An efficient non-standard finite difference scheme for an ionic model of cardiac action potentials

Raymond J. Spiteri[1,*], **and Mary C. MacLachlan**[**]

[*] *Faculty of Computer Science, Dalhousie University, Halifax, Nova Scotia, Canada, B3H 1W5.*
[**] *School of Biomedical Engineering, Dalhousie University, Halifax, Nova Scotia, B3H 3J5.*

Dedicated to Professor Ronald E. Mickens on the occasion of his 60th birthday.

Abstract

The ionic movements that generate the electrical activity in a human myocardial cell can be modelled mathematically by sets of ordinary differential equations (ODEs). The computational efficiency in solving these sets of ODEs is of paramount importance because their solution must be performed many hundreds of thousands of times over in realistic three-dimensional models of macroscopic regions of the heart such as the atria or the ventricles. In this paper, we propose a nonstandard finite difference scheme for the solution of the Luo-Rudy model for ionic currents in cardiac tissue. We demonstrate that the nonstandard finite difference scheme can be as much as 25 times more efficient than standard forward Euler while maintaining an acceptable level of accuracy.

1 Introduction

The electrical activity responsible for every beat of the heart can be modeled mathematically in a physiologically accurate way by means of ordinary differential equations (ODEs). It is the goal of this paper to describe one such ODE model due to Luo and Rudy [12] and to investigate efficient numerical methods for its solution. Specifically we are interested in the Luo-Rudy Phase I model, which was the most physiologically accurate model of its type in early 1990s. Although it has since been surpassed by other models for cardiac action potentials such as the Luo-Rudy Phase II model [13], the model due to Courtemanche et al. [5], and the model due to Nygren et al. [16], important features relevant to a numerical solution are still captured well with this model. These models all share similar characteristics and try to be more physiologically accurate by accounting for more and more ionic currents. In fact the Luo-Rudy Phase I model itself can be viewed as a more realistic generalization of the simpler and more well-known FitzHugh-Nagumo model [6, 15]. In all cases, the increasingly accurate physiology comes at the cost of greatly increasing the complexity of the underlying mathematical model.

The rate of beating of a heart is normally controlled by the sinoatrial node, the heart's natural pacemaker. It produces an electrical signal, which spreads throughout the atria and ventricles, controlling contraction. When other areas of the atria besides the sinoatrial node initiate contraction, the result is a type of arrythmia known as *atrial fibrillation* (AF). In other words, there are a number of *ectopic sites* that send out uncoordinated signals, causing the atria to pump too fast and not fully contract. As a

[1]Corresponding author, e-mail: spiteri@cs.dal.ca

result, the atria are unable to empty properly and blood pools in them forming clots. If a clot dislodges from the left atrium and enters the blood stream, it may lead to stroke [8].

One treatment option for AF is catheter ablation [8], whereby radio-frequency energy is used to destroy the cells that were firing irregularly. A computational model, which is both anatomically and electrophysiologically correct, can be of use in both investigating the mechanisms of AF as well as in aiding treatment through the non-invasive location of ectopic sites.

Designing a computational model of the heart is a challenging task. There are many important factors to consider that critically influence the results. These factors include the choice of propagation model, geometry, and cell model. When modeling physiological processes, it is necessary to make some simplifying assumptions in order to make the computations feasible. However, excessive simplification renders the model useless in clinical investigations, such as those needed for AF.

In this paper we focus only on the numerical solution of the cell model. Specifically we are interested in accurate and efficient numerical solutions of the cell model known as the Luo-Rudy Phase I model. Efficiency of the numerical scheme is of paramount importance because the solution of this model for one "cell" must typically be performed many times in a realistic three-dimensional macroscopic region of the heart such as the atria or the ventricles: essentially the cell model must be solved in each of the many hundreds of thousands of clusters of cells; see e.g., [11].

There are a total of eight ODEs that make up the Luo-Rudy Phase I model. Six of these describe the dynamics of the *voltage-gated channels* that control the permeability of each myocardial cell membrane to different ions. They are Hodgkin-Huxley type gating equations specific to cardiac cells and are *linear* ODEs. Precise relationships are given in Section 2 and in the Appendix.

