Accelerating the Simulations of Cardiac Arrhythmia with a Second-Order Numerical Method and High-Performance Computing

Guilherme Martins Couto(✉), Noemi Zeraick Monteiro, Bernardo Martins Rocha, and Rodrigo Weber dos Santos

Federal University of Juiz de Fora, Juiz de Fora, MG, Brazil
couto.guilherme@engenharia.ufjf.br,
{nzmonteiro,bernardomartinsrocha}@ice.ufjf.br, rodrigo.weber@ufjf.edu.br

Abstract. Cardiac arrhythmia is a complex and potentially fatal condition affecting millions worldwide. Computational simulations are an essential tool for understanding the mechanisms of arrhythmia and developing new treatments. However, simulating arrhythmia with high accuracy can be computationally demanding. This paper presents a second-order numerical method for simulating arrhythmia that is parallelized using OpenMP. We demonstrate the effectiveness of our method by simulating arrhythmia in a two-dimensional cardiac tissue domain. Our results show that our method is more efficient than classical methods, with significant reductions in execution times and numerical errors. Additionally, we investigate the vulnerable window, a critical period during which the heart is most susceptible to arrhythmia, and show how the new method improves the computation of this metric. In summary, our novel second-order numerical method, parallelized using OpenMP, has demonstrated significant improvements in the efficiency of simulating cardiac arrhythmia. By providing faster and more accurate simulations, this work has the potential to accelerate research and aid in the development of new treatments for this complex condition.

Keywords: Second-order method · Spiral waves · Vulnerability window · Parallel computing

1 Introduction

Recent advances in computational biology have led to the development of computational models that aid in understanding complex phenomena of cardiac electrophysiology, such as the formation of reentry waves associated with cardiac arrhythmias. These reentry waves require triggers, such as premature ventricular extrasystoles on specific timings. The time window where an extrasystole induces cardiac arrhythmias is called vulnerability window [1].

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
O. Gervasi et al. (Eds.): ICCSA 2023 Workshops, LNCS 14112, pp. 103–113, 2023.
https://doi.org/10.1007/978-3-031-37129-5_9

Modeling cardiac tissue is challenging as the transmembrane potential signal requires interconnected intracellular and extracellular regions to allow for continuous exchange of currents. To simplify the complex microstructure, homogenization techniques are often used to produce smoothed equations that describe the dynamics of the transmembrane potentials. The monodomain model, a nonlinear system of partial differential equations (PDEs), is one of the most used homogenized models.

This paper presents a second-order numerical method based in [2], named Second-order Semi-Implicit Alternating Direction Implicit (SSI_{ADI}), for simulating cardiac arrhythmias with the monodomain model. We parallelized our novel method using OpenMP [3], resulting in significant improvements in the efficiency of cardiac simulations. In particular, our results demonstrate faster and more accurate simulations, when compared to the commonly used forward Euler (FE) method.

In Sect. 2, we present the monodomain equations. Section 3 describes the numerical methods, including the classic FE and the SSI_{ADI}. Section 4 presents the results of the vulnerability window and the performance of the methods, along with the speedups due to the parallel programming. We discuss the results and draw some conclusions.

2 Mathematical Model

We use the monodomain to model the cardiac tissue. Assuming that the intracellular and the extracellular conductivity tensors, σ_i and σ_e, are proportional, $\sigma_i = \alpha\sigma_e$, with α being a constant, the monodomain is given by:

$$\chi \left(C_m \frac{\partial V}{\partial t} + I_{ion} - I_s \right) = \nabla \cdot \sigma \nabla V, \tag{1}$$

where $\sigma = \frac{\sigma_i}{\sigma_i + \sigma_e}\sigma_e = \frac{\alpha}{1+\alpha}\sigma_e$, χ is the surface-to-volume ratio of cells (mm^{-1}), C_m is the specific membrane capacitance per unit area ($\mu\text{F mm}^{-2}$), V is the membrane voltage (mV), I_{ion} is the ionic current ($\mu\text{A mm}^{-2}$), σ is the conductivity tensor (mS mm^{-1}), and I_s is the stimulation density ($\mu\text{A mm}^{-2}$) [4].

FitzHugh-Nagumo (FHN) equations [5] are used to model the ionic current, I_{ion}. The FHN model is able to reproduce the dynamics of the action potential (AP) in a simplified form, using only two variables, one fast (v) and one slow (w). The fast variable has a cubic nullcline and is called the excitation variable, while the slow variable is called the recovery variable and has a monotonically increasing nullcline. The nullclines have a single point of intersection, which, without loss of generality, is considered at the origin.

