# Efficient cardiac simulations using the Runge–Kutta–Chebyshev method

A Thesis Submitted to the

College of Graduate and Postdoctoral Studies

in Partial Fulfillment of the Requirements

for the degree of Master of Science

in the Department of Computer Science

University of Saskatchewan

Saskatoon

By

Wenxian Guo

# Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science

176 Thorvaldson Building

110 Science Place

University of Saskatchewan

Saskatoon, Saskatchewan

Canada

S7N 5C9

# Abstract

Heart disease is one of the leading causes of death in Canada, claiming thousands of lives each year. Cardiac electrophysiology that studies the electrical activity in the human heart has emerged as an active research field in response to the demand for providing reliable guidance for clinical diagnosis and treatment to heart arrhythmias. Computer simulation of electrophysiological phenomena provides a non-invasive way to study the electrical activity in the human heart and to provide quantitative guidance to clinical applications.

With the need to unravel underlying physiological details, mathematical models tend to be large and possess characteristics that are challenging to mitigate. In this thesis, we describe numerical methods for solving widely used mathematical models: the bidomain model and its simplified form, the monodomain model. The bidomain model is a multi-scale cardiac electrophysiology model that includes a set of reaction-diffusion partial differential equations (PDEs) with the reaction term representing cardiac cell models that describes the chemical reactions and flows of ions across the cell membrane of myocardial cells at the micro level and the diffusion term representing current propagation through the heart at the macro level. We use the method of lines (MOL) to obtain numerical solution of this model. The MOL first spatially discretizes the system of PDEs, resulting in a system of ordinary differential equations (ODEs) at each space point, and we obtain fully discrete solutions at each space-time point using time-integration methods for ODEs. In this thesis, we propose innovative numerical methods for the time integration of systems of ODEs based on the Runge–Kutta–Chebyshev (RKC) method. We implement and compare our methods with those used by current research on time integration of ODEs on three problems: time integration of individual cardiac cell models, time integration of the cell model of a monodomain problem, and time integration of spatially discretized tissue equation in a monodomain benchmark problem proposed by S. Niederer et al. in 2011.

Numerical methods in cardiac electrophysiology research for solving ODEs include the forward Euler (FE) method, the Rush–Larsen (RL) method, the backward Euler method, and the generalized RL method of first-order. We introduce multistage first-order RKC methods and multistage first-order RL methods that are constructed by replacing the FE method with multistage first-order RKC methods. We implement all the aforementioned methods and test their efficiencies in time integration of 37 cardiac cell models. We find introducing the multistage RKC and RL methods allows larger step sizes to meet prescribed numerical accuracy; the increased time steps sped up time integration of 19 cell models. We replace the FE method with two-stage RKC method in time integration of cell model in a monodomain model. We find the increased time step introduced by applying this method improved the entire solving process by up to a factor of 1.4. We also apply the $\text{RKC}(2,1)$ method to time integration of the tissue equation from a monodomain benchmark problem. Results show we have decreased the execution time of this benchmark problem by a factor of two. We note the increase of time step is from stability improvement brought by the numerical method. We finally give a quantitative explanation of stability improvement from introducing multistage RKC and RL methods for solving systems of ODEs considered in this thesis.

# ACKNOWLEDGEMENTS

```
I work hard everyday so that all who love me will be proud of me!  (From <To Survive>)
```

— Xiyao "Sweet Coffee" Liu (Counter-Strike filmmaker)

# CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| BE | Backward Euler |
| FE | Forward Euler |
| FEM | Finite Element Method |
| FVM | Finite Volumn Method |
| GRL1 | Generalized first-order Rush–Larsen method |
| GRL2 | Generalized second-order Rush–Larsen method |
| IVP | Initial-Value Problem |
| MRMS | Mixed Root Mean Square |
| MOL | Method of Lines |
| ODE | Ordinary Differential Equation |
| OS | Operator Splitting |
| PDE | Partial Differential Equation |
| RK | Runge–Kutta |
| RKC | Runge–Kutta–Chebyshev |
| $\mathrm{RKC}(s,1)$ | $s$-stage first-order Runge–Kutta–Chebyshev method |
| RL | Rush–Larsen |
| $\mathrm{RLC}(s,1)$ | Rush–Larsen method with the $\mathrm{RKC}(s,1)$ method instead of the FE method |
| SI | Semi-Implicit |

# CHAPTER 1

# INTRODUCTION

Heart disease is one of the leading causes of death in Canada. According to the most recent report, it claims more than 48,000 lives in 2012 [45]. One of the crucial factors contributing to normal heart function is the electrical activity in the heart. The study of electrical activity in the heart has emerged as an active research field — cardiac electrophysiology.

During clinical diagnoses and treatments, it is usually desirable to avoid invasive experiments on a live human heart in research studies. To mitigate these difficulties, researchers from physiology, mathematics, and computer science collaborate together to develop mathematical models describing physiological phenomena using knowledge and data from clinical practice and research. The confluence of physiology and computing science brings cardiac simulation — simulating cardiac electrical activity using numerical simulation methods — into the academic spotlight.

Cardiac electrical activity can be simulated using different methods, including cellular automata, the Eikonal mapping, and the bidomain or monodomain model [6]. One common practice adopted in this thesis is to solve the bidomain model, or alternatively, its simplified form, the monodomain model. The bidomain model is a multi-scale physiology model that couples a set of *partial differential equations* (PDEs) describing chemical reactions in myocardial cell, which are finally reduced to *ordinary differential equations* (ODEs) upon spatial discretization, and ion transport through cell membrane with a set of PDEs describing current propagation throughout the heart.

Success in mathematical modelling stimulates the development of numerical methods for solving mathematical models of cardiac electrophysiology, especially with special characteristics (as we mention in Section 3.1). However, due to the intrinsic large scale of problems as demanded by clinical applications, it is not presently possible to simulate the electrical activity of a meaningful piece of heart tissue in real-time. As of 2012, the fastest result for solving the monodomain model with spatial resolution of 0.1 mm on 370 million grid cells was on a supercomputer with 1,572,864 cores [36], but only nine heartbeats per CPU wall clock minute were obtained, compared with 60 to 100 heart beats in a healthy adult heart that consists of two billion cells [27]. Advanced numerical methods have a pronounced importance in facilitating computations and moving researchers towards real-time cardiac simulation.

## 1.1 Contributions of the thesis

In this thesis, we investigated the time integration to cardiac cell models in cardiac simulations by using the Runge–Kutta–Chebyshev (RKC) methods and the new methods by replacing the forward Euler (FE) method in the Rush–Larsen (RL) method with multistage RKC methods. We refer to the new methods as multistage Rush–Larsen–Chebyshev (RLC) methods. The $s$-stage first-order RKC methods and multistage RLC methods are denoted as $\text{RKC}(s, 1)$ and $\text{RLC}(s, 1)$, where $s = 2, 3$. In this experiment, all these methods are implemented in MATLAB, along with other widely used existing methods for cardiac simulation. While specifying prescribed numerical accuracy over time integrations of a wide range of cardiac cell models, we find and compare the methods' largest step sizes and corresponding execution times. Our findings indicate that the multistage RKC and RLC methods reduce the total execution time in time integration of 19 out of 37 selected cell models compared with single step methods. As well as providing a quantitative analysis relating to these improvements, we apply the $\text{RKC}(2, 1)$ to the time integration of cardiac cell models of a monodomain problem. By implementing the $\text{RKC}(2, 1)$ method in C++, this experiment contributes to a simulation toolkit for cardiac simulation named CHASTE [35]. With the hope of pushing the research boundary towards real-time cardiac simulation, we also test the capability of the RKC method by applying the $\text{RKC}(2, 1)$ method to time integration of a spatially discretized tissue equation in a monodomain benchmark problem. Replacing the FE method used in previous research [36] with the $\text{RKC}(2, 1)$ method reduces the computing time by a factor of two.

## 1.2 Structure of the thesis

In Chapter 2, we briefly introduce cardiac simulation and the electrophysiological models relevant to this thesis, specifically presenting those models concerned with individual cardiac cell and myocardial tissue. Chapter 3 introduces numerical methods for time integration and spatial discretization of the aforementioned electrophysiological models used by cardiac simulation research. We propose our innovative numerical methods for time integration of systems of ODEs, as well as presenting a special note on numerical stability, a crucial factor in determining the applicable numerical method. This presentation is followed by an introduction to the method of lines (MOL) used to solve the bidomain/monodomain model. The MOL includes the spatial discretization of a system of PDEs and time integration of spatially discretized PDEs that results a system of ODEs, which can be solved using numerical methods for ODEs. In Chapter 4, we present results from numerical experiments for solving the following three problems: the time integration of individual cardiac cell models, the time integration of the cell model of a monodomain problem, and the time integration of a spatially discretized tissue equation in a monodomain benchmark problem proposed by S. Niederer et al. in 2011 [36]. A theoretical explanation for the improvements on step sizes seen in our numerical is also provided. Chapter 5 summarizes the conclusions and offers suggestions for future work.

CHAPTER 2

ELECTROPHYSIOLOGICAL BACKGROUND

In this chapter, we give a brief introduction to the electrophysiological background used in this thesis. We begin by giving a brief overview on cardiac electrophysiology and the motivation of cardiac simulation — using computer simulation to study cardiac electrophysiology in Section 2.1. We further present electrophysiological details on specific cardiac cells and mathematical models describing cellular electrical activity in Section 2.2. We conclude this chapter by describing tissue-scale modeling in Section 2.3.

## 2.1 Introduction to cardiac simulation

### 2.1.1 Overview of electrical activity in the heart

The heart is a precise machine that controls blood pumping, which is the result of a rhythmic cycle of contraction and relaxation of billions of muscle cells. Those muscle cells are controlled by the pattern of electrical activity. Abnormalities and disturbance of the electrical activity may result in serious heart problems.

A popular non-invasive method in clinical practice for diagnosing heart conditions is the ECG (electrocardiogram), which is based on the link between electrical activity and potential heart problems. By treating human body as a volume conductor, the ECG reflects electrical potential difference that results from electrical activity in the heart. Pathological conditions, such as rhythm disturbance, can be detected through abnormal patterns from the ECG report. We refer to [25] for a detailed discussion on this topic.

### 2.1.2 Computer simulation of cardiac electrophysiology

Despite advances in understanding electrophysiological mechanisms, more knowledge of quantitative methods is needed to guide drug research and clinical treatment. To study the hearts physiological reactions under extreme conditions, such as those during defibrillation, physicians are increasingly turning to virtual hearts in both academic and settings. These models of the heart use today's computing power and sophisticated mathematical algorithms to simulate electrical activity in the heart and diagnose complex heart problems. A widely used cardiac cell model in cardiac electrophysiology research is the Luo–Rudy model (1991) [29]. The Luo–Rudy model (1991) is based on developments in the ventricular cell model proposed by Beeler and

Reuter [3].

Cardiac simulation, characterized by its non-invasive nature and providing quantitative information, has pronounced its importance in guiding drug research [36] and in understanding heart arrhythmias [2]. Real-time cardiac simulation affords us a promising future in diagnosing and predicting heart disease in a drop-in clinic, and it makes the drug research that requires investigating long-term electrical activity possible. However, with the intrinsic complex and complicated structure the heart has, the computing workload for real-time cardiac simulation is still formidable.

With the rapid development from semiconductor industries, computing facilities on the market is becoming more and more powerful. The development of parallel computing facilities enables us to tackle larger scale problems than before. However, those facilities are driven by mathematical algorithms that make full and efficient utilization of computing resources. Algorithm development thus has a significant impact in reducing the overall computing time given fixed facilities.

This thesis concerns the improvement of algorithms designed to move the field closer to real-time cardiac simulation.

## 2.2 Cellular electrophysiology

To describe electrical activity of a single cell, several mathematical models have been proposed. These models can be classified into three categories:

**Phenomenological models** These models describe cellular electrophysiology via fitting of observed behaviours of cells. An example is the cubic polynomial model [26]:

$$I_{ion} = f(v) = A^2(v - v_{rest})(v - v_{th})(v - v_{peak}),$$

where $A$, $v_{rest}$, $v_{th}$, and $v_{peak}$ are constants for this model.

**First-generation models** These models, such as the Luo–Rudy model (1991) [29] described later in this section, are a first attempt to describe electrical activity using the underlying electrophysiology. However, the electrophysiology being used is much simplified thus hides details of the real electrophysiology.

**Second-generation models** Benefitting from further observations and advanced experimental techniques, these models provide substantial amount of details for descriptions of the electrophysiology of the cells, such as the improved Luo–Rudy model (1994) [30].

The transmembrane potential of a single cell is affected by the ionic currents, which consist of ions carrying electric charges passing through the cell membrane, and any external stimulus exciting the cell; this can be written as the following *ordinary differential equation* (ODE):

$$C_m \frac{dv}{dt} = -I_{\text{ion}} + I_{\text{stim}}, \tag{2.1}$$

4

where $C_m$, $v$, $t$, and $I_{\text{ion}}$ are the capacitance of the cell membrane, the transmembrane potential, time, and the ionic current through cell membrane, respectively. $I_{\text{stim}}$ denotes a stimulus current to excite the cell.

The transmembrane ionic currents can be written as

$$I_{\text{ion}} = \sum_{j=1}^{n_{\text{ion}}} I_j,$$

where $j$ stands for a specific kind of ion, such as sodium or potassium, and $n_{\text{ion}}$ is the total number of ionic currents. Each ionic current is controlled by several state variables, denoted by an $m$-component vector $\boldsymbol{s}$:

$$I_j \equiv I_j(\boldsymbol{s}, v, t), \quad \frac{\mathrm{d}\boldsymbol{s}}{\mathrm{d}t} = \boldsymbol{f}(\boldsymbol{s}, v, t). \tag{2.2}$$

The state variables $\boldsymbol{s}$, depending on $v$ and $t$, can be categorized into two groups: $m_g$ gating variables $\boldsymbol{g}$ and $m_c$ non-gating variables $\boldsymbol{c}$, with $m = m_g + m_c$. The gating variables describe electrophysiological behaviours of ion channels, which are proteins controlling specific kind of ions transported through cell membrane (as shown in Figure 2.1). The non-gating variables describe dynamic variations of intracellular concentrations of ions. This effect can be written as

$$\frac{\mathrm{d}g_j}{\mathrm{d}t} = \alpha_j(1 - g_j) - \beta_j g_j, \quad j = 1, 2, \cdots, m_g, \tag{2.3}$$

$$\frac{\mathrm{d}c_j}{\mathrm{d}t} = h_j(c_j, g_j, v, t), \quad j = 1, 2, \cdots, m_c, \tag{2.4}$$

$\alpha_j$, $\beta_j$, and $h_j$ are functions depending on $v$ and $t$. In (2.3), $\alpha_j$ and $\beta_j$ are nonlinear functions of $v$.
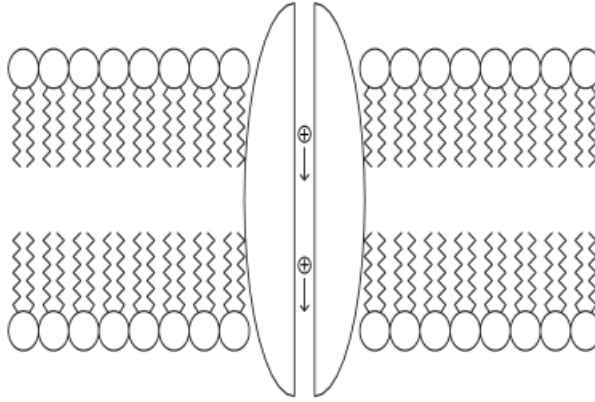


**Figure 2.1:** Schematic illustration of cell membrane showing an ion channel controlling ion transport [56].

**The Luo–Rudy model (1991)**

A widely used cardiac cell model in cardiac electrophysiology research is the Luo–Rudy model (1991) [29]. The Luo–Rudy model (1991) is based on further development of the ventricular cell model proposed by Beeler and Reuter [3].

This model includes seven gating variables controlling six ionic currents (as shown in Figure 2.2) and takes the form:

$$
\begin{aligned}
-C_m \frac{\mathrm{d}v}{\mathrm{d}t} &= I_{\mathrm{Na}} + I_{\mathrm{si}} + I_{\mathrm{K}} + I_{\mathrm{K1}} + I_{\mathrm{Kp}} + I_{\mathrm{b}} + I_{\mathrm{stim}}, \\
\frac{\mathrm{d}[Ca]_i}{\mathrm{d}t} &= -0.0001 I_{\mathrm{si}} + 0.07(0.0001 - [Ca]_i), \\
\frac{\mathrm{d}g}{\mathrm{d}t} &= \alpha_g(1-g) - \beta_g,
\end{aligned}
$$

with terms listed in Table 2.1.
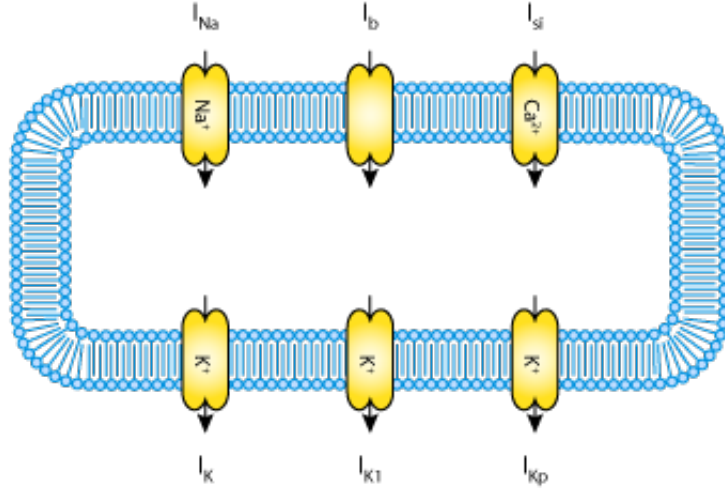


**Figure 2.2:** In the Luo–Rudy model (1991), different ion channels control different ionic currents [1].

**Table 2.1:** Parameters of the Luo–Rudy model (1991).

| | |
|---|---|
| $[Ca]_i$ | intracellular calcium concentration |
| $g = m, h, j, d, f, X, X_i$ | gating variables |
| $\alpha_g$ and $\beta_g$ | parameters for the gate $g$ |
| $I_{\mathrm{Na}}$ | normal fast sodium current |
| $I_{\mathrm{si}}$ | slow inward current of calcium |
| $I_{\mathrm{K}}$ and $I_{\mathrm{K1}}$ | time-dependent and time-independent potassium current |
| $I_{\mathrm{Kp}}$ | plateau potassium current |
| $I_{\mathrm{b}}$ | passive background current |

We refer to Appendix A for a complete form of the Luo–Rudy model (1991).

## 2.3  Tissue-scale modelling

Individual cells form the heart tissue. We describe mathematical models that couple the electrical activity of the individual cells together and describe current propagation throughout a slab of heart tissue.

Several mathematical models have been proposed by the research community focusing on cardiac electrophysiology in the past decades. In this thesis, we focus on a model that is widely used by several simulation toolkits (for example, `CHASTE` [35] developed by the Oxford University and `Nektar++` [24] developed by the Imperial College London) called the bidomain model. We also introduce a simplified (but still practically useful) form of the bidomain model known as the monodomain model.

### 2.3.1  The bidomain model

Originally proposed by Tung in 1978 [59], the bidomain model treats every point of myocardial tissue as being in two continuous domains: the intracellular domain and the extracellular domain. The two domains are separated by cell membrane, which is treated as an insulator — no ion shall pass through except via ion channels as described in Section 2.2.

Suppose the currents in the two domains are given by [56]

$$\boldsymbol{J}_i = -\sigma_i \nabla u_i, \tag{2.5a}$$

$$\boldsymbol{J}_e = -\sigma_e \nabla u_e, \tag{2.5b}$$

where $\sigma_i$ and $\sigma_e$ are the conductivity tensors in the intracellular and extracellular domains, $\boldsymbol{J}_i$ and $\boldsymbol{J}_e$ denote currents in these two domains, and $u_i$ and $u_e$ are the respective potentials. By definition, the transmembrane potential $v$ satisfies

$$v = u_i - u_e. \tag{2.6}$$

Because the cell membrane is thin, the total charge accumulation at any point in the membrane is zero. This implies

$$\frac{\partial}{\partial t}(q_i + q_e) = 0, \tag{2.7}$$

where $q_i$ and $q_e$ are the intracellular and extracellular charges, respectively.

