Published in final edited form as:

IEEE Trans Biomed Eng. 2008 December ; 55(12): 2701–2711. doi:10.1109/TBME.2008.925673.

Efficient Fully Implicit Time Integration Methods for Modeling Cardiac Dynamics

Wenjun Ying,

Department of Biomedical Engineering, Duke University, Durham, NC 27708-0281 USA

Donald J. Rose, and

Department of Computer Science, Duke University, Durham, NC 27708-0129 USA

Craig S. Henriquez

Departments of Biomedical Engineering and Computer Science, Duke University, Durham, NC 27708-0281 USA

Wenjun Ying: ying@celmail.bme.duke.edu

Abstract

Implicit methods are well known to have greater stability than explicit methods for stiff systems, but they often are not used in practice due to perceived computational complexity. This paper applies the Backward Euler method and a second-order one-step two-stage composite backward differentiation formula (C-BDF2) for the monodomain equations arising from mathematically modeling the electrical activity of the heart. The C-BDF2 scheme is an L-stable implicit time integration method and easily implementable. It uses the simplest Forward Euler and Backward Euler methods as fundamental building blocks. The nonlinear system resulting from application of the Backward Euler method for the monodomain equations is solved for the first time by a nonlinear elimination method, which eliminates local and non-symmetric components by using a Jacobian-free Newton solver, called Newton-Krylov solver. Unlike other fully implicit methods proposed for the monodomain equations in the literature, the Jacobian of the global system after the nonlinear elimination has much smaller size, is symmetric and possibly positive definite, which can be solved efficiently by standard optimal solvers. Numerical results are presented demonstrating that the C-BDF2 scheme can yield accurate results with less CPU times than explicit methods for both a single patch and spatially extended domains.

Index Terms

Reaction-Diffusion equation; monodomain equations; transmembrane potential; fully implicit methods; composite backward differentiation formula; C-BDF2

1. Introduction

The electrical activity of the heart, including that from a membrane patch or from a tissue of coupled cells, can be mathematically described by differential equations. When modeling the action potential of a single patch, the equations are simply ordinary differential equations (ODEs). When modeling the propagation of action potentials, the equations become partial differential equations (PDEs), mainly of reaction-diffusion type. For both cases, the differential equations usually present themselves as stiff problems since the cardiac electrical activity typically involves multiple and widely varying scales [1], [2].

For stiff problems, implicit time integration methods are in general more accurate and stable than explicit methods, semi-implicit methods [3], [4] and other nonstandard methods such as those based on operator splittings [5]–[9]. A discussion on the stability restriction associated with explicit/semi-implicit methods and the potential order-reduction associated with operator-splitting based methods can be found in several sources (e.g. [10]–[12]). Despite the advantages, however, the application of fully implicit methods for modeling cardiac impulse propagation has been limited to a narrow class of membrane models of the Hodgkin-Huxley [13] type or simpler, for which the semi-linear property of the model is used or the Jacobians are generally straightforward to compute [14]–[20].

In this paper, we apply the Backward Euler method and a second-order one-step two-stage, easily implementable and fully implicit, L-stable time integration method for solving cardiac electrical dynamics models, including both ODEs and PDEs. The one-step two-stage method, called C-BDF2 scheme, is a composite scheme of the Backward Euler method (BDF1) and the backward differentiation formula of second-order (BDF2) that uses the simplest Forward Euler (FE) and Backward Euler (BE) methods as fundamental building blocks. Application of the BE method for the PDEs yields two nonlinear systems, which are coupled: one is global, corresponding to the space-dependent part, and another one is local, corresponding to the space-independent part. The local system in general has a non-symmetric Jacobian and significantly complicates the coupled system. We solve the coupled system with the nonlinear elimination method [21]. Specifically, we eliminate the non-symmetric components due to the local system with a Jacobian-free Newton solver, which avoids direct evaluation of the Jacobian of complicated cardiac dynamics functions. During the nonlinear elimination, the linearized equations of the local system are solved with a Krylov subspace method, which only requires matrix-vector multiplications while the matrix-vector products are achieved through numerical differentiation. After the nonlinear elimination, the Jacobian of the resulting global system becomes symmetric and has much smaller size than those in [19], [20], which directly apply Newton-Krylov solvers on the coupled system and generally have global Jacobian matrices non-symmetric and much more difficult to invert. As one big advantage of the nonlinear elimination method over others such as [19], [20], the linearized equations of the global system could be more efficiently solved by standard optimal solvers. Results are presented that show both of the fully implicit methods perform efficiently and the second-order method, C-BDF2 scheme, can achieve good accuracy with large time steps and thus can solve the equations with less CPU time than explicit methods, despite the greater computational complexity.

2. Models

The electrical activity of the heart or a cardiac tissue can be modeled by the following reaction-diffusion equations in an appropriate computational domain Ω :

$$C_m \frac{\partial u}{\partial t} + I_{ion}(u, \mathbf{q}) = \frac{1}{\beta} \nabla \cdot (\sigma \nabla u), \quad (2.1a)$$

$$\frac{\partial \mathbf{q}}{\partial t} = \mathcal{M}(u, \mathbf{q}). \quad (2.1b)$$

Here, u is the transmembrane potential (units: mV); $\mathbf{q}$ is a set of gating and ion concentration variables; σ is the electrical conductivity (units: mS · cm⁻¹); β is the surface-to-volume ratio (units: cm⁻¹); C_m is the membrane capacitance per unit area (units: μF/cm²). $I_{ion}(u, \mathbf{q})$ and $\mathcal{M}(u, \mathbf{q})$ are typically nonlinear functions, defining the ionic/resistive current and the membrane dynamics, respectively. The system (2.1) is often referred to as the monodomain

equations [1], [2]. Together with some appropriate initial and boundary conditions, the monodomain equations can be solved with standard space discretization and time integration methods. This work is focused on the fully implicit time integration.

3. Methods

A. The Composite Backward Differentiation Formula

The second-order two-stage composite backward differentiation formula (C-BDF2) is a variant of the TR-BDF2 scheme [22] and can be extended to a larger class of one-step multi-stage methods as analogs of but with better stability properties than the standard multi-step backward differentiation formulas (BDFs) [23].

