COMM. APP. MATH. AND COMP. SCI.
Vol. 5, No. 1, 2010

A CUT-CELL METHOD FOR SIMULATING SPATIAL MODELS OF BIOCHEMICAL REACTION NETWORKS IN ARBITRARY GEOMETRIES

WANDA STRYCHALSKI, DAVID ADALSTEINSSON AND TIMOTHY ELSTON

Cells use signaling networks consisting of multiple interacting proteins to respond to changes in their environment. In many situations, such as chemotaxis, spatial and temporal information must be transmitted through the network. Recent computational studies have emphasized the importance of cellular geometry in signal transduction, but have been limited in their ability to accurately represent complex cell morphologies. We present a finite volume method that addresses this problem. Our method uses Cartesian-cut cells in a differential algebraic formulation to handle the complex boundary dynamics encountered in biological systems. The method is second-order in space and time. Several models of signaling systems are simulated in realistic cell morphologies obtained from live cell images. We then examine the effects of geometry on signal transduction.

1. Introduction

Cells must be able to sense and respond to external environmental cues. Information about external signals, such as hormones or growth factors, is transmitted by signaling pathways to the cellular machinery required to generate the appropriate response. Defects in these pathways can lead to diseases, such as cancer, diabetes, and heart disease. Therefore, understanding how intracellular signaling pathways function is not only a fundamental problem in cell biology, but also important for developing therapeutic strategies for treating disease.

In many pathways, proper signal transduction requires that both the spatial and temporal dynamics of the system be tightly regulated [10]. For example, recent experiments have revealed spatial gradients of protein activation in migrating cells [19]. Mathematical models can be used to elucidate the control mechanisms used to regulate the spatiotemporal dynamics of signaling pathways, and recent computational studies emphasize the importance of cellular geometry in signaling networks [16; 17; 23]. For computational simplicity, many of these investigations

MSC2000: 92-08, 65M06.
Keywords: systems biology, numerical methods, reaction-diffusion equation.

assume idealized cell geometries [12; 16], whereas others approximate irregularly shaped cells using a "staircase" representation of the cell membrane [22].

Both finite element and finite volume methods have been used to simulate spatial models of biochemical reaction networks [16; 22; 23; 31]. The most common finite volume algorithm to simulate reaction networks in two and three dimensions is the virtual cell algorithm [22]. Cellular geometries are represented by staircase curves. The authors note that the approximation of fluxes across membranes leads to a decrease in the spatial accuracy of the numerical method to first-order. The temporal accuracy of algorithm in [22] is also limited to first-order. For finite element methods, which typically require a triangulation of the computational domain, grid generation can be a challenge. This becomes especially true if the boundaries of the computational domain are moving.

To overcome the issues of accurate boundary representation and grid generation, we developed a finite volume method that utilizes a Cartesian grid. Our numerical scheme is based on a cut-cell method that accurately represents the cell boundary using a piecewise-linear approximation. The method presented here extends the results on embedded boundary methods to systems of nonlinear reaction diffusion equations with arbitrary boundary conditions. Embedded boundary methods [4; 5; 9; 13; 15; 25] have been used to solve Poisson's equation [9] and the heat equation [15; 25] with homogeneous Dirichlet and Neumann boundary conditions as well as hyperbolic conservation laws [5]. Surface diffusion of one species in three dimensions was simulated with an embedded boundary discretization in [24]. We also offer an alternative formulation to embedded boundary methods for handling the temporal update. In our formulation, the boundary conditions form a system of nonlinear algebraic equations that can be solved with existing differential algebraic equation solvers. We provide a novel use of DASPK (Differential Algebraic Solver Pack) [2] as a time integrator for the finite volume method. The embedded boundary spatial discretization combined with the differential algebraic formulation allows us to achieve second-order accuracy in space and time. Our method also provides an appropriate framework for addressing moving boundary problems using level set methods [18; 26].

The remainder of the article is organized as follows. In Section 2, we describe the mathematical formulation and governing equations. In Section 3, we describe the numerical scheme, the flux based formulation, and coupling reactions terms on the interior and boundary with spatial terms to form one interconnected system. We also outline how the system is adapted for the DASPK numerical solver [2]. In Section 4, we verify the numerical method. The computed solution is compared to a known solution on a circular domain. Additionally, we perform grid refinements of the computed solution on a well resolved grid to show convergence in the absence of an exact solution. The numerical method is then demonstrated on a more physically

relevant domain with an irregular domain. Finally, we simulate a biologically relevant reaction-diffusion model on a very irregular domain.

2. Mathematical formulation

Spatial models of biochemical reaction networks are typically represented using partial differential equations consisting of reaction and diffusion terms. Active transport, driven by molecular motors, also occurs within cells. This effect can be included in our numerical scheme by the use of advection terms and will be addressed in future work. For simplicity we restrict ourselves to two spatial dimensions x and y. For a given chemical species, the reaction terms encompass processes such as activation, degradation, protein modifications and the formation of molecular complexes. These reactions typically include nonlinear terms, such as those arising from Michaelis–Menten kinetics. In a system consisting of n chemical species, the concentration of the ith species c_i evolves in space and time according to the equation

$$\frac{\partial c_i}{\partial t} = -\nabla \cdot \mathbf{J} + f_i(\boldsymbol{c}), \tag{1}$$

where $\mathbf{J} = -D_i \nabla c_i$ is the flux density, D_i is the diffusion coefficient, and the function $f_i(\boldsymbol{c})$ models the reactions within the cell that affect c_i. The elements of the vector $\boldsymbol{c}$ are the concentrations of the n chemical species. Reactions also may occur on the cell membrane yielding nonlinear conditions on the boundary $\partial\Omega$:

$$-D\vec{n} \cdot \nabla c_i|_{\partial\Omega} + g(\boldsymbol{c})|_{\partial\Omega} = 0. \tag{2}$$

Equations (1) and (2) are solved subject to appropriate initial conditions $c_i(x, y, 0)$ for each species in the system.

3. Numerical methods

Our goal is to develop a simulation tool that can accurately and efficiently solve spatial models of signaling and regulatory pathways in realistic cellular geometries. We obtain the computational domain from live-cell images. The model equations are solved on a Cartesian grid by discretizing the Laplacian operator, which models molecular diffusion, with a finite volume method.

3.1. ***Computational domain.*** Figure 1 shows a gray-scale image of a mouse fibroblast [19]. Because the original image is noisy, the image was smoothed by convolving it twice with the standard five-point Gaussian smoothing filter. After smoothing, a suitable thresholding value was picked, and the front was computed by an iso-contour finder. A signed distance function is constructed with the smoothened boundary using fast marching methods [14]. The zero-level set of the signed distance

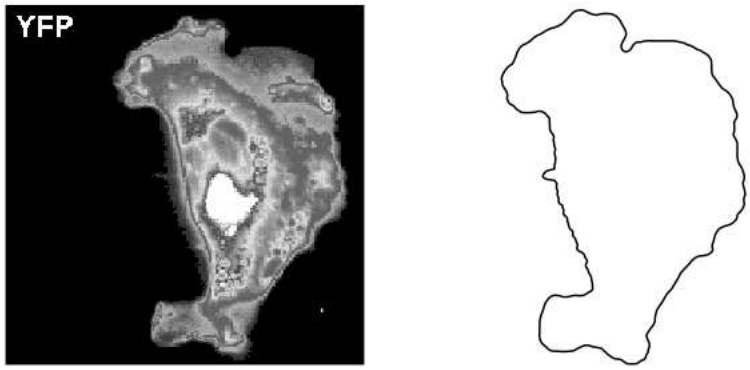


Figure 1. Grayscale image of a mouse fibroblast taken from supplemental data in [19] (left) and the smoothened boundary (right).

function yields piecewise linear segments used to define cut cells (Figure 2). Implicit representation of irregular boundaries has also been proposed in [4; 13].