In each domain, the net current into a point equals the sum of the rate of charge accumulation at that point and the existing ionic current in the domain at that point. This implies

$$-\nabla \cdot \boldsymbol{J}_i = \frac{\partial q_i}{\partial t} + \chi I_{\text{ion}}, \tag{2.8a}$$

$$-\nabla \cdot \boldsymbol{J}_e = \frac{\partial q_e}{\partial t} - \chi I_{\text{ion}}, \tag{2.8b}$$

where $I_{\text{ion}}$ is ionic current through cell membrane, $\chi$ represents area of cell membrane per unit volume, $I_{\text{ion}}$ is measured per unit area of the cell membrane and is considered to be positive towards extracellular domain, and densities of charge $q_i$ and $q_e$ and currents $\boldsymbol{J}_i$ and $\boldsymbol{J}_e$ are measured per unit volume.

From (2.7) and (2.8), we have

$$\nabla \cdot \boldsymbol{J}_i + \nabla \cdot \boldsymbol{J}_e = 0. \tag{2.9}$$

Substituting (2.5a) and (2.5b) into (2.9) gives us

$$\nabla \cdot (\sigma_i \nabla u_i) + \nabla \cdot (\sigma_e \nabla u_e) = 0.$$

Because capacitance equals charge divided by potential, if we further define total amount of charge as

$$q = \frac{1}{2}(q_i - q_e), \tag{2.10}$$

we thus have

$$v = \frac{q}{\chi C_m}. \tag{2.11}$$

Taking partial derivative with respect to $t$ on both sides of (2.11) and using (2.10), we have

$$\chi C_m \frac{\partial v}{\partial t} = \frac{1}{2} \frac{\partial (q_i - q_e)}{\partial t}.$$

From (2.7), we also have

$$\frac{\partial q_i}{\partial t} = -\frac{\partial q_e}{\partial t} = \chi C_m \frac{\partial v}{\partial t}.$$

Combining with (2.8) and adding on external stimulus $I_{\text{stim}}$ gives us

$$-\nabla \cdot \boldsymbol{J}_i = \chi C_m \frac{\partial v}{\partial t} + \chi I_{\text{ion}} + I_{\text{stim}}.$$

Using (2.5a) and (2.6) gives us

$$\chi C_m \frac{\partial v}{\partial t} + \chi I_{\text{ion}} + I_{\text{stim}} = \nabla \cdot (\sigma_i \nabla v) + \nabla \cdot (\sigma_i \nabla u_e), \tag{2.12a}$$

$$0 = \nabla \cdot (\sigma_i \nabla v) + \nabla \cdot ((\sigma_i + \sigma_e) \nabla u_e). \tag{2.12b}$$

If we assume the heart is surrounded by a non-conductive insulator, on the boundary we have

$$\hat{\boldsymbol{n}} \cdot \boldsymbol{J}_i = 0,$$

$$\hat{\boldsymbol{n}} \cdot \boldsymbol{J}_e = 0,$$

where $\hat{\boldsymbol{n}}$ is the outward unit normal of the surface of the heart. According to (2.5), we have the boundary conditions for the bidomain model:

$$\hat{\boldsymbol{n}} \cdot (\sigma_i \nabla v + \sigma_i \nabla u_e) = 0, \tag{2.13a}$$

$$\hat{\boldsymbol{n}} \cdot (\sigma_e \nabla u_e) = 0. \tag{2.13b}$$

The bidomain equations (2.12) is a multi-scale cardiac electrophysiology model. They are reaction diffusion *partial differential equations* (PDEs) with the reaction term $I_{ion}$ from cardiac cell models describing the chemical reactions and flow of ions across the cell membrane of individual myocardial cells at the micro level, which are ODEs at every grid points, and the diffusion term describing current propagation through the heart at the macro level. Because the bidomain model is a continuum-based model, the electrical activity of a single cell is not captured; rather, the average effect round a spatial point is included. This can be illustrated in Figure 2.3.
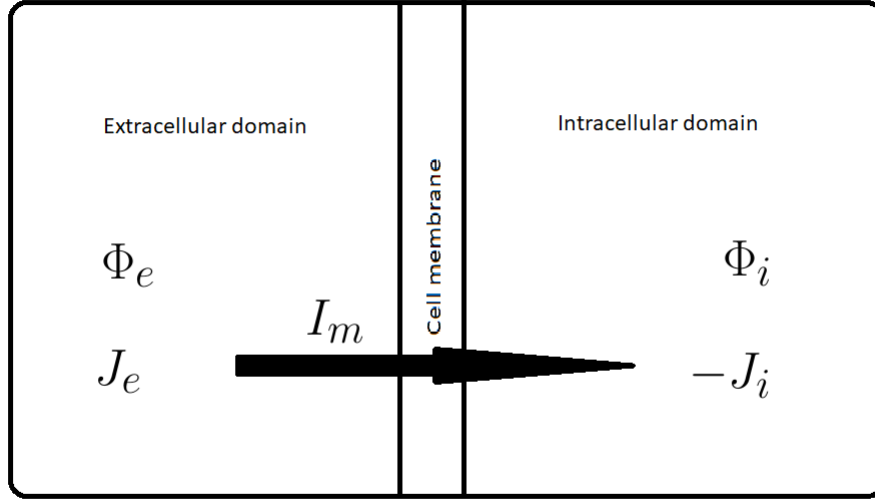


**Figure 2.3:** Schematic illustration of the bidomain model, showing ion transport through cell membrane in two domains: extracellular and intracellular domains. $\Phi_e$ and $\Phi_i$ indicate extracellular and intracellular electric potentials; $J_e$ and $J_i$ indicate extracellular and intracellular electric densities; $I_m$ is the transmembrane current.

### 2.3.2   The monodomain model

It is common in practice to simplify the bidomain model by assuming $\sigma_e = \lambda \sigma_i$. Substituting this assumption into (2.12b) gives us

$$\nabla \cdot (\sigma_i \nabla u_e) = -\frac{\lambda}{1+\lambda} \nabla \cdot (\sigma_i \nabla v). \tag{2.14}$$

Inserting (2.14) into (2.12a) gives us

$$\frac{\lambda}{1+\lambda} \nabla \cdot (\sigma_i \nabla v) = \chi C_m \frac{\partial v}{\partial t} + \chi I_{\text{ion}} + I_{\text{stim}}, \tag{2.15}$$

subject to the boundary condition

$$\hat{\boldsymbol{n}} \cdot (\sigma_i \nabla v) = 0.$$

The monodomain model is simpler to solve than the bidomain model, but it also hides some underlying physiology because it treats the myocardial tissue as being isotropic: the conductivities are uniform along different directions, which is not the case in real life. The parameter $\lambda$ is also hard to specify [56].

# CHAPTER 3

# NUMERICAL METHODS

This chapter presents the numerical methods used in this thesis. We begin by introducing potential challenges behind solving cardiac electrophysiology models and how we determine the accuracy of a numerical solution in Section 3.1. We follow by introducing the numerical methods, including the methods we proposed to use ($s$-stage first-order Rush–Larsen–Chebyshev methods and the $s$-stage first-order Runge–Kutta–Chebyshev methods, where $s = 2, 3$) for the time integration of systems of ODEs, especially targeted at those from cardiac cell models, in Section 3.2. We also describe a crucial factor in determining the capability of a numerical method applied to a problem, called numerical stability, in this section. We turn our focus from single cell simulation to tissue-scale modelling by introducing numerical methods for solving the bidomain model using the *method of lines* in Section 3.3. We proceed by explaining the two steps in the method of lines applied to solve the bidomain and monodomain model: spatial discretization methods for time-dependent parabolic PDEs and time integration methods for systems of ODEs in Section 3.3.1 and Section 3.3.2, respectively.

## 3.1   Challenges in cardiac simulation

The attempt to obtain numerical solutions for cardiac electrophysiology models faces several challenges:

**Scale** In practical applications, the scale of the intended problem must be big enough to unravel underlying electrophysiological details of the problem, and this scale imposes high demands on computing facilities. Advanced numerical methods help computing facilities to be capable for handling large-scale problems more efficiently.

**Stifness** Models studied in cardiac simulations sometimes require special techniques to deal with their special characteristics. One of these is called stiffness. Details on stiffness are introduced in Section 3.2.2.

In the following sections, we introduce the numerical methods widely adopted by the cardiac simulation community, as well as those used in this study.

## 3.2 Time-integration methods for systems of ODEs

We introduce numerical methods for time integration of systems of ODEs in this section, especially for systems from cardiac cell models. In these cases, the initial values of ODEs come from the initial status of the cell model. This brings us an initial-value problem (IVP). To illustrate this, let us consider an IVP over $t \in [t_0, +\infty)$:

$$\frac{\mathrm{d}\boldsymbol{y}}{\mathrm{d}t} = \boldsymbol{f}(t, \boldsymbol{y}), \quad t_0 > 0, \tag{3.1a}$$

subject to initial condition

$$\boldsymbol{y}(t_0) = \boldsymbol{y}_0, \tag{3.1b}$$

where $\boldsymbol{y}$ and $\boldsymbol{f}$ are both vector-valued functions and $\boldsymbol{y}_0$ is a vector denoting the initial condition.

Solution to system of ODEs may not always be expressed in analytical form. In these cases, we use numerical methods to get approximate values $(\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n)$ of $\boldsymbol{y}$ at $t = t_1, t_2, \ldots, t_n$. Numerical methods can be categorized into two major groups: explicit methods and implicit methods. Explicit methods are straightforward because they allow us to obtain solution at next step (from $t_n$ to $t_{n+1}$) directly. Implicit methods involve solving algebraic equations for the numerical solution at each step, and this extra solving process leads to a relatively high computing cost per step, but implicit methods are usually more stable (the concept of numerical stability is explained in Section 3.2.2) than their explicit counterparts. This allows implicit methods to take a larger step size in computing and could potentially reduce execution time for some problems. There are also methods that are neither fully explicit nor fully implicit, such as the *symplectic Euler* method [8] that takes an implicit form but can be solved explicitly.

### 3.2.1 Runge–Kutta methods

One large family of numerical methods for solving the systems of ODEs is the family of *Runge–Kutta* (RK) methods. RK methods are one-step methods, which are characterized by the fact that the solution at current step only depends on solution at previous step. However, in the one-step time integration, auxiliary intermediate solutions may also be computed depending on the RK method. With a large variety of the forms they take, RK methods range from fully explicit to fully implicit.

The $s$-stage RK method takes the following form [20]:

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + \Delta t \sum_{i=1}^{s} b_i \boldsymbol{f}(t_n + c_i \Delta t, \boldsymbol{y}_{ni}), \tag{3.2a}$$

$$\boldsymbol{y}_{ni} = \boldsymbol{y}_n + \Delta t \sum_{j=1}^{s} a_{ij} \boldsymbol{f}(t_n + c_j \Delta t, \boldsymbol{y}_{nj}), \tag{3.2b}$$

where $a_{ij}$ and $b_i$ define the specific RK method, and $c_i = \sum_{j=1}^{s} a_{ij}$.

The simplest RK method for solving an IVP is the *forward Euler* (FE) method:

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + \Delta t \boldsymbol{f}(t_n, \boldsymbol{y}_n).$$

The implicit counterpart of the FE method is the *backward Euler* (BE) method:

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + \Delta t \boldsymbol{f}(t_{n+1}, \boldsymbol{y}_{n+1}). \tag{3.3}$$

Because $\boldsymbol{y}_{n+1}$ appears at both sides of (3.3), the BE method generally requires a solution of a non-linear algebraic equation to obtain $\boldsymbol{y}_{n+1}$.

### 3.2.2 Numerical stability and stiffness

Stability is a crucial topic in the research of numerical methods, and it plays an important role in justifying the capability of a numerical method applied to a specific problem. To illustrate the concept of numerical stability, let us consider the following scalar test IVP as a typical example:

$$\frac{\mathrm{d}y}{\mathrm{d}t} = \lambda y(t), \quad y(t_0) = y_0, \tag{3.4}$$

where $\lambda \in \mathbb{C}$ and $Re(\lambda) \leq 0$. Applying the RK method (3.2) to (3.4) gives us [20]

$$y_{n+1} = R(z)y_n, \quad z = \lambda \Delta t,$$

where

$$R(z) = 1 + z\boldsymbol{b}^{\mathsf{T}}(\mathbf{I} - z\mathbf{A})^{-1}\boldsymbol{e}, \quad \boldsymbol{e} = (1, 1, \ldots, 1)^{\mathsf{T}},$$

$$\boldsymbol{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_s \end{pmatrix}, \mathbf{A} = \begin{pmatrix} a_{11} & \cdots & a_{1s} \\ \vdots & \ddots & \vdots \\ a_{s1} & \cdots & a_{ss} \end{pmatrix},$$

and $\mathbf{I}$ is the identity matrix. $R(z)$ is called the stability polynomial and plays a central role in determining numerical stability [20].

At each step, a perturbation error is introduced. The perturbation error derives from both finite machine precision in the current step and from the difference between the solution trajectory and the numerical solution in the previous step, resulting in different initial values for the current step. For each step, we assume that at step $n$ the exact solution is $y(t_n)$, and that the numerical solution is $y_n$, we have

$$y_{n+1} = R(z)y_n.$$

The perturbation error each step, denoted by $e_n = y(t_n) - y_n$, follows

$$e_{n+1} = R(z)e_n,$$

and this implies

$$e_n = (R(z))^n e_0.$$

To ensure the error does not grow while the computation is going on, we require

$$|R(z)| \leq 1.$$

This imposes a restriction on the scaled eigenvalue by time step $z = \lambda \Delta t$: $z$ belongs to the set

$$\mathbb{S} = \{z : |R(z)| \leq 1\}.$$

$\mathbb{S}$ is called the stability region.

Stability regions of the FE and BE method are shown in Figure 3.1. For example, if we are using the FE method, we require the scaled eigenvalues $z$ fall into the shaded unit disk in the left figure in Figure 3.1.
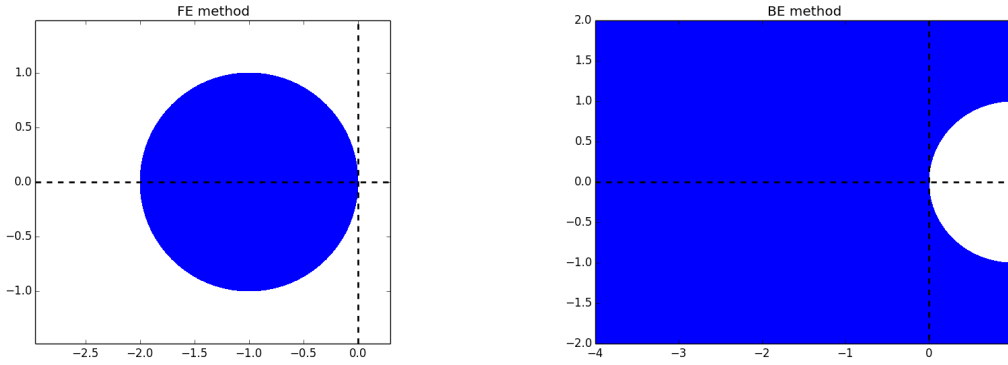


**Figure 3.1:** Stability regions of the FE and BE method.

We note that the cases with which we are dealing in this thesis involve non-linear problems, so we generalize the above analysis from scalar test equation (3.4) to a general non-linear system (3.1).

Consider the non-linear system (3.1). If we denote the numerical solution as $\tilde{\boldsymbol{y}}$, the perturbation error is

$$\boldsymbol{e}(t) = \boldsymbol{y}(t) - \tilde{\boldsymbol{y}}(t).$$

At time-space point $(t^*, \boldsymbol{y}^*)$, we have

$$\begin{aligned}
\frac{\mathrm{d}\boldsymbol{e}}{\mathrm{d}t} &= \frac{\mathrm{d}\boldsymbol{y}}{\mathrm{d}t} - \frac{\mathrm{d}\tilde{\boldsymbol{y}}}{\mathrm{d}t} \\
&= \boldsymbol{f}(t^*, \boldsymbol{y}) - \boldsymbol{f}(t^*, \boldsymbol{y}^*) \\
&= \boldsymbol{f}(t^*, \boldsymbol{y}^*) + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{y}}(t^*, \boldsymbol{y}^*)(\boldsymbol{y} - \boldsymbol{y}^*) - \boldsymbol{f}(t^*, \boldsymbol{y}^*) + O(|\boldsymbol{y}^* - \boldsymbol{y}|^2)
\end{aligned}$$

Dropping the second-order term $O(|\boldsymbol{y}^* - \boldsymbol{y}|^2)$, the perturbation error $\boldsymbol{e}$ satisfies

$$\frac{\mathrm{d}\boldsymbol{e}}{\mathrm{d}t} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{y}}(t^*, \boldsymbol{y}^*)\boldsymbol{e}.$$

The matrix $\mathbf{J} := \partial \boldsymbol{f}(t^*, \boldsymbol{y}^*)/\partial \boldsymbol{y}$ is called the Jacobian matrix of the system (3.1).

Jacobian matrix $\mathbf{J}$ of a vector-valued function $\boldsymbol{f} = (f_1, \ldots, f_m)$ with respect to variable $\boldsymbol{y} = (y_1, \ldots, y_m)$ is defined by

$$\mathbf{J} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{y}} = \begin{pmatrix} \frac{\partial f_1}{\partial y_1} & \cdots & \frac{\partial f_1}{\partial y_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial y_1} & \cdots & \frac{\partial f_m}{\partial y_m} \end{pmatrix}.$$

The eigenvalues of Jacobian matrix provide a lot of information we need in choosing numerical methods.

In the experiments considered in this thesis, we find the Jacobian matrix $\mathbf{J} := \partial \boldsymbol{f}(t^*, \boldsymbol{y}^*)/\partial \boldsymbol{y}$ is always diagonalizable. Therefore, there exists a non-singular matrix $\mathbf{V}$ with columns being eigenvectors of $\mathbf{J}$ such that $\mathbf{V}^{-1}\mathbf{J}\mathbf{V} = \boldsymbol{\Lambda}$, with $\boldsymbol{\Lambda}$ being a diagonal matrix with eigenvalues of $\mathbf{J}$ on its diagonal. Let $\tilde{\boldsymbol{e}} = \mathbf{V}^{-1}\boldsymbol{e}$; then we have

$$\frac{\mathrm{d}\tilde{\boldsymbol{e}}}{\mathrm{d}t} = \boldsymbol{\Lambda}\tilde{\boldsymbol{e}}.$$

For each component $\tilde{e}_i$ in $\tilde{\boldsymbol{e}}$, we have

$$\frac{\mathrm{d}\tilde{e}_i}{\mathrm{d}t} = \lambda_i \tilde{e}_i,$$

where $\lambda_i$ is diagonal element $i$ of $\boldsymbol{\Lambda}$. The analysis and illustrations of stability regarding (3.4) further apply to (3.1): for an $m$-dimensional system, we require step time scaled eigenvalues $z_1, z_2, \ldots, z_m$ fall into the stability region of the respective numerical method. We further investigate the relationship between eigenvalues and how these affect our choosing numerical methods in Section 4.4.

When determining suitable numerical methods to apply to a problem, one factor to consider is stiffness. Stiffness refers to the phenomenon of a numerical method for the solution of a differential equation requiring smaller step sizes than necessary to achieve a given accuracy [17]. A stiff problem's Jacobian matrix tends to contain eigenvalues with large negative real parts [32], which, in turn, restricts the step sizes needed to satisfy stability requirements: the scaled eigenvalues $z = \lambda \Delta t$ belong to the stability region (3.2.2). To satisfy prescribed accuracy, eigenvalues with large negative real parts result in step sizes that are smaller than expected. Furthermore, as seen in Figure 3.1, an implicit method, such as the BE method, has a much larger stability region than its explicit counterpart, the FE method. Thus, the BE method helps address the step size restrictions of stiff problems.

