to globally refined mesh of same maximum level with a varying adaptivity projection parameter, P_{dist}. Tests undertaken on a 30 mm × 30 mm mesh, diffusion set to isotropic with $D = 0.154$, time-step $dt = 0.078$ ms and measurements taken at $t = 5.52$ ms. The global RMS error is in comparision to a level 5 global system as per Table 6.2.*

taken early in the simulation, when t=5.52 ms, before the back of the wave could de-refine.

6.4.4 Testing whole wave

The tests in Section 6.4.3 demonstrate that a locally adaptive mesh can produce results that are the same (to machine precision) as a globally refined mesh at the front of the wave. The next set of tests investigate the accuracy of the solution that can be produced when considering the whole wave.

To investigate the differences between the locally adapted mesh and a globally refined mesh the tests were carried out on a long thin domain. This domain was 360 mm×3.75 mm and this ensured that an entire wave profile could fit within the domain. The AMR maximum level was set to 4 and the time step parameter was dt=0.08 ms. The initial mesh had 130 nodes and 128 elements and a single initial global refinement was undertaken in all cases to produce the quadratic nodes for the mechanical system.

For the first set of tests the simple monitor function were used, as described in Process 6.1. In these tests the voltage difference threshold, V_{tol0}, was set to 5 mV and the projection buffer, P_{dist}, was set to 5 mm. The de-refinement strategy is as described in Process 6.1, and the parameter for this time to de-refine (T_{tolD}) was varied throughout the tests. The higher the value the longer an element stays refined, and so this can also be set to an artificially high value to ensure an element never de-refines. The results in Table 6.4 show the RMS error calculated by comparing the AMR solution at the original 130 nodes common to all of the meshes with a 4 level globally refined solution.

Table 6.4 shows that if the monitor function is set to never de-refine an element then a solution which is exact to 6 decimal places can be produced. It also shows that by setting

Time (ms)	$T_{tolD} = 4$ ms RMS Error	$T_{tolD} = 8$ ms RMS Error	$T_{tolD} = 32$ ms RMS Error	No de-refinement RMS Error	Global RMS Error
384	6.460×10^{-2}	5.092×10^{-2}	4.477×10^{-2}	4.472×10^{-6}	23.84

Table 6.4: *RMS error for voltage compared to globally refined mesh of same maximum level, with varying T_{tolD} parameter. The domain was 360 mm$\times$3.75 mm, $A_{max} = 4$, $V_{tol0} = 5$ mV and $P_{dist} = 5$ mm, time-step dt=0.08 ms and diffusion set to isotropipc with $D = 0.154$. Global RMS error shows RMS error between level 4 and level 5 globally refined meshes.*

de-refine times of 4 ms, 8 ms and 32 ms a very good approximation to the globally refined solution can be produced.

Figures 6.7 and 6.8 illustrate the area of the domain which was refined during the tests for 8 ms and 32 ms. These figures have a line where each node is located and hence the density of these lines increases where the mesh is refined. The mesh using a 32 ms time de-refinement parameter has much more of the domain refined, including more of the back of the domain. In these two tests the number of nodes used by the meshes were 4902 for the 8 ms de-refinement case and 10466 for the 32 ms de-refinement case, and so the cost of this extra accuracy is the need to use twice as many nodes. This result indicates that the front of the wave is the primary cause of potential errors and the back of the wave contributes less to the overall error. Therefore a more sophisticated monitor function, that would be able to treat the front and back of the waves differently, may improve efficiency.

6.4.5 Testing with an improved monitor function

The tests in Section 6.4.4 demonstrate that a locally adaptive mesh can produce results that are comparable to a fully refined mesh. However using the same refinement strategy for the front and back of the wave may be inefficient. Also the use of parameters for time de-refinement is not ideal, as this depends on the model being used, and the refinement level. Therefore a more sophisticated monitor function was developed as described in Process 6.2, and tests using this are presented below.

As previously, the differences between the locally adapted and globally refined solution for the full wave tests were considered and the tests carried out on a long thin domain. This domain was 360 mm$\times$3.75 mm to ensure an entire wave profile could fit within the domain and the whole profile of the wave compared. The projection buffer, P_{dist}, was set to 5 mm and the AMR maximum level, A_{max}, set to 4. The time step parameter was

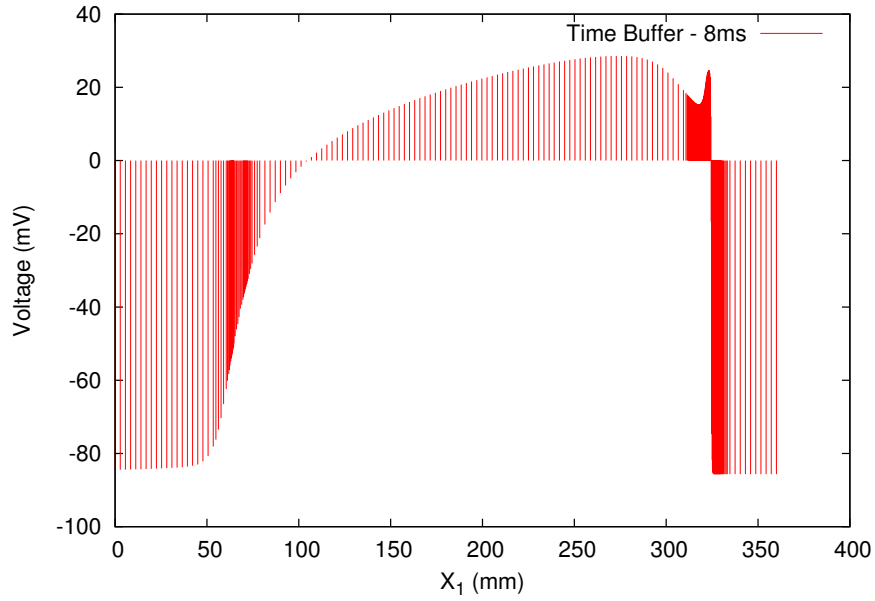


Figure 6.7: *Voltage at* $X_2 = 0$, *with* $T_{tolD} = 8$ *ms at t=384 ms. Adapted areas shown by denser shading, using monitor function defined in Process 6.1. The domain was 360 mm*$\times$*3.75 mm,* $A_{max} = 4$, $V_{tol0} = 5$ *mV and* $P_{dist} = 5$ *mm, time-step dt=0.08 ms and diffusion set to isotropic with* $D = 0.154$.

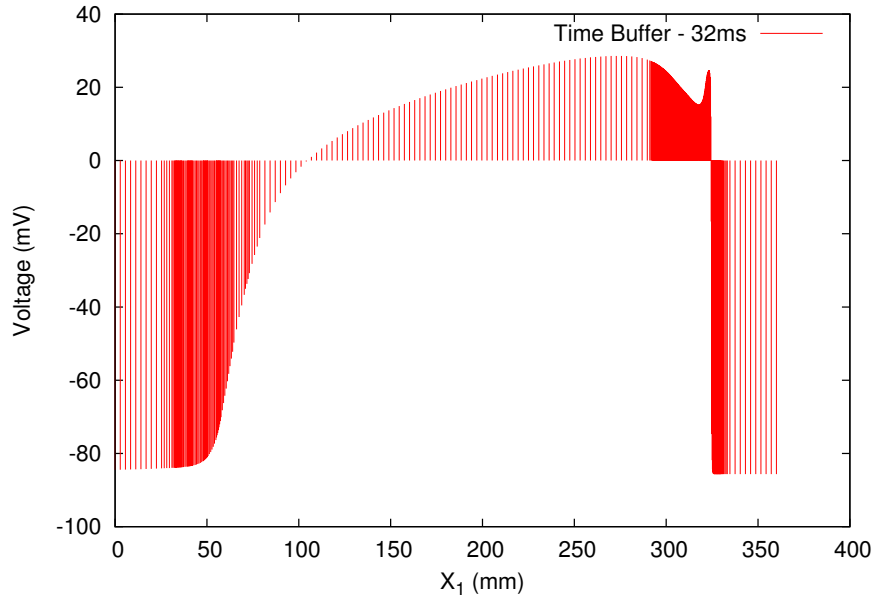


Figure 6.8: *Voltage at* $X_2 = 0$, *with* $T_{tolD} = 32$ *ms at t=384 ms. Adapted areas shown by denser shading, using monitor function defined in Process 6.1. The domain was 360 mm*$\times$*3.75 mm,* $A_{max} = 4$, $V_{tol0} = 5$ *mV and* $P_{dist} = 5$ *mm, time-step dt=0.08 ms and diffusion set to isotropic with* $D = 0.154$.

Test No	V_{tolR1}	V_{tolR2}	RMS Error	Global RMS Error
TR1	2 mV	2 mV	0.0246	23.84
TR2	2 mV	4 mV	0.0374	23.84
TR3	2 mV	1000 mV (No refinement when voltage falling)	0.0483	23.84

Table 6.5: *RMS error for voltage at coincident points at t=384 ms, with various AMR parameters. The domain was 360 mm×3.75 mm , $V_{tolD} = 1$ mV, $P_{dist} = 5$ mm, $A_{max} = 4$, the time-step was dt=0.08 ms and diffusion set to isotropic with $D = 0.154$. Global RMS error is RMS error between the level 4 and level 5 globally refined solutions.*

dt=0.08 ms. The initial mesh had 130 nodes and 128 elements and a global refinement was undertaken in all cases to produce the quadratic nodes for the mechanical system.

Table 6.5 shows the results from these tests. These show a very good level of RMS error using voltage thresholds of $V_{tolR1} = 2$ mV and $V_{tolR2} = 2$ mV for the refining when the voltage is rising and falling. Also if V_{tolR2} is set to an unachievable value (e.g. $V_{tolR2}2 = 1000$ mV) then the mesh will not refine when the voltage is falling, this means that there is no refinement at the back of the wave. This technique still provides an RMS error beneath 0.05, which compares very well to the inter-grid error of the order of 20. Figure 6.9 shows that only the area around the wave front has been adapted.

6.4.6 Adaptivity level for excited region

Further tests were then undertaken to investigate if the RMS error could be reduced further by keeping the adaptivity level above zero where the wave is excited. That is, set a parameter, A_{minE}, which ensures that if the voltage for an element is higher than the resting state (V_{tolE}), then the element cannot de-refine beneath level A_{minE}. Tests were undertaken with $A_{max} = 4$, and A_{minE} set to 0, 1, 2 or 3. The results from these tests can be seen in Table 6.6. These were undertaken with a projection buffer, P_{dist}, of 5 mm and with the refinement thresholds (V_{tolR1},V_{tolR1}) at 2 mV and de-refinement threshold V_{tolD} at 1 mV.

Table 6.6 demonstrates that by keeping the mesh adapted where the wave is excited the RMS error can be further reduced. If the mesh is allowed to fully de-refine then the RMS error is 0.0246 at t=384 ms, however by setting the excited area of the mesh to stay refined to 3 levels reduces this error to 0.0007. This RMS error is relative to the globally refined solution at four levels.

The above tests demonstrate that a solution with an RMS error of 0.0007, compared

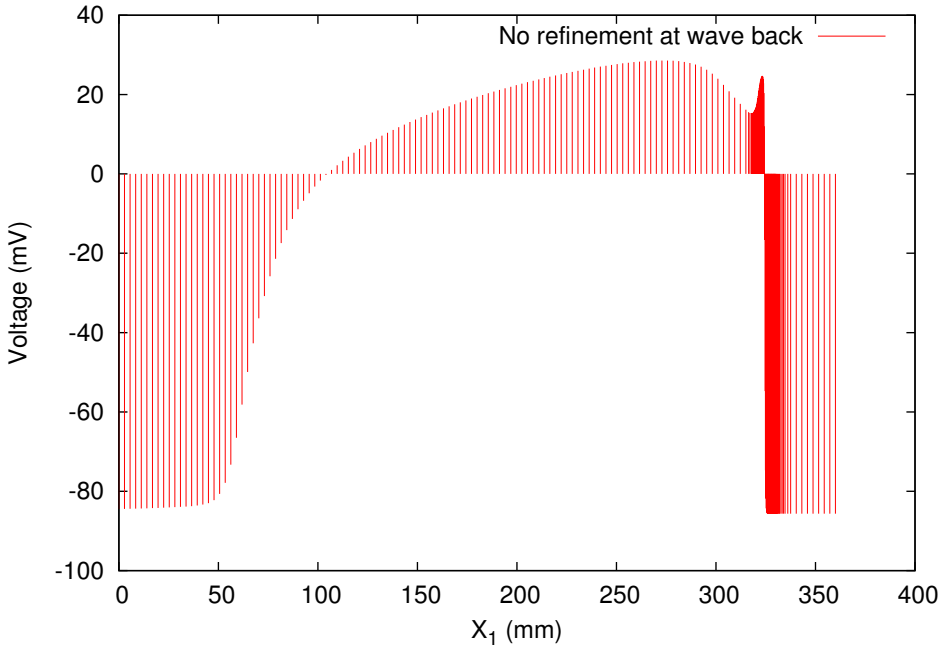


Figure 6.9: *Second monitor function (Process 6.2), with* $V_{tolR2} = 1000\,mV$ *(no refinement when voltage falling). Plot is Voltage at* $X_2 = 0$ *and t=384 ms. Adapted areas shown with dark shading. The domain was 360 mm*$\times$*3.75 mm ,* $P_{dist} = 5$ *mm,* $A_{max} = 4$*, the time-step was dt=0.08 ms and diffusion set to isotropic with* $D = 0.154$*.*

Test	A_{minE}	Nodes	RMS Error	Global RMS Error
TL0	0	3685	0.0246	23.84
TL1	1	5468	0.0217	23.84
TL2	2	7265	0.0093	23.84
TL3	3	16532	0.0007	23.84

Table 6.6: *RMS error for voltage at t=384 ms, with elements set to stay refined to varying levels when still excited. The domain was 360 mm*$\times$*3.75 mm ,* $P_{dist} = 5$ *mm,* $A_{max} = 4$*,* $V_{tolR1} = V_{tolR2} = 2\,mV$*,* $V_{tolD} = 1\,mV$*,* $V_{tolE} = -84\,mV$*, the time-step was dt=0.08 ms and diffusion set to isotropic with* $D = 0.154$*. Global RMS error is RMS error between level 4 and level 5 globally refined solutions.*

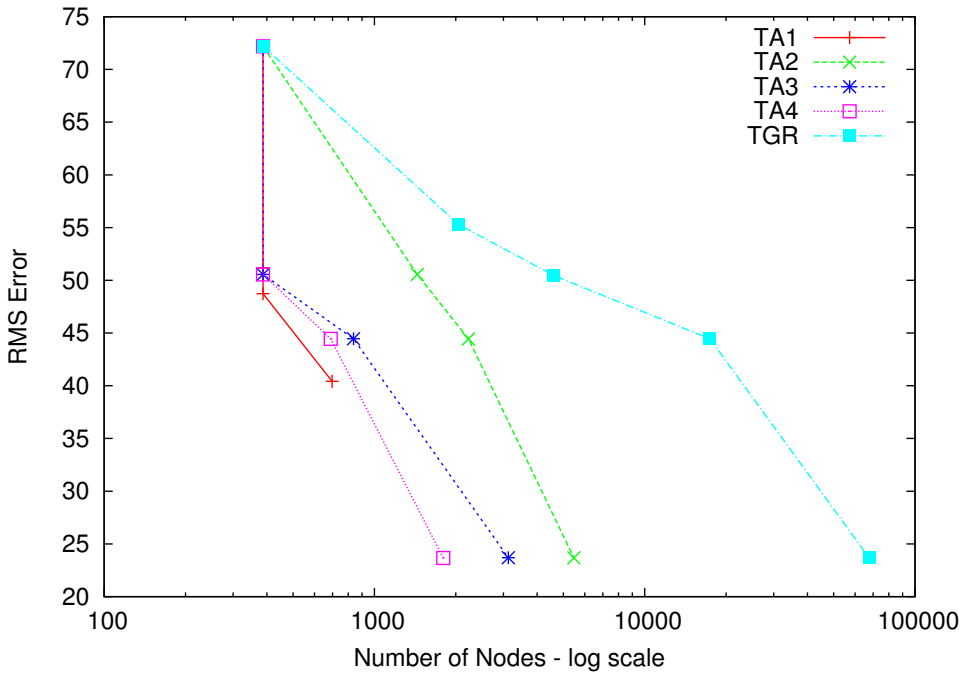


Figure 6.10: *Tests showing the RMS error produced by varying monitor function parameters. The RMS error compared to the results of the globally refined 5 level test. The domain was 360 mm×3.75 mm and t=384 ms, the time-step was dt=0.08 ms and diffusion set to isotropic with* $D = 0.154$.

to the fully refined mesh with 4 levels of refinement, can be produced with an AMR strategy. Tables 6.5 and 6.6 show the quantitative differences between an AMR solution and a globally refined solution (both with level 4 refinement).

6.4.7 Parameter permutations

The AMR software has parameters for refinement and de-refinement that are covered in Table 6.1. Table 6.7 shows the RMS errors, compared to a globally refined 5 level case, for various permutations of these parameters. The RMS errors can be also seen in Figure 6.10. This table groups tests together into similar sets of tests at different levels of A_{max}. These are labelled as TA1, TA2, TA3, TA4 for the AMR tests and TGR for the global refinement results. This table illustrates that the most important two factors for accuracy are the AMR refinement level (A_{max}) and then the projection buffer (P_{dist}). For the tests where A_{max}=4, if the projection buffer is large enough (2 mm or greater), the error of the AMR solutions, compared to the globally refined solution at the same level, has a maximum of 0.01.

Nodes	RMS	A_{max}	A_{minE}	V_{tolR1}	V_{tolR2}	V_{tolD}	P_{dist}
266305	-	5	NA	NA	NA	NA	NA
TGR							
387	72.19	0	N/A	N/A	N/A	N/A	N/A
2048	55.30	1	N/A	N/A	N/A	N/A	N/A
4617	50.45	2	N/A	N/A	N/A	N/A	N/A
17425	44.45	3	N/A	N/A	N/A	N/A	N/A
67617	23.68	4	N/A	N/A	N/A	N/A	N/A
TA1							
387	51.18	2	0	2	Disabled	1	0.1
387	48.74	3	0	2	Disabled	1	0.1
697	40.42	4	0	2	Disabled	1	0.1
TA2							
1440	50.56	2	1	2	2	1	5
2229	44.44	3	1	2	2	1	5
5468	23.68	4	1	2	2	1	5
TA3							
387	50.56	2	0	2	Disabled	1	5
837	44.45	3	0	2	Disabled	1	5
3128	23.69	4	0	2	Disabled	1	5
TA4							
387	50.56	2	0	2	Disabled	1	2
689	44.44	3	0	2	Disabled	1	2
1798	23.67	4	0	2	Disabled	1	2
TA5							
5211	0.03	5	0	2	Disabled	1	2

Table 6.7: *Tests showing the RMS error produced by varying monitor function parameters. The RMS error as compared to the results of the globally refined 5 level test. The domain was 360 mm$\times$3.75 mm and t=384 ms, the time-step was dt=0.02 ms where* $A_{max} = 5$ *and dt=0.08 ms in all other cases. The diffusion was set to isotropic with* $D = 0.154$.

6.5 Benefits of AMR

6.5.1 Efficiency benefits

Sections 6.4.3 - 6.4.6 illustrate that the AMR solution can provide accurate approximations to the electrical system. The aim of an AMR solution is to produce results of a specified accuracy, however using less computing resources. This section considers the performance improvements that can be achieved when using AMR and also the reduction in memory usage.

In the simulations in this thesis, the main data structures used store the nodes, elements and edges of the computational mesh. By reducing the number of elements and nodes there is a direct memory saving, enabling simulations with a finer mesh resolution to be undertaken on the same equipment. During the production of the results above the globally refined system would frequently run out of memory (using a workstation as per Section B.2), whereas the AMR solution at the same level of refinement would complete successfully. Another benefit with smaller numbers of nodes and elements is the disk storage needed for results and check-point files. These can be quite significant, with each time-step of the heart failure simulation (see Chapter 7) producing output files in the region of 60 megabytes. Files of this size take a long time to write to disk and then occupy a costly amount of space. The full cost of storage also needs to include the backup and management of files, and so large amounts of files incur ongoing management costs. To simplify this, this section will consider the number of nodes used in each AMR simulation and compare this with a globally refined solution. The number of nodes used is directly proportional to the memory needed to run the software.

Secondly this section will consider the time to undertake the simulations. As discussed in Section 5, the time to undertake cardiac simulations can run into months of elapsed time. Improving the speed of the production of these results has a number of obvious benefits, especially where researchers are running many simulations and then using the output from them to further refine parameters and amend test plans.

Therefore the objective is to determine whether the AMR solution can both improve the speed and lower the resources used to undertake the simulations, whilst maintaining an acceptable level of accuracy.