3.2. ***Discretization of the spatial operator.*** We utilize a Cartesian grid-based, finite volume algorithm originally presented in [9] to discretize the diffusion operator arising from (1). Finite volume methods store the average value of the concentration over a computational grid cell at the location (i, j). That is,

$$\bar{c}_{i,j} = \frac{1}{V_{i,j}} \iint_{V_{i,j}} c(x, y)\, dV, \tag{3}$$

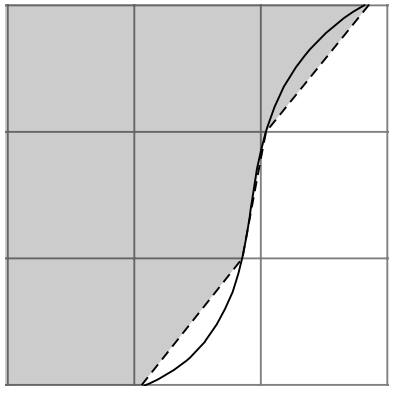


Figure 2. Computational boundary (dashed line) with an assumed higher-order representation of the cell boundary drawn as a solid line.

where $V_{i,j}$ is the volume of the (i, j) grid cell. Inserting (3) into (1) produces

$$\frac{\partial \bar{c}_{i,j}}{\partial t} - \overline{f(c)}_{i,j} = -\frac{1}{V_{i,j}} \iint_{V_{i,j}} \nabla \cdot \mathbf{J}\, dV. \tag{4}$$

The divergence theorem allows us to convert the above volume integral into a surface integral,

$$\frac{\partial \bar{c}_{i,j}}{\partial t} - \overline{f(c)}_{i,j} = -\frac{1}{V_{i,j}} \int_{\partial V_{i,j}} (\mathbf{J} \cdot \vec{n})\, dS. \tag{5}$$

For interior grid cells, we have

$$\frac{\partial \bar{c}_{i,j}}{\partial t} - \overline{f(c)}_{i,j} = -\frac{1}{V_{i,j}} \left[\int_{y_{j-1/2}}^{y_{j+1/2}} \left(J_x(x_{i+1/2}, y) - J_x(x_{i-1/2}, y) \right) dy + \int_{x_{i-1/2}}^{x_{i+1/2}} \left(J_y(x, y_{j+1/2}) - J_y(x, y_{j-1/2}) \right) dx \right], \tag{6}$$

where $J_x = -D(\partial c/\partial x)$ and $J_y = -D(\partial c/\partial y)$. Approximation of the integrals in (6) with the midpoint rule yields

$$\frac{\partial c_{i,j}}{\partial t} - f(c_{i,j}) \approx -\frac{1}{V_{i,j}} \big[\Delta y (J_x(x_{i+1/2}, y_j) - J_x(x_{i-1/2}, y_j)) + \Delta x \big(J_y(x_i, y_{j+1/2}) - J_y(x_i, y_{j-1/2}) \big) \big]. \tag{7}$$

By approximating the gradient terms with centered differences, we arrive at the standard five-point Laplacian. Therefore in computational grid cells with volume $V_{i,j} = 1$, the finite volume stencil is the same as the five-point Laplacian approximation.

The cut-cell method generalizes as follows. The boundary of the computational domain is approximated as a piecewise linear segments (Figure 2, dashed line), and grid cells that the boundary passes through are referred to as *cut cells*. The volume of a cut cell is computed by recasting the volume integral as a boundary integral:

$$V_{i,j} = \iint_{V_{i,j}} dV = \iint_{V_{i,j}} \nabla \cdot \left(\frac{x}{2}, \frac{y}{2} \right) dV = \int_{\partial V_{i,j}} \left(\left(\frac{x}{2}, \frac{y}{2} \right) \cdot \vec{n} \right) dS, \tag{8}$$

where $\vec{n}$ is the normal vector to the surface. The integral on the right can be computed exactly for the polygon. Each segment is evaluated, then summed. The center of mass can also be computed using a boundary integral, for example:

$$\iint_{V_{ij}} x dV = \iint_{V_{ij}} \nabla \cdot \left(\frac{x^2}{2}, 0 \right) dV = \int_{\partial V_{i,j}} \left(\left(\frac{x^2}{2}, 0 \right) \cdot \vec{n} \right) dS. \tag{9}$$

We initialize cut cells with values computed at the centroid as in [15].

Next, we construct the integral on the right side of (5) for a cut cell. In general, there are up to five surface integrals to approximate. Let $a_{l,m} \in [0, 1]$ represent the fraction of each of the four cell edges covered by the cut cell and a_f be the length of the line segment representing the boundary. Then (7) becomes

$$\frac{\partial c_{i,j}}{\partial t} - f(c_{i,j}) \approx -\frac{1}{V_{i,j}}\Big[\Delta y\left(a_{i+1/2,j}J_x(x_{c_{i+1/2}}, y_j) - a_{i-1/2,j}J_x(x_{c_{i-1/2}}, y_j)\right) + \Delta x\left(a_{i,j+1/2}J_y(x_i, y_{c_{j+1/2}}) - a_{i,j-1/2}J_y(x_i, y_{c_{j-1/2}})\right) + a^f J_f\Big]. \quad (10)$$

The notation $(x_{c_{i\pm1/2}}, y_j)$ indicates the midpoint of partially covered $(x_{i\pm1/2}, y_j)$ face. Let

$$F_{i\pm1/2,j} = -a_{i\pm1/2,j}\Delta y J_x(x_{c_{i\pm1/2}}, y_j), \quad F_{i,j\pm1/2} = -a_{i,j\pm1/2}\Delta x J_y(x_i, y_{c_{j\pm1/2}}).$$

With this notation, we rewrite the previous equation as

$$\frac{\partial c_{i,j}}{\partial t} - f(c_{i,j}) \approx \frac{1}{V_{i,j}}\left(F_{i+1/2,j} - F_{i-1/2,j} + F_{i,j+1/2} - F_{i,j-1/2} - F^f_{i,j}\right). \quad (11)$$

We refer to the Fs as the surface fluxes (Figure 3). On a full edge with $a_{l,m} = 1$ the surface flux is calculated with centered differences. For example, in Figure 3, we have

$$F_{i-1/2,j+1} = D\Delta y\frac{c_{i,j+1} - c_{i-1,j+1}}{\Delta x}. \quad (12)$$

The flux gradient across a cut edge, for example $(x_{i-1/2}, y_j)$, is approximated by a linear interpolation of two gradients, which are computed by centered differences. A linear interpolation formula between two points y_1 and y_2 as a function of a

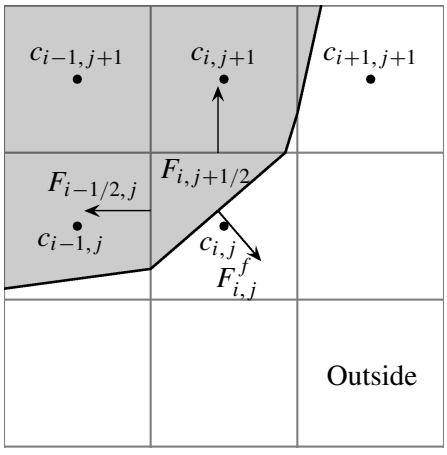


Figure 3. Diagram of fluxes for cut cells where shaded boxes indicate cells that are inside the boundary.

parameter $\mu \in [0, 1]$ is

$$y^I = (1-\mu)y_1 + \mu y_2. \tag{13}$$

In the case of a cut-cell edge, $\mu = (1 + a_{l,m})/2$. For example, to construct $F_{i-1/2,j}$ in Figure 3, the gradient at $(x_{i-1/2}, y_j)$ and $(x_{i-1/2}, y_{j+1})$ is used:

$$F_{i-1/2,j} = Da_{i-1/2,j}\,\Delta y\left[\frac{(1+a_{i-1/2,j})}{2}\frac{(c_{i,j}-c_{i-1,j})}{\Delta x} + \frac{(1-a_{i-1/2,j})}{2}\frac{(c_{i,j+1}-c_{i-1,j+1})}{\Delta x}\right]. \tag{14}$$

To calculate the flux through a boundary, for example, $F^f_{i,j}$, we compute the gradient along a line normal to the boundary and centered at the boundary midpoint. To find function values on the normal line, we interpolate using three equally spaced cell-centered points (Figure 4). If the normal line is oriented with an angle of $\pi/4 < |\theta| < 3\pi/4$ relative to the horizontal grid lines, horizontal grid points are used to compute the values on the line. Otherwise vertical points are used.

The two points computed along the normal line and the value on the boundary are then used to construct a quadratic polynomial. The concentration gradient is calculated by differentiating the quadratic polynomial and evaluating the result at the boundary point c^f:

$$G^f = \frac{1}{d_2 - d_1}\left[\frac{d_2}{d_1}(c^f - c_1^I) - \frac{d_1}{d_2}(c^f - c_2^I)\right], \tag{15}$$

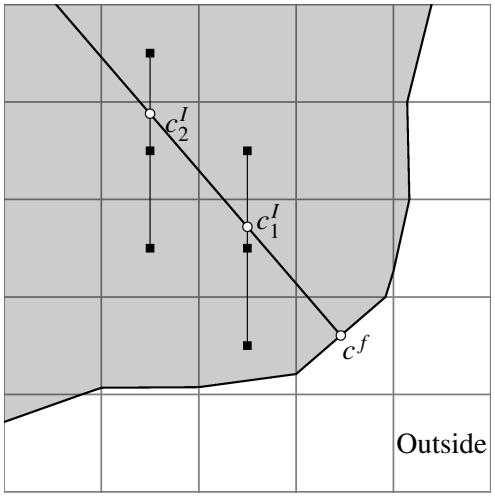


Figure 4. White circles indicate interpolated values that depend on the grid-based values.

where c_1^I and c_2^I are the interpolated values along the normal line and d_1 and d_2, respectively, are the distances of these two points from the boundary. The flux F_{ij}^f in (11) is calculated by multiplying G^f by the area of the cut-cell edge a_f and the diffusion coefficient D. The discretization of the boundary condition (2) becomes the algebraic equation

$$DG^f + g(\boldsymbol{c}^f) = 0. \tag{16}$$

Because all gradients are constructed with second-order methods, the overall discretization scheme is second-order in space. Further discussion on the accuracy of the spatial discretization scheme can be found in [9].