The above analysis that uses eigenvalues to justify stiffness of a problem holds most of the time in practice (and in this thesis), but there are exceptions. We refer to [17] for further discussions on this topic.

### 3.2.3 Runge–Kutta–Chebyshev methods

As shown in Figure 3.1, the stability region of the BE method is larger than that of its explicit counterpart, so the BE method can take a comparatively larger step size than the FE method while still being stable; however, satisfying its requirement for a solution of a non-linear algebraic equation each step is expensive in practical computations. On the other hand, the FE method suffers from step size restriction from small stability region especially on stiff problems. This process results in significant computing time and hence

sometimes is prohibitive. Exploiting the characteristics of the problem, eigenvalues of Jacobian matrix of the cell systems tend to cluster around negative real axis, and to balance the trade-off between reducing computational workload and satisfying numerical stability, we propose to use an explicit multistage method, the *Runge–Kutta–Chebyshev* (RKC) method, which belongs to the family of RK methods. We denote the $s$-stage first-order RKC methods by $\text{RKC}(s, 1)$. The FE method can be viewed as the $\text{RKC}(1, 1)$ method according to the form RKC methods take below.

RKC methods have extended stability regions along the negative real axis compared with other explicit RK methods. It can be proved [60] that for any $s$-stage explicit RK methods applying to the scalar test problem (3.4), the optimal stability polynomial is the shifted Chebyshev polynomial of the first kind:

$$P_s(z) = T_s\left(1 + \frac{z}{s^2}\right), \quad \beta_r = 2s^2,$$

where $[-\beta_r, 0]$ is the longest segment of the negative real axis contained in the stability region $\mathbb{S}$, $s$ is a positive integer, and $T_s(z)$ is the Chebyshev polynomial defined by

$$T_s(z) = \cos(s \arccos(z)), \quad z \in \mathbb{C}.$$

Using the Chebyshev polynomial as the stability polynomial, an $s$-stage RKC method is written as

$$\boldsymbol{y}_{n0} = \boldsymbol{y}_n,$$
$$\boldsymbol{y}_{n1} = \boldsymbol{y}_n + \tilde{\mu}_1 \Delta t \boldsymbol{f}_{n0},$$
$$\boldsymbol{y}_{nj} = (1 - \mu_j - \nu_j)\boldsymbol{y}_n + \mu_j \boldsymbol{y}_{n,j-1} + \nu_j \boldsymbol{y}_{n,j-2} + \tilde{\mu}_j \Delta t \boldsymbol{f}_{n,j-1} + \tilde{\gamma}_j \Delta t \boldsymbol{f}_{n0},$$
$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_{ns},$$

where $j = 2, 3, \ldots, s$, $\boldsymbol{f}_{nk} := \boldsymbol{f}(t_n + c_k \Delta t, \boldsymbol{y}_{nk})$, and

$$\tilde{\mu}_1 = b_1 w_1, \ \mu_j = \frac{2b_j w_0}{b_{j-1}}, \ \nu_j = \frac{-b_j}{b_{j-2}},$$
$$\tilde{\mu}_j = \frac{2b_j w_1}{b_{j-1}}, \ \tilde{\gamma}_j = -a_{j-1}\tilde{\mu}_j, \ a_j = 1 - b_j T_j(w_0).$$

For an $s$-stage first-order RKC method, we introduce damping parameter $\epsilon$, and the quantities $w_0$ and $w_1$ satisfy

$$w_0 = 1 + \epsilon/s^2, \quad w_1 = T_s(w_0)/T_s'(w_0),$$

and

$$b_j = \frac{1}{T_j(w_0)},$$

Damping parameters applied to the RKC method are designed to widen the stability region along the negative real axis with a small penalty of shortening the intersection of the stability region with the negative real axis, as shown in Figure 3.2. In practical implementations, a suitable choice for $\epsilon$ discussed in [20] is 0.05.
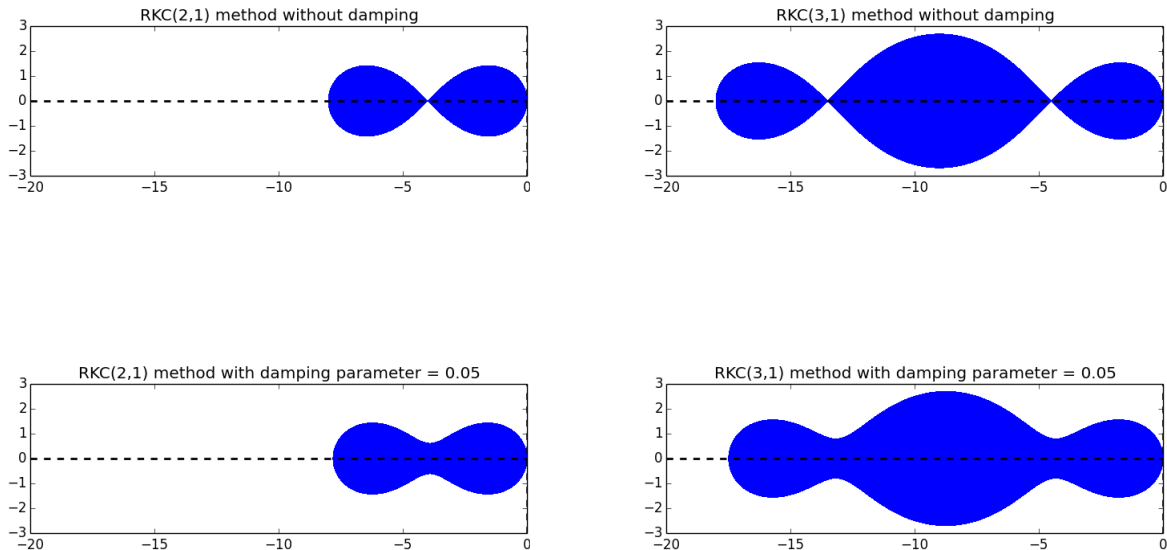
**Figure 3.2:** Stability regions of first-order RKC methods, showing two and three stages with and without damping parameter 0.05.

### 3.2.4 The Rush–Larsen method

As being specified in Section 2.2, systems of equations of the cardiac cell model can be split according to gating and non-gating variables. Exploiting this fact, Moore and Ramon developed a new numerical method by treating gating and non-gating equations using exponential Euler integrator and the RK4 method [37]. Experiment has shown the new method outperformed applying the FE method or the RK method solely to the whole system. Inspired by their research, Rush and Larsen proposed the *Rush–Larsen* (RL) method [50]. This method splits cell systems by treating gating equations with the exponential Euler integrator and treating the remaining equations with the FE method.

Consider the gating equation (2.3) of gate $g$; it can be reformulated as

$$\frac{\mathrm{d}g}{\mathrm{d}t} = \frac{g_\infty - g}{\tau_g}, \tag{3.7}$$

where

$$g_\infty = \frac{\alpha_g}{\alpha_g + \beta_g}, \quad \tau_g = \frac{1}{\alpha_g + \beta_g},$$

with $\alpha_g$ and $\beta_g$ being functions of only $v$.

The RL method freezes the transmembrane potential $v$ at each time step. This allows (3.7) to be a linear equation. This equation has exact solution $g_n$ at $t_n$:

$$g_n = g_\infty + (g_{n-1} - g_\infty)e^{-\frac{\Delta t}{\tau_g}}.$$

17

Advantages of the RL method come from its easy implementation and improved stability properties when dealing with stiff problems where stiffness is captured by gating variables. Previous studies [33] have shown that the RL method is then the most efficient method for the majority of the cardiac cell models in this thesis.

We consider replacing the FE method in the RL method with multistage RKC$(s, 1)$ methods. We denote these new methods with $s$-stage first-order *Rush–Larsen–Chebyshev* (RLC$(s, 1)$) method, where $s = 2, 3$.

### 3.2.5 The Generalized Rush–Larsen method

Several extensions of the RL method have been proposed, including the *generalized first-order Rush–Larsen* (GRL1) method and the *generalized second-order Rush–Larsen* (GRL2) method [55].

#### GRL1

Unlike the RL method that treats non-gating variables with the FE method, GRL1 decouples and linearizes the whole system at $(t_n, \boldsymbol{y}_n)$ and integrates it with an exponential integrator. We write this procedure as follows:

**Local linearization**

$$\frac{\mathrm{d}y_i}{\mathrm{d}t} = f_i(\boldsymbol{y}_n) + \frac{\partial f_i(\boldsymbol{y}_n)}{\partial y_i}(y_i - y_{n,i}), \quad i = 1, 2, \ldots, m, \tag{3.8}$$

where $y_i(t_n) \coloneqq y_{n,i}$, subscript $i$ denotes the component $i$ of $\boldsymbol{y}$ and $\boldsymbol{f}$ that have $m$ components.

**Exponential integration**

$$y_{n+1,i} = y_{n,i} + \frac{a}{b}(e^{b\Delta t} - 1), \quad i = 1, 2, \ldots, m, \tag{3.9}$$

where $a = f_i(\boldsymbol{y}_n)$ and $b = \partial f_i(\boldsymbol{y}_n)/\partial y_i$. In practical implementation, if $|\partial f_i(\boldsymbol{y})/\partial y_i| < \delta$, where $\delta$ is a small number relative to the double-precision of the calculations, we can treat $|\partial f_i(\boldsymbol{y})/\partial y_i| = 0$, and (3.9) degenerates to $y_i(t) = y_{n,i} + a(t - t_n)$.

Previous experiments show that the GRL1 method outperformed the RL method on the stiffest cardiac cell models by taking significantly larger step size while maintaining stability and meeting a prescribed numerical accuracy [33].

#### GRL2

The GRL2 method is a two-stage second-order method. It also decouples and linearizes an ODE system at $(t_n, \boldsymbol{y}_n)$ to get (3.8), but it uses two steps to achieve second-order accuracy:

$$y_{n+1/2,i} = y_{n,i} + \frac{a}{b}(e^{b\Delta t/2} - 1), \quad i = 1, 2, \ldots, m,$$

$$y_{n+1,i} = y_{n,i} + \frac{\bar{a}}{\bar{b}}(e^{\bar{b}\Delta t} - 1), \quad i = 1, 2, \ldots, m,$$

where

$$\bar{a} = f_i(\bar{\boldsymbol{y}}_{n+1/2}), \quad \bar{b} = \partial f_i(\bar{\boldsymbol{y}}_{n+1/2})/\partial y_i,$$

with $\bar{\boldsymbol{y}}_{n+1/2} = \boldsymbol{y}_{n+1/2}$ but the component $i$ being $y_{n,i}$.

Implementation of GRL methods requires computing Jacobian matrix $\partial \boldsymbol{f}/\partial \boldsymbol{y}$. This costly process can be reduced by only computing diagonal elements of Jacobian matrix as needed [32]. The diagonal elements are computed by

$$\frac{\partial f_i(\boldsymbol{y})}{\partial y_i} \approx \frac{f_i(y_1, \ldots, y_{i-1}, y_i + \delta, y_{i+1}, \ldots, y_m) - f_i(\boldsymbol{y})}{\delta}.$$

## 3.3    The method of lines

In this thesis, the multi-scale cardiac electrophysiology models we use are the bidomain model or the monodomain model. One of the predominant frameworks used by the cardiac simulation community to solve the bidomain model is the *method of lines* (MOL).

The MOL refers to a general procedure for numerically solving systems of time-dependent PDEs, which include the bidomain equations (2.12) (or the monodomain equation (2.15)). The MOL discretizes the system of PDEs in all but one dimension (usually leaving the time dimension undiscretized, and this procedure is usually referred to as the spatial discretization). The resulting system is only continuous in the time dimension and can be viewed as a system of ODEs. A time-integration method is then required to obtain the final numerical solution in fully space-time discretized form. We introduce methods for spatial discretization of systems of time-dependent PDEs in Section 3.3.1 and time-integration methods for systems of ODEs from spatial discretization of bidomain equations in Section 3.3.2. In solving the bidomain model with tissue equations (2.12) and state variables of cell models determined by (2.2), the reaction term $I_{\text{ion}}$ is computed by solving cell models with transmembrane potential $v$ obtained from tissue equations. Updated state variables $\boldsymbol{s}$ are fed back into tissue equations to advance the transmembrane potential $v$ and extracellular potential $u_e$.

We note that we introduce numerical methods by using examples from the bidomain model. In Chapter 4, we use the monodomain model for our numerical experiments. Refer from Section 2.3.2, the monodomain model is a simplified form of the bidomain model. Accordingly, although the results from applying numerical methods to the monodomain model differ from those from the bidomain model, we expect the results from the experiments on the monodomain model to apply equally well to the bidomain model.

### 3.3.1    Spatial discretization of time-dependent parabolic PDEs

We use two simplified examples to explain spatial discretization of time-dependent parabolic PDEs using the finite element method and the finite volume method.

**The finite element method**

To illustrate the spatial discretization for time-dependent parabolic PDEs using the *finite element method* (FEM), we consider the following example [56], where $u(x,t)$ is defined on $\Omega \times [0, +\infty)$:

$$\frac{\partial u}{\partial t} - \nabla^2 u = f(x), \quad x \in \Omega, t > 0, \tag{3.11a}$$

$$u(x,0) = u_0(x), \quad x \in \Omega, \tag{3.11b}$$

$$u = 0, \quad x \in \partial\Omega, \, t > 0. \tag{3.11c}$$

The FEM requires finding a numerical approximation to a weak solution of (3.11). Defining a Hilbert space $V$, the weak solution $u \in V$ of (3.11) is a solution of the following integral equation with boundary conditions (3.11b) and (3.11c):

$$\int_\Omega \frac{\partial u}{\partial t} \phi \, \mathrm{d}\Omega - \int_\Omega \nabla^2 u \, \phi \, \mathrm{d}\Omega = \int_\Omega f\phi \, \mathrm{d}\Omega, \tag{3.12}$$

where $\phi$ is any function in $V$ that satisfy boundary conditions. We can verify the solution of (3.11) satisfies (3.12), and we can prove solution of (3.12) with second-order smoothness also satisfies (3.11) [23].

Applying Stokes's theorem to both sides of (3.12) gives

$$\int_\Omega \frac{\partial u}{\partial t} \phi \, \mathrm{d}\Omega + \int_\Omega \nabla u \cdot \nabla \phi \, \mathrm{d}\Omega - \int_{\partial\Omega} n \cdot \nabla u \, \phi \, \mathrm{d}\Omega = \int_\Omega f \, \phi \, \mathrm{d}\Omega. \tag{3.13}$$

With $\phi$ being any function in $V$ satisfying the boundary conditions, the third term of (3.13) vanishes. We obtain the following variational formulation problem:

$$\text{Find } u \in V \text{ such that} \quad \int_\Omega \frac{\partial u}{\partial t} \phi \, \mathrm{d}\Omega + \int_\Omega \nabla u \cdot \nabla \phi \, \mathrm{d}\Omega = \int_\Omega f \, \phi \, \mathrm{d}\Omega, \quad \text{for any } \phi \in V. \tag{3.14}$$

The solution $u$ is a continuous function; thus the solution space $V$ has infinite dimension. For the purposes of numerical computation, we need to find a solution that is discrete. We achieve this by finding a numerical approximation $u_h$ of $u$ in a finite-dimensional subspace $V_h \subset V$. $V_h$ is a finite-dimensional function space with basis functions $\phi_1, \phi_2, \cdots, \phi_m \in V_h$ defined on a discretized domain $\Omega_h$ of $\Omega$ (domain being divided into non-overlapping sub areas). The solution $u_h \in V_h$ can be expressed as

$$u_h = \sum_{j=1}^{m} u_j \phi_j, \tag{3.15}$$

with $u_j$ being coefficients of basis functions. We then obtain the discretized form of the variational formulation problem (3.14):

$$\text{Find } u_h \in V_h \text{ such that} \quad \int_{\Omega_h} \frac{\partial u_h}{\partial t} \phi_h \, \mathrm{d}\Omega_h + \int_{\Omega_h} \nabla u_h \cdot \nabla \phi_h \, \mathrm{d}\Omega_h = \int_{\Omega_h} f \, \phi_h \, \mathrm{d}\Omega_h, \quad \text{for any } \phi_h \in V_h. \tag{3.16}$$

It is easy to prove the solution $u_h$ satisfies (3.16) if and only if

$$\int_{\Omega_h} \frac{\partial u_h}{\partial t} \phi_i \, \mathrm{d}\Omega_h + \int_{\Omega_h} \nabla u_h \cdot \nabla \phi_i \, \mathrm{d}\Omega_h = \int_{\Omega_h} f \, \phi_i \, \mathrm{d}\Omega_h, \quad \text{for all } \phi_i \in V_h, i = 1, 2, \cdots, m. \tag{3.17}$$

Substituting (3.15) into (3.17) gives us

$$\int_{\Omega_h} \left( \sum_{j=1}^m \frac{\partial u_j}{\partial t} \phi_j \right) \phi_i \, d\Omega_h + \int_{\Omega_h} \nabla \left( \sum_{j=1}^m u_j \, \phi_j \right) \cdot \nabla \phi_i \, d\Omega_h = \int_{\Omega_h} f \, \phi_i \, d\Omega_h, \quad i = 1, 2, \cdots, m.$$

This can be reformulated to

$$\sum_{j=1}^m \frac{\partial u_j}{\partial t} \int_{\Omega_h} \phi_j \, \phi_i \, d\Omega_h + \sum_{j=1}^m u_j \int_{\Omega_h} \nabla \phi_j \cdot \nabla \phi_i \, d\Omega_h = \int_{\Omega_h} f \, \phi_i \, d\Omega_h, \quad i = 1, 2, \cdots, m. \tag{3.18}$$

The basis functions $\phi_i$ are known, so we can write (3.18) into a system of ODEs:

$$\mathbf{M} \frac{d\boldsymbol{u}}{dt} + \mathbf{A}\boldsymbol{u} = \boldsymbol{f}, \tag{3.19}$$

where $\boldsymbol{u} = (u_1, u_2, \ldots, u_m)$, and elements on row $i$ and column $j$ of $\mathbf{A}$, $\mathbf{M}$ and element $i$ of $\boldsymbol{f}$ are given by

$$\mathbf{A}(i,j) = \int_{\Omega_h} \nabla \phi_i \cdot \nabla \phi_j \, d\Omega_h, \quad \mathbf{M}(i,j) = \int_{\Omega_h} \phi_i \, \phi_j \, d\Omega_h, \quad \boldsymbol{f}(i) = \int_{\Omega_h} f \, \phi_i \, d\Omega_h$$

This system is subject to the initial condition given by (3.11b).

The boundary condition (3.11c) can be applied to the linear system (3.19) by constructing $V$ (and $V_h$) accordingly. If (3.11c) is not satisfied, we need to adjust the linear system (3.19). Detailed discussions can be found in [56].

We note that the above process is a general framework of FEM and is independent of choosing the discretization $\Omega_h$ of domain $\Omega$ and basis functions $\phi_i$. The spatial discretization can be chosen based on the shape of the problem domain and characteristics of the problem. The basis functions can be chosen according to the required accuracy. For a one-dimensional problem, a useful yet simple form is the linear basis functions (piece-wise linear functions) on a triagulation of the domain:

$$\phi_j(x_i) = \begin{cases} 1 & \text{if } j = i, \\ 0 & \text{if } j \neq i, \end{cases}$$

where $x_i$ are mesh points. We refer to [56] for a detailed discussion.

Spatial discretization of problem (3.11) using the FEM yields a system of ODEs (3.19). A time-integration method is required to discretize this system in the time dimension and to produce the final solution in fully discretized form.

**The finite volume method**

Another spatial discretization method for time-dependent parabolic PDEs used in this thesis is the *finite volume method* (FVM) [20]. We introduce this method by looking at the following 1D problem.