Traditionally, the solution waveform of the FHN system shows a hyperpolarization of the potential in the refractory period. However, this feature is not characteristic of the cardiac AP and may adversely affect the model's recovery properties, particularly in reentrant activation patterns. Therefore, to adapt the

model to the problem of cardiac AP, the hyperpolarization is suppressed with an adaptation presented before in [6] and [7], leading to the following equations:

$$\frac{\partial v}{\partial t} = Gv\left(1 - \frac{v}{v_{th}}\right)\left(1 - \frac{v}{v_p}\right) + \eta_1 vw, \tag{2}$$

$$\frac{\partial w}{\partial t} = \eta_2\left(\frac{v}{v_p} - \eta_3 w\right), \tag{3}$$

where G, η_1, η_2 and η_3 are positive coefficients, v_{th} is the threshold and v_p is the peak potential.

To model cardiac tissue electrophysiology, we couple the FHN Eq. (2) with the monodomain Eq. (1) and apply Neumann boundary conditions, obtaining

$$C_m\frac{\partial v}{\partial t} = \frac{1}{\chi}\nabla\cdot\sigma\nabla v - Gv\left(1 - \frac{v}{v_{th}}\right)\left(1 - \frac{v}{v_p}\right) - \eta_1 vw + I_s, \tag{4}$$

$$\frac{\partial w}{\partial t} = \eta_2\left(\frac{v}{v_p} - \eta_3 w\right), \qquad x \in \Gamma, \tag{5}$$

$$\partial_{\mathbf{n}} v(x,t) = 0, \qquad x \in \partial\Gamma, \tag{6}$$

where Γ is the domain, and $\mathbf{n}$ is the normal with respect to the boundary $\partial\Gamma$.

3 Numerical Methods

We use FE with operator splitting and SSI_{ADI} methods, discussed below, to solve the system (4)–(6) for a 2D square tissue with $\Delta x = \Delta y$.

3.1 Forward Euler

In the Forward Euler method, also known as *explicit Euler*, the first-order derivative is approximated by progressive differences. Then, Eq. (5) is solved by

$$\frac{w_{i,j}^{n+1} - w_{i,j}^{n}}{\Delta t} = \eta_2\left(\frac{v_{i,j}^{n}}{v_p} - \eta_3 w_{i,j}^{n}\right). \tag{7}$$

Using $\delta_z^2 v_k = \left(v_{k-1} - 2v_k + v_{k+1}\right)/\Delta z^2$ to represent a generic centered second-order derivative approximation, the Eq. (4) is solved in two steps. The first one deals with the reaction term:

$$\frac{v_{i,j}^{*} - v_{i,j}^{n}}{\Delta t} = \frac{1}{C_m}\left[-Gv_{i,j}^{n}\left(1 - \frac{v_{i,j}^{n}}{v_{th}}\right)\left(1 - \frac{v_{i,j}^{n}}{v_p}\right) - \eta_1 v_{i,j}^{n} w_{i,j}^{n} + I_s\right]. \tag{8}$$

The second step refers to the diffusion term:

$$\frac{v_{i,j}^{n+1} - v_{i,j}^{*}}{\Delta t} = \frac{\sigma}{\chi C_m}\left(\delta_x^2 v_{i,j}^{*} + \delta_y^2 v_{i,j}^{*}\right). \tag{9}$$

This scheme is conditionally stable, first-order accurate in time and second-order in space [8].

3.2 SSI-Alternating Direction Implicit

We applied the SSI_{ADI} to Eq. (4), and, to (5), the Heun's method (modified forward Euler). Both are second-order accurate [8].

