Analysis of time-stepping methods for the monodomain model

Thomas Roy, Yves Bourgault, Charles Pierre

Analysis of time-stepping methods for the monodomain model

Thomas Roy [*1], Yves Bourgault[†2] and Charles Pierre [‡3]

[1] Mathematical Institute, University of Oxford, United Kingdom.
[2] Department of Mathematics and Statistics, University of Ottawa, Ontario, Canada.
[3] Laboratoire de Mathématiques et de leurs Applications, UMR CNRS 5142, Université de Pau et des Pays de l'Adour, France.

25 May, 2020

Abstract

To a large extent, the stiffness of the bidomain and monodomain models depends on the choice of the ionic model, which varies in terms of complexity and realism. In this paper, we compare and analyze a variety of time-stepping methods: explicit or semi-implicit, operator splitting, exponential, and deferred correction methods. We compare these methods for solving the bidomain model coupled with three ionic models of varying complexity and stiffness: the phenomenological Mitchell-Schaeffer model, the more realistic Beeler-Reuter model, and the stiff and very complex ten Tuscher-Noble-Noble-Panfilov (TNNP) model. For each method, we derive absolute stability criteria of the spatially discretized monodomain model and verify that the theoretical critical time-steps obtained closely match the ones in numerical experiments. We also verify that the numerical methods achieve an optimal order of convergence on the model variables and derived quantities (such as speed of the wave, depolarization time), and this in spite of the local non-differentiability of some of the ionic models. The efficiency of the different methods is also considered by comparing computational times for similar accuracy. Conclusions are drawn on the methods to be used to solve the monodomain model based on the model stiffness and complexity, measured respectively by the eigenvalues of the model's Jacobian and the number of variables, and based on strict stability and accuracy criteria.

Keywords: cardiac electrophysiology, monodomain model, stiff problems, time-stepping methods, absolute stability

Subject classification: 65M12, 65L04, 65L06, 35K57, 35Q92

Acknowledgments. The authors would like to thank the University of Ottawa for graduate scholarships to the first author, and the Natural Science and Engineering Research Council (NSERC) of Canada for a research grant to the second author. Research exchanges between France and Canada were funded by a grant from the Agence Nationale de la Recherche of France (ANR project HR-CEM no. 13-MONU-0004-01).

[*]thomas.roy@maths.ox.ac.uk
[†]ybourg@uottawa.ca
[‡]charles.pierre@univ-pau.fr

Introduction

The modelling of the electrical activity of the heart offers an interesting perspective on the understanding of cardiac pathologies and its treatments. This subject has great potential in biomedical sciences, as experiments on living hearts require considerable resources and provide only a partial picture of the electrical activity of the heart. For instance, realistically simulating the behaviour of the heart reduces the necessity for these kinds of experiments. Considering that heart diseases are a leading cause of death in Western countries [5], modelling in cardiac electrophysiology has the potential to significantly impact society. Also, this area has achieved great strides in the past few years, because of improved models, calculation methods and computer power [6, 4, 31].

The mono- and bi- domain models allow for an adequate modelling of the electrical activity in the heart tissue [13, 27]. These models form a complex system of partial differential equations (PDE) that are coupled with a system of ordinary differential equations (ODE) describing the ionic activity in cardiac cells. Solving these models numerically requires large computational time and computations are limited to few heartbeats on ventricles and atria.

The ODE systems describing ionic activity vary in terms of complexity depending on the physiological processes accounted for at the cell level. These ionic models, coupled with the mono- or bi- domain model, require discretization in space and time. This article focuses on various types of time-stepping methods: explicit, semi-implicit, operator splitting, exponential, and deferred correction methods. The monodomain model is a reaction-diffusion equation and solving the diffusion part of the model explicitly results in severe stability constraints on the time-step. Therefore, the use of semi-implicit or implicit methods is viewed as crucial by most authors [7, 29, 14]. To achieve this, operator splitting methods have been considered by some authors [27, 22, 30, 11]. Alternatively, an exponential method called the Rush-Larsen (RL) method was developed specifically to solve the ODEs resulting from the ionic activity of excitable cells, providing very stable numerical solutions [21]. This method is very popular and several other variations have been considered, including high-order RL methods [18, 16, 26].

The different time-stepping methods have been thoroughly studied for non-spatial ionic models [23, 24], i.e. only at the cell level, and relatively little at the level of the heart tissue for the mono- or bi-domain models. To date, no research in cardiac electrophysiology has compared a large number of time-stepping methods thoroughly for spatial models. In addition, first-order numerical methods such as Euler's method are still widely used in this field of research due to their easy implementation [19]. We seek to show that these first-order methods are very inefficient compared to high-order methods. Furthermore, we will consider a general ionic model for our theoretical stability analysis, allowing other researchers to use these results to determine critical time-steps for stability for the model of their choice. Earlier studies are specific to a given ionic model, which is limiting in a research area where models are constantly evolving. Thus, this research will provide a summary of the various methods that had not yet been compared, both for stability and accuracy.

The goal of this article is to find optimal time-stepping methods for ionic models of increasing stiffness, where optimality is judged based on the cost to reach a given accuracy.

In Section 1, we will start by choosing a number of ionic models that cover the range of complexity, stiffness and realism for these types of models. In Section 2, we will list a number of viable numerical methods for the monodomain model. The different methods will be studied with theoretical analysis and numerical tests in the later section. In Section 3, different stability analysis will be conducted for each numerical method. We will establish a technique to study the stability of the numerical methods for the monodomain model in an ODE setting. To ease the comparison from one model to another, a general ionic model will be used. The theoretical time-steps will then be compared with the numerically observed critical time-step of the different methods. Then, in Section 4, a convergence test will be performed to check whether the methods of high order exhibit the correct rate of convergence when used to solve the more complex ionic models. The accuracy of the methods studied will then be compared with respect to their accuracy, relatively to the size of the time-step and the computational time.

1 Models in cardiac electrophysiology

1.1 Modelling the ionic activity in cells

There exist many different models used to describe the ionic activity in cardiac cells, varying in terms of complexity and stiffness. Here, we choose to study three models that cover the whole spectrum.

The Mitchell-Schaeffer (MS) model is a phenomenological two-variable model proposed in [17], which can be easily understood analytically and is very efficient for numerical simulations. In spite of its very low complexity, this model still accurately represents the main characteristics of the cardiac action potential.

One of the physiological ionic models used in this paper is the Beeler-Reuter (BR) model [3]. This model has six gating variables and one concentration, besides the transmembrane potential. The source term in the potential equation includes four different ionic currents.

A more complex ionic model used in this article is the ten Tuscher-Noble-Noble-Panfilov (TNNP) model from [28]. There exist different versions of this model for different types of cells, and here we consider the epicardial cells. This version is the stiffest of the original TNNP models. This model has seventeen variables, including twelve gating variables and four concentrations. It also has 15 different ionic currents.

Physiological models can be written in the following form, where u is the transmembrane potential, $\mathbf{v}$ is the vector of gating variables, $\mathbf{X}$ is the vector of ionic concentrations, and t is the time:

$$\frac{du}{dt} = I(u, \mathbf{v}, \mathbf{X}, t), \tag{1}$$

$$\frac{d\mathbf{v}}{dt} = \mathbf{f}(u, \mathbf{v}), \tag{2}$$

$$\frac{d\mathbf{X}}{dt} = \mathbf{g}(u, \mathbf{v}, \mathbf{X}). \tag{3}$$

The total current I acts as a source/sink term and is defined as

$$I(u, \mathbf{v}, \mathbf{X}, t) = \frac{1}{C_m}(I_{\text{app}}(t) - I_{\text{ion}}(u, \mathbf{v}, \mathbf{X})), \tag{4}$$

where C_m is the membrane capacity, I_{app} is the applied stimulation current, and I_{ion} is the sum of all ionic currents across the cell membrane. In general, the r.h.s. f_i's in (2) mimic Hodgkin-Huxley type equations [12]:

$$f_i(u, v_i) = \frac{v_{i,\infty}(u) - v_i}{\tau_{v_i}(u)}, \tag{5}$$

where $v_{i,\infty}$ is a steady state value for v_i and τ_{v_i} is a time constant, both specific to the ionic model and the gating variable. The function $\mathbf{g}$ in (3) is specific to the ionic model.

1.2 Modelling the Heart Tissue

After modelling the electrical activity at the cell level, we now extend our models to the tissue level. For our test cases, we consider the monodomain model with constant extra- and intra-cellular conductivities, coupled with a general ionic cell model. For the monodomain model, we assume equal anisotropy ratio of the conductivities. The monodomain model reads as:

$$\frac{\partial u}{\partial t} = I(u, \mathbf{v}, \mathbf{X}, t, x) + \text{div}(\sigma \nabla u), \tag{6}$$

$$\frac{\partial \mathbf{v}}{\partial t} = \mathbf{f}(u, \mathbf{v}), \tag{7}$$

$$\frac{\partial \mathbf{X}}{\partial t} = \mathbf{g}(u, \mathbf{v}, \mathbf{X}), \tag{8}$$

where $\sigma = \frac{\lambda}{1+\lambda}\sigma_i/\chi C_m$, λ is the equal anisotropy ratio, σ_i is the intra-cellular conductivity tensor and χ is the cellular membrane area per unit volume of cardiac tissue. The applied stimulation current I_{app} in (4) can now vary through space, denoted by $x \in \Omega$, the spatial domain. We denote by $\mathbf{f}$ and $\mathbf{g}$ the vectors of f_i's and g_i's, respectively. In some cases, it is simpler to group in $\mathbf{v}$ the vector of v_i's and X_i's, and in $\mathbf{f}$ the vector of f_i's and g_i's.

We impose homogeneous Neumann boundary conditions, i.e. $\nabla u \cdot n = 0$ on $\partial\Omega$. This amounts to having no current leaking from the heart to the surrounding tissues. The initial condition is taken as the resting state of the chosen ionic model.

2 Numerical methods

We consider a variety of explicit and semi-implicit methods of first to third-order, using a constant time-step Δt, for the time discretization. When solving reaction-diffusion equations as the monodomain model, some authors suggest using semi-implicit methods with implicit diffusion and explicit reaction terms [7, 18, 27]. Implicit methods are usually too expensive for many ionic models and explicit methods have additional stability restrictions.

Let us introduce the notation for the fully discretized version of the problem (6)-(8). Given a spatial mesh with nodes x_i, $i = 0, 1, \ldots, N$ and a temporal mesh with equally spaced temporal nodes $t_n = n\Delta t$, $n \geq 0$, we denote by

$$U^n = [U_0^n, \ldots, U_N^n]^\top \simeq [u(x_0, t_n), \ldots, u(x_N, t_n)]^\top, \tag{9}$$

$$V^n = [V_{1,0}^n, \ldots, V_{p,0}^n, \ldots, V_{p,N}^n]^\top \simeq [v_1(x_0, t_n), \ldots, v_p(x_0, t_n), \ldots, v_p(x_N, t_n)]^\top, \tag{10}$$

$$X^n = [X_{1,0}^n, \ldots, X_{q,0}^n, \ldots, X_{q,N}^n]^\top \simeq [X_1(x_0, t_n), \ldots, X_q(x_0, t_n), \ldots, X_q(x_N, t_n)]^\top, \tag{11}$$

the approximate values for u, $\mathbf{v}$ and $\mathbf{X}$, respectively.

We write the equations for the finite difference and finite element methods. The term AU^n is the discrete version of $\mathrm{div}(\sigma \nabla u)$. For the finite difference method, A is a discretized Laplacian, and for the finite element method, $A = M^{-1}S$, where M is the mass matrix and S is the stiffness matrix.

For most methods, the gating variables and concentrations are treated the same way. Therefore, if the following schemes do not have X and G, it is implied they are included in V and F, respectively.

2.1 First-order methods

(i) Forward Euler (FE):

$$\begin{aligned} \frac{U^{n+1} - U^n}{\Delta t} &= I(U^n, V^n, t_n, x) + AU^n, \\ \frac{V^{n+1} - V^n}{\Delta t} &= F(U^n, V^n). \end{aligned} \tag{12}$$

(ii) Forward-Backward Euler (FBE):

$$\begin{aligned} \frac{U^{n+1} - U^n}{\Delta t} &= I(U^n, V^n, t_n, x) + AU^{n+1}, \\ \frac{V^{n+1} - V^n}{\Delta t} &= F(U^n, V^n). \end{aligned} \tag{13}$$

(iii) Rush-Larsen with Forward-Backward Euler (RL-FBE) [21, 18]:

$$\begin{aligned} \frac{U^{n+1} - U^n}{\Delta t} &= I(U^n, V^n, X^n, t_n, x) + AU^{n+1}, \\ \frac{V_i^{n+1} - V_i^n}{\Delta t} &= \Phi(a_i^n \Delta t)(a_i^n V_i^n + b_i^n), \quad i = 1, \ldots, p, \\ \frac{X^{n+1} - X^n}{\Delta t} &= G(U^n, V^n, X^n), \end{aligned} \tag{14}$$

where Φ, a_i^n and b_i^n are given below. This method corresponds to the case $c_{-1} = 0$, $c_0 = 1$, $c_1 = 0$ in the general RL methods described at the end of this section.

2.2 Second-Order Methods

(i) Second-order semi-implicit backward differentiation (SBDF2) [2, 7]:

$$\begin{aligned}\frac{\frac{3}{2}U^{n+1}-2U^n+\frac{1}{2}U^{n-1}}{\Delta t} &= 2I(U^n,V^n,t_n,x)-I(U^{n-1},V^{n-1},t_{n-1},x)+AU^{n+1},\\ \frac{\frac{3}{2}V^{n+1}-2V^n+\frac{1}{2}V^{n-1}}{\Delta t} &= 2F(U^n,V^n)-F(U^{n-1},V^{n-1}).\end{aligned} \tag{15}$$

(ii) Strang Splitting with Crank-Nicolson and Runge-Kutta 2 (CN-RK2) [25, 27]:
For this method, we do half a time-step of RK2 solely on the reaction part of the monodomain equations, followed by a step of CN on the diffusion part and another half-step of RK2 on the reaction part. Here we denote $Y^n = \begin{bmatrix}U^n\\V^n\end{bmatrix}$, $Y^* = \begin{bmatrix}U^*\\V^*\end{bmatrix}$ and $Y^{**} = \begin{bmatrix}U^{**}\\V^{**}\end{bmatrix}$.