This section will briefly present the C-BDF2 scheme, developed to solve the equation

$$\frac{du(t)}{dt} = f(u) \quad (3.2)$$

from time t^n to $t^{n+1} = t^n + \Delta t$. Here, $f(u)$ is regarded as a nonlinear operator, which may contain the Laplacian operator or the general elliptic operator. Let $w_0 = u^n$ be an approximation of $u(t^n)$ and w_1 be an intermediate solution, approximating the solution $u(t)$ at some point in the time interval (t^n, t^{n+1}) . The two-stage C-BDF2 scheme is given by

$$w_1 - \gamma \Delta t f(w_1) = w_0, \quad (3.3a)$$

$$w_2 - \gamma \Delta t f(w_2) = \frac{1-\gamma}{\gamma} w_1 + \frac{2\gamma-1}{\gamma} w_0, \quad (3.3b)$$

with the characteristic constant $\gamma > 0$ to be determined. The computed value w_2 in (3.3b) approximates the exact solution $u(t)$ at the new time t^{n+1} . This scheme (3.3) has a local truncation error of third-order,

$$E_{C-BDF2} = \left\{ \frac{1}{6} - \frac{2\gamma^2 - 2\gamma + 1}{4(1-\gamma)} \right\} f'^2 f \Delta t^3 + \left\{ \frac{1}{6} - \frac{1-\gamma}{4} \right\} f^2 f'' \Delta t^3, \quad (3.4)$$

and a stability function,

$$S(z) = \frac{1 + (1-2\gamma)z}{(1-\gamma z)^2} \text{ for } z \in \mathbb{C}. \quad (3.5)$$

With a particular value of the characteristic constant, $\gamma = 1 - \sqrt{2}/2$, the stability function (3.5) is bounded by one for all numbers on the left half complex plane and converges to zero as the real part of the complex number z tends to negative infinity [24]. This means the second-order two-stage C-BDF2 scheme (3.3) is both unconditionally stable and stiffly accurate for stiff problems, i.e., L-stable, too.

Note that the first-stage in the C-BDF2 scheme (3.3) is the first-order backward differentiation formula (BDF1), the Backward Euler method. The second-stage in the scheme is the second-order backward differentiation formula (BDF2) [23]. It is well-known [25] that the BDF2 scheme is not L-stable for stiff problems. As indicated, however, the composition of the BDF1 and BDF2 schemes is L-stable with the particular value $\gamma = 1 - \sqrt{2}/2$.

Note that the right hand side of the second-stage in (3.3) is an extrapolation of the past solution w_0 and the intermediate solution w_1 . The extrapolated value is denoted by

$$v_1 \equiv \frac{1-\gamma}{\gamma} w_1 + \frac{2\gamma-1}{\gamma} w_0. \quad (3.6)$$

Each stage in (3.3) can be regarded as time integration of the evolution equation (3.2) by a timestep $\gamma\Delta t$ with the simplest BE method and the initial data given by either w_0 or v_1 .

Naturally, a good initial guess, $w_1^{(0)}$ or $w_2^{(0)}$, for iteratively solving the corresponding nonlinear equation can be computed with the FE method,

$$w_1^{(0)} = w_0 + \gamma\Delta t f(w_0), \quad (3.7)$$

or

$$w_2^{(0)} = v_1 + \gamma\Delta t f(v_1). \quad (3.8)$$

In this sense, the C-BDF2 scheme (3.3) is made up of the simplest FE and BE methods, which are connected by an extrapolation operation on the past and intermediate solutions. Once the FE and BE methods are implemented, the two-stage C-BDF2 scheme can be constructed.

B. The Forward Euler Method

Let u^n and $\mathbf{q}^n$ be approximations of $u(t)$ and $\mathbf{q}(t)$ at time $t = t^n$, respectively. Discretizing the differential equations (2.1) with the FE method from time t^n to $t^{n+1} = t^n + \Delta t$ yields

$$C_m \frac{u^{n+1} - u^n}{\Delta t} + I_{ion}(u^n, \mathbf{q}^n) = \frac{1}{\beta} \nabla \cdot (\sigma \nabla u^n), \quad (3.9)$$

$$\frac{\mathbf{q}^{n+1} - \mathbf{q}^n}{\Delta t} = \mathcal{M}(u^n, \mathbf{q}^n). \quad (3.10)$$

The system (3.9)–(3.10) is a semi-discretization of the monodomain equations (2.1). The continuous elliptic operator on the right hand side of (3.9) need further discretization.

In this paper, we only discuss the discretization of the elliptic part with the second-order cell-centered three-point finite difference method as the space discretization is not the main focus. As a matter of fact, when another method such as the finite volume or finite element method is applied for the space discretization, an equivalent or identical discrete system can be derived if the integrals involved are evaluated with appropriate quadratures or the technique of mass lumping is applied.

Further space discretization of (3.9) with the finite difference method leads to

$$\mathbf{u}^{n+1} = \mathbf{u}^n - \frac{\Delta t}{\beta C_m} \mathbf{K} \mathbf{u}^{n+1} - \frac{\Delta t}{C_m} \mathbf{I}_{ion}(\mathbf{u}^{n+1}), \quad (3.11)$$

with $\mathbf{K}$ usually being a symmetric non-negative definite matrix, and $\mathbf{I}_{ion}(\mathbf{u}) = (I_{ion}(u_1), I_{ion}(u_2), \dots, I_{ion}(u_N))^T$ being a vector-valued current function of the potential vector $\mathbf{u} = (u_1, u_2, \dots, u_N)^T$. Here, N denotes the spatial degree of freedoms in the discretization.

It is obvious that the forward Euler method does not require any linear or nonlinear system solver but only evaluation of the nonlinear current functions and one matrix-vector product.

C. The Backward Euler Method

Discretizing the differential equations (2.1) with the BE method from time t^n to $t^{n+1} = t^n + \Delta t$ yields

$$C_m \frac{u^{n+1} - u^n}{\Delta t} + I_{ion}(u^{n+1}, \mathbf{q}^{n+1}) = \frac{1}{\beta} \nabla \cdot (\sigma \nabla u^{n+1}), \quad (3.12)$$

$$\frac{\mathbf{q}^{n+1} - \mathbf{q}^n}{\Delta t} = \mathcal{M}(u^{n+1}, \mathbf{q}^{n+1}). \quad (3.13)$$

The vector $\mathbf{q}^{n+1}$ of state variables can be computed in terms of u^{n+1} and $\mathbf{q}^n$ as a solution to the nonlinear sub-system (3.13). Here, assume that the system (3.13) has a unique solution or at least isolated solutions for timestep small enough. So, the vector $\mathbf{q}^{n+1}$ of state variables can be conveniently written as a function of u^{n+1} with parameters $\mathbf{q}^n$, i.e.,

$$\mathbf{q}^{n+1} = \mathbf{q}^{n+1}(u^{n+1}; \mathbf{q}^n), \quad (3.14)$$

and the ionic current I_{ion} in (3.12) can be regarded as a function of the potential, i.e.,

$$I_{ion}(u^{n+1}, \mathbf{q}^{n+1}) = I_{ion}(u^{n+1}, \mathbf{q}^{n+1}(u^{n+1}; \mathbf{q}^n)) = I_{ion}(u^{n+1}; \mathbf{q}^n) \equiv I_{ion}(u^{n+1}). \quad (3.15)$$

With this notation, the partial differential equation (3.12) becomes

$$u^{n+1} - \frac{\Delta t}{\beta C_m} \nabla \cdot (\sigma \nabla u^{n+1}) + \frac{\Delta t}{C_m} I_{ion}(u^{n+1}) = u^n. \quad (3.16)$$

Further space discretization of (3.16) with the finite difference method leads to

$$\mathbf{u}^{n+1} + \frac{\Delta t}{\beta C_m} \mathbf{K} \mathbf{u}^{n+1} + \frac{\Delta t}{C_m} \mathbf{I}_{ion}(\mathbf{u}^{n+1}) = \mathbf{u}^n. \quad (3.17)$$

As a matter of fact, the procedure by Eqns. (3.14)–(3.17) above is a non-linear elimination process as the state variables $\mathbf{q}^{n+1}$ are locally eliminated. After this nonlinear elimination, the resulting nonlinear system (3.17) is more concise and has a smaller-size and better conditioned Jacobian.