3.3. ***Time discretization.*** Spatial discretizations of (1) and (2) are treated as a differential-algebraic system of nonlinear equations (DAE). The general form for a differential-algebraic system is

$$F(t, \boldsymbol{C}, \boldsymbol{C}') = \boldsymbol{0}, \tag{17}$$

where $\boldsymbol{C}$ is an $(N_g + N_b) \times 1$ vector. The first N_g entries are associated with Cartesian grid based values in the differential-algebraic system from the discretization of (1) for the chemical species concentrations. These entries have an explicit time derivative term. The N_b remaining entries arise from discretizing the boundary conditions given in (2) that form algebraic constraints. As noted in [1], reformulating algebraic constraints in a nonlinear model as a system of ordinary differential equations may be time consuming or impossible. DAEs formed by reaction-diffusion equations described in Section 2 are semiexplicit, index-1 systems of the form

$$\begin{aligned} \boldsymbol{C}_1' &= F_1(\boldsymbol{C}_1, \boldsymbol{C}_2, t), \\ \boldsymbol{0} &= F_2(\boldsymbol{C}_1, \boldsymbol{C}_2, t). \end{aligned} \tag{18}$$

$\boldsymbol{C}_1$ represents the first N_g variables and $\boldsymbol{C}_2$ represents the remaining N_b variables. Equation (18) is an index-1 system if and only if $\partial F_2 / \partial \boldsymbol{C}_2$ is nonsingular [1]. Ordinary differential equations are index-0.

We use the DASPK solver described in [2] as a time integrator for our differential algebraic system. In DAPSK, backward differentiation formulas (BDF) discretize the time derivative in (17). A basic implicit method with a backward Euler time discretization of (17) is given by,

$$F\left(t^{n+1}, \boldsymbol{C}^{n+1}, \frac{\boldsymbol{C}^{n+1} - \boldsymbol{C}^n}{\Delta t}\right) = \boldsymbol{0}, \tag{19}$$

where n is defined such that $t^n = n\Delta t$. Newton's method can be used to solve the resulting nonlinear equations for $\boldsymbol{C}^{n+1}$,

$$\boldsymbol{C}_{m+1}^{n+1} = \boldsymbol{C}_m^{n+1} - \left(\frac{\partial F}{\partial \boldsymbol{C}} + \frac{1}{\Delta t}\frac{\partial F}{\partial \boldsymbol{C}'}\right)\Big|_{\boldsymbol{C}_m^{n+1}}^{-1} F\left(t^{n+1}, \boldsymbol{C}_m^{n+1}, \frac{\boldsymbol{C}_m^{n+1} - \boldsymbol{C}^n}{\Delta t}\right), \tag{20}$$

where m is the index of the Newton iteration. In order to achieve higher-order temporal accuracy, a higher-order interpolating polynomial is used to approximate the time derivative.

In a k-step BDF, the time derivative is replaced by the derivative of an interpolating polynomial at $k+1$ times $t^{n+1}, t^n \ldots, t^{n+1-k}$ evaluated at t^{n+1}. If we approximate the derivative using a kth order stencil using k known values and the implicit value $\boldsymbol{C}^{n+1}$ we get

$$\boldsymbol{C}'^{\,n+1} \approx \frac{1}{\Delta t}\left(\alpha_0 \boldsymbol{C}^{n+1} + \sum_{i=1}^{k} \alpha_i \boldsymbol{C}^{n+1-i}\right). \tag{21}$$

The coefficients of the BDF are given by α_is. In DAPSK, these values are coefficients of the Newton divided difference interpolating polynomial [1]. The default order of the BDF method in DASPK is five.

The new implicit equation to be solved at each time step is

$$F\left(t^{n+1}, \boldsymbol{C}^{n+1}, \frac{1}{\Delta t}\left(\alpha_0 \boldsymbol{C}^{n+1} + \sum_{i=1}^{k} \alpha_i \boldsymbol{C}^{n+1-i}\right)\right) = \mathbf{0}. \tag{22}$$

This can be rewritten as

$$F\left(t^{n+1}, \boldsymbol{C}^{n+1}, \frac{\alpha_0}{\Delta t}\boldsymbol{C}^{n+1} + \mathbf{v}\right) = \mathbf{0}, \tag{23}$$

where $\mathbf{v}$ is a vector that depends on previously computed time values. Details of choosing step-size, starting selection and variable order strategies are found in [1]. The nonlinear system is solved with a modified Newton's method, given by

$$\boldsymbol{C}_{m+1}^{n+1} = \boldsymbol{C}_m^n - \zeta\left(\frac{\partial F}{\partial \boldsymbol{C}} + \frac{\alpha_0}{\Delta t}\frac{\partial F}{\partial \boldsymbol{C}'}\right)\Big|^{-1}_{\boldsymbol{C}_m^{n+1}} F\left(t^{n+1}, \boldsymbol{C}_m^{n+1}, \frac{\alpha_0}{\Delta t}\boldsymbol{C}_m^{n+1} + \mathbf{v}\right), \tag{24}$$

where ζ is a constant chosen to speed up convergence and m is the iteration index. Each step of the Newton iteration requires inverting the matrix

$$A = \frac{\partial F}{\partial \boldsymbol{C}} + \frac{\alpha_0}{\Delta t}\frac{\partial F}{\partial \boldsymbol{C}'}. \tag{25}$$

We store this matrix in sparse triple format, and use routines from SPARSKIT [20] to solve the linear system iteratively. The generalized minimal residual (GMRES) method [21] with an incomplete LU (ILU) preconditioner is used to solve the linear system.

4. Results

4.1. ***Convergence tests.*** To demonstrate the accuracy of our method on a domain containing all types of cut cells, the convergence of our method is compared against

an exact solution on a circle. The exact solution to the diffusion equation with a zero Dirichlet boundary condition can be found in terms of Bessel functions. Let λ denote the first root of the Bessel function $J_0(x)$, and r be the radius of the circle centered at the point (0.5, 0.5). Then the expression

$$f(x, y, t) = \exp\left(-D\left(\frac{\lambda}{r}\right)^2 t\right) J_0\left(\lambda \frac{\sqrt{(x-0.5)^2 + (y-0.5)^2}}{r}\right) \tag{26}$$

is an exact solution to the diffusion equation.

For this example, the error is computed as the difference between computed solution values on a triangular grid subtracted from the exact solution. The grids for both two dimensional triangular meshes were the same. For purposes of generating the following convergence data, the spatial steps Δx and Δy are equal and set to $1/N$, where N is the grid size. The time step Δt is set to $\Delta x/4$ (that is, it is refined with the spatial step size). Because DASPK uses variable time steps, the output at the time step requested might be interpolated as described in [1]. A time series of the truncation error in the infinity norm over time is shown in Figure 5. Table 1 lists the truncation error at the simulation time $t = 0.4$. The convergence rate r is calculated as

$$r = \log \frac{e_1}{e_2} \Big/ \log \frac{\Delta x_1}{\Delta x_2}, \tag{27}$$

where e_1 and e_2 are errors computed in norms with grid spacing Δx_1 and Δx_2. A log-log plot of truncation error as a function of the spatial step is shown in Figure 6. The error was calculated with the computed and exact solutions at the time value of $t = 0.4$. The results of this analysis demonstrate global second-order accuracy of the numerical method.

