Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational and Applied Mathematics 169 (2004) 431–458

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

www.elsevier.com/locate/cam

# A memory-efficient finite element method for systems of reaction–diffusion equations with non-smooth forcing☆

Alexander L. Hanhart[a], Matthias K. Gobbert[a], Leighton T. Izu[b]

[a] *Department of Mathematics and Statistics, University of Maryland, Baltimore County, 1000 Hilltop Circle, Baltimore, MD 21250, USA*

[b] *Department of Medicine, Division of Cardiology, University of Maryland, Baltimore, 22 S. Greene St., Baltimore, MD 21201, USA*

Received 5 October 2003; received in revised form 4 November 2003

**Abstract**

The release of calcium ions in a human heart cell is modeled by a system of reaction–diffusion equations, which describe the interaction of the chemical species and the effects of various cell processes on them. The release is modeled by a forcing term in the calcium equation that involves a superposition of many Dirac delta functions in space; such a nonsmooth right-hand side leads to divergence for many numerical methods. The calcium ions enter the cell at a large number of regularly spaced points throughout the cell; to resolve those points adequately for a cell with realistic three-dimensional dimensions, an extremely fine spatial mesh is needed. A finite element method is developed that addresses the two crucial issues for this and similar applications: Convergence of the method is demonstrated in extension of the classical theory that does not apply to nonsmooth forcing functions like the Dirac delta function; and the memory usage of the method is optimal and thus allows for extremely fine three-dimensional meshes with many millions of degrees of freedom, already on a serial computer. Additionally, a coarse-grained parallel implementation of the algorithm allows for the solution on meshes with yet finer resolution than possible in serial.
© 2004 Elsevier B.V. All rights reserved.

*MSC:* 35K57; 65F10; 65M60; 65Y05; 92C30

*Keywords:* Reaction–diffusion equation; Non-smooth data; Galerkin method; Matrix-free iterative method; Cluster computing

☆ The computational hardware used in this study was provided by the University of Maryland, Baltimore County and the National Science Foundation under grant DMS-9805547.

*E-mail address:* gobbert@math.umbc.edu (M.K. Gobbert).

0377-0427/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2003.12.035

## 1. Introduction

A mathematical model for the concentration of calcium ions in a human heart cell has been developed [16–18]. This model comprises three coupled nonlinear reaction-diffusion equations throughout the interior of the cell $\Omega \subset \mathbb{R}^3$, which describe the concentrations of calcium ions ($C$), a fluorescent calcium indicator ($F$), and the endogenous calcium buffer ($B$):

$$\frac{\partial C}{\partial t} = \nabla \cdot (D_C \nabla C) + R_F + R_B - J_{\text{pump}} + J_{\text{leak}} + \sigma, \tag{1}$$

$$\frac{\partial F}{\partial t} = \nabla \cdot (D_F \nabla F) + R_F, \tag{2}$$

$$\frac{\partial B}{\partial t} = R_B. \tag{3}$$

Let $\mathbf{x} := (x, y, z)^{\mathrm{T}}$ denote a point in the three-dimensional domain $\Omega \subset \mathbb{R}^3$, and we want to determine the molar concentrations $C(\mathbf{x}, t)$, $F(\mathbf{x}, t)$, $B(\mathbf{x}, t)$ for $\mathbf{x} \in \Omega$ and $0 \leqslant t \leqslant t_{\text{fin}}$. Here, $D_C, D_F \in \mathbb{R}^{3\times 3}$ are constant diagonal matrices, whose diagonal entries describe the diffusivity of $C$ and $F$, respectively, in the $x$-, $y$-, and $z$-directions; the diffusivity of $B$ is so slow that its diffusion can be neglected. The reactions between the species are given by the rate functions $R_F(C, F)$ and $R_B(C, B)$, and $J_{\text{pump}}(C)$ and $J_{\text{leak}}$ are additional forcing terms in the model. The function $\sigma \equiv \sigma(C, \mathbf{x}, t; T_{\text{open}})$ is the most crucial part of the model. It models the inflow of calcium ions into the cell and will be described in more detail below after the domain is specified. To form a complete problem statement, we impose no-flux conditions on the boundary and let the initial conditions be given by constants $C_0$, $F_0$, $B_0$ uniformly throughout the domain at time $t = 0$. For background information, see [4], and for more detailed information on the model and its background, see [16–18] and the references therein. The appendix of this paper gives some additional information, all formulas used, and a full specification of all parameters values.

The goal of this work is to provide realistic three-dimensional transient numerical simulations of this model. The particular model is considered because it is a relevant application in its own right, but more importantly the structure of the equations and the desire for high-resolution discretizations also arise in many other possible application problems. There are two main problems that need to be overcome:

- An extremely fine grid is required to resolve the relevant features of the three-dimensional domain. This necessitates a method specially designed to be as memory-efficient as possible. We will explain below how this can be done and demonstrate that a problem with, for instance, over 11 million degrees of freedom can be solved on a serial machine. Using coarse-grained parallelism, even a problem with over 22 million degrees of freedom can be solved.
- The function $\sigma$ involves Dirac delta functions in the spatial variables. The finite element method can be formally applied to this problem, but standard convergence theory cannot be applied any more. It will be discussed below why a convergence order of $\frac{1}{2}$ can be expected and it will be demonstrated that the method agrees with this theoretical prediction. For other methods, e.g., finite difference or finite volume, the regularity requirements are more stringent, and divergence has actually been observed in numerical tests.

These issues and their resolution will be explained in more detail presently.

### 1.1. The matrix-free implementation of all linear solves

The cell is reasonably approximated as the three-dimensional domain $\Omega=(-32,32)\times(-6.4,6.4)\times(-6.4,6.4)$ with length units of μm. Calcium ions are released into the domain at a uniform grid of discrete positions, known as calcium release units (CRUs), which are distributed along a rectangular lattice interior to the cell. The rectangular array of CRUs is centered in the interior of the cell such that one CRU lies at the center of the domain, $(0,0,0)$. The remaining CRUs are distributed uniformly throughout the cell interior with spacings $\Delta x_s = 2.0$ μm and $\Delta y_s = \Delta z_s = 0.8$ μm.

The probability that calcium will be released from a CRU depends on the calcium concentration at that CRU. When a CRU releases calcium (it 'fires'), the local concentration of calcium ions increases sharply and briefly. By using a fluorescent calcium indicator, the firing of a CRU appears as a brief and localized increase in fluorescence called a calcium 'spark'. The calcium that is released diffuses and raises the probability of release at neighboring sites. As a consequence, CRUs begin to release calcium throughout the cell and the release self-organizes into a wave of increasing concentration.

To simulate the spontaneous release of calcium from any one of the CRUs and to capture the resulting wave, it is vital that the domain be resolved in a way such that the CRUs are nodes on the numerical mesh and that there are sufficiently many points between the CRUs to guarantee a proper resolution of the diffusion. To get a feel for how fine a mesh will be required for realistic simulations, consider the following: The CRUs are located 2 μm apart in the $x$-direction, and 0.8 μm in the $y$- and $z$-directions. Hence, the lattice of CRUs forms a $32 \times 16 \times 16$ grid in the domain $\Omega = (-32,32) \times (-6.4,6.4) \times (-6.4,6.4)$. In order to use at least 8 mesh points for each CRU spacing in each direction, let's say we wish to use a spatial grid of size $448 \times 128 \times 128$ [13]. Considering that the three variables $C$, $F$, and $B$ need to be stored at each node, we need to store over 22 million degrees of freedom (DOF). If the variables are stored in double-precision, the total memory requirement to store *one* complete solution (comprising all three species concentrations) at a *single* time step is approximately 168 MB. No matter which solution method is used, a few auxiliary variables will need to be stored, easily bringing up the total memory required to several multiples of 168 MB. Even on a computer with extended memory, e.g., 1 GB, it will become challenging to accommodate these variables. Moreover, a conventional numerical method for this type of problem will use implicit time-stepping and will require the assembly and storage of a system matrix for the linear solve required at each time step. This is clearly *not* feasible: A system matrix for a finite element discretization in three dimensions, as we will use, has $3 \times 3 \times 3 = 27$ nonzero diagonals, hence the matrix might require up to 27 times 168 MB of memory, even in sparse storage mode (only the nonzero elements are stored)!

We have overcome this problem by designing a specialized numerical method that takes advantage of the particular properties of the problem: (a) The regular distribution of CRUs allows for the discretization of the domain by a uniform mesh. (b) Since the diffusivities in (1)–(3) are constant and by evaluating the right-hand side terms at the old time step, the system matrix can be precomputed analytically. (c) By selecting the conjugate gradient method to solve the linear system that results from the implicit time discretization, only one matrix–vector product of the system matrix with a vector is needed. Since all coefficients of this matrix are precomputed analytically, we can supply a function for this matrix–vector product that never requires the explicit assembly of the matrix; this is called a matrix-free method and is obviously the most memory-efficient implementation possible, as no matrix is ever stored.

## 1.2. Convergence of the finite element method

The most crucial term of the model (1)–(3) is the function $\sigma(C, \mathbf{x}, t; T_{\text{open}})$ that describes the release of calcium at the calcium release units (CRUs). Let the set of CRU points in a cell be denoted by $X_{\text{CRU}}$. For any $\hat{\mathbf{x}} \in X_{\text{CRU}}$, the occurrence of a calcium spark event is correlated with a high local concentration of calcium ions. This is expressed in the model by a scaled Dirac delta function in space, $\hat{\sigma}\delta(\mathbf{x} - \hat{\mathbf{x}})$, where the scaling constant $\hat{\sigma}$ is the molar flux constant that gives the amount of calcium ions released into the cell per unit time at a CRU. Here, we use the short-hand notation $\delta(\mathbf{x}) \equiv \delta(x)\delta(y)\delta(z)$ for the Dirac delta function in three dimensions.

By the summation of sparks at all possible locations $\hat{\mathbf{x}} \in X_{\text{CRU}}$, the function

$$\sigma(C, \mathbf{x}, t; T_{\text{open}}) = \sum_{\hat{\mathbf{x}} \in X_{\text{CRU}}} \hat{\sigma} S_{\hat{\mathbf{x}}}(C, t; T_{\text{open}}) \delta(\mathbf{x} - \hat{\mathbf{x}}) \tag{4}$$

gives the superposition of all spark events active at time $t$. The function $S_{\hat{\mathbf{x}}}(C, t; T_{\text{open}})$ is an indicator function in time that switches a CRU on for a time period of $T_{\text{open}}$ depending on a probabilistic model. See the appendix for the details of this switching model. Since each $S_{\hat{\mathbf{x}}}$ is associated with a delta function centered at the CRU point $\hat{\mathbf{x}}$, the delta functions at $\hat{\mathbf{x}}$ affect the solution only if $S_{\hat{\mathbf{x}}} = 1$. In addition, notice that in our notation, $\sigma(C, \mathbf{x}, t; T_{\text{open}}) = 0$ unless $\mathbf{x}$ is a CRU.

The problem (1)–(3), for which a numerical method needs to be developed, is a system of three nonlinear reaction-diffusion equations. This is a well-understood problem and methods exist for its solution provided all data have sufficient regularity. For instance, theory exists for the application of the finite element method, provided the right-hand side function is an element of $L_2(\Omega)$, see, e.g., [5,12,14,22,25]. However, the model under consideration here includes a (superposition of many) Dirac delta function(s) in the forcing term on the right-hand side of the reaction-diffusion equation for the calcium concentration, and $\delta(\mathbf{x}) \equiv \delta(x)\delta(y)\delta(z)$ is certainly not in $L_2$.

We observe that the Dirac delta function can be formally evaluated, if the problem is posed in its weak form involving integrations of all terms over the domain $\Omega$. This motivates the use of a finite element method (FEM), which is based on this weak formulation of the problem. We stress that this approach is formal in nature at this point, and the conventional theory cannot be used to guarantee the convergence of the numerical solution in the usual finite element spaces. However, an estimate can be derived in the following way: In three dimensions, the Sobolev space $H^{3/2+\varepsilon}$ is continuously embedded in $C^0$ for any $\varepsilon > 0$ [1]. Consider the integral definition of the Dirac delta function as a functional over this space. Then through the dual embedding, we can approximate the space to which $\delta(\mathbf{x})$ belongs to as $H^{-3/2-\varepsilon}$, namely, the dual of $H^{3/2+\varepsilon}$. From [25], we can expect that the convergence order of the FEM is $h^{k+2}$, if linear finite elements are used and the right-hand side function lies in $H^k$, where $h$ is the largest side length of an element. So, if our method converges, we expect to see a convergence order of $2 - \frac{3}{2} = \frac{1}{2}$. This convergence order will be shown in the results, in agreement with these theoretical considerations, thus justifying the use of the finite element method with linear basis functions for this problem.