6.5.2 Improvements provided by AMR

Table 6.8 shows the average and peak number of nodes used over the tests previously described in Table 6.6. The table shows the average and peak number of nodes used in the

Test	Average number of nodes	Peak number of nodes	RMS Error
Global 4	67617	67617	-
TL3	10452	16918	0.0007
TL2	4642	8340	0.0093
TL1	3214	6051	0.0217
TL0	2891	6133	0.0246

Table 6.8: *Average and peak number of nodes over AMR simulations. The domain was* $360\,mm \times 3.75\,mm$, $P_{dist} = 5\,mm$, $A_{max} = 4$, $V_{tolR1} = V_{tolR2} = 2\,mV$, $V_{tolD} = 1\,mV$, $V_{tolE} = -84\,mV$, *the time-step was* dt*=0.08 ms and diffusion set to isotropic with* $D = 0.154$. *RMS error as compared to globally refined level 4 solution.*

Test	V_{tolR2}	Average number of nodes	Peak number of nodes	RMS Error
Global R4	N/A	67617	67617	-
AMR4-MR0B2	2 mV	2891	6133	0.0246
AMR4-MR0B4	4 mV	2453	3573	0.0374
AMR4-MR0NB	No refine	2352	2692	0.0483

Table 6.9: *Average and peak number of nodes over simulation with varying* V_{tolR2}. *The domain was* $360\,mm \times 3.75\,mm$, $P_{dist} = 5\,mm$, $A_{max} = 4$, $V_{tolR1} = 2\,mV$, $V_{tolD} = 1\,mV$, $V_{tolE} = -84\,mV$, *the time-step was* dt*=0.08 ms and diffusion set to isotropic with* $D = 0.154$. *The RMS error is compared to the globally refined level 4 solution.*

simulations, which were run until t=500ms. The long-thin domain was used and in this time the wave moves from the left edge and passes the right edge boundary. The globally refined level 4 system uses 67617 nodes.

The test 'TL0' demonstrates that the AMR solution uses less than 5% of the nodes of the globally refined solution, however it has been previously shown that this method of refinement produces an RMS error of only 0.0246 across the domain.

By adjusting the monitor function parameters further efficiencies can be made. In Table 6.9 the threshold for refining elements when the voltage is falling is adjusted. Test 'AMR4-MR0NB' is the most aggressive of these, with no elements refined when the voltage is falling, and this has an RMS error of 0.0483, however only uses at its peak 4% of the nodes. These tests illustrate that far fewer nodes are used with an AMR simulation.

Table 6.10 displays the CPU time used in tests where the refinement level under the wave was kept refined at 1, 2, or 3 levels. These are compared to the CPU time to undertake a simulation with a globally refined mesh. The simulations were run until t=500 ms, on a 360 mm$\times$3.75 mm domain, with 130 nodes initially (with one level of refinement un-

Test	A_{minE}	CPU Time (s)	Improvement Ratio	RMS Error
Global R4	4	2969.26	-	-
AMR4-MR3	3	598.96	4.96	0.0007
AMR4-MR2	2	251.39	11.81	0.0093
AMR4-MR1	1	166.26	17.86	0.0217
AMR4-MR0	0	147.47	20.13	0.0246

Table 6.10: *CPU time to complete 500 ms of a plane wave simulation with varying A_{minE}. The domain was 360 mm×3.75 mm , $P_{dist} = 5$ mm, $A_{max} = 4$, $V_{tolR1} = 2$ mV, $V_{tolR2} = 2$ mV, $V_{tolD} = 1$ mV, $V_{tolE} = -84$ mV, the time step was dt=0.08 ms and diffusion set to isotropic with $D = 0.154$. The RMS error is compared to the globally refined level 4 solution.*

dertaken to determine the quadratic nodes for the mechanical mesh). These results show that the reduction in node numbers translate to an improvement in performance of up to 20 times. The ratio of improvement in performance is not as high as the reduction in node numbers, however, as the AMR system has the overhead of building the new mesh, rebuilding the global matrices for the FEM method and processing the hanging nodes.

A further test was undertaken, with the threshold for refinement when the voltage was decreasing set to not refine (i.e. $V_{tolR2} = 9999$ mV), with $V_{tolR1} = 8$ mV, and using a buffer projection region (P_{dist}) of 4 mm. The CPU time for this run was 97.22 s, with peak nodes of 2692 and has an RMS error of 0.0354. Thus only using 4% of the CPU time, and giving a speed up of over 30 times.

The tests undertaken so far in this section were all with a plane wave travelling from the left edge of the domain to the right edge. To investigate the AMR process when a spiral wave forms, a test was undertaken with a spiralling electrical wave. This was undertaken on a 120 mm×120 mm domain, with 183 nodes and 330 elements on the initial mesh. This was refined once to produce the quadratic nodes for the mechanical system and then a further 4 levels of global refinement were undertaken and this results in a mesh with 169505 nodes and 337920 elements. This was compared to a four level AMR mesh and the results of this can be seen in Figure 6.11. On the AMR mesh the coarsest electrical elements have an approximate edge length of 4.67 mm and the finest elements have an approximate edge length of 0.29 mm. The number of nodes increases as the spiral builds, but then stabilises, varying between 50000 and 60000 nodes. This simulation is of a spiral wave and the figure shows the periodic nature of the refinement as the wave spirals around the mesh. Figures 6.14 and 6.15 show the distribution of the elements in a spiral wave and the increased density of refined elements (compared to a line wave).

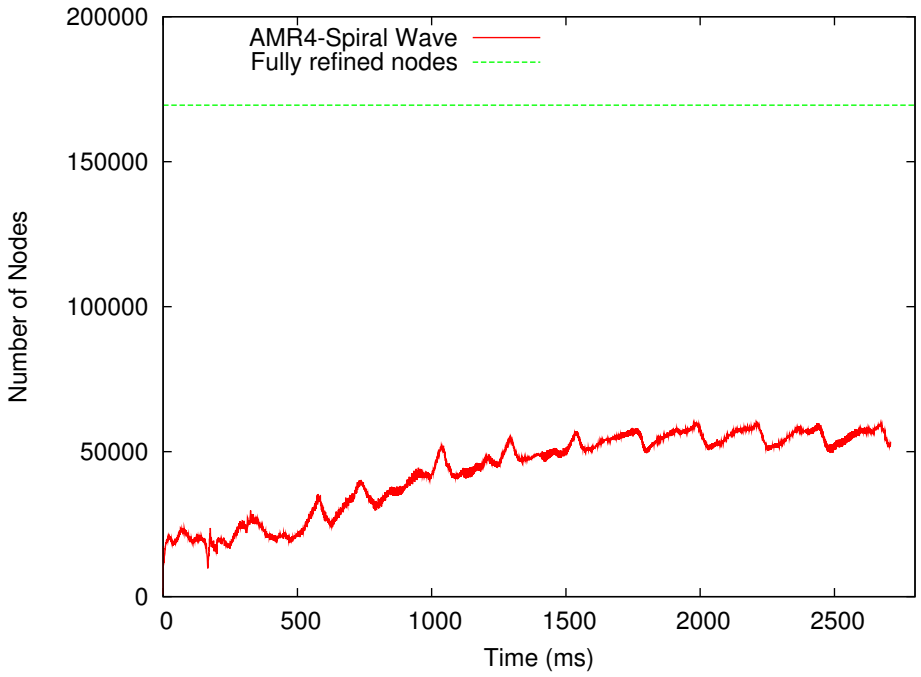


Figure 6.11: *Number of nodes used during a spiral wave spiral wave simulation on a 120 mm×120 mm domain, with 183 nodes initially. Comparing 4 levels of global refinement with 4 AMR levels.*

In the line wave examples previously discussed the excited region is a band across a narrow domain, with the length of the excited wave front being the X_2 dimension (typically 3.75 mm in the line wave tests). If the projection buffer was 5 mm in all directions and the wave back not refined that would mean that approximately 3% of the domain is refined to contain the wave. For a spiral wave on a square domain the length of the excited wave front is relatively longer. In Figure 6.13 the domain is 120 mm×120 mm and the wave front is approximately 265 mm in length. Again, if we assume a 5 mm buffer in all directions this would give an excited area within the domain of 18% of the domain. This needs to be considered when comparing the likely effectiveness of an AMR solution, as the more of the domain is excited the more elements will be refined (and hence there will be more nodes used).

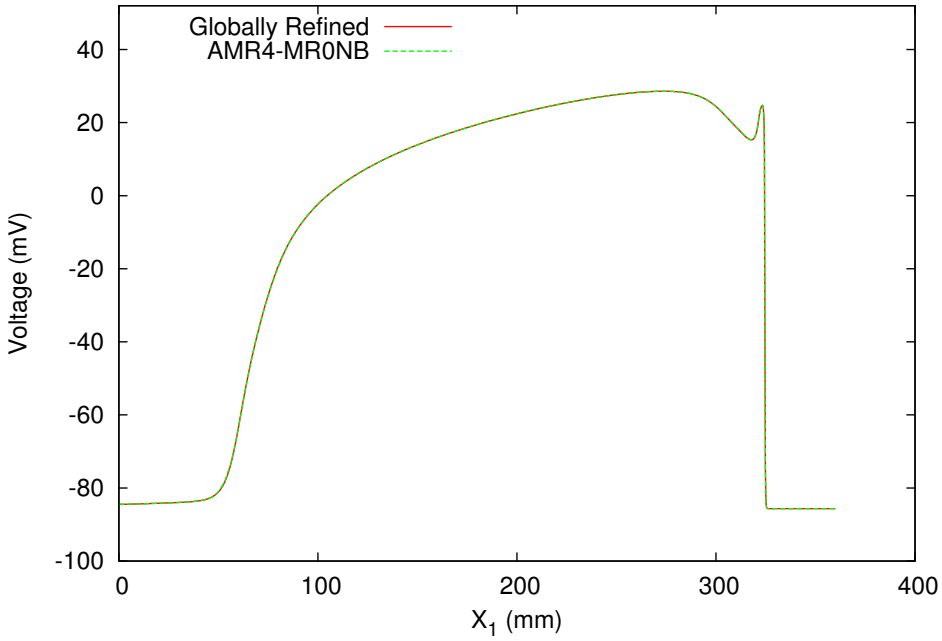
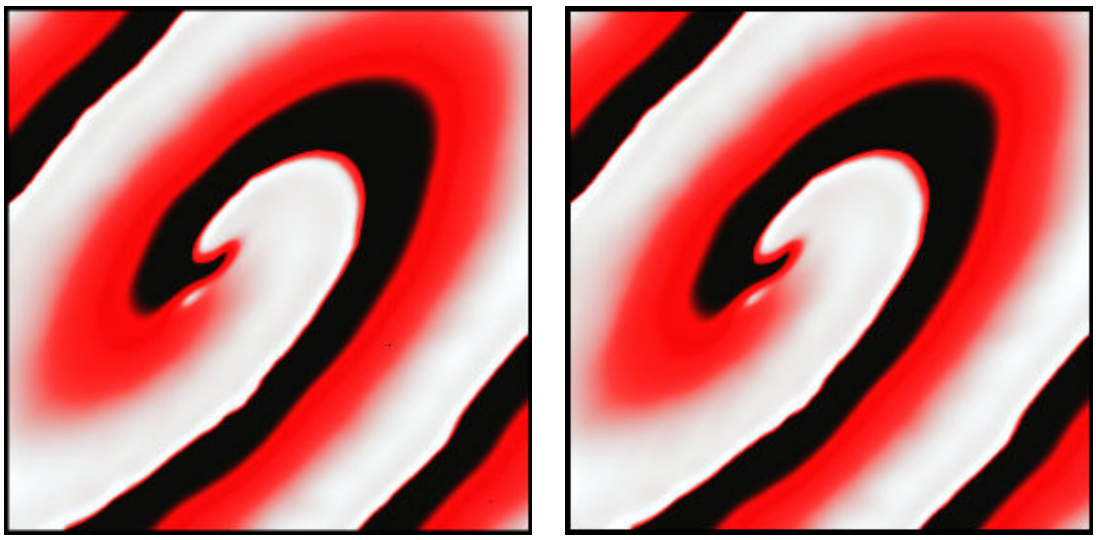


Figure 6.12: *Qualitative comparison of globally refined and AMR solutions. Voltage of solutions at* $X_2 = 0$*, with domain of 360 mm*$\times$*3.75 mm. Globally refined solution had 4 levels of refinement and AMR solution had the following parameters:* $A_{minE} = 0$*,* $P_{dist} = 5$ *mm,* $A_{max} = 4$*,* $V_{tolR1} = 2$ *mV,* $V_{tolD} = 1$ *mV,* $V_{tolE} = -84$ *mV, the time-step was dt=0.08 ms and diffusion set to isotropic with* $D = 0.154$*.*

6.5.3 Wave profile for a line wave

The results so far in this chapter have shown that the AMR system can produce results which compare closely to the fully refined solution at the same refinement level. The range of values for the voltage is typically between -86.2 mV and 30 mV and RMS errors when using the AMR solution were typically lower than 0.05.

Figure 6.12 illustrates how the waves are modelled across the domain, with the results from the test 'AMR4-MR0NB' compared to the globally refined solution. This test is defined in Table 6.9, where the mesh is not refined when voltage is falling. This illustrates the similarity in cardiac electrical wave that can be reproduced with significantly fewer nodes. The plots on this graph are almost exactly on top of each other and it is difficult to see any qualitative difference, however the peak number of nodes in the AMR solution was approximately 4% of the globally refined solution.



(a) AMR (b) Globally Refined

Figure 6.13: *Spiral wave at t=2240 ms using an AMR mesh and a globally refined mesh, with 3 levels of refinement. This was on a 120 mm×120 mm domain with 1295 initial nodes, with dt=0.08 ms. The setting used for globally refined test are from row 1 of Table 7.1.*

6.5.4 Testing with a spiralling wave

Figure 6.13 compares the results of running the AMR and globally refined solutions on a 120 mm×120 mm domain (with 1295 nodes initially) and a spiral wave formed by plane wave splitting. The results look almost identical.

Figure 6.14 shows the adapted mesh when a spiral wave has formed. This is with 3 levels of AMR refinement on a 120 mm×120 mm domain with 1295 initial nodes, after 1920 ms with dt=0.08 ms. Figure 6.15 shows the adapted mesh when a spiral wave has formed and the electrical system is coupled to the mechanical system to make a deforming domain. This is also on a 120 mm×120 mm domain with 1295 initial nodes, after 1920 ms with dt=0.08 ms. In these tests the number of nodes used grows over time (as per Figure 6.11) and then oscillates between 80000 and 90000 nodes as the wave spirals within the domain. A globally refined mesh for this domain has 318065 nodes and so the adaptive system uses up to (approximately) four times fewer nodes.

Figures 6.14 and 6.15 illustrate that even with a spiralling wave, in which a much greater proportion of the domain is excited, an AMR strategy utilises far fewer nodes than a globally refined solution. Figure 6.15 was produced with the mechanical coupling enabled and demonstrates the capability of the system to produce results for an adaptive mesh that is deforming due to the forces exerted by the mechanical system.

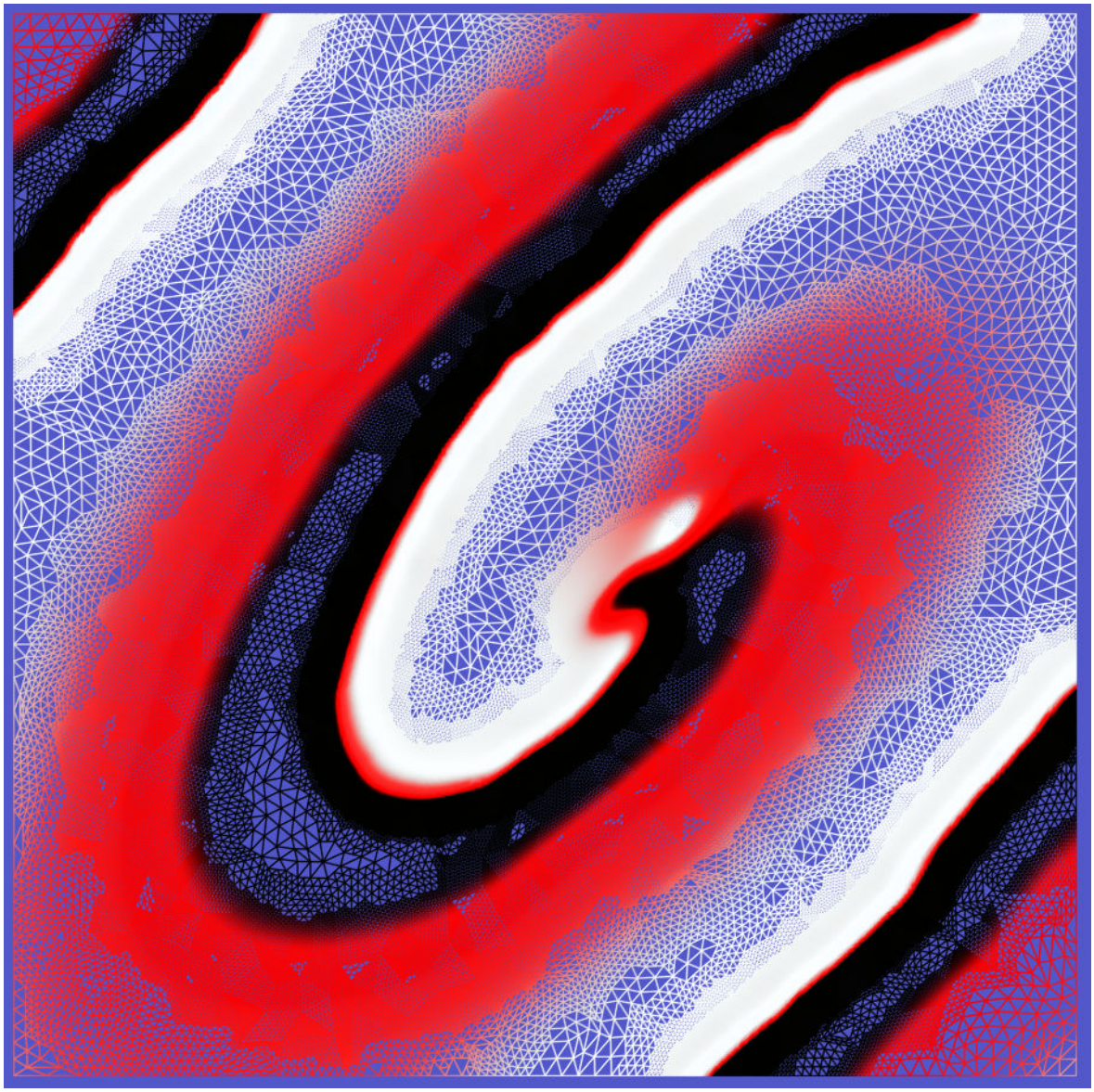


Figure 6.14: *Spiral wave at t=1920 ms using a AMR mesh with 3 levels of refinement. This was on a 120 mm×120 mm domain with 1295 initial nodes and with dt=0.08 ms.*

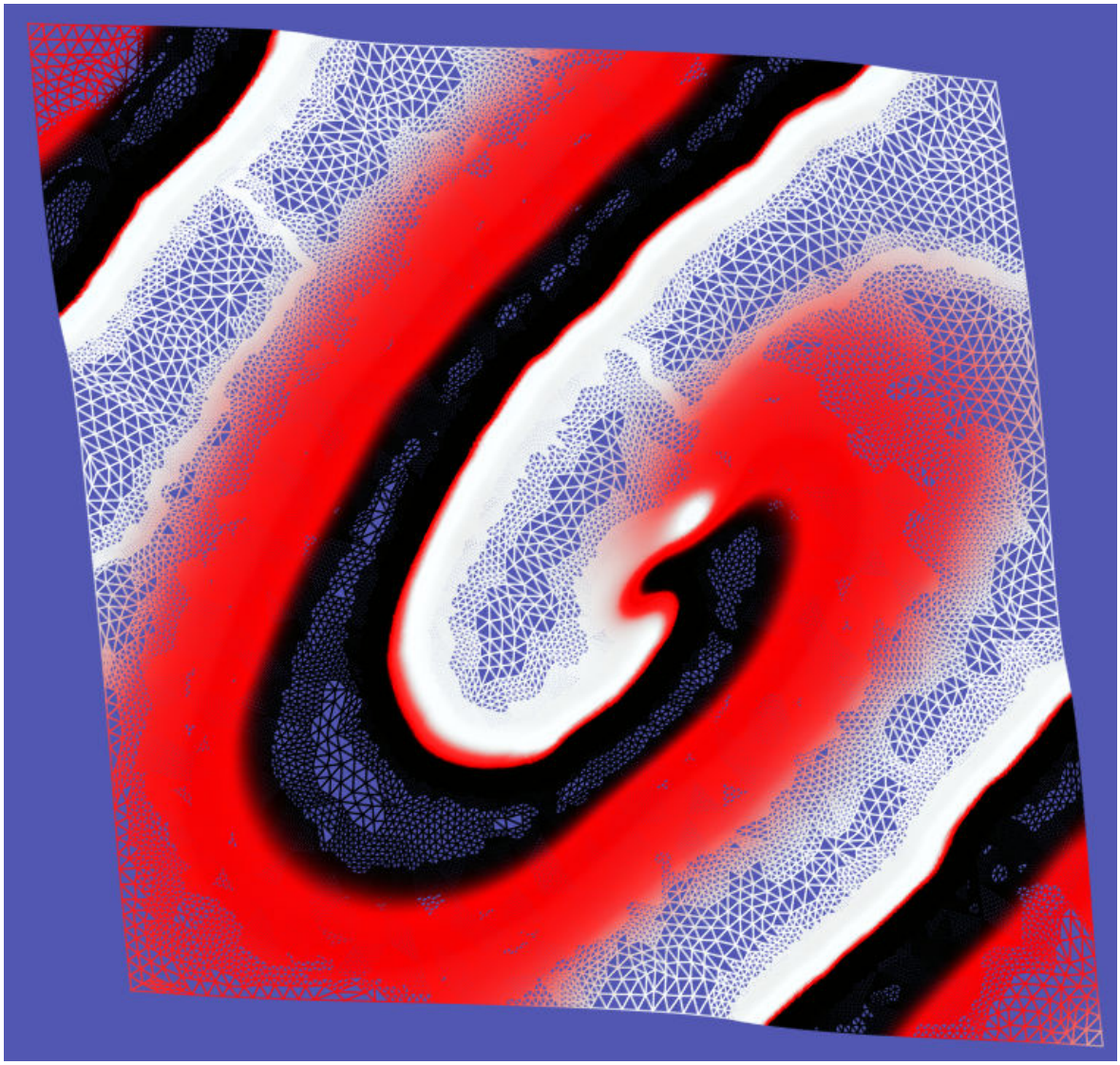


Figure 6.15: *Spiral wave on a deforming domain, at t=1920 ms using a AMR mesh with 3 levels of refinement. This was on a 120 mm×120 mm domain with 1295 initial nodes and with dt=0.08 ms.*

6.6 Conclusions

The objective of this chapter was to investigate whether applying a local adaptive mesh refinement strategy to the coupled solver could improve the efficiency of the system and produce results with an acceptable level of accuracy.

