Computational and Numerical Methods for the Efficient and Accurate Solution of the Bidomain Equations

J.P. Whiteley

Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford OX1 3QD, UK, Jonathan.Whiteley@comlab.ox.ac.uk

Summary. Two previously published algorithms for solving the bidomain equations are combined to yield an algorithm that efficiently computes an accurate numerical solution of the bidomain equations. The first of these algorithms utilises the multiscale nature of the governing equations by solving the more rapid processes at a much higher resolution than the slower processes, and is ideal for use when a steep action potential wavefront is propagating across tissue. The second algorithm is suitable when no fast processes are taking place. This combined algorithm results in a threefold increase in computational efficiency over the most efficient algorithm that it is compared to for the simulation presented here.

1 The Bidomain Equations

The bidomain equations, or a simplification known as the monodomain equations, are the most commonly used mathematical model of tissue level cardiac electrophysiology. The bidomain equations may be written as [2]:

$$\chi \left(\mathcal{C}_m \frac{\partial V_m}{\partial t} + I_{\text{ion}}(\mathbf{u}, V_m) \right) - \nabla \cdot (\sigma_i \nabla (V_m + \phi_e)) = I_{sv_i}, \tag{1}$$

$$\nabla \cdot ((\sigma_i + \sigma_e) \nabla \phi_e + \sigma_i \nabla V_m) = I_{sv_e}, \tag{2}$$

$$\frac{\partial \mathbf{u}}{\partial t} = \mathbf{f}(\mathbf{u}, V_m), \tag{3}$$

where $V_m(\mathbf{x}, t)$ is the transmembrane potential, $\phi_e(\mathbf{x}, t)$ is the extracellular potential, $\mathbf{u}(\mathbf{x}, t)$ is a vector containing various gating variables and chemical concentrations, σ_i is the intracellular conductivity tensor, σ_e is the extracellular conductivity tensor, χ is the surface to volume ratio, $\mathcal{C}_m$ is the membrane capacitance per unit area, I_{ion} is the ionic current, I_{sv_i} is the intracellular stimulus current applied within the tissue volume, and I_{sv_e} is the extracellular stimulus current applied within the tissue volume. Functional forms for I_{ion} and $\mathbf{f}$ are prescribed by an electrophysiological cell model – see [3] for a collection of these.

A.D. Fitt et al. (eds.), *Progress in Industrial Mathematics at ECMI 2008*, Mathematics in Industry 15, DOI 10.1007/978-3-642-12110-4_57,

Boundary conditions are required for (1) and (2) – these are given by

$$\mathbf{n} \cdot (\sigma_i \nabla (V_m + \phi_e)) = I_{sa_i}, \qquad \mathbf{n} \cdot (\sigma_e \nabla \phi_e) = I_{sa_e}, \tag{4}$$

where $\mathbf{n}$ is the outward pointing unit normal vector to the tissue, I_{sa_i} is the intracellular stimulus current applied across the boundary, and I_{sa_e} is the extracellular stimulus current applied across the boundary. The system (1)–(3) subject to boundary conditions (4) is then closed by specifying initial conditions for V_m and $\mathbf{u}$. We note that ϕ_e is only required to be defined up to an additive constant.

Under certain conditions, ϕ_e may be eliminated from (1) and (2) – the resulting equations are known as the monodomain equations [2]. The techniques described here may be applied to the monodomain equations as well as the bidomain equations.

2 Solving the Bidomain Equations Numerically

Computing an accurate solution of the bidomain equations on a realistic three-dimensional computational geometry is a significant computational challenge. This is mainly due to the multiscale nature of the problem – if one computational mesh is used for all dependent variables then this mesh must be sufficiently fine that it captures the fastest processes. Slower processes are then computed at a much higher resolution than is needed. A further complication caused by the multiscale nature of the problem is numerical stability: multiscale processes result in stiff differential equations, and the numerical solution of stiff equations is notoriously prone to numerical instabilities [1].