Step 1:

$$\frac{Y^*-Y^n}{\Delta t/2} = \begin{bmatrix} I\left(Y^n+\frac{\Delta t}{4}\begin{bmatrix}I(Y^n,t_n,x)\\F(Y^n)\end{bmatrix},t_n+\frac{\Delta t}{4},x\right)\\ F\left(Y^n+\frac{\Delta t}{4}\begin{bmatrix}I(Y^n,t_n,x)\\F(Y^n)\end{bmatrix}\right)\end{bmatrix}. \tag{16}$$

Step 2:

$$\frac{U^{**}-U^*}{\Delta t} = \frac{1}{2}A(U^{**}+U^*). \tag{17}$$

Step 3:

$$\frac{Y^{n+1}-Y^{**}}{\Delta t/2} = \begin{bmatrix} I\left(Y^{**}+\frac{\Delta t}{4}\begin{bmatrix}I(Y^{**},t_n+\Delta t/2,x)\\F(Y^{**})\end{bmatrix},t_n+\frac{3}{4}\Delta t,x\right)\\ F\left(Y^{**}+\frac{\Delta t}{4}\begin{bmatrix}I(Y^{**},t_n+\Delta t/2,x)\\F(Y^{**})\end{bmatrix}\right)\end{bmatrix}. \tag{18}$$

(iii) Strang Splitting with Crank-Nicolson and Runge-Kutta 4 (CN-RK4):
This scheme is written as the CN-RK2 scheme, but using instead the fourth-order Runge-Kutta method (see [20] for more details).

(iv) Second-order Rush-Larsen with Crank-Nicolson Adam-Bashforth (RL-CNAB) [21, 18]:

$$\begin{aligned}\frac{U^{n+1}-U^n}{\Delta t} &= \frac{3}{2}I(U^n,V^n,X^n,t_n,x)-\frac{1}{2}I(U^{n-1},V^{n-1},X^{n-1},t_{n-1},x)+\frac{1}{2}A(U^{n+1}+U^n),\\ \frac{V_i^{n+1}-V_i^n}{\Delta t} &= \Phi(a_i^{n+\frac{1}{2}}\Delta t)(a_i^{n+\frac{1}{2}}V_i^n+b_i^{n+\frac{1}{2}}),\quad i=1,\dots,p,\\ \frac{X^{n+1}-X^n}{\Delta t} &= \frac{3}{2}G(U^n,V^n,X^n)-\frac{1}{2}G(U^{n-1},V^{n-1},X^{n-1}),\end{aligned} \tag{19}$$

where Φ, a_i^n and b_i^n are given below. This method corresponds to the case $c_{-1}=0$, $c_0=\frac{3}{2}$, $c_1=-\frac{1}{2}$ in the general RL methods described at the end of this section.

2.3 Third-Order Methods

(i) Third-Order Deferred Correction (DC3) [15, 8]:
Here we denote $Y^n = \begin{bmatrix} U^n \\ V^n \end{bmatrix}$ and $Y_i^n = \begin{bmatrix} U_i^n \\ V_i^n \end{bmatrix}$. The initial values for the substeps are $Y_0^0 = Y_0$, $Y_1^0 = 0$, and $Y_2^0 = 0$. This method proceeds with three substeps:

$$\text{for } n \geq 0 \quad \begin{cases} ml_0^{n+1} = I(Y_0^n, t_n, x), \quad nl_0^{n+1} = F(Y_0^n), \\ \dfrac{U_0^{n+1} - U_0^n}{\Delta t} = ml_0^{n+1} + AU_0^{n+1}, \quad \dfrac{V_0^{n+1} - V_0^n}{\Delta t} = nl_0^{n+1}, \\ dY_0^{n+1} = (Y_0^{n+1} - Y_0^n)/\Delta t \end{cases} \tag{20}$$

$$\text{for } n \geq 1 \quad \begin{cases} d^2U_0^{n+1} = (dU_0^{n+1} - dU_0^n)/\Delta t, \quad ml_1^n = I(Y_0^n + \Delta t Y_1^{n-1}, t_n, x), \\ \dfrac{U_1^n - U_1^{n-1}}{\Delta t} = AU_1^n - \dfrac{1}{2}d^2U_0^{n+1} + \dfrac{ml_1^n - ml_0^n}{\Delta t}, \\ d^2V_0^{n+1} = (dV_0^{n+1} - dV_0^n)/\Delta t, \quad nl_1^n = F(Y_0^n + \Delta t Y_1^{n-1}), \\ \dfrac{V_1^n - V_1^{n-1}}{\Delta t} = -\dfrac{1}{2}d^2V_0^{n+1} + \dfrac{nl_1^n - nl_0^n}{\Delta t}, \\ dY_1^n = (Y_1^n - Y_1^{n-1})/\Delta t, \end{cases} \tag{21}$$

$$\text{for } n \geq 2 \quad \begin{cases} d^2U_1^n = (dU_1^n - dU_1^{n-1})/\Delta t, \quad d^3U_0^{n+1} = (d^2U_0^{n+1} - d^2U_0^n)/\Delta t, \\ ml_2^{n-1} = I(Y_0^{n-1} + \Delta t Y_1^{n-1} + \Delta t^2 Y_2^{n-2}, t_n, x), \\ \dfrac{U_2^{n-1} - U_2^{n-2}}{\Delta t} = AU_2^{n-1} - \dfrac{1}{2}d^2U_1^n + \dfrac{1}{6}d^3U_0^{n+1} + \dfrac{ml_2^{n-1} - ml_1^{n-1}}{\Delta t^2}, \\ d^2V_1^n = (dV_1^n - dV_1^{n-1})/\Delta t \quad d^3V_0^{n+1} = (d^2V_0^{n+1} - d^2V_0^n)/\Delta t, \\ nl_2^{n-1} = F(Y_0^{n-1} + \Delta t Y_1^{n-1} + \Delta t^2 Y_2^{n-2}), \\ \dfrac{V_2^{n-1} - V_2^{n-2}}{\Delta t} = -\dfrac{1}{2}d^2V_1^n + \dfrac{1}{6}d^3V_0^{n+1} + \dfrac{nl_2^{n-1} - nl_1^{n-1}}{\Delta t^2}, \\ Y^{n-1} = Y_0^{n-1} + \Delta t Y_1^{n-1} + \Delta t^2 Y_2^{n-1}. \end{cases} \tag{22}$$

It is possible to obtain a second order method by solving the first two substeps and defining $Y^{n-1} = Y_0^{n-1} + \Delta t Y_1^{n-1}$. We only provide numerical results for the third-order DC method.

2.4 The Rush-Larsen Method

A very popular method for solving physiological models is the first-order scheme proposed by Rush and Larsen [21]. Outside of cardiac electrophysiology, this method is known as the explicit exponential Euler method. The Rush-Larsen (RL) method solves problems of the form $\frac{dy}{dt} = a(t)y + b(t)$.

Perego and Veneziani [18] introduced a way to increase the order of the scheme, by taking a and b at time $t_{n+\frac{1}{2}}$,

$$\begin{cases} y^{n+1} = y^n + \Delta t \Phi(a^{n+\frac{1}{2}}\Delta t)(a^{n+\frac{1}{2}}y^n + b^{n+\frac{1}{2}}), \quad n = 0, \dots, N_t, \\ y(0) = y^0, \end{cases} \tag{23}$$

where

$$\Phi(x) = \begin{cases} \dfrac{e^x - 1}{x}, & x \neq 0, \\ 1, & x = 0, \end{cases}$$

$a^{n+\frac{1}{2}}$ and $b^{n+\frac{1}{2}}$ are approximations of $a(t_{n+\frac{1}{2}})$ and $b(t_{n+\frac{1}{2}})$ taken as

$$\begin{aligned} &a^{n+\frac{1}{2}} = c_{-1}a^{n+1} + c_0 a^n + c_1 a^{n-1}, \quad b^{n+\frac{1}{2}} = c_{-1}b^{n+1} + c_0 b^n + c_1 b^{n-1}, \ n = 1, \dots, N_t, \\ &a^{\frac{1}{2}} = c_{-1}a^1 + (c_0 + c_1)a^0, \quad b^{\frac{1}{2}} = c_{-1}b^1 + (c_0 + c_1)b^0, \end{aligned} \tag{24}$$

where c_{-1}, c_0 and c_1 are coefficients to be determined.

For the transmembrane potential u and concentrations $\mathbf{X}$, we take $a = 0$, which simply results in the Euler and Adams-Bashforth methods for the first and second order cases, respectively. In each case we also solve the diffusion implicitly, hence resulting in the FBE and CNAB methods for the transmembrane potential and concentrations.

For the gating variables, we write the function $\mathbf{f}$ in (5) as $f_i(u, \mathbf{v}, \mathbf{X}) = a_i(u)v_i + b_i(u)$. For the BR and TNNP models, we have:

$$a_i(u) = \frac{-1}{\tau_{v_i}(u)}, \quad \text{and} \quad b_i(u) = \frac{v_{i,\infty}(u)}{\tau_{v_i}(u)}. \tag{25}$$

3 Stability analysis

We now find stability conditions for the time-stepping methods introduced in the previous section.

3.1 Absolute Stability

The stability analysis will be done using the concept of absolute stability for ODE solvers. We linearize our PDE and then consider the linear ODE system resulting from the linearized problem semi-discretized in space. Since we will carry von Neumann stability analysis, we will consider 1D reaction-diffusion equations on the whole real line discretized in space by centred finite difference schemes. We find the stability region of each method for the linearized problem. This analysis will confirm that the stability conditions of semi-implicit methods applied to the monodomain model coincide with their explicit equivalent applied solely to the ionic models. The largest possible time-step Δt for which $\lambda \Delta t$ is in the stability region for all λ is called the *critical* time-step, where λ are the eigenvalues of the linearized problem.

Considering (6)-(8), but denoting by $\mathbf{v}$ the vector of v_i's and X_i's, and $\mathbf{f}$ the vector of f_i's and g_i's, we linearize around some state $(\tilde{u}, \tilde{\mathbf{v}})$. This state is constant in space and time, and will be determined later. We obtain

$$\frac{\partial u}{\partial t} = \frac{\partial I}{\partial u}(\tilde{u}, \tilde{\mathbf{v}})u + \frac{\partial I}{\partial \mathbf{v}}(\tilde{u}, \tilde{\mathbf{v}})\mathbf{v} + \operatorname{div}(\sigma \nabla u), \tag{26}$$

$$\frac{\partial \mathbf{v}}{\partial t} = \frac{\partial \mathbf{f}}{\partial u}(\tilde{u}, \tilde{\mathbf{v}})u + \frac{\partial \mathbf{f}}{\partial \mathbf{v}}(\tilde{u}, \tilde{\mathbf{v}})\mathbf{v}. \tag{27}$$

For the sake of the stability analysis, we consider uniformly spaced grid points $x_j = jh$ for each $j \in \mathbb{Z}$, $h > 0$, the space step. We denote

$$U = [\ldots, U_{-1}, U_0, U_1, \ldots]^\top \simeq [\ldots, u(x_{-1}, t), u(x_0, t), u(x_1, t), \ldots]^\top, \tag{28}$$

$$V = [\ldots, V_{-1}, V_0, V_1, \ldots]^\top \simeq [\ldots, \mathbf{v}(x_{-1}, t), \mathbf{v}(x_0, t), \mathbf{v}(x_1, t), \ldots]^\top. \tag{29}$$

Discretizing (26) and (27) we obtain

$$\frac{d}{dt}\begin{bmatrix} U_j \\ V_j \end{bmatrix} = \begin{bmatrix} \frac{\partial I}{\partial u}(\tilde{u}, \tilde{\mathbf{v}}) & \frac{\partial I}{\partial \mathbf{v}}(\tilde{u}, \tilde{\mathbf{v}}) \\ \frac{\partial \mathbf{f}}{\partial u}(\tilde{u}, \tilde{\mathbf{v}}) & \frac{\partial \mathbf{f}}{\partial \mathbf{v}}(\tilde{u}, \tilde{\mathbf{v}}) \end{bmatrix} \begin{bmatrix} U_j \\ V_j \end{bmatrix} + \sigma \begin{bmatrix} \frac{U_{j+1} - 2U_j + U_{j-1}}{h^2} \\ 0 \end{bmatrix}, \tag{30}$$

for each $j \in \mathbb{Z}$. As is usually done for von Neumann stability analysis, we set $U_j(t) = U_\omega(t)e^{i\omega jh}$ and $V_j(t) = V_\omega(t)e^{i\omega jh}$, for all $\omega \in [0, \frac{2\pi}{h}]$. Then

$$\frac{d}{dt}\begin{bmatrix} U_\omega \\ V_\omega \end{bmatrix} = (J + A_\omega)\begin{bmatrix} U_\omega \\ V_\omega \end{bmatrix}, \tag{31}$$

where J is the Jacobian matrix evaluated at $(\tilde{u}, \tilde{\mathbf{v}})$ as in (30) and A_ω is a diffusion matrix given by

$$A_\omega = \frac{2\sigma}{h^2}\begin{bmatrix} \cos(\omega h) - 1 & 0 \\ 0 & 0 \end{bmatrix}. \tag{32}$$

Equation (31) is discretized with the time-stepping methods of Section 2. A stability function $R = R(\Delta t, h, \omega))$ is then defined for each method. The condition for stability is that the spectral radius $\rho(R) \leq 1$, assuming that all the eigenmodes of $J + A_\omega$ satisfy $\operatorname{Re}(\lambda) < 0$.