This chapter has shown that the implemented AMR technique can produce results that are quantitatively very close to a fully refined solution and compare qualitatively very well. The RMS errors produced in the tests were typically under 0.05 (when compared to the global solution at the same level of refinement). The fact that the transmembrane voltage ranges from -86.2 mV to +30mV is also important as this error is very small compared to the solution range.

Two monitor algorithms were developed and implemented and both of these are driven by changes in the voltage variables through the domain. This enabled the control of the refinement at the front and back of the wave and also underneath the wave. By amending the monitor function parameters a trade off can be found between a very high level of accuracy and the number of nodes used. Test TL3 in Table 6.6 produces an RMS error of 0.0007, and only uses 16532 nodes compared to 67617 nodes for the globally refined solution, and test 'AMR4-MR0NB' in Table 6.9 only uses a peak of 2692 nodes, however has an RMS error of 0.0483.

The AMR solution was then tested with a spiralling electrical wave on both static and deforming domains. This produced good results also, with the number of nodes used in spiral wave simulations increasing over time as the spiral wave builds, however then settling down into a periodic oscillation between 25% to 33% of the nodes required in the globally refined system.

The AMR solution can also provide significant performance improvements, with line wave simulations performing up to 30 times faster when AMR was used. However with the current implementation and monitor function, the spiral wave has too great a proportion of the domain for the adaptive meshes to show performance benefits.

The direct benefits from AMR were noticed when undertaking this thesis, for example solutions could be calculated to a higher level of mesh refinement using the same computing equipment. The computing equipment specified in Appendix B.2, was able to undertake simulations on a $120\,\text{mm} \times 120\,\text{mm}$ domain starting with a coarse mesh of 1295 nodes and then having 5 levels of refinement (plus one refinement for the quadratic mechanical nodes), when AMR was enabled. Without AMR, only 4 levels were achievable. This had ancillary benefits in that the output files were smaller and it was less time consuming producing visualisation images.

Finally, in other published work (for example [114]), both spatial and time adaptivity are applied, with an operator splitting technique used to split the diffusion and reaction terms. It should also be noted in this thesis no temporal adaptivity is undertaken, however an operator splitting method is used for the time terms, and this would facilitate the development of temporal adaptivity in the future.

Chapter 7

Modelling heart failure

7.1 Introduction

The previous chapters of this thesis describe the development of an efficient, flexible tool to simulate deforming cardiac tissue. This utilises the TP06 model of cardiac electrophysiology [110], which is coupled to an incompressible tissue model, enabling the simulation of cardiac activity on a deforming domain.

The objective of this chapter is to use the developed model to simulate the conditions present in tissue with end-stage cardiac disease. The flexibility of the software developed allows various properties to be altered to model specific changes in pathology found in end-stage heart failure. The chapter will present the techniques used to model cardiac disease, introduce an alternative method for calculating the tension in the mechanical system and then present the results from the simulations undertaken.

7.2 Simulating heart failure

7.2.1 Electrophysiological mechanisms

The objective of this section is introduce amendments to the TP06 model of electrophysiology to reproduce changes seen in cardiac tissue in end-stage heart failure conditions. How underlying cardiac disease contributes to electrophysiological arrhythmia mechanisms is a well studied area [113], and the common findings from this include altered

calcium handling [10] and prolongation of the action potential duration [37].

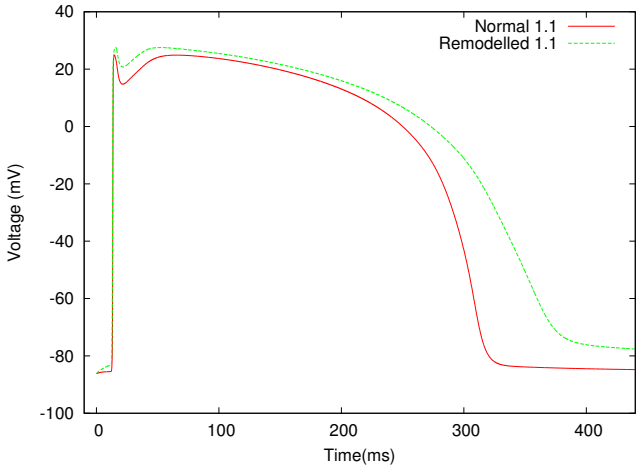
To simulate this, components of the ionic currents I_{ion} (described by [110]) in Equation (2.1) were amended. In Priebe and Beuckelmann [87] a study of the electric properties of cells in heart failure is undertaken, however in this thesis, more recent data obtained from human cardiac cells, rather than from animal cells, was used where possible. The changes made to the TP06 model are listed below:

- the transient outward current maximal conductance (I_{to} - see Equation (C.18)) was reduced by 48% [52],
- the inward rectifier potassium current (I_{K1} - see Equation (C.25)) was reduced by 44% [11],
- the sodium-potassium pump current (I_{NaK} - see Equation (C.30)) was reduced by 40% [98],
- the sodium-calcium exchanger current (I_{NaCa} - see Equation (C.29)) was increased by 80% [44]
- the sarcoplasmic reticulum uptake current (I_{up} - see Equation (C.67)) was reduced by 30% [25, 49].

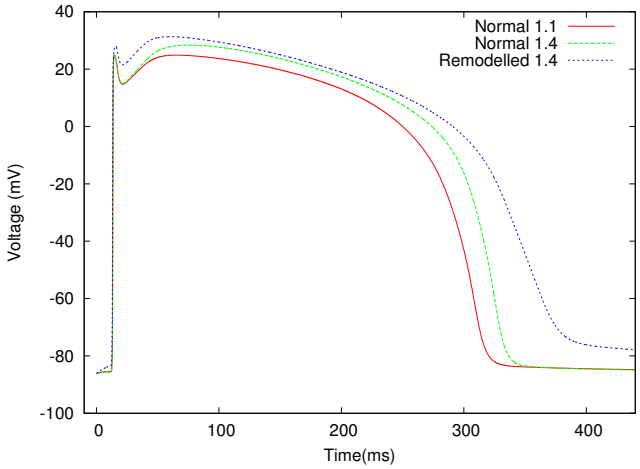
These changes are referred to as the 'remodelled electrophysiology' in the remainder of this chapter.

The remodelled electrophysiology was implemented in the I_{ion} component of Equation (2.1) [110] and tests undertaken on a 360 mm $\times$ 3.75 mm domain with the transmembrane voltage recorded for a single interior node at each time-step. The results from these simulations can be seen in Figure 7.1. These produce the expected changes in the cell action potential, that is, an increased resting membrane potential and prolonged duration.

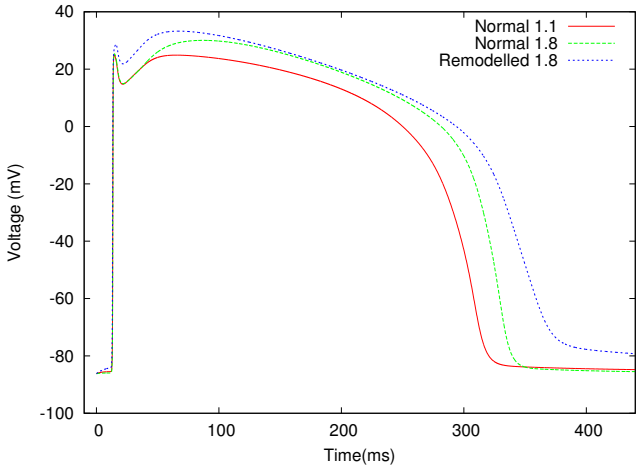
When the electrical wave depolarizes the cell membrane, it results in an increase in cytosolic (within the cell membrane) calcium and this is called the calcium transient. Tests were undertaken to see how the remodelled electrophysiology affected the calcium transient. The tests for the changes in the calcium transient were undertaken on a square 120 mm $\times$120 mm domain and a spiral wave was set to form using the plane wave splitting technique (see Section 2.6.1). Initially the simulation was undertaken for the first 5000 ms with the normal electrophysiology and then after 5000 ms the remodelled electrophysiology was introduced. The tests were undertaken in this way to enable a stable spiral wave to form. The results of these tests can be seen in Figure 7.2 and show that when the electrophysiology is remodelled the calcium transient has a decreased diastolic (resting) level and a decreased peak of the calcium transient.



(a)



(b)



(c)

Figure 7.1: *Transmembrane voltage for a node with normal and remodelled electrophysiology. Modelled on a 360 mm × 3.75 mm domain and time-step=0.08 ms. The voltage is recorded for a single node at each time step, (a) with a restitution slope of 1.1, (b) with a restitution slope of 1.4 and (c) with a restitution slope of 1.8.*

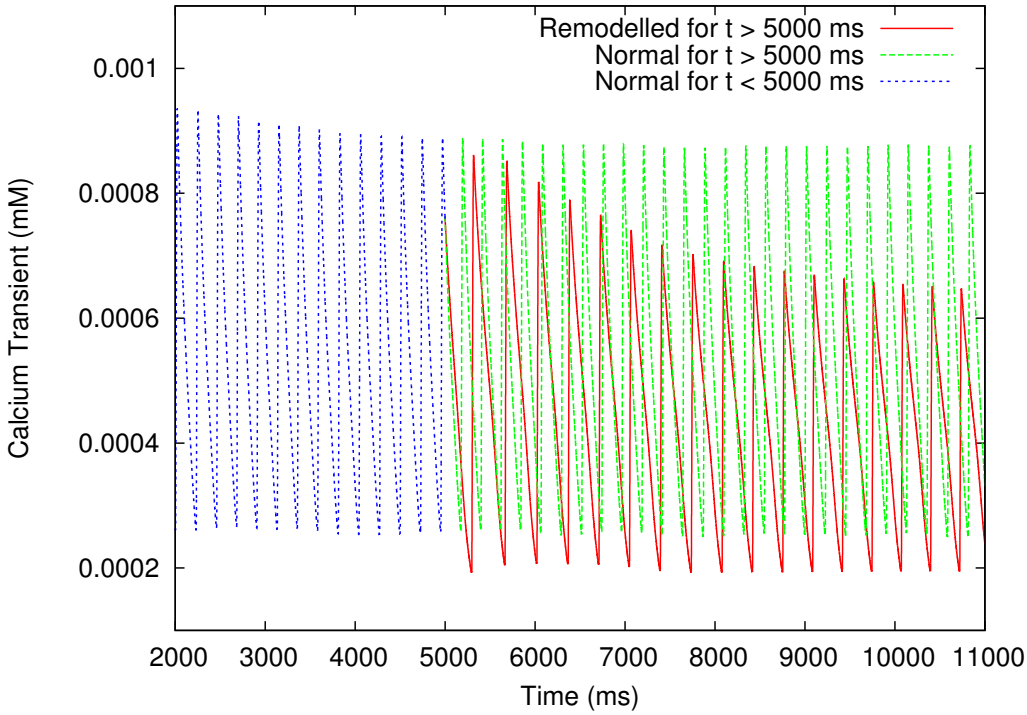


Figure 7.2: *Calcium remodelling for end-stage disease. Normal electrophysiology for* $t < 5000$ *ms. The remodelled (as per Section 7.2.1) electrophysiology introduced from t=5000ms. Tests undertaken on a domain as per row 1 of Table 7.1. The electrical wave was spiralling within the domain causing the periodic output.*

7.2.2 Gap junction remodelling

In heart failure, gap junctions (that allow current flow between cells) are re-organised, and instead of being principally located at the ends of the cells, they become "lateralised" so that there is an increase in transverse gap junction numbers and a decrease in longitudinal gap junction numbers within a single cell. However, because of the changes in cell size with the hypertrophy (an increase in the size of muscle tissue due to an increase in the size, rather than the number, of muscle cells) that accompanies heart failure, there is no net change in gap junction numbers in the transverse direction, but still a net decrease (of between 28% [1] and 40% [27]) in longitudinal gap junction numbers [51]. These changes can be modelled by reducing diffusion only.

Gap junction changes will affect cell resistivity but, because the electrical model, Equation (2.1), treats the tissue as a continuum, this resistivity is made up of a "myoplasmic resistivity" (caused by the structures inside the cell) and a "junctional resistivity" (caused by the gap junctions between cells). It is assumed that junctional resistivity

accounts for 22% of total resistivity [112] and that a reduced gap junction expression (33% for our simulations) is followed by a corresponding increase in junctional resistivity [112]. To simulate this the diffusion rate is reduced in the fibre direction (as compared to Equation 2.37), but remains the same in the cross fibre direction, as follows:

$$\mathbf{D} = \begin{bmatrix} 0.1390 & 0.01711 \\ 0.01711 & 0.1390 \end{bmatrix}. \tag{7.1}$$

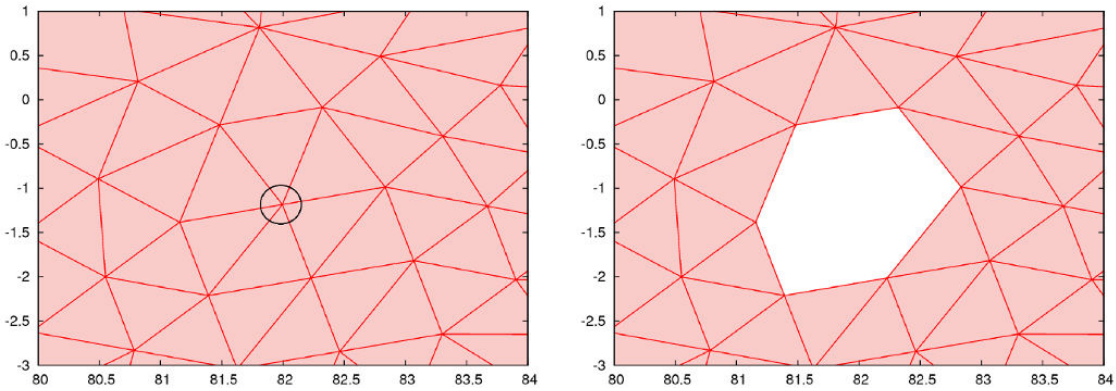
7.2.3 Tissue fibrosis

Cardiac tissue is composed of muscle cells and a network of fibrous non-conductive tissue which anchors the muscle cells and determines the muscle structure. This fibrous structure is known as the fibrotic tissue. The amount of fibrotic tissue within the cardiac muscle contributes to an increase in the incidence of atrial and ventricular arrhythmias [3, 28]. Exactly how the fibrosis contributes to the generation of arrhythmias is unknown, but impaired electrical conduction is a significant contributory factor [105,109]. In the normal healthy heart, approximately 6% of cardiac muscle is made from extra-cellular connective tissue [92,109]. However in a diseased heart, there is increased formation of fibrotic cells which increases the percentage of connective tissue to between 10% and 35% [92, 109].

To simulate the effects of diffuse fibrosis a large number of small areas within the domain were set as in-excitable. This is comparable to the technique used in [109], however as the simulations are undertaken on an unstructured mesh these fibrotic areas are not uniform. The technique used within this thesis is to select a proportion of the nodes within the mesh and mark these as fibrotic. The elements connected to these marked nodes are then flagged as fibrotic, creating small regions (typically of 6 individual elements) in the FEM mesh. These areas become "islands" within the electrical mesh, whereby the nodes bordering the island region become boundary nodes, with no-flux Neumann boundary conditions. The central node is removed from the electrical system altogether. For example, in Figure 7.3 image (a) has a node circled that is to be marked as fibrotic. The right hand mesh (b), illustrates the mesh resulting from this process, where 6 elements have been removed from the electrical mesh.

The fibrotic regions remain in the mechanical mesh and maintain their passive deformation properties, however their active tension is set to zero. As the mesh is made of unstructured triangles these fibrotic regions are irregularly shaped. The only constraint applied in the creation of these regions was to mandate that no two regions could be touching.

In the initial simulations, approximately 26% of the tissue was set as inexcitable,



(a) Mesh before fibrosis (b) Mesh after fibrosis

Figure 7.3: *Building a mesh for fibrosis modelling*

as this is well within the range seen experimentally in [92, 109]. In Section 7.4.5 the percentage of the tissue marked as fibrotic is varied.

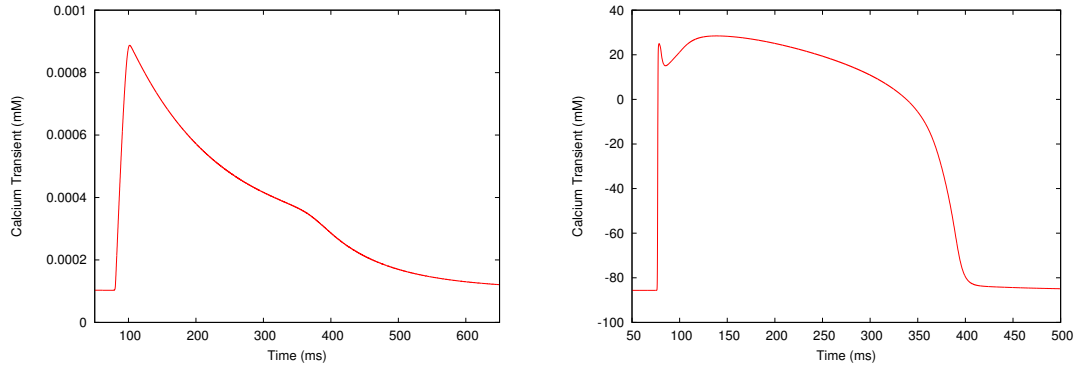
7.3 Calcium transient and tension

In the original coupled model presented in Chapter 4 the active tension is calculated from the transmembrane voltage using Equations (4.2) and (4.3). However in cardiac tissue the tension generated by the cells is determined by the calcium transient [47]. The calcium transient has a much steeper wave descent than the transmembrane voltage (see Figure 7.4). Additionally, the heart failure changes introduced in this chapter have an effect on the calcium transient, and a voltage-derived active tension variable would not be affected by this.

To consider the coupling effects of the difference in voltage and calcium wave profile, and to enable the electrophysiological remodelling changes to make a difference to the coupled model, the equations from [69] were amended to use the calcium transient as their input variable. The evolution of the active tension is now governed by:

$$\frac{\mathrm{d}T_a}{\mathrm{d}t} = \varepsilon(Ca_s)(K_{T_a}Ca_s - T_a), \tag{7.2}$$

where Ca_s is the calcium transient scaled between $0 \leq Ca_s \leq 1$, K_{T_a} controls the amplitude of the active tension (T_a), the scaling of Ca_s was achieved by running simulations and capturing the minimum and maximum values, T_a is initially set to zero and the function



(a) Calcium transient (b) Transmembrane voltage

Figure 7.4: *Calcium transient compared to transmembrane voltage*

$\varepsilon(Ca_s)$ is defined by:

$$\varepsilon(Ca_s) = 10\varepsilon_0 \text{ for } Ca_s < 0.005, \qquad \varepsilon(Ca_s) = \varepsilon_0 \text{ for } Ca_s \geq 0.005. \tag{7.3}$$

In common with the voltage based equations (see Equations (4.2) and (4.3)), Equation (7.2) is approximated using an explicit Euler method, and the upper bound of the active tension (T_a) is still governed by the constant K_{T_a}. The use of Equations (7.2) and (7.3) means that the active tension still has the same range of values, between zero and the K_{T_a} parameter. The profile of the tension changes however, and this can be seen when comparing Figure 7.5 and Figure 7.6. Figure 7.5 shows the active tension generated with the calcium-based equations, with the normal and remodelled electrophysiology. Figure 7.6 shows the active tension generated using the voltage based equations. With a voltage input for active tension the tension remains higher for longer, however with a calcium input the tension reduces more quickly.

7.3.1 Comparison of active tension from calcium and voltage

Tests were also undertaken to consider the effects of using a calcium-based or voltage-based equation (see Equations (4.2), (4.3) and (7.2), (7.3)) for the generation of active tension. The results can be seen in Figure 7.7 and show that the voltage-based active tension generates slightly more deformation in the domain than the calcium-based equation.