We begin the description of our numerical algorithm by describing a semi-implicit numerical algorithm for solving the bidomain equations that has good stability properties. We then explain how this algorithm may be adapted to efficiently handle the multiscale processes that are observed when an action potential propagates across cardiac tissue. Finally, we discuss how both the original algorithm and the multiscale modification to this algorithm may be combined to increase the efficiency of simulations without compromising on accuracy.

2.1 The Basic Numerical Algorithm

The semi-implicit algorithm on which this work is based is described in [5] – we only describe it briefly here. The dependent variables V_m, ϕ_e, $\mathbf{u}$ are calculated a collection of discrete times $t_0, t_1, \ldots, t_N$. We write

$$V_m^n(\mathbf{x}) = V_m(\mathbf{x}, t_n), \qquad \phi_e^n(\mathbf{x}) = \phi_e(\mathbf{x}, t_n), \qquad \mathbf{u}^n(\mathbf{x}) = \mathbf{u}(\mathbf{x}, t_n).$$

We discretise the partial differential equations (1) and (2) by treating the conduction terms implicitly and the reaction terms explicitly. This results in the following discretisation in time:

$$\frac{\chi \mathcal{C}_m}{\Delta t_n} V_m^n - \nabla \cdot (\sigma_i \nabla (V_m^n + \phi_e^n)) = \frac{\chi \mathcal{C}_m}{\Delta t_n} V_m^{n-1} + I_{sv_i} - \chi I_{\text{ion}}(V_m^{n-1}, \mathbf{u}^{n-1}), \tag{5}$$

$$\nabla \cdot ((\sigma_i + \sigma_e) \nabla \phi_e^n + \sigma_i \nabla V_m^n) = I_{sv_e}, \tag{6}$$

where $\Delta t_n = t_n - t_{n-1}$. Equations (5) and (6) may be discretised in space using the finite difference method or finite element method, yielding a matrix equation that must be solved on each timestep:

$$A \begin{pmatrix} \mathbf{V_m^n} \\ \boldsymbol{\phi}_\mathbf{e}^\mathbf{n} \end{pmatrix} = \begin{pmatrix} \mathbf{b_v} \\ \mathbf{b_e} \end{pmatrix}, \tag{7}$$

where A is a matrix arising from the discretisation of (5), (6), $\mathbf{V_m^n}$ is a vector of unknowns that arise from the discretisation of V_m^n, $\boldsymbol{\phi}_\mathbf{e}^\mathbf{n}$ is a vector of unknowns that arise from the discretisation of ϕ_e^n, $\mathbf{b_v}$ arises from the right-hand-side of (5), and $\mathbf{b_e}$ arises from the right-hand-side of (6). Note that A is the same on every timestep, and so this matrix need only be computed once at the start of the simulation.

Having solved (7) to compute $\mathbf{V_m^n}$ and $\boldsymbol{\phi}_\mathbf{e}^\mathbf{n}$, we then solve the ordinary differential equations given by (3) using the backward Euler method to ensure stability [1].

2.2 The Multiscale Algorithm

We now briefly describe the multiscale modification of the semi-implicit algorithm described in Sect. 2.1. For more details see [7]. The key to this algorithm is the use of two meshes as shown in Fig. 1. The fine mesh – denoted by thin lines has a nodal spacing h_{fast}, whilst the coarse mesh – denoted by thick lines – has a nodal spacing h_{slow} Rapidly varying quantities are computed using the fine mesh shown in this figure, whilst other variables are computed using the coarser mesh. When required, the variables calculated on the coarse mesh may be interpolated onto the fine mesh.