3.1.1 Forward Euler

We now discretize in time and apply the FE scheme to (31). We obtain

$$\begin{bmatrix} U_\omega^{n+1} \\ V_\omega^{n+1} \end{bmatrix} = (I_d + \Delta t(J + A_\omega))\begin{bmatrix} U_\omega^n \\ V_\omega^n \end{bmatrix}, \tag{33}$$

where I_d is the identity matrix.

The stability condition is $|1 + \Delta t\lambda| \leq 1$ for all λ eigenvalues of $(J + A_\omega)$. Computing numerically the spectrum of $(J + A_\omega)$, we noticed that this condition is the most stringent

for the most negative eigenvalue $\lambda_{\min}$ of $J + A_\omega$. The condition for stability is then $\Delta t \leq -2/\lambda_{\min}$. Because this eigenvalue depends on ω, we will consider the wave number ω that makes this inequality most restrictive (in the sense that it makes $-2/\lambda_{\min}$ the smallest), i.e. $\omega = \pi/h$, a fact that can be verified numerically. Therefore our critical time-step for the stability of the FE method is $\Delta t^*_{\text{theo}} = -2/\lambda_{\min}$, where $\lambda_{\min}$ is the minimum eigenvalue of

$$\begin{bmatrix} \dfrac{\partial I}{\partial u}(\tilde{u}, \tilde{\mathbf{v}}) - \dfrac{4\sigma}{h^2} & \dfrac{\partial I}{\partial \mathbf{v}}(\tilde{u}, \tilde{\mathbf{v}}) \\ \dfrac{\partial \mathbf{f}}{\partial u}(\tilde{u}, \tilde{\mathbf{v}}) & \dfrac{\partial \mathbf{f}}{\partial \mathbf{v}}(\tilde{u}, \tilde{\mathbf{v}}) \end{bmatrix}. \tag{34}$$

To determine $\lambda_{\min}$, we calculate numerically the solution of the problem on a fine grid. We then evaluate the matrix $J + A_\omega$ at each node of our domain, and for each of these matrices, we calculate their eigenvalues. We choose the most negative eigenvalue as our $\lambda_{\min}$. The constant state $(\tilde{u}, \tilde{\mathbf{v}})$ is then chosen as the numerical solution evaluated at this node. The most negative eigenvalue typically appears during the resting state of the solution, but for some models, the stimulation current can sometimes make this eigenvalue more negative. The theoretical critical time-steps are shown in Table 2 for the BR model, Table 5 for the MS model and in Table 8 for the TNNP model. For small values of h, the critical time-step is proportional to h^2.

3.1.2 Forward-Backward Euler

We define the stability function $R(\Delta t, h, \omega) = (I - \Delta t A_\omega)^{-1}(I_d + \Delta t J)$. One can easily see that the matrix $(I_d - \Delta t A_\omega)^{-1}$ is diagonal with value $\left(1 - \Delta t \sigma \dfrac{2\cos \omega h - 2}{h^2}\right)^{-1}$ in the first position and 1 on the rest of the diagonal, implying that the spectral radius $\rho((I_d - \Delta t A_\omega)^{-1}) = 1$. This yields $\rho(R(\Delta t, h, \omega)) \leq \rho(I_d + \Delta t J)$, which inequality is sharp when $\omega = 0$. This removes the dependence on h in the stability condition.

The condition for stability then becomes $|1 + \Delta t \lambda| \leq 1$ for all λ eigenvalues of J. Therefore, similarly to the FE method, our critical time-step for FBE is $\Delta t^*_{\text{theo}} = -2/\lambda_{\min}$, where $\lambda_{\min}$ is the most negative eigenvalue of J. As explained for the previous method, we calculate J at each node of a fine grid and we choose its most negative eigenvalue as our $\lambda_{\min}$. The constant state $(\tilde{u}, \tilde{\mathbf{v}})$ is then chosen as the numerical solution evaluated at that node. The critical time-steps for the BR, MS and TNNP models are shown in Tables 2, 5 and 7, respectively.

On coarse meshes, FE and FBE are expected to be stable for the same time-step Δt, but as the grid is refined, FE's stability deteriorates while FBE's remains unchanged. In fact, the diffusion term taken implicitly in the FBE method makes the critical time-step independent of the grid size h.

3.1.3 Strang Splitting

We use the technique presented above to look at the stability of the Strang splitting scheme. Let us start with the CN-RK2 method.

Step 1:

$$\begin{bmatrix} U_\omega^* \\ \mathbf{V}_\omega^* \end{bmatrix} = \left(I_d + \frac{\Delta t}{2} J + \frac{(\Delta t/2)^2}{2} J^2 \right) \begin{bmatrix} U_\omega^n \\ \mathbf{V}_\omega^n \end{bmatrix}. \tag{35}$$

Step 2:

$$\begin{bmatrix} U_\omega^{**} \\ \mathbf{V}_\omega^{**} \end{bmatrix} = \left(I_d - \frac{1}{2}\Delta t A_\omega \right)^{-1} \left(\frac{1}{2}\Delta t A_\omega + I_d \right) \begin{bmatrix} U_\omega^* \\ \mathbf{V}_\omega^* \end{bmatrix}. \tag{36}$$

Step 3:

$$\begin{bmatrix} U_\omega^{n+1} \\ \mathbf{V}_\omega^{n+1} \end{bmatrix} = \left(I_d + \frac{\Delta t}{2} J + \frac{(\Delta t/2)^2}{2} J^2 \right) \begin{bmatrix} U_\omega^{**} \\ \mathbf{V}_\omega^{**} \end{bmatrix}. \tag{37}$$

Combining (35), (36) and (37) we obtain

$$\begin{bmatrix} U_\omega^{n+1} \\ \mathbf{V}_\omega^{n+1} \end{bmatrix} = \left(I_d + \frac{\Delta t}{2} J + \frac{(\Delta t/2)^2}{2} J^2 \right) \left(I_d - \frac{1}{2}\Delta t A_\omega \right)^{-1} \left(\frac{1}{2}\Delta t A_\omega + I_d \right) \left(I_d + \frac{\Delta t}{2} J + \frac{(\Delta t/2)^2}{2} J^2 \right) \begin{bmatrix} U_\omega^n \\ \mathbf{V}_\omega^n \end{bmatrix}. \tag{38}$$

For stability we need $\rho(R(\Delta t, h, \omega)) \leq 1$. Noticing that

$$\rho\left(\left(I_d - \frac{1}{2}\Delta t A_\omega \right)^{-1} \left(\frac{1}{2}\Delta t A_\omega + I_d \right) \right) \leq 1, \tag{39}$$

the condition for stability becomes

$$\rho\left(I_d + \frac{\Delta t}{2} J + \frac{(\Delta t/2)^2}{2} J^2 \right)^2 \leq 1, \tag{40}$$

independently of h.

If J is diagonalizable, (40) is equivalent to

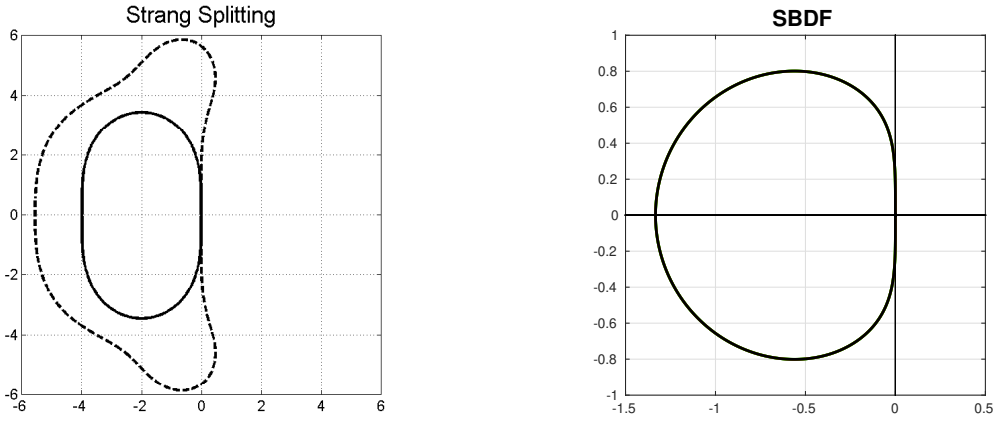
$$\left| 1 + \Delta t \lambda_i/2 + \frac{(\Delta t \lambda_i/2)^2}{2} \right| \leq 1, \tag{41}$$

for all eigenvalue λ_i of J. As seen with the previous methods, and with the contour of the stability region shown in Figure 1a, the most restrictive eigenvalue will be the most negative. Using this most negative eigenvalue we obtain the critical time-steps shown in Tables 2, 5 and 7.

With a similar derivation, a condition for stability of the CN-RK4 method is obtained:

$$\left| 1 + \Delta t \lambda_i/2 + \frac{(\Delta t \lambda_i/2)^2}{2} + \frac{(\Delta t \lambda_i/2)^3}{6} + \frac{(\Delta t \lambda_i/2)^4}{24} \right| \leq 1, \tag{42}$$

for all eigenvalue λ_i of J. Again, we solve for the stability contour and get the critical time-steps Δt^*_{theo} in Tables 2, 5 and 7. The stability region of CN-RK4 is shown in Figure 1a and contains the stability region of CN-RK2.



(a) CN-RK4 (dotted), CN-RK2 (solid) (b) SBDF2 method

Figure 1: Stability Regions

3.1.4 Second-Order Semi-Implicit Backward Differentiation

Following the approach used previously, we write the SBDF2 scheme as:

$$\frac{1}{\Delta t}\left(\frac{3}{2}\begin{bmatrix}U_\omega^{n+1}\\V_\omega^{n+1}\end{bmatrix}-2\begin{bmatrix}U_\omega^{n}\\V_\omega^{n}\end{bmatrix}+\frac{1}{2}\begin{bmatrix}U_\omega^{n-1}\\V_\omega^{n-1}\end{bmatrix}\right)=J\left(2\begin{bmatrix}U_\omega^{n}\\V_\omega^{n}\end{bmatrix}-\begin{bmatrix}U_\omega^{n-1}\\V_\omega^{n-1}\end{bmatrix}\right)+A_\omega\begin{bmatrix}U_\omega^{n+1}\\V_\omega^{n+1}\end{bmatrix}. \quad (43)$$

For simplicity, we denote $Y^j=\begin{bmatrix}U_\omega^{j}\\V_\omega^{j}\end{bmatrix}$, for all $j\in\mathbb{Z}$. Equation (43) becomes

$$(\frac{3}{2}I_d-A_\omega)Y^{n+1}-(2I_d+2\Delta tJ)Y^n+(\frac{1}{2}I_d+\Delta tJ)Y^{n-1}=0. \quad (44)$$

As done for the previous semi-implicit methods, we take $\omega=0$, which leads to a stability condition independent of h. See the remark below for a proof that this value is the most restrictive for the stability of this method. Equation (44) becomes

$$\frac{3}{2}Y^{n+1}-(2I_d+2\Delta tJ)Y^n+(\frac{1}{2}I_d+\Delta tJ)Y^{n-1}=0. \quad (45)$$

Equation (45) is solved using the Lagrange's method found in [9]. We set $Y^j=\zeta^jW$, where W is an eigenvector of J with eigenvalue λ. We divide by ζ^{n-1} and obtain the following equation

$$\frac{3}{2}\zeta^2W-(2I_d+2\Delta tJ)\zeta W+(\frac{1}{2}I_d+\Delta tJ)W=0, \quad (46)$$

which implies

$$\frac{3}{2}\zeta^2-(2+2\Delta t\lambda)\zeta+(\frac{1}{2}+\Delta t\lambda)=0. \quad (47)$$

Equation (45) has stable solutions *iff* for any eigenvalue λ, all simple roots $\zeta(\lambda\Delta t)$ of (47) satisfy $|\zeta(\lambda\Delta t)|\leq 1$, and additional multiple roots must satisfy $|\zeta(\lambda\Delta t)|<1$ [10]. The stability region of the method is defined as

$$S=\left\{\mu\in\mathbb{C}\ ;\ \begin{array}{l}\text{all simple roots } \zeta(\mu) \text{ of (47) satisfy } |\zeta(\mu)|\leq 1,\\ \text{multiple roots satisfy } |\zeta(\mu)|<1\end{array}\right\}. \quad (48)$$

Solving for $\mu = \Delta t\lambda$ in (47), we get that our method is stable for

$$\mu = \frac{\frac{-3}{2}\zeta^2 + 2\zeta - \frac{1}{2}}{1 - 2\zeta}, \qquad \text{with} \quad |\zeta| \leq 1. \tag{49}$$

We take $\zeta = e^{i\theta}$ for $0 \leq \theta \leq 2\pi$ to find the contour of the stability region:

$$\mu = \frac{\frac{-3}{2}e^{2i\theta} + 2e^{i\theta} - \frac{1}{2}}{1 - 2e^{i\theta}}, \quad \theta \in [0, 2\pi]. \tag{50}$$

The contour of the stability region is shown in Figure 1b. As with the previous methods, the most restrictive eigenvalue will be the most negative. This occurs for a real eigenvalue, hence $-4/3 \leq \lambda\Delta t \leq 0$ for stability. The critical time-step is then $\Delta t^*_{\text{theo}} = \dfrac{-4}{3\lambda_{\min}}$ and its value is shown in Table 2 for the BR model, in Table 5 for the MS model and in Table 7 for the TNNP model.