So far, with the BE method, two non-linear systems, (3.13) and (3.17), are derived. Each of them will be solved with (inexact) Newton iterations [26], [27]. The local system (3.13) is spatially independent, has a small dimension and in general its Jacobian matrix is nonsymmetric and complicated to compute for the cardiac models. The global system (3.17) is space-dependent, has a large dimension but its Jacobian matrix is symmetric and could be positive definite as well. Here, note that the Jacobian component of the global system (3.17) due to the nonlinear function $\mathbf{I}_{ion}(\mathbf{u}^{n+1})$ is a diagonal matrix. In one space dimension, since the three-point finite difference method is used to discretize the elliptic operator, the final Jacobian matrix of (3.17) remains tridiagonal and the linear system can be solved with the tridiagonal matrix algorithm if the matrix is diagonally dominant [28]. In multiple space

dimensions, as long as the timestep size is small enough, the diagonal matrix will not change the positiveness of the global Jacobian matrix. In this case, efficient linear solvers such as (preconditioned) conjugate gradient methods or multigrid methods [29], [30] could be applied. In the worst case, when the Jacobian matrix is indefinite, a Krylov subspace method such as GMRES [31] or BI-CGSTAB [32] is still applicable.

D. Jacobian-free Inexact Newton Iteration for the Local Systems

As more and more experimental findings are added to the cardiac models [4], the nonlinear functions, $I_{ion}(u, \mathbf{q})$ and $\mathcal{M}(u, \mathbf{q})$, describing membrane dynamics of a cardiac cell are becoming very complicated. It is normal in a modern cardiac model that the vector-valued function $\mathcal{M}(u, \mathbf{q})$ and the state variable vector $\mathbf{q}$ contain tens of entries. The function components in $\mathcal{M}(u, \mathbf{q})$ may be highly nonlinear and only piecewisely defined. This makes the analytical calculation of its Jacobian matrix very difficult, tedious and error-prone. In addition, even if the Jacobian matrices are correctly computed, the repeated factorization during each Newton iteration for (3.13) could be expensive.

For the reasons stated above, an inexact Newton method, which does not compute the Jacobian matrices explicitly, is adapted for solving the local system (3.13) at each grid node. The original linear systems involved in each Newton iteration are solved with a Krylov subspace method [31]–[33]. The matrix-vector multiplication encountered in the Krylov subspace solver is approximated by numerical differentiation. For example, let $\mathbf{J}(\mathbf{q})$ be the Jacobian matrix of the vector-valued function $\mathcal{M}(\cdot, \mathbf{q})$, and $\mathbf{p}$ be a Newton direction, which has the same dimensions as the state variable vector $\mathbf{q}$. Numerical differentiation of the nonlinear function $\mathcal{M}(\cdot, \mathbf{q})$ along the direction $\mathbf{p}$ yields a good approximation of the matrix-vector multiplication. In other words, the product can be either approximated by the forward difference

$$\mathbf{J}(\mathbf{q})\mathbf{p} \approx \frac{\mathcal{M}(\cdot, \mathbf{q} + \epsilon\mathbf{p}) - \mathcal{M}(\cdot, \mathbf{q})}{\epsilon} \quad (3.18)$$

or the central difference

$$\mathbf{J}(\mathbf{q})\mathbf{p} \approx \frac{\mathcal{M}(\cdot, \mathbf{q} + \epsilon\mathbf{p}) - \mathcal{M}(\cdot, \mathbf{q} - \epsilon\mathbf{p})}{2\epsilon} \quad (3.19)$$

with the perturbation parameter ϵ small enough but not too small for the stability considerations [28]. This approach, which is also used by Moore [19] and Murillo and Cai [20] for the Newton-Krylov method, is becoming popular in scientific computing communities [27], [34]–[37].

E. Partial Derivatives of the Current Vector for the Global System

To compute the Jacobian matrix for the global system (3.17), the partial derivatives of the nonlinear function $\mathbf{I}_{ion}(\mathbf{u})$ are also approximated by numerical differentiation, either the forward difference

$$\frac{\partial I_{ion}(u_k)}{\partial u_k} \approx \frac{I_{ion}(u_k + \delta) - I_{ion}(u_k)}{\delta} \quad (3.20)$$

or the central difference

$$\frac{\partial I_{ion}(u_k)}{\partial u_k} \approx \frac{I_{ion}(u_k + \delta) - I_{ion}(u_k - \delta)}{2\delta} \quad (3.21)$$

for $1 \leq k \leq N$, with a small parameter $\delta > 0$. By the definition of the current function in (3.15), the non-linear local system (3.13) must be solved every time a component $I_{ion}(u_k)$ of the current vector $\mathbf{I}_{ion}(\mathbf{u})$ is evaluated, for $k \in \{1, 2, \dots, N\}$.

4. Algorithm Summary

In this section, we only summarize the implementation of the BE process described previously. The assembly of the fundamental building blocks into the second-order implicit C-BDF2 scheme (3.3) is straightforward.

For conciseness, let us first introduce two nonlinear vector-valued functions,

$$\mathbf{f}(\mathbf{q}) \equiv \mathbf{q} - \Delta t \mathcal{M}(u^{n+1}, \mathbf{q}) - \mathbf{q}^n \quad (4.22)$$

and

$$\mathbf{F}(\mathbf{u}) \equiv \left(\mathbf{I} + \frac{\Delta t}{\beta C_m} \mathbf{K} \right) \mathbf{u} + \frac{\Delta t}{C_m} \mathbf{I}_{ion}(\mathbf{u}) - \mathbf{u}^n, \quad (4.23)$$

respectively corresponding to the local and global systems, (3.13) and (3.17). So, the systems are simply re-written as

$$\mathbf{f}(\mathbf{q}^{n+1}) = 0, \quad (4.24)$$

and

$$\mathbf{F}(\mathbf{u}^{n+1}) = 0. \quad (4.25)$$

To solve the global system (4.25), do the following steps:

- **Step 1.** Let v be the iteration index. Get a guess $\mathbf{u}_v^{n+1}$ ($v=0$) by integrating the monodomain equation (2.1) with the FE method and the initial data given by $(\mathbf{u}^n, \mathbf{q}^n)$. Details are omitted here.
- **Step 2.** Check whether the norm (see below) of the vector-valued function $\mathbf{F}(\mathbf{u}_v)^{n+1}$ is less than a prescribed (absolute) Newton tolerance $tol_{Newton}^{(G)}$. If

$$\|\mathbf{F}(\mathbf{u}_v^{n+1})\| < tol_{Newton}^{(G)}, \quad (4.26)$$

the vector $\mathbf{u}_v^{n+1}$ is used as an approximation of $\mathbf{u}^{n+1}$ and the iteration stops.