First, denoting by $\mathcal{R}$ the reaction term, a prediction v^* is calculated from Eq. (4) by explicit Euler scheme

$$v^* = v^n + \frac{\Delta t}{2\chi C_m}(\nabla \cdot \sigma \nabla v^n) + \frac{\Delta t}{2}\mathcal{R}(v^n, w^n). \tag{10}$$

Then, considering $\phi = \dfrac{\Delta t}{2}\dfrac{\sigma}{\chi C_m \Delta x^2}$, and

$$\mathcal{A}^1(v) = \left(v_{i-1,j}^{n+\frac{1}{2}} - 2v_{i,j}^{n+\frac{1}{2}} + v_{i+1,j}^{n+\frac{1}{2}}\right) + \left(v_{i,j-1}^{n} - 2v_{i,j}^{n} + v_{i,j+1}^{n}\right), \tag{11}$$

$$\mathcal{A}^2(v) = \left(v_{i,j-1}^{n+1} - 2v_{i,j}^{n+1} + v_{i,j+1}^{n+1}\right) + \left(v_{i-1,j}^{n+\frac{1}{2}} - 2v_{i,j}^{n+\frac{1}{2}} + v_{i+1,j}^{n+\frac{1}{2}}\right). \tag{12}$$

The resulting system of linear equations is solved with second-order ADI method [8]:

$$v_{i,j}^{n+\frac{1}{2}} - v_{i,j}^{n} = \phi\mathcal{A}^1(v) + \frac{\Delta t}{2}\mathcal{R}(v^*, w^*), \tag{13}$$

$$v_{i,j}^{n+1} - v_{i,j}^{n+\frac{1}{2}} = \phi\mathcal{A}^2(v) + \frac{\Delta t}{2}\mathcal{R}(v^*, w^*), \tag{14}$$

When rearranged, (13) and (14) can be expressed in the compact form:

$$-\phi v_{i-1,j}^{n+\frac{1}{2}} + (1+2\phi)v_{i,j}^{n+\frac{1}{2}} - \phi v_{i+1,j}^{n+\frac{1}{2}} = \mathbf{F}_i^n, \tag{15}$$

$$-\phi v_{i,j-1}^{n+1} + (1+2\phi)v_{i,j}^{n+1} - \phi v_{i,j+1}^{n+1} = \mathbf{F}_j^{n+\frac{1}{2}}, \tag{16}$$

where $\mathbf{F}_i^n$ and $\mathbf{F}_j^{n+\frac{1}{2}}$ are the following vectors:

$$\mathbf{F}_i^n = \phi v_{i,j-1}^{n} + (1-2\phi)v_{i,j}^{n} + \phi v_{i,j+1}^{n} + \frac{\Delta t}{2}\mathcal{R}(v^*, w^*), \tag{17}$$

$$\mathbf{F}_j^{n+\frac{1}{2}} = \phi v_{i-1,j}^{n+\frac{1}{2}} + (1-2\phi)v_{i,j}^{n+\frac{1}{2}} + \phi v_{i+1,j}^{n+\frac{1}{2}} + \frac{\Delta t}{2}\mathcal{R}(v^*, w^*). \tag{18}$$

Note that the matrices associated with each system of Eqs. (15) and (16) are tridiagonal. For example, the tridiagonal system of (15) has the following form:

$$\begin{bmatrix} 1+2\phi & -\phi & \dots & \dots & \mathbf{0} \\ -\phi & 1+2\phi & -\phi & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & -\phi & 1+2\phi & -\phi \\ \mathbf{0} & \dots & \dots & -\phi & 1+2\phi \end{bmatrix} \begin{bmatrix} \mathbf{X}_1^{n+\frac{1}{2}} \\ \mathbf{X}_2^{n+\frac{1}{2}} \\ \vdots \\ \mathbf{X}_N^{n+\frac{1}{2}} \\ \mathbf{X}_{N+1}^{n+\frac{1}{2}} \end{bmatrix} = \begin{bmatrix} \mathbf{F}_1^n \\ \mathbf{F}_2^n \\ \vdots \\ \mathbf{F}_N^n \\ \mathbf{F}_{N+1}^n \end{bmatrix}, \tag{19}$$

where $\mathbf{X}_k$ for $k = 1, \ldots, N+1$ is the system solution. The two tridiagonal linear systems are solved with Thomas algorithm [9].

Moreover, it is possible to prove, from [2], that SSI_{ADI} is an unconditionally stable method. However, in absence of diffusive terms, it degenerates into Heun's method, which is conditionally stable. This implies conditional stability in this work.

4 Numerical Experiments

The numerical experiments of this work were carried out considering an isotropic cardiac tissue of 2×2 cm. The spatial discretization is $\Delta x = \Delta y = 0.02$ cm and the total time of the simulation is $T = 400$ ms. The numerical methods were evaluated with different time steps. Each method was executed in two different phases. For the FE, the first part consists in calculating (7) and (8). Equation (9) is computed in the second part. On the other hand, for the SSI_{ADI}, initially the prediction (10) and the Heun's method are calculated. Secondly, for the diffusion, Eqs. (15) and (16) are solved.