The complete Luo-Rudy model is quite intricate. This in itself poses a challenge to its efficient numerical solution. However, the ODEs governing the gating variables are linear. We take advantage of the linearity of these ODEs by making use of their known exact solution as part of a non-standard finite difference (NSFD) scheme [14]. We then compare the accuracy and efficiency of such an approach with conventional numerical methods for the solution of initial-value problems in ODEs. Another challenge to the efficient solution of the Luo-Rudy equations comes from the fact that they turn out to be *stiff* for common explicit numerical methods and the accuracy levels relevant to this application. Stiffness is an important yet often misunderstood concept in the numerical solution of differential equations. Many different definitions appear in the literature. For a nice discussion of the various interpretations of stiffness, we refer to [10]. In this paper, we define a problem to be stiff if the time step used by an explicit numerical method such as forward Euler [10, 1] is based on considerations of stability and not accuracy. In other words, the time step is generally much smaller than necessary to meet the user's accuracy requirements, but a larger stepsize would cause the method to become unstable. An implication of this definition of stiffness is that no problem is stiff if the user-supplied error tolerance is small enough because in that case a small stepsize is necessitated by the user's accuracy requirements. This plays an important role in our discussion in Section 3.

The accepted theory in choosing the most efficient numerical method to solve a given problem is to first try an explicit method, such as forward Euler or other explicit Runge-Kutta or linear multistep method. If the problem has sufficient smoothness, then higher-order methods are generally more efficient. If the problem is stiff then explicit methods are no longer the methods of choice because (by definition) their stepsizes are excessively small. In this case implicit methods are deemed to be the most efficient methods. The degree of implicitness varies from linearly implicit to fully implicit. Recently there has been much interest in deriving *implicit-explicit methods* (see e.g., [2, 3, 19] and the references therein) that treat different parts of the right-hand side of the ODE differently, perhaps according to stiffness. For example, nonstiff nonlinear terms can be conveniently handled explicitly whereas stiff linear terms can be most effectively handled implicitly. Other variations on this theme exist, and we are particularly interested in NSFD schemes. Such schemes are often cast as linearly implicit schemes where the linear system is inverted explicitly at each step, e.g., in a Gauss-Seidel-type manner, effectively making them implementable as explicit schemes.

Perhaps a less well-appreciated fact is that *stepsize control* forms a critical part in the efficiency of a numerical method. Stepsize controllers are designed to take the largest step possible while maintaining the level of the local error estimate does not exceed that requested by the user. Although the implications on accuracy are much more recognized, the implications on efficiency are equally important. In most cases, the added cost of estimating and controlling the local error at every step of the integration is more than offset by the savings from being able to use larger stepsizes. The most notable exception to this is the situation when for example steady-state or other long-term or qualitative properties are desired from the numerical solution, and accuracy in transients is less important. For clinical purposes, it is important to resolve all phases of the cardiac action potential, and so we will not exclude variable-stepsize methods in our investigations.

The remainder of this paper is organized as follows. In Section 2 we give a detailed description of the Luo-Rudy Phase I model for cardiac action potential. In Section 3 we investigate the numerical solution of this model. In particular, we propose a NSFD scheme and compare its accuracy and efficiency to some standard explicit numerical schemes. Finally, in Section 4, we summarize our findings.

2 The Luo-Rudy Phase I Model for the Cardiac Action Potential

The Luo-Rudy Phase I model [12] for the cardiac action potential accounts for what are believed to be the most important ionic currents responsible for cardiac action potentials. More advanced models such as [13, 5, 16] attempt to be more realistic physiologically by incorporating variable ionic concentrations and the contribution of pump currents. In the Luo-Rudy model, voltage-gated channels control the conductance of sodium, potassium, and calcium ions in cardiac muscle cells. The Luo-Rudy model describes the operation of these channels with ODEs that are Hodgkin-Huxley type gating equations specific to cardiac cells. The Luo-Rudy model describes the resulting ionic currents as products of channel conductance and the difference between membrane potential and the reversal potential for a particular ionic species. For an ionic species s, the current I_s is described as follows:

$$I_s = G_s \cdot (V_m - E_s),$$

where G_s is the channel conductance for ionic species s, V_m is the membrane potential, and E_s is the reversal potential for the species as calculated by the Nernst equation[1]. The conductance of the channel changes with changes in the membrane potential, hence the name *voltage-gated channels*. The Luo-Rudy model describes the channel conductance as a function of the maximum channel conductance and activation and inactivation parameters:

$$G_s = \bar{G}_s \times [\text{activation/inactivation parameter(s)}],$$

where $\bar{G}_s$ is the maximum conductance of the channel.