When using many cardiac electrophysiological models – including the model used in our simulations [4] – the most rapidly varying quantities are those directly related to the fast sodium current I_{Na}, namely the transmembrane potential, V_m, the extracellular potential ϕ_e, the sodium m-gate and the sodium h-gate. We therefore compute these variables on the fine mesh, and all other variables on the coarse mesh. Splitting the right-hand-side of (5) and (6) into quantities computed on the fine mesh and quantities computed on the coarse mesh we have

$$\frac{\chi \mathcal{C}_m}{\Delta t_n} V_m^n - \nabla \cdot (\sigma_i \nabla (V_m^n + \phi_e^n)) = \Gamma_1 + \Gamma_2, \tag{8}$$

$$\nabla \cdot ((\sigma_i + \sigma_e) \nabla \phi_e^n + \sigma_i \nabla V_m^n) = \Gamma_3, \tag{9}$$

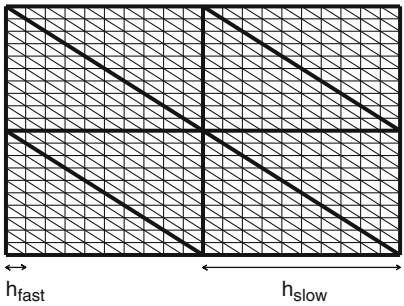

Fig. 1. The two meshes used in the multiscale algorithm. The fine mesh is denoted by *thin lines*, the coarse mesh is denoted by *thick lines*

where

$$\begin{aligned}\Gamma_1 &= \frac{\chi \mathcal{C}_m}{\Delta t_n} V_m^{n-1} + I_{sv_i} - \chi I_{Na},\\ \Gamma_2 &= -\chi(I_{\text{ion}} - I_{Na}),\\ \Gamma_3 &= I_{sv_e}.\end{aligned}$$

Note that Γ_1 and Γ_3 are fast processes, and Γ_2 is a slow process. This allows us to write (7) as

$$A \begin{pmatrix} \mathbf{V_m^n} \\ \boldsymbol{\phi}_\mathbf{e}^\mathbf{n} \end{pmatrix} = \begin{pmatrix} \mathbf{b_v^{fast}} \\ \mathbf{b_e^{fast}} \end{pmatrix} + \begin{pmatrix} \mathbf{b_v^{slow}} \\ \mathbf{0} \end{pmatrix}, \tag{10}$$

where $\mathbf{b_v^{fast}}$ arises from Γ_1, $\mathbf{b_v^{slow}}$ arises from Γ_2 and $\mathbf{b_e^{fast}}$ arises from Γ_3. We then compute the right-hand-side of (10) on each timestep as follows. $\mathbf{b_v^{fast}}$ and $\mathbf{b_e^{fast}}$ are computed as usual using the fine mesh shown in Fig. 1. The quantities required to calculate $\mathbf{b_v^{slow}}$ are calculated at the nodes of the coarse mesh and then interpolated onto the fine mesh. Equation (10) is solved using a timestep Δt_{fast}. However, as the quantities included in $\mathbf{b_v^{slow}}$ vary on a slower timescale, this vector is updated less frequently using a timestep Δt_{slow}. This results in a significant computational saving – computing $\mathbf{b_v^{slow}}$ is generally the most computationally expensive part of the right-hand-side of (10), and so computing it less often will generate a significant saving in volume of computing.

We now turn our attention to computing the numerical solution of the ordinary differential equations given by (3). Fortunately we only have to compute two of these – those for the sodium m-gate and the sodium h-gate – at each node of the fine mesh. All other quantities are only required at the nodes of the coarse mesh, and so the ordinary differential equations representing these variables need only be solved at the nodes of the coarse mesh. We see in Fig. 1 that there are many fewer nodes in the coarse mesh, thus allowing yet another significant computational saving.

2.3 Combining the Algorithms

The multiscale algorithm described in Sect. 2.2 has been shown to give an increase in computational efficiency of two orders of magnitude with negligible loss of accuracy for a simulation of an action potential wavefront travelling across tissue [7]. However, in many simulations the propagation of the steep action potential wavefront occupies only a small portion of the whole simulation. This has been utilised in [6] where, in the absence of a propagating action potential wavefront, the whole problem is solved with all variables computed on the coarse mesh shown in Fig. 1 and a longer timestep Δt_{slow} using the semi-implicit algorithm described in Sect. 2.1. In this study we combine these algorithms – we use the multiscale algorithm described in Sect. 2.2 when the steep action potential wavefront is propagating across the tissue, and then switch to using the semi-implicit algorithm described in Sect. 2.1 on the coarse mesh shown in Fig. 1 with timestep Δt_{slow} at other times. In the next section we demonstrate the performance of this combined algorithm.