Consider a time-dependent parabolic PDE $u(x, t)$ defined on $(x, t) \in [0, 1] \times [0, \infty)$:

$$\frac{\partial u}{\partial t} - \nabla(k(x)\nabla u) = f(x, t), \quad 0 < x < 1, \, t > 0, \, k(x) > 0, \tag{3.20a}$$

$$u(x, 0) = u_0(x), \quad 0 < x < 1, \tag{3.20b}$$

$$\frac{\partial u}{\partial x}(0, t) = \frac{\partial u}{\partial x}(1, t) = 0. \tag{3.20c}$$

We first equally divide the interval $[0, 1]$ into $N$ subintervals (called "cells" in FVM terminology) denoted by $I_0, I_1, \ldots, I_{N-1}$. The length $h$ of each cell is $h = 1/N$. So the center $x_j$ of the cell $I_j$ is $x_j = h/2 + jh$. We also denote the left and right endpoint of $I_j$ by $x_{j-1/2}$ and $x_{j+1/2}$, respectively. In the FVM, we intend to approach the solution by approximating the average of the solution over a cell. We denote the approximation to the real solution $u$ as $q$. Then in each cell $I_j$, $q$ follows

$$q_j(t) := q(x_j, t) \approx u_j(t) := \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} u(x, t) \, \mathrm{d}x. \tag{3.21}$$

We integrate equation (3.20a) over cell $I_j$ and take the average:

$$\frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{\partial u}{\partial t}(x, t) \, \mathrm{d}x = \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} \nabla(k\nabla u) \, \mathrm{d}x + \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} f(x, t) \, \mathrm{d}x \tag{3.22a}$$

$$= \frac{k(x_{j+1/2})\nabla u(x_{j+1/2}) - k(x_{j-1/2})\nabla u(x_{j-1/2})}{h} + \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} f(x, t) \, \mathrm{d}x. \tag{3.22b}$$

We define the flux $F_j(t)$ through $I_j$ and corresponding source $f_j$ by

$$F_j(t) := F(x_j, t) = -k(x_j)\nabla u(x_j, t), \tag{3.23}$$

and

$$f_j(t) = \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} f(x, t) \, \mathrm{d}x.$$

Based on (3.22), we get the following equation for $u_j$ in every cell $j$:

$$\frac{\mathrm{d}u_j(t)}{\mathrm{d}t} = -\frac{F_{j+1/2}(t) - F_{j-1/2}(t)}{h} + f_j(t). \tag{3.24}$$

We now approximate the flux (3.23) using a second-order central difference scheme:

$$F_{j-1/2}(t) = -k(x_{j-1/2})\nabla u(x_{j-1/2}, t) \tag{3.25a}$$

$$= -k(x_{j-1/2})\frac{u_j(t) - u_{j-1}(t)}{h} + O(h^2). \tag{3.25b}$$

Substituting (3.25) into (3.24) and omitting the second-order term, we have the approximation $q_j(t)$ of $u_j(t)$ at midpoint $x_j$ of $I_j$ that satisfies

$$\frac{\mathrm{d}q_j(t)}{\mathrm{d}t} = \frac{k_{j+1/2}q_{j+1} - (k_{j+1/2} + k_{j-1/2})q_j + k_{j-1/2}q_{j-1}}{h^2} + f_j(t), \tag{3.26}$$

where $k_j := k(x_j)$, $q_j := q_j(t)$, and $j = 1, 2, \ldots, N - 2$.

We now consider boundary condition (3.20c). We extend the solution to $x < 0$ by adding a cell $I_{-1} = [-1/N, 0]$, and we denote the midpoint of this interval with $x_{-1}$. The boundary condition (3.20c) implies

$$0 = \nabla u(t, 0) = \frac{u_0(t) - u_{-1}(t)}{h} + O(h^2).$$

We thus have

$$u_{-1}(t) = u_0(t) + O(h^3).$$

The approximation $q_j$ satisfies

$$q_{-1}(t) = q_0(t).$$

Then for $j = 0$, we have

$$\frac{\mathrm{d}q_0}{\mathrm{d}t} = \frac{k_{1/2}q_1 - (k_{1/2} + k_{-1/2}q_0 + k_{-1/2}q_{-1})}{h^2} + f_0 = k_{1/2}\frac{q_1 - q_0}{h^2} + f_0. \tag{3.27}$$

Following the derivation of (3.27), we can also derive

$$\frac{\mathrm{d}q_{N-1}}{\mathrm{d}t} = k_{N-3/2}\frac{q_{N-2} - q_{N-1}}{h^2} + f_{N-1}. \tag{3.28}$$

Putting (3.26), (3.27), and (3.28) together, we have a system of ODEs of $\boldsymbol{q}$:

$$\frac{\mathrm{d}\boldsymbol{q}(t)}{\mathrm{d}t} = \mathbf{A}\boldsymbol{q}(t) + \boldsymbol{F}(t), \tag{3.29}$$

where

$$\boldsymbol{q} = \begin{pmatrix} q_0 \\ \vdots \\ q_{N-1} \end{pmatrix}, \quad \boldsymbol{F} = \begin{pmatrix} f_0 \\ \vdots \\ f_{N-1} \end{pmatrix}, \quad \boldsymbol{q}, \boldsymbol{F} \in \mathbb{R},$$

and

$$\boldsymbol{A} = \frac{1}{h^2} \begin{pmatrix} -k_{1/2} & k_{1/2} & & & \\ k_{1/2} & -(k_{1/2} + k_{3/2}) & k_{3/2} & & \\ & \ddots & \ddots & & \ddots & \\ & & k_{N-5/2} & -(k_{N-5/2} + k_{N-3/2}) & k_{N-3/2} \\ & & & -k_{N-3/2} & k_{N-3/2} \end{pmatrix}.$$

The initial condition of this system of ODEs is derived from (3.20b):

$$q_j(0) = u_0(h/2 + jh).$$

Similar to Section 3.3.1, spatial discretization of a time-dependent parabolic PDE system using the FVM results in a system of ODEs. We need time-integration methods for systems of ODEs to obtain a fully discretized numerical solution. Such methods targeted at the spatially discretized bidomain/monodomain model are introduced in Section 3.3.2.

### 3.3.2 Time-integration methods for the bidomain model

The spatial discretization of bidomain equations (2.12) results in systems of ODEs at every space point in the problem domain. We need a time-integration method to obtain the fully discretized form to get numerical solutions at each space-time point.

**The semi–implicit method**

Originally proposed by Whiteley [62] in 2006, the *semi–implicit* (SI) method is used in some cardiac simulation toolkits as default, for example, `CHASTE` and `FEniCS` [39].

At spatial mesh point $\boldsymbol{x}$ and time $t \in [t_n, t_{n+1}]$, the SI method applies the BE method to (2.12) but with $I_{\text{ion}}$ evaluated at $t_n$ instead of $t_{n+1}$ and applies the FE method to (2.2). With $\Delta t$ being time step, the SI method is written by

$$\frac{\boldsymbol{s}_{n+1} - \boldsymbol{s}_n}{\Delta t} = \boldsymbol{f}(\boldsymbol{s}_n, v_n, t_n),$$

$$\chi C_m \frac{v_{n+1} - v_n}{\Delta t} + \chi I_{\text{ion}}(\boldsymbol{s}_{n+1}, v_n, t_n) + I_{\text{stim}}(\boldsymbol{x}, t) = \nabla \cdot (\sigma_i \nabla v_{n+1}) + \nabla \cdot (\sigma_i \nabla u_{e,n+1}),$$

$$0 = \nabla \cdot (\sigma_i \nabla v_{n+1}) + \nabla \cdot ((\sigma_i + \sigma_e) \nabla u_{e,n+1}).$$

As an example, we consider applying the SI method after spatial discretization of the bidomain model using the FEM to obtain a fully discretized form [53]:

$$\begin{pmatrix} \frac{\chi C_m}{\Delta t} \mathbf{A} + \mathbf{A}_i & \mathbf{A}_i \\ \mathbf{A}_i & \mathbf{A}_{i+e} \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_{n+1} \\ \boldsymbol{u}_{e,n+1} \end{pmatrix} = \begin{pmatrix} \frac{\chi C_m}{\Delta t} \mathbf{A} \boldsymbol{v}_n - \chi \mathbf{A} \boldsymbol{I}_{\text{ion}}(\boldsymbol{S}_{n+1}, \boldsymbol{v}_n, t_n) \\ 0 \end{pmatrix},$$

where $\boldsymbol{S}$, $\boldsymbol{v}$, and $\boldsymbol{u}_e$ are vectors of values of $\boldsymbol{s}$, $v$, and $u_e$ in spatially discretized domain; $\mathbf{A}$, $\mathbf{A}_i$, and $\mathbf{A}_{i+e}$ are matrices with elements on row $i$ and column $j$ given by

$$\mathbf{A}(i, j) = \int_\Omega \phi_i \phi_j \, \mathrm{d}\Omega,$$

$$\mathbf{A}_i(i, j) = \int_\Omega \sigma_i \nabla \phi_i \cdot \nabla \phi_j \, \mathrm{d}\Omega,$$

$$\mathbf{A}_{i+e}(i, j) = \int_\Omega (\sigma_i + \sigma_e) \nabla \phi_i \cdot \nabla \phi_j \, \mathrm{d}\Omega,$$

where $i_{max} \cdot j_{max} = M$, $i_{max}$ and $j_{max}$ are maximum values of row and column indices, $M$ is the number of mesh points, and $\phi$ are the finite element basis functions.

**The Godunov operator-splitting method**

When applying to the bidomain problem, Spiteri and Torabi tested the *Godunov operator-splitting* (OS) method [53], which takes the following form:

$$\frac{\boldsymbol{s}_{n+1} - \boldsymbol{s}_n}{\Delta t} = \boldsymbol{f}(\boldsymbol{s}_n, v_n, t_n), \tag{3.31a}$$

$$\frac{\hat{v}_{n+1} - v_n}{\Delta t} = -\frac{1}{C_m} I_{\text{ion}}(\boldsymbol{s}_n, v_n, t_n), \tag{3.31b}$$

$$\chi C_m \frac{v_{n+1} - \hat{v}_{n+1}}{\Delta t} + I_{\text{stim}}(\boldsymbol{x}, t) = \nabla \cdot (\sigma_i \nabla v_{n+1}) + \nabla \cdot (\sigma_i \nabla u_{e,n+1}), \tag{3.31c}$$

$$0 = \nabla \cdot (\sigma_i \nabla v_{n+1}) + \nabla \cdot ((\sigma_i + \sigma_e) \nabla u_{e,n+1}). \tag{3.31d}$$

The Godunov OS method couples the $I_{\text{ion}}$ term from (2.12a) with systems of ODEs of cell models (2.2) to form a new system of ODEs. The new system is solved using the FE method to obtain an interim solution $\hat{v}_{n+1}$. The interim solution from (3.31b) is substituted into the tissue equations (3.31c) to obtain $v_{n+1}$ by applying the BE method. Numerical experiments from Spiteri and Torabi have shown that the Godunov OS method permits us to take a larger time step size than the SI method but still satisfies a prescribed numerical accuracy widely used in related researches (the numerical accuracy is given by 5% mixed root mean square error, which is explained in Section 4.1). Because the cost per step of the Godunov OS method is comparable to that of the SI method, Spiteri and Torabi propose that in solving bidomain problems, the Godunov OS method should generally be favoured over the SI method given the above accuracy requirement.

# CHAPTER 4

# NUMERICAL EXPERIMENTS

This chapter presents numerical experiments showing applications of the theory introduced in Chapter 3. In Section 4.1, we introduce how we determine the quality of the numerical solutions. In Section 4.2, we extend research in [33] by implementing more numerical methods (specifically, $\mathrm{RKC}(s,1)$ methods and $\mathrm{RLC}(s,1)$ methods where $s = 2, 3$) to the time integration of cardiac cell models. We compare their efficiency by specifying a numerical accuracy. We also test the $\mathrm{RKC}(2,1)$ method by applying it to the time integration of cardiac cell models in a pre-defined monodomain problem. In Section 4.3, we consider the application of the $\mathrm{RKC}(2,1)$ method to the time integration of a spatially discretized monodomain tissue equation using the FVM. In Section 4.4, we give a quantitative analysis of the improvement of computational efficiency attributed to numerical stability from proposed numerical methods.

## 4.1   Numerical accuracy

To determine the accuracy of numerical solutions produced by different numerical methods applied to the same problem, we usually generate a reference solution that serves as a gold standard for numerical solutions to compare against. The reference solution is generated by comparing a sequence of solutions by taking decreasing time steps and increasing spatial resolutions (if applicable) until they have sufficiently converged. Sometimes the Richardson extrapolation [49] is used to obtain higher-order accuracy.

To compare numerical solutions with the reference solution, we use the *mixed root mean square* (MRMS) error [33] given by

$$[\mathrm{MRMS}]_y = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left( \frac{\bar{y}_i - y_i}{1 + |\bar{y}_i|} \right)^2},$$

where $\bar{y}_i$ and $y_i$ denote the reference solution and the numerical solution at space-time point $i$, respectively, and $N$ denotes the number of solution points. If the reference solution has more solution points than the numerical solution, we sample the reference solution at respective solution points from the numerical solution; otherwise we sample only at those points of the reference solution. Linear interpolation is also used when necessary.

The MRMS error was chosen to provide a pointwise global error measure in space-time that is sensitive to large-scale pointwise errors. In the experiments outlined in Section 4.2.1 and 4.2.2, we treat solutions with less than 5% MRMS error as reliable. We chose a 5% MRMS error threshold because this figure represents a

balance between a faithful solution to the mathematical model and uncertainties in the model. The 5% MRMS error has been used in other cardiac simulation research related to this thesis, such as [52] and [53]. Recent research tends to favour a lower MRMS error, such as 1%, which may provide more confidence than higher errors in the electrophysiology model. We refer to [13] and [28] as examples.

## 4.2   Time integration of cardiac cell models

This section presents numerical experiments on different numerical methods applied to the time integration of cardiac cell models. In Section 4.2.1, we apply different numerical methods to the time integration of individual cardiac cell models. In Section 4.2.2, we apply the $\text{RKC}(2,1)$ method to the time integration of cell models in a monodomain problem.

### 4.2.1   Individual cell simulations

This experiment is an extension of the study [33]. The cell models selected by [33] are from the `CellML` project [1], which is an online repository for mathematical models of cellular biological functions. A brief summary of the cell models used is presented in Table 4.1. The cell models we have chosen are based on previous studies in [33] and cover a range of stiffness, from the non-stiff model of Noble (1962) [40], to the moderately stiff model of Courtemanche et al. [7], to the stiff model of Pandit et al. (2003) [47]. The cell models used in this thesis are considered to be "validated" according to specifications provided by the `CellML` project — cell models that have successfully reproduced published results and are compatible with certain environments (e.g., the OpenCell [1]) working with the `CellML` code.

We perform time integration on the aforementioned cardiac cell models using various first-order numerical methods implemented in `MATLAB`. The methods are the FE method, the RL method, the GRL1 method, the $\text{RKC}(2,1)$ method, the $\text{RKC}(3,1)$ method, the $\text{RLC}(2,1)$ method, and the $\text{RLC}(3,1)$ method. The multistage RKC methods and RKC methods in RLC methods use a damping parameter of 0.05. For each cell model, we report the largest step sizes that produce solutions with less than 5% MRMS error and their corresponding execution times. The reference solutions are generated using the `MATLAB` built-in ODE solver `ode15s`. We compute solutions with decreasing tolerances until the solutions have seven to ten matching digits at 100 equally spaced points spanned over the targeted integration interval. We then pick the solution from using the smallest step size as the reference solution.

The experiments are performed on a personal computer with an Intel Quad core i7 processor (with hyper-threading turned on) and 16 GB memory running Ubuntu Linux 16.04 LTS. We turn on the `MATLAB` profiler during the experiment because we find this eliminates unwanted optimizations introduced by `MATLAB`. We report execution times as the minimum of 20 runs. We find our results to be slightly different from the previous study [33], but we believe our results are more reliable because of the elimination of unwanted optimization from `MATLAB`.

**Table 4.1:** Details on 37 validated cardiac cell models [32]. Models labeled with an asterisk (*) indicate three cell variants (endocardial cell, epicardial cell, and M-cell). The model referred as Winslow31 is a reduced model with 31 variables compared to its original form.

| Model | Reference | Number of variables | Description |
|---|---|---|---|
| Beeler–Reuter (1977) | [3] | 8 | Mammalian ventricular model |
| Bondarenko et al. (2004) | [4] | 41 | Mouse ventricular model |
| Courtemanche et al. (1998) | [7] | 21 | Human atrial model |
| Demir et al. (1994) | [10] | 27 | Rabbit sinoatrial node model |
| Demir et al. (1999) | [9] | 29 | Rabbit sinoatrial node model |
| DiFrancesco–Noble (1985) | [11] | 16 | Mammal Purkinje fibre model |
| Dokos et al. (1996) | [12] | 18 | Rabbit sinoatrial node model |
| Faber–Rudy (2000) | [14] | 19 | Guinea pig ventricular model |
| FitzHugh–Nagumo (1961) | [15, 38] | 2 | Nerve membrane model |
| Fox et al. (2002) | [16] | 13 | Canine ventricular model |
| Hilgemann–Noble (1987) | [18] | 15 | Rabbit atrial model |
| Hund–Rudy (2004) | [19] | 29 | Canine ventricular model |
| Jafri et al. (1998) | [22] | 31 | Guinea pig ventricular model |
| Luo–Rudy (1991) | [29] | 8 | Guinea pig ventricular model |
| Maleckar et al. (2008) | [31] | 30 | Human atrial model |
| McAllister et al. (1975) | [34] | 10 | Canine Purkinje fibre model |
| Noble (1962) | [40] | 4 | Mammal Purkinje fibre model |
| Noble–Noble (1984) | [41] | 15 | Rabbit sinoatrial node model |
| Noble et al. (1991) | [42] | 17 | Guinea pig ventricular model |
| Noble et al. (1998) | [43] | 22 | Guinea pig ventricular model |
| Nygren et al. (1998) | [44] | 29 | Human atrial model |
| Pandit et al. (2001) | [46] | 26 | Rat left-ventricular model |
| Pandit et al. (2003) | [47] | 26 | Rat left-ventricular model |
| Puglisi–Bers (2001) | [48] | 17 | Rabbit ventricular model |
| Sakmann et al. (2000)* | [51] | 21 | Guinea pig ventricular model |
| Stewart et al. (2009) | [54] | 20 | Human Purkinje fibre model |
| Ten Tusscher et al. (2004)* | [57] | 17 | Human ventricular model |
| Ten Tusscher et al. (2006)* | [58] | 19 | Human ventricular model |
| Wang–Sobie (2008) | [61] | 35 | Neonatal mouse ventricular model |
| Winslow31 (1999) | [63] | 31 | Canine ventricular model |
| Zhang et al. (2000) | [64] | 15 | Rabbit sinoatrial node model |

We report step sizes and corresponding execution times of numerical methods applied to the cell models in Table 4.2. We summarize the number of cell models according to which numerical method takes the smallest execution time in this experiment and previous research [33] in Figure 4.1. We note that previous research [33] does not introduce multistage RKC and RLC methods ($RKC(2,1)$ and $RKC(3,1)$, $RLC(2,1)$, and $RLC(3,1)$).
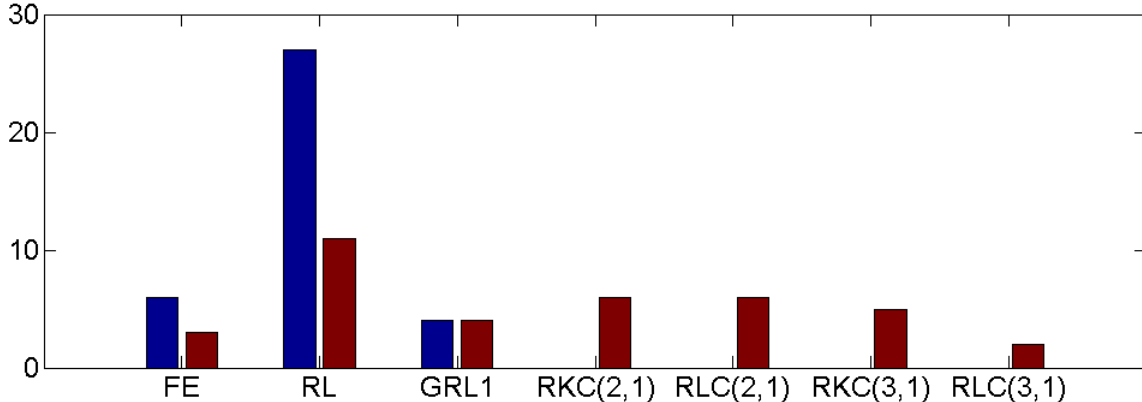


**Figure 4.1:** Counts on cell models out of 37 that a specific numerical method, while specifying solution less than 5% MRMS error, takes the smallest execution in the study from previous research [33] (blue) and in this study (red).