These activation/inactivation parameters control the opening and closing of the channel and thus the amount of ions crossing the cell membrane. The activity of these parameters y is described mathematically by ordinary differential equations of the form:

$$\frac{dy}{dt} = \frac{y_\infty - y}{\tau_y}, \tag{1}$$

[1]The Nernst equation for the reversal potential of ionic species s is

$$E_s = \frac{RT}{z_s F} \ln \frac{[s]_e}{[s]_i},$$

where R is the gas constant, T is temperature, z_s is the valence of ion s, F is Faraday's constant, $[s]_e$ is the extracellular concentration of s, and $[s]_i$ is the intracellular concentration of s.

where y_∞ is the steady-state value of the gate's activation/inactivation parameter, and τ_y is the time constant of the gate. Both y_∞ and τ_y are functions of rate parameters α_y and β_y, both of which are voltage dependent:

$$y_\infty = \frac{\alpha_y}{\alpha_y + \beta_y}, \qquad \tau_y = \frac{1}{\alpha_y + \beta_y}. \tag{2}$$

There are six ionic currents in the Luo-Rudy model: a fast sodium current I_{Na}; a slow inward calcium current I_{si}; a time-dependent potassium current I_K; a time-independent potassium current I_{K1}; a plateau potassium current I_{K_p}; and a background potassium current I_b. Of these six, three are directly dependent on gating parameters. There are six gating parameters in total, denoted m, h, j, d, f, X. Parameters m, h, and j are for the fast sodium current, parameters d and f are for the slow inward calcium current, and parameter X is for the time-dependent potassium current. Each gating parameter is described by a linear ODE of the form (1). These six ODEs for gating parameters are coupled with an ODE for the calcium concentration in the cell and an ODE for the membrane voltage, making eight ODEs in total. The calcium concentration satisfies

$$\frac{d([Ca]_i)}{dt} = -10^{-4} I_{si} + 0.07(10^{-4} - [Ca]_i), \tag{3}$$

where $[Ca]_i$ is the intracellular calcium concentration and I_{si} is the slow inward calcium current. The membrane voltage is the result of the total transmembrane current. An electrical schematic of the membrane is shown in Figure 1.

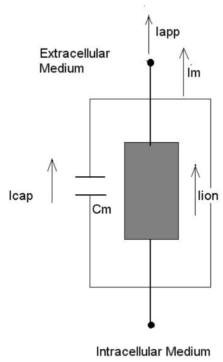

Figure 1: Circuit representation of the cell membrane.

The three contributing currents are I_{ion}, the sum of the six ionic currents previously mentioned, I_{app}, the stimulus current which triggers the action potential, and I_{cap}, the capacitative current. This capacitative current is due to the separation of charges by the membrane. The membrane is impermeable to ions so it acts as an insulator, or dielectric, separating charged ions. When a voltage is applied across the membrane, the negative ions will gather close to the membrane on the positive side (extracellular side) and the positive ions will gather near the membrane on the negative (intracellular) side [18]. The membrane is thin enough so that the ions can interact electrostatically across it to produce the capacitative current,

$$I_{cap} = C_m \frac{dV_m}{dt}, \tag{4}$$

where C_m is the membrane capacitance. The total transmembrane current I_m is given by the equation,

$$I_m = I_{cap} + I_{ion}, \tag{5}$$

where I_{ion} is the sum of the ionic currents. Noting the fact that $I_m = I_{app}$, substituting from (4) for I_{cap} in (5), and solving for $\frac{dV_m}{dt}$ gives the ODE for membrane voltage,

$$\frac{dV_m}{dt} = -\frac{1}{C_m} \times (I_{ion} - I_{app}). \tag{6}$$

The complete Luo-Rudy Phase I model with all the parameter values used for the simulations in this paper can be found in the Appendix.

3 Numerical Methods and Results

In this section we discuss the various numerical methods used to solve the Luo-Rudy ODEs (3), (6)–(A.21). After generating an accurate reference solution, the efficiency of the numerical methods is determined by measuring errors with respect to the reference solution. The reference solution was generated using Matlab's `ode15s`, which is a variable-stepsize stiff solver based on a family of linear multistep methods, with both absolute and relative tolerances (`atol` and `rtol`, respectively) set to 10^{-10}. The ODEs were solved over a timespan of 450ms. This length of time is an accepted value for the action potential duration [4]. The initial conditions are assigned as follows: $V_m(0) = V_{rest}$, $y(0) = y_\infty$ for each gating variable, and $[Ca]_i(0) = 2.0 \times 10^{-4}$. Finally, the following stimulus current is applied in order to initiate an action potential:

$$I_{app} = \begin{cases} 60\ \mu\text{A/cm}^2 & 0 \le t \le 0.5\ \text{ms}, \\ 0 & t > 0.5\ \text{ms}. \end{cases}$$

The most important variable in a model of the cardiac action potential is the membrane voltage V_m. Arrythmias and other irregularities are often detected through analysis of the V_m waveform. Accordingly, as a measure of the error for the membrane voltage with respect to the reference solution $\hat{V_m}$ in a numerical solution V_m, we use the following relative root-mean-square (RRMS) error norm [9]:

$$RRMS = \sqrt{\frac{\sum_{i=1}^{N}(V_i - \hat{V}_i)^2}{\sum_{i=1}^{N}\hat{V}_i^2}},$$

where $V_i = V_m(t_i)$ and $\hat{V}_i = \hat{V}_m(t_i)$ are the numerical solutions produced at time t_i, $t_i \in [0, 450]$. With the many other approximations made in the modelling process, a RRMS error of 5% can be deemed to be of sufficient accuracy for researchers in biomedical engineering. A plot of the reference solution for V_m is given in Figure 2.

