High-Performance Cardiac Electrophysiology Simulation with SSI-ADI: Second-Order Accuracy and GPU-Driven Acceleration

Guilherme M Couto[1], Noemi Z Monteiro[1], Marcelo Lobosco[1], Bernardo M Rocha[1], Joventino O Campos[1], Rodrigo W dos Santos[1]

[1] Federal University of Juiz de Fora, Juiz de Fora, Brazil

Abstract

We present a numerical study of the Second-Order Semi-Implicit Alternating Direction Implicit (SSI-ADI) method applied to cardiac electrophysiology simulations. We compared its accuracy, stability, and performance against the Forward Euler (FE) and Operator Splitting ADI (OS-ADI) methods. Results showed that SSI-ADI achieved second-order temporal convergence with superior accuracy and stability. Its parallel implementation with CUDA provided a speedup of over 12x, enabling high-accuracy simulations. This balance of precision and efficiency makes SSI-ADI a compelling tool for large-scale and time-sensitive cardiac modeling applications.

1. Introduction

Computational modeling of cardiac electrophysiology plays a crucial role in developing new therapies, medical devices, and clinical procedures. As simulations become increasingly integrated into medical practice, the need for accurate and efficient numerical methods grows. However, the complexity of the equations that govern electrical propagation in cardiac tissue poses significant computational challenges [1, 2].

Among the most widely used models is the monodomain, a nonlinear reaction-diffusion (NRD) system that describes transmembrane potential dynamics across cardiac tissue. Solving this equation numerically requires careful consideration of both stability and efficiency. First-order methods, such as the FE and the OS-ADI, based on the Godunov splitting [3], are simple to implement but are often limited by stability constraints or reduced accuracy. Second-order methods, while more accurate, typically involve higher costs.

In this work, we evaluated the SSI-ADI method, developed for reaction-diffusion systems [4], applied to the monodomain model. We compared its accuracy, stability, and performance against the FE and OS-ADI methods. In a fine-resolution 2D experiment, the SSI-ADI method demonstrated superior accuracy and robustness. All methods were parallelized using OpenMP and CUDA, and a simulation was designed to replicate arrhythmogenic activity. To the best of our knowledge, this is the first application of the SSI-ADI method in cardiac electrophysiology with parallel execution on GPUs.

2. Methods

2.1. Mathematical model

The electrical activity of cardiac tissue was described in this work using the monodomain model, a widely adopted formulation that captures the propagation of the transmembrane potential V_m across cardiac cells. This model combines diffusive and reactive processes: the diffusive term, given by a partial differential equation (PDE), accounts for the spread of electrical signals, while the reactive term represents the ionic exchanges across the cell membrane, governed by nonlinear ordinary differential equations (ODEs). The monodomain equation is expressed as:

$$\begin{aligned} \frac{\partial V_m}{\partial t} &= D\nabla^2 V_m - I_{ion}(V_m, \boldsymbol{\eta}) + I_{stim}, \\ \frac{\partial \boldsymbol{\eta}}{\partial t} &= \boldsymbol{f}(V_m, \boldsymbol{\eta}), \end{aligned} \tag{1}$$

where D is the diffusion coefficient, I_{ion} denotes the total ionic current, I_{stim} is the external stimulation, and $\boldsymbol{\eta}$ represents the state variables that are governed by a set of ODEs, $\boldsymbol{f}$, associated with the cellular model.

In this study, we considered an isotropic diffusion regime with a constant coefficient $D = 0.65 \times 10^{-3}$ cm^2/s. To complete the formulation, initial conditions were specified for all dynamic variables, and the domain was subject to zero-flux Neumann boundary conditions, which are mathematically expressed as $\mathbf{n} \cdot D\nabla V_m = 0$, where $\mathbf{n}$ denotes the normal vector to the boundary. This imposes electrical isolation of the domain periphery.