We make several conclusions based on the above experiment:

1. Referring to Figure 4.1, the RL method still wins (takes the smallest execution time) on the majority of the 37 cell models. This echoes the argument presented in [33] that because the selected models are mostly non-stiff or moderately stiff for a 5% MRMS error, and the stiff parts are captured by gating variables that are integrated using an exponential integrator, the RL method is the most efficient method in dealing with those cell models.

2. Referring to Table 4.2 and Figure 4.1, we note that, for all models considered, multistage RKC methods increase the largest step sizes, hence reduced the total computing steps for obtaining numerical solutions with less than 5% MRMS error. However, additional computing stages bring extra computing costs each step. The trade-off between reduced computing steps and increased computing costs each step determines the optimal numerical method for a cell model. For example, considering the model of Demir et al. (1994) [10], the $RKC(2,1)$ method takes step size three times larger than that of the FE method. Because computing costs each step for the $RKC(2,1)$ method are roughly twice that that of the FE method, the $RKC(2,1)$ method beats the FE method by reducing execution time by a factor of 1.6. We expect that the $RKC(2,1)$ method to outperforms on cell models where it takes step size over two times larger than that of the FE method; however, introducing one more stage (the $RKC(3,1)$ method) only slightly increases the step size in solving this model but increases computing costs by a

**Table 4.2:** Step sizes and corresponding execution times for different numerical methods applying to various cell models. For each model, the largest step size ($\Delta t$, in milliseconds) producing solution below 5% MRMS error is marked with a dagger ($^\dagger$), and the smallest execution time (CPU time in seconds) is in a shaded table cell. Note that the model of FitzHugh–Nagumo (1961) [15, 38] does not have gating variables; thus it is not feasible to apply RLC methods (RL, RLC(2,1), and RLC(3,1)).

| Model | FE | | RL | | GRL1 | | RKC(2,1) | | RLC(2,1) | | RKC(3,1) | | RLC(3,1) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\Delta t$ (ms) | Time (s) | $\Delta t$ (ms) | Time (s) | $\Delta t$ (ms) | Time (s) | $\Delta t$ (ms) | Time (s) | $\Delta t$ (ms) | Time (s) | $\Delta t$ (ms) | Time (s) | $\Delta t$ (ms) | Time (s) |
| Beeler–Reuler (1977) | 2.53e-02 | 6.88e-01 | 6.78e-01 | 2.94e-02 | 8.08e-01 | 4.71e-02 | 9.87e-02 | 2.48e-01 | 7.90e-01 | 2.89e-02 | 2.14e-01 | 1.60e-01 | 8.34e-01$^\dagger$ | 3.62e-02 |
| Bondarenko et al. (2004) | 2.13e-04 | 4.06e+01 | 2.13e-04 | 4.17e+01 | 7.47e-03$^\dagger$ | 4.19e+00 | 8.34e-04 | 2.02e+01 | 8.33e-04 | 1.96e+01 | 1.86e-03 | 1.44e+01 | 1.85e-03 | 1.36e+01 |
| Courtemanche et al. (1998) | 1.94e-02 | 2.63e+00 | 8.37e-02 | 5.77e-01 | 9.60e-02 | 1.41e+00 | 7.97e-02 | 1.25e+00 | 8.93e-02 | 9.03e-01 | 1.63e-01$^\dagger$ | 9.19e-01 | 1.01e-01 | 1.10e+00 |
| Demir et al. (1994) | 5.95e-05 | 2.85e-01 | 5.32e-05 | 3.06e-01 | 1.18e-04 | 4.92e-01 | 1.96e-04 | 1.72e-01 | 5.71e-05 | 8.93e-01 | 2.10e-04$^\dagger$ | 2.29e-01 | 5.79e-05 | 7.18e-01 |
| Demir et al. (1999) | 5.96e-05 | 3.21e-01 | 4.73e-05 | 4.46e-01 | 9.99e-05 | 6.67e-01 | 1.66e-04 | 2.35e-01 | 5.10e-05 | 6.83e-01 | 1.78e-04$^\dagger$ | 3.32e-01 | 5.16e-05 | 9.45e-01 |
| DiFrancesco–Noble (1985) | 7.73e-05 | 1.47e+00 | 1.95e-04 | 6.49e-01 | 2.07e-04 | 1.66e+00 | 2.99e-04 | 7.25e-01 | 2.27e-04 | 6.65e-01 | 5.34e-04$^\dagger$ | 5.54e-01 | 2.34e-04 | 1.10e+00 |
| Dokos et al. (1996) | 7.02e-05 | 4.16e-01 | 1.22e-04 | 2.57e-01 | 8.02e-05 | 1.24e+00 | 2.70e-04 | 2.05e-01 | 1.33e-04 | 3.01e-01 | 4.39e-04$^\dagger$ | 1.79e-01 | 1.90e-04 | 3.75e-01 |
| FitzHugh–Nagumo (1961) | 2.72e-03 | 1.49e+00 | NA | NA | 2.60e-03 | 3.04e+00 | 5.55e-03 | 9.18e-01 | NA | NA | 8.54e-03$^\dagger$ | 7.20e-01 | NA | NA |
| Faber–Rudy (2000) | 1.12e-02 | 2.90e+00 | 2.01e-02 | 1.64e+00 | 4.06e-02 | 2.46e+00 | 4.04e-02 | 1.57e+00 | 2.35e-02 | 2.34e+00 | 4.22e-02$^\dagger$ | 2.18e+00 | 2.43e-02 | 2.98e+00 |
| Fox et al. (2002) | 4.62e-03 | 4.61e+00 | 4.73e-02 | 4.85e-01 | 1.16e-01$^\dagger$ | 4.86e-01 | 1.80e-02 | 1.72e+00 | 8.23e-02 | 3.36e-01 | 4.02e-02 | 1.35e+00 | 9.79e-02 | 4.73e-01 |
| Hilgemann–Noble (1987) | 6.25e-05 | 4.77e-01 | 8.17e-05 | 7.53e-01 | 1.52e-04 | 6.18e-01 | 2.44e-04 | 2.02e-01 | 2.93e-04 | 3.91e-01 | 4.22e-04$^\dagger$ | 1.86e-01 | 2.97e-04 | 3.18e-01 |
| Hund–Rudy (2004) | 7.80e-03 | 4.44e+00 | 5.03e-03 | 7.19e+00 | 5.47e-03 | 2.14e+01 | 1.06e-02 | 6.46e+00 | 5.20e-03 | 1.16e+01 | 1.13e-02$^\dagger$ | 9.17e+00 | 6.10e-03 | 1.33e+01 |
| Jafri et al. (1998) | 5.76e-04 | 5.47e+01 | 5.77e-04 | 6.11e+01 | 1.41e-03 | 7.84e+01 | 1.75e-03 | 3.49e+01 | 1.76e-03 | 3.38e+01 | 2.42e-03 | 3.86e+01 | 2.44e-03$^\dagger$ | 3.64e+01 |
| Luo–Rudy (1991) | 1.35e-02 | 1.50e+00 | 1.23e-01 | 1.80e-01 | 3.15e-01$^\dagger$ | 1.40e-01 | 5.34e-02 | 5.81e-01 | 1.80e-01 | 1.53e-01 | 1.18e-01 | 3.67e-01 | 1.80e-01 | 2.04e-01 |
| Maleckar et al. (2008) | 5.02e-05 | 1.91e+00 | 8.90e-05 | 1.36e+00 | 4.20e-04 | 7.57e-01 | 1.90e-04 | 9.78e-01 | 3.48e-04 | 5.73e-01 | 4.03e-04 | 6.60e-01 | 7.77e-04$^\dagger$ | 3.80e-01 |
| McAllister et al. (1975) | 2.47e-02 | 1.17e+00 | 4.69e-01$^\dagger$ | 7.21e-02 | 2.53e-01 | 2.96e-01 | 9.60e-02 | 4.44e-01 | 3.53e-01 | 1.08e-01 | 9.66e-02 | 6.37e-01 | 3.76e-01 | 1.26e-01 |
| Noble (1962) | 2.02e-04 | 5.28e-02 | 1.47e-04 | 7.97e-02 | 1.10e-04 | 2.13e-01 | 5.32e-04 | 2.66e-02 | 1.56e-04 | 8.42e-02 | 5.73e-04$^\dagger$ | 3.27e-02 | 1.58e-04 | 1.03e-01 |
| Noble–Noble (1984) | 2.04e-04 | 1.11e-01 | 1.21e-04 | 2.04e-01 | 9.27e-05 | 6.98e-01 | 3.71e-04 | 8.98e-02 | 1.36e-04 | 2.14e-01 | 3.97e-04$^\dagger$ | 1.49e-01 | 1.40e-04 | 3.60e-01 |
| Noble et al. (1991) | 5.15e-05 | 4.62e-01 | 1.53e-04 | 1.70e-01 | 1.04e-04 | 6.53e-01 | 2.01e-04 | 2.18e-01 | 6.06e-04$^\dagger$ | 1.01e-01 | 4.50e-04 | 1.40e-01 | 6.38e-04 | 8.72e-02 |
| Noble et al. (1998) | 5.56e-05 | 8.46e-01 | 1.57e-04 | 3.30e-01 | 8.86e-05 | 1.61e+00 | 1.82e-04 | 5.02e-01 | 5.61e-04 | 1.52e-01 | 1.87e-04 | 7.12e-01 | 5.68e-04$^\dagger$ | 1.98e-01 |
| Nygren et al. (1998) | 5.36e-05 | 1.63e+00 | 8.88e-05 | 1.04e+00 | 2.06e-04 | 1.37e+00 | 2.03e-04 | 8.40e-01 | 2.85e-04 | 5.15e-01 | 4.54e-04$^\dagger$ | 5.31e-01 | 2.85e-04 | 7.45e-01 |
| Pandit et al. (2001) | 2.91e-07 | 8.28e+01 | 2.91e-07 | 8.51e+01 | 2.40e-05$^\dagger$ | 3.06e+00 | 1.13e-06 | 4.00e+01 | 1.12e-06 | 3.49e+01 | 2.54e-06 | 2.49e+01 | 2.54e-06 | 2.17e+01 |
| Pandit et al. (2003) | 2.65e-08 | 9.71e+02 | 2.65e-08 | 1.00e+03 | 1.57e-05$^\dagger$ | 4.67e+00 | 1.03e-07 | 4.77e+02 | 1.03e-07 | 4.18e+02 | 2.32e-07 | 3.24e+02 | 2.32e-07 | 2.55e+02 |
| Puglisi–Bers (2001) | 1.08e-02 | 3.73e+00 | 1.45e-02 | 3.02e+00 | 3.27e-02$^\dagger$ | 3.11e+00 | 2.28e-02 | 3.04e+00 | 1.69e-02 | 4.40e+00 | 2.50e-02 | 3.95e+00 | 1.95e-02 | 5.01e+00 |
| Sakmann et al. (2000) – Endocardial | 6.90e-05 | 8.13e-01 | 4.99e-05 | 1.23e+00 | 4.16e-05 | 4.05e+00 | 1.10e-04 | 9.89e-01 | 5.49e-05 | 1.83e+00 | 1.22e-04$^\dagger$ | 1.32e+00 | 5.70e-05 | 2.49e+00 |
| Sakmann et al. (2000) – Epicardial | 6.80e-05 | 8.27e-01 | 4.16e-05 | 1.45e+00 | 3.83e-05 | 4.41e+00 | 1.02e-04 | 1.07e+00 | 4.09e-05 | 2.48e+00 | 1.08e-04$^\dagger$ | 1.47e+00 | 4.42e-05 | 3.46e+00 |
| Sakmann et al. (2000) – M-cell | 6.86e-05 | 8.22e-01 | 2.32e-04 | 2.61e-01 | 4.21e-04$^\dagger$ | 4.04e-01 | 2.52e-04 | 6.84e-01 | 3.41e-04 | 2.98e-01 | 2.55e-04 | 1.01e+00 | 3.46e-04 | 4.16e-01 |
| Stewart et al. (2009) | 1.52e-02 | 7.05e+00 | 2.05e-01 | 5.60e-01 | 1.74e-01 | 1.89e+00 | 5.90e-02 | 3.54e+00 | 2.11e-01 | 7.89e-01 | 1.31e-01 | 2.18e+00 | 2.20e-01$^\dagger$ | 1.04e+00 |
| Ten Tusscher et al. (2004) –Endocardial | 1.78e-03 | 2.91e+01 | 1.25e-01 | 4.20e-01 | 1.37e-01 | 1.08e+00 | 6.98e-03 | 1.43e+01 | 1.46e-01 | 4.25e-01 | 1.55e-02 | 8.80e+00 | 1.51e-01$^\dagger$ | 7.15e-01 |
| Ten Tusscher et al. (2006) –Endocardial | 1.62e-03 | 2.22e+01 | 7.03e-02 | 5.22e-01 | 1.29e-01$^\dagger$ | 8.19e-01 | 6.33e-03 | 1.09e+01 | 8.97e-02 | 4.96e-01 | 1.42e-02 | 6.73e+00 | 9.34e-02 | 8.32e-01 |
| Ten Tusscher et al. (2004) –Epicardial | 1.78e-03 | 2.94e+01 | 1.11e-01 | 4.75e-01 | 1.19e-01 | 1.25e+00 | 6.98e-03 | 1.41e+01 | 1.29e-01 | 4.80e-01 | 1.55e-02 | 8.84e+00 | 1.33e-01$^\dagger$ | 7.93e-01 |
| Ten Tusscher et al. (2006) –Epicardial | 2.14e-03 | 1.70e+01 | 1.16e-01 | 3.17e-01 | 1.75e-01$^\dagger$ | 6.05e-01 | 8.36e-03 | 8.33e+00 | 1.45e-01 | 3.92e-01 | 1.87e-02 | 5.12e+00 | 1.49e-01 | 5.12e-01 |
| Ten Tusscher et al. (2004) –M-cell | 1.76e-03 | 2.19e+01 | 1.21e-01 | 3.34e-01 | 1.02e-01 | 1.09e+00 | 6.91e-03 | 1.07e+01 | 1.36e-01$^\dagger$ | 3.52e-01 | 1.54e-02 | 6.76e+00 | 1.29e-01 | 6.39e-01 |
| Ten Tusscher et al. (2006) –M-cell | 2.06e-03 | 1.75e+01 | 1.27e-01 | 2.90e-01 | 1.38e-01 | 7.66e-01 | 8.05e-03 | 8.63e+00 | 1.52e-01 | 3.74e-01 | 1.80e-02 | 5.32e+00 | 1.57e-01$^\dagger$ | 4.96e-01 |
| Wang–Sobie (2008) | 1.66e-02 | 1.01e+00 | 5.27e-02 | 3.47e-01 | 9.36e-02 | 5.93e-01 | 6.48e-02 | 5.01e-01 | 1.05e-01 | 2.89e-01 | 1.31e-01$^\dagger$ | 3.85e-01 | 1.11e-01 | 4.05e-01 |
| Winslow et al. (1999) (31 equations) | 1.07e-04 | 3.45e+02 | 1.07e-04 | 3.53e+02 | 9.38e-05 | 1.32e+03 | 4.14e-04 | 1.76e+02 | 4.15e-04 | 1.63e+02 | 9.25e-04 | 1.27e+02 | 9.35e-04$^\dagger$ | 1.06e+02 |
| Zhang et al. (2000) | 9.97e-05 | 8.09e-01 | 4.57e-04 | 1.77e-01 | 3.04e-04 | 6.79e-01 | 3.72e-04 | 4.07e-01 | 4.77e-04 | 2.05e-01 | 8.45e-04$^\dagger$ | 2.49e-01 | 4.81e-04 | 3.42e-01 |

factor of 1.5 compared with the RKC(2, 1) method; thus the RKC(2, 1) method beats the RKC(3, 1) method and the FE method in solving this model.

3. The above analysis also applies to the comparison of step sizes and execution times for the RL, RLC(2, 1), and RLC(3, 1) methods, but there are several outliers. Considering the computing workload, a typical example is the model of Maleckar et al. [31]. We note that in multistage RLC(s, 1) methods, replacing the FE method with multistage RKC methods also helps to increase step size producing solutions with less than 5% MRMS error. By treating gating and non-gating variables with different numerical methods, the RL method has been successful in efficiently solving cardiac cell models. We note that such a replacement enhances ability of multistage RLC(s, 1) methods to outperform on more models. In Section 4.4, we further show the stability improvement from treating non-gating variables with multistage RKC methods and give a quantitative explanation of this improvement. We note that the computing workload of multistage RLC(s, 1) methods is not necessarily s times greater than that of the RL method. This is because the percentage of variables that are treated by exponential integrator affects this difference.

4. The RLC(2, 1) and RLC(3, 1) methods do not always permit a larger step size specifying 5% MRMS error. As an example, consider the model of McAllister et al. [34]. In this model, the RL method takes a larger step size than that of the RLC(2, 1) method. As we mentioned above, the stability region of the RKC(2, 1) method is only partially included in that of the FE method. Thus, because the locations of scaled eigenvalues vary to different cell models, there is no theoretical guarantee that multistage RKC methods permit increased step sizes for producing numerical solutions that meet a prescribed numerical accuracy.

5. Referring to Figure 4.1 and comparing the results of this thesis with those from the study in [33], the GRL1 method is still suitable for solving the stiffest cell models. We can conclude multistage RLC methods and multistage RKC methods are rather incapable of handling stiff problems compared to the exponential integrator applied to locally linearized system. However, we should note that multistage RKC methods mostly increase step sizes for solving non-stiff or moderately stiff problems. By justifying the trade-off between increased step sizes and extra computing costs, we are able to choose suitable numerical methods for certain cardiac cell models.

### 4.2.2 Applications in the monodomain simulation

We consider applying the RKC(2, 1) method to the time integration of ODE systems from cell models in a monodomain simulation. This experiment is taken on the same computer described in Section 4.2. CHASTE 3.4 is used, and the cell model we use is from the CellML repository.

We set up a monodomain problem on an interval (from 0 cm to 1.0 cm) for 20 ms in CHASTE, and we couple the Hilgemann–Noble model [18] with the monodomain tissue equation (2.15). We apply an artificial

stimulus to the cells by setting transmembrane potential $v$ at each spatial point $x$ to $v + 100 \cdot (1 - \sin(x))$. To evaluate the accuracies of our numerical solutions, we generate a reference solution for comparison with decreasing time steps and increasing spatial resolutions until we have two solutions that matched to four decimal places. We visualize our reference solution in Figure 4.2. The numerical solutions are not visually distinct from the reference solution so we omit visualizations of numerical solutions. The visualization shows a tail of propagation of a traveling wave on this 1D interval; thus we can justify our problem has a non-trivial solution.
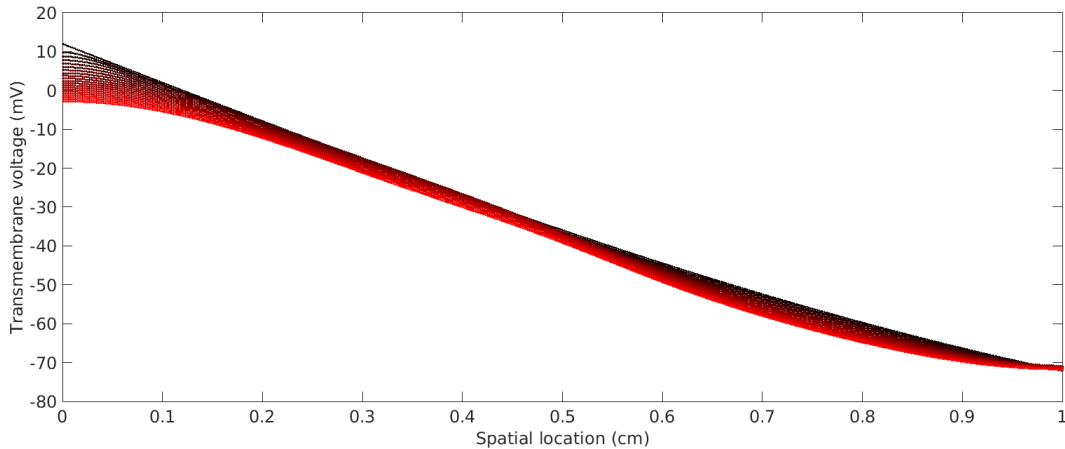


**Figure 4.2:** Visualization of the reference solution for a monodomain problem incorporating the model of Hilgemann–Noble (1987), showing solutions from $0\,\mathrm{ms}$ (black) to $20\,\mathrm{ms}$ (red).