Simulations were performed using the following set of parameters [6] (Table 1):

Table 1. Parameters

Parameter	Values
G	$1.5\ \Omega^{-1}\text{cm}^{-2}$
η_1	$4.4\ \Omega^{-1}\text{cm}^{-1}$
η_2	0.012
η_3	1
v_{th}	13 mV
v_p	100 mV
σ	$1.2 \cdot 10^{-3}\ \Omega^{-1}\ \text{cm}^{-1}$
χ	$10^3\ \text{cm}^{-1}$
C_m	$10^{-3}\ \text{mFcm}^{-2}$

Numerical errors and execution times for the calculation of both parts of the solution algorithms (the first part is the reaction term, while the second is the diffusion term) were analyzed. The parallel speedups were also measured. The code was implemented in C and parallelized with OpenMP. Simulations were performed in a computer equipped with a processor Intel® Core™ i5-12600.

4.1 Vulnerability to Reentry

In this study, we assessed how the time discretization and the choice of the method affect the vulnerability to reentry. To this end, we considered time steps varying from $\Delta t = 0.02$ to $\Delta t = 0.2$, and observed the velocity of the first stimulus S1. This relates to the vulnerability window in which the second stimulus S2 generates reentry. In these simulations, the S1 stimulus is applied at $t = 0$, during 1 ms, in the left edge $[0, 0.2] \times [0, 2]$. Then, the second stimulus S2 is

applied during 3 ms in the quarter $[0, 1] \times [0, 1]$, at time t_{s2}. We vary the time t_{s2} with a discretization of 1 ms to calculate the vulnerable window, i.e., the times t_{s2} that lead to cardiac arrhythmia.

Table 2. Vulnerable window for reentry

Δt (ms)	S1 velocity (m/s)		Vulnerability window (ms)	
	FE	SSI_{ADI}	FE	SSI_{ADI}
0.02	0.599600	0.594845	[119, 130]	[119, 130]
0.04	0.601604	0.594452	[119, 130]	[119, 130]
0.06	0.603622	0.594059	[119, 130]	[119, 130]
0.08	0.606469	0.592105	[119, 130]	[119, 130]
0.10	–	0.592105	–	[119, 130]
0.20	–	0.580645	–	[120, 130]

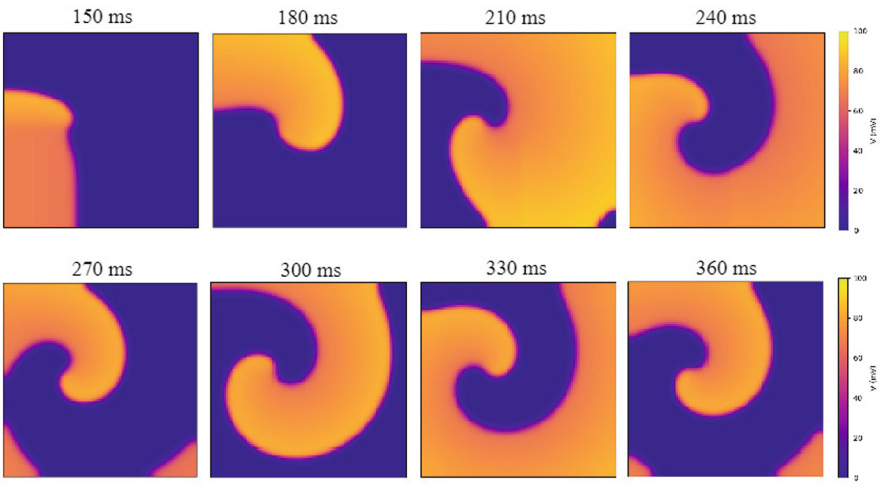

Fig. 1. Spiral snapshots - SSI_{ADI} method, with S2 applied at $t = 125$ ms

The results are presented in Table 2. We report instabilities with "−" in the tables. One can observe that the FE method presents an unstable behavior for large time discretizations, i.e., for $t > 0.08$.

It is possible to observe the reentry forming auto-sustained spiral waves, as indicated in the snapshots of the numerical solution shown in Fig. 1. However, there is also figure-of-eight formations in the right extreme of the windows, as showed in snapshots in Fig. 2.