3 Computational Results

In this section we verify the accuracy and efficiency of the algorithm described in Sect. 2.3.

3.1 Description of Simulations

We perform an identical simulation to that used in an earlier study [7]. A square occupying the region $0 < x, y < 20\,\text{mm}$, with fibres running perpendicular to the x-axis, was stimulated at one corner at time $t = 0.001\,\text{s}$. A period of time $0 < t < 0.35\,\text{s}$ was simulated. Intracellular conductivities are $0.13\,\text{mS}\,\text{mm}^{-1}$ along the fibre and $0.026\,\text{mS}\,\text{mm}^{-1}$ perpendicular to the fibre. Extracellular conductivities are $0.13\,\text{mS}\,\text{mm}^{-1}$ along the fibre and $0.065\ \text{mS}\ \text{mm}^{-1}$ perpendicular to the fibre. In common with [7] we use $h_{\text{fast}} = 0.1\,\text{mm}$, $h_{\text{slow}} = 1.0\,\text{mm}$, $\Delta t_{\text{fast}} = 0.1\,\text{ms}$, $\Delta t_{\text{slow}} = 1.0\,\text{ms}$. We use the multiscale algorithm described in Sect. 2.2 initially. In common with [6] we switch to solving the equations on the coarse mesh with timestep Δt_{slow} when the fast sodium current has dropped below 10 pA pF^{-1} at all points in the computational domain. V_m was recorded at the central point of the square.

To assess the accuracy and efficiency of the algorithm described in Sect. 2.3 the simulation described above was repeated: (a) using the basic algorithm described in Sect. 2.1, the fine mesh and timestep Δt_{fast}; and (b) the multiscale algorithm described in Sect. 2.2.

3.2 Results of Simulations

The action potential at the central point of the square calculated using all three algorithms is shown in Fig. 2. We see that the plots are visually indistinguishable, thus verifying the accuracy of the combined algorithm. We now turn

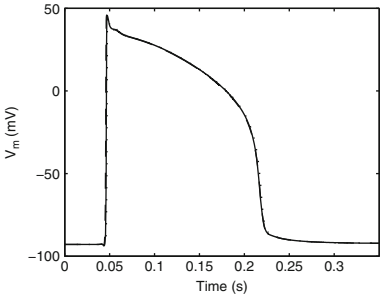

Fig. 2. The action potential at the central point of the square calculated using all three algorithms

our attention to the efficiency of the three algorithms. The basic algorithm using a fine mesh required 13,784 s of computation time. The multiscale algorithm required 786 s of computation time. The combined algorithm required 253 s of computation time. We therefore conclude that, for the simulation presented here, the combined algorithm described in Sect. 2.3 allows an increase in computational efficiency by a factor of roughly 3.

4 Discussion

Two previously published algorithms have been combined, allowing a computational speedup by a factor of around three for the simulation considered here. Although a physiologically detailed electrophysiological cell model [4] was used the geometry was very simple, being a two-dimensional square with regular fibre orientation. Future work is currently being directed towards implementing this algorithm in a realistic, irregular three-dimensional cardiac geometry with irregular fibre orientation.

References

1. Iserles, A.: A First Course in the Numerical Analysis of Differential Equations. Cambridge University Press, Cambridge (1996)
2. Keener, J.P., Sneyd, J.: Mathematical Physiology. Springer, New York (1998)
3. Nickerson, D.P.: Modelling Cardiac Electro-mechanics: From cellML to the Whole Heart. PhD Thesis, University of Auckland (2004)
4. Noble, D., Varghese, A., Kohl, P., Noble, P.: Can. J. Cardiol. **14**, 123–134 (1998)
5. Whiteley, J.P.: IEEE Trans. Biomed. Eng. **53**, 2139–2147 (2006)
6. Whiteley, J.P.: Ann. Biomed. Eng. **35**, 1510–1520 (2007)
7. Whiteley, J.P.: Ann. Biomed. Eng. **36**, 1398–1408 (2008)