We apply the FE method and the RKC(2, 1) method, respectively, to time integration of cell models in the monodomain problem. The monodomain problem is solved by using the SI method by default in `CHASTE` with FEM for spatial discretization. We keep the time step for time integration of cell models the same as that for solving the tissue equation in monodomain problem; because in `CHASTE` substepping on tissue equation is not permitted. We further find tissue equation contributes most to the error in monodomain simulation. We run the simulations in serial with using either the FE method or the RKC(2, 1) method for time integration of cell model, and we record the largest step sizes we can take that produces solution with less than 5% MRMS error for both using the FE or the RKC(2, 1) method, their corresponding % MRMS errors, the total execution times, and times spent on integrating cell equations in Table 4.3.

Referring to Table 4.3, we note that we have sped up the solving process by a factor of 1.4. For this problem, the time integration of cell model takes a large percentage of the monodomain simulation. We note that from Section 4.2.1, the step size from using RKC(2, 1) method to solve the Hilgemann–Noble model is more than two times than that used by FE method to satisfy prescribed numerical accuracy. Because cell solving takes most percentage of the whole simulation process, this explains reduced execution time in solving this monodomain problem.

Monodomain simulations, as we shall see in Section 4.3, have important applications in research on

**Table 4.3:** Results from using FE/RKC(2, 1) for the time integration of a cell model in a monodomain simulation. Step sizes in milliseconds are the largest ones that produce results less than 5% MRMS error. The percentage of execution time (in seconds) that the cell solving process takes and the corresponding % MRMS errors are reported.

| Method | Step size (ms) | Execution time (s) | Cell solving % | % MRMS error |
|---|---|---|---|---|
| FE | 0.10 | 2.52 | 95.70 | 0.80 |
| RKC(2, 1) | 0.25 | 1.76 | 91.56 | 2.31 |

cardiac electrophysiology. Our improvement on monodomain simulation helps the academic community to move a step towards real-time cardiac simulations. We intend to provide a quantitative explanation of this improvement in Section 4.4.

In doing this monodomain experiment, we have made contributions to `CHASTE` by implementing the RKC(2, 1) method.

## 4.3 Improvement of the monodomain simulation

We consider applying the RKC(2, 1) method to the ODE system from the spatially discretized tissue equation following the MOL scheme in a famous monodomain problem — the benchmark problem of S. Niederer et al. [39].

The benchmark problem, introduced by S. Niederer et al. in 2011, was first designed to set up a gold standard for cardiac simulation community to measure efficiency and reliability for their simulation toolkits. The benchmark defines a set of unambiguous monodomain problems on a cuboid mesh (see Figure 4.3) with configuration listed in Table 4.4 and model-specific parameters listed in Table 4.5.

In 2012, the benchmark was solved on a computing cluster named `Sequoia` using `Cardioid`, which is a cardiac simulation toolkit aiming at performing high-performance simulation with detailed cell representations and heart geometries with near-cellular spatial resolution [36]; it was developed by researchers from IBM and the Lawrence Livermore National Laboratory. By performing an experiment using settings as similar as possible to that in [36], we intend to reproduce the experiment and seek potential improvement to the monodomain simulation. We attempt to replace the FE method with the RKC(2, 1) method for the time integration of the spatially discretized tissue equation, and we expect to gain increased step size but maintain numerical stability.

We generate the reference solution using FEM with high-order elements in `Nektar++` [5]. We first generate a set of numerical solutions in sequentially decreasing step sizes $\Delta t_j = 2^j \cdot 10^{-2}$ ms, where $j = 1, 0, \cdots, -5$. We then use Richardson extrapolation [49] on those solutions to obtain the reference solution to order seven.

The numerical simulation aimed at partially reproducing results in [36] is performed by using an unpublished `MATLAB` code from the Numerical Simulation Research Lab. The numerical solutions are obtained by
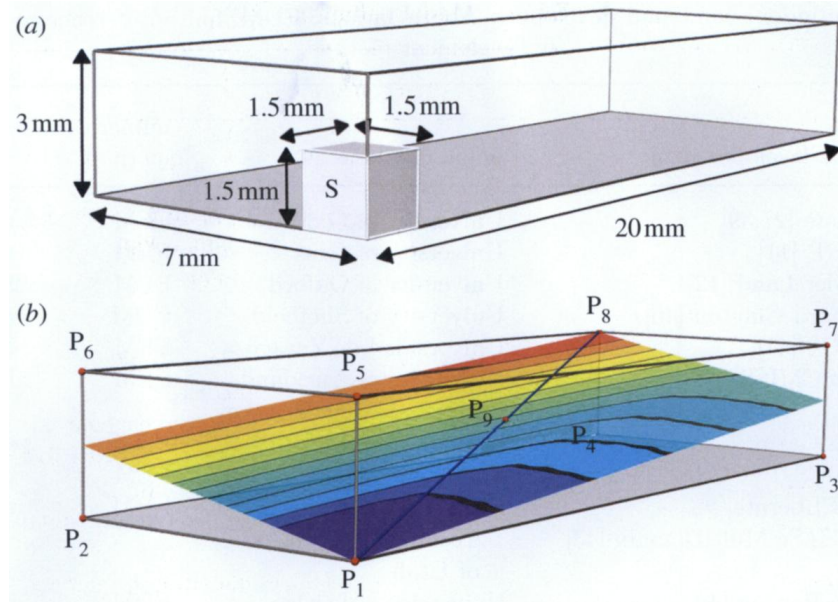
**Figure 4.3:** These figures show the cuboid mesh in the benchmark problem. $S$ is a cuboid in which there is an applied stimulus. $P_1$ to $P_9$ show points in the mesh where activation times are measured [39].

**Table 4.4:** Definition of the benchmark problem [39]. For variables that have a set of options, options used in this thesis are marked in bold.

| Variable | Description |
|---|---|
| geometric domain | cuboid |
| tissue equations | monodomain |
| material | transversely isotropic |
| PDE solver | **semi-implicit** or fully explicit |
| cell model | model of Ten Tusscher et al. (2006) [58] |
|    variant | epicardium cell model |
| cell model numerical integration scheme | fully explicit (FE, RK, RKC, **RL**), fully implicit (BE). |
| mesh type | **hexahedral**, tetrahedral |
| spatial discretization method | FEM, **FVM**, and finite difference method |
| basis function for FEM (if applicable) | Lagrange or Hermite/linear, quadratic, and cubic |
| pre-conditioner | Jacobi, incomplete LU, and **None** |
| matrix solver | conjugate gradient, generalized minimal residual, and **direct method** |
| system architecture | shared memory, distributed memory, and **serial** |

34

**Table 4.5:** Model-specific parameters of the benchmark problem [39]. For variables that have a set of options, options used in this thesis are marked in bold.

| Variable | Description |
|---|---|
| geometric domain | cuboid |
| dimensions | $20\,\mathrm{mm} \times 7\,\mathrm{mm} \times 3\,\mathrm{mm}$ |
| fibre orientation | fibres are aligned in the long axis |
| discretization | $0.5\,\mathrm{mm}$, **$0.2$** $\mathrm{mm}$, and $0.1\,\mathrm{mm}$ isotropic |
| PDE time steps | **$0.05$** ms, $0.01$ ms, $0.005$ ms, and **$0.15$** ms |
| stimulation geometry | $1.5\,\mathrm{mm} \times 1.5\,\mathrm{mm} \times 1.5\,\mathrm{mm}$ cube from a corner |
| stimulation protocol | $2\,\mathrm{ms}$ at $50\,000\,\mu\mathrm{A\,cm^{-3}}$ |
| surface area to volume ratio | $140\,\mathrm{mm^{-1}}$ |
| membrane capacitance | $1\,\mu\mathrm{Fcm^{-2}}$ |
| intra-longitudinal, intra-transversal, extra-longitudinal and extra-transversal conductivities, using $\sigma = \sigma_i \sigma_e (\sigma_e + \sigma_i)^{-1}$ in the monodomain (isotropic) mode | $0.17\,\mathrm{Sm^{-1}}$, $0.019\,\mathrm{Sm^{-1}}$, $0.62\,\mathrm{Sm^{-1}}$, and $0.243\,\mathrm{Sm^{-1}}$ |

first splitting the system using the Godunov OS method, spatially discretizing using the FVM, and applying the FE or RKC$(2,1)$ method for the time integration of the spatially discretized tissue equation. We point out because the numerical solutions are generated using low-order methods with a coarse mesh, we are not able to obtain solutions with less than 5% MRMS error. We instead visualize isosurfaces of solutions at some time points and evaluate accuracies of solutions as follows: we first pick solutions with isosurface visualization close to the reference solution. We note those solutions are less than 200% MRMS error. We then gradually increase step size until one of the two following things happens: the solution obtained by increasing the step size is visually distinct from previous solution and tends to differ significantly from the reference solution, or the MRMS error is significantly larger. We find out given a fixed spatial resolution and the required accuracy, the largest time step for integration of spatially discretized tissue equation is $0.05$ ms. In the RKC$(2,1)$ case, $0.15$ ms is the largest. Visualizations of isosurfaces of selected numerical solutions together with the reference solution are shown in Figure B.1 to Figure B.4 in Appendix B.

We run simulations using the FE and RKC$(2,1)$ methods adopting the corresponding time step sizes chosen above for the spatially discretized tissue equation, respectively. The step size for solving the cell model is kept consistent throughout with step size used for the solving tissue equation. We record our results in Table 4.6.

Referring to Table 4.6, the total execution time for solving the Niederer benchmark problem was significantly reduced by a factor of two. The reasons are two-fold: introducing the RKC$(2,1)$ method in solving the tissue equation allows us to take a larger step size to produce a stable numerical solution than using the

**Table 4.6:** Results on using FE/RKC$(2,1)$ in time integration of tissue equation in the benchmark problem with selected step sizes. Step sizes are in milliseconds. Cell solving times and total execution times are in seconds.

| Method | Step size (ms) | Cell solving (s) | Execution time (s) |
|---|---|---|---|
| FE | 0.05 | 2969.88 | 2988.62 |
| RKC$(2,1)$ | 0.15 | 955.01 | 1003.32 |

FE method; the step size bottleneck for the benchmark is solving the tissue equation, and introducing the RKC$(2,1)$ method in solving tissue equation breaks the bottleneck by ameliorating the step size restriction from small stability region of the FE method. We also note that solving the cell model also benefits from the increased step size from introducing the RKC$(2,1)$ method to time integration of tissue equation.

## 4.4 A theoretical explanation of the new approaches

We present a theoretical explanation of the improvements of stability brought by the multistage RKC and RLC methods. The explanations are presented by using the model of Courtemanche et al. [7] and the model of Maleckar et al. [31].

To explain stability improvement brought by the RKC$(2,1)$ and RKC$(3,1)$ methods, we solve the model of Courtemanche et al. from 0 to $500\,\text{ms}$. We record the step sizes used for three numerical methods (the FE, RKC$(2,1)$, and RKC$(3,1)$ methods), the corresponding % MRMS errors, and the respective execution times in Table 4.7. In the experiments of this section, we sample the Jacobian matrix of the ODE system of cell model 100 times during the simulation. We plot the scaled eigenvalues ($\lambda \Delta t$) of the sampled Jacobian matrices with stability regions of the FE, RKC$(2,1)$, and RKC$(3,1)$ methods in Figure 4.4. Referring to Section 3.2.2, for the ODE system in which we are interested, we require that the scaled eigenvalues of Jacobian matrices stay within the stability region of numerical method we are using. Although eigenvalues of the time-sampled Jacobian matrices of the system are not all included in the stability region of the numerical method, we note these eigenvalues only cause the numerical method to be unstable for a short time, but the method is stable in the whole interval if the scaled eigenvalues fall into the stability region sufficiently often.

We note that for the FE and RKC$(2,1)$ method, the largest step sizes we can take produce MRMS errors far less than 5%. We further note that in our experiment, a slightly larger step size causes the solution to become unbounded. This result is attributed to a constraint from stability not accuracy. We also note that for the RKC$(3,1)$ method, the largest step size produces an MRMS error close to 5%. In our experiment, we can increase the step size to get MRMS error slightly larger than 5% but still satisfy numerical stability. We refer to Figure 4.4 for an explanation: the reason that the RKC$(2,1)$ method being able to take a larger step size is from its extended stability region over negative real axis compared with the FE method, where the

scaled eigenvalues tend to cluster around. For the $RKC(3,1)$ method, we note that the eigenvalues scaled by the $RKC(3,1)$ step size generally fall in the stability region of the $RKC(3,1)$. This indicates for 5% MRMS error, the $RKC(3,1)$ method is not stiff for this problem — the step size is restricted by the prescribed accuracy, not numerical stability. We note from the execution times in Table 4.7 that the extra computing costs per step from multistage methods are compensated by increased step size; thus we conclude that the $RKC(2,1)$ method and the $RKC(3,1)$ method bring us benefit over the FE method.

**Table 4.7:** Numerical methods, their respective largest step sizes (in milliseconds) for computing numerical solutions that have less than 5% MRMS error, the corresponding % MRMS errors and execution times (in seconds) for solving the model of Courtemanche et al. [7].

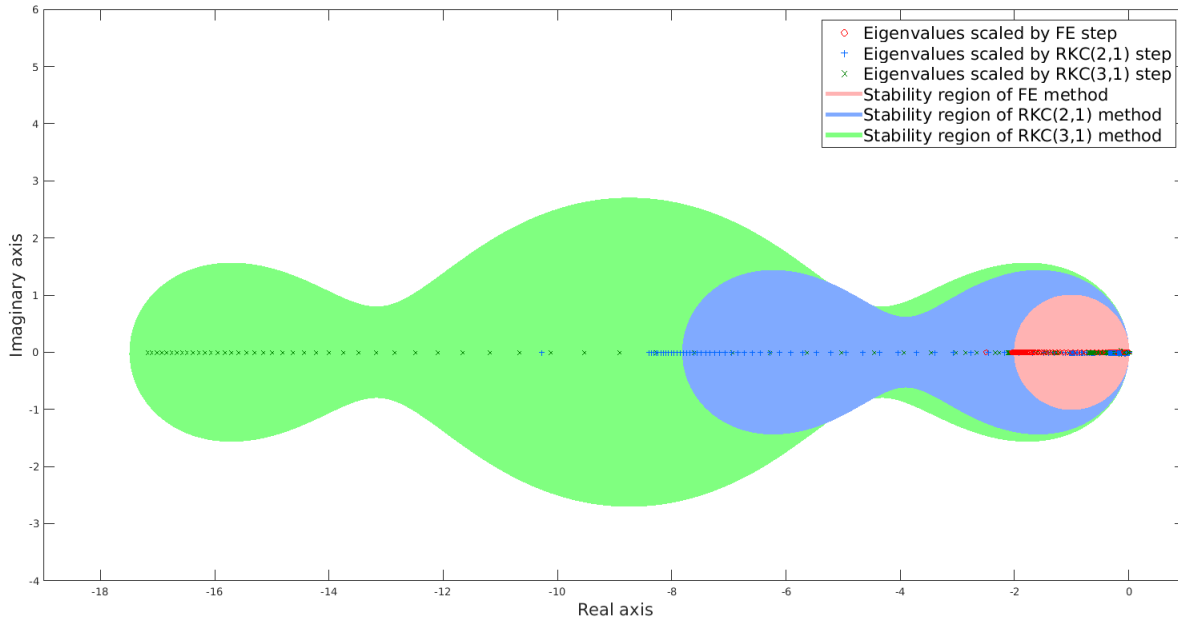| Numerical method | FE | $RKC(2,1)$ | $RKC(3,1)$ |
|---|---|---|---|
| Step size (ms) | 1.94e-02 | 7.97e-02 | 1.63e-01 |
| % MRMS error | 0.62 | 2.39 | 4.99 |
| Execution time (s) | 2.63e+00 | 1.25e+00 | 9.19e-01 |



**Figure 4.4:** Plots of stability region of different RKC methods and scaled eigenvalues of time-sampled Jacobian matrices over time of the model of Courtemanche et al.

We refer to solving the model of Maleckar et al. from 0 to 1 ms for illustration on stability improvement brought by the $RLC(2,1)$ and $RLC(3,1)$ methods. Recall that the $RLC(s,1)$ method treats gating and non-

gating variables with exponential integrator and RKC$(s, 1)$ method respectively, the exponential integrator does not have stability issue for linear system because it integrates the system exactly. We only need to focus on potential stability issues from time integration of non-gating variables, which are treated by the RKC$(s, 1)$ method. Attempting to exploit the eigenvalue analysis as in the previous example, we first identify eigenvalues that have a strong relationship with non-gating variables by the following steps:

1. We denote eigenvalue decomposition of Jacobian matrix $\mathbf{J}$ of the cell system with $\mathbf{V}^{-1}\mathbf{J}\mathbf{V} = \mathbf{\Lambda}$, where $\mathbf{V}$ is a matrix with columns being eigenvectors of $\mathbf{J}$ and $\mathbf{\Lambda}$ is a diagonal matrix with corresponding eigenvalues on its diagonal.

2. We identify columns $\mathbf{v}_{i_1}, \cdots, \mathbf{v}_{i_m}$ of $\mathbf{V}$ corresponding to non-gating variables $c_{i_1}, \cdots, c_{i_m}$.

3. For each column $\mathbf{v}_{i_k}, k = 1, 2, \cdots, m$, from $\mathbf{V}$, we pick $v_{i_k} \in \mathbf{v}_{i_k}$ that has largest absolute value in $\mathbf{v}_{i_k}$.

4. We associate non-gating variable $c_{i_k}$ with eigenvalue $i_k$ in $\mathbf{\Lambda}$.

We also plot the scaled eigenvalues of the sampled Jacobian matrices associated with non-gating variables together with the stability regions of the FE, RKC$(2, 1)$, and RKC$(3, 1)$ methods in Figure 4.5. We summarize the corresponding largest step sizes for obtaining solutions with less than 5% MRMS error, the % MRMS errors, and the corresponding execution times in Table 4.8. We note that the RKC$(3, 1)$ method takes the largest step size to satisfy 5% MRMS error. Because the RKC$(3, 1)$ method takes three times the step size of the RKC$(2, 1)$ method, this step size increase compensates for the extra computing costs it takes per step, and the RKC$(3, 1)$ method becomes the most efficient method for solving the model of Maleckar et al.

We note the same analysis presented above is still valid in the RL/RLC case. We have extended stability regions from using the RKC$(2, 1)$ and RKC$(3, 1)$ method for dealing with non-gating variables. In [33], Marsh, Torabi, and Spiteri pointed out the success occurs when the stiffness of the system is captured by gating variables. We conclude from our study that the RLC$(2, 1)$ and RLC$(3, 1)$ methods are good at treating moderately stiff models even when stiffness is not captured by gating variables. The model we choose is moderately stiff when we specify 5% MRMS error and use the RL method. From Figure 4.5, we note replacing FE part in RL method with multistage RKC methods increases stability region for numerical methods treating non-gating variables. This explains increase in step size for obtaining 5% MRMS error in Table 4.8.

**Table 4.8:** Numerical methods, their respective largest step sizes (in milliseconds) for getting numerical solutions that have less than 5% MRMS error, the corresponding % MRMS errors and execution times (in seconds) for solving the model of Maleckar et al. [31].