Figures 7.8 and 7.9 illustrate the effects that the two different approaches to calculating the active tension have on the location of the top left hand node of the domain. These images show that by changing to a calcium-based active tension generation method, the maximum deformation of the domain is slightly smaller when the electrophysiology is

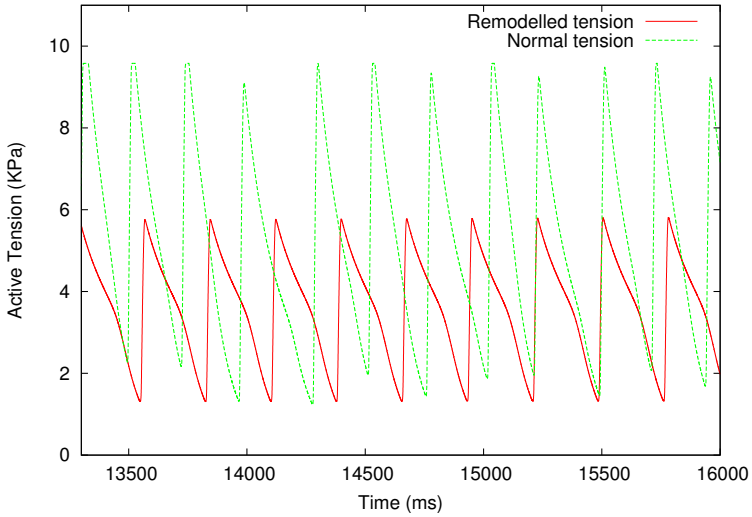


Figure 7.5: *Active tension using remodelled electrophysiology. Comparison of active tension generated using Equations (7.2) and (7.3), with normal and remodelled electrophysiology. Tests undertaken on a domain as per row 1 of Table 7.1. The electrophysiology remodelling introduced from t=5000 ms. The electrical wave was spiralling and hence the tension oscillates as the wave spirals.*

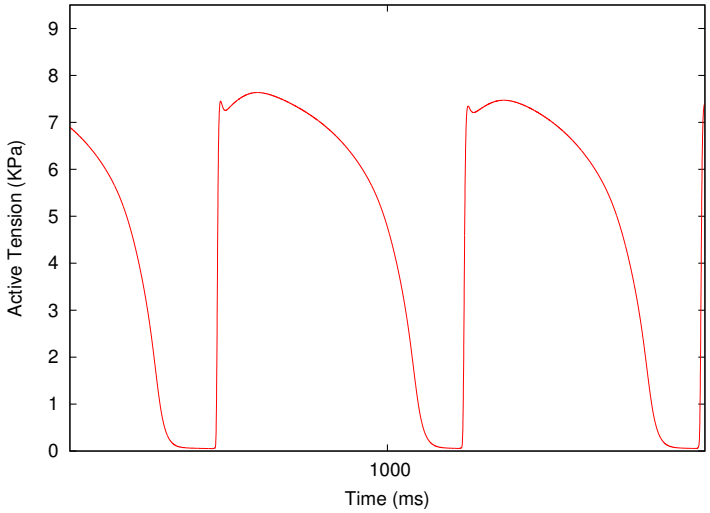
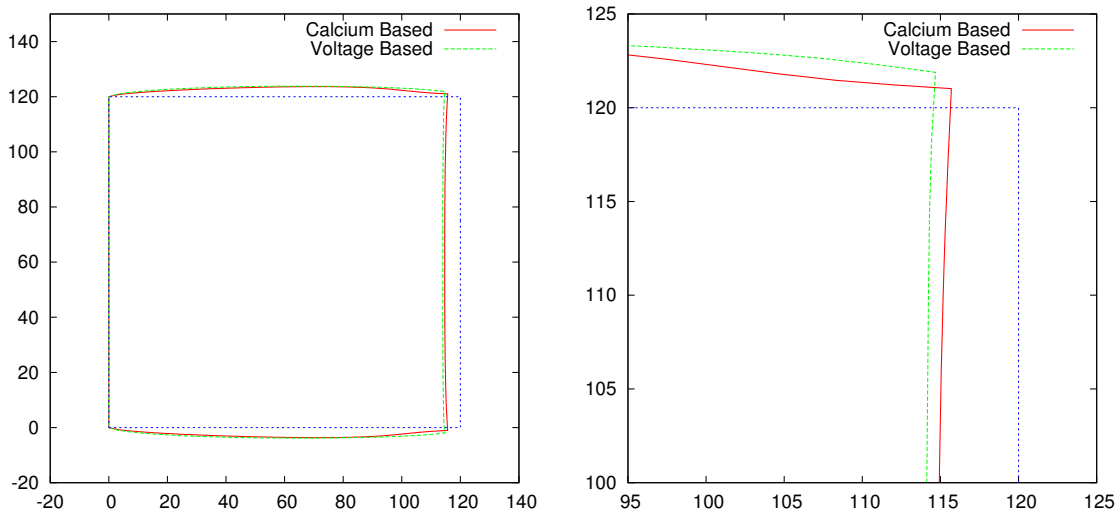


Figure 7.6: *Active tension at a single interior node using voltage-based equations (see Equations (4.2) and (4.3)). Tests undertaken on a domain as per row 1 of Table 7.1. The electrical wave was spiralling and hence the tension oscillates as the wave spirals.*



(a) Full domain (b) Top right Corner

Figure 7.7: *Deformation effects of voltage-based and calcium-based active tension. Left edge $X_1 = 0$ fixed and stimulated to form a plane wave. Using both the calcium and voltage based equations for active tension generation, at t=136 ms, on a mesh with 2309 initial nodes and 20781 degrees of freedom. Original domain edge is also displayed as a reference.*

remodelled. As can be seen in Figure 7.5, when the electrophysiology is remodelled the maximum tension is reduced and so the behaviour seen in Figure 7.8 is expected. It is also noteworthy that although the maximum tension in the remodelled electrophsyiology drops by approximately a third, the deformation is only slightly decreased.

It should be noted that in this chapter the domain is rotated 45 degrees as described in Section 3.2.5, which provides the means of modelling fibre orientation parallel to the X_1 axis, and this should be considered when viewing the displacements in Figure 7.8 and Figure 7.9.

7.4 Simulations and Results

7.4.1 Simulation settings

The results in this section were obtained by running simulations on the domains as specified in Table 7.1. The initial starting values for the TP06 state variables were as per Appendix C.2, the parameters for K_{T_a}, c_1, c_2 and the I_{Na} dynamics were the same throughout the tests (see Table 7.2).

The tests were designed to determine the stability of the electrical spiral wave under

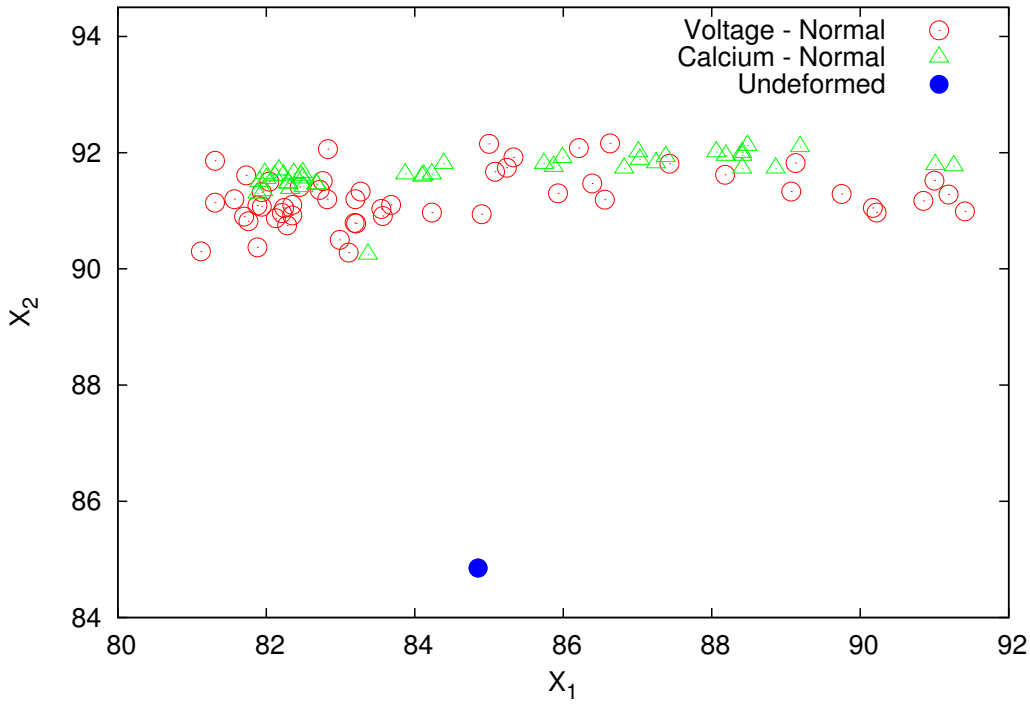
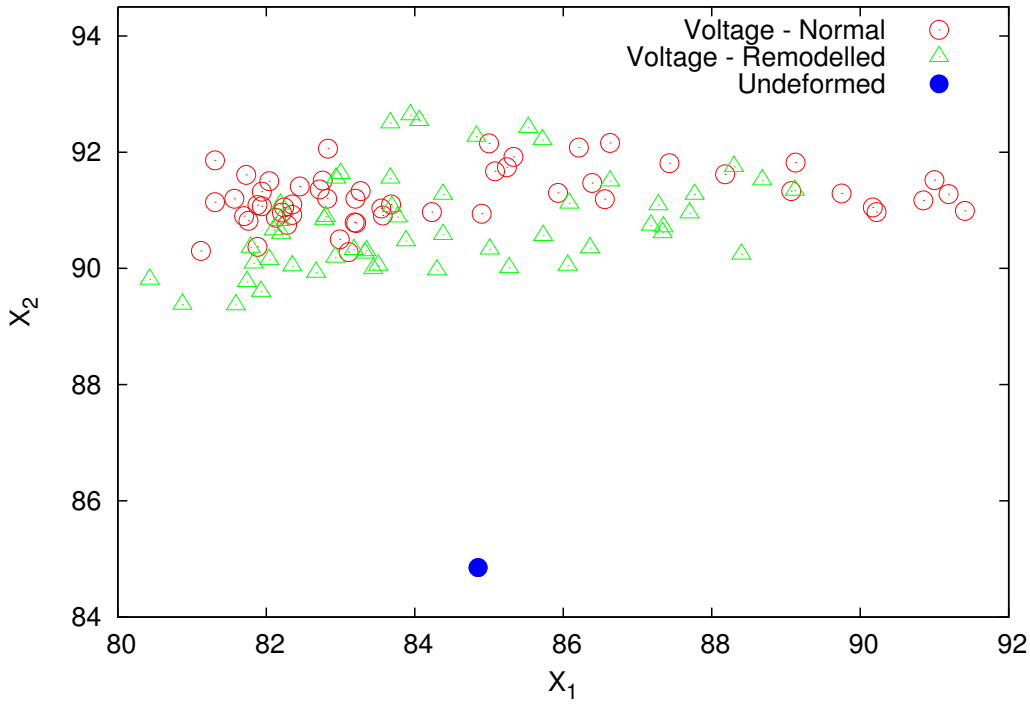
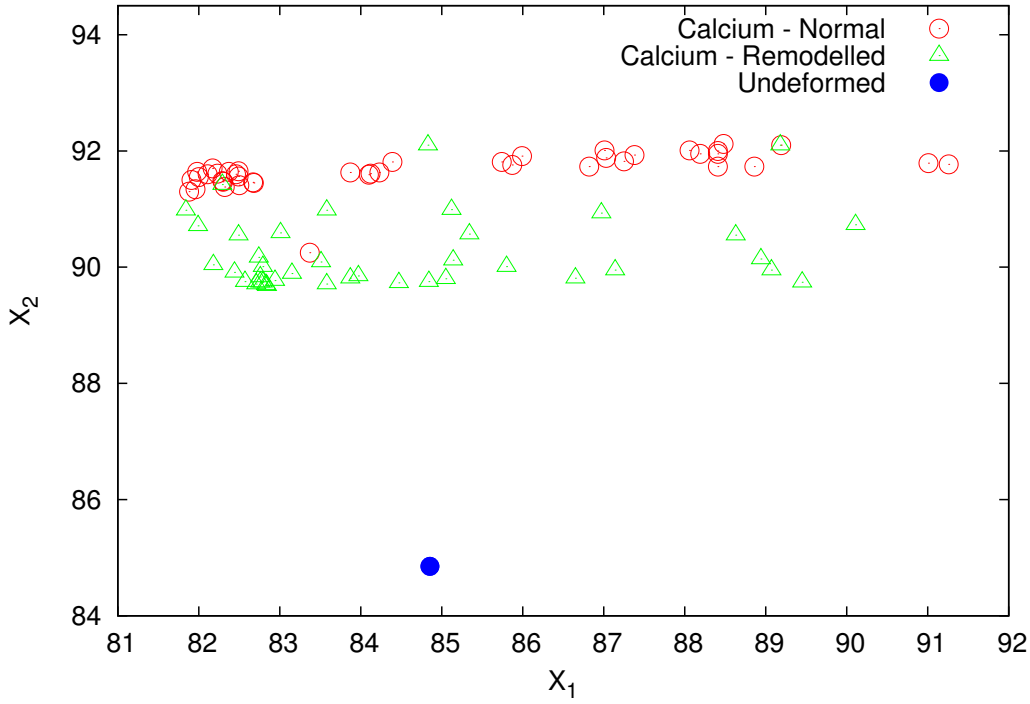


Figure 7.8: *Corner node displacement using voltage and calcium based active tension. Range of positions of top left hand corner node. For voltage runs samples were taken from t= 4000 ms to t=6120 ms, with samples every 40 ms. For calcium runs time period was from was t=4800 ms to t=12800 ms with samples every 200 ms.*



(a) Voltage-based



(b) Calcium-based

Figure 7.9: *Corner node displacement using voltage and calcium based active tension. Range of positions of top left hand corner node. For voltage runs samples were taken from t= 4000 ms to t=6120 ms, with samples every 40 ms. For calcium runs time period was from was t=4800 ms to t=12800 ms with samples every 200 ms.*

Num	Domain Size	Nodes	Elements	Time-step	Diffusion	Fibrosis
1	120×120 mm	318065	634368	0.08 ms	Anisotropic - Equation (2.37)	No
2	120×120 mm	243739	465856	0.08 ms	Anisotropic - Equation (7.1)	Yes

Table 7.1: *General domain settings for heart failure tests.*

Parameter	Description	Value
K_{T_a}	Active tension maximum	9.58 kPa
c_1	Strain energy function parameter	2 kPa
c_2	Strain energy function parameter	6 kPa
I_{Na} dynamics	I_{Na} dynamics	Standard I_{Na} dynamics [110]

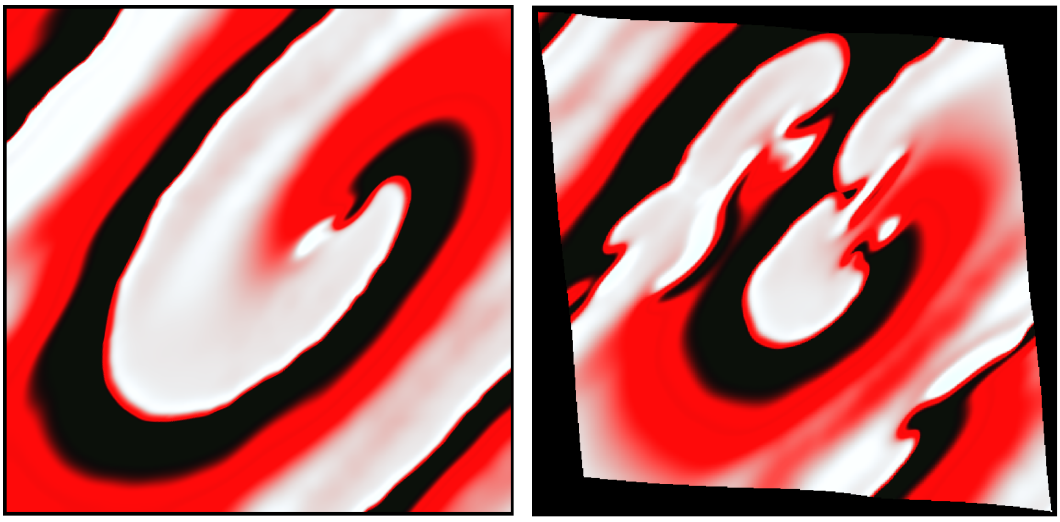
Table 7.2: *Unchanging parameters for heart failure tests.*

different conditions and so all the simulations were run until $t = 5000$ ms with a restitution slope of 1.1 and the normal electrophysiology. This was to ensure a stable spiral wave had formed. After $t = 5000$ ms the simulation settings were amended to introduce the conditions being tested. In all cases tests were carried out with restitution slopes of 1.1, 1.4 and 1.8. For the simulations with fibrotic regions the simulations were run from the outset on the fibrotic domain (see row 2 of Table 7.1). For the simulations with normal (no fibrotic regions) tissue they were run from the outset on a normal domain.

7.4.2 Deforming domain

In Section 4.4 and in [57], it was demonstrated that spiral wave stability can be affected by the deformation of the domain. The active tension model in [57] was taken from [69] and uses transmembrane voltage to determine the active tension generated at each node over time. However, as discussed in Section 7.3, the active tension is determined by the transient calcium and therefore, in the simulations in this chapter, Equations (7.2) and (7.3) are used to calculate the active tension.

When the calcium-based active tension equations were implemented it was noticed that the calcium transient takes a long time to build to the range of values seen in Figure 7.2, therefore to enable the calcium transient to be within the normal range immediately, the initial values of the TP06 state variables within the I_{ion} component of the electrical model (see Equation (2.1)) were changed. The new values were obtained by simulating a single cell repeatedly at a 1 Hz frequency until the stable range of values was produced,



(a) Static (b) Deforming

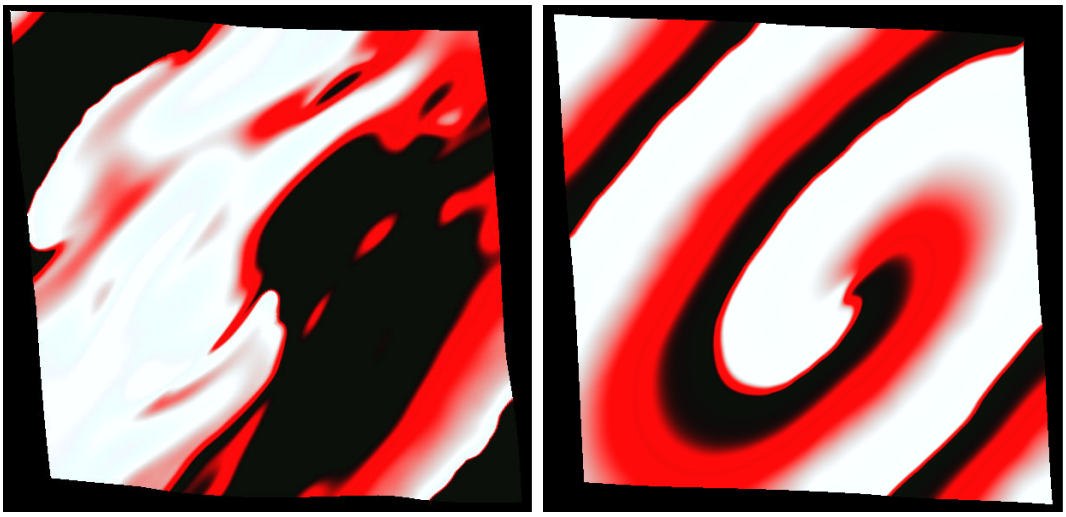
Figure 7.10: *Static and deforming domain with restitution slope of 1.4 (t=22,800ms). Tests undertaken on a domain as per row 1 of Table 7.1. The restitution slope of 1.4 was set from t=5000 ms and the stable initial values for the TP06 model were used (see C.1 were used).*

the initial values were then set to the resting value of this range. This was undertaken to ensure the calcium transient varied within the expected range from the outset of the simulations.

The first sets of simulations considered the effects of using active tension derived from the calcium transient (as opposed to voltage-based active tension), with the stable initial values, within a deforming domain (see Table C.1 for stable and original initial values). In Figure 7.10 the left panel shows the spiral wave in a static domain after $t = 22,800$ ms and the right panel shows the spiral wave in a deforming domain at the same time. It is noted that the simulations need to run for a longer time period than in [57] for the deformation to break up in the deforming tissue. Tests were undertaken with the original starting values for I_{ion} and it was determined that using the stable variable settings causes the spiral to maintain its stability for longer.

7.4.3 Electrical Remodelling

In these simulations the electrical remodelling changes described in Section 7.2.1 were introduced at $t = 5000$ ms. The simulations were undertaken on a deforming domain and undertaken for restitution slopes of 1.1, 1.4 and 1.8. The results of the simulation at $t = 12000$ ms can be seen in Figure 7.11.



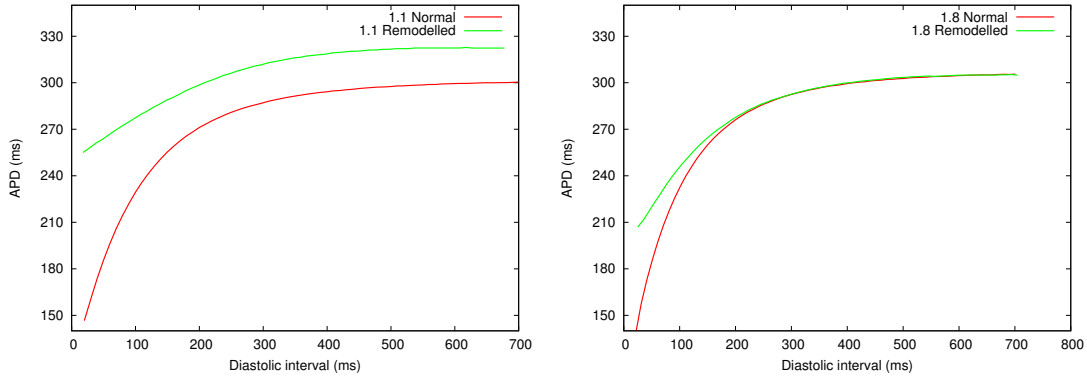
(a) Normal (b) Remodelled

Figure 7.11: *Normal electrophysiology with restitution slope of 1.8 and corresponding remodelled electrophysiology. Tests undertaken on a domain as per row 1 of Table 7.1. The restitution slope parameter was set to 1.8 from t=5000 ms.*

Figure 7.11 demonstrates that the electrophysiology remodelling has a stabilising effect on the wave. With normal electrophysiology the spiral wave has become chaotic (as is seen in other published work [110]), however the remodelled electrophysiology simulation has a steady spiral wave.

When the electrical remodelling is introduced (at $t = 5000$ ms), the speed of the electrical spiral wave decreases from a 5.0 Hz rotation frequency to 3.2 Hz. The transmembrane voltage of a single node in the domain was recorded and Figure 7.2 shows the increased wave length, higher resting potential and higher plateau than with the normal electrophysiological parameters.