Next we tested a nonlinear system in which a protein C can exist in two distinct chemical states: active and inactive. The reactions that convert the protein between the two states are assumed to follow Michaelis–Menten kinetics, which describes the kinetics of many enzymatic reactions including phosphorylation and dephosphorylation events [11]. The protein C is deactivated in the interior of the

Grid size	Time step	L^2 norm	r	L^1 norm	r	L^∞ norm	r
50×50	$5.00{\cdot}10^{-3}$	$2.95{\cdot}10^{-4}$	–	$2.61{\cdot}10^{-4}$	–	$5.46{\cdot}10^{-4}$	–
100×100	$2.50{\cdot}10^{-3}$	$4.94{\cdot}10^{-5}$	2.58	$4.32{\cdot}10^{-5}$	2.59	$9.28{\cdot}10^{-5}$	2.56
200×200	$1.25{\cdot}10^{-3}$	$1.05{\cdot}10^{-5}$	2.24	$9.20{\cdot}10^{-6}$	2.23	$2.09{\cdot}10^{-5}$	2.15
400×400	$6.25{\cdot}10^{-4}$	$2.42{\cdot}10^{-6}$	2.11	$2.13{\cdot}10^{-6}$	2.11	$5.42{\cdot}10^{-6}$	1.95

Table 1. The norms and convergence rates for the diffusion equation at the time value of 0.4.

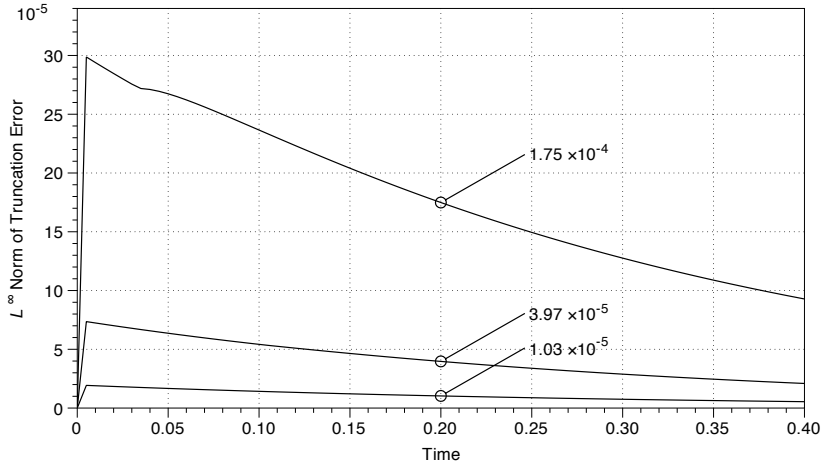


Figure 5. L^∞-norm truncation error for the diffusion equation, with size $N = 100, 200, 400$ (from top to bottom). The time step at each refinement was set to $1/(4N)$.

computational domain according to the following equations:

$$\frac{\partial C_i}{\partial t} = D\Delta C_i + \frac{k_2 C_a}{K_{m2} + C_a}, \quad \frac{\partial C_a}{\partial t} = D\Delta C_a - \frac{k_2 C_a}{k_{m2} + C_a}, \tag{28}$$

where C_i and C_a are the concentrations of inactive and active protein, respectively, k_2 is the maximum deactivation rate, and K_{m2} is the Michaelis constant. Activation

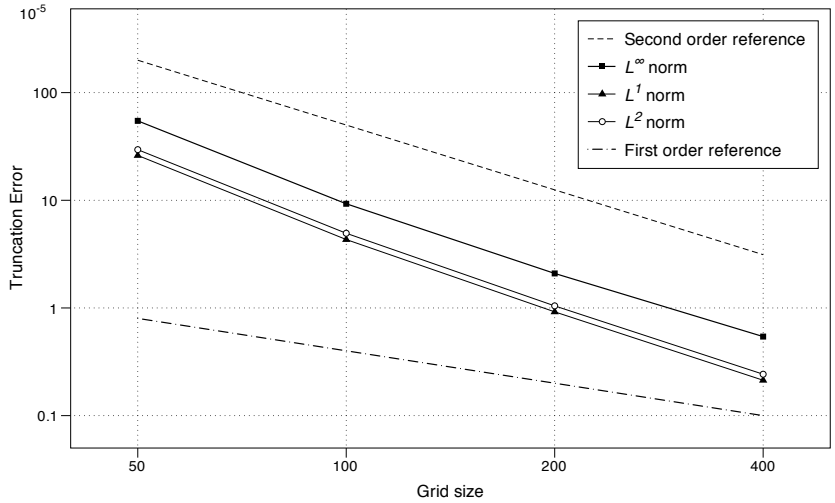


Figure 6. Truncation error for the diffusion equation at the time value of 0.4. The convergence data is the same as in Table 1.

occurs on the boundary, $\partial\Omega$, according to the following boundary conditions:

$$-D\vec{n}\cdot\nabla C_i|_{\partial\Omega} = \frac{k_1 S C_i}{K_{m1}+C_i}\Big|_{\partial\Omega}, \quad -D\vec{n}\cdot\nabla C_a|_{\partial\Omega} = -\frac{k_1 S C_i}{K_{m1}+C_i}\Big|_{\partial\Omega}, \tag{29}$$

where k_1 is the maximum activation rate and K_{m1} is the Michaelis constant. The equations are solved in the domain

$$\Omega(r,\theta) = r \leq 0.3 - 0.09\sin(4\theta). \tag{30}$$

In our simulation, Ω is shifted to the center of the unit box. The initial concentration of inactive protein is assumed to be constant and equal to 1. There is initially no active protein. Figure 7 shows a plot of the active concentration at $t = 0.25$. For visualization purposes, the computational domain and boundary points are triangulated with Triangle [27]. The concentration of the active protein is shown as a cross-section of the two-dimensional geometry at several time values in Figure 7, bottom. The constants (see figure caption) were arbitrarily chosen to generate a gradient. Execution times for Mac Pro desktop computer with dual-core 2.66 GHz Intel Xeon processors for different grid sizes are listed in Table 2.

We compute the error as the difference between successive grid refinements as follows. The truncation error function $E(x, y, t)$ is defined on interior values of the coarser grid. Computed solution values located in coarse grid cut cells are excluded from the domain. This includes some values located in interior points for the more refined grid (Figure 8). The truncation error function is defined as

$$E(x, y, t) = c_{\Delta x}(x, y, t) - c_{\Delta x/2}(x, y, t). \tag{31}$$

The coarse grid values are located in the center of a box defined by four refined grid values. Four refined grid values are averaged and subtracted from one coarse value. Because the time integration is handled implicitly, a different convergence rate of the truncation error in cut cells and boundary values would affect the convergence

Grid size	Time step	Execution time
50×50	$5.000{\cdot}10^{-3}$	1.76 s
100×100	$2.500{\cdot}10^{-3}$	5.81 s
200×200	$1.250{\cdot}10^{-3}$	32.15 s
400×400	$6.250{\cdot}10^{-4}$	203.95 s
800×800	$3.125{\cdot}10^{-4}$	1202.18 s

Table 2. Execution times for the two-species model. The end time of the simulation was $t = 0.5$.

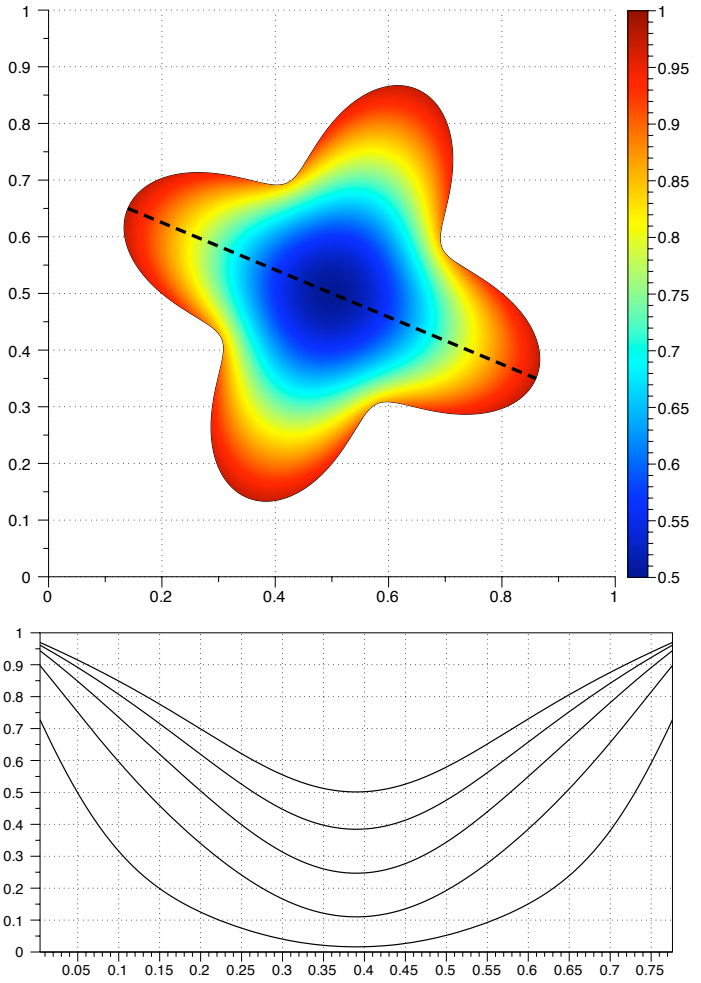


Figure 7. Concentration of the active species C_a at $t = 0.25$ (top) and at evenly spaced time values for $t \in [0, 0.25]$ (bottom) along the section shown with a dashed line in the top figure. Values chosen for the constants: $D = K_{m1} = k_{m2} = 0.2$, $S = k_1 = k_2 = 1.0$.

rate of the truncation error for interior cells. Therefore, by computing the error with interior cells, we are still able to draw conclusions about the order of the method.