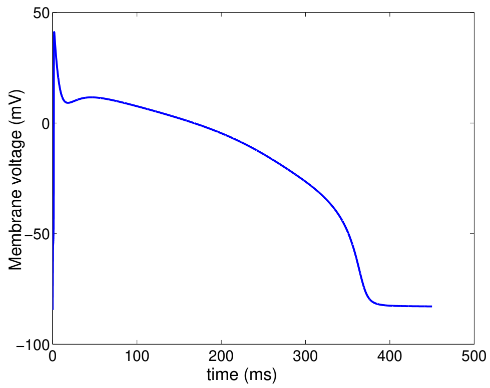

Figure 2: Reference solution generated for the membrane voltage V_m.

It is generally impossible to determine *a priori* or by inspection whether a set of nonlinear ODEs is stiff for a given (explicit) numerical method and error tolerance. So the natural first attempt to solve the ODE is by means of a high-order, variable-stepsize, explicit numerical method, such as Matlab's `ode45`. This routine is based on the well-known Dormand-Prince 5(4) embedded explicit Runge-Kutta pair [7]. If this routine produces a result and the solution is relatively smooth, it is not likely that significant efficiency gains would be easily realized by trying other methods.

However, when run with the default tolerances (`atol`$= 10^{-6}$, `rtol`$= 10^{-3}$), `ode45` immediately became unstable. This behaviour can be attributed to the failure of the local error control mechanism in selecting a suitable stable step size. This is an indication of problem stiffness with respect to the numerical method and user tolerance. This problem can be mitigated by decreasing the tolerance because the problem would then no longer be stiff. As expected, `ode45` executed normally with `atol`=`rtol`$= 10^{-10}$.

A similar experiment was performed using the software package `odeToJava` [17], which is an initial-value problem solver for ODEs written in the Java programming language. Specifically, the two tests just described were run with the `DormandPrince` class, which is essentially equivalent to `ode45`. Not surprisingly the same results were obtained; i.e., the method became unstable for the default tolerances but executed normally for the more stringent tolerances. However, the `DormandPrince` class also supports a built-in stiffness detection test as in `DOPRI5` [7]. It is noteworthy that when the tolerances were lowered slightly to `atol`=`rtol`$= 10^{-6}$, the stiffness detection test indicated that the problem became stiff approximately at time 371 ms. This is reasonable because by this time V_m has essentially returned to its resting voltage and the code would like to take larger steps; the fact that it cannot without becoming unstable is an indication of stiffness. So-called *type-insensitive codes* use information such as this to automatically switch integration methods from non-stiff to stiff solvers without intervention from the user.

Given the adverse effect of stiffness on variable-stepsize methods based on embedded Runge-Kutta methods, one is generally left with two ways to proceed. The first is to switch to a constant-step method. In this case, the hope is that the stepsize is not so restricted by stiffness as to render the overall computation infeasible. The second is to switch to an implicit method. In particular, we choose NSFD methods as the implicit methods in order to take advantage of the exact solution to the linear ODEs (1) for the gating variables. In fact, we investigate both possibilities, mainly in order to compare their relative efficiency.

We first implemented a constant-step forward Euler method. To recall, for the ODE

$$\frac{dx}{dt} = f(t,x),$$

the forward Euler scheme for advancing the solution from $x_{n-1} \approx x(t_{n-1})$ at time $t = t_{n-1}$ to $x_n \approx x(t_n)$ at time $t = t_n = t_{n-1} + \Delta t$ is given by

$$x_n = x_{n-1} + \Delta t f(t_{n-1}, x_{n-1}).$$

We found that the largest stable stepsize allowed by this method was $\Delta t = 0.01$ms and resulted in a RRMS error of 2.75%. A plot of this result appears in Figure 3. In the left panel, we see that the forward Euler solution and the reference solution are coincident to resolution plotted. For clarity, in the right panel we show the two different solutions over the first 10ms, in which the solutions differ most.