Notice that other methods, e.g., finite difference methods that approximate the equation directly, cannot even be formally stated for (1)–(3), because the Dirac delta function cannot be evaluated at a point. This leads to using integral forms of the problem, in which case we can formally evaluate the term involving the Dirac delta function. A finite volume method, based on integrating (1)–(3), can then be formulated formally, but our test calculations have in fact demonstrated *divergence* of the numerical solution. This explains our insistence of using the finite element method, because it is the only numerical method that possesses the combination of features: (i) A theoretical rationale can be given why convergence can be expected in the face of the Dirac delta functions and this convergence can be actually demonstrated, and (ii) the structure of the method allows for a highly memory-efficient implementation that are vital to achieve the high resolution of the spatial discretization required for this application.

### 1.3. *Outline of the paper*

This paper shows concretely how to design the most memory-efficient implementation of a finite element method and demonstrates that the method convergences in the face of a highly nonsmooth right-hand side. These techniques are useful beyond this particular application problem, hence Section 2 presents the specialized finite element method designed to address the issue of memory requirements on a generic scalar reaction-diffusion equation. Section 3 contains three different types of numerical results: In Section 3.1, it will be demonstrated that the specialized FEM gives the predicted order of convergence for a scalar equation, both in the case of a smooth right-hand side and a Dirac delta function; these results validate the underlying approach using the finite element method and are independent from the actual model problem. In Section 3.2, results show convergence of the method also for simulations of the full three-species model and further results illustrate that the simulator can exhibit the wave of increasing calcium throughout the domain. Finally, Section 3.3 shows how a coarse-grained parallelization using as many parallel processors as there are chemical species allows for the solution of problems that could not be solved on a single processor. Section 4 summarizes our conclusions. Appendix A contains the complete definition of the model and the values of all physical coefficients used in the simulations.

## 2. Numerical method

In order to numerically simulate the calcium spark model (1)–(3), a numerical method must be designed that is very efficient in memory use. The goal is to use properties of the model to design a specialized numerical method for this model. The uniform rectangular CRU lattice naturally induces a regular numerical mesh. The model also uses constant diffusion coefficients. Using a finite element method (FEM) that takes advantage of these properties (constant coefficients, regular mesh) will allow for the analytic computation of the mass and stiffness matrices. This analytic computation of the mass and stiffness matrices uses global basis functions directly and is unusual compared to many other applications of finite elements, hence it is described in more-than-usual detail below. Additionally lagging all nonlinear terms in time yields system matrices for which matrix–vector products can be designed without an explicitly stored system matrix. Such a matrix-free method will dramatically reduce the memory requirements of the method, thereby making realistic simulations feasible.

### 2.1. *Semi-discrete formulation*

The domain is discretized by a mesh of brick elements generated by dividing the $x$-, $y$-, and $z$-directions into equidistant nodes. Using $N_x$, $N_y$, and $N_z$ points in the $x$-, $y$-, and $z$-directions, respectively, gives a mesh with a total of $N = N_x N_y N_z$ nodes. This discretization is done in such a way that the uniformly spaced CRUs become nodes of the mesh, that is, the nodes $\mathbf{x}_j$ in $X_N := \{\mathbf{x}_j : j = 1,\dots,N\}$ are chosen such that $X_{\mathrm{CRU}} \subset X_N$.

On this uniform mesh, define the family of $N$ tri-linear nodal basis functions $\varphi_i : \Omega \to \mathbb{R}$, $i = 1,\dots,N$, which are affine functions in each component of $\mathbf{x} = (x, y, z)^{\mathrm{T}}$ and satisfy $\varphi_i(\mathbf{x}_j) = 1$ if $i = j$ and $\varphi_i(\mathbf{x}_j) = 0$ otherwise, for all nodes of the mesh $\mathbf{x}_j \in X_N$, $j = 1,\dots,N$. The goal will be to approximate each of $C(\mathbf{x},t)$, $F(\mathbf{x},t)$, and $B(\mathbf{x},t)$ by functions in the span of $\{\varphi_i\}_{i=1}^N$. Since Eqs. (1)–(3) all share the same form, we will derive the finite element discretization using a generic reaction-diffusion equation.

Consider the parabolic prototype problem

$$\begin{aligned} &\frac{\partial u}{\partial t} = \nabla\cdot(D\nabla u) + f && \text{in } \Omega,\\ &\nu\cdot(D\nabla u) = 0 && \text{on } \partial\Omega,\\ &u = u_0 && \text{at } t = 0, \end{aligned} \tag{5}$$

where $\nu = \nu(\mathbf{x})$ denotes the outward unit normal vector at $\mathbf{x}\in\partial\Omega$. Here, $D \in \mathbb{R}^{3\times 3}$ is assumed to be diagonal with positive entries. For a test function $v$, we have from (5)

$$\int_\Omega \frac{\partial u}{\partial t} v \,\mathrm{d}\mathbf{x} - \int_\Omega \nabla\cdot(D\nabla u) v \,\mathrm{d}\mathbf{x} = \int_\Omega v f \,\mathrm{d}\mathbf{x}.$$

Apply Green's theorem and use the no-flux boundary conditions to obtain the weak formulation of (5): Find $u \in H^1(\Omega)$ so that

$$\int_\Omega \frac{\partial u}{\partial t} v \,\mathrm{d}\mathbf{x} + \int_\Omega \nabla v \cdot D\nabla u \,\mathrm{d}\mathbf{x} = \int_\Omega v f \,\mathrm{d}\mathbf{x} \quad \forall v \in H^1(\Omega), \tag{6}$$

with notation as in [25]. The goal will be to approximate the solution $u(\mathbf{x},t)$ of (5) by the finite element solution $u_h(\mathbf{x},t)$:

$$u(\mathbf{x},t) \approx u_h(\mathbf{x},t) = \sum_{j=1}^N U_j(t)\varphi_j(\mathbf{x}). \tag{7}$$

Substituting (7) into (6) and choosing $v = \varphi_i$, $i = 1,\dots,N$, will give the $N$ equations

$$\sum_{j=1}^N \underbrace{\int_\Omega \varphi_i\varphi_j \,\mathrm{d}\mathbf{x}}_{M_{ij}} \frac{\mathrm{d}U_j}{\mathrm{d}t} + \sum_{j=1}^N \underbrace{\int_\Omega \nabla\varphi_i \cdot D\nabla\varphi_j \,\mathrm{d}\mathbf{x}}_{K_{ij}} U_j = \underbrace{\int_\Omega \varphi_i f \,\mathrm{d}\mathbf{x}}_{\hat{F}_i}, \quad i = 1,\dots,N.$$

for the solution coefficients $U_j(t)$, $j = 1,\dots,N$, which constitute approximations for the solution $u(\mathbf{x},t)$ at $\mathbf{x} = \mathbf{x}_j$. Here, we introduce the standard definition of the mass matrix $M \in \mathbb{R}^{N\times N}$, whose elements are defined as $M_{ij} = \int_\Omega \varphi_i\varphi_j \,\mathrm{d}\mathbf{x}$, and the stiffness matrix $K \in \mathbb{R}^{N\times N}$ with elements defined by $K_{ij} = \int_\Omega \nabla\varphi_i \cdot D\nabla\varphi_j \,\mathrm{d}\mathbf{x}$. Additionally, we define the right-hand side vector $\hat{F} \in \mathbb{R}^N$ temporarily with coefficients $\hat{F}_i = \int_\Omega \varphi_i f \,\mathrm{d}\mathbf{x}$. Using this notation, the semi-discrete system of ordinary differential equations can be stated in vector form as

$$M\frac{\mathrm{d}U}{\mathrm{d}t} + KU = \hat{F} \tag{8}$$

for the vector of coefficient functions $U = (U_j(t))$. Notice that $M$ and $K$ are in fact constant matrices.

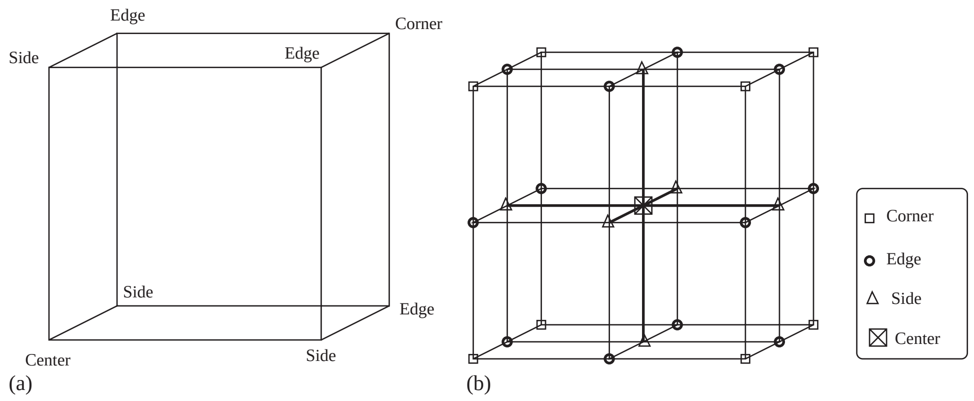

Fig. 1. (a) Reference element $\hat{\Omega}$ with corners labeled. (b) Support $\Omega_i$ of a basis function $\varphi_i$ associated with an interior node $\mathbf{x}_i \notin \partial\Omega$.

We will assume that only the nodal values of $f$ are available, that is, we have a vector $F=(F_j(t))$ where $F_j(t)=f(\mathbf{x}_j,t)$. Therefore, a projection of $f$ from (5) into the space spanned by $\varphi_i$ will have the form

$$f(\mathbf{x},t)=\sum_{j=1}^{N} F_j(t)\varphi_j(\mathbf{x}). \tag{9}$$

The components of $\hat{F}$ in (8) are given by

$$\hat{F}_i=\int_\Omega \varphi_i f \,\mathrm{d}\mathbf{x}=\sum_{j=1}^{N} F_j(t)\left(\int_\Omega \varphi_i\varphi_j \,\mathrm{d}\mathbf{x}\right)=\sum_{j=1}^{N} M_{ij}F_j, \tag{10}$$

and we can consider the slightly modified system

$$M\frac{\mathrm{d}U}{\mathrm{d}t}+KU=MF, \tag{11}$$

where $F=(F_j)$ denotes the vector of nodal values $F_j=f(\mathbf{x}_j,t)$.

### *2.1.1. Determining the mass and stiffness matrices*

The regular mesh, constant coefficient matrix $D$, and the choice of nodal tri-linear basis functions allow the analytical pre-computation of the mass and stiffness matrices in closed-form. This derivation uses global basis functions explicitly, even though the actual integration are carried out on a reference element and naturally arising symmetries are used. We start by explaining the computation of the mass matrix $M$, followed by the stiffness matrix $K$.

Consider first a reference element $\hat{\Omega}\subset\mathbb{R}^3$ given by a parallelepiped with one corner at the origin and the sides extending in the positive axis directions at lengths $\Delta x$, $\Delta y$, and $\Delta z$. Fig. 1(a) shows an example and labels each of the eight corner points. Let the local basis function associated with the node 'center' in Fig. 1(a) be defined as

$$\varphi_{\text{center}}(x,y,z)=\frac{\Delta x-x}{\Delta x}\frac{\Delta y-y}{\Delta y}\frac{\Delta z-z}{\Delta z}.$$

If $\varphi$ is associated with any one of the nodes labeled 'center', 'side', 'edge', or 'corner' in Fig. 1(a), we can compute the integral $\int_{\hat{\Omega}} \varphi_{\text{center}}\varphi \, d\mathbf{x}$ explicitly. To simplify the calculations, note that $\int_{\hat{\Omega}} \varphi_{\text{center}}\varphi_{\text{side}} \, d\mathbf{x}$ will be the same for all three points labeled 'side' in Fig. 1(a), using either of the basis functions

$$\varphi_{\text{side}}(x,y,z) = \frac{x}{\Delta x}\,\frac{\Delta y - y}{\Delta y}\,\frac{\Delta z - z}{\Delta z}, \quad \varphi_{\text{side}}(x,y,z) = \frac{\Delta x - x}{\Delta x}\,\frac{y}{\Delta y}\,\frac{\Delta z - z}{\Delta z}, \quad \text{and}$$

$$\varphi_{\text{side}}(x,y,z) = \frac{\Delta x - x}{\Delta x}\,\frac{\Delta y - y}{\Delta y}\,\frac{z}{\Delta z}.$$

The same is true for $\int_{\hat{\Omega}} \varphi_{\text{center}}\varphi_{\text{edge}} \, dx$. In summary, we obtain the four integrals over the reference element $\hat{\Omega}$

$$\int_{\hat{\Omega}} \varphi_{\text{center}}\varphi_{\text{center}} \, d\mathbf{x} = \frac{8V}{216},$$

$$\int_{\hat{\Omega}} \varphi_{\text{center}}\varphi_{\text{side}} \, d\mathbf{x} = \frac{4V}{216},$$

$$\int_{\hat{\Omega}} \varphi_{\text{center}}\varphi_{\text{edge}} \, d\mathbf{x} = \frac{2V}{216},$$

$$\int_{\hat{\Omega}} \varphi_{\text{center}}\varphi_{\text{corner}} \, d\mathbf{x} = \frac{V}{216}.$$

Here, $V = \Delta x \Delta y \Delta z$ denotes the volume of the reference element $\hat{\Omega}$.