Table 3 lists convergence data for the two-species system given by (28) and (29). The data used for calculating the error was taken from computed solutions at the simulation time value of $t = 0.5$. Note that the norms of truncation errors for both C_i and C_a are the same. The system is mass conservative, and the computed solution is also conservative to machine precision. Therefore we only show convergence

Species C_i							
Grid size	Time step	L^2 norm	r	L^1 norm	r	L^∞ norm	r
50×50	$5.000{\cdot}10^{-3}$	–	–	–	–	–	–
100×100	$2.500{\cdot}10^{-3}$	$7.59{\cdot}10^{-4}$	–	$1.67{\cdot}10^{-4}$	–	$1.49{\cdot}10^{-3}$	–
200×200	$1.250{\cdot}10^{-3}$	$1.92{\cdot}10^{-4}$	1.98	$4.44{\cdot}10^{-5}$	1.91	$5.01{\cdot}10^{-4}$	1.57
400×400	$6.250{\cdot}10^{-4}$	$4.57{\cdot}10^{-5}$	2.07	$1.08{\cdot}10^{-5}$	2.04	$1.25{\cdot}10^{-4}$	2.00
800×800	$3.125{\cdot}10^{-4}$	$1.09{\cdot}10^{-5}$	2.07	$2.61{\cdot}10^{-6}$	2.05	$3.12{\cdot}10^{-5}$	2.00
Species C_a							
Grid size	Time step	L^2 norm	r	L^1 norm	r	L^∞ norm	r
50×50	$5.000{\cdot}10^{-3}$	–	–	–	–	–	–
100×100	$2.500{\cdot}10^{-3}$	$7.59{\cdot}10^{-4}$	–	$1.67{\cdot}10^{-4}$	–	$1.49{\cdot}10^{-3}$	–
200×200	$1.250{\cdot}10^{-3}$	$1.92{\cdot}10^{-4}$	1.98	$4.44{\cdot}10^{-5}$	1.91	$5.01{\cdot}10^{-4}$	1.57
400×400	$6.250{\cdot}10^{-4}$	$4.57{\cdot}10^{-5}$	2.07	$1.08{\cdot}10^{-5}$	2.04	$12.5{\cdot}10^{-4}$	2.00
800×800	$3.125{\cdot}10^{-4}$	$1.09{\cdot}10^{-5}$	2.07	$2.61{\cdot}10^{-6}$	2.05	$3.12{\cdot}10^{-5}$	2.00

Table 3. Norms and convergence rates for the two-species model at the time value of 0.5.

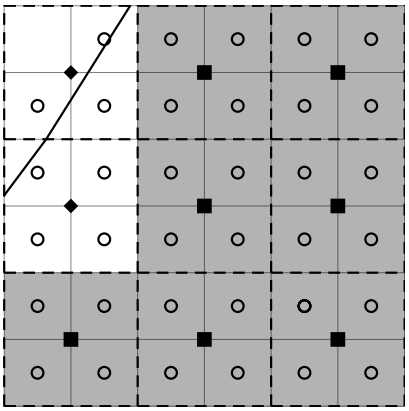


Figure 8. Interior grid cells on the coarser grid (dashed lines) are shaded. Square- and diamond-filled points indicate locations of cell-centered values on the coarse grid. Values associated with diamond-grid points represent cut cells for the coarser grid. Coarse and refined values in these cut cells are not used in the averaging scheme. The refined grid is indicated by solid lines. Circles mark the cell centers of the refined grid cells. Four-refined point values are averaged and compared to the square point on the coarse grid.

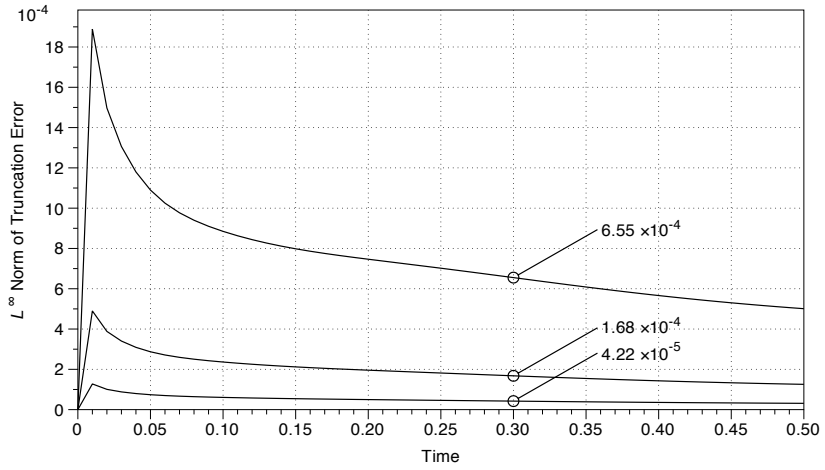


Figure 9. L^∞-norm truncation error for species C_i for the reaction-diffusion equation. The values for the top plot were computed by subtracting the solution at grid size $N = 200$ from the one at $N = 100$ (see text). The middle plot was calculated with $N = 200$ and $N = 400$, and the bottom one with $N = 400$ and $N = 800$.

figures for species C_i. The truncation error for species C_i computed in the infinity norm as a function of time is listed in Figure 9. A log-log plot of the truncation error as a function of the grid size is listed in Figure 10. From this analysis, we conclude second-order accuracy.

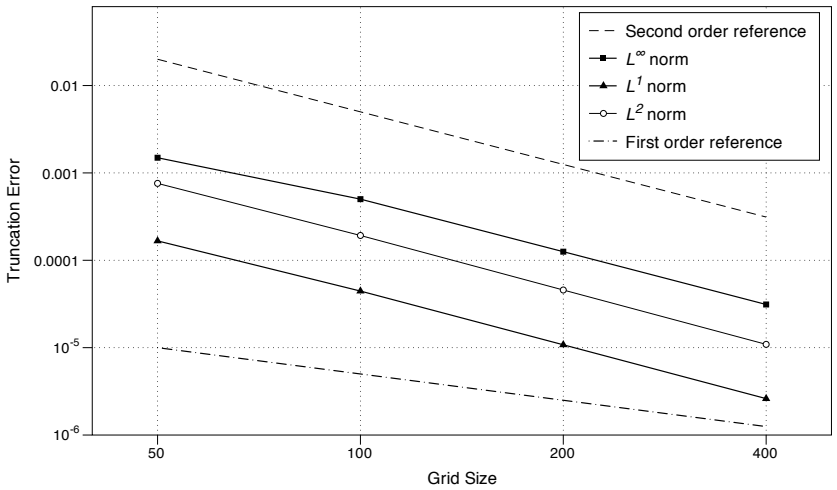


Figure 10. Truncation error for the C_i at the time value of 0.5. The convergence data is the same as that in Table 3.

4.2. ***A two-compartment model.*** In this model, we have two compartments: cytoplasm and nucleus. The cellular geometry was taken from a yeast cell undergoing chemotrophic growth in the direction of a pheromone gradient [8]. Proteins involved in the pheromone response pathway are known to localize on the plasma membrane, the nucleus, and in the cytosol [7]. The nucleus is modeled as a circle located toward the front of the cell. Because yeast cells are three dimensional, we model the top view of the cell as in [6], where membrane-bound species are located in the interior of the computational domain but are assumed to diffuse slower than cytosolic forms.

The model consists of two species, A and C, with inactive and active forms. Protein C is allowed to enter and exit the nucleus, whereas protein A is restricted to the cytoplasm (Figure 11, top).