The introduction of the remodelled electrophysiology for heart failure also impacts upon the restitution slope of the system. Figure 7.12, which is unpublished data courtesy of Alan Benson and Victoria Peacock, University of Leeds, shows the restitution slope for normal and remodelled electrophysiology. In these Figures, APD is the action potential duration and diastolic interval is the time between the end of one action potential to the initiation of the next. This illustrates that for the remodelled electrophysiology the restitution slope is less steep and this may explain why the spiral waves do not break up. The steepness of the restitution slope and how this affects the stability of a spiral wave is discussed in [17, 67, 110] with a less steep slope providing a more stable spiral wave.



(a) Restitution slope 1.1 (b) Restitution slope 1.8

Figure 7.12: *Restitution slopes of normal and remodelled electrophysiology with restitution slope settings of 1.1 (a) and 1.8 (b). APD is the action potential duration and diastolic interval is the time between the end of one action potential to the initiation of the next. This is unpublished data courtesy of Alan Benson and Victoria Peacock, University of Leeds.*

7.4.4 Fibrotic Regions

In these simulations, regions of non-conducting tissue and gap junction remodelling are introduced as described in Section 7.2. From $t = 5000$ ms the dynamic restitution slope was then kept at 1.1 or changed to either 1.4 or 1.8, and the simulations undertaken until $t = 15000$ ms.

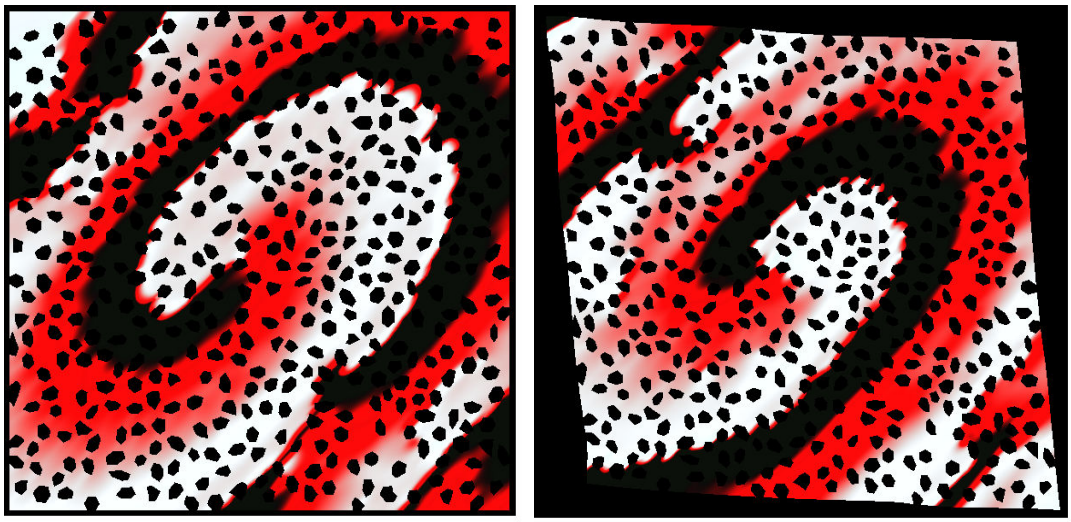
Figure 7.13 shows the results of the simulations at $t = 15000$ ms with the restitution slope set to 1.4. The left panel shows the spiral wave without the fibrotic regions and the right panel shows the results with fibrotic panels enabled. It can be seen that the electrical wave in the simulation with the fibrotic regions has started to break-up into a chaotic state. The results show that the fibrotic regions have an effect on the spiral wave break up. In the simulations with a restitution slope of 1.1 the spiral wave was maintained over time and did not become chaotic. The results in Figure 7.13 are from simulations on a static domain to demonstrate that the fibrotic regions on their own can contribute to spiral wave break-up.

In the deforming domain simulations the break-up of the spiral wave with fibrotic regions occurred at a similar time to a domain without the fibrotic regions. The deforming domain did not noticeably change the time for the break-up to begin. In both the static and deforming domain with fibrotic regions, small wave anomalies appear after $t = 11000$ ms (see Figure 7.14) and then progressively become more chaotic.

In ten Tusscher and Panfilov [109], the introduction of diffuse fibrosis stabilizes the

(a) 1.4 Restitution slope (b) 1.4 Restitution slope (c) 1.1 Restitution slope

Figure 7.13: *Effect of introducing fibrotic regions, with results shown t=15000 ms. Test (a) undertaken on a domain as per row 1 of Table 7.1. Tests (b) and (c) undertaken on a domain as per row 2 of Table 7.1. The restitution slope of 1.4 was set from t=5000 ms.*



(a) Static (b) Deforming

Figure 7.14: *Fibrotic regions after t=11000 ms on static and deforming domains. Tests undertaken on a domain as per row 2 of Table 7.1. The restitution slope of 1.4 was set from t=5000 ms.*

Mesh	Approx. area of each fibrotic region	Percentage of domain area	Nodes	Elements
Mesh F0	8.72 mm^2	0.0605%	243739	465856
Mesh F1	1.93 mm^2	0.0134%	284782	523632
Mesh F2	0.53 mm^2	0.0037%	275121	476212
Mesh F3	0.27 mm^2	0.0019%	539848	934984
Mesh F4	1.11 mm^2	0.0077%	495219	912448

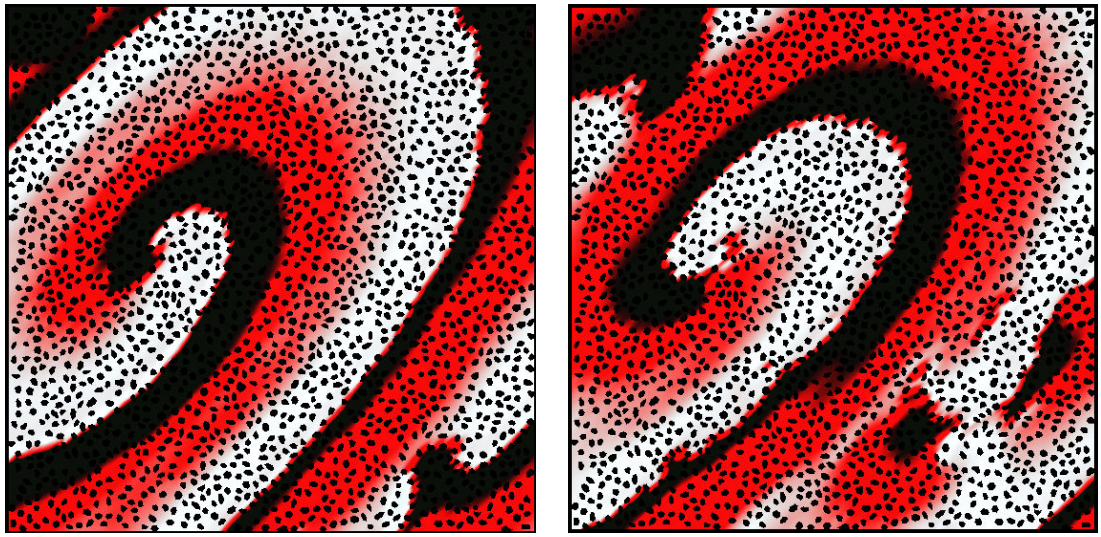
Table 7.3: *Fibrotic region sizes and percentage of overall areas. In all instances the domain size was 120 mm$\times$120 mm, and the diffusion set to anisotropic as per Equation (7.1).*

spiral wave, however the simulations shown here do not show this effect. For these simulations, a single in-excitable region has an area of 0.06% of the whole domain and the fibrotic regions are non-uniform in shape. For the spiral wave break-up tests in [109] an 800x800 uniform mesh is used, with a total of 640000 finite difference 'elements'. Each fibrotic region is a uniform 1x1 square and so as a percentage of the overall domain is 0.00016 %. In the simulations in this section the fibrotic regions are approximately 380 times larger. The increased size of the fibrotic regions and their irregularity may be the cause of this opposite effect. This is investigated further in Section 7.4.5.

7.4.5 Varying fibrotic element size and concentration

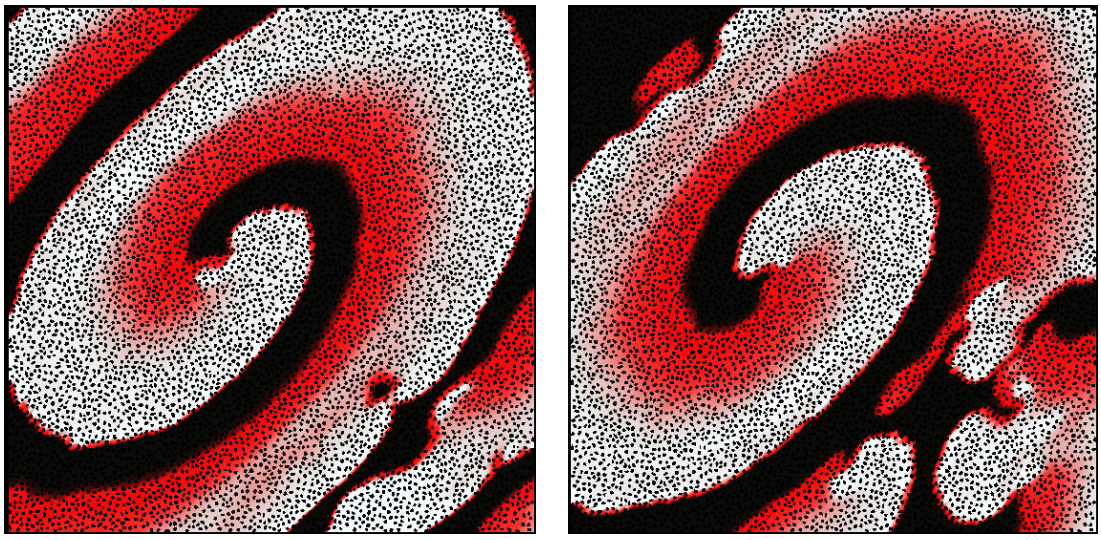
To investigate the role of the size of the fibrotic area, further tests were undertaken with smaller fibrotic regions. These tests were over the same domain size (120 mm $\times$ 120 mm) and with the same overall area of tissue marked as in-excitable, however each individual fibrotic area was smaller in area (and hence there were more of them). The approximate area of each of the individual fibrotic elements was progressively reduced, as can be seen in Table 7.3, and tests undertaken with a restitution slope of 1.4 implemented at t=5000 ms. For the Mesh F3 in Table 7.3, the number of elements and nodes approximately doubles from the previous meshes. This is because the fibrotic elements are built from the coarsest mesh (before refinement), and so to make the fibrotic areas smaller requires a finer initial mesh. The other parameters were set as per row 2 of Table 7.1.

The results presented in Figures 7.15, 7.16 and 7.17 show that the size of the individual fibrotic regions has an impact on the spiral wave stability. In Figures 7.15 and 7.16 the spiral wave breaks up into a chaotic state, but in Figure 7.17, which uses the smallest individual fibrotic regions, the spiral wave is maintained over time. This provides a closer



(a) t=1040 ms (b) t=1360 ms

Figure 7.15: *Simulation of fibrotic regions on Mesh F1 (see Table 7.3 for settings).*



(a) t=12000 ms (b) t=17360 ms

Figure 7.16: *Simulation of fibrotic regions on Mesh F2 (see Table 7.3 for settings).*

Figure 7.17: *Simulation of fibrotic regions on Mesh F3 (see Table 7.3 for settings), at t=22400 ms.*

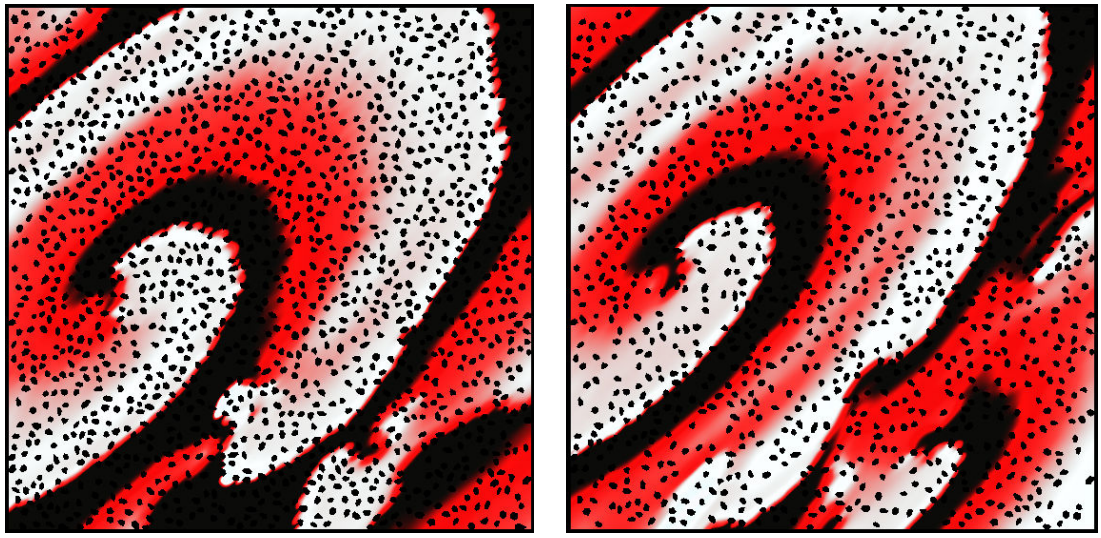
Mesh	Area of excitable elements	Percent domain excitable	Stable wave?
Mesh F5	10539mm^2 of 14400mm^2	73.19%	No
Mesh F6	10827mm^2 of 14400mm^2	75.19%	No
Mesh F7	11496mm^2 of 14400mm^2	79.83%	No
Mesh F8	12184mm^2 of 14400mm^2	84.61%	No
Mesh F9	13667mm^2 of 14400mm^2	94.91%	No

Table 7.4: *Fibrotic region sizes and different percentages of overall areas*

comparison to [109], where the fibrotic regions were at a smaller level still. As Mesh F3 is a finer mesh than the previous results in this thesis, further tests were undertaken using Mesh F4 from Table 7.3. This mesh has a similar number of elements, however the fibrotic regions were larger. The tests undertaken on Mesh F4 also caused the spiral wave to break-up and confirmed that it was not the mesh resolution that caused the spiral wave to be stable.

Further tests were then undertaken that varied the quantity of fibrotic elements in the domain, and hence the overall percentage of the domain that is in-excitable. Table 7.4 shows the different scenarios that were tested. The original meshes for these were based on Mesh F1 in Table 7.3, which has an approximate fibrotic region area of 1.93 mm^2, and each is approximately 0.0134% of the domain area. The results for these tests can be seen in Figure 7.18 and these show that with a smaller percentage of the domain set as in-excitable the spiral wave still becomes unstable.

Further simulations were undertaken with reduced fibrotic area, in which only 5% of the domain was marked as in-excitable (this is Mesh F9 in Table 7.4). The results from these tests can be see in Figure 7.19 and these are compared to the same resolution domain with no fibrotic areas. Both these tests were undertaken with a restitution slope of 1.4. These results show that a small number of fibrotic elements can have a significant impact on the spiral wave. In images (c) and (d) of Figure 7.19 a smaller spiral wave has split from the main wave and formed in the top left hand corner of the domain. This does not happen in the normal domain (images (a) and (b)) at the same time step. This would suggest that the size and location of fibrotic regions can play an important role in destabilising a spiral wave. If this is compared with the results presented in Figure 7.17 it would further suggest that the overall area of in-excitable tissue is less important than the size of the fibrotic regions and their distribution in the domain.



(a) Mesh F7, t=13,600 ms (b) Mesh F8, t=13,200 ms

Figure 7.18: *Varying area of total fibrotic regions (see Table 7.3 for domain settings)*

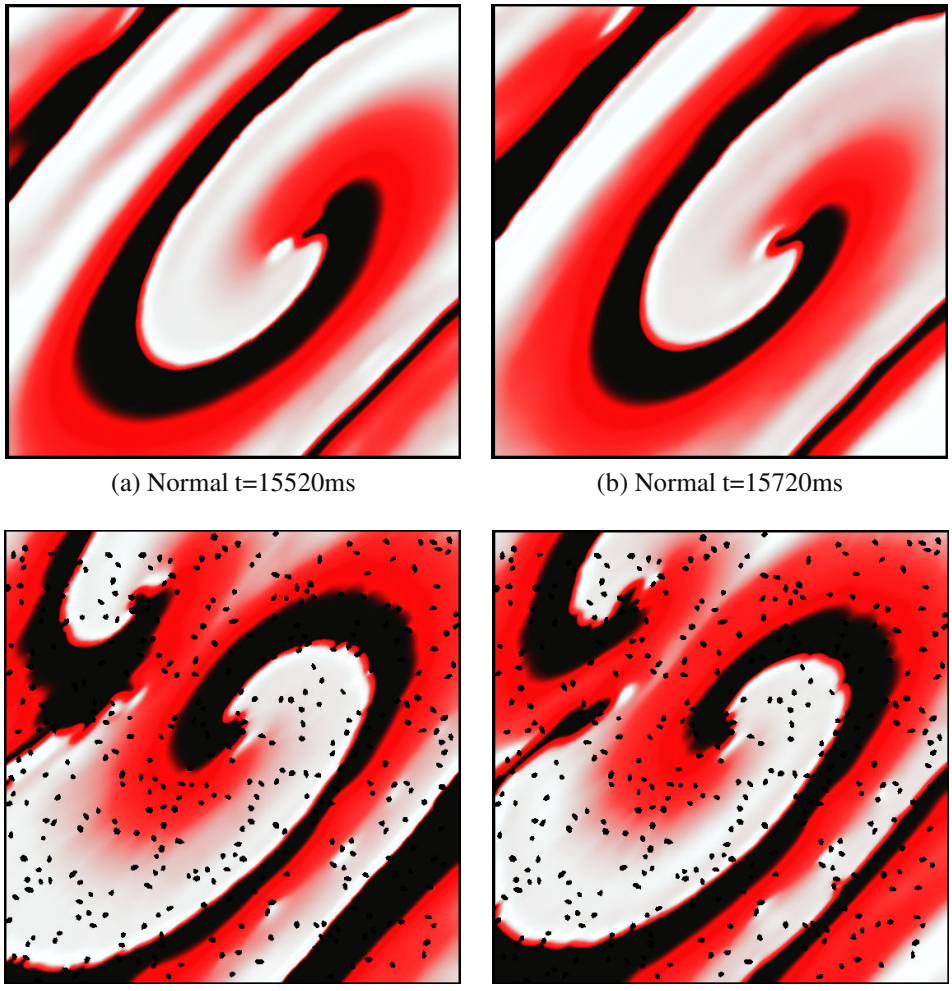
7.4.6 Electrical remodelling and fibrotic regions

In these simulations the in-excitable fibrotic regions are present and the remodelled electrophysiology was introduced at $t = 5000$ ms and run until $t = 12000$ ms (as described in Section 7.2). This was repeated for dynamic restitution curves of 1.1 (no change), 1.4 and 1.8. These simulations were undertaken on both a static and a deforming domain. The deforming domain results at $t = 12000$ ms can be seen in Figure 7.20. Panel (b) of Figure 7.20 shows the spiral wave is stable over time. These simulations have a restitution slope of 1.8, have fibrotic regions and are in a deforming tissue. These three parameters have caused the spiral wave to become chaotic in the previous simulations, however with the remodelled electrophysiology the spiral wave remains stable.

7.5 Conclusions

This chapter has demonstrated that by changing the active tension equations to be based on the calcium transient the amount of deformation in the system will change as the calcium transient changes. The two models of active tension used in this thesis produce similar deformations and the change to calcium-based active tension (with the normal electrophsyiology) did not have a qualitative effect on the spiral wave stability.

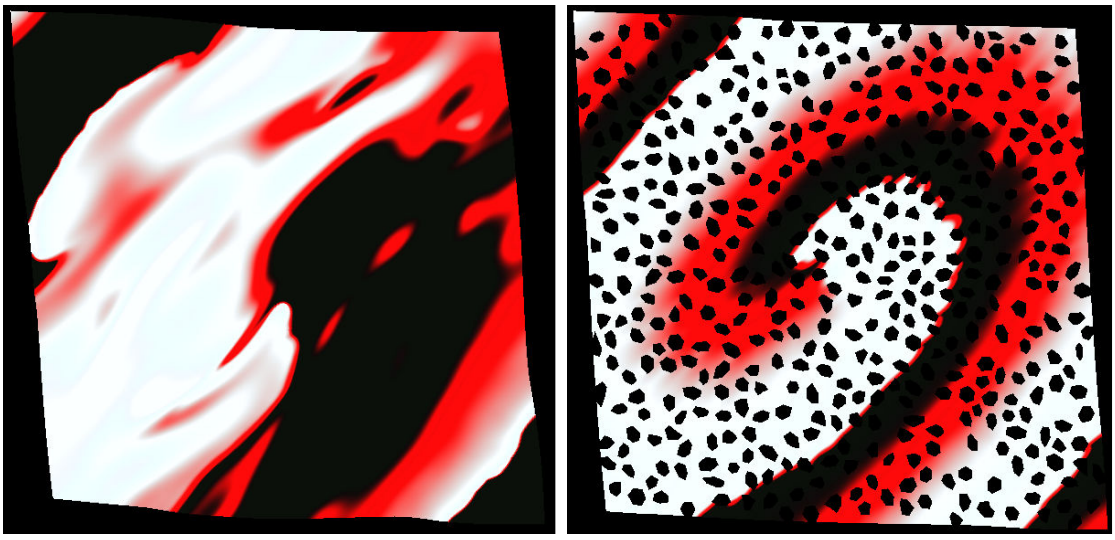
The chapter has further demonstrated (along with Section 4.4) that a spiral wave that is on the edge of stability (e.g. with a restitution slope of 1.4) can be made chaotic within a coupled deforming domain. This suggests that electrophysiology modelling should be



(a) Normal t=15520ms (b) Normal t=15720ms

(c) Mesh F9 t=15520ms (d) Mesh F9 t=15720ms

Figure 7.19: *Fibrotic regions of 5% the domain area (Mesh F9 from Table 7.4). Comparison of results with a 5% of domain set as fibrotic with normal domain. The normal mesh is 120 mm×120 mm with 359401 nodes and 716928 elements.*



(a) Normal (b) Fibrotic with remodelled electrophysiology

Figure 7.20: *Normal domain with restitution slope of 1.8 and corresponding fibrotic domain with remodelled electrophysiology. Test (a) undertaken on a domain as per row 1 of Table 7.1. Test (b) was undertaken on a domain as per row 2 of Table 7.1.*

carried out on a moving domain if stability of the wave is being considered. Also it was noticed that the initial values of the TP06 state variables need to be carefully considered. If these are not set correctly, then early in the simulations, the voltage may be within the expected range, however other variables (e.g. calcium transient) may not be.