- **Step 3.** Solve the linearized system

$$\mathbf{F}'(\mathbf{u}_v^{n+1}) \mathbf{w}_v + \mathbf{F}(\mathbf{u}_v^{n+1}) = 0 \quad (4.27)$$

with a direct method (tridiagonal solver in one space dimension) or a Krylov iterative method such that

$$\|\mathbf{F}'(\mathbf{u}_v^{n+1}) \mathbf{w}_v + \mathbf{F}(\mathbf{u}_v^{n+1})\| < tol_{Krylov}^{(G)} \|\mathbf{F}(\mathbf{u}_v^{n+1})\|. \quad (4.28)$$

- **Step 4.** Update the potential vector $\mathbf{u}_v^{n+1}$ by

$$\mathbf{u}_{v+1}^{n+1} = \mathbf{u}_v^{n+1} + \lambda_v \mathbf{w}_v, \quad (4.29)$$

with $\lambda_v \in (0, 1]$ being a damping parameter [26] such that

$$\|\mathbf{F}(\mathbf{u}_{v+1}^{n+1})\| < \|\mathbf{F}(\mathbf{u}_v^{n+1})\|. \quad (4.30)$$

Increase index v by one and go to step G2 above.

For each fixed action potential u^{n+1} , to solve the local system (4.24), we follow a procedure similar to Steps 1–4 above except the global nonlinear function $\mathbf{F}$ is replaced by the local nonlinear function $\mathbf{f}$ and a different Newton iteration tolerance, denoted by $tol_{Newton}^{(G)}$, may be used.

In our implementation, the norm $\|\cdot\|$ of a vector $\mathbf{v} = (v_1, v_2, \dots, v_d)^T \in \mathbb{R}^d$ takes the scaled discrete $\hat{L}$ -norm, i.e.,

$$\|\mathbf{v}\| \equiv \sqrt{\frac{1}{d} \sum_{k=1}^d v_k^2}. \quad (4.31)$$

For one-dimensional problems, the global linear system (4.27) is solved with the standard tridiagonal solver. The linearized equations for the local nonlinear system are solved with the BI-CGSTAB Krylov method [32].

For zero-dimensional problems, the ODEs (the monodomain equations (2.1) with conductivity $\sigma = 0$), a procedure analogous to steps L1–L4 above can be applied.

5. Implementation Details

For efficient implementation of the fully implicit methods, the BE method and the C-BDF2 scheme, some tuning of the algorithm and certain parameters is needed. In our current implementation, as the timestep sizes used are only reasonably large, no damping is necessary during the Newton iterations for solving the local system. In other words, the damping parameter in (4.30) is fixed to be one, i.e., $\lambda_v = 1$. The parameter damping procedure is applied only for solving the global system. During the damping process, the parameter λ_v starts with value one and is halved every time it is detected that the residual norm is non-decreasing. To avoid un-necessary (inefficient) damping, the implementation only allows maximum three dampings in a row. That is, the parameter λ_v cannot be smaller than 2^{-3} .

Note that the evaluation of the global nonlinear function could be expensive. To avoid the times the function is un-necessarily evaluated, the parameter damping for the global system is activated only when the iteration count v is greater than a prescribed number ($v = 6$ in our implementation).

To further reduce the cost of the global nonlinear solver, the full Jacobian matrix is computed only when the iteration count v is greater than a prescribed number ($v = 3$ in our

implementation). Otherwise, only the component $(\mathbf{I} + \frac{\Delta t}{\beta C_m} \mathbf{K})$ corresponding to the linear part in (4.23) is used for solving the linearized system, in place of the full Jacobian matrix.

Recall that each Newton iteration for solving the global system involves one or multiple times evaluation of the ionic current as a function of the action potential at each grid node. One evaluation of the current function corresponds to solving the local system once with the Newton-Krylov method similar to Steps 1–4. We note that, at each grid node, the computed gating/concentration variables at two consecutive iterations should be close. There is no need in every occurrence to compute the initial guess for solving the local system with the FE method. Instead, in each timestep for integrating the global system, it is sufficient to compute the initial guess $\mathbf{q}_0^{n+1}$ for solving the local system with the FE method only in the first iteration for each grid node. For the second and thereafter iterations, the values of the gating/concentration variables computed from the previous iteration are good enough as initial guess for the Newton-Krylov solver for the local system.

Finally, we note that, when the differential equations described by the ionic models are integrated with the Jacobian-free solver, the absolute tolerances used for stopping the Newton and BI-CGSTAB iterations could not be arbitrarily small. As the linearized system is only solved approximately (inexactly) and even the matrix-vector multiplications are only approximately computed by numerical differentiation, the inexact Newton-Krylov method is not guaranteed to converge with a quadratic rate, or could fail to converge. These tolerances and the difference parameters must be carefully selected for the Jacobian-matrix free solvers to work efficiently.

6. Results

This section presents two sets of simulation results: one is for solving the ODEs associated with a membrane patch and the other is for solving the PDE associated with a one dimensional cardiac fiber. The fully implicit time integration methods described above, the BE method and the C-BDF2 scheme, were implemented in custom codes written in C++. The simulations presented in this section were all performed in double precision on a dual Xeon 3.6 GHz computer. The CPU times that are presented correspond to the case when (almost) no output was generated. In addition, the number of Newton iterations is given to indicate the number of times the corresponding linear system is solved.

Membrane Patch

Both the BE method and the C-BDF2 scheme were applied to the ODEs governing the ion fluxes in a patch of membrane described by DiFrancesco and Noble model [38], [39] (DFN), which has 15 state variables other than the transmembrane potential, Courtemanche et al. model [40] (CRN), which has 20 state variables other than the transmembrane potential, and Bondarenko et al. model [41] (BON), which has 40 state variables other than the transmembrane potential. The three models have disparate stiffness, which are indicated below by the restricted sizes of the timesteps on the Forward Euler method applied. In each of the simulations, the patch is assumed to be initially at rest. A current stimulus with strength $I_{stim} = -20.0 \mu\text{A}/\text{cm}^2$ is applied in the time interval between $t = 10.0$ msec and $t = 12.0$ msec.

When solving the ODEs, the Newton and Krylov tolerances, $tol_{Newton}^{(L)}$ and $tol_{Krylov}^{(L)}$, are both selected to be equal to 10^{-6} . The matrix-vector multiplication required by the BI-CGSTAB solver is approximated by the central difference (3.19) with parameter $\epsilon = 10^{-6}$.

We first compare the performance of the two-stage second-order C-BDF2 scheme (3.3) with the first-order BE and FE methods for the three different membrane models when all use the same timestep. For the DFN model [38], the timestep is chosen to be 0.01 msec. For the CRN model [40], the timestep is chosen to be 0.01 msec. For the BON model, the timestep

is chosen to be 0.1 μ s. In each case, the time step was chosen to be (nearly) at the stability limit of the FE Method. In addition, the time over which the action potential was simulated is different due to the difference in action potential duration associated with the three models. As expected, for the same time step, the explicit FE simulation was faster than the BE and the C-BDF2 schemes. Table I shows that the FE method requires approximately 3 – 4 times less work than the other two schemes.