Based on the computed quantities on the reference element, we can now compute the elements of the mass matrix associated with an interior point of the domain $\mathbf{x}_i \notin \partial\Omega$. From Fig. 1(b), where $\mathbf{x}_i$ lies at the center, it is clear that $\Omega_i$ is the union of eight reference elements $\hat{\Omega}$. Since $\int_{\Omega} \varphi_i\varphi_j \, d\mathbf{x} = \int_{\Omega_i \cap \Omega_j} \varphi_i\varphi_j \, d\mathbf{x}$ and due to symmetry, only the following four forms exist involving a basis function at an interior node $\mathbf{x}_i \notin \partial\Omega$:

$$\int_{\Omega} \varphi_{\text{center}}\varphi_{\text{center}} \, d\mathbf{x} = 8\int_{\hat{\Omega}} \varphi_{\text{center}}\varphi_{\text{center}} \, d\mathbf{x} = \frac{64V}{216},$$

$$\int_{\Omega} \varphi_{\text{center}}\varphi_{\text{side}} \, d\mathbf{x} = 4\int_{\hat{\Omega}} \varphi_{\text{center}}\varphi_{\text{side}} \, d\mathbf{x} = \frac{16V}{216},$$

$$\int_{\Omega} \varphi_{\text{center}}\varphi_{\text{edge}} \, d\mathbf{x} = 2\int_{\hat{\Omega}} \varphi_{\text{center}}\varphi_{\text{edge}} \, d\mathbf{x} = \frac{4V}{216},$$

$$\int_{\Omega} \varphi_{\text{center}}\varphi_{\text{corner}} \, d\mathbf{x} = 1\int_{\hat{\Omega}} \varphi_{\text{center}}\varphi_{\text{corner}} \, d\mathbf{x} = \frac{V}{216},$$

To connect these auxiliary computations to the mass matrix $M$ with elements $M_{ij} = \int_{\Omega} \varphi_i\varphi_j \, d\mathbf{x}$, we define a one-dimensional counting scheme that counts through all $N = N_x N_y N_z$ nodes of the mesh. If $\mathbf{x}_i$ is an interior point of the domain, i.e., $\mathbf{x}_i \notin \partial\Omega$, it lies at the center of a $3 \times 3 \times 3$ array of nodes; Fig. 2 indicates the one-dimensional counting scheme used for the global basis functions $\varphi_i$, $i = 1,\ldots,N$. Using this notation, we can now give the values for all elements of the mass matrix $M$

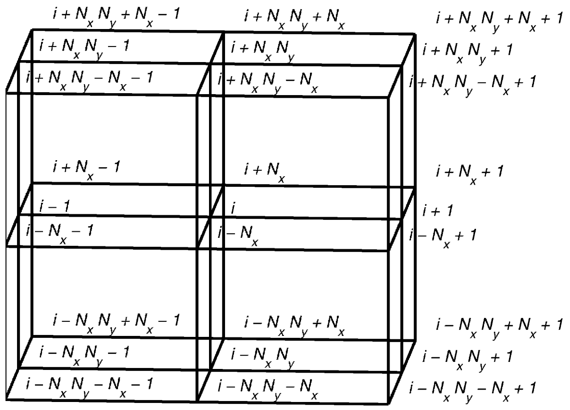

Fig. 2. One-dimensional counting scheme around interior node $\mathbf{x}_i \notin \partial\Omega$.

as follows:

$$M_{ii} = \int_\Omega \varphi_{\text{center}}\varphi_{\text{center}}\,\mathrm{d}\mathbf{x} = 8\int_{\hat\Omega} \varphi_{\text{center}}\varphi_{\text{center}}\,\mathrm{d}\mathbf{x} = \frac{64V}{216}, \tag{12}$$

$$\begin{aligned} M_{ii-1} &= M_{ii+1} = M_{ii-N_x} = M_{ii+N_x} = M_{ii-N_xN_y} = M_{ii+N_xN_y} = \int_\Omega \varphi_{\text{center}}\varphi_{\text{side}}\,\mathrm{d}\mathbf{x} \\ &= 4\int_{\hat\Omega} \varphi_{\text{center}}\varphi_{\text{side}}\,\mathrm{d}\mathbf{x} = \frac{16V}{216}, \end{aligned} \tag{13}$$

$$\begin{aligned} M_{ii-N_x-1} &= M_{ii-N_x+1} = M_{ii+N_x-1} = M_{ii+N_x+1} \\ &= M_{ii-N_xN_y-1} = M_{ii-N_xN_y+1} = M_{ii+N_xN_y-1} = M_{ii+N_xN_y+1} \\ &= M_{ii-N_xN_y-N_x} = M_{ii-N_xN_y+N_x} = M_{ii+N_xN_y-N_x} = M_{ii+N_xN_y+N_x} \\ &= \int_\Omega \varphi_{\text{center}}\varphi_{\text{edge}}\,\mathrm{d}\mathbf{x} = 2\int_{\hat\Omega} \varphi_{\text{center}}\varphi_{\text{edge}}\,\mathrm{d}\mathbf{x} = \frac{4V}{216}, \end{aligned} \tag{14}$$

$$\begin{aligned} M_{ii-N_xN_y-N_x-1} &= M_{ii-N_xN_y-N_x+1} = M_{ii-N_xN_y+N_x-1} = M_{ii-N_xN_y+N_x+1} \\ &= M_{ii+N_xN_y-N_x-1} = M_{ii+N_xN_y-N_x+1} = M_{ii+N_xN_y+N_x-1} = M_{ii+N_xN_y+N_x+1} \\ &= \int_\Omega \varphi_{\text{center}}\varphi_{\text{corner}}\,\mathrm{d}\mathbf{x} = 1\int_{\hat\Omega} \varphi_{\text{center}}\varphi_{\text{corner}}\,\mathrm{d}\mathbf{x} = \frac{V}{216}. \end{aligned} \tag{15}$$

The previous calculations assumed that the node $\mathbf{x}_i$ is an interior node of the domain. If $\mathbf{x}_i \in \partial\Omega$ lies on the boundary of the domain, its support will not consist of eight reference elements any more, as in Fig. 1(b). Rather a node on the face of the boundary would have a support of four reference elements. Algebra shows then that the final results for the coefficients of the mass matrix are scaled by a factor $\frac{1}{2}$ compared to the above equations. Similarly, the results are scaled by $\frac{1}{4}$, if $\mathbf{x}_i$ lies on an edge of the domain, and by $\frac{1}{8}$, if $\mathbf{x}_i$ is a corner.

The stiffness matrix $K$ consists of elements $K_{ij} = \int_\Omega \nabla\varphi_i \cdot (D\nabla\varphi_j)\,\mathrm{d}\mathbf{x}$. $D$ is a $3\times 3$ diagonal matrix of diffusivity coefficients given by $D = \mathrm{diag}(D_x, D_y, D_z)$. As with $M$, an analytic formula can be found for any element of $K$. However, there arise eight distinct formulas, since the integrals $\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{\text{side}})\,\mathrm{d}\mathbf{x}$ and $\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{\text{edge}})\,\mathrm{d}\mathbf{x}$ are not the same for all three 'side' or 'edge' points, respectively, in Fig. 1(a). Rather, let '$x$-side', '$y$-side', and '$z$-side' refer to the side points in the $x$-, $y$-, and $z$-directions given by the coordinates $(\Delta x, 0, 0)$, $(0, \Delta y, 0)$, and $(0, 0, \Delta z)$ in the reference element $\hat{\Omega}$, respectively. Let '$xy$-edge', '$xz$-edge', and '$yz$-edge' refer to the edge points corresponding to $(\Delta x, \Delta y, 0)$, $(\Delta x, 0, \Delta z)$, and $(0, \Delta y, \Delta z)$ in the reference element $\hat{\Omega}$, respectively. Then the following eight integrals have distinct values:

$$\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{\text{center}})\,\mathrm{d}\mathbf{x} = \frac{V}{9}\left[\frac{D_x}{(\Delta x)^2} + \frac{D_y}{(\Delta y)^2} + \frac{D_z}{(\Delta z)^2}\right],$$

$$\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{x\text{-side}})\,\mathrm{d}\mathbf{x} = \frac{V}{18}\left[-2\frac{D_x}{(\Delta x)^2} + \frac{D_y}{(\Delta y)^2} + \frac{D_z}{(\Delta z)^2}\right],$$

$$\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{y\text{-side}})\,\mathrm{d}\mathbf{x} = \frac{V}{18}\left[\frac{D_x}{(\Delta x)^2} - 2\frac{D_y}{(\Delta y)^2} + \frac{D_z}{(\Delta z)^2}\right],$$

$$\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{z\text{-side}})\,\mathrm{d}\mathbf{x} = \frac{V}{18}\left[\frac{D_x}{(\Delta x)^2} + \frac{D_y}{(\Delta y)^2} - 2\frac{D_z}{(\Delta z)^2}\right],$$

$$\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{xy\text{-edge}})\,\mathrm{d}\mathbf{x} = \frac{V}{36}\left[-2\frac{D_x}{(\Delta x)^2} - 2\frac{D_y}{(\Delta y)^2} + \frac{D_z}{(\Delta z)^2}\right],$$

$$\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{xz\text{-edge}})\,\mathrm{d}\mathbf{x} = \frac{V}{36}\left[-2\frac{D_x}{(\Delta x)^2} + \frac{D_y}{(\Delta y)^2} - 2\frac{D_z}{(\Delta z)^2}\right],$$

$$\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{yz\text{-edge}})\,\mathrm{d}\mathbf{x} = \frac{V}{36}\left[\frac{D_x}{(\Delta x)^2} - 2\frac{D_y}{(\Delta y)^2} - 2\frac{D_z}{(\Delta z)^2}\right],$$

$$\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{\text{corner}})\,\mathrm{d}\mathbf{x} = \frac{V}{72}\left[-2\frac{D_x}{(\Delta x)^2} - 2\frac{D_y}{(\Delta y)^2} - 2\frac{D_z}{(\Delta z)^2}\right].$$

Using again the counting scheme from Fig. 2, the elements of the stiffness matrix $K$ are given by the values

$$K_{ii} = 8\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{\text{center}})\,\mathrm{d}\mathbf{x} \tag{16}$$

$$\begin{aligned}
K_{ii-1} &= K_{ii+1} = 4\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{x\text{-side}})\,\mathrm{d}\mathbf{x},\\
K_{ii-N_x} &= K_{ii+N_x} = 4\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{y\text{-side}})\,\mathrm{d}\mathbf{x},\\
K_{ii-N_xN_y} &= K_{ii+N_xN_y} = 4\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{z\text{-side}})\,\mathrm{d}\mathbf{x},
\end{aligned} \tag{17}$$

$$\begin{aligned}
K_{ii-N_x-1} &= K_{ii-N_x+1} = K_{ii+N_x-1} = K_{ii+N_x+1}\\
&= 2\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{xy\text{-edge}})\,\mathrm{d}\mathbf{x},
\end{aligned}$$

$$\begin{aligned} K_{ii-N_xN_y-1} &= K_{ii-N_xN_y+1} = K_{ii+N_xN_y-1} = K_{ii+N_xN_y+1} \\ &= 2\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{xz\text{-edge}})\,\mathrm{d}\mathbf{x}, \\ K_{ii-N_xN_y-N_x} &= K_{ii-N_xN_y+N_x} = K_{ii+N_xN_y-N_x} = K_{ii+N_xN_y+N_x} \\ &= 2\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{yz\text{-edge}})\,\mathrm{d}\mathbf{x}, \end{aligned} \tag{18}$$

$$\begin{aligned} K_{ii-N_xN_y-N_x-1} &= K_{ii-N_xN_y-N_x+1} = K_{ii-N_xN_y+N_x-1} = K_{ii-N_xN_y+N_x+1} \\ &= K_{ii+N_xN_y-N_x-1} = K_{ii+N_xN_y-N_x+1} = K_{ii+N_xN_y+N_x-1} = K_{ii+N_xN_y+N_x+1} \\ &= 1\int_{\hat{\Omega}} \nabla\varphi_{\text{center}} \cdot (D\nabla\varphi_{\text{corner}})\,\mathrm{d}\mathbf{x}. \end{aligned} \tag{19}$$

Basis functions related to boundary nodes are handled by scaling the above results by $\frac{1}{2}$, $\frac{1}{4}$, or $\frac{1}{8}$, for $\mathbf{x}_i$ lying on a face, an edge, or a corner of the boundary $\partial\Omega$, respectively.