By remodelling the electrophysiology (as in Section 7.2.1) the cardiac wave becomes longer, with a higher plateau and higher resting state (see Figure 7.1). When this electrophysiology is introduced the spiral waves in the system become much more stable and introducing a higher restitution slope (1.8), fibrotic areas and deforming the domain still does not cause the wave to break-up. The remodelled electrophysiology has the effect of making the restitution slope less steep (see Figure 7.12), and this may explain stabilisation.

The introduction of in-excitable regions of fibrotic tissue can cause a stable spiral wave to break-up into a chaotic state. This effect can be removed if the individual fibrotic regions are very small (see Figure 7.17), for example using Mesh F3 of Table 7.3 where an individual fibrotic element is approximately 0.27 mm^2.

By varying the amount of the tissue marked as in-excitable it was possible to demonstrate that the size of the individual fibrotic areas, rather than the total area of them, plays a role in the stability of the spiral wave. In Figure 7.19 a micro-spiral wave is formed in the top left hand corner of the domain, and only 5% of this domain is marked as fibrotic.

Further work could be carried out here to consider a more realistic fibrotic region size and shape, and to consider whether the distributions of these areas is also a contributory factor to wave break-up.

Chapter 8

Conclusions and further work

8.1 Conclusions and discussion

This thesis has presented a coupled model of cardiac electromechanical activity, using the finite element method to model both electrophysiology and mechanics. The performance and efficiency of the electrical component of the model were improved with the addition of local mesh adaptivity and the performance of the mechanical component was improved by the addition of preconditioning. The resulting solver was then used to consider how known changes in cardiac electrophysiology, which are manifest in end-stage heart disease, affect the stability of the electrical wave when spiralling.

The electrophysiology model uses the ten Tusscher-Panfilov (TP06) second generation detailed cellular model [110], which includes anisotropic diffusion, uses a semi-implicit time stepping scheme, stores data in an efficient sparse storage format and uses a Reverse Cuthill-McKee ordering algorithm to reduce the matrices' bandwidths. Spatial and temporal convergence tests were undertaken and, in common with other authors, it was demonstrated that the steep up-slope of the electrical wave front requires a fine mesh to be properly resolved.

The cardiac mechanical model is based on finite deformation elasticity theory, enforces the incompressibility of the tissue and uses a technique (from [82]) to provide anisotropic tension to simulate fibre orientation. This uses isoparametric quadratic elements for deformation, linear elements for pressure, was integrated with numerical quadrature and the resulting non-linear system solved with the iterative Newton method.

The electrical and mechanical systems were coupled together using the active tension variable (T_a), with a phenomenological approach taken from [69]. Convergence tests were undertaken on the coupled system and the results were compared to other published work [82]. The convergence tests undertaken demonstrated that node displacements converge

at a rate that is approaching quadratic, as the element size decreases.

The coupled solver was used to simulate changes in an electrical wave with varying restitution slopes of 1.1, 1.4 and 1.8 (as per [110]). These tests show that the deformation of the domain can affect the stability of an electrical spiral wave, with a previously stable wave (with a restitution slope of 1.4), breaking up into chaotic patterns. This illustrates that the modelling of cardiac electrophysiology on a static domain may not accurately represent the wave form as the deformation can contribute to spiral wave break-up.

The mechanical component of the coupled solver required significantly more CPU time to solve than the electrical component, and ILUT preconditioning was added to the mechanical solve to improve this. In the best-performing example the preconditioned system used 27 times less CPU time than the non-preconditioned system. This is the result of a large drop in the required number of inner iterations for the Newton method. Furthermore, the ILUT preconditioner provides parameters which can be altered to optimise the solver as the system size increases. By introducing preconditioning, coupled simulations could be undertaken in a number of days (rather than weeks).

A local adaptive mesh refinement strategy was applied to the electrical component of the coupled solver. This demonstrated that an AMR strategy can improve efficiency, lower the system resources required and produce results with a level of accuracy that preserves the qualitative behaviour of the system. This also means that with given computer resources, the AMR-based solution can provide finer resolution around the wave front than a globally refined system would be able to. The AMR technique introduced in this thesis enabled the investigation of how targeting the extra mesh refinement at the front and/or back of the wave front affects the overall accuracy. The conclusion was that it is critical to ensure the front of the wave is approximated within a fine mesh, however the remainder of the wave can be approximated on a less refined mesh and still produce quantitatively accurate results. The technique employed does not require a posteriori error estimates, rather the monitor functions employed consider changes in the transmembrane voltage, and so the overhead of calculating the error estimates is removed.

The AMR technique was tested on static and deforming domains with both line waves and spiral waves. In both instances significant improvements were seen in computer resource usage, and the line wave simulations performed up to 30 times faster when AMR was used. The improvement achieved is proportional to the amount of the domain that has excited tissue, with a spiral wave having a higher percentage of the domain excited. The author is not aware of other published work in which an AMR technique within the electrical system is undertaken on a deforming domain.

In Chapter 7, changes were made to the TP06 electrical model to represent tissue

in end-stage disease. These changes affected both the action potential wave profile and the calcium transient. To take advantage of the remodelling of the calcium transient, a new method for calculating the active tension was proposed. This was based on the phenomenological approach of [69], but used the calcium transient as the means of varying the tension within the domain. The two models of active tension used in this thesis produce similar deformations and the switch to calcium-based active tension (with the normal electrophsyiology) did not have a qualitative effect on the spiral wave stability.

It was further demonstrated (along with Section 4.4) that a spiral wave that is on the edge of stability (e.g. with a restitution slope of 1.4) can be made chaotic within a coupled deforming domain. This suggests that electrophysiology modelling should be carried out on a moving domain if stability of the wave is being considered. Also it was noticed that the initial values of the TP06 state variables need to be carefully considered. If these are not set correctly, then early in the simulations, the voltage may be within the expected range, however other variables (e.g. calcium transient) may not be.

When the electrophysiology is remodelled to simulate end-stage heart failure conditions, the spiral waves in the system become much more stable and introducing a higher restitution slope, fibrotic areas and deforming the domain still does not cause the wave to break up. The remodelled electrophysiology has the effect of making the restitution slope less steep (see Figure 7.12), and this may explain the stabilisation.

The introduction of in-excitable regions of fibrotic tissue can cause a stable spiral wave to break up into a chaotic state. This effect can be removed if the individual fibrotic regions are very small. By varying the amount of the tissue marked as in-excitable it was possible to illustrate that, with these parameters, the size of the individual fibrotic areas, rather than the total area of them, plays a role in the stability of the spiral wave.

8.2 Further work

During the course of the thesis a number of areas in which the research could be developed were noted. These are in relation to improving the functionality of the developed solver, progressing the numerical techniques used and introducing more realistic biological simulations. In relation to the biological simulations, areas for further work include:

- Investigation into the size, shape and distribution of the fibrotic regions. In this thesis the fibrotic regions are two-dimensional irregular polygons constructed from a set of triangular mesh elements. Introducing realistic fibrotic region geometries could further highlight their role in the disruption of stable spiral waves.

- The simulations undertaken in this thesis were on two-dimensional domains and this provided a clear way of considering spiral wave stability, however it would be interesting to use three dimensions, especially in the development of accurate fibrotic regions.

- The introduction of more realistic calcium-based active tension generation techniques would provide the ability to better consider the effect that remodelling the electrophysiology has on the deformation of the tissue, and what effect this has on the formation and dissipation of arrhythmias.

- In the process of developing the mechanical solver used in this thesis, a compressible model was initially produced. It would be interesting to undertake a comparison exercise using both compressible and incompressible models. Also the effects of introducing an anisotropic strain energy function would be of interest.

- The electrophysiology and mechanics are coupled with weak coupling, in that they are each solved independently. By adding strong coupling, other biological effects, for example stretch activated channels, could be considered.

In relation to the numerical techniques used in this thesis, potential areas for further work are described below:

- The AMR solution involves the calculation of a projection buffer and the method used for this was to project the buffer a given distance from the wave front. This technique could be improved by using an element connectivity based algorithm, where elements were 'walked-between' to set the projection region. This would involve the determination and storage of linked sets of 'neighbour-elements', however once built, these should provide speed reductions in the mesh refinement and mesh validity check processes.

- The preconditioning technique in this thesis uses a parameter for the re-use of the numerical Jacobian, however in [58], the convergence of the solve is monitored and the output of this used to amend the re-use parameter. Although in this work the re-use of the Jacobian was not the most significant contributor to the preconditioning improvements, extra efficiency may be introduced with this technique.

- The monitor functions implemented provide a simple and efficient means of determining elements for refinement and de-refinement, however these could be developed further, or other monitor functions introduced, to see if this provides extra

increases in efficiency. Similarly the use of other adaptive techniques, for example p-refinement, may increase the performance and efficiency gains.

- Adding adaptivity to the mechanical solver could provide large benefits, as the mechanical solver still dominates the overall solution time.
- Temporal adaptivity has been undertaken by several other research teams, and this would provide further improvement.

Finally, due to the increasing complexity of the available modelling geometries, there is still a need for further improvement in simulation solve time and the use of parallelisation within the solver may be a route to enabling the modelling of accurate geometries with coupling enabled.

Bibliography

[1] A. I. Xun Ai and S. M. Pogwizd. Connexin 43 downregulation and dephosphorylation in nonischemic heart failure is associated with enhanced colocalized protein phosphatase type 2a. *Circulation Research*, 96:54–63, 2005.

[2] R. R. Aliev and A. V. Panfilov. A simple two-variable model of cardiac excitation. *Chaos, Solutions and Fractals*, 7:293–301, 1996.

[3] R. G. Assomull, S. K. Prasad, E. Burman, M. Khan, M. Sheppard, P. A. Poole-Wilson, and D. J. Pennell. Cardiovascular magnetic resonance, fibrosis and prognosis in dilated cardiomyopathy. *Journal of the American College of Cardiology*, 48:1977–1985, 2006.

[4] I. Babuska and W.C. Rheinbolt. Error estimates for adaptive finite element computation. *SIAM Journal on Numerical Analysis*, 15, 1978.

[5] M. J. Baines, M. E. Hubbard, and P.K. Jimack. A moving mesh finite element algorithm for fluid flow problems with moving boundaries. *International Journal for Numerical Methods in Fluids*, 47:1077–1083, 2005.

[6] D. Barkley. A model for fast computer simulation of waves in excitable media. *Physica D*, 49:61–70, 1991.

[7] J. Bassingthwaighte, P. Hunter, and D. Noble. The cardiac physiome: perspectives for the future. *Experimental Physiology*, 94:597605, 2009.

[8] G. W. Beeler and H. Reuter. Reconstruction of the action potential of ventricular myocardial fibres. *The Journal of Physiology*, 268:177–210, 1977.

[9] D. M. Bers. Review article: Cardiac excitation contraction coupling. *Nature*, 415:198–2005, 2002.

[10] D. M. Bers. Altered cardiac myocyte ca regulation in heart failure. *Physiology*, 21:380–387, 2006.

[11] D. J. Beuckelmann, M Nabauer, and R. Erdmann. Alterations of K^+ current in isolated human ventricle myocytes from patients with terminal heart failure. *Circulation Research*, 73:379–385, 1993.

[12] A. Brandt. Multi-level adaptive solutions to boundary-value problems. *Mathematics of Computation*, 31:333–390, 1977.

[13] M. O. Bristeau, R. Glowinski, L. Dutto, J. Priaux, and G. Rog. Compressible viscous flow calculations using compatible finite element approximations. *International Journal for Numerical Methods in Fluids*, 11:719–749, 1990.

[14] S. G. Campbell, E. Howard, J. Aguado-Sierra, B. A. Coppola, J. H. Omens, L. J. Mulligan, A. D. McCulloch, and R. C. Kerckhoffs. Effect of transmurally heterogeneous myocyte excitation-contraction coupling on canine left ventricular electromechanics. *Experimental Physiology*, 94(5):541–552, 2009.

[15] P. J. Capon. *Adaptive Stable Finite Element Methods for the Compressible Navier Stokes Equations*. PhD thesis, University of Leeds, 1995.

[16] R. Y. Chang and C. H. Hsu. A variable-order spectral element method for incompressible viscous flow simulation. *International Journal for Numerical Methods in Engineering*, 39:2865–2887, 1996.

[17] E. M. Cherry and F. H. Fenton. Suppression of alternans and conduction blocks despite steep apd restitution: electrotonic, memory, and conduction velocity restitution effects. *American Journal of Physiology - Heart and Circulatory Physiology*, 286:2332–2341, 2004.

[18] E. M. Cherry, H. S. Greenside, and C. S. Henriquez. A space-time adaptive method for simulating complex cardiac dynamics. *Physical Review Letters*, 84:1343–1346, 2000.

[19] E. M. Cherry, H. S. Greenside, and C. S. Henriquez. Efficient simulation of three-dimensional anisotropic cardiac tissue using an adaptive mesh refinement method. *Chaos*, 13:853–865, 2003.

[20] C. Cherubini, S. Filippi, P. Nardinocchi, and L. Teresi. An electromechanical model of cardiac tissue: Constitutive issues and electrophysiological effects. *Progress in Biophysics and Molecular Biology*, 97:562573, 2008.

[21] R. H. Clayton and A. V. Panfilov. A guide to modelling cardiac electrical activity in anatomically detailed ventricles. *Progress in Biophysics and Molecular Biology*, 96:19–43, 2008.

[22] P. Coorevits and E. Bellenger. Alternative mesh optimality criteria for h-adaptive finite element method. *Finite Elements in Analysis and Design*, 40:1195–1215, 2004.

[23] G. R Cowper. Gaussian quadrature formulae for triangles. *International Journal for Numerical Methods in Engineering*, 7:405–408, 1973.

[24] E. Cuthill and J. McKee. Reducing the bandwidth of sparse symmetric matrices. *Proceedings of the 24th National Conference of the ACM*, 1969.

[25] R. Dash, K. F. Frank, and A. N. Carr. Gender influences on sarcoplasmic reticulum $Ca2^{+}$ handling in failing human myocardium . *Journal of Molecular and Cellular Cardiology*, 33:1345–1353, 2001.

[26] D. DiFrancesco and D. Noble. A model of cardiac electrical activity incorporating ionic pumps and concentration changes. *Philosophical Transactions of the Royal Society B*, 307:353–398, 1985.

[27] E.Dupont, T. Matsushita, R. A. Kaba, C. Vozzi, S. R. Coppen, N. Khan, R. Kaprielian, M. H. Yacoub, and N. J. Severs. Altered connexin expression in human congestive heart failure. *Journal of Molecular and Cellular Cardiology*, 33(2):359–371, 2001.

[28] T. H. Everett and J. E. Olgin. Atrial fibrosis and the mechanisms of atrial fibrillation. *Heart Rhythm*, 4(3 Suppl):S24–7, 2007.

[29] F. Fenton and A. Karma. Vortex dynamics in three-dimensional continuous myocardium with fiber rotation: Filament instability and fibrillation. *Chaos*, 8:20–47, 1998.

[30] F. H. Fenton and E. M. Cherry. Models of cardiac cell. *Scholarpedia*, 3, 2008.

[31] M. Fink, S.A. Niederer, E.M. Cherry, F.H. Fenton, J.T. Koivumki, G. Seemann, R. Thul, H. Zhang, F.B. Sachse, D. Beard, E.J. Crampin, and N.P. Smith. Cardiac cell modelling: Observations from the heart of the cardiac physiome project. *Progress in Biophysics and Molecular Biology*, 104(1-3):2–21, 2011.

[32] J. Fish and T. Belytschko. *A First Course in Finite Elements*. Wiley, 2007.

[33] R. A. FitzHugh. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical Journal*, 1(6):445–466, 1961.

[34] P. Colli Franzone, L. F. Pavarino, and B. Taccardi. Effects of transmural electrical heterogeneities and electrotonic interactions on the dispersion of cardiac repolarization and action potential duration: a simulation study. *Mathematical Biosciences*, 204:132–165, 2006.

[35] A. George and J. W. H. Liu. *Computer solution of large sparse positive definite systems. Prentice-Hall series in computational mathematics.* Prentice-Hall, 1981.

[36] L . Glass, P. J. Hunter, and A. D. McCulloch, editors. *Theory of Heart: Biomechanics, Biophysics, and Nonlinear Dynamics of Cardiac Function.* Springer, New York, 1991.

[37] A. V. Glukov, V. V. Fedorov, Q. Lou, V. K. Ravikumar, P. W. Kalish, R. B. Schuessler, N. Moazami, and R. Efimiov. Transmural dispersion of repolarization in failing and nonfailing human ventricle. *Circulation Research*, 106:981–991, 2010.

[38] C. Goodyer, J. Hodrien, J. Wood, P. Kohl, and K. Brodlie. Using high-resolution displays for high-resolution cardiac data. *Philosophical Transactions of the Royal Society A*, 367:2667–2677, 2009.

[39] A. E. Green and J. E. Adkins. *Large Elastic Deformations*. Clarendon Press, Oxford., 1970 (2nd Edition).

[40] P. M. Gresho and R. L. Sani. *Incompressible Flow and the Finite Element Method, Volume 2, Isothermal Laminar Flow*. Wiley, 2000.

[41] J. M. Guccione, A. D. McCulloch, and L. K. Waldman. Passive material properties of intact ventricular myocardium determined from a cylindrical model. *Journal of Biomechanical Engineering*, 113:42–55, 1991.

[42] R. M. Gulrajani. *Bioelectricity and Biomagnetism. Wiley, New York.* Wiley, 1998.

[43] I. Gussak, C. Antzelevitch, S. C. Hammill, W. K. Shen, and P. Bjerregaard, editors. *Cardiac Repolarization: Bridging Basic and Clinical Science*. Humana Press, 2003.

[44] G. Hasenfuss, S. Wolfgang, S. E. Lehnart, M. Preuss, B. Pieske, L. S. Maier, J. Prestle, K. Minami, and H. Just. Relationship between $Na^{-}Ca^{-}$ exchanger protein levels and diastolic function of failing human myocardium. *Circulation*, 99:641–648, 1999.

[45] A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology (London)*, 117:500–544, 1952.

[46] W. Huang and R. D Russell. A high dimensional moving mesh strategy. *Applied Numerical Mathematics*, 26:63–76, 1998.

[47] P. J. Hunter, A. D. McCulloch, and H. E. ter Keurs. Modelling the mechanical properties of cardiac muscle. *Progress Biophysics and Molelcular Biology*, 69, 1998.

[48] P. J. Hunter and B. H. Smaill. The analysis of cardiac function: a continuum approach. *Progress Biophysics and Molelcular Biology*, 52:101–164, 1988.

[49] M. T. Jiang, A. J. Lokuta, E. F. Farrell, M. R. Wolff, R A. Haworth, and H. Valdivia. Abnormal $Ca2^{+}$ release, but normal ryanodine receptors, in canine and human heart failure. *Circulation Research*, 91:1015–1022, 2002.

[50] A. C. Jones. *A projected multigrid method for the solution of nonlinear finte elements problems on adaptively refined grids*. PhD thesis, School of Computing, University of Leeds, 2005.

[51] H. J. Jongsma and R. Wilders. Gap junctions in cardiovascular disease. *Circulation Research*, 86:1193–1197, 2000.

[52] S. Kaab, J. Dixon, and J. Duc. Molecular basis of transient outward potassium current downregulation in human heart failure. *Circulation*, 98:1383–1393, 1998.

[53] J. Keener and J. Sneyd. *Mathematical Physiology. 1998.* New York: Springer-Verlag, 1998.

[54] R. H. Keldermann, M. P. Nash, H. Gelderblom, V. Y. Wang, and A. V. Panfilov. Electromechanical wavebreak in a model of the human left ventricle. *American Journal of Physiology - Heart and Circulatory Physiology*, 299:H134–H143, 2010.

[55] R. C. P. Kerckhoffs, S. N. Healy, T. P. Usyk, and A. D. McCulloch. Computational methods for cardiac electromechanics. *Proceedings of the IEEE*, 94, No 4, 2006.

[56] A. M. Khokhlov. Fully threaded tree algorithms for adaptive refinement fluid dynamics simulations. *Journal of Computational Physics*, 143:519–543, 1998.

[57] N. Kirk, A. P. Benson, C. E. Goodyer, and M. E. Hubbard. Application of an efficient coupled electromechanical solver to investigate end stage human heart failure. *Computing in Cardiology 2011*, 38:17–20, 2011.

[58] S. Land, S. A. Niederer, and N. P. Smith. Efficient computational methods for strongly coupled cardiac electromechanics. *IEEE Transactions on Biomedical Engineering*, 2011.