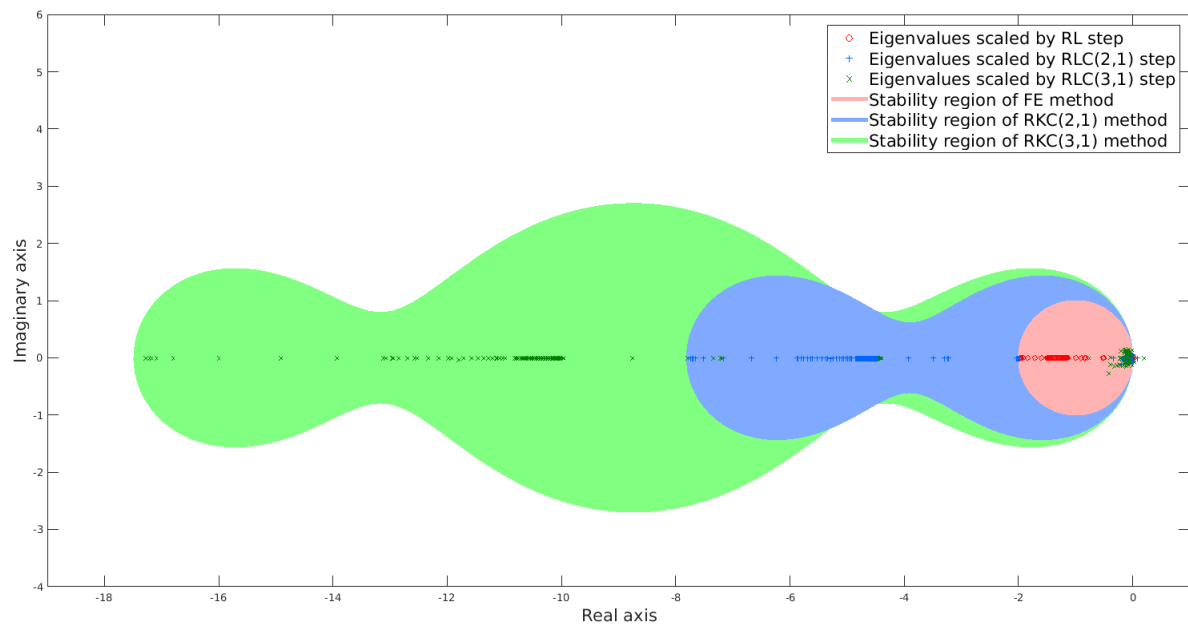| Numerical method | RL | RLC$(2, 1)$ | RLC$(3, 1)$ |
|---|---|---|---|
| Step size (ms) | 8.90e-05 | 3.48e-04 | 7.77e-04 |
| % MRMS error | 2.25 | 2.92 | 3.06 |
| Execution time (s) | 1.36e+00 | 5.73e-01 | 3.80e-01 |

**Figure 4.5:** Plots of stability region of different RKC methods and scaled eigenvalues of time-sampled Jacobian matrices over time of the model of Maleckar et al. The scaled eigenvalues we plot are only relate to system components that are related to non-gating variables and are integrated by RKC methods.

# Chapter 5

# Conclusions and Future Work

Cardiac simulation research helps address the need from clinical community to provide non-invasive and quantitative approaches to investigate electrical activity in the heart. With research on advanced computing techniques to solve cardiac mathematical models, cardiac simulation is playing a more important role in clinical applications.

This thesis contributes to research on cardiac simulation in four aspects:

1. We propose and implement several numerical methods for the time integration of cardiac cell models in Section 4.2. To the best of our knowledge, some of the numerical methods (the $RKC(s, 1)$ method applied to cardiac cell models and the new $RLC(s, 1)$ method where $s \geq 2$) have not been investigated by other researchers before. Compared with the study presented in [32], we find introducing new numerical methods reduces overall computing time for 19 of 37 cardiac cell models when specifying numerical accuracy of 5% MRMS error. We also see significant step size improvement through the use of multistage RKC and RL methods.

2. We implement the $RKC(2, 1)$ method in `CHASTE`. We use this method in the time integration of cell equations in a pre-defined monodomain problem. The results show a reduced overall execution time by a factor of up to 1.4 from increased step size the $RKC(2, 1)$ method took when specifying 5% MRMS error. With the fact that the monodomain model is an important benchmark used by the cardiac simulation community [39], we conclude that the $RKC(2, 1)$ method is practically useful in solving cardiac mathematical models. The implementation also extends `CHASTE`'s ability to solve cardiac mathematical models.

3. With a hope to partially reproduce and seek improvement to experiment presented in [36], we also apply the RKC method to the time integration of spatially discretized tissue equations in the benchmark problem proposed by S. Niederer et al. [39] in Section 4.3. The tissue equations are solved by using the FE method in [36] that is replaced with the $RKC(2, 1)$ method in our experiment. When specifying desired level of visual quality of the numerical solution, we are able to obtain the solution in only $1/3$ of the previous execution time. This indicates that the spatially discretized tissue equations are rather non-stiff equations for the $RKC(2, 1)$ method compared with the FE method and are more suitable to be tackled by the $RKC(2, 1)$ method.

4. We provide a theoretical explanation to improvement on time integration of cardiac cell models in Section 4.4. This analysis is based on looking at eigenvalues, which are from the time-sampled Jacobian matrix of the corresponding system of the cell model, scaled by the different step sizes of the numerical methods, and the stability region of the numerical methods. The numerical method is stable when for the majority of time-sampled Jacobian matrices, the scaled eigenvalues fall into their stability regions. The same analysis holds valid for multistage RLC methods by associating eigenvalues with non-gating variables that are treated by multistage RKC methods.

With a hope to extend studies in this thesis to further benefit research on cardiac simulation, we propose some possible future work:

1. We hope to implement the proposed numerical methods, especially the $\text{RKC}(s,1)$ and the $\text{RLC}(s,1)$ method in various cardiac simulation toolkits included in [39]. We point out the implementations from those methods can be expected to help those toolkits in dealing with problems that are moderately stiff.

2. We wish to explore further to the $\text{RKC}(s,1)$ and the $\text{RLC}(s,1)$ methods for $s \geq 4$. We note from Table 4.2 that the $\text{RKC}(3,1)$ and the $\text{RLC}(3,1)$ methods win on four and three models, respectively. This indicates increasing stages might still offer benefits for some models.

3. We are interested in investigating and implementing the $\text{RKC}(s,2)$ method. We expect the second-order RKC method could bring benefits over second-order RK methods in general.

4. Our study of cardiac cell models does not stop at the 37 models considered in this thesis. The research community keeps bringing new models to the `CellML` project. With growing needs from clinical applications, we expect to study more cardiac cell models and provide informative guidance to research community.

5. It is a natural idea that we can extend our study to a bidomain problem with physiological settings close to what is desired in clinical applications. We hope to have a chance to do a large-scale bidomain simulation similar to [36]. We believe this proposed experiment could bring useful guidance towards understanding heart arrhythmia.

# References

[1] The CellML Project. http://cellml.org/.

[2] A.J. Atkinson, S.R. Kharche, M.G. Bateman, P.A. Iaizzo, and H. Dobrzynski. 3d anatomical reconstruction of human cardiac conduction system and simulation of bundle branch block after TAVI procedure. In *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 5583–5586. IEEE, 2016.

[3] G.W. Beeler and H. Reuter. Reconstruction of the action potential of ventricular myocardial fibres. *The Journal of Physiology*, 268(1):177, 1977.

[4] V.E. Bondarenko, G.P. Szigeti, G.C.L. Bett, S. Kim, and R.L. Rasmusson. Computer model of action potential of mouse ventricular myocytes. *American Journal of Physiology-Heart and Circulatory Physiology*, 287(3):H1378–H1403, 2004.

[5] C.D. Cantwell, D. Moxey, A. Comerford, A. Bolis, G. Rocco, G. Mengaldo, D. De Grazia, S. Yakovlev, J-E Lombard, D. Ekelschot, et al. Nektar++: An open-source spectral/hp element framework. *Computer Physics Communications*, 192:205–219, 2015.

[6] R.H. Clayton, O. Bernus, E.M. Cherry, H. Dierckx, F.H. Fenton, L. Mirabella, A.V. Panfilov, F.B. Sachse, G. Seemann, and H. Zhang. Models of cardiac tissue electrophysiology: progress, challenges and open questions. *Progress in Biophysics and Molecular Biology*, 104(1):22–48, 2011.

[7] M. Courtemanche, R.J. Ramirez, and S. Nattel. Ionic mechanisms underlying human atrial action potential properties: insights from a mathematical model. *American Journal of Physiology-Heart and Circulatory Physiology*, 275(1):H301–H321, 1998.

[8] R. de Vogelaere. Methods of integration which preserve the contact transformation property of the hamiltonian equations. *Department of Mathematics, University of Notre Dame, Report*, 4:30, 1956.

[9] S.S. Demir, J.W. Clark, and W.R. Giles. Parasympathetic modulation of sinoatrial node pacemaker activity in rabbit heart: a unifying model. *American Journal of Physiology–Heart and Circulatory Physiology*, 276(6):H2221–H2244, 1999.

[10] S.S. Demir, J.W. Clark, C.R. Murphey, and W.R. Giles. A mathematical model of a rabbit sinoatrial node cell. *American Journal of Physiology-Cell Physiology*, 266(3):C832–C852, 1994.

[11] D. DiFrancesco and D. Noble. A model of cardiac electrical activity incorporating ionic pumps and concentration changes. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 307(1133):353–398, 1985.

[12] S. Dokos, B. Celler, and N. Lovell. Ion currents underlying sinoatrial node pacemaker activity: a new single cell mathematical model. *Journal of Theoretical Biology*, 181(3):245–272, 1996.

[13] M. Ethier and Y. Bourgault. Semi-implicit time-discretization schemes for the bidomain model. *SIAM Journal on Numerical Analysis*, 46(5):2443–2468, 2008.

[14] G.M. Faber and Y. Rudy. Action potential and contractility changes in $[Na^+]_i$ overloaded cardiac myocytes: a simulation study. *Biophysical Journal*, 78(5):2392–2404, 2000.

[15] R. FitzHugh. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical Journal*, 1(6):445–466, 1961.

[16] J.J. Fox, J.L. McHarg, and R.F. Gilmour. Ionic mechanism of electrical alternans. *American Journal of Physiology-Heart and Circulatory Physiology*, 282(2):H516–H530, 2002.

[17] D.J. Higham and L.N. Trefethen. Stiffness of odes. *BIT Numerical Mathematics*, 33(2):285–303, 1993.

[18] D.W. Hilgemann and D. Noble. Excitation-contraction coupling and extracellular calcium transients in rabbit atrium: reconstruction of basic cellular mechanisms. *Proceedings of the Royal Society of London B: Biological Sciences*, 230(1259):163–205, 1987.

[19] T.J. Hund and Y. Rudy. Rate dependence and regulation of action potential and calcium transient in a canine cardiac ventricular cell model. *Circulation Research*, 110(20):3168–3174, 2004.

[20] W.H. Hundsdorfer and J.G. Verwer. *Numerical solution of time-dependent advection-diffusion-reaction equations*, volume 33. Springer Science & Business Media, 2013.

[21] A.O. Ibrahim, A.I. Adediji, and D.J. Olufemi. Dynamical modelling of cardiac electrical activity using bidomain approach: The effects of variation of ionic model parameters. *Journal of Biomedical Science and Engineering*, 6(06):598, 2013.

[22] M.S. Jafri, J.J. Rice, and R.L. Winslow. Cardiac $Ca^{2+}$ dynamics: the roles of ryanodine receptor adaptation and sarcoplasmic reticulum load. *Biophysical Journal*, 74(3):1149–1168, 1998.

[23] C. Johnson. *Numerical solution of partial differential equations by the finite element method*. Courier Corporation, 2012.

[24] G. Karniadakis and S. Sherwin. *Spectral/hp element methods for computational fluid dynamics*. Oxford University Press, 1st edition, 1999.

[25] A.M. Katz. *Physiology of the Heart*. Lippincott Williams & Wilkins, 2010.

[26] J.P. Keener and J. Sneyd. *Mathematical physiology*, volume 1. Springer, 1998.

[27] E. Laskowski. What's a normal resting heart rate? *Health Lifestyle Fitness*, 2012.

[28] R.J. Le Roy. Level: A computer program for solving the radial schrödinger equation for bound and quasibound levels. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 186:167–178, 2017.

[29] C. Luo and Y. Rudy. A model of the ventricular cardiac action potential. depolarization, repolarization, and their interaction. *Circulation Research*, 68(6):1501–1526, 1991.

[30] C. 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.

[31] M.M. Maleckar, J.L. Greenstein, N.A. Trayanova, and W.R. Giles. Mathematical simulations of ligand-gated and cell-type specific effects on the action potential of human atrium. *Progress in Biophysics and Molecular Biology*, 98(2):161–170, 2008.

[32] M.E. Marsh. An assessment of numerical methods for cardiac simulation. Master's thesis, Department of Mathematics and Statistics, University of Saskatchewan, Saskatoon, 2012.

[33] M.E. Marsh, S. Torabi, and R.J. Spiteri. The secrets to the success of the Rush–Larsen method and its generalizations. *IEEE Transactions on Biomedical Engineering*, 59(9):2506–2515, 2012.

[34] R.E. McAllister, D. Noble, and R.W. Tsien. Reconstruction of the electrical activity of cardiac purkinje fibres. *The Journal of Physiology*, 251(1):1, 1975.

[35] G.R. Mirams, C.J. Arthurs, M.O. Bernabeu, R. Bordas, J. Cooper, A. Corrias, Y. Davit, S. Dunn, A.G. Fletcher, D.G. Harvey, et al. CHASTE: an open source C++ library for computational physiology and biology. *PLoS Computational Biology*, 9(3):e1002970, 2013.

[36] A.A. Mirin, D.F. Richards, J.N. Glosli, E.W. Draeger, B. Chan, J. Fattebert, W.D. Krauss, T. Oppelstrup, J.J. Rice, J.A. Gunnels, et al. Toward real-time modeling of human heart ventricles at cellular resolution: Multi-hour simulation of drug-induced arrhythmias. Technical report, Lawrence Livermore National Laboratory (LLNL), Livermore, CA, 2012.

[37] J.W. Moore and F. Ramon. On numerical integration of the Hodgkin and Huxley equations for a membrane action potential. *Journal of theoretical Biology*, 45(1):249–273, 1974.

[38] J. Nagumo, S. Arimoto, and S. Yoshizawa. An active pulse transmission line simulating nerve axon. *Proceedings of the IRE*, 50(10):2061–2070, 1962.

[39] S.A. Niederer, E. Kerfoot, A.P. Benson, M.O. Bernabeu, O. Bernus, C. Bradley, E.M. Cherry, R. Clayton, F.H. Fenton, A. Garny, et al. Verification of cardiac tissue electrophysiology simulators using an n-version benchmark. *Physiological Transactions of the Royal Society A*, 369(1954):4331–4351, 2011.

[40] D. Noble. A modification of the hodgkin–huxley equations applicable to purkinje fibre action and pacemaker potentials. *The Journal of Physiology*, 160(2):317, 1962.

[41] D. Noble and S.J. Noble. A model of sino-atrial node electrical activity based on a modification of the difrancesco–noble (1984) equations. *Proceedings of the Royal Society of London B: Biological Sciences*, 222(1228):295–304, 1984.

[42] D. Noble, S.J. Noble, G.C.L. Bett, Y.E. Earm, W.K. Ho, and I.K. So. The role of sodium-calcium exchange during the cardiac action potential. *Annals of the New York Academy of Sciences*, 639(1):334–353, 1991.

[43] D. Noble, A. Varghese, P. Kohl, and P. Noble. Improved guinea-pig ventricular cell model incorporating a diadic space, IKr and IKs, and length-and tension-dependent processes. *Canadian Journal of Cardiology*, 14(1):123–134, 1998.

[44] A. Nygren, C. Fiset, L. Firek, J.W. Clark, D.S. Lindblad, R.B. Clark, and W.R. Giles. Mathematical model of an adult human atrial cell. *Circulation Research*, 82(1):63–81, 1998.

[45] Government of Canada. Heart disease — heart health. `https://www.canada.ca/en/public-health/services/diseases/heart-disease-heart-health.html`, January 2017.

[46] S.V. Pandit, R.B. Clark, W.R. Giles, and S.S. Demir. A mathematical model of action potential heterogeneity in adult rat left ventricular myocytes. *Biophysical Journal*, 81(6):3029–3051, 2001.

[47] S.V. Pandit, W.R. Giles, and S.S. Demir. A mathematical model of the electrophysiological alterations in rat ventricular myocytes in type-I diabetes. *Biophysical Journal*, 84(2):832–841, 2003.

[48] J.L. Puglisi and D.M. Bers. LabHEART: an interactive computer model of rabbit ventricular myocyte ion channels and Ca transport. *American Journal of Physiology-Cell Physiology*, 281(6):C2049–C2060, 2001.

[49] L.F. Richardson. The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 210:307–357, 1911.

[50] S. Rush and H. Larsen. A practical algorithm for solving dynamic membrane equations. *IEEE Transactions on Biomedical Engineering*, (4):389–392, 1978.

[51] B.F.A.S. Sakmann, A.J. Spindler, S.M. Bryant, K.W. Linz, and D. Noble. Distribution of a persistent sodium current across the ventricular wall in guinea pigs. *Circulation Research*, 87(10):910–914, 2000.

[52] R.J. Spiteri and R.C. Dean. Stiffness analysis of cardiac electrophysiological models. *Annals of Biomedical Engineering*, 38(12):3592–3604, 2010.

[53] R.J. Spiteri and S. Torabi. Operator splitting for the bidomain model revisited. *Journal of Computational and Applied Mathematics*, 296:550–563, 2016.

[54] P. Stewart, O.V. Aslanidi, D. Noble, P.J. Noble, M.R. Boyett, and H. Zhang. Mathematical models of the electrical action potential of purkinje fibre cells. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 367(1896):2225–2255, 2009.

[55] J. Sundnes, R. Artebrant, O. Skavhaug, and A. Tveito. A second-order algorithm for solving dynamic cell membrane equations. *IEEE Transactions on Biomedical Engineering*, 56(10):2546–2548, 2009.

[56] J. Sundnes, G.T. Lines, X. Cai, B.F. Nielsen, K. Mardal, and A. Tveito. *Computing the electrical activity in the heart*, volume 1. Springer Science & Business Media, 2007.

[57] 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(4):H1573–H1589, 2004.

[58] 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(3):H1088–H1100, 2006.

[59] L. Tung. *A bi-domain model for describing ischemic myocardial dc potentials.* PhD thesis, Massachusetts Institute of Technology, 1978.

[60] P.J. Van der Houwen. The development of Runge-Kutta methods for partial differential equations. *Applied Numerical Mathematics*, 20(3):261–272, 1996.

[61] L.J. Wang and E.A. Sobie. Mathematical model of the neonatal mouse ventricular action potential. *American journal of physiology. Heart and circulatory physiology*, 294(6):H2565, 2008.

[62] J.P. Whiteley. An efficient numerical technique for the solution of the monodomain and bidomain equations. *IEEE Transactions on Biomedical Engineering*, 53(11):2139–2147, 2006.

[63] R.L. Winslow, J. Rice, S. Jafri, E. Marban, and B. O'Rourke. Mechanisms of altered excitation-contraction coupling in canine tachycardia-induced heart failure, II. *Circulation Research*, 84(5):571–586, 1999.

[64] H. Zhang, A.V. Holden, I. Kodama, H. Honjo, M. Lei, T. Varghese, and M.R. Boyett. Mathematical models of action potentials in the periphery and center of the rabbit sinoatrial node. *American Journal of Physiology-Heart and Circulatory Physiology*, 279(1):H397–H421, 2000.