### 2.1.2. *Semi-discretization of model*

We apply now the finite element discretization obtained for (5) to the original system. Approximate each unknown function $C(\mathbf{x},t)$, $F(\mathbf{x},t)$, and $B(\mathbf{x},t)$ by an expansion in the nodal basis functions $\varphi_i(\mathbf{x})$ analogous to (7), and let $U_C$, $U_F$, and $U_B$ denote the vectors of expansion coefficients (i.e., the nodal values). Let the forcing terms $R_F$, $R_B$, $J_{\text{pump}}$, and $J_{\text{leak}}$ in (1)–(3) be written as expansions with respect to the $\varphi_i$ as in (9) so that testing with respect to $\varphi_i(\mathbf{x})$ in the Galerkin ansatz yields a multiplication by the mass matrix $M$ with the respective coefficient vectors as in (10). Therefore, applying the form (11) to (1)–(3) yields the semi-discretization

$$M\,\frac{\mathrm{d}U_C}{\mathrm{d}t} + K^{(C)}U_C = M(R_F + R_B - J_{\text{pump}} + J_{\text{leak}}) + \Sigma, \tag{20}$$

$$M\,\frac{\mathrm{d}U_F}{\mathrm{d}t} + K^{(F)}U_F = MR_F, \tag{21}$$

$$\frac{\mathrm{d}U_B}{\mathrm{d}t} = R_B. \tag{22}$$

The nodal values of $R_F$, $R_B$, $J_{\text{pump}}$, and $J_{\text{leak}}$ in (20)–(22) are denoted by the same letters as the corresponding functions in (1)–(3), for convenience.

The vector $\Sigma$ holds the constant $\hat{\sigma}$ in the components that correspond to nodes at which a CRU is currently firing. It is derived by evaluating

$$\Sigma_i = \int_{\Omega} \sigma\varphi_i\,\mathrm{d}\mathbf{x} = \sum_{\hat{\mathbf{x}}\in X_{\text{CRU}}} \left(\hat{\sigma}S_{\hat{\mathbf{x}}}(C,t;T_{\text{open}}) \int_{\Omega} \varphi_i\delta(\mathbf{x}-\hat{\mathbf{x}})\,\mathrm{d}\mathbf{x}\right)$$

with

$$\int_{\Omega} \varphi_i\delta(\mathbf{x}-\hat{\mathbf{x}})\,\mathrm{d}\mathbf{x} = \varphi_i(\hat{\mathbf{x}}).$$

Recall that $X_{\mathrm{CRU}} \subset X_N$ by construction, hence $\varphi_i(\hat{\mathbf{x}}) = 1$ if and only if $\mathbf{x}_i = \hat{\mathbf{x}}$ is a CRU, that is, $\mathbf{x}_i \in X_{\mathrm{CRU}}$. Therefore, the vector $\Sigma$ has the following values:

- $\Sigma_i = 0$, if $\mathbf{x}_i$ is *not* a CRU ($\mathbf{x}_i \notin X_{\mathrm{CRU}}$),
- $\Sigma_i = 0$, if $\mathbf{x}_i$ is a CRU ($\mathbf{x}_i \in X_{\mathrm{CRU}}$), but *not* open ($S_{\hat{\mathbf{x}}} = 0$),
- $\Sigma_i = \hat{\sigma}$, if $\mathbf{x}_i$ is a CRU ($\mathbf{x}_i \in X_{\mathrm{CRU}}$), which is open ($S_{\hat{\mathbf{x}}} = 1$).

## 2.2. Complete discretization

In order to avoid undue restrictions on the size of the time step used, it is conventional to use an implicit time-discretization [25]. Hence, we will evaluate the diffusive terms at the new time step $t_n$. However, in order to facilitate the evaluation of the stochastic model for the spark events, $\sigma(C, \mathbf{x}, t; T_{\mathrm{open}})$ has to be taken at the old time step $t_{n-1}$; this is also reasonable from a physical point of view. To avoid the use of a nonlinear solver in the time-stepping and since the spark events are expected to be the dominant forcing term, we lag all reaction terms in time by evaluating them at $t_{n-1}$.

Let $U_\ell^n \approx U_\ell(t_n)$ denote the time discretization of $U_\ell(t)$ at $t = t_n$ for $\ell = C, F, B$. The semi-implicit time discretization described above yields then the fully discretized system

$$(M + \Delta t\, K^C)U_C^n = MU_C^{n-1} + \Delta t\, M(R_F^{n-1} + R_B^{n-1} - J^{n-1}) + \Sigma^{n-1}, \tag{23}$$

$$(M + \Delta t\, K^F)U_F^n = MU_F^{n-1} + \Delta t\, MR_F^{n-1}, \tag{24}$$

$$U_B^n = U_B^{n-1} + \Delta t\, MR_B^{n-1} \tag{25}$$

with the short-hand notation $J := J_{\mathrm{pump}} - J_{\mathrm{leak}}$. Here, $K^C$ and $K^F$ denote the stiffness matrices obtained for the appropriate diffusivity matrices $D_C$ and $D_F$, respectively; the mass matrix $M$ is the same for all species. Note that with this time discretization (20)–(22) has been decoupled into three smaller systems (23)–(25), each of which is linear in its unknown vector $U_\ell^n$, $\ell = C, F, B$. This observation will motivate the coarse-grained parallelism introduced below.

For best stability properties, the NDF1 method with automatic step size control following [23] is used, which also minimizes the memory requirements. The NDF1 step size control mechanism compares an estimated local truncation error to a user defined tolerance. If the error is too large, our implementation halves the step size and recomputes the step. After several successful computations with the same time step, the step size is increased by a factor two. Additionally, the step size is set to a minimum value upon reaching a spark time, because we expect that a small step size is needed to compute solutions immediately after the rapid change in solution due to a spark event.

## 2.3. The lumped mass method

The conjugate gradient (CG) method will be used for all linear solves inside the NDF1 method. From Section 2.1 we have analytic forms for the mass and stiffness matrices. These are used to obtain a matrix-free matrix–vector product routine, which is sufficient for the CG method; no system matrix is ever assembled. This approach is the key to the major savings in memory achieved by our method over conventional finite element methods that rely on matrix assembly.

In order to apply the CG method, the system matrices $A^\ell = (M + \Delta t K^\ell)$, $\ell = C, F$, in (23)–(24) must be symmetric positive definite (SPD). Since $M$ and $K^\ell$ are both SPD, we have that $A^\ell$ is SPD for all $\Delta t > 0$. However, since the components of the solution vectors represent concentrations, we wish to guarantee additionally that the system matrix $A^\ell$ is an $M$-matrix. An $M$-matrix is a nonsingular matrix, whose inverse has only nonnegative entries; this can be guaranteed, if the matrix is nonsingular, has positive diagonal and nonpositive off-diagonal entries; the stiffness matrix $K^\ell$ is an example of an $M$-matrix, because it is SPD (hence nonsingular), has positive diagonal entries, and all off-diagonal entries are either zero or negative.

Since $A^\ell$ is SPD for all $\Delta t > 0$, we can guarantee nonsingularity. Also, the diagonal entries of $A^\ell$ are positive for all $\Delta t > 0$. But the nonzero off-diagonal entries of $M$ are positive, hence the nonzero off-diagonal entries of $A^\ell$ may become positive for $\Delta t > 0$ sufficiently small. In order to ensure that the off-diagonal entries of $A^\ell$ stay nonpositive for all $\Delta t > 0$, we use the method of lumped masses following [25]. This method approximates $M$ by a diagonal matrix $\hat{M}$ of positive entries; its off-diagonal entries are thus all zero, hence the off-diagonal entries of $A^\ell$ are nonpositive for all $\Delta t > 0$.

Recall that the mass matrix $M = (M_{ij})$ is defined by

$$M_{ij} = (\varphi_i, \varphi_j) := \int_\Omega \varphi_i \varphi_j \, \mathrm{d}\mathbf{x} \tag{26}$$

The lumped mass matrix $\hat{M} = (\hat{M}_{ij})$ is then introduced by defining [25]

$$\hat{M}_{ij} = (\varphi_i, \varphi_j)_q := V \sum_{\mathbf{x}_k \in X_N} \varphi_i(\mathbf{x}_k)\varphi_j(\mathbf{x}_k), \tag{27}$$

where $X_N$ denotes the set of grid points and $V$ the volume of one finite element $V = \Delta x \Delta y \Delta z$, as introduced in Section 2.1. The theory for the lumped mass method is based on the observation that $(\cdot,\cdot)_q$ is a quadrature rule that gives an approximation to $(\cdot,\cdot)$ [25]. To prove that this is equivalent to diagonalizing the mass matrix $M$, we need to show that

$$\hat{M}_{ij} = \begin{cases} \displaystyle\sum_{k=1}^{N} M_{ik} & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases} \tag{28}$$

The nodal basis functions have the property $\varphi_i(\mathbf{x}_j) = \delta_{ij}$, where $\delta_{ij}$ denotes the Kronecker delta function, which is 1 if $i = j$ and 0 otherwise. Hence, we have from (27)

$$\hat{M}_{ij} = V \sum_{\mathbf{x}_k \in X_N} \varphi_i(\mathbf{x}_k)\varphi_j(\mathbf{x}_k) = V \sum_{\mathbf{x}_k \in X_N} \delta_{ik}\delta_{jk} = V\delta_{ij}.$$

Therefore, $\hat{M}_{ij} = 0$, if $i \neq j$. It remains to be shown that $\hat{M}_{ii}$ equals to the row sums of the elements of $M$. Using the integral forms (12)–(15), derive the sum

$$\sum_{k=1}^{N} M_{ik} = \frac{64V}{216} + 6\,\frac{16V}{216} + 12\,\frac{4V}{216} + 8\,\frac{V}{216} = V,$$

hence, $\hat{M}_{ii} = \sum_{k=1}^{N} M_{ik}$. This completes the proof for (28).

To answer the question whether the lumped mass method will degrade the accuracy of the numerical method, the equivalence with a quadrature rule is used. From [25], for smooth right-hand sides, the semi-discrete formulation of the problem with a lumped mass method will converge with second order accuracy provided the quadrature error is on the order of $h^2$ and the method converges in at least second order when using the standard mass matrix formulation. A proof of the appropriate bound for the quadrature error using triangular elements in two dimensions is given in [6,20] and can be extended to this problem as in [2]. Therefore, we can expect second order convergence of the lumped mass finite element method in a scalar equation with sufficiently smooth forcing function.

With the mass matrix diagonalized by the lumped mass method, the new system matrices $\hat{A}^{\ell} = (\hat{M} + \Delta t K^{\ell})$ are guaranteed to be $M$-matrices for all $\Delta t > 0$ sufficiently small. We will use this definition for the system matrices in all of our computations. Additional information on the choice of the mesh spacings $\Delta x$, $\Delta y$, and $\Delta z$ that make use of the particular values of the diffusivities in Table 8 can be found in [13].

## 2.4. Parallel implementation

With the time discretization described in Section 2.2 applied to the semi-discrete problem (20)–(22), the linear system of equations decouples into three smaller systems (23)–(25) for each coefficient vector $U_C^n$, $U_F^n$, and $U_B^n$ at the new time step. This decoupling leads immediately to a coarse-grained parallel method using three processors. All variables are only stored and computed on the processor that needs them and communication is restricted to the needed variables. This design is appropriate for a small cluster of loosely coupled workstations.