It is well known that higher-order explicit Runge-Kutta methods have larger regions of absolute stability than that of forward Euler; see e.g., [7, 1]. However, this does not necessarily translate into more efficient methods because the higher order comes at a higher cost of more function evaluations per step. We implemented a constant-stepsize classical (fourth-order) explicit Runge-Kutta method (`erk4`) [7] as a pre-defined class in `odeToJava`. We found that the largest stable stepsize allowed by this method was $\Delta t = 0.018$ms and resulted in a RRMS error of 0.54%. We can make the following two observations from these results. First, `erk4` can take a larger stepsize than forward Euler; however, the tradeoff in terms of efficiency is actually detrimental because the 300% increase in cost per step from using `erk4` only returns

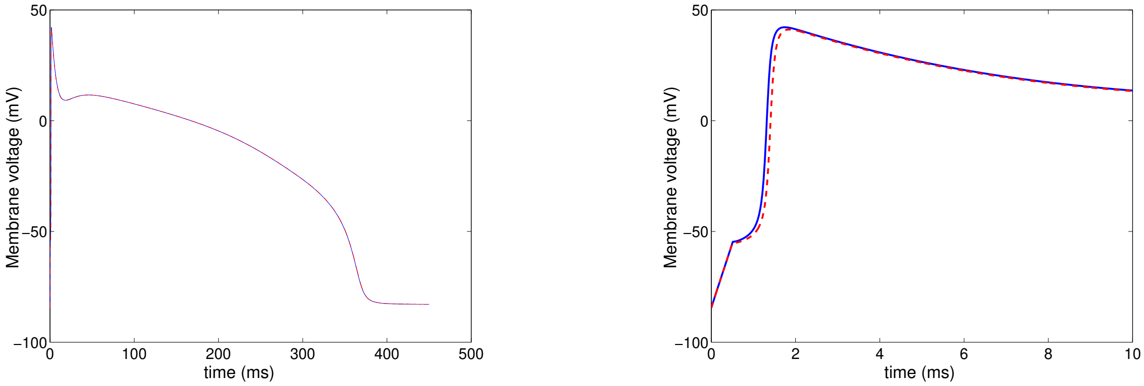

Figure 3: Forward Euler solution for the membrane voltage V_m with $\Delta t = 0.01$ms.

an 80% increase in stable step size. Second, the accuracy of `erk4` is significantly higher than that of forward Euler. Unfortunately, in this case this tradeoff is also detrimental because the accuracy provided is *well below* the 5% RRMS error requested by the user. In other words, the computation is not as efficient as it could be because efficiency is sacrificed for unnecessary accuracy. Hence `erk4` is actually a worse choice of method given the goals of this application.

Because of our implementation in `odeToJava`, we were able to conveniently test a more expensive but more robust error control mechanism: `erk4` can be run with error control by means of *step doubling* [1]. This form of error control is suitable for schemes that do not have a companion scheme (e.g., in embedded explicit Runge-Kutta methods or predictor-corrector methods). Indeed with the default tolerances, this form of error control managed to get approximately to time $t = 354$ms before becoming unstable. Also as expected, when the tolerances were lowered, `erk4` with step-doubling error control executed normally.

Thus far, the most efficient method for the solution of the Luo-Rudy equations is seen to be the forward Euler method. We now describe a NSFD method with superior stability properties that ultimately lead to a more efficient method. It is in fact a formally implicit method because use is made of some of the solution components at end of the time step; however, it can be implemented in an explicit fashion. This avoids the common difficulty in implementing implicit methods, namely the costly nonlinear system solution at each step. The NSFD will have a comparable cost per step as forward Euler; hence a direct comparison based on stepsize can be made.

The NSFD we propose makes use of the exact solution of the linear ODEs (1) for the six gating parameters. The exact solution used to advance any gating variable y from time $t = t_{n-1}$ to time $t = t_n = t_{n-1} + \Delta t$ is

$$y_n = y_\infty + (y_{n-1} - y_\infty)e^{\left(-\frac{\Delta t}{\tau_y}\right)}, \tag{7}$$

where y_∞ and τ_y are as previously defined. Because y_∞ and τ_y depend on the membrane potential V_m, they can be evaluated explicitly using the value of V_m from the previous time step. This leaves the ODEs for membrane potential (V_m) and calcium concentration ($[Ca]_i$). These are solved numerically by forward Euler. This NSFD method using an exact solution for the six gating variables and forward Euler for V_m and $[Ca]_i$ was coded in Matlab. The result is a system that is no longer stiff; a larger step size can now be taken (i.e., one that is based only on accuracy requirements and not stability requirements), and efficiency is dramatically increased over forward Euler and other explicit methods. The largest step size that can be taken, while still retaining the required accuracy of 5% RRMS error, is $\Delta t = 0.25$ with RRMS=4.82%; see Table 1. This value of Δt is twenty-five times greater than the largest step size possible with the forward Euler method described earlier. A plot of this result appears in Figure 4. In the left panel, we see that the NSFD solution and the reference solution are again coincident to resolution plotted. Again in the right panel we show the two different solutions over the first 10ms. Here we see that the error is indeed larger for the NSFD method compared to forward Euler, but it is in fact the most efficient method to meet the user's specified error tolerance. We note that this approach was also used by Courtemanche et al. [5] on their atrial action potential ODEs, but the motivation for why such a scheme was proposed is not given. Modifying the ODE system in this manner is a much easier way to handle stiffness than coding a fully implicit method.