The advantage of implicit methods, however, is that they normally allow much larger timesteps than explicit methods. Because the C-BDF2 method is second order, we expect greater accuracy than the first order BE for the same time step. Simulations were performed using larger time steps to solve the ODEs for the three membrane models with the BE method and the two-stage second-order C-BDF2 scheme (3.3).

Figure 1 shows the plots of action potentials from the simulations with the DFN model. The action potential traces shown in Fig. 1(a) were from the simulations with the C-BDF2 and BE methods, both of which use the same timestep, $\Delta t = 1.0$ msec. The simulation of 800.0 msec of activity used 0.11 sec of CPU time with the BE method and 0.12 sec of CPU time with the C-BDF2 scheme. The maximum numbers of Newton iterations used by the BE and the C-BDF2 methods are 45 and 3, respectively. The average numbers of Newton iterations per timestep are 1.06 and 1.66, respectively. The maximum numbers of BI-CGSTAB iterations used by the BE and the C-BDF2 methods are 13 and 8, respectively. The average numbers of BI-CGSTAB iterations are 2.1 and 1.3, respectively. Fig. 1(a), shows that for the same time step, the two methods give slightly different results. The largest difference between the action potential traces occurs near peak (approximately 8.0mV). Reducing the time step of the BE scheme to ($\Delta t = 0.01$ msec) produced a solution that was nearly identical to that with the C-BDF2 scheme with a $\Delta t = 1.0$ msec (see Fig. 1(b)).

The difference between solutions obtained using the BE and the C-BDF2 schemes depends on the membrane model. Figure 2 shows the action potentials from the simulations with the CRN model. As was the case with the DFN model, the BE and C-BDF2 methods used nearly the same CPU time with the same time step of $\Delta t = 0.5$ msec. For a simulation of 400.0 msec of activity, both schemes required 0.11 sec. The maximum numbers of Newton iterations used by the BE and the C-BDF2 methods are 53 and 4, respectively. The average numbers of Newton iterations per timestep are 1.1 and 1.8, respectively. The maximum numbers of BI-CGSTAB iterations used by the BE and the C-BDF2 methods are 16 and 81, respectively. The average numbers of BI-CGSTAB iterations are 2.8 and 1.7, respectively. Figure 2 shows that the difference in the two solutions was greater than DFN, with deviations in morphology throughout the duration of the action potential. Again reducing the time step of the BE scheme to ($\Delta t = 0.01$ msec) produced a solution that was nearly identical to that obtained with C-BDF2 with a time step of $\Delta t = 0.5$ msec (see Fig. 2(b)).

Because the BON model required a much smaller time step than the other models, the difference in solutions obtained by the BE and C-BDF2 was smaller than that observed for the other membrane models. Note that a time step larger than $\Delta t = 0.05$ msec produced an instability in the model. This instability is different from the one associated with explicit time integration methods for linear simple ODEs. Instead, it is because of the inherent nonlinear complexity of the ionic model. For example, the presence of logarithm functions in the model requires each iterated value be physically meaningful while this requirement can be easily violated when a large timestep is used. Figure 3 shows the action potentials from the simulations. Again, the BE and C-BDF2 methods used approximately the same CPU time, 0.29 sec, for 60.0 msec of activity. The larger CPU time than the other models reflects the considerably greater complexity of the BON formulation even with a

significantly shorter action potential duration. The maximum numbers of Newton iterations used by the BE and the C-BDF2 methods are 51 and 14, respectively. The average numbers of Newton iterations per timestep are 1.0 and 1.76, respectively. The maximum numbers of BI-CGSTAB iterations used by the BE and the C-BDF2 methods are 61 and 20, respectively. The average numbers of BI-CGSTAB iterations are 5.1 and 2.5, respectively. As shown in Fig. 3(a) and was the case for the DFN model, the largest difference between these two traces occurs at the peak of about 1.4mV. Reducing the time step of the BE scheme to ($\Delta t = 0.1 \mu s$) produced a solution that was identical to that obtained with C-BDF2 using a time step of $\Delta t = 0.05$ msec (see Fig. 3(b)).

The CPU times used for solving the ODEs with the three ionic models are summarized in Table I. The first row in each sub-table contains the simulation period. The second row has the timesteps used. Those in the other rows are the CPU times.

One Dimensional Fiber

We also used the C-BDF2 scheme to simulate propagation along a one-dimensional fiber with length $\mathcal{L} = 1.0$ cm. The membrane capacitance per unit area, the electrical conductivity and the surface-to-volume ratio are, respectively, selected to $C_m = 1.0 \mu F/cm^2$, $\sigma = 3mS \cdot cm^{-1}$ and $\beta = 1500.0 cm^{-1}$, which are consistent with those of R. Weber dos Santos et al. [42]. The computational domain $\Omega = (0, \mathcal{L})$ is partitioned into a uniform grid. The elliptic operator in (2.1a) is discretized with the second order cell-centered finite difference method. No-flux boundary conditions are applied at the endpoints of the domain.

To solve the PDEs, the matrix-vector multiplications for both the local and the global systems are approximated by the forward differences as one function evaluation can be re-used for computing the residual, and this makes the resulting algorithm more efficient.

The parameter δ for computing the partial derivatives of the current vector $\mathbf{I}_{ion}(\mathbf{u})$ with the forward difference (3.20) is chosen to be 10^{-4} for all models. The difference parameter ϵ for computing the matrix-vector multiplication involved in the Newton-Krylov solver for the local systems is model-dependent. For the DFN and the BON models, $\epsilon = 10^{-5}$. For the CRN model, $\epsilon = 10^{-6}$.

The Newton and Krylov tolerances, $tol_{Newton}^{(L)}$ and $tol_{Krylov}^{(L)}$ for solving the local systems are both selected to be 10^{-6} . The Newton tolerance $tol_{Newton}^{(G)}$ for the global nonlinear system is set as 10^{-5} .

For simplicity, we only present numerical results with the two-stage second-order C-BDF2 time integration method (3.3) while we remark that as shown for the patch case, the second-order method can use larger timesteps than the BE method to obtain the same accuracy, and the iteration tolerances and difference parameters may be chosen differently if the BE method is used. In each simulation, the gating and ion concentration variables $\mathbf{q}$ along the fiber are all assumed to be at rest at the beginning, except the potential u , which is given by

$$u(x) = \frac{100}{1 + e^{200(x-1/5)}} - 80. \quad (6.32)$$

Simulations were performed using all three membrane models for three spatial step sizes of $\Delta x = 0.02$ cm, $\Delta x = 0.01$ cm and $\Delta x = 0.005$ cm. As was the case with the patch simulations, the time steps and the total time of simulated activity varied depending on the model. In the simulation with membrane dynamics described by the DFN model, the timestep is fixed to be $\Delta t = 0.5$ msec. For the CRN model, the timestep is fixed to be $\Delta t =$

0.25 msecs. Finally, for the BON model, timestep is fixed to be $\Delta t = 0.01$ msecs. The time steps are generally smaller than those used in a patch. As a comparison, we also ran a simulation with the FE method. For FE, the timestep was chosen to be near the stability limit. For the DFN and CRN models, this required that the time steps be reduced to $\Delta t = 0.005$ msecs when $\Delta x = 0.005$ cm. For the BON, the stability limit requires such a small time step for FE ($\Delta t = 0.0002$ msecs) that no further reduction is required when the spatial step is refined.