Label the three processors '$C$', '$F$', and '$B$'. In an initialization step, store each of $U_C^0$, $U_F^0$, and $U_C^0$ on the appropriate processor. Then the following algorithm, which describes the parallel routine for solving (23)–(25) is run in parallel on '$C$', '$F$', and '$B$' for each time step:

(1) Broadcast (send) $U_C^{n-1}$ from Process '$C$' to Processes '$F$' and '$B$'.
(2) Compute $R_C^{n-1} := J_{\text{leak}}^{n-1} - J_{\text{pump}}^{n-1} + \Sigma^{n-1}$ on Process '$C$', $R_F^{n-1}$ on Process '$F$', and $R_B^{n-1}$ on Process '$B$' in parallel.
(3) Reduce (gather and add) $R_F^{n-1}$ and $R_B^{n-1}$ from Processes '$F$' and '$B$' to Process '$C$'. This will complete the rate term $R_C^{n-1} + R_F^{n-1} + R_B^{n-1}$ for '$C$'.
(4) Solve linear systems for $U_C^n$, $U_F^n$, and $U_B^n$ in parallel.
(5) Check whether estimated local truncation error for $U_C^n$ is less than the desired tolerance; if not, halve the step size and go to step 4.

It will be shown in the next section that this coarse-grained parallel method is successful in obtaining numerical solutions for a finer mesh than the serial code.

This coarse-grained parallelism is a first approach and designed to demonstrate in principle that a splitting of the equations in (1)–(3) is a viable approach to use parallelism for this model. Clearly, the load-balancing cannot be expected to be perfect, because the equation for $C$ is substantially more complex than the others. This equation also requires more auxiliary variables, hence the memory-savings over a very efficient serial code are limited, as well. Nonetheless though, it does allow for the solution of a problem that is twice as large as possible on a serial machine.

## 3. Results

The computational results of our work are presented in three subsections. Section 3.1 presents two convergence studies for the scalar partial differential equation (5). One case has a right-hand side function that satisfies $f \in L_2(\Omega)$ and hence is sufficiently smooth for classical theory to apply. The other test case uses $f = \delta(\mathbf{x})$, the Dirac delta function. The classical theory does not apply in this case, but a consideration will be presented that justifies the expectation of square-root convergence, which is borne out by the numerical results. The remaining Sections 3.2 and 3.3 both consider the original system of model equations given in (1)–(3). Section 3.2 will demonstrate the convergence of the numerical solution also for this system and will show examples of the behavior of the sparking mechanism; this confirms that the method is able to simulate the phenomenon in question. Section 3.3 presents a parallel performance study that shows that the parallel code is able to solve a problem of twice the size as the serial code.

Recall that the time steps are chosen automatically in NDF1 [23]. In our implementation of NDF1, the estimated local truncation error is compared to a relative tolerance set at $10^{-2}$ in order to accept a time step. Additionally, this time step is accepted only if all components of the solution are nonnegative. If a time step is rejected, then the step size is halved, and the solution is recomputed. If three successive time steps are accepted, then the time step is increased by a factor two.

The matrix-free implementation of the conjugate gradient method without preconditioning, described in Section 2 and using the lumped mass matrix, is used to solve each linear system. For all cases the user defined CG tolerance is fixed at $10^{-6}$. This tolerance is compared to the relative residual in Euclidean vector norm. The following tables use the $L_2$-norm at a point in time defined by

$$\|u(\cdot,t)\|_{L_2} = \left( \int_\Omega |u(\mathbf{x},t)|^2 \, \mathrm{d}\mathbf{x} \right)^{1/2}.$$

The parallel performance studies use an 8-processor cluster of four dual Linux PCs with 1000 MHz Pentium III processors and 1 GB of memory per node. The nodes are connected by 100 Mbps commodity cables on a dedicated network, forming a Beowulf cluster. Files are served from one of the nodes using a SCSI hard drive. Code development and initial convergence studies were performed on two dual Linux PCs with 800 MHz Pentium III processors and 1 GB of memory per node.

### 3.1. Scalar convergence studies

#### 3.1.1. Smooth scalar problem

For a smooth right-hand side, the standard finite element method is guaranteed to show second order convergence of the $L_2$-norm of the finite element solution. To demonstrate that second order convergence is maintained when using a lumped mass matrix, we consider the scalar parabolic partial differential equation

$$\begin{aligned} &\frac{\partial u}{\partial t} - \nabla \cdot (\nabla u) = f && \text{in } \Omega, \\ &\nu \cdot (\nabla u) = 0 && \text{on } \partial\Omega, \\ &u = 0.1 && \text{at } t = 0, \end{aligned} \tag{29}$$

Table 1
Convergence study for scalar partial differential equation with smooth right-hand side. The third and fourth columns indicate that the numerical solutions $u_{\Delta x}$ converge to the true solution $u$ at $t=1$. The last column shows that the computed solutions converge to the solutions $v_{\Delta x}$ obtained without using the lumped mass method

| $N_x$ | $\Delta x$ | $\|u_{\Delta x} - u\|_{L_2}$ | $\frac{\|u_{\Delta x} - u\|_{L_2}}{\|u\|_{L_2}}$ | $\|u_{\Delta x} - v_{\Delta x}\|_{L_2}$ |
|---|---|---|---|---|
| 4 | 1/2 | 0.1757 | 0.2998 | 0.1812 |
| 8 | 1/4 | 0.0484 | 0.0788 | 0.0478 |
| 16 | 1/8 | 0.0123 | 0.0198 | 0.0122 |
| 32 | 1/16 | 0.0032 | 0.0051 | 0.0031 |
| 64 | 1/32 | 0.0011 | 0.0018 | 0.0008 |

with $f \in L_2(\Omega)$. Define the domain as $\Omega = (-1,1) \times (-1,1) \times (-1,1)$. If we have the true solution

$$u(x,y,z,t) = 0.1 + 0.9(x^2-1)^2(y^2-1)^2(z^2-1)^2(1-\mathrm{e}^{-t^2}), \tag{30}$$

then it is clear that the initial condition and boundary conditions hold. Divide the domain into a regular mesh of brick elements using $\Delta x = \Delta y = \Delta z$. If we vary the mesh size $\Delta x = 2^{-n}$ for $n = 1,2,\ldots$, the number of points is $N_x = 2^{n+1}$.

For this smooth test problem, finite element theory predicts second-order convergence, that is,

$$\|u_{\Delta x}(\cdot,t) - u(\cdot,t)\|_{L_2} \leqslant C\Delta x^2 \quad \text{as } \Delta x \to 0 \tag{31}$$

with a constant $C$ independent of $\Delta x$ for all times $t$ [25]. The observed order of convergence for the method can be approximated by

$$p = \log_2\left(\frac{\|u_{2\Delta x} - u_{4\Delta x}\|_{L_2}}{\|u_{\Delta x} - u_{2\Delta x}\|_{L_2}}\right). \tag{32}$$

We expect that this estimated order of convergence tends toward the value 2 as the mesh becomes finer.

In Table 1, $u_{\Delta x}$ is the solution of the numerical method to (29) using a mesh defined by $\Delta x$. $u$ is the true solution given in the preceding paragraph. $v_{\Delta x}$ is the standard FEM solution without using the lumped mass matrix. In Table 1, absolute and relative errors given in the $L_2$-norm clearly converge to zero. Also, it can be seen that the solutions to FEM with and without the lumped mass method converge to each other, and both converge.

Table 2 shows similar computations as Table 1, but with the solution of the numerical method over the finest mesh used as a "true" solution. The third and fourth columns of Table 2 show the same behavior as the corresponding columns in Table 1, thus justifying the use of the solution on the finest mesh as approximation to the true solution. The final column confirms that in agreement with the theory (31), we have $\|u_{\Delta u} - u\|_{L_2}/\Delta x^2 \leqslant C$.

Table 3 shows the computed approximate values for $p$ from (32) and the $L_2$-norm of the difference between successive meshes. The $L_2$-norm of the difference between meshes reduces by a factor of four for each mesh refinement. Also, the approximations for the order of convergence tend toward the value 2 as the mesh becomes finer. This agrees with the classical theory and validates the use of the method in this problem.

Table 2
Second convergence study for scalar partial differential equation with smooth right-hand side. The numerical solutions $u_{\Delta x}$ converge to $u_{\Delta x_{\min}}$ at $t = 1$

| $N_x$ | $\Delta x$ | $\|u_{\Delta x} - u_{\Delta x_{\min}}\|_{L_2}$ | $\frac{\|u_{\Delta x} - u_{\Delta x_{\min}}\|_{L_2}}{\|u_{\Delta x_{\min}}\|_{L_2}}$ | $\frac{\|u_{\Delta x} - u_{\Delta x_{\min}}\|_{L_2}}{\Delta x^2}$ |
|---|---|---|---|---|
| 4 | 1/2 | 0.3296 | 0.5239 | 1.3185 |
| 8 | 1/4 | 0.0823 | 0.1308 | 1.3168 |
| 16 | 1/8 | 0.0196 | 0.0312 | 1.2570 |
| 32 | 1/16 | 0.0039 | 0.0062 | 1.0062 |

Table 3
Convergence order estimates for scalar partial differential equation with smooth right-hand side. The last column tends to a convergence order estimate of 2

| $N_x$ | $\Delta x$ | $\|u_{\Delta x} - u_{2\Delta x}\|_{L_2}$ | $\frac{\|u_{\Delta x} - u_{2\Delta x}\|_{L_2}}{\|u_{\Delta x}\|_{L_2}}$ | $\log_2\left(\frac{\|u_{2\Delta x} - u_{4\Delta x}\|_{L_2}}{\|u_{\Delta x} - u_{2\Delta x}\|_{L_2}}\right)$ |
|---|---|---|---|---|
| 4 | 1/2 | 0.8145 | 1.7498 | N/A |
| 8 | 1/4 | 0.2464 | 0.4274 | 1.7249 |
| 16 | 1/8 | 0.0626 | 0.1016 | 1.9761 |
| 32 | 1/16 | 0.0157 | 0.0251 | 1.9953 |
| 64 | 1/32 | 0.0039 | 0.0062 | 1.9989 |

### 3.1.2. *Discontinuous scalar problem*

To demonstrate that the lumped mass finite element method can be applied to a problem with a severe discontinuity resulting from a Dirac delta function as forcing term, consider the scalar partial differential equation problem

$$\begin{aligned}
&\frac{\partial u}{\partial t} - \nabla \cdot (\nabla u) = \delta(\mathbf{x} - 0) && \text{in } \Omega,\\
&v \cdot (\nabla u) = 0 && \text{on } \partial\Omega,\\
&u = 0.1 && \text{at } t = 0.
\end{aligned} \tag{33}$$

The numerical test will be conducted in the same fashion as the previous test for the smooth scalar problem. Let $\Omega = (-1,1) \times (-1,1) \times (-1,1)$ be discretized into a regular mesh with $\Delta x = \Delta y = \Delta z$. Note that for $\Delta x = 2^{-n}$ the point $(0,0,0)$ will be an element of the mesh. If the convergence order approximation given in (32) tends toward a constant as $n$ becomes large, we may conclude that the method converges in the case of a Dirac delta forcing term.

In the smooth test problem (29), we were able to compare the numerical solution to the given true solution. In lieu of the true solution, we will use the numerical solution on the finest available mesh here; this approach is justified by the observations in Table 2. Additionally in the smooth case, we were able to compare the convergence of the error to the theoretically predicted value of $\Delta x^2$. It would be helpful if a similar comparison could be done for the discontinuous problem (33), as well.