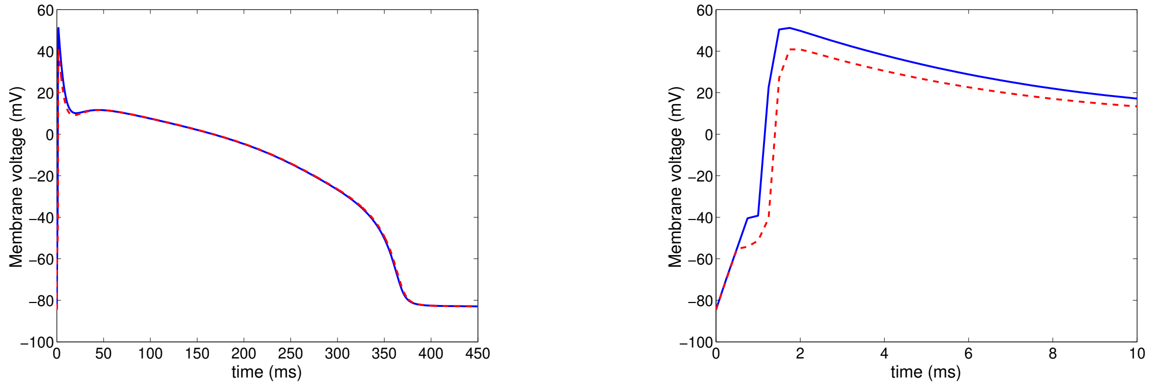

Figure 4: NSFD solution for the membrane voltage V_m with $\Delta t = 0.25$ms.

Table 1: RRMS values for the NSFD scheme for some step sizes Δt.

Δt	RRMS
0.005	0.0070
0.01	0.0129
0.05	0.0471
0.10	0.0490
0.15	0.0452
0.20	0.0359
0.21	0.0382
0.22	0.0566
0.23	0.0796
0.24	0.0423
0.25	0.0482
0.26	0.0858
0.27	0.0721
0.28	0.0563
0.29	0.0675
0.30	0.0656

4 Conclusion

In this paper we performed a thorough analysis of numerical methods for the solution of the Luo-Rudy ODEs describing the cardiac action potential. We are interested in finding the most efficient numerical method because inefficiencies will be magnified hundreds of thousands of times over as these equations are solved in a realistic three-dimensional domain. We find that these equations are stiff for the moderate tolerances required by researchers in biomedical engineering. The stiffness of the equations critically affects the optimal choice of numerical method. We find that standard explicit methods are ultimately not suitable: constant-step methods must take a step that is excessively small to maintain stability, and variable-step methods become unstable at moderate tolerances. That is, both approaches are not efficient because ultimately stability considerations force them to provide *too much accuracy*. A NSFD method that takes advantage of the linear nature of some of the Luo-Rudy equations is shown to be up to 25 times more efficient than the most efficient explicit method studied. Because the solutions to the Luo-Rudy equations are relatively smooth, it can be anticipated that higher-order linearly implicit IMEXRK methods with variable stepsizes may be more efficient than the proposed NSFD scheme. Indeed, preliminary studies show that the four-stage, third-order IMEXRK scheme $(4,4,3)$ [2] is approximately 7 times more efficient than the proposed NSFD scheme with constant stepsize. We report on these findings elsewhere.

Appendix

The following is the complete Luo-Rudy model with all the parameters used in this paper.