4.2 Error, Execution Times and Parallel Performance

The S2 stimulus was applied at $t = 120$ ms for all tests in this Subsection. The error and the execution time of each method were analyzed for different Δt. Since

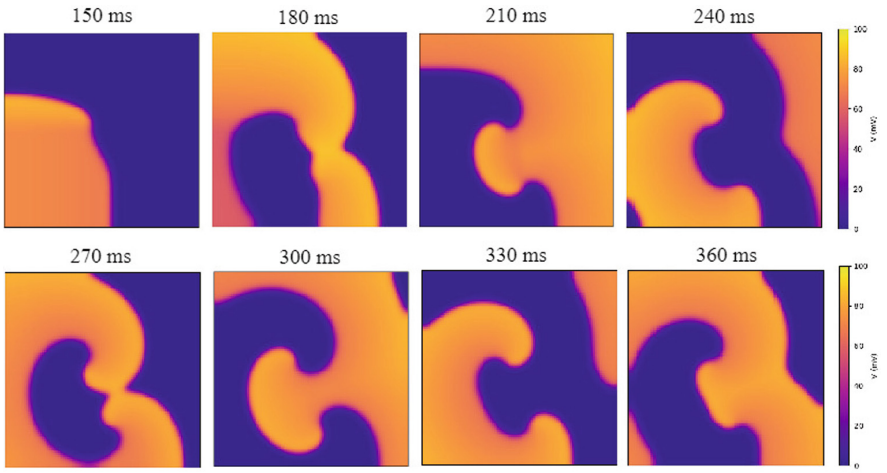

Fig. 2. Figure-of-eight snapshots - SSI_{ADI}, with S2 applied at $t = 130$ ms

there is no analytical solution for this problem, we consider the results obtained with FE using a small time step, of $\Delta t = 0.005$, as the reference solution. For the errors, we use the metric Relative Error in Percentage (REP):

$$REP = 100 \times \frac{\sqrt{\sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \left(v_{i,j} - \hat{v}_{i,j}\right)^2}}{\sqrt{\sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \hat{v}_{i,j}^2}}, \tag{20}$$

where $\hat{v}$ is the reference solution, and N_x, N_y are the number of spatial steps in each direction. The results are given by Table 3 and Fig. 3.

Table 3. Relative Error in Percentage (REP)

Δt (ms)	FE	SSI_{ADI}
0.02	1.6339	1.8227
0.04	3.9438	1.8008
0.06	6.0675	2.0656
0.08	8.7651	2.0854
0.10	–	1.9852
0.20	–	3.3317

It is important to observe that Table 3 and Fig. 3 do not concern the convergence study of methods. The goal is to see the relative error versus the time step, maintaining $\Delta x = \Delta y = 0.02$ fixed. We note that the error of SSI_{ADI} with $\Delta t = 0.2$ is yet bellow the error of FE with $\Delta t = 0.04$.

For the parallel executions, the simulations were executed with 1, 2, 4 and 6 cores. The results for FE and SSI_{ADI} are exhibited in Tables 4 and 5.

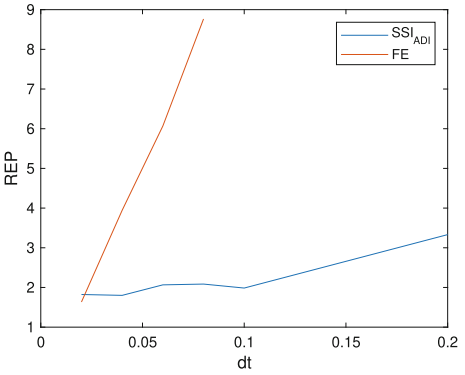

Fig. 3. Relative Error in Percentage (REP) versus time step for each method

Table 4. Total execution time with different thread numbers for FE (s)

Δt (ms)	1	2	4	6
0.02	2.500	1.338	0.826	0.639
0.04	1.293	0.713	0.480	0.391
0.06	0.874	0.495	0.362	0.307
0.08	0.676	0.396	0.307	0.292

Table 5. Total execution time with different thread numbers for SSI_{ADI} (s)

Δt (ms)	1	2	4	6
0.02	8.768	4.550	2.534	1.924
0.04	4.409	2.298	1.332	1.026
0.06	2.968	1.560	0.926	0.734
0.08	2.237	1.177	0.733	0.583
0.10	1.806	0.964	0.612	0.494
0.20	0.931	0.518	0.373	0.317

For both methods, this total execution times come from two parts, or steps, as presented in Sect. 3. However, as the steps are different for the FE and the SSI_{ADI} methods, each part has a different influence on the total execution time. Tables 6 and 7 show the proportion of each numerical step when executing the codes sequentially.