As summarized in Tables II–IV, the C-BDF2 method required less CPU time than FE for all three models. For the DFN and CRN models, the difference is greater as the spatial step size is reduced. It is observed that the CPU times scale linearly with respect to the mesh parameter, and the average number of Newton iterations for solving the global systems is independent of the grid size. Because of the stability constraints, the C-BDF2 method was most efficient for the BON model. For the choices in time steps, the solutions with the two methods were nearly the same, and thus are not shown. Over most of the fiber, the difference is within an error less than 0.1 mV.

Finally, we also ran simulations with the C-BDF2 scheme for the ionic models on the fixed grid ($\Delta x = 0.01$ cm) but using different timesteps. The CPU times and the average numbers of Newton iterations are listed in Tables V–VII. For each set of simulations, we collected action potentials at five evenly spaced points along the cable, and observed the action potential converges as the timestep is decreased (decreasing the timestep tends to decrease the amplitude of the maximum action potential at the peak).

To further illustrate the accuracy of the C-BDF2 scheme for the one dimensional fiber, the average conduction velocities (CV) of the propagating wave fronts over the middle half of the fiber are reported in Tables II–VII. The action potential durations (APD) at the middle point of the fiber are also reported in Tables II–IV. In each experiment, the threshold value of action potential used for computing the CVs and APDs is fixed to be negative sixty millivolts, i.e., $u_{threshold} = -60$ mvolts.

7. Discussion

Numerical experiments demonstrate that both fully implicit methods, the BE method and the C-BDF2 scheme, for modeling the electrical activity of the heart are efficient. For integrating ODEs, the implicit methods run only a few times slower than the simplest FE method even if all use the same small timestep, the stability limit imposed on the explicit method. Because the implicit methods are both A-stable and L-stable for the stiff problems, much larger timesteps can be used with the fully implicit methods than the FE method. In particular, with the second-order fully implicit method, C-BDF2, large timesteps do not degrade the solution accuracy while the simulations are still several times faster than those with the FE method. This is true for both the ODEs and PDEs. The C-BDF2 scheme outperforms the first-order BE and FE methods. At present, all of the simulations with the implicit methods used constant timesteps. The important point is that the implicit methods allow variable-size timesteps. We could expect even greater efficiency when adaptive time stepping is used because larger timesteps can be used after the steep action-potential upstroke [24].

The approach adapted in this work for solving the coupled system, (3.12)–(3.13), looks like but is not the nonlinear block Gauss-Seidel iteration that is used in the early applications by Cooley and Dodge [14] and Mascagni [15]. Instead, it is more like the method used by Cox and Griffith [18], and actually is a variant of the nonlinear elimination method [21], which involves explicit Jacobian evaluation for the local systems though. The most significant

difference of the current approach from Cox and Griffith [18] is that it does not require the membrane model to have the special semi-linear property as the Hodgkin-Huxley model, and so it works for more general models.

Both the BE method and the C-BDF2 scheme are unconditionally stable (A-stable) and L-stable for stiff problems. Here, by convention, the A-stability and L-stability are defined for solving linear simple ODEs [11], [25], [43]. An L-stable scheme may encounter another stability issue when the problem to be solved has its own inherent instability such as that of the BON model, in which the presence of logarithm functions requires each iterated value be physically meaningful while this requirement can be easily violated when a large timestep is used. Nevertheless, the implicit methods in general are expected to have better performance than others. For example, the operator splitting based methods may suffer from order reduction for general problems due to either stiffness of the problems or different boundary conditions [12]. In addition, there is no stable operator splitting methods with accuracy order greater than two for the PDEs [44], [45], while in principle arbitrarily high order extensions of the fully implicit and L-stable methods exist and could be (relatively easily) obtained.

The fully implicit methods proposed here, which are based on the nonlinear elimination approach, for the reaction-diffusion equations may be more efficient than Murillo and Cai [20] as they use the Jacobian-free solver only in the node-by-node basis while Murillo and Cai [20] apply the Newton-Krylov solver for the spatially discretized global system. The proposed algorithm can be made adaptive and, therefore, should be more appropriate for combination with an adaptive mesh refinement algorithm, such as the one proposed by Trangenstein [7] or Ying [24], with which the Newton-Krylov solver for the sub-problem (3.13) could be avoided at coarse grid nodes and the solutions are replaced by simple time interpolation. With respect to parallel implementation, the implicit methods introduced here will involve much less inter-processor communication when the cardiac model used has tens of or more state variables as the variable updating by (3.13) is space-independent and highly localized.

While this paper only presents the numerical results for zero- and one-dimensional problems, it is straightforward to apply the implicit scheme for modeling cardiac electrical activities in multiple space dimensions. In higher dimensions, however, the tridiagonal solver used for the linear systems involved in each global Newton iteration should be replaced with another solver, such as a nonlinear multigrid [46] or a multilevel Newton method [47]. In addition, the method can be used with the bidomain model provided that the equations are solved as a degenerate parabolic system such that the Jacobian matrix is symmetric. The application of the implicit schemes to higher dimensions with and without the use of an adaptive mesh refinement algorithm will be reported in future works.

Acknowledgments

The authors would like to thank John B. Pormann for providing the C code for the DiFrancesco-Noble model and Vincent Jacquemet for providing the C code for the Courtemanche and Bondarenko models.

This work was supported in part by NIH grant R01HL76767.

References

1. Keener, JP.; Sneyd, J. *Mathematical Physiology*. New York: Springer-Verlag; 1998.
2. Panfilov, AV.; Holden, AV., editors. *Computational Biology of the Heart*. Wiley; 1997.
3. Keener JP, Bogar K. A numerical method for the solution of the bidomain equations in cardiac tissue. *Chaos*. 1998; vol. 8:234–241. [PubMed: 12779724]