Table 4
Convergence study for scalar partial differential equation with discontinuous right-hand side. The numerical solutions $u_{\Delta x}$ converge to $u_{\Delta x_{\min}}$ at $t = 1$

| $N_x$ | $\Delta x$ | $\|u_{\Delta x} - u_{\Delta x_{\min}}\|_{L_2}$ | $\frac{\|u_{\Delta x} - u_{\Delta x_{\min}}\|_{L_2}}{\|u_{\Delta x_{\min}}\|_{L_2}}$ | $\frac{\|u_{\Delta x} - u_{\Delta x_{\min}}\|_{L_2}}{\Delta x^{1/2}}$ |
|---|---|---|---|---|
| 4 | 1/2 | 2.5488 | 3.9134 | 3.6045 |
| 8 | 1/4 | 0.8467 | 1.3000 | 1.6934 |
| 16 | 1/8 | 0.2615 | 0.4015 | 0.7396 |
| 32 | 1/16 | 0.0669 | 0.1026 | 0.2674 |

Table 5
Convergence order estimates for scalar partial differential equation with discontinuous right-hand side. The last column tends to a convergence order estimate of $\frac{1}{2}$

| $N_x$ | $\Delta x$ | $\|u_{\Delta x} - u_{2\Delta x}\|_{L_2}$ | $\frac{\|u_{\Delta x} - u_{2\Delta x}\|_{L_2}}{\|u_{\Delta x}\|_{L_2}}$ | $\log_2\left(\frac{\|u_{2\Delta x} - u_{4\Delta x}\|_{L_2}}{\|u_{\Delta x} - u_{2\Delta x}\|_{L_2}}\right)$ |
|---|---|---|---|---|
| 4 | 1/2 | 0.3148 | 0.4833 | N/A |
| 8 | 1/4 | 0.1927 | 0.2956 | 0.7084 |
| 16 | 1/8 | 0.1340 | 0.2057 | 0.5234 |
| 32 | 1/16 | 0.0946 | 0.1452 | 0.5033 |
| 64 | 1/32 | 0.0669 | 0.1026 | 0.5004 |

While the classical theory does not apply here, an estimate can be derived in the following way: In three dimensions, the Sobolev space $H^{3/2+\varepsilon}$ is continuously embedded in $C^0$ for any $\varepsilon > 0$ [1]. Consider the integral definition of the Dirac delta function as a functional over this space. Then through the dual embedding, we can approximate the space to which $\delta(\mathbf{x})$ belongs to as $H^{-3/2-\varepsilon}$, namely, the dual of $H^{3/2+\varepsilon}$. From [25], we can expect that the convergence order of the FEM is $h^{k+2}$, if the right-hand side function satisfies $f \in H^k$, where $h$ is the largest side length of an element. So, if our method converges, we can compare our convergence estimate to $\Delta x^{1/2}$ by computing $\|u_{\Delta x} - u\|_{L_2}/\Delta x^{1/2}$, which should stay bounded by a constant $C$ analogously to $\|u_{\Delta x} - u\|_{L_2}/\Delta x^2$ being bounded for the smooth test problem.

In Table 4, we compare the error between the numerical solution and the solution over the finest available mesh. The third and fourth column indicate that the solutions converge toward the approximate true solution $u_{\Delta x_{\min}}$. If we assume the convergence order $\frac{1}{2}$ given in the preceding paragraph, then we can compute the error constant by $\|u_{\Delta x} - u\|_{L_2}/\Delta x^{1/2}$; the fact that this quantity in the final column actually tends to zero indicates a better than expected convergence order.

Table 5 shows the estimates for $p$ and the $L_2$-norm of the difference between successively finer meshes. Note that the approximation for the order of convergence tends toward the value of $\frac{1}{2}$ as the mesh becomes finer. This computational evidence indicates that the lumped mass finite element method with parallelepiped elements converges as $\Delta x^{1/2}$ in space. In fact, we can claim that the lumped mass method has no negative effect on the convergence order. Since the method converges, we conclude that this method is appropriate for the discontinuous nature of the model equation (1).

### 3.2. Model convergence studies

The previous section showed that the numerical method converges with convergence rates that agree with theoretical considerations. In particular, it was demonstrated numerically that our method is able to compute a convergent solution, if one Dirac delta function is present on the right-hand side of the scalar prototype problem (5). In this section, our method is applied to the original system of reaction-diffusion equations in (1)–(3), in which Eq. (1) for the concentration $C$ of calcium ions includes a superposition of many Dirac delta functions on the right-hand side. Convergence will be demonstrated by showing results for $C$ obtained on two different meshes; additionally, we will show that the sparking mechanism performs as intended. The first example uses a small prototype domain, but realistic time scales. The second example uses a realistic domain but is only computed up to the small final time of 10 ms; it confirms the proper behavior of the simulator and provides numerical performance data for our method.

#### 3.2.1. Model test case: multiple spark events in small domain

Let the domain $\Omega = (-2, 2) \times (-1, 1) \times (-1, 1)$ be discretized using a mesh with $64 \times 32 \times 32$ subintervals. CRU spacings of $\Delta x_s = \Delta y_s = \Delta z_s = 0.25$ are used here. The domain and CRU spacings have been chosen smaller than in Table 8 for this example in order to present a visual impression of the wave behavior of the concentration by aggressively promoting the triggering of spark events. Fig. 3 shows snapshots each taken at times immediately after spark events have occurred. Each plot shows an isosurface of the numerical solution $C$ with surface value corresponding to $C = 65$. The initial spark occurs at a CRU close to the near corner, and the calcium concentration increases sharply in the vicinity. At the following spark times, the increased level of calcium concentration triggers sparks in adjacent CRUs, leading to the self-organized wave in Figs. 3(a)–(f). The results for the remaining variables $F$ and $B$ are not shown, because $C$ is the most important species.

Fig. 4 shows the solution for the same case as Fig. 3 using a finer mesh of $128 \times 64 \times 64$ subintervals. Comparing the snapshot at each time in Figs. 3 and 4, we observe that the shapes of the plots agree. This demonstrates that the numerical solution converges also for the model problem. Notice that the finer solutions in Fig. 4 appears darker than the corresponding solutions in Fig. 3, because the isosurface mesh is finer; that is to be expected for a solution with finer resolution.

#### 3.2.2. Model test case: multiple spark events in large domain

Fig. 5 presents a solution to the model problem for a small time frame 10 ms, but on a realistic cell domain and CRU distribution. The domain is given by $\Omega = (-33.6, 33.6) \times (-6.4, 6.4) \times (-6.4, 6.4)$ and is discretized using a mesh with $256 \times 64 \times 64$ subintervals. The CRU spacings are physically correct as given in Table 8. The lower left-hand corner is initially set to a high concentration in order to incite spark events. Note that some sparks appear in areas of low concentration, but do not cause a wave to occur; this is the correct, experimentally observed behavior. Figs. 5(a) and (b) give the solution for $C$ immediately after the first round of sparks $t = 1$ ms and then at $t = 10$ ms.

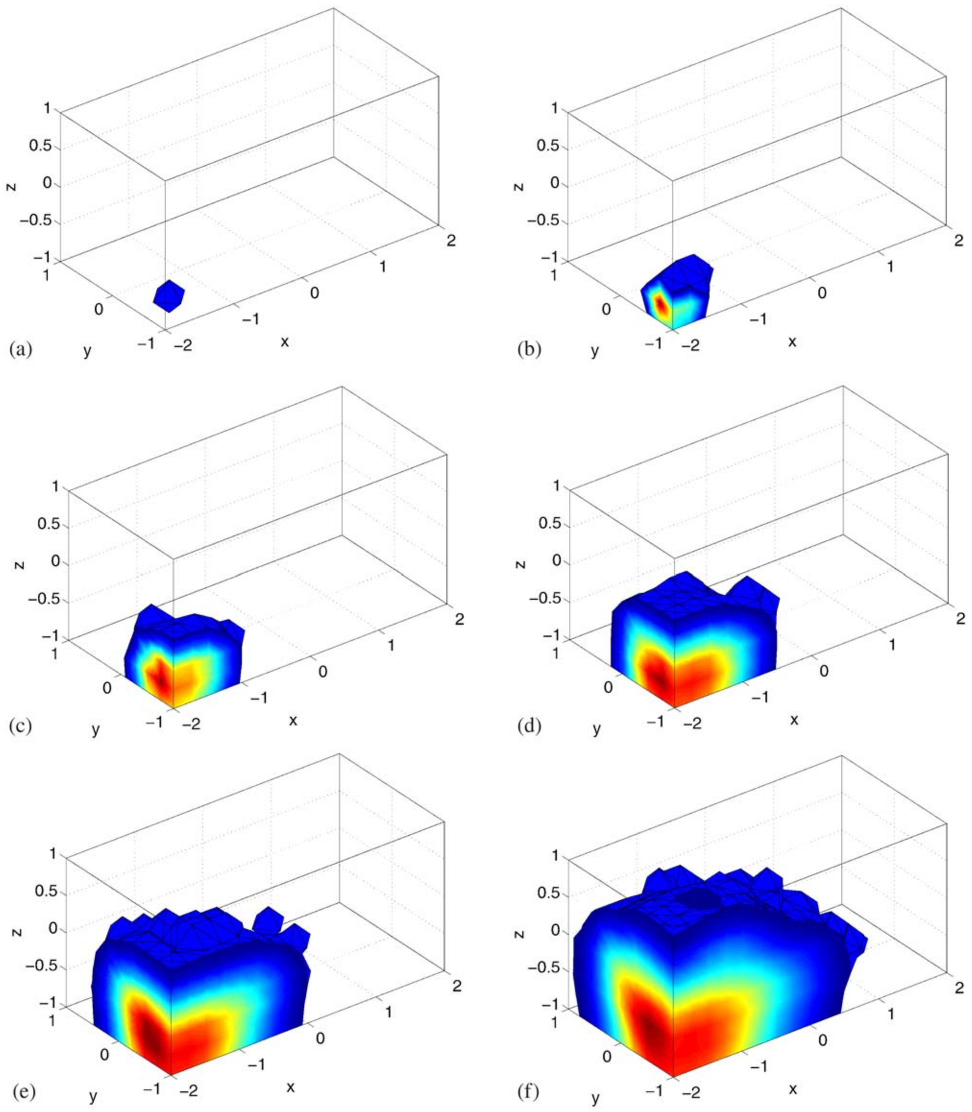

Fig. 3. Isosurface plots for the calcium concentration $C$ at times (a) 1 ms, (b) 2 ms, (c) 3 ms, (d) 4 ms, (e) 5 ms, and (f) 6 ms. The domain $\Omega = (-2, 2) \times (-1, 1) \times (-1, 1)$ is discretized using a mesh with $64 \times 32 \times 32$ subintervals.

Finally, Fig. 6 shows numerical data acquired while computing the solutions in Fig. 5. Since spark events are allowed only at integer times, Fig. 6(a) confirms that the time step is set to its minimum at the spark times. After each spark times, the time steps grow again until the next spark time. Fig. 6(b) shows the total CG iterations used in evaluating all of the linear systems (for $U_C^n$, $U_F^n$, and $U_B^n$ together). This is done to emphasize that relatively few CG iterations are needed at each time step, despite the rather tight tolerance used for the CG method.

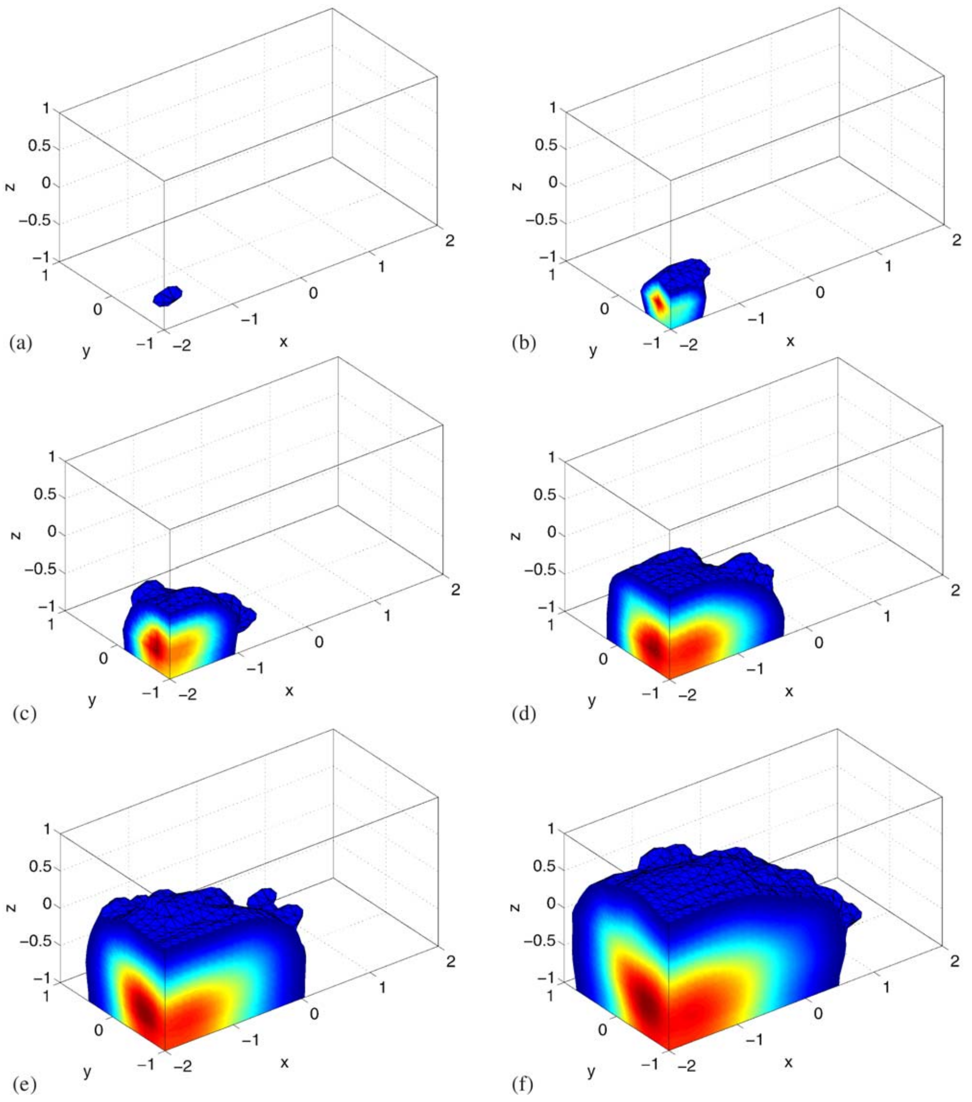

Fig. 4. Isosurface plots for the calcium concentration $C$ at times (a) 1 ms, (b) 2 ms, (c) 3 ms, (d) 4 ms, (e) 5 ms, and (f) 6 ms. The domain $\Omega = (-2,2) \times (-1,1) \times (-1,1)$ is discretized using a mesh with $128 \times 64 \times 64$ subintervals.