Next, we present speedup results for the particular case $\Delta t = 0.02$ ms in Tables 8a and 8b.

5 Discussion

For each time step and method, we analyse the vulnerable window for reentry. For this, we also study the S1 velocity in each case. Particularly, when $\Delta t = 0.2$

Table 6. Forward Euler proportion of time spent by each part (s)

	1st		2nd	
Δt (ms)	Time	% of total	Time	% of total
0.02	1.489	59.56	0.988	39.52
0.04	0.772	59.71	0.512	39.60
0.06	0.524	59.95	0.347	39.70
0.08	0.403	59.62	0.267	39.50

Table 7. SSI_{ADI} proportion of time spent by each part (s)

Δt (ms)	1st		2nd	
	Time	% of total	Time	% of total
0.02	3.640	41.51	5.126	58.46
0.04	1.833	41.57	2.575	58.40
0.06	1.233	41.54	1.735	58.46
0.08	0.927	41.44	1.309	58.52
0.10	0.749	41.47	1.057	58.53
0.20	0.387	41.57	0.544	58.43

Table 8. Speedups for each part with $\Delta t = 0.02$ ms

(a) FE method

Threads	1st	2nd	Total
2	1.886	1.875	1.868
4	3.030	3.174	3.027
6	4.113	3.959	3.912

(b) SSI_{ADI} method

Threads	1st	2nd	Total
2	1.948	1.914	1.927
4	3.556	3.406	3.460
6	4.806	4.421	4.557

ms, the SSI_{ADI} drives S1 stimulus at a velocity of 0.580645 m/s, which is still very close to the velocity observed with $\Delta t = 0.02$ ms, 0.594845 m/s. For $\Delta t = 0.2$, if S2 stimulus is applied at $t = 119$ ms, there is no spiral formation, but it is worth to note that SSI_{ADI} is able to compute almost the same window with larger Δt. It is also interesting to note, from Table 2, that the velocity of S1 is increasing with time step for FE, but decreasing with it for SSI_{ADI}.

In terms of performance, from Tables 3, 4 and 5, we note that, when comparing the two methods, the SSI_{ADI} was more efficient and accurate (see Table 3 and Fig. 3). To achieve an error smaller than 3.3317%, obtained by the fastest SSI_{ADI} solution ($\Delta t = 0.2$ ms), the FE takes 0.639 s, in the best case with 6 threads, while only 0.317 s are needed for the SSI_{ADI}. However, if the application demands an error lower than 5%, the FE performs slightly better and can delivery a result in 0.391 s, decreasing the distance to the SSI_{ADI}. Moreover, if the application has even greater flexibility, allowing an error of up to 10%, FE outperforms SSI_{ADI} and completes the simulation in just 0.292 s.

Tables 6 and 7 clearly show that FE execute all of its parts faster than SSI_{ADI}. These results are associated with the different approach each method adopts. While the first part of FE consists in calculating the ODE (7) and the reaction term (8), for the SSI_{ADI}, it involves the prediction (10), which has both diffusion and reaction, and the Heun's method. For the second part, the SSI_{ADI} also has more work to do, because it has to implicitly calculate the diffusion on one axis, as Eqs. (15) and (16), and to explicitly calculate the diffusion on the other, as shown in (17) and (18). The second part of FE only takes one diffusion computation, as shown in Eq. (9).

Also, it is possible to observe in Table 7 that the second part of the SSI_{ADI} took more time than the first one. This, however, occurs because the adapted FHN has only two variables. More realistic models, with multiple variables, take longer to compute the first part of the method as show [10].

In addition, the analysis of Tables 8a and 8b also allows observing that SSI_{ADI} has slightly more scalability with the increase in the number of threads. A speedup around 4.6 was obtained in a processor with only 6 cores. On the other hand, the maximum speed-up for FE was 3.9. These comes from the fact that both steps of SSI_{ADI} involve more calculations than those of the FE method.