Remark 1 (Justification for the choice of ω). We rewrite equation (44) as a one-step recursive equation:

$$\begin{bmatrix} Y^{n+1} \\ Y^n \end{bmatrix} = \begin{bmatrix} (\frac{3}{2}I_d - A_\omega)^{-1}(2I_d + 2\Delta tJ) & (\frac{3}{2}I_d - A_\omega)^{-1}(-\frac{1}{2}I_d - \Delta tJ) \\ I_d & 0 \end{bmatrix} \begin{bmatrix} Y^n \\ Y^{n-1} \end{bmatrix}. \tag{51}$$

Let

$$R(\Delta t, h, \omega) = \begin{bmatrix} (\frac{3}{2}I_d - A_\omega)^{-1}(2I_d + 2\Delta tJ) & (\frac{3}{2}I_d - A_\omega)^{-1}(-\frac{1}{2}I_d - \Delta tJ) \\ I_d & 0 \end{bmatrix}, \tag{52}$$

be the stability function for the SBDF2 method. We can factor out from R the matrix

$$\begin{bmatrix} (I_d - \frac{2}{3}A_\omega)^{-1} & 0 \\ 0 & I_d \end{bmatrix}, \tag{53}$$

which has a spectral radius of 1. This leads to the stability condition $\rho(\tilde{R}) \leq 1$, where

$$\tilde{R}(\Delta t, h, \omega) = \begin{bmatrix} \frac{2}{3}(2I_d + 2\Delta tJ) & \frac{2}{3}(-\frac{1}{2}I_d - \Delta tJ) \\ I_d & 0 \end{bmatrix}. \tag{54}$$

The matrix $\tilde{R}$ appears in the one-step formulation of the difference equation (45).

3.1.5 Rush-Larsen

We now look at the stability of the RL-FBE and the RL-CNAB schemes. These schemes differ from the methods previously studied in that the differential equations for the gating variables are solved using a different method than for the transmembrane potential and the concentrations. The major advantage of using Rush-Larsen methods is to be able to use large time-steps compared to more classical methods. In fact, it can be observed that in the case of most problems in electrophysiology, the region of stability of RL methods of the form (23) covers the entire negative half plane [18], i.e. A-stability. The stability of the scheme then depends on the methods used to solve the differential equations for the

transmembrane potential and the concentrations. The following discussion on the stability of the Rush-Larsen methods uses heuristic arguments that provide critical time-steps relatively close to those in numerical tests, but this derivation cannot yet be formalized.

We use a different scheme for the differential equations of the gating variables and the concentrations. Therefore, we look at the problem in the form (6)-(8). As done previously, we linearize around some constant state $(\tilde{u}, \tilde{\mathbf{v}}, \tilde{\mathbf{X}}) = \tilde{Y}$ and take the Fourier transform. We apply the scheme (14) to the linearized problem. The scheme is identical to the Forward-Backward Euler method for the transmembrane potential and concentrations. We get

$$\left(1 - \sigma \frac{2\cos\omega h - 2}{h^2}\right) U_\omega^{n+1} = U_\omega^n + \Delta t \begin{bmatrix} -\dfrac{\partial I}{\partial u}(\tilde{Y}) & -\dfrac{\partial I}{\partial \mathbf{v}}(\tilde{Y}) & -\dfrac{\partial I}{\partial \mathbf{X}}(\tilde{Y}) \end{bmatrix} \begin{bmatrix} U_\omega^n \\ V_\omega^n \\ X_\omega^n \end{bmatrix}, \quad (55)$$

$$X_\omega^{n+1} = X_\omega^n + \Delta t \begin{bmatrix} \dfrac{\partial \mathbf{g}}{\partial u}(\tilde{Y}) & \dfrac{\partial \mathbf{g}}{\partial \mathbf{v}}(\tilde{Y}) & \dfrac{\partial \mathbf{g}}{\partial \mathbf{X}}(\tilde{Y}) \end{bmatrix} \begin{bmatrix} U_\omega^n \\ V_\omega^n \\ X_\omega^n \end{bmatrix}. \quad (56)$$

We solve the ODEs for the gating variables with the RL method, which in this case is an exponential integrator method. As opposed to previously studied methods, it is not a linear multistep method. Because the Rush-Larsen method is considered A-stable when applied to the gating equations [18], we assume for the sake of analysis that we have an A-stable one-step linear method for V, which can be written as

$$V_\omega^{n+1} = R(\Delta t, \omega) V_\omega^n, \quad (57)$$

where $\rho(R(\Delta t, \omega)) \leq 1$, for any $\Delta t > 0$. We then consider the most restrictive case where $\rho(R(\Delta t, \omega)) = 1$. This occurs when $R(\Delta t, \omega)) = I$, which is equivalent to setting

$$V_\omega^{n+1} = V_\omega^n. \quad (58)$$

As for the previous semi-implicit methods, we take $\omega = 0$ because it is the choice that is most restrictive for stability. Combining (55)-(58), we have

$$Y^{n+1} = (I + \Delta t J_{RL}) Y^n, \quad (59)$$

where

$$J_{RL} = \begin{bmatrix} -\dfrac{\partial I}{\partial u}(\tilde{Y}) & -\dfrac{\partial I}{\partial \mathbf{v}}(\tilde{Y}) & -\dfrac{\partial I}{\partial \mathbf{X}}(\tilde{Y}) \\ 0 & 0 & 0 \\ \dfrac{\partial \mathbf{g}}{\partial u}(\tilde{Y}) & \dfrac{\partial \mathbf{g}}{\partial \mathbf{v}}(\tilde{Y}) & \dfrac{\partial \mathbf{g}}{\partial \mathbf{X}}(\tilde{Y}) \end{bmatrix}. \quad (60)$$

As we did before for the Euler methods, the stability condition for this scheme is $\Delta t \leq -2/\lambda_{\min}$, where $\lambda_{\min}$ is the most negative eigenvalue of J_{RL}. The critical time-step, $\Delta t^*_{\text{theo}} = -2/\lambda_{\min}$ is shown in Table 2 for the BR model and in Table 7 for the TNNP model.

Similarly, for the RL-CNAB method, we get from the stability analysis of the Adams Bashforth method, $\Delta t^*_{\text{theo}} = -1/\lambda_{\min}$.

3.1.6 Deferred Correction

Due to the more complex nature of the third-order deferred correction scheme, we cannot easily find a stability condition using absolute stability analysis. However, the numerically observed critical time-step is very close to the one from the Forward-Backward Euler method.

3.2 Numerical Results

In this section, we compare the critical time-steps obtained through our absolute stability analysis from the previous section to those observed numerically in the case of the BR, MS and TNNP models. All parameters used and the details on the ionic models are given in [20].

For the 1D case, we use a spatial domain of length 100 cm, discretized by equally spaced nodes $x_i = ih$, $h = 100/N$ or its equivalent for the nondimensionalized MS model. We used a final time T of 400 ms for the BR model, 350 ms for the MS model and 300 ms for the TNNP model. The applied stimulation current is the C^∞ function given by

$$I_{app}(\mathbf{x},t) = \begin{cases} 50\exp\left(1-\dfrac{1}{1-(t-1.5)^2}\right)\exp\left(1-\dfrac{1}{1-4(x-0.5)^2}\right), & \text{if } 0<x<1, \\ & 0.5<t<2.5, \\ 0, & \text{otherwise.} \end{cases} \tag{61}$$

All computations for the 1D case were made with MATLAB.

We also tested the stability of the BR model in the 2D case using the code CHORAL [1]. For the 2D simulations, we used a 1cm × 1cm square domain discretized with 3432 grid points and a final time T of 16 ms. The applied stimulation current in the 2D case is a C^1 function of x and t given by

$$I_{app}(\mathbf{x},t) = \begin{cases} 50\dfrac{1+\cos(\pi r)}{2}\dfrac{1+\cos(\pi\tau)}{2}, & \text{if } 0\le r\le 1,\ 0\le\tau\le 1, \\ 0, & \text{otherwise,} \end{cases} \tag{62}$$

where $r = |\mathbf{x}-\mathbf{x_0}|/r_0$ with $\mathbf{x_0}$ the point at the center of a circular simulation zone and r_0, the radius of this zone. Similarly, $\tau = (t-t_0)/\tau_0$ sets a stimulation interval in time starting at $t = t_0$ with duration τ_0. We set $t_0 = 5$ms, $\tau_0 = 1.5$ms and $r_0 = 0.125$cm.

For each method, the theoretical critical time-step Δt^*_{theo} is determined by the stability conditions from the previous section and the most negative eigenvalue of the Jacobian of the ionic model used, $\lambda_{\min}$, which is calculated on the domain as explained in the previous section. The numerically observed critical time-step Δt^* is the largest possible time-step for which the numerical solution remains bounded. When using a relatively large time-step with the more stable Rush-Larsen methods, the solution remains bounded but its shape degenerates and does not represent the real shape of the wave. Therefore, for the RL methods the value of Δt^* is taken as the largest possible time-step for which the potential wave does not degenerate.

Remark 2 (Computing the Jacobian for discontinuous models). The functions I, F and G of the BR model are all continuously differentiable with respect to each variable. We thus

found the Jacobian analytically. Other ionic models have discontinuities which prevent us from defining a Jacobian for discontinuity points. To avoid deriving expressions on both sides of the discontinuities, we decided to approximate the Jacobian numerically, for e.g. by using MATLAB's `numjac` function. Several functions in the ODEs for the gating variables in the TNNP model have discontinuities at $u = -40$mV and the function in the ODE of the gating variable of the MS model has a discontinuity at $u = u_{gate}$. These are null sets of the phase space and in general we will not observe these discontinuities when approximating the Jacobian or solving the differential equations numerically. However, if we approximate the Jacobian at the discontinuity point or close enough to be below the tolerance for `numjac`, we obtain extremely large eigenvalues. These discontinuities might become a problem for extremely small time-steps because there could be values of u very close to the singularities. This has not been a problem for our simulations.

3.2.1 Beeler-Reuter

The most negative eigenvalue of the Jacobian for the Beeler-Reuter model is $\lambda_{\min} = -81.782$. Note that one can find the most negative eigenvalues of the Jacobian of 37 different ionic models in [23, 16], where the authors obtained a value of -82.0 for the BR model. This small difference is most likely due to a distinct applied stimulation current or initial conditions. The theoretical critical time-steps for methods studied above are shown in Table 2 and the numerically observed critical time-steps in Table 1 for the 1D case and in Table 3 for the 2D case. As expected, the critical time-steps for the semi-implicit methods are independent of h.

We see that for all methods except the RL methods, the critical time-steps obtained numerically are very close to those obtained through the absolute stability analysis. In the case of the RL methods, the critical time-steps are similar for RL-FBE, but Δt^* is approximately half of Δt^*_{theo} for RL-CNAB. This is likely a consequence of the difficulty to identify the critical time step in numerical tests.

For the Strang splitting methods, we observe smaller Δt^* in 2D compared to their 1D equivalent. Otherwise, the analysis done in the last section for the 1D monodomain model seems to apply for the semi-implicit methods in the 2D case. Due to the more complex nature of the Finite Element method for higher dimensions, the analysis of explicit methods such as FE depends on the nature of the mesh used (uniform vs non-uniform).

We observe that the critical time-steps of the FE and FBE methods are initially the same for $h = 0.0625$, but as h gets smaller FE becomes less stable with a critical time-step of order h^2. This indicates that the use of explicit methods to solve the BR model is only justifiable for very coarse meshes.

The RL methods are the most stable of all the methods studied. The RL-CNAB has a Δt^* more than three times larger than the one of the next most stable method, CN-RK4. This method has a Δt^* slightly larger than the one for CN-RK2, which reflects the fact that the stability region of CN-RK2 is included in the stability region of CN-RK4. The Δt^* for the SBDF2 method is three to four times smaller than the ones for the Strang splitting methods and 50% smaller than the one for FBE. These relations between the Δt^* of the linear multistep methods, i.e. excluding the RL methods, are the same for all

Table 1: Size of Δt^* for the numerical methods used with BR model

Methods	h=0.062 500	h=0.031 250	h=0.015 625
2nd Order SBDF	0.016 848	0.016 848	0.016 848
Strang Splitting (CN-RK4)	0.071 480	0.071 480	0.071 480
Strang Splitting (CN-RK2)	0.050 289	0.050 289	0.050 289
Forward Euler	0.025 373	0.020 094	0.005 063 9
Forward-Backward Euler	0.025 373	0.025 373	0.025 373
RL-CNAB	0.235 29	0.235 29	0.235 29
RL-FBE	>0.800 00	>0.800 00	>0.800 00
DC3	0.024 465	0.024 465	0.024 465

Table 2: Size of Δt^*_{theo} for the numerical methods used with BR model

Methods	h=0.062 500	h=0.031 250	h=0.015 625
2nd Order SBDF	0.016 304	0.016 304	0.016 304
Strang Splitting (CN-RK4)	0.068 115	0.068 115	0.068 115
Strang Splitting (CN-RK2)	0.048 911	0.048 911	0.048 911
Forward Euler	0.024 455	0.020 238	0.005 066 2
Forward-Backward Euler	0.024 455	0.024 455	0.024 455
RL-CNAB	0.423 42	0.423 42	0.423 42
RL-FBE	0.846 83	0.846 83	0.846 83

Table 3: Size of Δt^* for the numerical methods used with BR model in 2D

Methods	Δt^*
2nd Order SBDF	0.016 131
Strang Splitting (CN-RK4)	0.059 566
Strang Splitting (CN-RK2)	0.044 933
Forward-Backward Euler	0.024 242
RL-CNAB	0.200 00

the ionic models used.

As mentioned in the last section, the critical time-step for the DC3 method is very close to the value for FBE: it is slightly smaller.