[59] W. Li, P. Kohl, and N. Trayanova. Induction of ventricular arrhythmias following mechanical impact: a simulation study in 3d. *Journal of Molecular Histology*, 35:679e86., 2004.

[60] S. Linge, G. T. Lines, and J. Sundnes. Solving the heart mechanics equations with Newton and quasi Newton methods - a comparison. *Computer Methods in Biomechanics and Biomedical Engineering*, 8(1):1–8, 2005.

[61] R. Lohner. An adaptive finite element scheme for transient problems in cfd. *Computer Methods in Applied Mechanics and Engineering*, 61:323–338, 1987.

[62] C. H. Luo and Y. Rudy. A dynamic model of the cardiac ventricular action potential: I. simulations of ionic currents and concentration changes. *Circulation Research*, 74 (6):1071–1096, 1994.

[63] C.H. Luo and Y. Rudy. A model of the ventricular cardiac action potential. depolarization, repolarization, and their interaction. *Circulation Research*, 68:1501–1526, 1991.

[64] L. E. Malvern. *Introduction to the Mechanics of a Continuous Medium*. Prentice-Hall, Inc, 1969.

[65] A. D. McCulloch, B. H. Smaill, and P. J. Hunter. Left ventricular epicardial deformation in isolated arrested dog heart. *American Journal of Physiology*, 252(1 Pt 2):H233–41, 1987.

[66] M. Mooney. A theory of large elastic deformation. *Journal of Applied Physics*, 11(9):582–592, 1940.

[67] M. P. Nash, C. P. Bradley, P. M Sutton, R. H. Clayton, P. Kallis, M. Hayward, D. J. Paterson, and P. Taggart. Whole heart apd restitution properties in cardiac patients: A combined clinical and modeling study. *Experimental Physiology*, g1:33g35g, 2006.

[68] M. P. Nash and P. J. Hunter. Computational mechanics of the heart. *Journal of Elasticity*, 61:29, 2000.

[69] M. P. Nash and A. V. Panfilov. Electromechanical model of excitable tissue to study reentrant cardiac arrhythmias. *Progress Biophysics and Molelcular Biology*, 85:501–522, 2004.

[70] D. Nickerson and M. Buist. Practical application of CellML 1.1: The integration of new mechanisms into a human ventricular myocyte model. *Progress in Biophysics and Molecular Biology*, 98:38–51, 2008.

[71] D. Nickerson, N. Smith, and P. Hunter. New developments in a strongly coupled cardiac electromechanical model. *The European Society of Cardiology*, 2005.

[72] S. A. Niederer and N. P. Smith. An improved numerical method for strong coupling of excitation and contraction models in the heart. *Progress Biophysics and Molelcular Biology*, 96:90–111, 2008.

[73] S.A. Niederer, P.J. Hunter, and N.P. Smith. A quantitative analysis of cardiac myocyte relaxation: A simulation study. *Biophysical Journal*, Volume 90:16971722, 2006.

[74] D. Noble. A modification of the Hodgkin Huxley equations applicable to Purkinje fibre action and pace-maker potentials. *Journal of Physiology*, 160:317352, 1962.

[75] D. Noble, A. Varghese, P. Kohl, and P. Noble. Improved guinea-pig ventricular cell model incorporating a diadic space, I_{Kr} and I_{Ks}, and length and tension-dependent processes. *The Canadian Journal of Cardiology*, 14(1):123–34., 1998.

[76] D.A. Nordsletten, S.A. Niederer, M.P. Nash, P.J. Hunter, and N.P. Smith. Coupling multi-physics models to cardiac mechanics. *Progress in Biophysics and Molecular Biology,*, 104 (1-3):77–88, 2011.

[77] J. T. Oden and M. Ainsworth. *A posterior error estimation in finite element analysis*. John Wiley and Sons, 2000.

[78] J. T. Oden, I. Babuska, and C.E Baumann. A discontinuous hp finite element method for diffusion problems. *Journal of Computational Physics*, 146:491519, 1998.

[79] J. T. Oden, T. Strouboulis, and Ph. Devloo. Adaptive finite element methods for high-speed compressible flows. *International Journal for Numerical Methods in Fluids*, 7:1211–1228, 1987.

[80] J. M. Ortega and W. C. Rheinboldt. *Iterative solution of nonlinear equations in several variables*. Academic Press, 1970.

[81] A. V. Panfilov and A. V. Holden, editors. *Computational Biology of the Heart*. John Wiley & Sons, 1997.

[82] P. Pathmanathan and J. P. Whiteley. A numerical method for cardiac mechanoelectric simulations. *Annals of Biomedical Engineering*, 37:860–873, 2009.

[83] M. Pennacchio. The mortar finite element method for the cardiac "bidomain" model of extracellular potential. *Journal of Scientific Computing*, 20:191–210, 2004.

[84] A. M. Pertsov, J. M. Davidenko, R. Salomonsz, W. T. Baxter, and J. Jalife. Spiral waves of excitation underlie reentrant activity in isolated cardiac muscle. *Circulation Research*, Vol 72:631–650, 1993.

[85] M. Potse, B. Dub, J. Richer, A. Vinet, and R. M. Gulrajani. A comparison of monodomain and bidomain reaction-diffusion models for action potential propagation in the human heart. *IEEE Transactions on Biomedical Engineering*, 53:2425–2435, 2006.

[86] W. H. Press, B. P. Flannery, S. A Teukolsky, and W. T Vetterling. *Numerical Recipes in C. The Art of Scientific Computing*. Cambridge University Press, 1992.

[87] L. Priebe and D. J. Beuckelmann. Simulation study of cellular electric properties in heart failure. *Circulation Research*, 82:1206–1223, 1998.

[88] Z. L. Qu and A. Garfinkel. An advanced algorithm for solving partial differential equation in cardiac conduction. *IEEE Transactions on Bio-Medical Engineering*, 46:1166–1168, 1999.

[89] J. N. Reddy. *An introduction to the finite element method*. McGraw-Hill International Editions, 1993.

[90] W. C. Rheinboldt and C. K. Mesztenyi. On a data structure for adaptive finite element mesh refinements. *ACM Transactions on Mathematical Software*, 6:166–187, 1980.

[91] R. S. Rivlin. Large elastic deformations of isotropic materials. iv. further developments of the general theory. *Philosophical Transactions of the Royal Society A*, 241(835):379–397., 1948.

[92] M. A. Rossi. Connective tissue skeleton in the normal left ventricle and in hypertensive left ventricular hypertrophy and chronic chagasic myocarditis. *Medical Science Monitor*, 7(4):820–32, 2001.

[93] B. J. Roth. Art Winfree and the bidomain model of cardiac tissue. *Journal of Theoretical Biology*, 230:445–449, 2004.

[94] S. Rush and H. Larsen. A practical algorithm for solving dynamic membrane equations. 25, 389392. *IEEE Transactions on Biomedical Engi*, 25:389–392, 1978.

[95] Y. Saad. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, 1996.

[96] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7:856–869, 1986.

[97] P. Scarborough, P. Bhatnagar, K. Wickramasinghe, K. Smolina, C. Mitchell, and M. Rayner. *Coronary heart disease statistics 2010 edition*. British Heart Foundation Health Promotion Research Group, 2010.

[98] R.H.G Schwinger, H. Bundgaard, J. Mller-Ehmsen, and K. Kjeldsen. The Na, $K-ATPase$ in the failing human heart. *Cardiovascular Research*, 57:913–920, 2003.

[99] J. R. Shewchuk. Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry: Theory and Applications*, 22(1-3):21–74, 2002.

[100] Jonathan Richard Shewchuk. Triangle: Engineering a 2d quality mesh generator and Delaunay triangulator. *Lecture Notes in Computer Science*, 1148:203–222, 1996.

[101] S. Sicouri and C. Antzelevitch. A subpopulation of cells with unique electrophysiological properties in the deep subepicardium of the canine ventricle. the m cell. *Circulation Research*, 68(6):1729–41, 1991.

[102] B.H. Smaill and P.J. Hunter. *Structure and function of the diastolic heart: material properties of passive myocardium. Theory of Heart: Biomechanics, Biophysics, and Nonlinear Dynamics of Cardiac Function*. Springer, 1991.

[103] G. D. Smith. *Numerical solution of partial differential equations : finite difference methods*. Oxford : Clarendon, 1985.

[104] N. P. Smith, M. L. Buist, and A. J. Pullan. Altered T wave dynamics in a contracting cardiac model. *Journal of Cardiovascular Electrophysiology*, 14:S203e9, 2003.

[105] M.S. Spach, J. F. Heidlage, and P.C. Dolber. Mechanism of origin of conduction disturbances in aging human atrial bundles: Experimental and model study. *Heart Rhythm*, 4:175–185, 2007.

[106] W. Speares and M. Berzins. A 3d unstructured mesh adaption algorithm for time-dependent shock-dominated problems. *International Journal for Numerical Methods in Fluids*, 25:81–104, 1997.

[107] Q.F. Stout, D. L. De Zeeuw, T. I. Gombosi, C. P. T. Groth, H. G. Marshall, and K. G. Powell. Adaptive blocks: A high performance data structure. *Tech. Report T, University of Michigan*, 1997.

[108] K. H. W. J. ten Tusscher, D. Noble, P. J. Noble, and A. V. Panfilov. A model for human ventricular tissue. *American Journal of Physiology - Heart and Circulatory Physiology*, 286:H1573–H1589., 2004.

[109] K. H. W. J. ten Tusscher and A. V. Panfilov. Influence of diffuse fibrosis on wave propagation in human ventricular tissue. *Europace*, 9:vi38–vi45, 2007.

[110] K. H. W. J. ten Tusscher and A.V. Panfilov. Alternans and spiral breakup in a human ventricular tissue model. *American Journal of Physiology - Heart and Circulatory Physiology*, 291:H1088–H1100, 2006.

[111] J. W. Thomas. *Numerical Partial Differential Equations: Finite Difference Methods. Texts in Applied Mathematics*. New York: Springer-Verlag, 1995.

[112] S. P. Thomas, J. P. Kucera, L. Bircher-Lehmann, Y. Rudy, J. E. Saffitz, and A. G. Klber. Impulse propagation in synthetic strands of neonatal cardiac myocytes with genetically reduced levels of connexin43. *Circulation Research*, 92:1209–1216, 2003.

[113] G. F. Tomaselli and D. P. Zipes. What causes sudden death in heart failure? *Circulation Research*, 95:754–763, 2004.

[114] J. A. Trangenstein and C. Kim. Operator splitting and adaptive mesh refinement for the Luo-Rudy model. *Journal of Computational Physics*, 194:645679, 2004.

[115] L. Tung. *A bi-domain model for describing ischemic myocardial d-c potentials*. PhD thesis, MIT, Cambridge, Mass., 1978.

[116] R. Verfurth. A posteriori error estimation and adaptive mesh refinement techniques. *Journal of Computational and Applied Mathematics*, 50:6783, 1994.

[117] F. J. Vetter and A. D. McCulloch. Three-dimensional analysis of regional cardiac function: A model of rabbit ventricular anatomy. *Progress Biophysics and Molelcular Biology*, 69:157–183, 1998.

[118] E. J. Vigmond, R. Weber dos Santos, A. J. Prassl, M. Deo, and G. Plank. Solvers for the cardiac bidomain equations. *Progress Biophysics and Molelcular Biology*, 96(1-3):3–18, 2008.

[119] R. A. Walters and E. J. Barragy. Comparison of h and p finite element approximations of the shallow water equations. *International Journal for Numerical Methods in Fluids*, 24:61–79, 1997.

[120] J. P. Whiteley, M. J. Bishop, and D. J. Gavaghan. Soft tissue modelling of cardiac fibres for use in coupled mechano-electric simulations. *Bulletin of Mathematical Biology*, 69:2199–2225, 2007.

[121] F. C. Yin. Applications of the finite-element method to ventricular mechanics. *Critical Reviews Biomedical Engineering*, 12(4):311–42, 1985.

[122] W. Ying. *A Multilevel adaptive approach for computational cardialogy*. PhD thesis, Duke University, 2005.

[123] W. Ying and C. S. Henriquez. Adaptive mesh refinement for modeling cardiac electrical dynamics. *submitted to Applied Numerical Mathematics*.

[124] W. Ying, D.J. Rose, and C.S. Henriquez. Adaptive mesh refinement and adaptive time integration for electrical wave propagation on the Purkinje system. *submitted to SIAM Journal on Scientific Computing*.

[125] O. C. Zienkiewicz and K Morgan. *Finite Elements and Approximation*. Wiley, 1983.

[126] O. C. Zienkiewicz and R. L. Taylor. *The finite element method. Vol.1, The basis*. Oxford : Butterworth-Heinemann, 2000.

[127] D. Zipes and J. Jalife. *Cardiac Electrophysiology*. Elsevier, 2000.

[128] Zoofari. From the Wikipedia Commons.

Appendix A

Preconditioning within Kinsol

Undertaking preconditioning with KINSOL is a complex process and an overview of the process is provided in A.1.

Process A.1 Preconditioning within Kinsol

1: Call a KINSOL controlling function and pass the initial guess for the outer solve (from the extrapolation of the previous two solves).
2: Undertake memory setup with definitions now including sparse format structure needed for SPARSKIT
3: Inform KINSOL of the function being used to build the function vector $F(x_n)$
4: Define the parameters of the solver
5: Define the KINSOL solve component required, in our case *KINSpgmr*
6: Inform KINSOL of the two functions being used for preconditioning. One is the setup function and the other is the inner solve function.
7: The setup function builds a numerical Jacobian from $F(x_n)$.
8: The solve function calls SPARKSIT to setup the ILUT preconditioning and then SPARSKIT GMRES solve the inner system
9: Finally the outer solve takes place with KINSOL

Appendix B

Computer Equipment

During the course of the thesis various computing equipment was used and the specification of these is described below. Please note that ARC1 is a high performance computer with the ability to undertake either serial or parallel processes.

Item	Specification
Processor	Intel Core 2 CPU 4300
Clock Speed / Memory	1.8GHz / 2.0 Gigabytes
Operating System	Fedora Release 11 (Leonidas 64 bit)

Table B.1: *Basic PC Specification*

Item	Specification
Processor	Intel Quad-core Xeon E5420
Clock Speed / Memory	2.5GHz / 4 Gigabytes
Operating System	Fedora Release 11 (Leonidas 64 bit)

Table B.2: *Workstation PC specification*

Item	Specification
Processor	Intel Quad-core X5560
Clock Speed / Memory	2.8GHz / 12.0 Gigabytes of DDR3 1333MHz memory
Operating System	CentOS (5.4) 64-bit

Table B.3: *Arc1 Specification: Serial jobs*

Appendix C

Ten Tusscher-Panfilov model

C.1 Ten Tusscher-Panfilov human ventricular cell model

For completeness, the TP06 model [110] of cardiac electrophysiology is reproduced below. The equations below include the TP06 model [110] equations where mandated and include the remaining equations from the 2004 model [108].

Reversal Potentials

$$E_X = \frac{RT}{zF} log \frac{X_o}{X_i} \quad for \quad X = Na^+, K^+, Ca^{2+} \tag{C.1}$$

$$E_{Ks} = \frac{RT}{F} log \frac{K_o + p_{KNa} Na_o}{K_i + p_{KNa} Na_i} \tag{C.2}$$

Membrane Currents

$$I_{Na} = G_{Na} m^3 hj (V - E_{Na}) \tag{C.3}$$

$$m_\infty = \frac{1}{[1 + e^{(-56.86 - V)/9.03}]^2} \tag{C.4}$$

$$\alpha_m = \frac{1}{1 + e^{(-60 - V)/5}} \tag{C.5}$$

$$\beta_m = \frac{0.1}{1 + e^{(V+35)/5}} + \frac{0.1}{1 + e^{(V-50)/200}} \tag{C.6}$$

$$\tau_m = \alpha_m \beta_m \tag{C.7}$$

$$h_\infty = \frac{1}{[1+e^{(V+71.55)/7.43}]^2} \tag{C.8}$$

$$\alpha_h = 0 \quad if \quad V \geq -40 \tag{C.9}$$

$$\alpha_h = 0.057 e^{-(V+80)/6.8} \quad otherwise \tag{C.10}$$

$$\beta_h = \frac{0.77}{0.13[1+e^{-(V+10.66)/11.1}]} \quad if \quad V \geq -40 \tag{C.11}$$

$$\beta_h = 2.7 e^{0.079V} + 3.1 \times 10^5 e^{0.3485V} \quad otherwise \tag{C.12}$$

$$\tau_h = \frac{1}{\alpha_h + \beta_h} \tag{C.13}$$

$$j_\infty = \frac{1}{[1+e^{(V+71.55)/7.43}]^2} \tag{C.14}$$

$$\alpha_j = 0 \quad if \quad V \geq -40 \tag{C.15}$$

$$\alpha_j = \frac{(-2.5428 \times 10^4 e^{0.2444V} - 6.948 \times 10^{-6} e^{-0.04391V})(V+37.78)}{1+e^{0.311(V+79.23)}} \quad otherwise$$

$$\beta_j = \frac{0.6 e^{0.057V}}{1+e^{-0.1(V+32)}} \quad if \quad V \geq -40 \tag{C.16}$$

$$\beta_j = \frac{0.02424 e^{-0.01052V}}{1+e^{-0.1378(V+40.14)}} \quad otherwise$$

$$\tau_j = \frac{1}{\alpha_j + \beta_j} \tag{C.17}$$

Transient Outward Current

$$I_{to} = G_{to} rs (V - E_k) \tag{C.18}$$

For all cell types

$$r_\infty = \frac{1}{1+e^{(20-V)/6}} \tag{C.19}$$

$$\tau_r = 9.5e^{-(V+40)^2/1800} + 0.8 \tag{C.20}$$

For epicardial and M cells

$$s_\infty = \frac{1}{1+e^{(V+20)/5}} \tag{C.21}$$

$$\tau_s = 85e^{-(V+45)^2/320} + \frac{5}{1+e^{(V-20)/5}} + 3 \tag{C.22}$$

For endocardial cells

$$s_\infty = \frac{1}{1+e^{(V+28)/5}} \tag{C.23}$$

$$\tau_s = 1,000e^{-(V+67)^2/1,000} + 8 \tag{C.24}$$

Inward Rectifier K^+ Current

$$I_{K1} = G_{K1}\sqrt{\frac{K_o}{5.4}}x_{K1\infty}(V - E_k) \tag{C.25}$$

$$\alpha_{K1} = \frac{0.1}{1+e^{0.06(V-E_K-200)}} \tag{C.26}$$

$$\beta_{K1} = \frac{3e^{0.0002(V-E_K+100)} + e^{0.1(V-E_K-10)}}{1+e^{-0.5(V-E_K)}} \tag{C.27}$$

$$x_{K1\infty} = \frac{\alpha_{K1}}{\alpha_{K1}+\beta_{K1}} \tag{C.28}$$

Na^+ / Ca^{2+} Exchanger Current

$$I_{NaCa} = k_{NaCa}\frac{e^{\gamma VF/RT}Na_i^3Ca_o - e^{(\gamma-1)VF/RT}Na_o^3Ca_i\alpha}{(K_{mNai}^3+Na_o^3)(K_{mCa}+Ca_o)(1+k_{sat}e^{(\gamma-1)VF/RT})} \tag{C.29}$$

Na^+ / K^+ Pump Current

$$I_{NaK} = P_{NaK}\frac{K_o Na_i}{(K_o + K_{mK})(Na_i + K_{mNa})(1 + 0.1245e^{-0.1VF/RT} + 0.0353e^{-VF/RT})} \tag{C.30}$$

$$I_{pCa} = G_{pCa}\frac{Ca_i}{K_{pCa} + Ca_i} \tag{C.31}$$

$$I_{pK} = G_{pK}\frac{V - E_K}{1 + e^{(25-V)/5.98}} \tag{C.32}$$

Background Currents

$$I_{bNa} = G_{bNa}(V - E_{Na}) \tag{C.33}$$

$$I_{bCa} = G_{bCa}(V - E_{Ca}) \tag{C.34}$$

L-Type Ca^{2+} Current

$$I_{CaL} = G_{CaL} d f f_2 f_{cass} 4\frac{(V-15)F^2}{RT}\frac{0.25Ca_{SS}e^{2(V-15)F/RT} - Ca_o}{e^{2(V-15)F/RT} - 1} \tag{C.35}$$

$$d_\infty = \frac{1}{1 + e^{(-8-V)/7.5}} \tag{C.36}$$

$$\alpha_d = \frac{1.4}{1 + e^{(-35-V)/13}} + 0.25 \tag{C.37}$$

$$\beta_d = \frac{1.4}{1 + e^{(V+5)/5}} \tag{C.38}$$

$$\gamma_d = \frac{1}{1 + e^{(50-V)/20}} \tag{C.39}$$

$$\tau_d = \alpha_d \beta_d + \gamma_d \tag{C.40}$$

$$f_\infty = \frac{1}{1 + e^{(V+20)/7}} \tag{C.41}$$

$$\tau_f = 1125e^{-(V+27)^2/240} + \frac{165}{1 + e^{(25-V)/10}} + 80 \tag{C.42}$$

$$\alpha_{fca} = \frac{1}{1+(Ca_i/0.000325)^8} \tag{C.43}$$

$$\beta_{fca} = \frac{0.1}{1+e^{(Ca_i-0.0005)/0.0001}} \tag{C.44}$$

$$\gamma_{fca} = \frac{0.2}{1+e^{(Ca_i-0.00075)/0.0008}} \tag{C.45}$$

$$f_{ca\infty} = \frac{\alpha_{fca}+\beta_{fca}+\gamma_{fca}+0.23}{1.46} \tag{C.46}$$

$$\tau_{fca} = 2ms \tag{C.47}$$

$$\frac{\mathrm{d}f_{ca}}{\mathrm{d}t} = k\frac{f_{ca\infty}-f_{ca}}{\tau_{fca}} \tag{C.48}$$

$$k = 0 \quad if \quad f_{ca\infty} > f_{ca} \quad and \quad V > -60mV \tag{C.49}$$

$$k = 1 \quad otherwise$$

$$\alpha_f = 1102.5e^{-\left(\frac{V+27}{15}\right)^2} \tag{C.50}$$

$$\beta_f = \frac{200}{1+e^{(13-V)/10}} \tag{C.51}$$

$$\gamma_f = \frac{180}{1+e^{(V+30)/10}} + 20 \tag{C.52}$$

$$\tau_f = \alpha_f + \beta_f + \gamma_f \tag{C.53}$$

$$f_{2\infty} = \frac{0.67}{1+e^{(V+35)/7}} + 0.33 \tag{C.54}$$

$$\alpha_{f2} = 600e^{-\frac{(V+25)^2}{170}} \tag{C.55}$$

$$\beta_{f2} = \frac{31}{1+e^{(25-V)/10}} \tag{C.56}$$

$$\gamma_{f2} = \frac{16}{1+e^{(V+30)/10}} \tag{C.57}$$

$$\tau_{f2} = \alpha_{f2} + \beta_{f2} + \gamma_{f2} \tag{C.58}$$

$$f_{cass\infty} = \frac{0.6}{1+\left(\frac{Ca_{ss}}{0.05}\right)^2} + 0.4 \tag{C.59}$$

$$\tau_{fcass} = \frac{80}{1+\left(\frac{Ca_{ss}}{0.05}\right)^2} + 2 \tag{C.60}$$