4. Pullan, AJ.; Buist, ML.; Cheng, LK. Mathematically modelling the electrical activity of the heart: from cell to body surface and back again. World Scientific; 2005.
5. Qu Z, Garfinkel A. An advanced algorithm for solving partial differential equations in cardiac conduction. *IEEE Transactions on Biomedical Engineering*. 1999 Sep; vol. 46(no. 9):1166–1168. [PubMed: 10493080]
6. Vigmond E, Aguel F, Trayanova N. Computational techniques for solving the bidomain equations in three dimensions. *IEEE Trans. Biomed. Eng.* 2002; vol. 49(no. 11):1260–1269. [PubMed: 12450356]
7. Trangenstein JA, Kim C. Operator splitting and adaptive mesh refinement for the Luo-Rudy I model. *J. Computational Physics*. 2004; vol. 196:645–679.
8. Sundnes J, Lines GT, Tveito A. An operator splitting method for solving the bidomain equations coupled to a volume conductor model for the torso. *Mathematical Biosciences*. 2005; vol. 194(no. 2):233–248. [PubMed: 15854678]
9. Sundnes, J.; Lines, GT.; Cai, X-C.; Nielsen, BF.; Mardal, K-A.; Tveito, A. Computing the electrical activity in the heart. Springer-Verlag; 2006.
10. Hairer, E.; Wanner, G. Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems. Springer-Verlag; 1991.
11. Shampine, LF. Numerical Solution of Ordinary Differential Equations. Chapman & Hall; 1994.
12. Hundsdorfer, W.; Verwer, JG. Numerical solution of time-dependent advection-diffusion-reaction equations. Berlin Heidelberg: Springer-Verlag; 2003.
13. Hodgkin AL, Huxley AF. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology*. 1952; vol. 117:500–544. [PubMed: 12991237]
14. Cooley JW, Dodge FA Jr. Digital computer solutions for excitation and propagation of the nerve impulse. *Biophysical Journal*. 1966; vol. 6:583–599. [PubMed: 5970564]
15. Mascagni M. The Backward Euler method for numerical solution of the Hodgkin-Huxley equations of nerve conduction. *SIAM J. Numer. Anal.* 1990; vol. 27:941–962.
16. Hooke, NF. Ph.D. dissertation. Department of Computer Science, Duke University; 1992. Efficient simulation of action potential propagation in a bidomain.
17. Pollard AE, Hooke N, Henriquez CS. Cardiac propagation simulation. *Critical Reviews in Biomedical Engineering*. 1992; vol. 20(no.3–4):171–210. [PubMed: 1478091]
18. Cox, SJ.; Griffith, BE. Rice University, Tech. Rep. TR00-32; 2000. A fast, fully implicit Backward Euler solver for dendritic neurons.
19. Moore PK. An adaptive h-refinement finite element method for parabolic differential systems: some algorithmic considerations in solving in three space dimensions. *SIAM J. Sci. Comput.* 2000; vol. 21(no.4):1567–1586.
20. Murillo M, Cai X-C. A fully implicit parallel algorithm for simulating the non-linear electrical activity of the heart. *Numer. Linear Algebra Appl.* 2004; vol. 11:261–277.
21. Lanzkron PJ, Rose DJ, Wilkes JT. An analysis of approximate nonlinear elimination. *SIAM J. Sci. Comput.* 1996; vol. 17(no.2):538–559. [Online]. Available: <http://epubs.siam.org/sam-bin/dbq/article/25154>.
22. Bank RE, Coughran WM, Fichtner JW, Grosse EH, Rose DJ, Smith RK. Transient simulation of silicon devices and circuits. *IEEE Transactions on Electron Devices*. 1985; vol. 32(no.10):1992–2007.
23. Gear, CW. Numerical Initial Value Problems in Ordinary Differential Equations. Prentice Hall; 1971.
24. Ying, W-J. Ph.D. dissertation. Department of Mathematics, Duke University; 2005. A multilevel adaptive approach for computational cardiology.
25. Butcher, JC. Numerical Methods for Ordinary Differential Equations. Wiley; 2003.
26. Bank RE, Rose DJ. Global approximate Newton methods. *Numer. Math.* 1981; vol. 37:279–295.
27. Deufhard, P. Newton methods for nonlinear problems: Affine invariance and adaptive algorithms. Springer; 2004.
28. Atkinson, K. An Introduction to Numerical Analysis. Second Edition. Wiley; 1989.

29. Briggs, WL.; Henson, VE.; McCormick, SF. A Multigrid Tutorial. second edition. Philadelphia: SIAM Books; 2000.
30. Stüben K. A review of algebraic multigrid. J. Comput. Appl. Math. 2001; vol. 128(no.1–2):281–309.
31. Saad Y, Schultz MH. GMRES: A generalized minimal residual method for solving nonsymmetric linear systems. SIAM J. Sci. Statist. Comput. 1986; vol. 7:856–869.
32. van der Vorst HA. BI-CGSTAB: A fast and smoothly converging variant of BI-CG for the solution nonsymmetric linear systems. SIAM J. Sci. Stat. 1992; vol. 13:631–644.
33. Saad, Y. Iterative methods for sparse linear systems. Boston: PWS Publishing Company; 1996.
34. Brown P, Saad Y. Convergence theory of nonlinear Newton-Krylov algorithms. SIAM J. Optimization. 1994; vol. 4:297–330.
35. Kelley, CT. Iterative Methods for Linear and Nonlinear Equations. SIAM; 1995.
36. Brown PN, Hindmarsh AC, Petzold LR. Using Krylov methods in the solution of large-scale differential-algebraic systems. SIAM J. Sci. Comput. 1994; vol. 15:1467–1488.
37. Kelley, CT. Solving Nonlinear Equations with Newton's Method. SIAM; 2001.
38. DiFrancesco D, Noble D. A model of cardiac electrical activity incorporating ionic pumps and concentration changes. Phil. Trans of the Royal Soc. of London. 1985; vol. B-307:353.
39. Cabo C, Barr RC. Propagation model using the difrancesco-noble equations: Comparison to reported experimental results. Med. Biol. Eng. Comput. 1992; vol. 30:292–302. [PubMed: 1453801]
40. Courtemanche M, Ramirez R, Nattel S. Ionic mechanisms underlying human atrial action properties: Insights from a mathematical model. Am. J. Physiol. 1998; vol. 275:H301–H321. (Heart Circ. Physiol. 44). [PubMed: 9688927]
41. Bondarenko VE, Sziget G, Bett GCL, Kim SJ, Rasmusson RL. A computer model of the action potential of mouse ventricular myocytes. Am J Physiol (Heart Circ Physiol). 2004; vol. 287(no. 3):H1378–H1403. [PubMed: 15142845]
42. dos Santos RW, Plank G, Bauer S, Vigmond EJ. Parallel multigrid preconditioner for the cardiac bidomain model. IEEE Trans. Biomed. Eng. 2004; vol. 51(no.11):1960–1968.
43. Butcher, J. The Numerical Analysis of Ordinary Differential Equations: Runge-Kutta and General Linear Methods. Wiley; 1987.
44. Sheng Q. Solving partial differential equations by exponential splittings. IMA J. Numer. Anal. 1989; vol. 9:199–212.
45. Goldman D, Kaper TJ. N-th order split operator schemes and non-reversible systems. SIAM J. Numer. Anal. 1996; vol. 33:349–367.
46. Brandt A. Multi-level adaptive solutions to boundary-value problems. Math. Comp. 1977; vol. 31(no.193):333–390.
47. Bank RE, Rose DJ. Analysis of a multilevel iterative method for nonlinear finite element equations. Math. Comp. 1981; vol. 39:453–465.

Biographies

Wenjun Ying was born in 1976 in Zhejiang, China. He received the B.S. degree and the M.S. degree in Applied Mathematics from Tsinghua University, Beijing, in 1997 and 2000, respectively. He received the Ph.D. degree in Mathematics from Duke University, Durham, NC, in May, 2005. His Ph.D. degree dissertation is on the application of a multilevel adaptive method to modeling electrical wave propagation in the heart.