Inward currents

Fast sodium current

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

Activation gate, m

$$\begin{aligned} \frac{dm}{dt} &= \alpha_m(1-m) - \beta_m m && \text{(A.2a)} \\ \alpha_m &= \frac{0.32(V_m+47.13)}{1-e^{-0.1(V_m+47.13)}} && \text{(A.2b)} \\ \beta_m &= 0.08e^{-V_m/11} && \text{(A.2c)} \end{aligned}$$

Fast inactivation gate, h

$$\begin{aligned} \frac{dh}{dt} &= \alpha_h(1-h) - \beta_h h && \text{(A.3a)} \\ \alpha_h &= \begin{cases} 0.135e^{(V_m+80)/-6.8} & V_m < -40\mathrm{mV} \\ 0 & V_m \geq -40\mathrm{mV} \end{cases} && \text{(A.3b)} \\ \beta_h &= \begin{cases} 3.56e^{0.079V_m} + 3.1\cdot 10^5 e^{0.35V_m} & V_m < -40\mathrm{mV} \\ \dfrac{1}{0.13(1+e^{(V_m+10.66)/-11.1})} & V_m \geq -40\mathrm{mV} \end{cases} && \text{(A.3c)} \end{aligned}$$

Slow inactivation gate, j

$$\begin{aligned} \frac{dj}{dt} &= \alpha_j(1-j) - \beta_j j && \text{(A.4a)} \\ \alpha_j &= \begin{cases} \dfrac{-1.2714\cdot 10^5 e^{0.2444V_m} - 3.474\cdot 10^{-5}e^{-0.04391V_m}\cdot (V_m+37.78)}{1+e^{0.311(V_m+79.23)}} & V_m < -40\mathrm{mV} \\ 0 & V_m \geq -40\mathrm{mV} \end{cases} && \text{(A.4b)} \\ \beta_j &= \begin{cases} \dfrac{0.1212e^{-0.01052V_m}}{1+e^{-0.1378(V_m+40.14)}} & V_m < -40\mathrm{mV} \\ \dfrac{0.3e^{-2.535\cdot 10^{-7}V_m}}{1+e^{-0.1(V_m+32)}} & V_m \geq -40\mathrm{mV} \end{cases} && \text{(A.4c)} \end{aligned}$$

Slow inward current

$$\begin{aligned} I_{\mathrm{si}} &= \overline{G}_{\mathrm{si}} \cdot d \cdot f \cdot (V_m - E_{\mathrm{si}}) && \text{(A.5)} \\ E_{\mathrm{si}} &= 7.7 - 13.0287 \cdot \ln\left([\mathrm{Ca}]_i\right) && \text{(A.6)} \end{aligned}$$

Activation gate, d

$$\begin{aligned} \frac{dd}{dt} &= \alpha_d(1-d) - \beta_d d && \text{(A.7a)} \\ \alpha_d &= \frac{0.095e^{-0.01(V_m-5)}}{1+e^{-0.072(V_m-5)}} && \text{(A.7b)} \\ \beta_d &= \frac{0.07e^{-0.017(V_m+44)}}{1+e^{0.05(V_m+44)}} && \text{(A.7c)} \end{aligned}$$

Inactivation gate, f

$$\begin{aligned} \frac{df}{dt} &= \alpha_f(1-f) - \beta_f f && \text{(A.8a)} \\ \alpha_f &= \frac{0.012e^{-0.008(V_m+28)}}{1+e^{0.15(V_m+28)}} && \text{(A.8b)} \\ \beta_f &= \frac{0.0065e^{-0.02(V_m+30)}}{1+e^{-0.2(V_m+30)}} && \text{(A.8c)} \end{aligned}$$

Calcium uptake

$$\frac{d\left([\mathrm{Ca}]_i\right)}{dt} = -10^{-4} I_{\mathrm{si}} + 0.07(10^{-4} - [\mathrm{Ca}]_i) \qquad \text{(A.9)}$$

Outward Currents
Time-dependent potassium current

$$\begin{aligned} I_{\mathrm{K}} &= \overline{G}_{\mathrm{K}} \cdot X \cdot X_{\mathrm{i}} \cdot (V_m - E_{\mathrm{K}}) && \text{(A.10)} \\ \overline{G}_{\mathrm{K}} &= 0.282 \cdot \sqrt{[\mathrm{K}]_o/5.4} && \text{(A.11)} \end{aligned}$$

Activation gate, X

$$\begin{aligned} \frac{dX}{dt} &= \alpha_X(1-X) - \beta_X X && \text{(A.12a)} \\ \alpha_X &= \frac{0.0005e^{0.083(V_m+50)}}{1+e^{0.057(V_m+50)}} && \text{(A.12b)} \\ \beta_X &= \frac{0.0013e^{-0.06(V_m+20)}}{1+e^{-0.04(V_m+20)}} && \text{(A.12c)} \end{aligned}$$

Inactivation gate, X_{i}

$$X_{\mathrm{i}} = \begin{cases} \dfrac{2.837(e^{0.04(V_m+77)}-1)}{(V_m+77)e^{0.04(V_m+35)}} & V_m > -100\mathrm{mV} \\ 1 & V_m \le -100\mathrm{mV} \end{cases} \qquad \text{(A.13)}$$