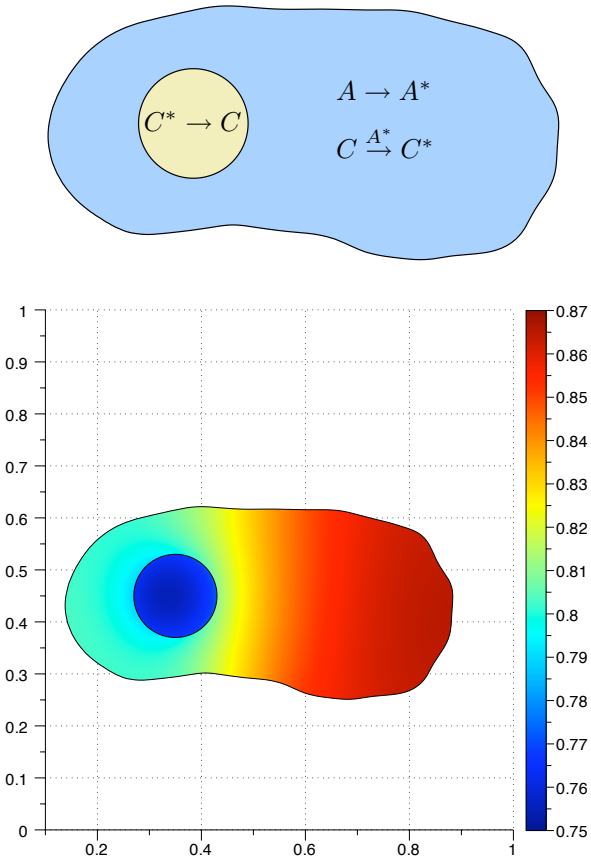


Figure 11. Two-compartment model. Top: reactions and species in the two-compartment model. Bottom: steady-state concentration values for active C species in the cytoplasm and nucleus.

Initially both A and C are in their inactive forms. At the beginning of the simulation, the reaction rate for the activation of A, k_0, is instantaneously increased from 0 to 1. This is meant to model the cell receiving an external signal. Once A is activated it is assumed to interact with the cell membrane, causing a reduction in the protein's diffusion coefficient [30]. The active form of A can then activate protein C. The active form of C is only deactivated within the nucleus. This simple model captures some of the signaling events that occur during the pheromone response of yeast [28]. If we denote the concentration of a chemical species with brackets, the equations for the cytoplasmic species are:

$$\begin{aligned} \frac{\partial [A_c]}{\partial t} &= D_1 \Delta [A_c] - k_0 [A_c], & \frac{\partial [A_c^*]}{\partial t} &= D_2 \Delta [A_c^*] + k_0 [A_c], \\ \frac{\partial [C_c]}{\partial t} &= D_1 \Delta [C_c] - k_1 [A_c^*][C_c], & \frac{\partial [C_c^*]}{\partial t} &= D_1 \Delta [C_c^*] + k_1 [A_c^*][C_c], \end{aligned} \tag{32}$$

where the asterisks denote the active form of the protein, D_1 is the diffusion coefficient in the cytoplasm, D_2 is diffusion coefficient in the membrane, and the ks represent the reaction rates. Subscripts indicate cytosolic and nuclear species. The boundary conditions at the cell membrane $\partial\Omega_1$ are no flux for all chemical species. The nuclear boundary conditions for A species are also no flux, whereas C species are allowed to move through the nuclear membrane $\partial\Omega_2$ and satisfy the conditions

$$\begin{aligned} -D_1(\vec{n} \cdot \nabla [C_c])|_{\partial\Omega_2} &= -k_2([C_n] - [C_c])|_{\partial\Omega_2}, \\ -D_1(\vec{n} \cdot \nabla [C_c^*])|_{\partial\Omega_2} &= -k_2([C_n^*] - [C_c^*])|_{\partial\Omega_2}, \\ -D_1(\vec{n} \cdot \nabla [C_n])|_{\partial\Omega_2} &= k_2([C_n] - [C_c])|_{\partial\Omega_2}, \\ -D_1(\vec{n} \cdot \nabla [C_n^*])|_{\partial\Omega_2} &= k_2([C_n^*] - [C_c^*])|_{\partial\Omega_2}. \end{aligned} \tag{33}$$

Nuclear C^* is deactivated according to the equations

$$\frac{\partial [C_n]}{\partial t} = D_1 \Delta [C_n] + k_3 [C_n^*], \quad \frac{\partial [C_n^*]}{\partial t} = D_1 \Delta [C_n^*] - k_3 [C_n^*]. \tag{34}$$

The steady-state spatial distribution of active C is illustrated in Figure 11, bottom. All reaction constants were arbitrarily chosen to be 1, $D_1 = 0.1$, $D_2 = 0.01$, and $\Delta x = 1/200$. The initial values were zero except for $[A_c](x, y, 0) = [C_c](x, y, 0) = 1$. The execution time of the simulation to run from $t = 0$ until $t = 20$ was 150 seconds on Mac Pro desktop computer with dual-core 2.66 GHz Intel Xeon processors.

To verify that the system is close a steady-state solution at $t = 20$, we subtracted the solution of active C in the cytoplasm $[C_c^*]$ for all times from the assumed steady-state solution at the time value of $t = 20$. If the system exponentially converges to the computed solution at $t = 20$, we assume this time value is close to steady-state. Figure 12 shows the infinity norm of the difference between the computed solution

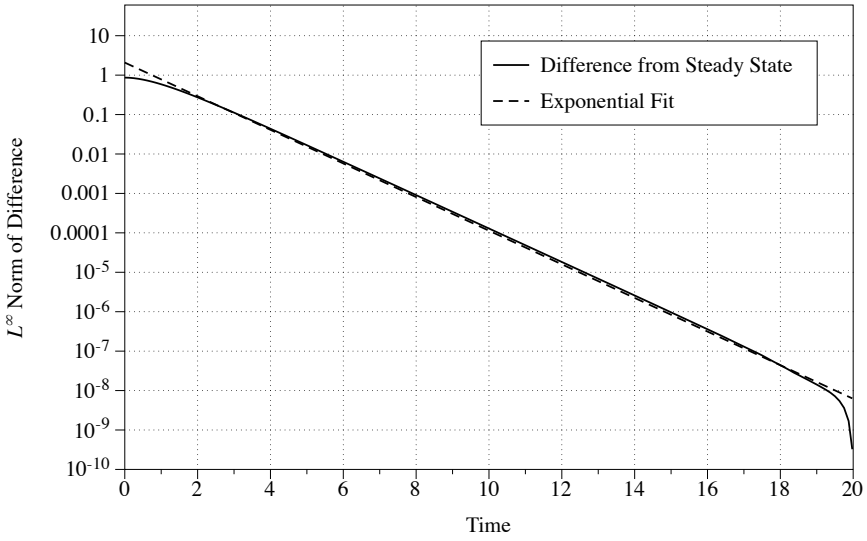
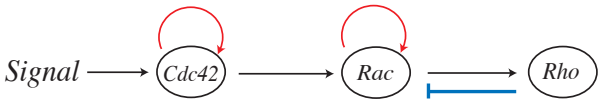


Figure 12. Solid line: norm of difference of the computed solution at the assumed steady-state value at $t = 20$ from computed solution over time. Dashed line: exponential fit.

and the solution at $t = 20$ sampled over time. Based on this data, the system is close to its steady-state solution.

The model simulation suggests a spatial activation gradient can be generated by the position of the nucleus. The inactivation of C in the nucleus leads to a higher concentration of active protein in the rear of the cell in spite of a uniform spatial signal from active A.

4.3. ***Rho family GTPase model.*** The Rho family of GTPases regulates many cellular functions, including polarization and motility. We created a model with three key members of this family, Cdc42, Rac, and Rho; the interactions, based on [3], can be schematically represented as follows:



A more complicated model involving these proteins in one dimension can be found in [6]. As in the previous example, we assume a top view of a three dimensional cell with membrane bound active forms and cytosolic inactive forms of the three proteins. The model has a total of six species. The cell boundary $\partial\Omega$ is taken from supplemental material from [19].