To describe the ionic current I_{ion}, the monodomain was coupled with the Minimal Ventricular (MV) model, a simplified cellular model that captures key electrophysiolog-

ical features of the action potential of human ventricular cells using only four state variables. Its parameters were adjusted to reproduce the morphology of the more detailed ten Tusscher model, thus preserving biological plausibility while significantly reducing simulation costs [5].

2.2. Second-order semi-implicit ADI

To numerically solve the NRD system derived from the monodomain model, Equation (1), we employed the SSI-ADI method. This scheme was specifically designed to achieve second-order accuracy in both time and space [4], while also ensuring high computational efficiency, a key requirement for cardiac electrophysiology simulations.

The SSI-ADI framework split the Equation (1) into diffusion and reaction components, which were handled separately. The linear diffusion term was discretized using the Crank–Nicolson (CN) method, a well-established second-order scheme in time [6]. However, preserving second-order temporal accuracy for the full system also required a consistent treatment of the nonlinear reaction term. To this end, the second-order Runge–Kutta method (also known as the midpoint method) was adopted, evaluating the reaction term at the intermediate time $t_n + \Delta t/2$. This strategy led to the following coupled scheme:

$$\frac{V_m^{n+1} - V_m^n}{\Delta t} = \frac{D}{2}\left(\nabla^2 V_m^{n+1} + \nabla^2 V_m^n\right) + R(V_m^{n+\frac{1}{2}}, \boldsymbol{\eta}^{n+\frac{1}{2}}), \quad (2)$$

where $R(V_m^{n+\frac{1}{2}}, \boldsymbol{\eta}^{n+\frac{1}{2}}) = -I_{ion}(V_m^{n+\frac{1}{2}}, \boldsymbol{\eta}^{n+\frac{1}{2}}) + I_{stim}$ represents the reaction term.

The intermediate values $V_m^{n+\frac{1}{2}}$ and $\boldsymbol{\eta}^{n+\frac{1}{2}}$ were not known a priori and had to be approximated explicitly. The potential at the midpoint was estimated through a predictor step:

$$V_m^* = V_m^n + \frac{\Delta t}{2}\left(D\nabla^2 V_m^n + R(V_m^n, \boldsymbol{\eta}^n)\right). \quad (3)$$

In the MV model, the state variables are governed by Hodgkin–Huxley-type equations, and we integrated them using the Rush–Larsen method, which is particularly well-suited for stiff ODEs arising in ionic models [7]. The midpoint estimate $\boldsymbol{\eta}^*$ was computed as:

$$\boldsymbol{\eta}^* = \boldsymbol{\eta}_\infty^n - (\boldsymbol{\eta}_\infty^n - \boldsymbol{\eta}^n) exp\left(-\frac{\Delta t}{2\tau_\eta^n}\right), \quad (4)$$

and the updated value at t_{n+1} was given by:

$$\boldsymbol{\eta}^{n+1} = \boldsymbol{\eta}_\infty^* - (\boldsymbol{\eta}_\infty^* - \boldsymbol{\eta}^n) exp\left(-\frac{\Delta t}{\tau_\eta^*}\right). \quad (5)$$

To further enhance computational efficiency, the ADI strategy was applied to decompose the two-dimensional problem described by Equation (2) into a sequence of one-dimensional problems at each time step. This decomposition resulted in two successive tridiagonal systems:

$$-\phi V_{m_{i-1,j}}^{n+\frac{1}{2}} + (1+2\phi) V_{m_{i,j}}^{n+\frac{1}{2}} - \phi V_{m_{i+1,j}}^{n+\frac{1}{2}} = \mathbf{F}_i^n, \quad (6)$$

$$-\phi V_{m_{i,j-1}}^{n+1} + (1+2\phi) V_{m_{i,j}}^{n+1} - \phi V_{m_{i,j+1}}^{n+1} = \mathbf{F}_j^{n+\frac{1}{2}}, \quad (7)$$

where $\phi = \dfrac{D\Delta t}{2\Delta x^2}$, assuming $\Delta x = \Delta y$. The right-hand sides (RHSs) $\mathbf{F}_i^n$ and $\mathbf{F}_j^{n+\frac{1}{2}}$ were given by:

$$\mathbf{F}_i^n = \phi V_{m_{i,j-1}}^n + (1-2\phi) V_{m_{i,j}}^n + \phi V_{m_{i,j+1}}^n + \frac{\Delta t}{2} R(V_m^*, \boldsymbol{\eta}^*), \quad (8)$$

$$\mathbf{F}_j^{n+\frac{1}{2}} = \phi V_{m_{i-1,j}}^{n+\frac{1}{2}} + (1-2\phi) V_{m_{i,j}}^{n+\frac{1}{2}} + \phi V_{m_{i+1,j}}^{n+\frac{1}{2}} + \frac{\Delta t}{2} R(V_m^*, \boldsymbol{\eta}^*). \quad (9)$$

In Equation (6), the diffusion operator was applied implicitly in the y-direction, while the RHS $\mathbf{F}_i^n$, Equation (8), included an explicit treatment of diffusion in the x-direction. Conversely, Equation (7) applied diffusion implicitly in the x-direction, with its RHS $\mathbf{F}_j^{n+\frac{1}{2}}$, Equation (9), incorporating diffusion explicitly in the y-direction. This alternating implicit-explicit treatment across directions allowed for stable time integration while avoiding the computational cost of solving fully two-dimensional systems.

2.3. Parallelization of the SSI-ADI method

To accelerate the numerical solution of the SSI-ADI method, we implemented a CUDA-based parallel version of the algorithm in C.

The parallel implementation was organized around three main CUDA kernels. The first kernel was responsible for computing the nonlinear reaction term $R(V_m, \boldsymbol{\eta})$, using the explicit scheme described in Equations (3) and (4). Additionally, this kernel updates the state variables according to Equation (5). Each thread handled the complete set of operations for a single spatial point (i, j), ensuring data locality and minimizing global memory traffic. The computed reaction term was stored for use in the subsequent ADI diffusion stages.

The second kernel constructed the RHS of the linear systems in the ADI steps and appeared in two variants, one for each direction. The first one computed Equation (8), performing explicit diffusion in the x-direction and incorporating half of the previously computed reaction term. The

second one solved Equation (9), applying explicit diffusion in the y-direction and adding the remaining half of the reaction term. As with the first kernel, one thread per grid point was launched, ensuring full parallelism across the spatial domain.

Finally, the third kernel solved batches of independent tridiagonal linear systems that arose from the implicit discretization of the diffusion terms. They were handled using the Thomas algorithm and were implemented in two directional variants. The first one solved N_x independent systems of size N_y, corresponding to Equation (6). Conversely, the other one solved N_y systems of size N_x, as described in Equation (7). Here, N_x and N_y represent the number of grid points in the horizontal and vertical directions, respectively. By using direction-specific kernels, transposing data between steps was not necessary, thus reducing global memory operations.

2.4. Computational experiments

We conducted two computational experiments to evaluate accuracy, stability, and performance. The first focused on temporal accuracy. A 2D rectangular domain (1 cm × 0.01 cm) was initialized with a planar excitation along the left edge (0.2 cm × 0.01 cm, 2 ms duration), inducing a wave propagating rightward. The total simulation time was 36 ms. To minimize spatial errors, we used a fine spatial resolution of 1 μm and varied the time step from 0.016 ms to 0.256 ms. Since no analytical solution is available, a high-resolution simulation using SSI-ADI with $\Delta t = 0.0001$ ms served as the reference. To ensure consistent initial conditions, a 10 ms pre-simulation was performed with the same configuration as the reference, and its resulting wavefront was shifted 0.7 cm left before starting the new simulations for another 26 ms. Accuracy was assessed via the L2-norm of the error.