3.2.2 Mitchell-Schaeffer

For the MS model, the most negative eigenvalue obtained is $\lambda_{\min} = -2.6651$. The theoretical critical time-steps for the methods studied are shown in Table 5 and the numerically observed critical time-steps in Table 4. We see that for all methods, Δt^* is very close to Δt^*_{theo}. As expected, the critical time-steps for the semi-implicit implicit methods are independent of the space step h. Because the MS model is not very stiff, the stability of the FE method depends on the size of h even for relatively large space step h. This indicates the necessity for taking the diffusion implicitly when solving less stiff models

Table 4: Size of Δt^* for the numerical methods used with MS model

Methods	h=1	h=0.500 00	h=0.250 00
2nd Order SBDF	0.526 32	0.526 32	0.526 32
Strang Splitting (CN-RK4)	1.8519	1.8519	1.8519
Strang Splitting (CN-RK2)	1.5217	1.5217	1.5217
Forward Euler	0.122 70	0.035 874	0.008 940 5
Forward-Backward Euler	0.769 23	0.769 23	0.769 23
DC3	0.769 23	0.769 23	0.769 23

Table 5: Size of Δt^*_{theo} for the numerical methods used with MS model

Methods	h=1	h=0.500 00	h=0.250 00
2nd Order SBDF	0.500 29	0.500 29	0.500 29
Strang Splitting (CN-RK4)	2.0902	2.0902	2.0902
Strang Splitting (CN-RK2)	1.5009	1.5009	1.5009
Forward Euler	0.111 79	0.033 672	0.008 872 8
Forward-Backward Euler	0.750 43	0.750 43	0.750 43

such as the MS model.

We also observe that the CN-RK4 method requires a slightly smaller time-step than Δt^*_{theo}. This is most likely resulting from the oscillations caused by the use of the Crank-Nicolson method [29]. These oscillations can be observed for large time-steps for both Strang splitting methods.

For the MS model, the Strang splitting methods are the most stable of all the methods studied. They have a Δt^* three to four times larger than for the SBDF2 method.

The DC3 and FBE methods have the same numerically observed critical time-step.

3.2.3 ten Tuscher-Noble-Noble-Panfilov

For the TNNP model, the most negative eigenvalue obtained is $\lambda_{\min} = -1191.7$. The value given in [23, 16] is -1170. Again, this difference is most likely due to a distinct applied stimulation current or initial conditions. The theoretical critical time-steps for the methods studied are shown in Table 7 and the numerically observed critical time-steps in Table 6.

As for the BR model, we see that for all methods except the RL methods, the critical time-steps obtained numerically are very close to those obtained through absolute stability analysis. As expected, the critical time-steps for the semi-implicit methods are independent of the space step h.

In the case of the RL methods, Δt^* is approximately half of Δt^*_{theo} for RL-CNAB and two thirds for RL-FBE.

For the Forward Euler scheme, Δt^* is the same for $h = 0.0625, 0.03125, 0.015625$. For smaller values of h, we begin to see a dependence of Δt^* on h^2, as shown in Table 8. These results indicate that for very stiff models such as the TNNP model, the use of fully explicit methods could still be acceptable, except on finer spatial meshes.

Table 6: Size of Δt^* for the numerical methods used with TNNP model

Methods	h=0.062 500	h=0.031 250	h=0.015 625
2nd Order SBDF	0.001 134 8	0.001 134 8	0.001 134 8
Strang Splitting (CN-RK4)	0.004 836 2	0.004 836 2	0.004 836 2
Strang Splitting (CN-RK2)	0.003 456 2	0.003 456 2	0.003 456 2
Forward Euler	0.001 704 4	0.001 704 4	0.001 704 4
Forward-Backward Euler	0.001 704 4	0.001 704 4	0.001 704 4
RL-CNAB	0.091 380	0.091 380	0.091 380
RL-FBE	>0.600 00	>0.600 00	>0.600 00
DC3	0.001 686 9	0.001 686 9	0.001 686 9

Table 7: Size of Δt^*_{theo} for the numerical methods used with TNNP model

Methods	h=0.062 500	h=0.031 250	h=0.015 625
2nd Order SBDF	0.001 118 9	0.001 118 9	0.001 118 9
Strang Splitting (CN-RK4)	0.004 674 6	0.004 674 6	0.004 674 6
Strang Splitting (CN-RK2)	0.003 356 6	0.003 356 6	0.003 356 6
Forward Euler	0.001 678 3	0.001 678 3	0.001 678 3
Forward-Backward Euler	0.001 678 3	0.001 678 3	0.001 678 3
RL-CNAB	0.206 12	0.206 12	0.206 12
RL-FBE	0.412 23	0.412 23	0.412 23

Table 8: Size of Δt^* for the Forward Euler's method with TNNP model

h	Δt^*	Δt^*_{theo}
0.015625	0.001 704 4	0.001 678 3
0.0078125	0.001 267 7	0.001 266 9
0.00390625	0.000 316 82	0.000 316 76

The RL methods are the most stable of all the methods studied. The RL-CNAB has a Δt^* almost twenty times larger than the next most stable method, CN-RK4.

As for the BR model, the DC3 method has a Δt^* slightly smaller than the one for FBE.

As expected from the analysis of the last sections, for all ionic models, only the fully explicit Forward Euler's method has a critical time-step depending on the size of h. Indeed, as h gets smaller we see that Δt^* is eventually proportional to h^2. Looking at the Jacobian for large h, we see that the minimum eigenvalue comes from the gating variables, in the case of the models studied. For the FE method, small values of h are required for the eigenvalue coming from the diffusion term in the monodomain equation to become the most negative. This leads to the conclusion that for very stiff models such as the TNNP model, taking the diffusion implicitly is only necessary for very fine meshes.

For all models, the semi-implicit method which requires the smallest time-step is the SBDF2 method.

4 Accuracy of the numerical methods

In this section, we investigate the accuracy of the different time-stepping methods when solving the monodomain model coupled with the three different ionic models studied. We start by conducting a convergence test for each method to verify if they have the correct rate of convergence. The accuracy of the methods will also be compared relatively to the size of the time-step used. Afterwards, we will compare the accuracy of the methods relatively to the computational time needed to run the simulations.

4.1 Convergence tests

We now study the convergence of the different time-stepping methods for a given spatial mesh. We split the error in space and time given that their order may not be the same and study the error in time only. We have

$$\|u - u_{h,\Delta t}\| \leq \|u - u_h\| + \|u_h - u_{h,\Delta t}\| = \mathcal{O}(h^p + \Delta t^q), \tag{63}$$

where u is the exact solution for the transmembrane potential, u_h is the solution of the semi-discretized problem in space and $u_{h,\Delta t}$ is the solution of the fully discretized problem. Since we want to study the convergence of the time-stepping methods, we will only consider the error between the solutions of the semi-discretized and fully discretized problems. A consequence of considering only the error in time is that relative error levels requested in our test cases will be smaller (sometimes much smaller) than relative error levels in space and time usually found in the cardiac electrophysiology literature.

We test convergence with respect to the following errors:

$$\begin{aligned} e_{L^2} &= \|u_{h,\Delta t}(T) - u_h(T)\|_{L^2}, \qquad & e_{H^1} &= |u_{h,\Delta t}(T) - u_h(T)|_{H^1}, \\ e_c &= |c_{h,\Delta t} - c_h|, & e_{T_1} &= |T_{1,h,\Delta t} - T_{1,h}|, \end{aligned} \tag{64}$$

where u_h is a reference semi-discretized solution for the transmembrane potential, which is calculated using a very small time-step. The norms used are discrete approximations of the $L^2(\Omega)$ norm and $H^1(\Omega)$ seminorm using Simpson's rule. The solutions $u_{h,\Delta t}$ are calculated with the same spatial mesh and at the same final time T as the reference solution, but using larger values for Δt. We denote by $c_{h,\Delta t}$ the wave velocity, and by $T_{1,h,\Delta t}$ the depolarization time, i.e. the time at which a given point of the domain reaches a given super-threshold value of the transmembrane potential. The wave velocity and depolarization time of the reference solution u_h are denoted by c_h and $T_{1,h}$, respectively. The wave velocity is defined by $c = (x_2 - x_1)/(T_2 - T_1)$, where T_i is the time when the depolarization front of the potential wave passes through chosen nodes x_i, $i = 1, 2$. The wave front passes through a point x_i during the time-step from t_n to $t_{n+1} = t_n + \Delta t$ if $u(x_i, t_n) < \hat{u}$, but $u(x_i, t_{n+1}) \geq \hat{u}$ for some chosen value $\hat{u}$ on the wave front. For better approximations of T_i, we use linear interpolation and define

$$T_i = t_n + \Delta t \frac{\hat{u} - u(x_i, t_n)}{u(x_i, t_{n+1}) - u(x_i, t_n)}. \tag{65}$$

Assuming that the error is proportional to Δt^α, the estimated convergence rate is calculated with

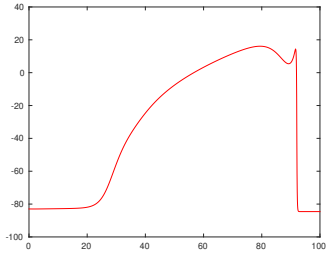
$$\alpha = \frac{\log(|e_1/e_2|)}{\log(\Delta t_1/\Delta t_2)}, \tag{66}$$

where Δt_1 and Δt_2 are consecutive time-steps in a sequence of decreasing time-steps, and e_1 and e_2 are the corresponding errors. All solutions are calculated using the same stimulation current and parameters from Section 3.2. The largest Δt used for each method is taken close to the critical time-step found in the previous section, except for the RL methods where we use a starting time-step similar to the other methods, that is when the RL methods start to show their asymptotic behaviour.

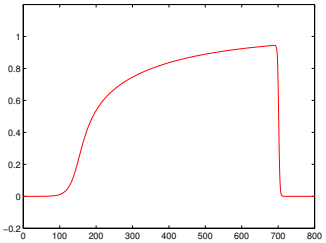
The order of convergence in the L^2 norm and H^1 seminorm are expected to be the same because the numerical solutions $u_{h,\Delta t}$ are in the same finite-dimensional space and thus the equivalence of norms applies.

4.1.1 Beeler-Reuter with 1D Monodomain

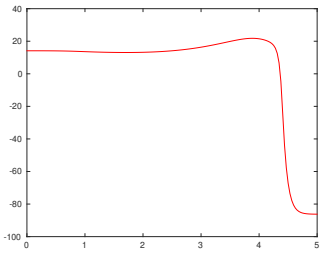
The reference solution u_h for the BR model is obtained using SBDF2 on a domain of length 100 cm discretized in space with 1600 nodes and computed at time $T = 400\,\text{ms}$ with $\Delta t = 1/125\,000\,\text{ms}$. The graph of the reference solution is shown in Figure 2a.



(a) BR model: u at time $T = 400\,\text{ms}$, plotted for $x \in [0, 100]$.



(b) MS model: u at time $T = 350\,\text{ms}$, plotted for $x \in [0, 800]$.



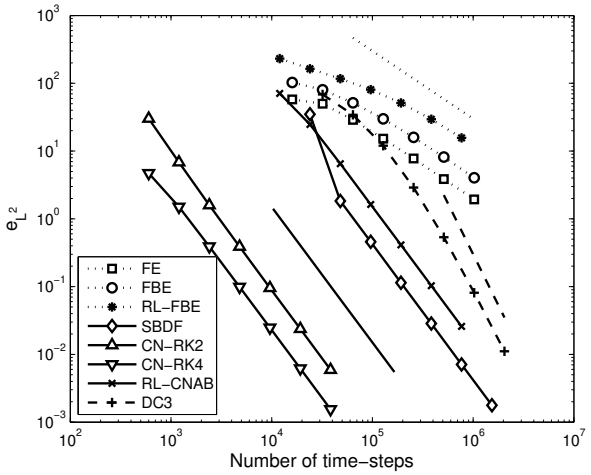
(c) TNNP model: u at time $T = 12\,\text{ms}$, plotted for $x \in [0, 5]$.

Figure 2: Reference solutions for the transmembrane potential

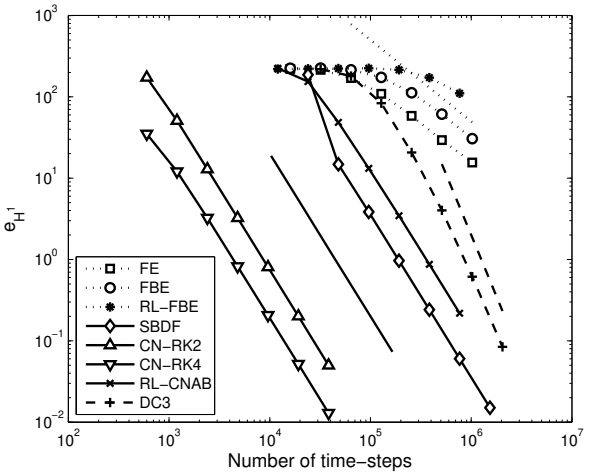
To determine c and T_1, we take $\hat{u} = -30\,\text{mV}$, $x_1 = 20\,\text{cm}$ and $x_2 = 50\,\text{cm}$. The results of the convergence tests for the 1D BR model are shown in Figure 3. We illustrate how the errors defined in (64) vary as the size of the time-step is decreased. We observe that T_1 and c reach their asymptotic rate of convergence faster than the L^2 norm and H^1 seminorm of the error.

For the 1D BR model, all of the methods studied showed their expected asymptotic order of convergence. For the first-order methods, we observe that for the same Δt, FE is almost twice as accurate as its semi-implicit version, FBE, and both methods are significantly more accurate than RL-FBE. It takes very small values of Δt for RL-FBE to reach its asymptotic order of convergence.