Time-independent potassium current

$$\begin{aligned} I_{\mathrm{K1}} &= \overline{G}_{\mathrm{K1}} \cdot \mathrm{K1}_\infty \cdot (V_m - E_{\mathrm{K1}}) && \text{(A.14)} \\ \overline{G}_{\mathrm{K1}} &= 0.6047 \cdot \sqrt{[\mathrm{K}]_o/5.4} && \text{(A.15)} \end{aligned}$$

Inactivation gate, K1

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

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

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

Plateau potassium current

$$I_{\mathrm{Kp}} = \overline{G}_{\mathrm{KP}} \cdot \mathrm{Kp} \cdot (V_m - E_{\mathrm{Kp}}) \tag{A.17}$$

$$E_{\mathrm{Kp}} = E_{\mathrm{K1}} \tag{A.18}$$

$$\mathrm{Kp} = \frac{1}{1 + e^{(7.488 - V_m)/5.98}} \tag{A.19}$$

Background potassium current

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

Total ionic current

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

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

Table 2: Parameters for the Luo-Rudy Phase I model; the conductances are in mS/cm^2 and the reversal potentials in mV [4].

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

Acknowledgement

The authors wish to thank M. Patterson for his help with `odeToJava`.

References

[1] U.M. Ascher and L.R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, Society for Industrial and Applied Mathematics, Philadelphia, 1998.

[2] U.M. Ascher, S.J. Ruuth, and R.J. Spiteri, *Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations*, Appl. Numer. Math., Vol. 25, pp. 151–167, 1997.

[3] 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.

[4] C.J. Clements, *Nonlinear Wave Propagation in an Anisotropic Medium*, Masters Thesis, Department of Mathematics, Statistics, and Computing Science, Dalhousie University, 1996.

[5] M. Courtemanche, R.J. Ramirez, S. Nattel, *Ionic mechanisms underlying human atrial action potential properties: insights from a mathematical model*, Am. J. Physiol. 275 (Heart Circ. Physiol. **44**), H301-H321, 1998.

[6] R. FitzHugh, *Impulses and physiological states in theoretical models of nerve membrane*, Biophys. J., **1**,445–466, 1961.

[7] E. Hairer, S. Norsett, and G. Wanner, *Solving Ordinary Differential Equations I*, Springer-Verlag, 1987.

[8] The Heart and Stroke Foundation of Canada, `http://www.heartandstroke.ca`, October 1, 2002.

[9] B.M. Horacek, J.W. Warren, P. Stovicek, and C.L. Feldman, *Diagnostic accuracy of derived versus standard 12-lead electrocardiograms*, Journal of Electrocardiology, **33**, Suppl:155-60, 2000.

[10] J.D. Lambert, *Numerical Methods for Ordinary Differential Systems*, John Wiley & Sons, 1991.

[11] G.T. Lines, *Simulating the Electrical Activity of the Heart, A Bidomain Model of the Ventricles Imbedded in a Torso*, Doct. Scient. thesis, Department of Informatics, University of Oslo, 1999.

[12] C-H. Luo and Y. Rudy, *A model of the ventricular cardiac action potential: Depolarization, repolarization, and their interaction*, Circ. Res., **68(6)**, 1501–1526, 1991.

[13] C-H. Luo and Yoram Rudy, *A Dynamic Model of the Cardiac Ventricular Action Potential: I. Simulations of Ionic Currents and Concentration Changes*, Circ. Res., **74**, 1071-1096, 1994.

[14] R.E. Mickens, *Nonstandard Finite Difference Models of Differential Equations*, World Scientific, Singapore, 1994.

[15] J.S. Nagumo, S. Arimoto, and S. Yoshizawa, *An active pulse transmission line simulating nerve axon*, Proc. IRE. **50**, 2061–2071, 1962.

[16] A. Nygren, C. Fiset, L. Firek, J.W. Clark, D.S. Lindblad, R.B. Clark, and W.R. Giles, *Mathematical Model of an Adult Human Atrial Cell, The Role of K^+ Currents in Repolarization*, Circ Res., **80**, 63-81, 1998.

[17] M. Patterson and R.J. Spiteri, *odeToJava software package*, available at http://www.netlib.org/ode/odeToJava.tgz, 2002.

[18] D. Randall, W. Burggren, and K. French, *Eckert Animal Physiology, Mechanisms and Adaptations, Fourth Edition*, W. H. Freeman and Company, New York, 1997.

[19] J.G. Verwer, J.G. Blom, and W. Hundsdorfer, *An implicit-explicit approach for atmospheric transport-chemistry problems*, Appl. Numer. Math. **20(1–2)**, 191–209, 1996.