The second experiment was designed to replicate arrhythmogenic activity and to evaluate execution time and parallel speedup. A square domain (6 cm × 6 cm) was simulated with 100 μm spatial resolution and a fixed time step of 0.01 ms for 500 ms. An S1–S2 stimulation protocol induced a spiral wave: S1 was applied at the left boundary (0.2 cm width), and S2 was delivered at 340 ms over a 3 cm × 3 cm region in the lower-left quadrant, both with a duration of 2 ms. All methods (FE, OS-ADI, SSI-ADI) were executed using both OpenMP (6 threads) and CUDA. Performance was measured on a workstation with an NVIDIA RTX 4070 Ti Super GPU and an Intel Core i5-13400F CPU.

3. Results and discussions

The first set of experiments aimed to evaluate the numerical stability and temporal accuracy of the tested schemes. As expected, FE method failed to complete any of the simulations, consistently diverging regardless of the chosen time step. This outcome aligns with the well-known limitation of explicit schemes imposed by the Courant–Friedrichs–Lewy (CFL) condition, reinforcing the method's unsuitability for stiff problems like those arising in cardiac electrophysiology.

In contrast, both implicit approaches demonstrated excellent stability across all tested time steps, including relatively coarse values up to 0.256 ms. However, their accuracy levels diverged significantly. At every temporal resolution, the SSI-ADI method consistently yielded lower errors when compared to OS-ADI.

Figure 1 presents a log-log plot of the L2 error norm as a function of time step for both OS-ADI (in red) and SSI-ADI (in blue). The dashed lines correspond to linear regressions obtained via least squares fitting, which allowed estimation of the empirical convergence rate in time. The SSI-ADI method exhibited a convergence slope of approximately 2.10, matching the expected second-order behavior. In contrast, the OS-ADI method showed a slope near 1.19, consistent with its first-order temporal accuracy.

An important practical implication of this difference is the significantly larger time steps that SSI-ADI can employ to achieve the same level of accuracy. For instance, to obtain an error of the same order as that produced by OS-ADI with a time step of 0.016 ms, SSI-ADI required only a time step of 0.128 ms, making the simulation approximately eight times faster. This advantage underscores the potential of SSI-ADI for efficient and accurate simulations, as it achieves a given error threshold with substantially lower computational effort, thereby offsetting its per-step cost.

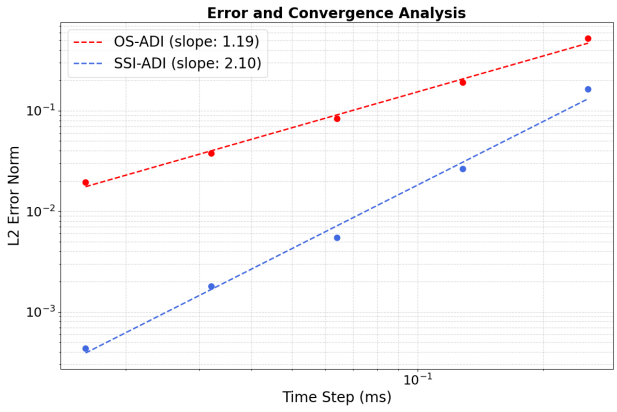

Figure 1. Temporal convergence analysis for the OS-ADI (red) and SSI-ADI (blue) methods. The plot shows the L2 error norm as a function of the time step size. Dashed lines represent least-squares fits. The SSI-ADI method exhibits a slope of 2.10, while the OS-ADI, a slope of 1.19.

To evaluate the computational performance of the nu-

merical schemes in simulating arrhythmias, we measured execution times for the FE, OS-ADI, and SSI-ADI methods using two parallelization strategies: OpenMP (with 6 threads) and CUDA. As in this experiment, the space and temporal steps were close to each other in magnitude, the CFL condition was not a problem for FE. As expected, the FE method was the fastest due to its simplicity, followed by the OS-ADI and SSI-ADI methods, the latter being the most computationally intensive due to its additional steps.

Table 1 summarizes the execution times and GPU speedups achieved by each method when transitioning from CPU (using OpenMP) to GPU (using CUDA). Among all schemes, SSI-ADI stood out with the highest acceleration, achieving a 12.48x speedup and reducing its runtime from 747 s to just 59.88 s. This impressive gain is largely attributed to the method's structure: in the first stage, each thread handles all computations for a single grid point, enabling full exploitation of spatial parallelism. That underscores not only the numerical advantages of SSI-ADI but also its exceptional scalability when deployed on GPU architectures.