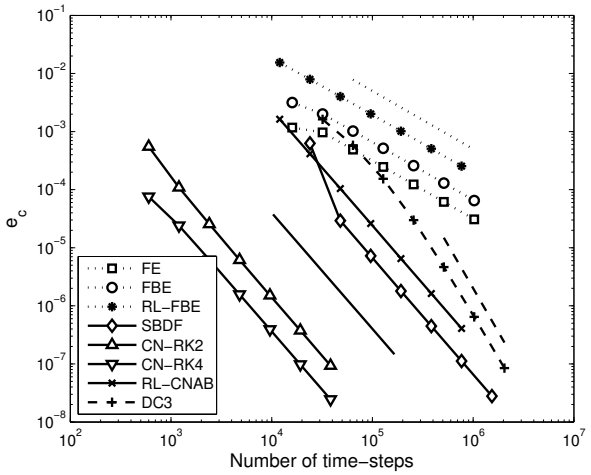
For the second-order methods, we observe that for the same Δt, CN-RK4 is about ten times more accurate than CN-RK2, which in turn is about two times more accurate



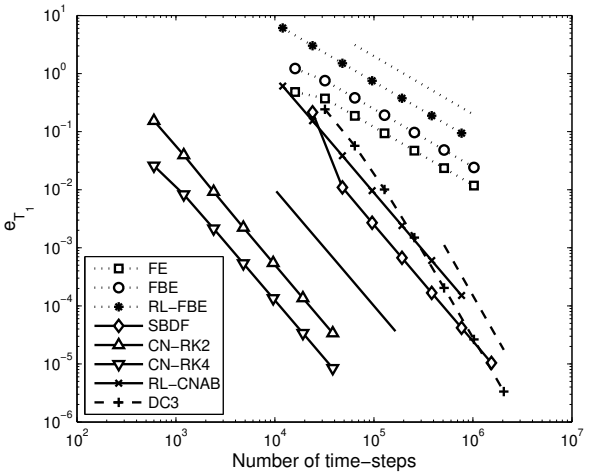
(a) Error in $L^2(\Omega)$ norm



(b) Error in $H^1(\Omega)$ seminorm



(c) Error of the wave velocity c



(d) Error of the depolarization time T_1

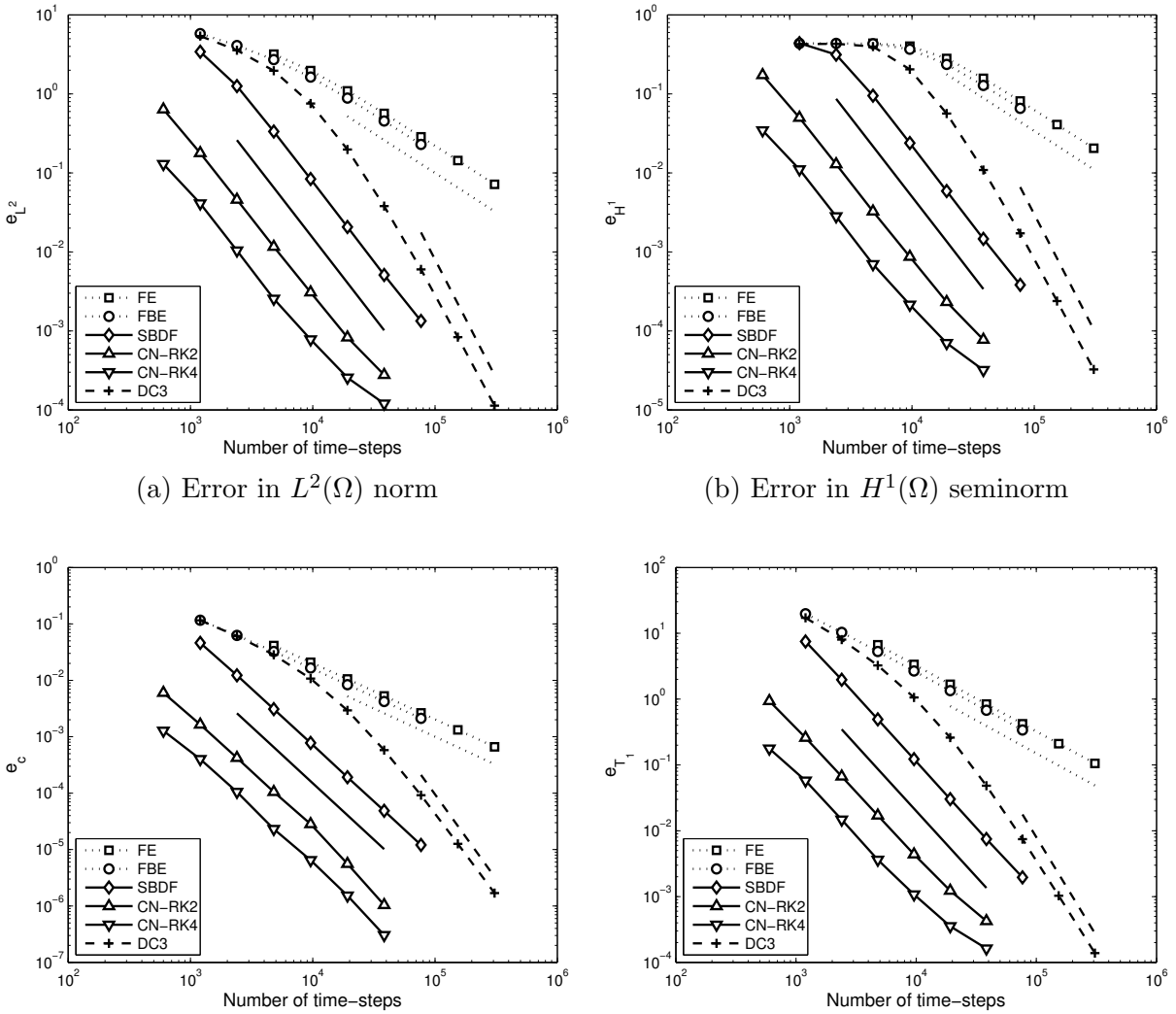
Figure 3: Errors for the BR model at fixed $h = 1/16$. The order of convergence can be observed by comparing with lines of negative slope one (dotted), two (solid), and three (dashed).

than SBDF2. This last method is two times more accurate than RL-CNAB. For the DC3 method, it takes very small values of Δt before it exhibits its correct order of convergence. Even for $\Delta t = 1/5120$, the error is still larger than what we would obtain by using a second-order method. We noticed that this unexpected reduced accuracy of DC3 sometimes results when the stimulation current is applied close to the boundary of the domain.

4.1.2 Mitchell-Schaeffer with 1D Monodomain

The reference solution for the MS model is computed using SBDF2 on a domain of length 800 discretized in space with 800 nodes and computed at time 350 with $\Delta t = 7/60000$. The plot of the reference solution is shown in Figure 2b.

To determine c and T_1, we take $\hat{u} = 0.5$, $x_1 = 50$ and $x_2 = 80$. For the convergence of the DC3 method, we used the same method instead of SBDF2 to calculate a reference solution with $\Delta t = 7/60000$. The results of the convergence tests for the 1D MS model are shown in Figure 4.



(a) Error in $L^2(\Omega)$ norm

(b) Error in $H^1(\Omega)$ seminorm

(c) Error of the wave velocity c

(d) Error of the depolarization time T_1

Figure 4: Errors for the MS model at fixed $h = 1$. The order of convergence can be observed by comparing with lines of negative slope one (dotted), two (solid), and three (dashed).

For the MS model, all methods studied showed their expected asymptotic order of convergence, but the Strang splitting methods lose their order when Δt gets smaller. This could be caused by the discontinuities in the ionic model. For the first-order methods, we observe that for the same Δt, FBE is slightly more accurate than its fully explicit version, FE.

For second-order methods, we observe that for the same Δt, CN-RK4 is almost ten times more accurate than CN-RK2, which in turn is about twenty times more accurate

than SBDF2. For the DC3 method, it takes very small values of Δt before the method exhibits its correct order of convergence. Even for $\Delta t = 7/6144$, its error is still larger than what we would obtain by using a good second-order method.

4.1.3 ten Tuscher-Noble-Noble-Panfilov with 1D Monodomain

Due to the complexity and stability requirements of the TNNP model, we study the convergence of the methods on a smaller spatial domain. The domain will be too small for the whole wave to develop, but still includes the depolarization wavefront. The reference solution is computed using SBDF2 on a domain of length 5 cm discretized in space with 160 nodes and computed at time 12 ms with $\Delta t = 6e-7$. The reference solution is shown in Figure 2c.

To determine c and T_1, we take $\hat{u} = -30\,\text{mV}$, $x_1 = 1\,\text{cm}$ and $x_2 = 2.5\,\text{cm}$. For the convergence of the DC3 method, we used the same method instead of SBDF2 to calculate a reference solution with $\Delta t = 7.5e-7$. The results of the convergence tests for the 1D TNNP model are shown in Figure 5.

For the TNNP model, all methods studied showed their expected asymptotic order of convergence, but the convergence of the RL-CNAB and DC3 methods is erratic when Δt gets very small. The solution obtained with the RL-CNAB method first converges to the reference solution obtained with SBDF2 then the convergence stagnates when Δt gets very small. The L^2 error between the solutions seems to stabilize around $e_{L^2} = 0.00115$, which is less than 0.0012% of the reference solution's L^2 norm. The DC3 method shows similar trends with third-order convergence that deteriorates when Δt gets very small. This is likely due to the difficulty of computing a reference solution at such a small level of errors. Another reason for this strange behaviour with very small time-steps might be caused by the discontinuities in the right-hand-side of the ODEs for the TNNP model. For a small time-step, it is more likely that the numerical solution falls close to a discontinuity at some of the grid points. Before its order of convergence deteriorates, the DC3 method is actually more accurate than the second-order methods.

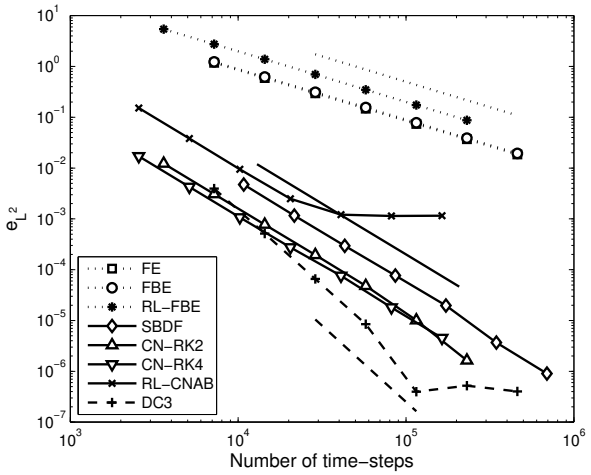
It is important to note that the TNNP model is very stiff and that is why we use very small time-steps for the convergence tests. In practice, it is not relevant to have errors as small as those obtained for the smallest time-steps used since the modelling error is then much larger then the numerical error.

For the first-order methods, we observe that for the same Δt, FE is slightly more accurate than its semi-implicit version, FBE, and both methods are more than twice as accurate as RL-FBE.

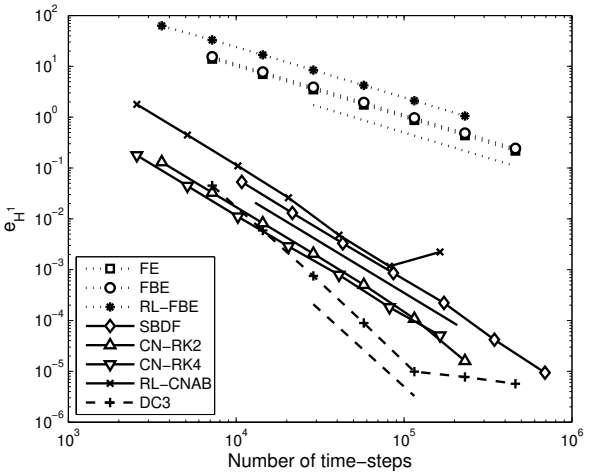
For second-order methods, the Δt used are not the same, but one can easily see that for the same Δt, CN-RK4 is almost two times more accurate than CN-RK2, which in turn is about two times more accurate than SBDF2. This last method is about two times more accurate than RL-CNAB.

4.1.4 Beeler-Reuter with 2D Monodomain

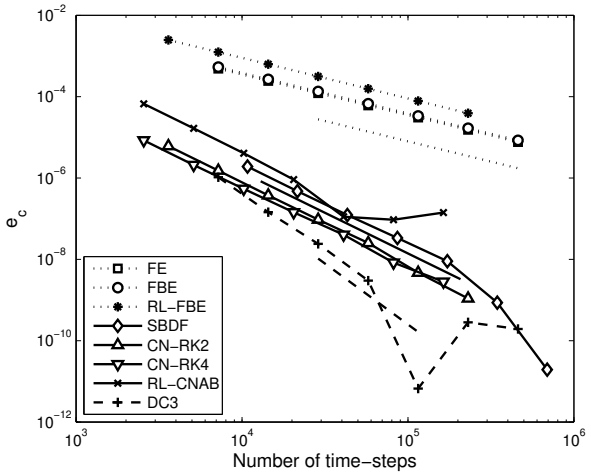
In the 2D case, the reference solution is computed using SBDF2 on a 1cm × 1cm square domain discretized with an unstructured mesh of 3432 points (roughly of size 59 × 59)



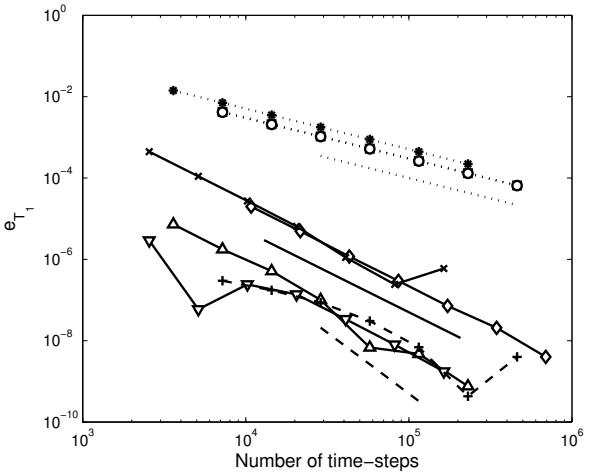
(a) Error in $L^2(\Omega)$ norm



(b) Error in $H^1(\Omega)$ seminorm



(c) Error of the wave velocity c



(d) Error of the depolarization time T_1

Figure 5: Errors for the TNNP model at fixed $h = 1/32$. The order of convergence can be observed by comparing with lines of negative slope one (dotted), two (solid), and three (dashed).

and computed at time $T = 16\,\text{ms}$ with $\Delta t = 5.98e - 6$. The reference solution is shown in Figure 6.