# Appendix A

# The Luo–Rudy model

We present a complete form of the Luo–Rudy model [29].
**Inward currents:**

- **Fast sodium current**

$$I_{\mathrm{Na}} = \overline{G}_{\mathrm{Na}} \cdot m^3 \cdot h \cdot j \cdot (v - E_{\mathrm{Na}}) \tag{A.1}$$

Activation gate, $m$

$$\frac{\mathrm{d}m}{\mathrm{d}t} = \alpha_m(1-m) - \beta_m m \tag{A.2a}$$

$$\alpha_m = \frac{0.32(v+47.13)}{1 - e^{-0.1(v+47.13)}} \tag{A.2b}$$

$$\beta_m = 0.08 e^{-v/11} \tag{A.2c}$$

Fast inactivation gate, $h$

$$\frac{\mathrm{d}h}{\mathrm{d}t} = \alpha_h(1-h) - \beta_h h \tag{A.3a}$$

$$\alpha_h = \begin{cases} 0.135 e^{(v+80)/-6.8} & v < -40 \text{ mV} \\\\ 0 & v \geq -40 \text{ mV} \end{cases} \tag{A.3b}$$

$$\beta_h = \begin{cases} 3.56 e^{0.079v} + 3.1 \cdot 10^5 e^{0.35v} & v < -40 \text{ mV} \\\\ \frac{1}{0.13(1+e^{(v+10.66)/-11.1})} & v \geq -40 \text{ mV} \end{cases} \tag{A.3c}$$

Slow inactivation gate, $j$

$$\frac{\mathrm{d}j}{\mathrm{d}t} = \alpha_j(1-j) - \beta_j j \tag{A.4a}$$

$$\alpha_j = \begin{cases} \frac{-1.2714 \cdot 10^5 e^{0.2444v} - }{1+e^{0.311(v+79.23)}} \cdots \\\\ \frac{3.474 \cdot 10^{-5} e^{-0.04391v} \cdot (v+37.78)}{1+e^{0.311(v+79.23)}} & v < -40 \text{ mV} \\\\ 0 & v \geq -40 \text{ mV} \end{cases} \tag{A.4b}$$

$$\beta_j = \begin{cases} \frac{0.1212 e^{-0.01052v}}{1+e^{-0.1378(v+40.14)}} & v < -40 \text{ mV} \\\\ \frac{0.3 e^{-2.535 \cdot 10^{-7} v}}{1+e^{-0.1(v+32)}} & v \geq -40 \text{ mV} \end{cases} \tag{A.4c}$$

- **Slow inward current**

$$I_{\mathrm{si}} = \overline{G}_{\mathrm{si}} \cdot d \cdot f \cdot (v - E_{\mathrm{si}}) \tag{A.5}$$

$$E_{\mathrm{si}} = 7.7 - 13.0287 \cdot \ln([\mathrm{Ca}]_i) \tag{A.6}$$

Activation gate, $d$

$$\frac{\mathrm{d}d}{\mathrm{d}t} = \alpha_d(1-d) - \beta_d d \tag{A.7a}$$

$$\alpha_d = \frac{0.095 e^{-0.01(v-5)}}{1 + e^{-0.072(v-5)}} \tag{A.7b}$$

$$\beta_d = \frac{0.07 e^{-0.017(v+44)}}{1 + e^{0.05(v+44)}} \tag{A.7c}$$

46

Inactivation gate, $f$

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \alpha_f(1-f) - \beta_f f \tag{A.8a}$$

$$\alpha_f = \frac{0.012e^{-0.008(v+28)}}{1 + e^{0.15(v+28)}} \tag{A.8b}$$

$$\beta_f = \frac{0.0065e^{-0.02(v+30)}}{1 + e^{-0.2(v+30)}} \tag{A.8c}$$

Calcium uptake

$$\frac{([\mathrm{Ca}]_i)}{t} = -10^{-4}I_{\mathrm{si}} + 0.07(10^{-4} - [\mathrm{Ca}]_i) \tag{A.9}$$

**Outward Currents:**

– **Time-dependent potassium current**

$$I_{\mathrm{K}} = \overline{G}_{\mathrm{K}} \cdot X \cdot X_{\mathrm{i}} \cdot (v - E_{\mathrm{K}}) \tag{A.10}$$
$$\overline{G}_{\mathrm{K}} = 0.282 \cdot \sqrt{[\mathrm{K}]_o/5.4} \tag{A.11}$$

Activation gate, $X$

$$\frac{\mathrm{d}X}{\mathrm{d}t} = \alpha_X(1-X) - \beta_X X \tag{A.12a}$$

$$\alpha_X = \frac{0.0005e^{0.083(v+50)}}{1 + e^{0.057(v+50)}} \tag{A.12b}$$

$$\beta_X = \frac{0.0013e^{-0.06(v+20)}}{1 + e^{-0.04(v+20)}} \tag{A.12c}$$

Inactivation gate, $X_{\mathrm{i}}$

$$X_{\mathrm{i}} = \begin{cases} \frac{2.837(e^{0.04(v+77)}-1)}{(v+77)e^{0.04(v+35)}} & v > -100 \text{ mV} \\ \\ 1 & v \leq -100 \text{ mV} \end{cases} \tag{A.13}$$

– **Time-independent potassium current**

$$I_{\mathrm{K1}} = \overline{G}_{\mathrm{K1}} \cdot \mathrm{K1}_\infty \cdot (v - E_{\mathrm{K1}}) \tag{A.14}$$
$$\overline{G}_{\mathrm{K1}} = 0.6047 \cdot \sqrt{[\mathrm{K}]_o/5.4} \tag{A.15}$$

Inactivation gate, K1

$$\mathrm{K1}_\infty = \frac{\alpha_{\mathrm{K1}}}{\alpha_{\mathrm{K1}} + \beta_{\mathrm{K1}}} \tag{A.16a}$$

$$\alpha_{\mathrm{K1}} = \frac{1.02}{1 + e^{0.2385(v - E_{\mathrm{K1}} - 59.215)}} \tag{A.16b}$$

$$\beta_{\mathrm{K1}} = \begin{cases} \frac{0.49124e^{0.08032(v - E_{\mathrm{K1}} + 5.476)} +}{1 + e^{-0.5143(v - E_{\mathrm{K1}} + 4.753)}} \cdots \\ \frac{e^{0.06175(v - E_{\mathrm{K1}} - 594.31)}}{1 + e^{-0.5143(v - E_{\mathrm{K1}} + 4.753)}} \end{cases} \tag{A.16c}$$

$$\tag{A.16d}$$

– **Plateau potassium current**

$$I_{\mathrm{Kp}} = \overline{G}_{\mathrm{KP}} \cdot \mathrm{Kp} \cdot (v - E_{\mathrm{Kp}}) \tag{A.17}$$
$$E_{\mathrm{Kp}} = E_{\mathrm{K1}} \tag{A.18}$$
$$\mathrm{Kp} = \frac{1}{1 + e^{(7.488 - v)/5.98}} \tag{A.19}$$

– **Background potassium current**

$$I_{\mathrm{b}} = \overline{G}_{\mathrm{b}} \cdot (v - E_{\mathrm{b}}) \tag{A.20}$$

**Total ionic current:**

$$\begin{aligned}
I_{\mathrm{ion}} &= I_{\mathrm{Na}} + I_{\mathrm{si}} + I_{\mathrm{K}} + I_{\mathrm{K1}} + I_{\mathrm{Kp}} + I_{\mathrm{b}} \\
&= \overline{G}_{\mathrm{Na}} \cdot m^3 \cdot h \cdot j \cdot (v - E_{\mathrm{Na}}) + \overline{G}_{\mathrm{si}} \cdot d \cdot f \cdot (v - E_{\mathrm{si}}) \\
&\quad + \overline{G}_{\mathrm{K}} \cdot X \cdot X_{\mathrm{i}} \cdot (v - E_{\mathrm{K}}) + \overline{G}_{\mathrm{K1}} \cdot \mathrm{K1}_{\mathrm{inf}} \cdot (v - E_{\mathrm{K1}}) \\
&\quad + \overline{G}_{\mathrm{KP}} \cdot \mathrm{Kp} \cdot (v - E_{\mathrm{Kp}}) + \overline{G}_{\mathrm{b}} \cdot (v - E_{\mathrm{b}})
\end{aligned} \tag{A.21}$$

For an individual cardiac cell we have that the transmembrane potential $v$ is given by:

$$\frac{\mathrm{d}v}{\mathrm{d}t} = -\frac{1}{C_m}(I_{\mathrm{ion}} + I_{\mathrm{stim}}), \tag{A.22}$$

where $C_m$ is the membrane capacitance and $I_{\mathrm{stim}}$ is the stimulus current applied by the sinoatrial node.

The following table shows the values of the channel conductances, the reversal potentials for the ions, and other parameters.

**Table A.1:** Parameters for the Luo–Rudy model; the conductances are in mS/cm$^2$ and the reversal potentials are in mV

| Channel Conductance (mS/cm$^2$) | | Reversal Potential (V) | | Other Parameters |
|---|---|---|---|---|
| $\overline{G}_{\mathrm{Na}}$ | = 23.0 | $E_{\mathrm{Na}}$ | = 54.4 | Resting Membrane Potential $V_{\mathrm{rest}} = -84.0$ mV |
| $\overline{G}_{\mathrm{si}}$ | = 0.09 | $E_{\mathrm{si}}$ | = 118.7 | Membrane Threshold Potential $V_{\mathrm{threshold}} = -60$ mV |
| $\overline{G}_{\mathrm{K}}$ | = 0.282 | $E_{\mathrm{K}}$ | = –77 | $[K]_o = 5.4$ mM |
| $\overline{G}_{\mathrm{K1}}$ | = 0.6047 | $E_{\mathrm{K1}}$ | = –87.2 | Membrane Capacitance $C_m = 1\ \mu\mathrm{F\,cm}^2$ |
| $\overline{G}_{\mathrm{Kp}}$ | = 0.0183 | $E_{\mathrm{Kp}}$ | = –87.2 | |
| $\overline{G}_{\mathrm{b}}$ | = 0.03921 | $E_{\mathrm{b}}$ | = –59.87 | |

# Appendix B

## Visualizations of isosurfaces of various solutions

We present visualizations of solution isosurfaces obtained at different times in the experiment in Section 4.3 in Figure B.1 to Figure B.4. The solutions presented are sampled at time $t = 10\,\text{ms}$, $t = 20\,\text{ms}$, $t = 30\,\text{ms}$, and $t = 40\,\text{ms}$. For each figure, the solutions were obtained by applying the FE method, by applying the RKC$(2, 1)$ method with $dt = 0.10\,\text{ms}$, by applying the RKC$(2, 1)$ method with $dt = 0.15\,\text{ms}$ to tissue equations, and placing the reference solution at the top-left, top-right, bottom-left, and bottom-right, respectively.

**Figure B.1:** Isosurface visualizations for solutions to the benchmark problem by Niederer et al. in Section 4.3 at $t = 10\,\text{ms}$ with different step sizes used for time integration of spatially discretized monodomain tissue equation.
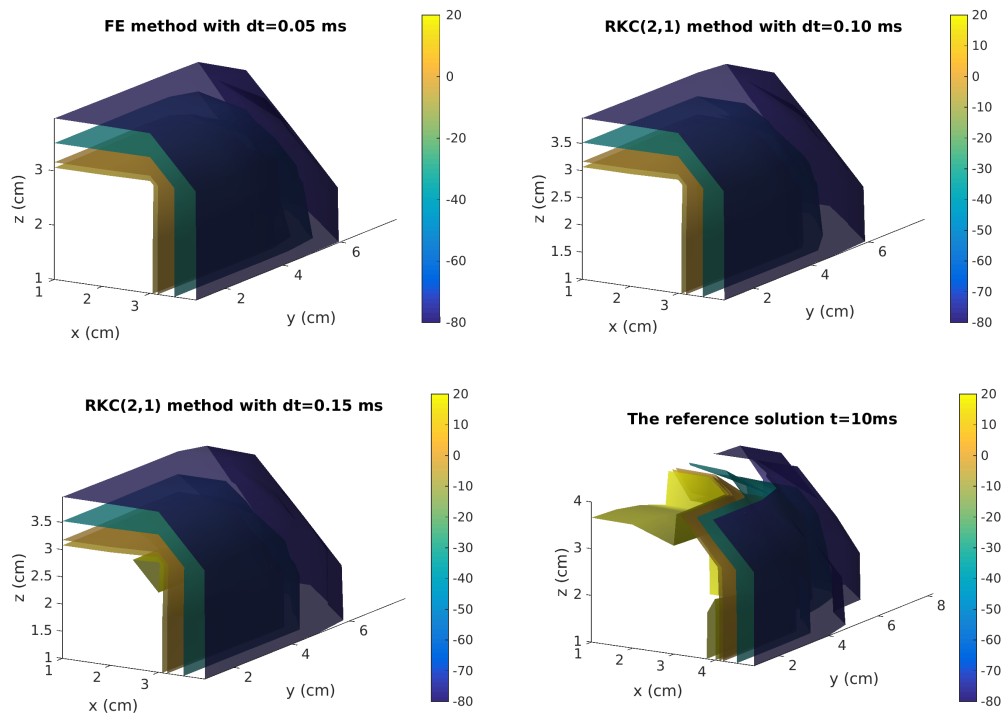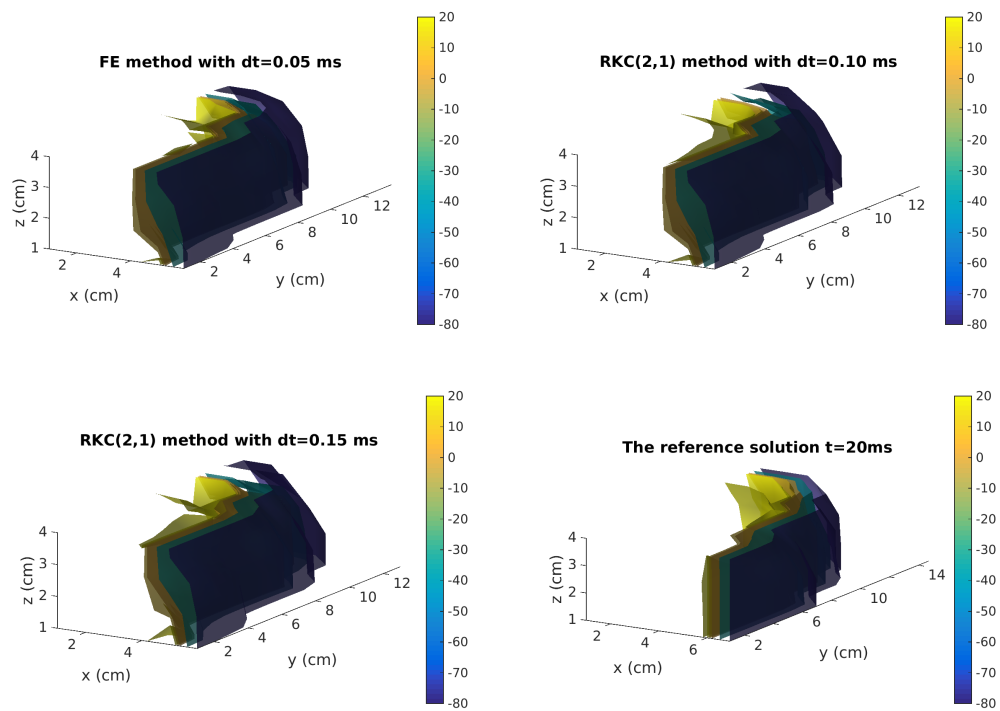
**Figure B.2:** Isosurface visualizations for solutions to the benchmark problem by Niederer et al. in Section 4.3 at $t = 20$ ms with different step sizes used for time integration of spatially discretized monodomain tissue equation.
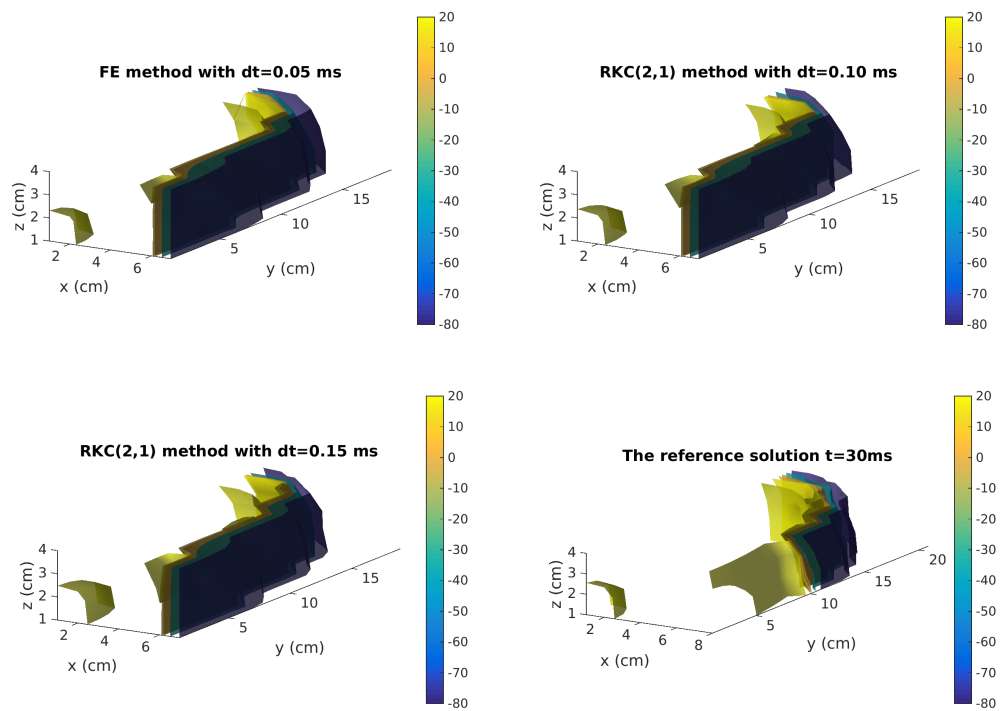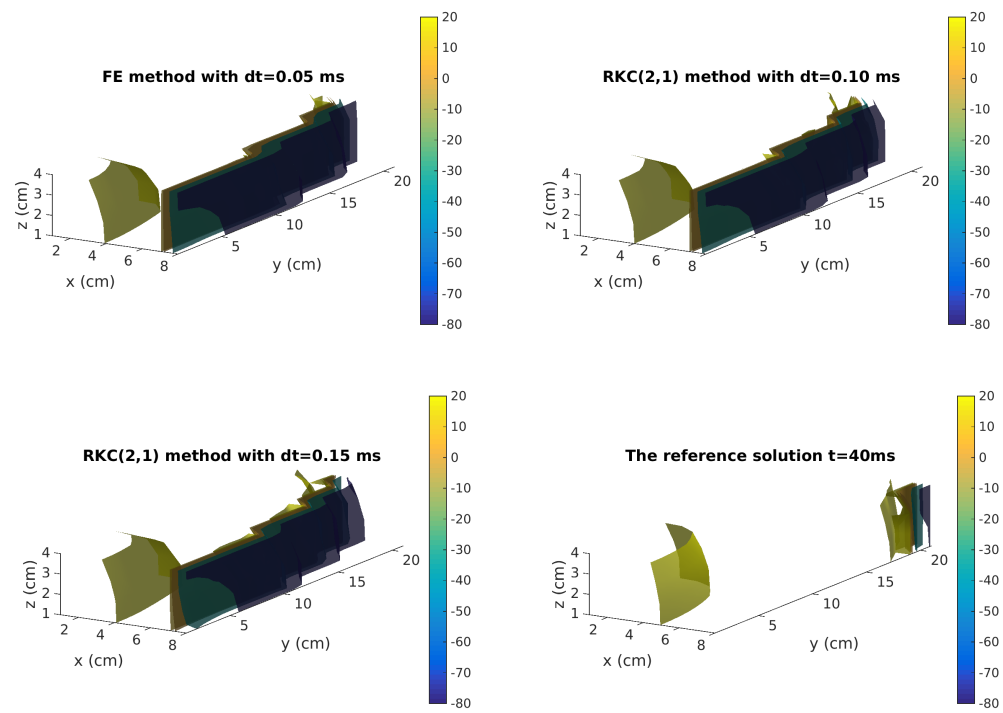
**Figure B.3:** Isosurface visualizations for solutions to the benchmark problem by Niederer et al. in Section 4.3 at $t = 30\,\text{ms}$ with different step sizes used for time integration of spatially discretized monodomain tissue equation.

**Figure B.4:** Isosurface visualizations for solutions to the benchmark problem by Niederer et al. in Section 4.3 at $t = 40\,\text{ms}$ with different step sizes used for time integration of spatially discretized monodomain tissue equation.