These results confirmed that — by integrating CN discretization for diffusion, Runge–Kutta approximation for reaction, and directional splitting — the SSI-ADI method established a powerful and scalable framework. It offered a compelling combination of accuracy, stability, and computational robustness.

Table 1. Execution times (in seconds) for OpenMP (6 threads) and CUDA, and GPU speedup.

Method	OpenMP	CUDA	GPU Speedup
FE	155.82	20.223	7.71
OS	231.32	40.317	5.74
SSI	747.00	59.88	12.48

4. Conclusion

This study highlighted the untapped potential of the SSI-ADI method as a powerful alternative for simulating cardiac electrophysiology. While it involves a higher per-step computational cost, its second-order temporal convergence and robust stability enable significantly larger time steps without compromising accuracy. As a result, SSI-ADI can achieve a given error level much faster than first-order schemes such as OS-ADI, effectively reducing total simulation time. When combined with GPU parallelization, the SSI-ADI becomes even more efficient, achieving a 12.48x speedup over its already optimized OpenMP version. More than just a numerically elegant scheme, SSI-ADI proves to be a practical and scalable solution for high-fidelity cardiac simulations. These results establish it as a compelling tool for researchers seeking both precision and performance in the modeling of complex heart dynamics.

Future work involves extending SSI-ADI to anisotropic conductivity, realistic anatomies, and large-scale 3D simulations. In this context, adapting the scheme to additional directional sweeps might be guided by the Douglas–Gunn [8] framework. Further analysis will also address heterogeneous conditions and explore adaptive strategies for improved efficiency, reinforcing SSI-ADI as a versatile tool in computational cardiology.

Acknowledgments

The authors acknowledge support from the Wellcome Trust (214290/Z/18/Z), EPSRC via the CompBioMedX project (EP/X019446/1), CompBioMed2 (GA 675451, 823712), FAPEMIG (PCE-00048-25; APQ-02752-24; APQ-02445-24; APQ-02513-22), FINEP (SOS Equipamentos AV020062/22), SINAPAD Santos-Dumont, CAPES, EBSERH, and UFJF.

References

[1] Niederer SA, Lumens J, Trayanova NA. Computational models in cardiology. Nature Reviews Cardiology 2019; 16(2):100–111.

[2] Niederer SA, Sacks MS, Girolami M, Willcox K. Scaling digital twins from the artisanal to the industrial. Nature Computational Science 2021;1(5):313–320.

[3] Couto GM, Monteiro NZ, dos Santos RW. Accelerating simulations of cardiac arrhythmias through robust numerical techniques and parallel computing. In Simpósio Brasileiro de Computação Aplicada à Saúde (SBCAS). SBC, 2023; 72–77.

[4] Monteiro NZ, Pereira RR, Rocha BM, dos Santos RW, Mazorche SR, Loula AFD. A novel second-order adi scheme for solving epidemic models with cross-diffusion. Journal of Computational Science 2024;81:102341.

[5] Bueno-Orovio A, Cherry EM, Fenton FH. Minimal model for human ventricular action potentials in tissue. Journal of Theoretical Biology 2008;253(3):544–560.

[6] Strikwerda JC. Finite difference schemes and partial differential equations. SIAM, 2004.

[7] Gomes JM, Oliveira RS, Lobosco M, dos Santos RW. Adaptive-step methods for markov-based membrane models. Communications in Nonlinear Science and Numerical Simulation 2020;85:105249.

[8] Douglas Jr J, Gunn JE. A general formulation of alternating direction methods: Part i. parabolic and hyperbolic problems. Numerische Mathematik 1964;6(1):428–453.

Address for correspondence:

Guilherme Martins Couto

Programa de Pós-Graduação em Modelagem Computacional (PPGMC-UFJF) - Juiz de Fora - MG, Brasil, 36036-330

guilherme.couto@estudante.ufjf.br