The results of the convergence tests for the 2D BR model are shown in Figure 7. We present errors and convergence rates in L^2 norm and H^1 seminorm only; wave velocity and depolarization time were not considered. We also only considered a subset of the methods studied in the 1D case.

As for the 1D BR model, all of the methods studied showed their expected asymptotic order of convergence for the 2D case. For the same Δt, CN-RK4 is about five times more accurate than CN-RK2, which in turn is almost four times more accurate than SBDF2. This last method and RL-CNAB have approximately the same accuracy. The first-order FBE method is significantly less accurate than the higher-order methods.

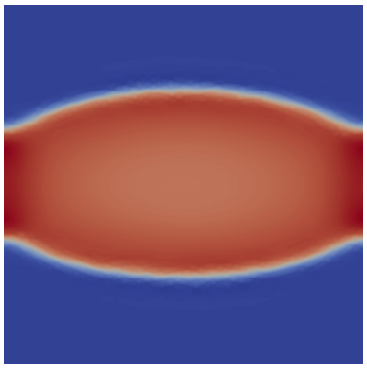
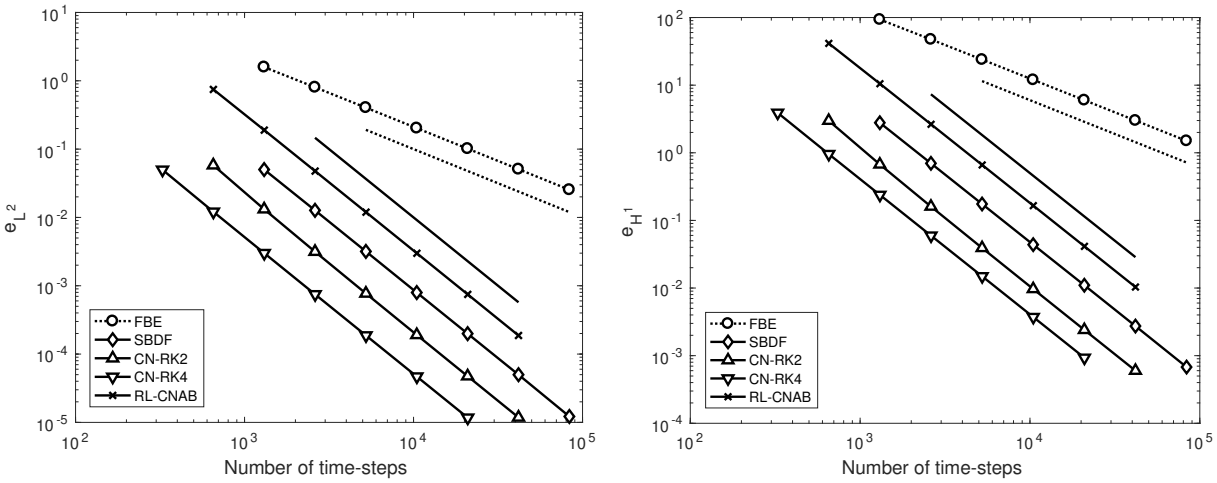


Figure 6: Reference solution for the transmembrane potential at time $T = 16\,\text{ms}$ for the BR model in 2D plotted as a function of $x \in [0,1] \times [0,1]$.



(a) Error in $L^2(\Omega)$ norm

(b) Error in $H^1(\Omega)$ seminorm

Figure 7: Errors for the BR model in 2D. The order of convergence can be observed by comparing with lines of negative slope one (dotted), and two (solid).

4.2 CPU Performance of the Numerical Methods

In the last sections, we compared the accuracy of the methods for given time-steps. However, this does not take into account the difference of computational cost for an iteration depending on the method used.

We now compare the accuracy of the different numerical methods studied with respect to the computational time of the simulation. For all 1D test cases, we use the same Matlab scripts, spatial domain and spatial discretization as for the stability and convergence tests on a Lenovo ThinkCentre M900 with 4 Intel Core i5-6500T 2.10GHz processors and 7.7 GB of RAM. The 2D/3D cases were done with the compiled Fortran 2003 code CHORAL [1] on a HP ZBook 17 with 4 Intel i7 2.70GHz processors and 16 GB of RAM. For the 2D simulations, we used the same 1cm $\times$ 1cm square domain as above discretized with 3241

grid points. For the 3D simulations, we used a 1cm $\times$ 1cm $\times$ 1cm cubic domain discretized with 109620 grid points. In 2D/3D cases, the final time T was set to 20 ms for the BR model and 15 ms for the TNNP model since the wave travels faster in this latter case. The applied stimulation current is given by (62) with $t_0 = 4$ms, $\tau_0 = 1.5$ms and $r_0 = 0.125$cm.

For the sake of comparing the relative errors for different norms, we give the L^2 norm and H^1 seminorm for the different reference solutions. These are, respectively, 519.6 and 155.1 for the 1D BR model, 19.00 and 0.3154 for the 1D MS model, 96.26 and 201.8 for the 1D TNNP model, 134.3 and 728.9 for the 2D BR model, 141.5 and 876.4 for the 2D TNNP model, 223.2 and 775.2 for the 3D BR model.

We choose a target value of the L^2 error on the numerical solution, e_{L^2}, and for each method, we find the time-step needed to achieve this level of accuracy and evaluate the computational time of the simulation. For all test cases except the BR and TNNP models in 1D, we present results for two target values of the L^2 error, corresponding to about 0.1% and 1% relative L^2 error. In 1D, the relative errors were set to 0.5% and 0.005% for BR and TNNP models, respectively. The target relative error was set to a lower value for the 1D TNNP model to present at least one test case where most time-stepping methods are within their stability region. These percentages of relative of relative errors (0.1% to 1%) are representative of the level of numerical error usually expected from computations in cardiac electrophysiology, given that ionic models hardly achieve this level of accuracy. These percentages also ensure that the time steps used are as close as possible to values used in common cardiac electrophysioloy simulations. When a particular method is not stable for a given percentage of relative error, results are presented for the largest time step that gives a stable solution.

Results are shown for the different methods in Tables 9 to 18. Each table contains the L^2 error, e_{L^2}, close enough to the chosen target error, with its corresponding H^1 error, e_{H^1}, the time-step Δt required, the total CPU time of the simulation and the average CPU time per iteration. We indicate in the title of each table the relative size of the error e_{L^2} with respect to the L^2 norm of the reference solution. Note that the relative H^1 seminorm errors usually differ from the target percentage of relative L^2 error, and are often greater than their respective relative L^2 errors.

Table 9: CPU time of the numerical methods for the BR model in 1D for 0.5% relative L^2 error

Method	e_{L^2}	e_{H^1}	Δt	CPU time (s)	CPU/time-step (ms)
SBDF2	2.590	18.67	0.010 00	37.21	0.9303
RL-CNAB	2.604	18.74	0.005 333	86.10	1.148
CN-RK4	2.594	18.53	0.045 45	57.11	6.489
CN-RK2	2.591	18.63	0.016 00	77.01	3.080
FBE	25.97	153.0	0.002 694	117.8	0.7932
DC3	2.606	18.76	0.001 493	780.2	2.911

For the 1D BR model, we see that the most efficient method is the SBDF2 method. It is almost twice as fast as the next most efficient method, CN-RK4, which is itself faster than the CN-RK2 method. For each time-step of the SBDF2 method, we only have to

compute the functions of the ionic model once, as opposed to the Strang-splitting methods where we compute them four or eight times for CN-RK2 and CN-RK4, respectively. As for the DC3 method, the functions are computed only three times per time-step, but it takes small values of Δt for the method to reach a higher rate of convergence. We observe that the third-order method, DC3, is not efficient compared to the second-order methods. Indeed, it takes more than twenty times more CPU time to compute the solution than the SBDF2 method for the chosen level of error. By choosing very small errors, we expect the DC3 method to eventually surpass the second-order methods. At the chosen level of error e_{L^2}, the DC3 method has not yet entered its asymptotic zone.

We only tested the computational time of the FBE method at 5% relative L^2 error because it would take extremely small time-steps to reach the level of error that we chose for the other methods. It is clear that for 0.5% relative error, the FBE method is not efficient compared to high-order methods.

The computational time per time-step of the SBDF2 method is only slightly larger than for the FBE method and thus there is no advantage in terms of efficiency to use a first-order method. One time-step of CN-RK4 is two times more costly than for CN-RK2, which is almost four times more costly than the SBDF2 method. This is directly related to the number of computations of the ionic functions. One time-step of DC3 is more than three times the cost of an SBDF2 time-step, which also relates to the computation of the ionic functions, but there is additional cost due to the more complex nature of the DC3 method. These relations extend to the different models studied. Even though RL-CNAB also only has one computation of the ionic functions per time-step, the use of the exponential function makes an iteration for this method slightly more expensive than for the SBDF2 method. This impact is lessened in the case of the TNNP model because the computation of the ionic functions takes up most of the computational cost, due to the complexity of the TNNP model.

Table 10: CPU time of the numerical methods for the MS model in 1D for 0.1% L^2 error

Method	e_{L^2}	e_{H^1}	Δt	CPU time (s)	CPU/time-step (ms)
SBDF2	0.01898	0.005431	0.01728	2.670	0.1319
CN-RK4	0.01884	0.005091	0.1971	1.597	0.8990
CN-RK2	0.01894	0.005329	0.09272	1.736	0.4599
FBE	0.1897	0.05432	0.003763	12.09	0.1300
DC3	0.01903	0.005461	0.006972	23.25	0.4671

Table 11: CPU time of the numerical methods for the MS model in 1D for 1% L^2 error

Method	e_{L^2}	e_{H^1}	Δt	CPU time (s)	CPU/time-step (ms)
SBDF2	0.18997	0.05426	0.05488	0.9671	0.1516
CN-RK4	0.18970	0.05093	0.7447	0.4793	1.019
CN-RK2	0.18855	0.05276	0.3017	0.6061	0.5225
FBE	0.18991	0.05437	0.003767	13.75	0.1480
DC3	0.18991	0.05422	0.01766	10.22	0.5157

For the 1D MS model, we see that the most efficient method is the CN-RK4 method, followed closely by the CN-RK2 method. The CN-RK4 method is almost twice as efficient as the SBDF2 method. We observe that our third-order method, DC3, is not efficient compared to the second-order methods. Indeed, it takes almost 8-10 times more CPU time to compute the solution than the SBDF2 method for the chosen error. These conclusions hold for both levels of relative error.

We only tested the computational time of the FBE method at 1% relative L^2 error because it would take extremely small time-steps to reach the 0.1% relative error level. It is easy to figure that for 0.1% relative error, the FBE method is not efficient compared to high-order methods.

Table 12: CPU time of the numerical methods for the TNNP model in 1D for 0.005% relative L^2 error

Method	e_{L^2}	e_{H^1}	Δt	CPU time (s)	CPU/time-step (ms)
SBDF2	0.004 800	0.053 86	0.001 116	4.249	0.3953
RL-CNAB	0.004 802	0.054 67	0.000 834 5	6.863	0.4772
CN-RK4	0.004 785	0.049 52	0.002 487	14.33	2.971
CN-RK2	0.004 797	0.050 84	0.002 078	12.54	2.172
FBE	0.047 96	0.5968	6.417×10^{-5}	73.47	0.3929
DC3	0.003 961	0.045 10	0.001 667	9.389	1.304

For the 1D TNNP model, most methods require very small time-steps to have stable solutions. Therefore the relative error is very small compared to those of the previous models. We see that the most efficient method is, as for BR, the SBDF2 method. For 0.005% relative L^2 error, it is about 1.5 times faster than the next most efficient second-order method, RL-CNAB, nearly three times as fast as the CN-RK2 method, and 3.4 times faster than the CN-RK4 method. Interestingly, we observe that the DC3 method is the third most efficient method, since DC3 is unexpectedly accurate for TNNP.

We only tested the computational time of the FBE method at 0.05% relative L^2 error because it would take extremely small time steps to reach the error we chose for the other methods. Because of the stability constraints on the other methods, only the RL-CNAB can have an error as high as 0.05%. It would be easy to show that when RL-CNAB is in its asymptotic zone, it is more efficient than FBE. It is also easy to figure that for 0.005% relative error, the FBE method is not efficient compared to high-order methods.