In our model, a uniform extracellular signal triggers the activation of Cdc42 protein on the cell edge,

$$\begin{aligned} -D\vec{n}\cdot\nabla\left[\mathrm{Cdc42}_i\right]\big|_{\partial\Omega} &= \frac{k_1 S\left[\mathrm{Cdc42}_i\right]}{K_{m1}+\left[\mathrm{Cdc42}_i\right]}\Big|_{\partial\Omega}, \\ -D\vec{n}\cdot\nabla\left[\mathrm{Cdc42}_a\right]\big|_{\partial\Omega} &= -\frac{k_1 S\left[\mathrm{Cdc42}_i\right]}{k_{m2}+\left[\mathrm{Cdc42}_i\right]}\Big|_{\partial\Omega}. \end{aligned} \tag{35}$$

In the cell interior, active Cdc42 is inactivated. A positive feedback loop increases the activation of Cdc42,

$$\begin{aligned} \frac{\partial\left[\mathrm{Cdc42}_i\right]}{\partial t} &= D\,\Delta\left[\mathrm{Cdc42}_i\right] + \frac{k_2\left[\mathrm{Cdc42}_a\right]}{K_{m3}+\left[\mathrm{Cdc42}_a\right]} - \frac{k_3\left[\mathrm{Cdc42}_a\right]\left[\mathrm{Cdc42}_i\right]}{K_{m4}+\left[\mathrm{Cdc42}_i\right]}, \\ \frac{\partial\left[\mathrm{Cdc42}_a\right]}{\partial t} &= D\,\Delta\left[\mathrm{Cdc42}_a\right] - \frac{k_2\left[\mathrm{Cdc42}_a\right]}{k_{m5}+\left[\mathrm{Cdc42}_a\right]} + \frac{k_3\left[\mathrm{Cdc42}_a\right]\left[\mathrm{Cdc42}_i\right]}{k_{m6}+\left[\mathrm{Cdc42}_i\right]}. \end{aligned} \tag{36}$$

Rac is activated by Cdc42, and a positive feedback loop increases the concentration of active Rac. Active Rho increases the deactivation of Rac in the cytosol,

$$\begin{aligned} \frac{\partial[\mathrm{Rac}_i]}{\partial t} &= D\,\Delta[\mathrm{Rac}_i] + \frac{(k_4[\mathrm{Rho}_a]+k_5)[\mathrm{Rac}_a]}{K_{m7}+[\mathrm{Rac}_a]} - \frac{(k_6[\mathrm{Cdc42}_a]+k_7[\mathrm{Rac}_a])[\mathrm{Rac}_i]}{K_{m8}+[\mathrm{Rac}_i]}, \\ \frac{\partial[\mathrm{Rac}_a]}{\partial t} &= D\,\Delta[\mathrm{Rac}_a] - \frac{(k_4[\mathrm{Rho}_a]+k_5)[\mathrm{Rac}_a]}{k_{m9}+[\mathrm{Rac}_a]} + \frac{(k_6[\mathrm{Cdc42}_a]+k_7[\mathrm{Rac}_a])[\mathrm{Rac}_i]}{k_{m10}+[\mathrm{Rac}_i]}. \end{aligned}$$

Rho is activated by the active form of Rac and deactivated in the interior,

$$\begin{aligned} \frac{\partial\left[\mathrm{Rho}_i\right]}{\partial t} &= D\,\Delta\left[\mathrm{Rho}_i\right] + \frac{k_8\left[\mathrm{Rho}_a\right]}{K_{m11}+\left[\mathrm{Rho}_a\right]} - \frac{k_9\left[\mathrm{Rac}_a\right]\left[\mathrm{Rho}_i\right]}{K_{m12}+\left[\mathrm{Rho}_i\right]}, \\ \frac{\partial\left[\mathrm{Rho}_a\right]}{\partial t} &= D\,\Delta\left[\mathrm{Rho}_a\right] - \frac{k_8\left[\mathrm{Rho}_a\right]}{k_{m13}+\left[\mathrm{Rho}_a\right]} + \frac{k_9\left[\mathrm{Rac}_a\right]\left[\mathrm{Rho}_i\right]}{k_{m14}+\left[\mathrm{Rho}_i\right]}. \end{aligned} \tag{37}$$

The boundary conditions for Rac and Rho species are no flux. The steady-state distribution is displayed in Figure 13. To achieve these results, a step size $\Delta x = 1/200$ and a diffusion coefficient $D = 0.1$ were used. The reaction constants for the simulation were arbitrarily chosen as follows: $S = k_3 = k_5 = k_7 = 1.0$, $k_2 = k_4 = k_8 = 3.0$, $k_1 = k_6 = k_9 = 5.0$, and all K_{mi} and k_{mi} equal to 0.2.

The initial concentration of inactive chemical species was set to one and zero for active species. The execution time was 217 seconds for 1600 time steps on a Mac Pro desktop computer with dual-core 2.66 GHz Intel Xeon processors.

In this model, a gradient is formed by protein activation on the cell edge, and propagated to the downstream signaling components Rac and Rho. Figure 13 shows that filopodia and thin protrusions have higher activation levels due the increased ratio of cell membrane to cell volume in these regions [16].

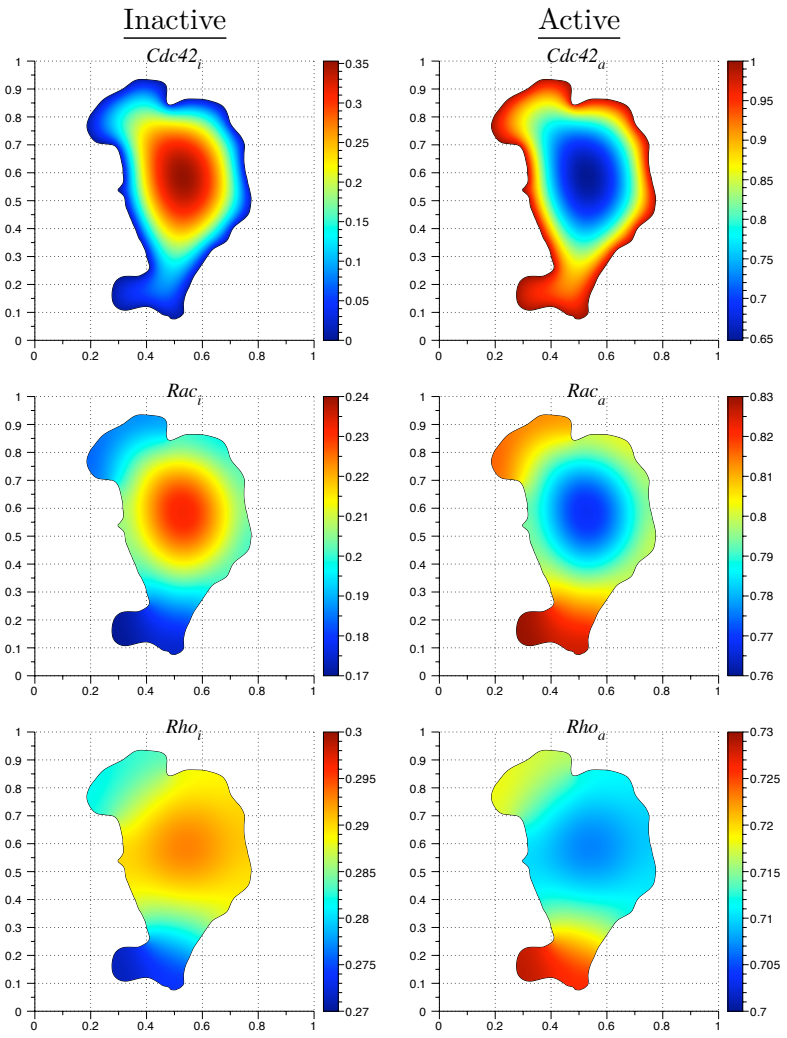


Figure 13. Rho GTPase model: steady-state distribution of protein concentration amounts in a fibroblast. The boundary was taken from a live cell image [19]. Values chosen for the constants: $S = k_3 = k_5 = k_7 = 1.0$, $k_2 = k_4 = k_8 = 3.0$, $k_1 = k_6 = k_9 = 5.0$, and all K_{mi} and k_{mi} equal to 0.2.

5. Conclusions

We have developed an accurate and efficient cut-cell method for simulating spatial models of signaling pathways in realistic cellular geometries. We demonstrated our method using models that consist of multiple species interacting in multiple

compartments. The examples were chosen to illustrate the numerical methods and therefore lack many details found in real biological signaling systems. In particular, feedback and feed forward control mechanisms that regulate pathway activity were not considered in detail. Our numerical methods provide important tools for investigating such regulatory mechanisms in realistic cell geometries and, therefore, should provide important insights into the ways signaling networks process and transmit information.

Our algorithm extends previous work on embedded boundary methods [5; 9; 15; 25]. These methods have been implemented in two and three dimension for Poisson's equation, the heat equation, and hyperbolic conservation laws. Our formulation extends these methods to systems of reaction-diffusion equations with nonlinear reactions in the interior as well as nonlinear reactions affecting boundary values. The boundary conditions treated in previous work [9; 15; 25] have been homogeneous Dirichlet and Neumann, which is not sufficient for many models of signaling pathways [16]. In [15], a second-order implicit method was used to update the heat equation in time [29]. In our method, we use an implicit nonlinear solver to handle nonlinear reactions occurring in the interior. An advantage of the differential-algebraic formulation is the ability to treat the boundary conditions as algebraic constraints. This allows us to handle reactions that take place on the physical boundary of the reaction-diffusion equation.