He currently works as a Research Associate at the Computational Electrophysiology Laboratory in the Department of Biomedical Engineering, Duke University. His research interests include computational electrophysiology, scientific computing and numerical analysis.

Donald J. Rose received the B.A. degree in mathematics from University of California-Berkeley in 1966, the A.M. and Ph.D. degrees in mathematics from Harvard University in 1967 and 1970 respectively.

He is now a professor of computer science and mathematics of Duke University. His research covers the areas of numerical solution of nonlinear algebraic and differential equations, numerical linear algebra and scientific computing.

Craig S. Henriquez received the B.S.E. degree in biomedical and electrical engineering from Duke University, Durham, NC, in 1981. After teaching high school for two years, he returned to Duke and received the Ph.D. degree in biomedical engineering in 1988.

He became a Research Assistant Professor in 1989 and an Assistant Professor of Biomedical Engineering in 1991 at Duke University. He is currently the Jeffrey N. Vinik Professor of Biomedical Engineering and Computer Science at Duke University. He is also the Co-Director of Duke's Center for Neuroengineering. His research interests include cardiac and neural electrophysiology, large-scale computer modeling, and neural analysis.

(a)

(b)

Fig. 1.

DiFrancesco-Noble model [38], [39]. (a) two traces of action potentials from two constant timestep ODE simulations with the membrane dynamics described by DiFrancesco-Noble model. The solid line is with the two-stage second-order C-BDF2 scheme (3.3) with $\Delta t = 1.0$ msecs. The dashed line is with the one-stage first-order BDF1 scheme, the BE method, with $\Delta t = 1.0$ msecs. (b) an enlarged part of panel a. The third one is from a simulation with the BE method too but with a much smaller timestep ($\Delta t = 0.01$ msecs).

(a)

(b)

Fig. 2. Courtemanche model [40]. (a) two traces of action potentials from two constant timestep ODE simulations with the membrane dynamics described by CRN model. The solid line is with the two-stage second-order C-BDF2 scheme (3.3) with $\Delta t = 0.5$ msecs. The dashed line is with the one-stage first-order BDF1 scheme, the BE method, with $\Delta t = 0.5$ msecs. (b) an enlarged part of panel a. The third one is from a simulation with the BE method too but with a much smaller timestep ($\Delta t = 0.01$ msecs).

(a)

(b)

Fig. 3.

Bondarenko model [41]. (a) two traces of action potentials from two constant timestep ODE simulations with the membrane dynamics described by BON model. The solid line is with the two-stage second-order C-BDF2 scheme (3.3) with $\Delta t = 0.05$ msecs. The dashed line is with the one-stage first-order BDF1 scheme, the BE method, with $\Delta t = 0.05$ msecs. (b) an enlarged part of panel a. The third one is from a simulation with the BE method too but with a much smaller timestep ($\Delta t = 0.1 \mu s$), which overlaps the solid curve.

TABLE I

The CPU Times Used for Solving the Odes.

	DFN	CRN	BON
duration	800.0 ms	400.0 ms	60.0 ms
timestep	0.01 ms	0.01 ms	0.1 μ s
F-Euler	0.77 secs	0.33 secs	5.13 secs
B-Euler	1.45 secs	0.59 secs	10.5 secs
C-BDF2	2.78 secs	1.08 secs	20.4 secs

	DFN	CRN	BON
duration	800.0 ms	400.0 ms	60.0 ms
timestep	1.0 ms	0.5 ms	0.05 ms
B-Euler	0.11 secs	0.11 secs	0.29 secs
C-BDF2	0.12 secs	0.11 secs	0.28 secs

DiFrancesco-Noble Model ($\bar{v}$: the average number of Newton Iterations used for solving the global system per timestep)

TABLE II

Δx (cm)	C-BDF2					FE			
	Δt (ms)	CPU (secs)	$\bar{v}$	CV (cm/s)	APD (ms)	Δt (ms)	CPU (secs)	CV (cm/s)	APD (ms)
0.02	0.5	7.76	1.94	65.2	332.6	0.01	27.90	64.7	332.6
0.01	0.5	14.8	1.93	65.9	332.5	0.01	55.47	65.8	332.5
0.005	0.5	29.3	1.92	66.1	332.5	0.005	219.8	66.3	332.4

TABLE III

Courtemanche model ($\bar{\nu}$: the average number of Newton iterations used for solving the global system per timestep)

Δx (cm)	C-BDF2					FE			
	Δt (ms)	CPU (secs)	$\bar{\nu}$	CV (cm/s)	APD (ms)	Δt (ms)	CPU (secs)	CV (cm/s)	APD (ms)
0.02	0.25	11.0	1.88	88.4	210.7	0.01	18.91	87.7	210.7
0.01	0.25	21.9	2.06	90.7	210.7	0.01	37.55	90.7	210.5
0.005	0.25	42.9	2.04	91.2	210.6	0.005	149.2	92.1	210.5

TABLE IV

Bondarenko model ($\bar{\nu}$: the average number of Newton iterations used for solving the global system per timestep)

Δx (cm)	C-BDF2					FE			
	Δt (ms)	CPU (secs)	$\bar{\nu}$	CV (cm/s)	APD (ms)	Δt (ms)	CPU (secs)	CV (cm/s)	APD (ms)
0.02	0.01	36.46	0.613	56.6	15.1	0.0002	198.7	56.6	15.1
0.01	0.01	71.84	0.605	57.6	15.0	0.0002	395.6	57.6	15.0
0.005	0.01	142.6	0.603	57.9	15.0	0.0002	790.0	57.9	15.0

DiFrancesco-noble model ($\bar{v}$: the average number of Newton iterations used for solving the global system per timestep)

TABLE V

C-BDF2 for DiFrancesco-Noble					
Δx (cm)	Δt (ms)	CPU (secs)	$\bar{v}$	CV (cm/s)	APD (ms)
0.01	0.5	14.8	1.93	65.9	332.5
0.01	0.25	25.8	1.64	65.9	332.5
0.01	0.125	37.1	1.42	66.1	332.4

Courtemanche model ($\bar{v}$: the average number of Newton iterations used for solving the global system per timestep)

TABLE VI

Δx (cm)	C-BDF2 for Courtemanche				
	Δt (ms)	CPU (secs)	$\bar{v}$	CV (cm/s)	APD (ms)
0.01	0.25	21.9	2.06	90.7	210.7
0.01	0.125	36.7	1.75	91.4	210.6
0.01	0.0625	50.5	1.10	91.7	210.6

TABLE VII
Bondarenko model ($\bar{v}$: the average number of Newton iterations used for solving the global system per timestep)

C-BDF2 for Bondarenko					
Δx (cm)	Δt (ms)	CPU (secs)	$\bar{v}$	CV (cm/s)	APD (ms)
0.01	0.01	71.84	0.60	57.6	15.0
0.01	0.005	119.7	0.50	57.6	15.0
0.01	0.0025	191.6	0.48	57.7	15.0