For the 2D case with the BR model, all methods were stable enough to achieve a relative error of 0.1%. At this level of error, the SBDF2 method is, as in the 1D case, the most competitive, followed by RL-CNAB and the CN-RKn methods. The DCn methods are not competitive as they are not yet in their asymptotic region of convergence. The first order methods FBE and RL-FBE are not recommended. For a relative error of 1%, only the FBE, DC2 and RL methods can provide solutions at this exact error level. All other methods had their time step restricted by stability, resulting in error levels below 1%. Nevertheless, the SBDF2 and CN-RKn methods remains competitive with RL-CNAB, which is marginally more efficient in terms of CPU time. Accounting for the fact that SBDF2 is about four times more accurate for about the same CPU time as RL-CNAB,

Table 13: CPU time of the numerical methods for the BR model in 2D for 0.1% L^2 error

Method	e_{L^2}	e_{H^1}	Δt	CPU time (s)	CPU/time-step (ms)
SBDF2	0.1291	5.082	0.010 81	1.620	0.8759
CN-RK4	0.1285	4.220	0.023 53	3.241	3.813
CN-RK2	0.1317	4.486	0.017 86	2.652	2.367
DC2	0.1335	4.983	0.004 246	8.421	1.788
DC3	0.1349	5.108	0.010 81	5.106	2.760
RL-CNAB	0.1329	5.056	0.006 780	3.036	1.029
FBE	0.1338	5.080	0.000 500 0	30.12	0.7530
RL-FBE	0.1343	5.020	0.000 173 9	98.54	0.8568

Table 14: CPU time of the numerical methods for the BR model in 2D for 1% L^2 error

Method	e_{L^2}	e_{H^1}	Δt	CPU time (s)	CPU/time-step (ms)
SBDF2	0.2644	10.40	0.015 63	1.045	0.8162
CN-RK4	0.9222	30.28	0.062 50	1.191	3.723
CN-RK2	0.7081	24.05	0.040 00	1.130	2.260
DC2	1.272	46.95	0.015 63	2.023	1.580
DC3	0.5795	21.71	0.020 00	2.449	2.449
RL-CNAB	1.125	42.33	0.020 00	0.9516	0.9516
FBE	1.344	50.47	0.005 000	2.978	0.7444
RL-FBE	1.231	45.43	0.001 600	10.51	0.8410

Table 15: CPU time of the numerical methods for the BR model in 3D for 0.1% L^2 error

Method	e_{L^2}	e_{H^1}	Δt	CPU time (s)	CPU/time-step (ms)
SBDF2	0.1135	3.776	0.010 000	137.4	68.69
CN-RK4	0.2214	7.186	0.024 54	168.5	206.7
CN-RK2	0.2227	7.252	0.018 69	167.5	156.5
DC2	0.2202	7.285	0.004 386	703.2	154.2
DC3	0.2273	7.518	0.011 66	436.4	254.5
RL-CNAB	0.2223	7.381	0.007 605	192.3	73.10
FBE	0.2596	8.924	0.000 475 1	2090	49.64
RL-FBE	0.2771	9.317	0.000 177 5	5555	49.32

this is again the winning method. The RL-FBE method is to be avoided due to its lack of accuracy, requiring a time step ten times smaller than SBDF2 to reach 1% error. In 3D, both with a relative error of 0.1% and 1%, all the comments made above for the 2D case on stability, accuracy and efficiency ordering of the methods remain valid, except that at 1% error the RL-CNAB method is twice as fast as SBDF2 but with a L^2 error twenty times larger. Also, at 1% error, the gap in efficiency between first and second order methods is reduced.

For the 2D case with the TNNP model, the only methods that could provide a solution

Table 16: CPU time of the numerical methods for the BR model in 3D for 1% L^2 error

Method	e_{L^2}	e_{H^1}	Δt	CPU time (s)	CPU/time-step (ms)
SBDF2	0.1135	3.776	0.010 000	139.0	69.50
CN-RK4	0.3601	11.70	0.031 25	133.8	209.1
CN-RK2	0.6434	20.98	0.031 25	103.0	160.9
DC2	1.957	64.15	0.015 63	198.7	155.2
DC3	0.4638	15.31	0.015 63	318.2	248.6
RL-CNAB	2.253	74.75	0.024 39	63.27	77.16
FBE	2.246	76.94	0.004 808	257.7	61.96
RL-FBE	2.238	75.10	0.001 786	676.5	60.40

Table 17: CPU time of the numerical methods for the TNNP model in 2D for 0.1% L^2 error

Method	e_{L^2}	e_{H^1}	Δt	CPU time (s)	CPU/time-step (ms)
SBDF2	0.002 167	0.099 24	0.000 600 0	40.50	1.620
CN-RK4	0.003 589	0.1562	0.002 000	58.11	7.748
CN-RK2	0.003 190	0.1400	0.001 563	42.75	4.454
DC2	0.019 77	0.8941	0.000 899 8	54.05	3.242
DC3	0.000 772 2	0.035 45	0.000 899 8	84.54	5.072
RL-CNAB	0.1474	6.832	0.005 245	5.666	1.981
FBE	0.1363	6.512	0.000 403 2	57.93	1.557
RL-FBE	0.1372	6.331	0.000 154 4	169.2	1.742

Table 18: CPU time of the numerical methods for the TNNP model in 2D for 1% L^2 error

Method	e_{L^2}	e_{H^1}	Δt	CPU time (s)	CPU/time-step (ms)
SBDF2	0.002 167	0.099 24	0.000 600 0	40.50	1.620
CN-RK4	0.003 589	0.1562	0.002 000	58.11	7.748
CN-RK2	0.003 190	0.1400	0.001 563	42.75	4.454
DC2	0.019 77	0.8941	0.000 899 8	54.05	3.242
DC3	0.000 772 2	0.035 45	0.000 899 8	84.54	5.072
RL-CNAB	1.424	65.90	0.017 44	1.824	2.121
FBE	0.3170	15.13	0.000 949 4	25.14	1.591
RL-FBE	1.322	60.12	0.001 563	18.28	1.904

with a relative error of 0.1% were FBE and RL methods, while at 1% error only the RL methods are stable enough. FBE can provide solutions with such error levels not because it is more stable. FBE simply requires such a small time step for accuracy that this one falls within its stability region. The clear winner for CPU time is the RL-CNAB method. Among the other higher order methods, SBDF2 wins in terms of CPU time, this one being seven times larger than for RL-CNAB at 0.1% error but with a L^2 error 70 times smaller than for RL-CNAB. Comparing our 1D and 2D results for the TNNP model, second order

methods are nearly equally competitive at low error levels, but it is obvious that the lack of stability of non-RL methods makes them less attractive for solving the stiffest ionic models with larger numerical error levels and affordable time steps.

The difference in the performance of the methods between the 1D and 2D/3D cases is due to the relative cost of solving the linear systems in 1D versus 2D/3D, compared to the cost of evaluating the ionic functions. Besides the different linear solvers used, the implementation is done with MATLAB in 1D versus the compiled language Fortran for the 2D/3D code. In 1D, we use a direct solver with a tridiagonal matrix that makes the cost of solving the linear system about 1/16 of doing one evaluation of the BR reaction terms. In 2D, we use an iterative solver that makes the cost of solving the linear system about 3 times more expensive than doing one evaluation of the reaction terms.

5 Conclusion

In this article, we provide a strategy to practically determine precise values of the critical time step for linearly implicit time-stepping methods applied to reaction-diffusion equations. The methodology requires solving numerically the reaction-diffusion equation in dimension one, but gives critical time steps that equally applies to n-dimensional settings. The efficiency of the methodology is illustrated using the monodomain model with ionic models of varying stiffness.

We also investigated the impact of ionic model complexity and stiffness on finding an efficient numerical solution through a variety of time-stepping methods. The simplest model that we tested is the MS model. Its stiffness is low and hence there is no need for very stable methods to be used. The most accurate method to solve this problem is Strang splitting using Runge-Kutta methods (RK2 or RK4) to solve the ionic model and the reaction part of the monodomain model, and Crank-Nicholson (CN) for the diffusion part of the monodomain model. Due to the nature of operator splitting methods, they become less competitive when solving more complex models. Indeed, for each time-step, we do two RK substeps which, for the second-order RK method, implies computing the ionic currents four times, and for the original fourth order RK, eight times. As the models get more complex and realistic, computing the ionic currents takes up most of the computational cost of the algorithms. Therefore, multistep methods such as SBDF2 become more efficient with respect to computational time when solving more complex models because the ionic currents are only computed once per time-step. For the Beeler-Reuter model, the SBDF2, RL-CNAB and the Strang splitting methods are the most efficient unless a coarse solution is needed. The easier implementation of the SBDF2 scheme makes it faster with interpreted languages such as MATLAB. For the TNNP model, the SBDF2 method outclasses the other methods in terms of efficiency when a very accurate solution is required, while the RL-CNAB methods clearly wins for more affordable simulations at larger error levels. Therefore, the choice of the time-stepping method depends on the stiffness of the model. Strang splitting methods are best for the models with less stiffness, while SBDF2 becomes better as the stiffness increases and RL-CNAB is the obvious choice for the stiffest models where stability is critical. The third-order DC3 method is the most efficient only in the case of extremely accurate

solutions (not shown here). In practice, simulations do not require this much accuracy as the balance between modelling and numerical error allows for a lower level of numerical accuracy.

For the models studied here, using the largest possible time-steps needed for the stability of the semi-implicit methods results in solutions that are accurate enough relative to modelling errors. If this level of accuracy is satisfactory and the remaining goal is to reduce computational time, the stability of the methods is thus the important factor. The stiffness of the more complex models calls for more stable methods, such as Rush-Larsen methods. As we have seen in Section 3, these are almost unconditionally stable. However, in Section 4, we saw that they are less accurate than other semi-implicit methods such as SBDF2 or Strang splitting methods. For the stiffer models such as TNNP, a lower level of accuracy is not available with the less stable SBDF2 and Strang splitting methods, because even at the largest possible time-step necessary for stability, the errors are extremely low. Such accuracy is not necessary in most cases, for which Rush-Larsen type methods with larger time-steps are an obvious winner in terms of computational time.

References

[1] The finite element code choral is available at http://cpierre1.perso.univ-pau.fr/ or alternatively at https://plmlab.math.cnrs.fr/cpierre1/choral.

[2] U. M. Ascher, S. J. Ruuth, and B. T. R. Wetton. Implicit-explicit methods for time-dependent partial differential equations. *SIAM J. Numer. Anal.*, 32(3):797–823, 1995.

[3] G. W. Beeler and H. Reuter. Reconstruction of the action potential of ventricular myocardial fibres. *J. Physiol.*, 268:177–210, 1977.

[4] J. W. Cain. Taking math to heart: mathematical challenges in cardiac electrophysiology. *Notice of the AMS*, 58(4):542–549, 2011.

[5] Statistics Canada. Mortality, summary list of causes, 2008.

[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. Ethier and Y. Bourgault. Semi-implicit time-discretization schemes for the bidomain model. *SIAM J. Numer. Anal.*, 46(5):2443–2468, 2008.

[8] J.-L. Guermond and P. Minev. High-order time stepping for the incompressible navier–stokes equations. *SIAM Journal on Scientific Computing*, 37(6):A2656–A2681, 2015.

[9] E. Hairer, S.P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*, volume 8 of *Springer Series in Comp. Math.* Springer, 1993.

[10] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff Problems and Differential-Algebraic Problems*, volume 14 of *Springer Series in Comp. Math.* Springer, 1996.

[11] M. Hanslien, R. Artebrant, A. Tveito, G. T. Lines, and X. Cai. Stability of two time-integrators for the Aliev-Panfilov system. *Int. J. Numer. Anal. and Model.*, 8(3):427–442, 2011.

[12] A.L. Hodgkin and A.F. Huxley. A quantitative description of membrane current and its applications to conduction and excitation in nerves. *J. Physiol.*, 117:500–544, 1952.

[13] J. Keener and J. Sneyd. *Mathematical Physiology*. Springer, 2004.

[14] J.P. Keener and K. Bogar. A numerical method for the solution of the bidomain equations in cardiac tissue. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 8(1):234–241, 1998.

[15] W. Kress and B. Gustafsson. Deferred correction methods for initial boundary value problems. *J. Sci. Comp.*, 17(1-4):241–251, 2002.

[16] M.E. Marsh, S.Z. 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.

[17] C. Mitchell and D. Schaeffer. A two-current model for the dynamics of cardiac membrane. *Bull. Math. Bio.*, 65:767–793, 2003.

[18] M. Perego and A. Veneziani. An efficient generalization of the Rush-Larsen method for solving electro-physiology membrane equations. *Electronic Transactions on Numerical Analysis*, 35:234–256, 2009.

[19] S. Puwal and B.J. Roth. Forward Euler stability of the bidomain model of cardiac tissue. *IEEE Transactions on Biomedical Engineering*, 54(5):951, 2007.

[20] T. Roy. Time-Stepping Methods in Cardiac Electrophysiology. Master's thesis, University of Ottawa, Canada, 2015.

[21] S. Rush and H. Larsen. A practical algorithm for solving dynamic membrane equations. *IEEE Transactions on Biomedical Engineering*, 25(4):389–392, 1978.

[22] H.J. Schroll, G.T. Lines, and A. Tveito. On the accuracy of operator splitting for the monodomain model of electrophysiology. *International Journal of Computer Mathematics*, 84(6):871–885, 2007.

[23] R.J. Spiteri and R. Dean. Stiffness analysis of cardiac electrophysiological models. *Ann. Biomed. Eng.*, 38:3592–3604, 2010.

[24] R.J. Spiteri and R.C. Dean. On the performance of an implicit–explicit Runge–Kutta method in models of cardiac electrical activity. *IEEE Transactions on Biomedical Engineering*, 55(5):1488–1495, 2008.

[25] G. Strang. On the construction and comparison of difference schemes. *SIAM J. Numer. Anal.*, 5:506–517, 1968.

[26] 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.

[27] J. Sundnes, G.T. Lines, B.F. Nielsen X. Cai, K.-A. Mardal, and A. Tveito. *Computing the Electrical Activity in the Heart*, volume 1 of *Monogr. Comput. Sci. Eng.* Springer, 2006.

[28] K.H.W. ten Tusscher, D. Noble, P.J. Noble, and A. Panfilov. A model for human ventricular tissue. *Am. J. Physiol. Heart. Circ. Physiol.*, 286:H1573–H1589, 2004.

[29] S. Torabi Z., M.E. Marsh, J. Sundnes, and R.J. Spiteri. Stable time integration suppresses unphysical oscillations in the bidomain model. *Computational Physics*, 2:40, 2014.

[30] J.A. Trangenstein and C. Kim. Operator splitting and adaptive mesh refinement for the Luo–Rudy I model. *Journal of Computational Physics*, 196(2):645–679, 2004.

[31] R.L. Winslow, D.F. Scollan, A. Holmes, C.K. Yung, J. Zhang, and M.S. Jafri. Electrophysiological modeling of cardiac ventricular function: from cell to organ. *Annual Review of Biomedical Engineering*, 2:119, 2000.