## *3.3. Parallel performance studies*

For the performance studies of the coarse-grained parallel method, let $\Omega=(-L_x, L_x)\times(-6.4, 6.4)\times(-6.4, 6.4)$ with CRU spacings of $\Delta x_s = 1.98$ and $\Delta y_s = \Delta z_s = 0.8$; In order to obtain eight nodes between CRUs in the $y$- and $z$-directions, we use a $128 \times 128$ mesh in the $(y,z)$-plane in all cases. Both the domain size $L_x$ and the number of subintervals $N_x$ in the $x$-direction are varied proportionally

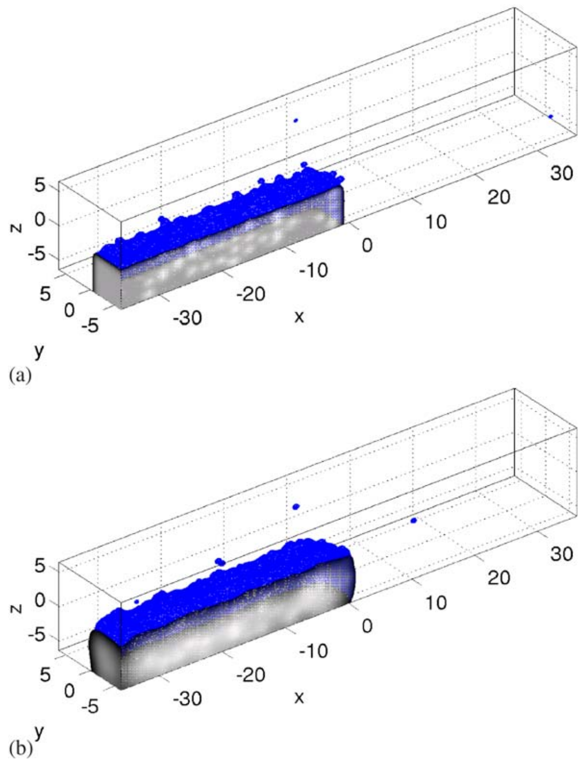

Fig. 5. Isosurface plots of $C$ at times (a) $t = 1$ ms, and (b) $t = 10$ ms. The domain $\Omega = (-33.6, 33.6) \times (-6.4, 6.4) \times (-6.4, 6.4)$ is discretized using a mesh with $256 \times 64 \times 64$ subintervals.

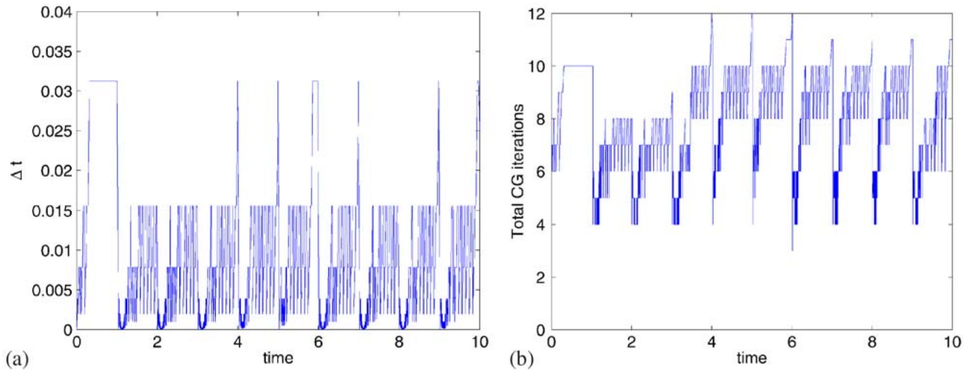

Fig. 6. Numerical data for model problem simulations; (a) size of time steps taken vs. time, and (b) total number of CG iterations vs. time.

for the performance studies. On the one hand, this guarantees identical numerical accuracy (because $\Delta x = 2L_x/N_x$ is constant). On the other hand, we expect both timing results and memory usage to scale proportionally to $N_x$, for convenience. For the studies, we use $N_x = 28, 56, 122, 224, 448$.

Table 6
Memory usage per processor in MB for the serial and coarse-grained parallel code. The last column shows the ratio of memory used by parallel code over the serial code

| $N_x$ | DOF | Serial | Parallel | Ratio |
|---|---|---|---|---|
| 28 | 1,376,256 | 57.7 | 41.5 | 0.72 |
| 56 | 2,752,512 | 143 | 106 | 0.74 |
| 112 | 5,505,024 | 284 | 208 | 0.73 |
| 224 | 11,010,048 | 531 | 390 | 0.74 |
| 448 | 22,020,096 | N/A | 728 | N/A |

Table 7
Run time in minutes of the serial and coarse-grained parallel code. The last column shows the observed speedup of the parallel code over the serial code

| $N_x$ | DOF | Serial | Parallel | Ratio |
|---|---|---|---|---|
| 28 | 1,376,256 | 940.3 | 486.6 | 1.9 |
| 56 | 2,752,512 | 1836.8 | 967.5 | 1.9 |
| 112 | 5,505,024 | 3628.9 | 1825.2 | 2.0 |
| 224 | 11,010,048 | 6634.7 | 3483.4 | 1.9 |

Table 6 shows the observed memory use per processor for the serial and parallel codes. That is, for instance for $N_x = 28$, 41.5 MB were used on the '$C$' processor and somewhat less on the '$F$' and '$B$' processors; since the largest memory usage prevents the solution of a larger problem, it is reported in the table. Savings of about 25% over the memory required for the extremely memory-efficient serial code can be observed for the parallel code in all cases. The final test case ($N_x = 448$) corresponds to $L_x = 31.76$, which approaches the domain in Section 1. The improvement in memory usage for this initial design of the memory code is not optimal, but the parallelism used does allow for solutions to be computed over meshes which are too fine for the serial code, as described in Table 6. Specifically, on a serial machine, the most memory-optimal implementation allowed for the solution of a problem with over 11 million degrees of freedom; the parallel implementation managed to double the problem size to over 22 million degrees of freedom.

Table 7 compares runtimes of the serial and parallel codes for the smaller domains in Table 6. The timings shown are for runs with final time 2 ms. Speedup of a factor 2.0 by the parallel implementation over the serial is observable in all cases. Since we are interested in long term behavior of the spark wave (final time on the order of 100 ms), the speedup will become much more important. The speedup is not optimal, considering that three processors are used. This is caused by the fact that the numerical effort is controlled by the calcium equation (1), which is by far the most complex of the three equations in the model (1)–(3). Namely, a rough estimate of operations indicates that the '$C$' processor has to compute about twice as many elements in $J_{\text{leak}}^{n-1}$ and $\Sigma^{n-1}$ than the '$F$' and '$B$' processors in $R_F^{n-1}$ and $R_B^{n-1}$, respectively. To improve the performance, more sophisticated data distributions would be required.

## 4. Conclusions

The time development of calcium concentration in human heart cells is modeled by a system of three reaction-diffusion equations. The key term models the flow of calcium ions into the cell and uses Dirac delta functions in space and indicator functions in time. A semi-implicit time-stepping with lagged nonlinear terms is used that decouples the linear solves of each of the partial differential equations at each time step. A specialized finite element method is derived that allows for the discretization of the delta functions and for the development of memory-efficient matrix-free linear solves. These features of the discretization are exploited to obtain a coarse-grained parallel simulator that uses one process for each reaction-diffusion equation.

Numerical results demonstrate that the method has convergence orders that agree with theoretical considerations for scalar problems with smooth and nonsmooth forcing terms. Additionally, the results indicate that the method is convergent when applied to the full reaction-diffusion system, despite the nonlinear and discontinuous forcing terms, and that the sparking mechanism is represented realistically. Results on memory usage show that the approach allows for the solution of discretizations with a finer resolution than possible on a single-processor machine. The speedup results indicate that the solution is still dominated by the most complex equation, namely the one for calcium that includes the model of the sparking mechanism.

## Appendix A. The application model

Contraction in mammalian cardiac myocytes is initiated by a rise in the cytoplasmic calcium concentration resulting from calcium release from the sarcoplasmic reticulum (SR), a major intracellular calcium store [4]. The efflux of calcium from the SR occurs through clusters of ryanodine receptors (RyRs) called calcium release units (CRUs). RyRs have the remarkable property that the probability of them being open (allowing calcium to flow out of the SR) increases with the ambient cytoplasmic calcium concentration ($C$) [8–10]. This regenerative release of calcium underlies the propagating waves of calcium release (‘calcium wave’, [3,7,18,19]).

The CRUs are small, approximately 200 nm in diameter [11], and are distributed on the $z$-lines of the cell. The $z$-lines are spaced $\sim 2.0$ μm apart along the longitudinal axis of cell and within the plane of the $z$-line (normal to the longitudinal axis), the CRUs are spaced between 0.4 and 0.8 μm apart [11,15,21]. It is the *discrete* distribution of calcium release sites that confers stability to this system that would otherwise be unstable [24]. Calcium released from a random opening of RyRs in a single CRU is unlikely to raise the calcium concentration of neighboring CRUs sufficiently to trigger further calcium release. In experiments where calcium is measured with a confocal microscope, calcium ions released from a single CRU appear as a bright, brief increase of fluorescence and are called ‘sparks’ because of their visual similarity to their namesake. When the calcium-sensitivity of the RyR (indexed by $K_{\text{prob}}$ in (A.11)) is heightened, for example under stress-induced $\beta$-adrenergic stimulation, stability may be lost and self-organizing calcium waves may develop. These spontaneous calcium waves could alter the timing of the electrical depolarization of the cell and engender life threatening ventricular arrhythmias.

In our model the distribution of CRUs is approximated by regular three-dimensional rectangular lattice. The small size of the CRU relative to the cell dimensions makes treatment of the calcium release sites as point sources a reasonable one.

The mathematical model accounts for the concentrations of calcium ions $C$, a fluorescent calcium indicator $F$, and the endogenous calcium buffer $B$; the latter two can appear in unbound form as $F$ and $B$ and in bound form with $C$, for which we write the short-hand $G$ and $H$, respectively. The reversible binding/unbinding of the indicator and buffer species are modeled by the reaction model

$$C + F \rightleftharpoons G,$$

$$C + B \rightleftharpoons H.$$

Using the same letters again for the molar concentration of the species, the reaction rates for these reactions are

$$R_F = -k_F^{\mathrm{f}} CF + k_F^{\mathrm{b}} G, \tag{A.1}$$

$$R_B = -k_B^{\mathrm{f}} CB + k_B^{\mathrm{b}} H. \tag{A.2}$$

In these rates, the $k_\ell^{\mathrm{f}}$, $\ell = B, F$, are the forward kinetic constants, which describe the loss of free molecules due to binding, and $k_\ell^{\mathrm{b}}$ are the backward kinetic constants, which give the production of free molecules from a previously bound state [16]. Indicator and buffer molecules bind with the calcium ions, but not with each other. Notice that the system conserves mass, because the number of atoms of each species in the reaction model are conserved.