6 Conclusions

In this study, we implemented the monodomain equation with an adapted version of the FitzHugh-Nagumo model to simulate cardiac cells. Two numerical methods, the Forward Euler and the SSI-Alternating Direction Implicit, were used and compared in both serial and parallel execution. We were able to reproduce reentry spiral auto-sustained and figure-of-eight formations in both methods, while also calculating the important metric of vulnerability window.

Our findings suggest that the SSI_{ADI} outperforms the FE method in terms of robustness and speed. Although the SSI_{ADI} is not commonly used for complex models, it is a powerful method for simpler ones in cardiac electrophysiology studies, due to its accuracy, stability, and speed compared to FE. Moreover, the parallel programming has the potential to enable fast simulations of heart dynamics and to support the development of new treatments and therapies for heart diseases, as demonstrated in [11].

Future research aims to test the SSI_{ADI} method with more complex cardiac cell models, such as the ten Tusscher-Noble-Noble-Panfilov model [12], which may provide more realistic results. Additionally, the results need to be extended to 3D simulations. In this regard, parallel programming with GPUs will be indispensable given the increasing complexity of models [10].

In summary, our newly developed second-order numerical method, parallelized with OpenMP, has demonstrated substantial improvements in the efficiency of simulating cardiac arrhythmia. By enabling faster and more accurate simulations, this work has the potential to accelerate research and facilitate the development of new treatments for this complex condition. Ultimately, our approach may aid in advancing our understanding of cardiac arrhythmia, leading to better clinical outcomes for patients.

Acknowledgements. This work has been supported by Universidade Federal de Juiz de Fora (UFJF), by a scholarship from Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) - Brazil - Finance Code 001. This research was funded by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e Empresa Brasileira de Serviços Hospitalares (Ebserh) grant numbers 423278/2021-5, 310722/2021-7, and 315267/2020-8; by Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG) - Brazil.

References

1. Tran, D.X., Yang, M.J., Weiss, J.N., Garfinkel, A., Qu, Z.: Vulnerability to reentry in simulated two-dimensional cardiac tissue: effects of electrical restitution and stimulation sequence. Chaos Interdisc. J. Nonlinear Sci. **17**(4), 043115 (2007)
2. Pereira, R.R.: Métodos de diferenças finitas para problemas de difusão e reação não lineares. Ph.D. thesis, Laboratório Nacional de Computação Científica, Petrópolis (2018)
3. OpenMP Homepage. https://www.openmp.org/. Accessed 10 Apr 2023
4. Keener, J., James, S. (eds.): Mathematical Physiology II: Systems Physiology, 2nd edn. Springer, New York (2009). https://doi.org/10.1007/978-0-387-79388-7
5. FitzHugh, R.: Impulses and physiological states in theoretical models of nerve membrane. Biophys. J. **1**(6), 445–466 (1961)
6. Franzone, P.C., Pavarino, L.F.: A parallel solver for reaction-diffusion systems in computational electrocardiology. Math. Models Methods Appl. Sci. **14**(06), 883–911 (2004)
7. Rogers, J.M., McCulloch, A.D.: A collocation-Galerkin finite element model of cardiac action potential propagation. IEEE Trans. Biomed. Eng. **41**(8), 743–757 (1994)
8. Strikwerda, J.C.: Finite Difference Schemes and Partial Differential Equations. Society for Industrial and Applied Mathematics, Edition. SIAM (2004)
9. Sultanian, B.: Thomas algorithm for solving a tridiagonal system of linear algebraic equations. In: Gas Turbines: Internal Flow Systems Modeling. Cambridge Aerospace Series, pp. 337–339. Cambridge University Press, Cambridge (2018). https://doi.org/10.1017/9781316755686.012
10. Oliveira, R.S., Rocha, B.M., Burgarelli, D., Meira Jr, W., Constantinides, C., dos Santos, R.W.: Performance evaluation of GPU parallelization, space-time adaptive algorithms, and their combination for simulating cardiac electrophysiology. Int. J. Numer. Methods Biomed. Eng. **34**(2), e2913 (2018)
11. Kaboudian, A., Cherry, E.M., Fenton, F.H.: Real-time interactive simulations of large-scale systems on personal computers and cell phones: toward patient-specific heart modeling and other applications. Sci. Adv. **5**(3), eaav6019 (2019)
12. Ten Tusscher, K.H., Panfilov, A.V.: Alternans and spiral breakup in a human ventricular tissue model. Am. J. Physiol.-Heart Circulatory Physiol. **291**(3), H1088–H1100 (2006)