One limitation of the finite volume discretization arises from the interpolation method to obtain the normal derivative to the surface as shown in Figure 4. Cut cells must not have a zero volume cell within two rows or columns. For biological cells with long, thin or irregularly-shaped components such as neurons, mesh adaptive refinement may be needed to resolve the cellular geometry.

The underlying Cartesian-grid based finite volume discretization allows us to use advection schemes originally developed for hyperbolic conservation laws to simulate active transport or motility. In future reports, we will show how level set methods [18; 26] can be combined with biochemical reaction networks to investigate the effect of moving boundaries on cell signaling. Future work also includes a three-dimensional implementation of our fixed boundary algorithm. A three-dimensional extension of our method could be coupled with the method for simulating diffusion on a surface presented in [24] to obtain an algorithm for simulating models that take into account processes that occur both in the cytoplasm and on the cell membrane.

6. Acknowledgments

We thank Meng Jin and Yi Wu for insightful discussions. This work was supported by NIH grants R01-GM079271 and R01-GM078994.

References

[1] K. E. Brenan, S. L. Campbell, and L. R. Petzold, *Numerical solution of initial-value problems in differential-algebraic equations*, Classics in Applied Mathematics, no. 14, SIAM, Philadelphia, 1996. MR 96h:65083 Zbl 0844.65058

[2] P. N. Brown, A. C. Hindmarsh, and L. R. Petzold, *Using Krylov methods in the solution of large-scale differential-algebraic systems*, SIAM J. Sci. Comput. **15** (1994), no. 6, 1467–1488. MR 95g:65092 Zbl 0812.65060

[3] K. Burridge and K. Wennerberg, *Rac and Rho take center stage*, Cell **116** (2004), 167–179.

[4] P. Colella, D. Graves, T. Ligocki, D. Trebotich, and B. V. Straalen, *Embedded boundary algorithms and software for partial differential equations*, J. Phys. Conf. Ser. **125** (2008), 012084.

[5] P. Colella, D. T. Graves, B. J. Keen, and D. Modiano, *A Cartesian grid embedded boundary method for hyperbolic conservation laws*, J. Comput. Phys. **211** (2006), no. 1, 347–366. MR 2006i:65142 Zbl 1120.65324

[6] A. T. Dawes and L. Edelstein-Keshet, *Phosphoinositides and Rho proteins spatially regulate actin polymerization to initiate and maintain directed movement in a one-dimensional model of a motile cell*, Biophys. J. **92** (2007), no. 3, 744–768.

[7] H. G. Dohlman, *G Proteins and pheromone signaling*, Annu. Rev. Physiol. **64** (2002), 129–152.

[8] N. Hao, S. Nayak, M. Behar, R. H. Shanks, M. J. Nagiec, B. Errede, J. Hasty, and T. C. Elston, *Regulation of cell signaling dynamics by the protein kinase-scaffold ste5*, Mol. Cell. **30** (2008), no. 5, 649–656.

[9] H. Johansen and P. Colella, *A Cartesian grid embedded boundary method for Poisson's equation on irregular domains*, J. Comput. Phys. **147** (1998), no. 1, 60–85. MR 99m:65231 Zbl 0923.65079

[10] B. N. Kholodenko, *Cell-signalling dynamics in time and space*, Nat. Rev. Mol. Cell. Biol. **7** (2006), no. 3, 165–176.

[11] B. N. Kholodenko, G. C. Brown, and J. B. Hoek, *Diffusion control of protein phosphorylation in signal transduction pathways*, Biochem. J. **350** (2000), no. Pt. 3, 901–907.

[12] H. Levine, D. A. Kessler, and W. J. Rappel, *Directional sensing in eukaryotic chemotaxis: A balanced inactivation model*, Proc. Natl. Acad. Sci. USA. **103** (2006), no. 26, 9761–9766.

[13] T. J. Ligocki, P. O. Schwartz, J. Percelay, and P. Colella, *Embedded boundary grid generation using the divergence theorem, implicit functions, and constructive solid geometry*, J. Phys. Conf. Ser. **125** (2008), 012080 (5pp).

[14] R. Malladi, J. A. Sethian, and B. C. Vemuri, *A fast level set based algorithm for topology-independent shape modeling*, J. Math. Imaging Vision **6** (1996), 269–289. MR 97a:68174

[15] P. McCorquodale, P. Colella, and H. Johansen, *A Cartesian grid embedded boundary method for the heat equation on irregular domains*, J. Comput. Phys. **173** (2001), no. 2, 620–635. MR 2002h:80009 Zbl 0991.65099

[16] J. Meyers, J. Craig, and D. J. Odde, *Potential for control of signaling pathways via cell size and shape*, Curr. Biol. **16** (2006), no. 17, 1685–1693.

[17] S. R. Neves, P. Tsokas, A. Sarkar, E. A. Grace, P. Rangamani, S. M. Taubenfeld, C. M. Alberini, J. C. Schaff, R. D. Slizter, I. I. Moraru, and R. Iyengar, *Cell shape and negative links in regulatory motifs together control spatial information flow in signaling networks*, Cell **133** (2008), no. 4, 666–680.

[18] S. Osher and J. A. Sethian, *Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations*, J. Comput. Phys. **79** (1988), no. 1, 12–49. MR 89h:80012 Zbl 0659.65132

[19] O. Pertz, L. Hodgson, R. Klemke, and K. Hahn, *Spatiotemporal dynamics of RhoA activity in migrating cells*, Nature **440** (2006), no. 7087, 1069–1072.

[20] Y. Saad, *SPARSKIT: A basic tool-kit for sparse matrix computations* (version 2), 2005, user manual.

[21] Y. Saad and M. H. Schultz, *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput. **7** (1986), 856–869. MR 87g:65064 Zbl 0599.65018

[22] J. C. Schaff, B. M. Slepchenko, Y. Choi, J. Wagner, D. Resasco, and L. Loew, *Analysis of nonlinear dynamics on arbitrary geometries with the Virtual Cell*, Chaos **11** (2001), no. 1, 115–131. Zbl 0992.92021

[23] I. C. Schneider, E. M. Parrish, and J. M. Haugh, *Spatial analysis of 3' phosphoinositide signaling in living fibroblasts, III: influence of cell morphology and morphological polarity*, Biophys. J. **89** (2005), no. 2, 1420–1430.

[24] P. Schwartz, D. Adalsteinsson, P. Colella, A. Arkin, and M. Onsum, *Numerical computation of diffusion on a surface*, P. Natl. Acad. Sci. USA **102** (2005), no. 32, 11151–11156.

[25] P. Schwartz, M. Barad, P. Colella, and T. Ligocki, *A Cartesian grid embedded boundary method for the heat equation and Poisson's equation in three dimensions*, J. Comput. Phys. **211** (2006), no. 2, 531–550. MR 2006e:65194 Zbl 1086.65532

[26] J. A. Sethian, *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, 2nd ed., Cambridge Univ. Press, Cambridge, 1999. MR 2000c:65015 Zbl 0973.76003

[27] J. R. Shewchuk, *Engineering a 2D quality mesh generator and Delaunay triangulator*, Applied computational geometry (M. C. Lin and D. Manocha, eds.), Lecture Notes in Computer Science, no. 1148, Springer, Berlin, 1996, pp. 203–222. MR 97k:68004

[28] Y. Shimada, M. P. Gulli, and M. Peter, *Nuclear sequestration of the exchange factor Cdc24 by Far1 regulates cell polarity during yeast mating*, Nat. Cell. Biol. **2** (2000), no. 2, 117–124.

[29] E. H. Twizell, A. B. Gumel, and M. A. Arigu, *Second-order, L_0-stable methods for the heat equation with time-dependent boundary conditions*, Adv. Comput. Math. **6** (1996), no. 3-4, 333–352 (1997). MR 97m:65164 Zbl 0872.65084

[30] J. Valdez-Taubas and H. R. B. Pelham, *Slow diffusion of proteins in the yeast plasma membrane allows polarity to be maintained by endocytic cycling*, Curr. Biol. **13** (2003), no. 18, 1636–1640.

[31] O. C. Zienkiewicz and R. L. Taylor, *The finite element method set*, 6th ed., Butterworth–Heinemann, 2005.

Received June 24, 2009. Revised December 4, 2009.

WANDA STRYCHALSKI: wandastr@email.unc.edu
Carolina Center for Interdisciplinary Applied Mathematics, Department of Mathematics, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599, United States
http://www.unc.edu/∼wandastr

DAVID ADALSTEINSSON: david@amath.unc.edu
Carolina Center for Interdisciplinary Applied Mathematics, Department of Mathematics, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599, United States
http://amath.unc.edu/David/David

TIMOTHY ELSTON: telston@med.unc.edu
Department of Pharmacology, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599, United States
http://www.amath.unc.edu/Faculty/telston/