The time-development of the five species concentrations is then given by the system of five reaction-diffusion equations

$$\frac{\partial C}{\partial t} = \nabla \cdot (D_C \nabla C) + R_F + R_B - J_{\mathrm{pump}} + J_{\mathrm{leak}} + \sigma, \tag{A.3}$$

$$\frac{\partial F}{\partial t} = \nabla \cdot (D_F \nabla F) + R_F, \tag{A.4}$$

$$\frac{\partial B}{\partial t} = R_B, \tag{A.5}$$

$$\frac{\partial G}{\partial t} = \nabla \cdot (D_F \nabla G) - R_F, \tag{A.6}$$

$$\frac{\partial H}{\partial t} = -R_B. \tag{A.7}$$

Since reactions between any two of the three species are simple one-to-one binding reactions, the same rate functions appear in different equations [16]. It is also reasonable to assume that the diffusion coefficients of bound and unbound species are the same. Furthermore, $J_{\mathrm{pump}}(C)$ and $J_{\mathrm{leak}}$ are the forcing terms associated with the influence of the sarcoplasmic reticulum (SR) on calcium ion concentration. The Dirac delta functions associated with the sparking of the CRU points are housed within the $\sigma$ term. The values of all constants and model parameters including their units are summarized in Table 8.

In the case of vanishing forcing terms, mass conservation can be demonstrated in the model equations, after one converts the molar concentration units to mass concentration units, thereby introducing the molecular weights as factors on the right-hand side of the reaction-diffusion equation. The time change of the total mass is then the sum of the volume integral over the domain $\Omega$ of all five species; the divergence theorem leads to cancellation of the diffusion terms, and the reaction

Table 8
Constants used in the model case. The molar unit M is short for moles per liter

| | |
|---|---|
| $\mathrm{diag}(D_C)$ | $(0.30, 0.15, 0.15)$ μm²/ms |
| $\mathrm{diag}(D_F)$ | $(0.02, 0.01, 0.01)$ μm²/ms |
| $T_{\mathrm{open}}$ | 5.0 ms |
| $\Delta T$ | 1.0 ms |
| $V_{\mathrm{pump}}$ | 0.2 μM/ms |
| $K_{\mathrm{pump}}$ | 0.184 μM |
| $n_{\mathrm{pump}}$ | 4 |
| $(k_F^{\mathrm{f}}, k_B^{\mathrm{f}})$ | $(0.08, 0.1)$/μM/ms |
| $(k_F^{\mathrm{b}}, k_B^{\mathrm{b}})$ | $(0.09, 0.1)$/ms |
| $n_{\mathrm{prob}}$ | 1.6 |
| $K_{\mathrm{prob}}$ | 15.0 μM |
| $P_{\max}$ | 0.3/ms |
| $(\bar{F}, \bar{B})$ | $(50.0, 123.0)$ μM |
| $J_{\mathrm{leak}}$ | 0.016 μM/ms |
| $\hat{\sigma}$ | 103.64 μM μm³/ms |
| $(C_0,\ F_0,\ B_0)$ | $(0.1, 111.8182, 45.9184)$ μM |
| $(\Delta x_s, \Delta y_s, \Delta z_s)$ | $(2.0, 0.8, 0.8)$ μm |

rates from the right-hand sides cancel, because the molecular weights as their factors add up to zero, which is a consequence of the conservation of atoms in the reaction model.

For simplicity of the simulations, this study assumes an initial distributions of $F(\mathbf{x},0)$ and $G(\mathbf{x},0)$, such that their total value $F(\mathbf{x},0)+G(\mathbf{x},0)=\bar{F}$ for all $\mathbf{x}\in\Omega$. Because of $\mathrm{d}(F+G)/\mathrm{d}t=\nabla\cdot(D_F\nabla(F+G))$, it follows then that $F(\mathbf{x},t)+G(\mathbf{x},t)=\bar{F}$ for all times [16,17]. An analogous argument holds for $B(\mathbf{x},t)+H(\mathbf{x},t)=\bar{B}$. Therefore, we do not have to solve for the species concentrations $G$ and $H$, and we use $G=\bar{F}-F$ and $H=\bar{B}-B$ to eliminate them. The reaction terms $R_F$ and $R_B$ are then given by

$$R_F(C,F) = -k_F^{\mathrm{f}}CF + k_F^{\mathrm{b}}(\bar{F}-F), \tag{A.8}$$

$$R_B(C,B) = -k_B^{\mathrm{f}}CB + k_B^{\mathrm{b}}(\bar{B}-B), \tag{A.9}$$

and it is possible to simulate only the system of three reaction-diffusion equations (1)–(3).

In addition to reactions described, the cytosolic $Ca^{2+}$ concentration is also controlled by $Ca^{2+}$ fluxes across the membrane of the sarcoplasmic reticulum, a $Ca^{2+}$ storage organelle. These fluxes include (i) $J_{\mathrm{pump}}$, which ‘pumps’ $Ca^{2+}$ from the cytoplasma back into the SR against its concentration gradient, (ii) a constant ‘leak’ of $Ca^{2+}$ from the SR into the cytoplasma $J_{\mathrm{leak}}$, and (iii) the term $\sigma(C,\mathbf{x},t;T_{\mathrm{open}})$. Here, $J_{\mathrm{pump}}$ is given as a function of $C$ as

$$J_{\mathrm{pump}}(C) = \frac{V_{\mathrm{pump}}C^{n_{\mathrm{pump}}}}{(K_{\mathrm{pump}})^{n_{\mathrm{pump}}} + C^{n_{\mathrm{pump}}}}. \tag{A.10}$$

$J_{\mathrm{leak}}$ is defined so that $J_{\mathrm{leak}} - J_{\mathrm{pump}}(C_0) = 0$; for the purposes of our simulations that are intended to model the phenomenon of calcium waves this is a reasonable approximation. The actions of the SR are accounted for at every point inside the cell domain.

The key term of the model is $\sigma(C, \mathbf{x}, t; T_{\text{open}})$, which describes the release of calcium at the calcium release units. The release of $Ca^{2+}$ ions into the cell is modeled by (4); the spatial part of the model is given by the superposition of delta functions as described already. In time, the CRUs are allowed to open only at a regular set of discrete times $T_{\text{CRU}}$, at intervals of $\Delta T = 1$ ms. An element of the set $T_{\text{CRU}}$ is called a spark time. The probability per unit time that a CRU will fire at a spark time $t_0 \in T_{\text{CRU}}$ is a function of calcium concentration given by

$$P(C) = \frac{P_{\max} C^{n_{\text{prob}}}}{(K_{\text{prob}})^{n_{\text{prob}}} + C^{n_{\text{prob}}}}. \tag{A.11}$$

Here $P_{\max}$ is the maximum probability per unit time, $n_{\text{prob}}$ is a Hill coefficient, and $K_{\text{prob}}$ is the $Ca^{2+}$ sensitivity parameter [18].

If a CRU $\hat{\mathbf{x}}$ begins to fire at a spark time $t_0 \in T_{\text{CRU}}$, then the CRU will continue to fire for all $t \in [t_0, t_0 + T_{\text{open}}]$. Hence, in time, we can model an individual calcium spark event as an indicator function over the interval $[t_0, t_0 + T_{\text{open}}]$.

We introduce a function $S_{\hat{\mathbf{x}}}(C, t; T_{\text{open}})$ to have these properties, and its evaluation is outlined below:

- $S_{\hat{\mathbf{x}}} = 0$ until the probability that $\hat{\mathbf{x}}$ will fire is larger than a uniformly distributed random number.
- If the CRU at $\hat{\mathbf{x}}$ has opened at some time $t_0 \in T_{\text{CRU}}$, then $S_{\hat{\mathbf{x}}} = 1$ for $t \in [t_0, t_0 + T_{\text{open}}]$ and $S_{\hat{\mathbf{x}}} = 0$ otherwise.

By the summation of sparks at all possible locations $\hat{\mathbf{x}} \in X_{\text{CRU}}$, the function $\sigma(C, \mathbf{x}, t; T_{\text{open}})$ in (4) gives the superposition of all spark events that are active at time $t$.

## References

[1] R.A. Adams, Sobolev Space, Academic Press, New York, 1975.

[2] A.B. Andreev, R.D. Lazarov, Lumped mass finite element method for parabolic and eigenvalue problems, Math. Balkanica 2 (1988) 85–92.

[3] P.H. Backx, P.P.D. Tombe, J.H.K.V. Denn, B.J.M. Mulder, H.E.D.J.T. Keurs, A model of propagating calcium-induced calcium release mediated by calcium diffusion, J. Gen. Physiol. 93 (1989) 963–977.

[4] D.M. Bers, Excitation-Contraction Coupling and Cardiac Contractile Force, Kluwer Academic Publishers, Dordrecht, 2001.

[5] D. Braess, Finite Elements, 2nd Edition, Cambridge University Press, Cambridge, 2001.

[6] C.M. Chen, V. Thomée, The lumped mass finite element method for a parabolic problem, J. Austral. Math. Soc. Ser. B 26 (1985) 329–354.

[7] H. Cheng, M.R. Lederer, W.J. Lederer, M.B. Cannell, Calcium sparks and $[Ca^{2+}]_i$ waves in cardiac myocytes, Amer. J. Physiol. 270 (1996) C148–C159.

[8] M. Endo, M. Tanaka, Y. Ogawa, Calcium induced release of calcium from the sarcoplasmic reticulum of skinned skeletal muscle fibers, Nature 228 (1970) 34–36.

[9] A. Fabiato, F. Fabiato, Excitation-contraction coupling of isolated cardiac fibers with disrupted or closed sarcomeres: calcium-dependent cyclie and tonic contractions, Circ. Res. 31 (1972) 293–307.

[10] L.E. Ford, R.J. Podosky, Regenerative calcium release within muscle cells, Science 167 (1970) 58–59.

[11] C. Franzini-Armstrong, F. Protasi, V. Ramesh, Shape, size, and distribution of $Ca^{2+}$ release units and couplons in skeletal and cardiac muscles, Biophys. J. 77 (1999) 1528–1539.

[12] C.A. Hall, T.A. Porsching, Numerical Analysis of Partial Differential Equations, Prentice-Hall, Englewood Cliffs, NJ, 1990.

[13] A.L. Hanhart, Coarse-grained parallel solution of a three-dimensional model for calcium concentration in human heart cells, M.S. Thesis, University of Maryland, Baltimore County, 2002.
[14] A. Iserles, A First Course in the Numerical Analysis of Differential Equations, Cambridge Texts in Applied Mathematics, Cambridge University Press, Cambridge, 1996.
[15] L.T. Izu, Y. Chen-Izu, C.W. Ward, C. Balke, Unpublished observations.
[16] L.T. Izu, J.R.H. Mauban, C.W. Balke, W.G. Wier, Large currents generate cardiac $Ca^{2+}$ sparks, Biophys. J. 80 (2001) 88–102.
[17] L.T. Izu, W.G. Wier, C.W. Balke, Theoretical analysis of the $Ca^{2+}$ spark amplitude distribution, Biophys. J. 75 (1998) 1144–1162.
[18] L.T. Izu, W.G. Wier, C.W. Balke, Evolution of cardiac calcium waves from stochastic calcium sparks, Biophys. J. 80 (2001) 103–120.
[19] J. Keizer, G.D. Smith, S. Konce-Dawson, J.E. Pearson, Saltatory propagation of $Ca^{2+}$ waves by $Ca^{2+}$ sparks, Biophys. J. 75 (1998) 595–600.
[20] Y.-Y. Nie, V. Thomée, A lumped mass finite-element method with quadrature for a non-linear parabolic problem, IMA J. Numer. Anal. 5 (1985) 371–396.
[21] I. Parker, W.-J. Zang, W.G. Wier, $Ca^{2+}$ sparks involving multiple $Ca^{2+}$ release sites along $Z$-lines in rat heart cells, J. Physiol. 497 (1996) 31–38.
[22] A. Quarteroni, A. Valli, Numerical Approximation of Partial Differential Equations, in: Springer Series in Computational Mathematics, Vol. 23, Springer, Berlin, 1994.
[23] L.F. Shampine, M.W. Reichelt, The MATLAB ODE suite, SIAM J. Sci. Comput. 18 (1) (1997) 1–22.
[24] M.D. Stern, Theory of excitation-contraction coupling in cardiac muscle, Biophys. J. 63 (1992) 497–517.
[25] V. Thomée, Galerkin Finite Element Methods for Parabolic Problems, in: Springer Series in Computational Mathematics, Vol. 25, Springer, Berlin, 1997.