Slow Delayed Rectifier Current

$$I_{Ks} = G_{Ks} x_s^2 (V - E_{Ks}) \tag{C.61}$$

$$x_{s\infty} = \frac{1}{1+e^{(-5-V)/14}} \tag{C.62}$$

$$\alpha_{xs} = \frac{1400}{\sqrt{1+e^{(5-V)/6}}} \tag{C.63}$$

$$\beta_{xs} = \frac{1}{1+e^{(V-35)/15}} \tag{C.64}$$

$$\tau_{xs} = \alpha_{xs}\beta_{xs} + 80 \tag{C.65}$$

Calcium Dynamics

$$I_{leak} = V_{leak}(Ca_{SR} - Ca_i) \tag{C.66}$$

$$I_{up} = \frac{V_{maxup}}{1+K_{up}^2/Ca_i^2} \tag{C.67}$$

$$I_{re1} = V_{re1} O (Ca_{SR} - Ca_{SS}) \tag{C.68}$$

$$I_{rel} = \left(a_{rel}\frac{Ca_{sr}^2}{b^2 rel + Ca_{sr}^2} + c_{rel}\right) dg \tag{C.69}$$

$$g_\infty = \frac{1}{1+Ca_i^6/0.00035^6} \quad if \quad Ca_i \leq 0.00035 \tag{C.70}$$

$$g_\infty = \frac{1}{1+Ca_i^{16}/0.00035^{16}} \quad otherwise$$

$$\tau_g = 2ms \tag{C.71}$$

$$\frac{\mathrm{d}g}{\mathrm{d}t} = k\frac{g_\infty - g}{\tau_g} \tag{C.72}$$

$$k = 0 \quad if \quad g_\infty > g \quad and \quad V > -60mV \tag{C.73}$$

$$k = 1 \quad otherwise$$

$$I_{xfer} = V_{xfer}(Ca_{SS} - Ca_i) \tag{C.74}$$

$$O = \frac{k_1 Ca_{SS}^2 \bar{R}}{k_3 + k_1 Ca_{SS}^2} \tag{C.75}$$

$$\frac{\mathrm{d}\bar{R}}{\mathrm{d}t} = -k_2 Ca_{SS}\bar{R} + k_4(1-\bar{R}) \tag{C.76}$$

$$k_1 = \frac{k_{1'}}{k_{casr}} \tag{C.77}$$

$$k_2 = k_{2'}k_{casr} \tag{C.78}$$

$$k_{casr} = max_{sr} - \frac{max_{sr} - min_{sr}}{1 + (EC/Ca_{SR})^2} \tag{C.79}$$

$$Ca_{ibufc} = \frac{Ca_i \times Buf_c}{Ca_i + K_{bufc}} \tag{C.80}$$

$$\mathrm{d}Ca_{itotal}/\mathrm{d}t = -\frac{I_{bCa} + I_{pCa} - 2I_{NaCa}}{2V_c F} + \frac{V_{sr}}{V_c}(I_{1eak} - I_{up}) + I_{xfer} \tag{C.81}$$

$$Ca_{srbufsr} = \frac{Ca_{sr} \times Buf_{sr}}{Ca_{sr} + K_{bufsr}} \tag{C.82}$$

$$\mathrm{d}Ca_{SRtotal}/\mathrm{d}t = (I_{up} - I_{1eak} - I_{re1}) \tag{C.83}$$

$$Ca_{ssbufss} = \frac{Ca_{ss} \times Buf_{ss}}{Ca_{ss} + K_{bufss}} \tag{C.84}$$

$$\mathrm{d}Ca_{SStotal}/\mathrm{d}t = -\frac{I_{CaL}}{2V_{SS}F} + \frac{V_{sr}}{V_{ss}}I_{re1} - \frac{V_c}{V_{ss}}I_{xfer} \tag{C.85}$$

Rapid Delayed Rectifier Current

$$I_{Kr} = G_{Kr}\sqrt{\frac{K_o}{5.4}}x_{r1}x_{r2}(V - E_k) \tag{C.86}$$

$$x_{r1\infty} = \frac{1}{1 + e^{(-26-V)/7}} \tag{C.87}$$

$$\alpha_{xr1} = \frac{450}{1 + e^{(-45-V)/10}} \tag{C.88}$$

$$\beta_{xr1} = \frac{6}{1 + e^{(V+30)/11.5}} \tag{C.89}$$

$$\tau_{xr1} = \alpha_{xr1}\beta_{xr1} \tag{C.90}$$

$$x_{r2\infty} = \frac{1}{1 + e^{(V+88)/24}} \tag{C.91}$$

$$\alpha_{xr2} = \frac{3}{1 + e^{(-60-V)/20}} \tag{C.92}$$

$$\beta_{xr2} = \frac{1.12}{1 + e^{(V-60)/20}} \tag{C.93}$$

$$\tau_{xr2} = \alpha_{xr2}\beta_{xr2} \tag{C.94}$$

Sodium and Potassium Dynamics

$$\frac{\mathrm{d}\mathrm{Na}_i}{\mathrm{d}t} = -\frac{I_{Na} + I_{bNa} + 3I_{NaK} + 3I_{NaCa}}{V_c F} \tag{C.95}$$

$$\frac{\mathrm{dK}_i}{\mathrm{d}t} = -\frac{I_{K1} + I_{to} + I_{Kr} + I_{Ks} - 2I_{Nak} + I_{pK} + I_{stim} - I_{ax}}{V_c F} \tag{C.96}$$

C.2 TP06 Inital Parameters

Table C.1 contains the initial values of the state variables in the TP06 model. During the thesis two sets of initial values were used.

Variable	Stable range value	Original value	Description
cai	0.000104	0.00007	Initial intracellular calicum (mM)
casr	3.500442	1.3	Initial SR calcium (mM)
cass	0.000213	0.00007	Initial subspace calcium (mM)
nai	9.768751	7.67	Initial intracellular sodium (mM)
ki	135.750655	138.3	Initial intracellular potassium (mM)
m	0.001647	0.0	Initial m gate
h	0.750091	0.75	Initial h gate
j	0.749714	0.75	Initial j gate
d	0.000033	0.0	Initial d gate
f	0.977278	1.0	Initial f gate
f2	0.999502	1.0	Initial f2 gate
fca	0.999973	1.0	Initial fca gate
r	0.000000	0.0	Initial r gate
s	0.999998	1.0	Initial s gate
xr1	0.000206	0.0	Initial xr1 gate
xr2	0.473155	1.0	Initial xr2 gate
xs	0.003221	0.0	Initial xs gate
rbar	0.989218	1.0	rbar

Table C.1: *TP06 initial state variable values - original and steady state values*

Appendix D

Solving the cardiac mechanics

D.1 Introduction

Solving the cardiac mechanical system determines the deformed coordinate positions (x_1, x_2) of nodes within the mesh and the pressures (p) at these nodes. This involves a number of processes, including formulating the system of equations, approximating with the FEM and solving with the Newton iterative method.

The objective of this appendix is to provide a step-by-step walk-through of the processes involved in this and also further explain the use of isoparametric elements, within the FEM, to build the function vector needed for the Newton method.

D.2 Process walk-through

1. The governing equations for the deformed coordinates (x_1, x_2) and the pressures (p) are the stress equilibrium equation (see Equation (3.11)) and the incompressibility constraint (see Equation (3.37)).

2. For the FEM, the governing equations need converting into their weak forms. These are Equations (3.33) and (3.38).

3. The domain being modelled needs discretising into triangular elements. Each node (for example node i) of a triangular element has undeformed coordinates represented by (X_1^i, X_2^i).

4. The continuous variables for the deformed coordinates (x_1, x_2) are replaced with discrete approximations within the elements using basis functions. Quadratic Lagrange interpolation polynomials are used to approximate the deformed coordinates

(x_1^i, x_2^i) and these are given in Equations (D.4) to (D.9). Linear Lagrange basis functions are used to approximate the pressures (p^i), and these are given in Equation (D.31).

5. To provide flexibility in the solver, isoparametric elements are used. These map a given element (in the global coordinates) onto a local reference element where the FEM calculations are then undertaken. More information is given in Appendix D.3.

6. Because quadratic functions are used for the deformations, numerical integration is needed over the elements. This is undertaken using a 3-point quadrature rule, defined in Table 3.1.

7. The system is solved using the Newton iterative method (as described in Process 3.1), and for this a system in the form $\mathbf{f}(\mathbf{x}) = \mathbf{0}$, needs to be formulated, where $\mathbf{f}(\mathbf{x})$ is a system of n non-linear equations (these are described further in Equation (3.68)). This is calculated numerically using the FEM assembly loop.

8. For each iteration of the Newton method it is necessary to assemble $\mathbf{f}(\mathbf{x}^j)$, where j is the index of the Newton iterate. At the start of the method an initial estimate of $\mathbf{x}^j$ is provided (see Section 3.4.1 for more details), and the assembly will be repeated (within the Newton method process), until convergence is reached.

D.3 Isoparametric elements

D.3.1 Introduction to isoparametric elements

To solve the cardiac mechanics, it was necessary to use quadratic basis functions for the deformation unknowns. Isoparametric elements use a local reference coordinate system and implement a mapping process to specify the relationship to the original undeformed coordinate system. The local coordinate system is a simple element, for example a right angled triangle. This simplifies the implementation of the quadratic functions and also provides a more modular approach, whereby it is easier to substitute other higher order basis functions.

For isoparametric elements, shape functions are used to specify the relation between the global (X_1, X_2) and local (ξ, η) coordinate systems. Shape functions are defined for an idealized mapped element, as per Figure 3.2. The coordinate transformation is therefore:

$$X_1 = \sum_{i=1}^{6} \Psi_i \xi_i, \tag{D.1}$$

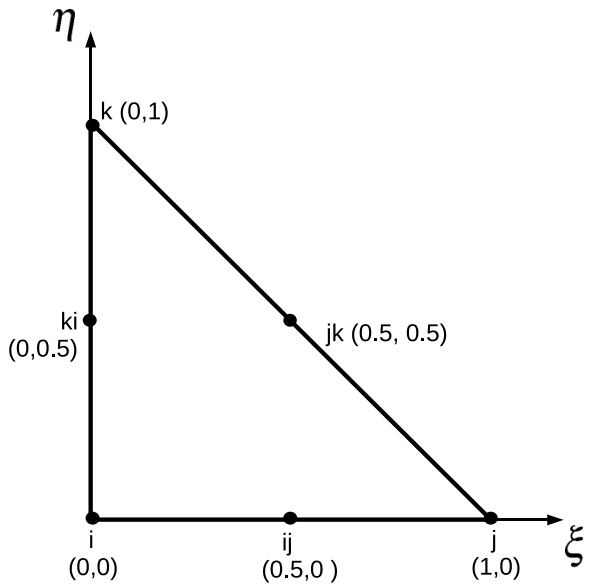


Figure D.1: *Reference (local) triangular element*

$$X_2 = \sum_{i=1}^{6} \Psi_i \eta_i, \tag{D.2}$$

where Ψ_i are the shape functions (given in Equations (D.4) to (D.9)) for the six-node triangular element.

For the FEM, the governing equations need converting into their weak form. These are Equations (3.33) and (3.38) and are a combination of a number of partial derivatives of the form:

$$\frac{\partial x_j}{\partial X_M}, \quad \frac{\partial \Psi_n}{\partial X_M}, \tag{D.3}$$

where $n = 1,2,3$ for linear approximations (and $n = 1..6$ for quadratic approximations), $M = 1,2$, $j = 1,2$. To solve the cardiac mechanical system using isoparametric elements, the partial derivatives in Equation (D.3) need to be represented in terms of the isoparametric coordinate system.

D.3.2 Functions for mapping local coordinates

For quadratic elements, a 6 node right-angled triangle is used for the local reference element (as shown in Figure D.1) and shape functions are required to map each node to the global coordinates. The functions used for the triangle in Figure D.1 are the Lagrange basis functions:

$$\Psi_i = 2(1-\xi-\eta)(\frac{1}{2}-\xi-\eta) \tag{D.4}$$

$$\Psi_{ij} = 4\xi(1-\xi-\eta), \tag{D.5}$$

$$\Psi_j = 2\xi(\xi-\frac{1}{2}), \tag{D.6}$$

$$\Psi_{ki} = 4\eta(1-\xi-\eta), \tag{D.7}$$

$$\Psi_{jk} = 4\xi\eta, \tag{D.8}$$

$$\Psi_k = 2\eta(\eta-\frac{1}{2}). \tag{D.9}$$

The derivatives of these with respect to ξ are:

$$\frac{\partial\Psi_i}{\partial\xi} = -3+4\xi+4\eta, \tag{D.10}$$

$$\frac{\partial\Psi_{ij}}{\partial\xi} = 4-8\xi-4\eta, \tag{D.11}$$

$$\frac{\partial\Psi_j}{\partial\xi} = -1+4\xi, \tag{D.12}$$

$$\frac{\partial\Psi_{ki}}{\partial\xi} = -4\eta, \tag{D.13}$$

$$\frac{\partial\Psi_{jk}}{\partial\xi} = 4\eta, \tag{D.14}$$

$$\frac{\partial\Psi_k}{\partial\xi} = 0, \tag{D.15}$$

and with respect to η these are:

$$\frac{\partial\Psi_i}{\partial\eta} = -3+4\xi+4\eta, \tag{D.16}$$

$$\frac{\partial\Psi_{ij}}{\partial\eta} = -4\xi, \tag{D.17}$$

$$\frac{\partial\Psi_j}{\partial\eta} = 0, \tag{D.18}$$

$$\frac{\partial\Psi_{ki}}{\partial\eta} = 4-4\xi-8\eta, \tag{D.19}$$

$$\frac{\partial\Psi_{jk}}{\partial\eta} = 4\xi, \tag{D.20}$$

$$\frac{\partial\Psi_k}{\partial\eta} = -1+4\eta. \tag{D.21}$$

D.3.3 Building the transformation Jacobian

To obtain the derivatives of shape functions expressed in local element coordinates (η and ξ) with respect to the global physical coordinates (X_1, X_2), first the chain rule is used:

$$\begin{pmatrix} \frac{\partial \Psi_i}{\partial \xi} \\ \frac{\partial \Psi_i}{\partial \eta} \end{pmatrix} = \begin{pmatrix} \frac{\partial \Psi_i}{\partial X_1}\frac{\partial X_1}{\partial \xi} + \frac{\partial \Psi_i}{\partial X_2}\frac{\partial X_2}{\partial \xi} \\ \frac{\partial \Psi_i}{\partial X_1}\frac{\partial X_1}{\partial \eta} + \frac{\partial \Psi_i}{\partial X_2}\frac{\partial X_2}{\partial \eta} \end{pmatrix}. \tag{D.22}$$

This can be rearranged into the Jacobian ($\mathbf{J}^e$) multiplied by a vector of derivatives as follows:

$$\begin{pmatrix} \frac{\partial \Psi_i}{\partial \xi} \\ \frac{\partial \Psi_i}{\partial \eta} \end{pmatrix} = \begin{pmatrix} \frac{\partial X_1}{\partial \xi} & \frac{\partial X_2}{\partial \xi} \\ \frac{\partial X_1}{\partial \eta} & \frac{\partial X_2}{\partial \eta} \end{pmatrix} \begin{pmatrix} \frac{\partial \Psi_i}{\partial X_1} \\ \frac{\partial \Psi_i}{\partial X_2} \end{pmatrix}, \tag{D.23}$$

where the Jacobian, $\mathbf{J}^e$, is the 2x2 matrix on the right hand side of Equation (D.23). The required derivatives can then be obtained by rearranging Equation (D.23), to give

$$\begin{pmatrix} \frac{\partial \Psi_i}{\partial X_1} \\ \frac{\partial \Psi_i}{\partial X_2} \end{pmatrix} = (\mathbf{J}^e)^{-1} \begin{pmatrix} \frac{\partial \Psi_i}{\partial \xi} \\ \frac{\partial \Psi_i}{\partial \eta} \end{pmatrix}. \tag{D.24}$$

From Equations (D.1) and (D.2), the derivative of the global coordinates, with respect to the local coordinates, can be calculated, giving

$$\mathbf{J}^e = \begin{pmatrix} \sum_{i=1}^{6} \frac{\partial \Psi_i}{\partial \xi} X_1^i & \sum_{i=1}^{6} \frac{\partial \Psi_i}{\partial \xi} X_2^i \\ \sum_{i=1}^{6} \frac{\partial \Psi_i}{\partial \eta} X_1^i & \sum_{i=1}^{6} \frac{\partial \Psi_i}{\partial \eta} X_2^i \end{pmatrix}. \tag{D.25}$$

The determinant of the Jacobian ($det(\mathbf{J}^e)$) can be calculated directly and the inverse of the Jacobian ($(\mathbf{J}^e)^{-1}$) can also be calculated directly by the standard method of inverting a 2x2 matrix:

$$(\mathbf{J}^e)^{-1} = \frac{1}{det(\mathbf{J}^e)} \begin{pmatrix} \frac{\partial X_2}{\partial \eta} & -\frac{\partial X_2}{\partial \xi} \\ -\frac{\partial X_1}{\partial \eta} & \frac{\partial X_1}{\partial \xi} \end{pmatrix}. \tag{D.26}$$

D.3.4 Calculating the derivatives

It is now possible to calculate the twelve derivatives of the nodal shape functions Ψ_i with respect to X_1 and X_2. If the inverse of the Jacobian ($(\mathbf{J}^e)^{-1}$) is defined as a 2×2 matrix **JE**, these are as follows:

$$\frac{\partial \Psi_i}{\partial X_1} = JE_{11}\frac{\partial \Psi_i}{\partial \xi} + JE_{12}\frac{\partial \Psi_i}{\partial \eta} \tag{D.27}$$

and

$$\frac{\partial \Psi_i}{\partial X_2} = JE_{21}\frac{\partial \Psi_i}{\partial \xi} + JE_{22}\frac{\partial \Psi_i}{\partial \eta}. \tag{D.28}$$

where $i = 1...6$. The next step is to calculate the derivatives of the deformation unknowns (x_1, x_2) with respect to X_1 and X_2. For the x_1 unknowns these are:

$$\frac{\partial x_1}{\partial X_1} = \sum_{i=1}^{6} \frac{\partial \Psi_i}{\partial X_1} x_1^i \tag{D.29}$$

and

$$\frac{\partial x_1}{\partial X_2} = \sum_{i=1}^{6} \frac{\partial \Psi_i}{\partial X_2} x_1^i. \tag{D.30}$$

Using the derivatives defined above it is now possible to formulate the governing equations (as defined in Equation (3.33) and Equation (3.38)), in terms of the local coordinate system.

It should be noted that the pressure terms use linear elements and so the linear version of the basis functions are used to calculate these terms, and these only use the nodes at the vertices of the triangle. These linear functions are:

$$\Psi_j^{lin} = \xi, \tag{D.31}$$

$$\Psi_k^{lin} = \eta, \tag{D.32}$$

$$\Psi_i^{lin} = (1 - \xi - \eta). \tag{D.33}$$

D.3.5 Calculating the numerical integral

Quadrature is used to calculate the integral over each element and this is defined in more detail in Section 3.3.5. When using isoparametric elements over a local reference coordinate system, it has the benefit that the quadrature loop uses the same triangle each time, and so the quadrature points are always the same. The quadrature equation used is as follows:

$$\int\int_{\Omega^e} f(\xi, \eta) \mathrm{d}\xi \, \mathrm{d}\eta \approx \triangle^e \sum_{q=1}^{n} W_q f(\xi_q, \eta_q) \, det(\mathbf{J^e}(\xi_q, \eta_q)), \tag{D.34}$$

where ξ_q and η_q are the evaluation points for the function, n is the number of points evaluated over, Ω^e is the triangular element, $\triangle^e$ is the area of the triangular isoparametric element and W_q are the weighting values. As the coordinate system has changed the resulting values in the quadrature loop are multiplied by $det(\mathbf{J}^e(\xi_q, \eta_q))$.

The function f is calculated using the derivatives defined in Section D.3.4 to build the weak form of the two governing equations for the deformation and pressure unknowns (see Equations (3.33) and (3.38)). For each element of the domain the quadrature loop is

undertaken and this calculates the values which are distributed to the rows of the $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ system.