Published in final edited form as:

J Comput Phys. 2014 August 1; 270: 203–213. doi:10.1016/j.jcp.2014.03.014.

A conservative algorithm for parabolic problems in domains with moving boundaries

Igor L. Novak and Boris M. Slepchenko

Richard D. Berlin Center for Cell Analysis and Modeling, Department of Cell Biology, University of Connecticut Health Center, Farmington, Connecticut 06030

Abstract

We describe a novel conservative algorithm for parabolic problems in domains with moving boundaries developed for modeling in cell biology. The spatial discretization is accomplished by applying Voronoi decomposition to a fixed rectangular grid. In the vicinity of the boundary, the procedure generates irregular Voronoi cells that conform to the domain shape and merge seamlessly with regular control volumes in the domain interior. Consequently, our algorithm is free of the CFL stability issue due to moving interfaces and does not involve cell-merging or mass redistribution. Local mass conservation is ensured by finite-volume discretization and natural-neighbor interpolation. Numerical experiments with two-dimensional geometries demonstrate exact mass conservation and indicate an order of convergence in space between one and two. The use of standard meshing techniques makes extension of the method to three dimensions conceptually straightforward.

Keywords

numerical algorithm; moving boundaries; parabolic equations; exact mass conservation; cell migration

1. Introduction

In this paper, we describe a fully conservative algorithm for solving parabolic problems in domains with moving boundaries. The work was motivated by the need to develop a capability of modeling cell migration within the Virtual Cell (VCell), a computational framework for simulating cellular phenomena in realistic geometries [1–5].

Mechanisms underlying dynamical changes of cell shape and cell migration have long been of interest to cell biologists and biophysicists [6,7]. Mathematical models of varying degree of complexity have been proposed [8–17], and a recently released computer program called

corresponding author. Boris M. Slepchenko Richard D. Berlin Center for Cell Analysis and Modeling, University of Connecticut Health Center 263 Farmington Avenue, Farmington, Connecticut 06030-1507 boris@neuron.uchc.edu Phone: (860) 679-4025 Fax: (860) 679-1039.

Publisher's Disclaimer: This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final citable form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

‘Cytopede’ was specifically designed for modeling mechanical and signaling behaviors of plated cells [18]. Implementation of numerical algorithms, required for the modeling of dynamic cell shape and cell migration, within a general-purpose computational framework will further facilitate research in this area.

Numerical methods for solving systems with time-dependent discontinuities constitute an area of active research in applied mathematics and computational physics. They involve particular techniques for tracking boundaries and are often classified as Lagrangian and Eulerian, depending on whether the spatial grids are moving or stationary [19]. The interface-tracking techniques used in Lagrangian methods treat dynamic discontinuities as explicit sharp interfaces. The moving grids, constructed so that they fit the boundary, are recomputed each time the surface is updated [20]. This type of approach is used in the Cytopede code [18] that simulates shape changes of fibroblasts cultured on flat surfaces. In Eulerian methods, computations are performed on fixed grids, often based on structured meshes. These methods usually employ interface-capturing techniques in which the shape of the surface is defined implicitly. One class of interface-capturing techniques is level-set methods [21], in which the surface is determined by a zero level of an auxiliary volumetric scalar field.

There are also hybrid methods, in which an Eulerian approach to solving a governing equation is combined with a particular method of tracking or capturing the moving interface [22–26]. These methods are attractive as they avoid complexities of maintaining moving grids, which is particularly challenging in three dimensions, and have the advantages of using regular volumetric grids. Combining the Eulerian approach with interface-tracking techniques allows one to define explicit grids on the surface that are beneficial for cell-biological applications which often include membrane-bound molecules. However, exact mass conservation, another extremely valuable feature for applications in cell biology, becomes an issue in the hybrid methods, because the conservative discretization of equations subjected to boundary conditions at interfaces moving with respect to a fixed grid is nontrivial [23].

The conservative formulations of hyperbolic problems in domains with moving boundaries [23, 25] interpret the intersection of the Cartesian cells with the regions on either side of a moving front as a collection of time-dependent control volumes, over which the conservation law is integrated in space-time to derive a finite-volume discretization. The presence of arbitrary small control volumes in this approach leads to a Courant-Friedrich-Lewy (CFL) stability issue, which is resolved differently by different methods.

In an algorithm proposed by Glimm and co-workers [23], the CFL singularities are resolved through merging the small control volumes with adjacent cells [27]. The method works in conjunction with robust front-tracking methods implemented in FronTier [28], a publicly available C++ library for accurate tracking of moving interfaces both in two (2D) and three dimensions (3D). A similar approach was used by Udaykumar and coworkers [29, 30] in applications to solidification and fluid dynamics problems. Cell-merging, however, while well-defined in one and two dimensions, may not be straightforward to apply to 3D problems.

In the embedded boundary methods [24] based on the Volume-of-Fluid front-capturing techniques [31], cell-merging is avoided by performing two updates, a conservative update and a nonconservative, but CFL stable, update. The difference between the two updates is then redistributed to neighboring computational cells to maintain overall stability and local conservation. Recent improvements to the method have been shown to yield second-order accuracy in one dimension [25]. While the redistribution of mass is simpler to apply in multiple dimensions, maintaining the second-order accuracy of the method in 2D and 3D is a challenging problem.

In a recently published hybrid algorithm [26], specifically designed for applications to cell biology, the contour of a moving cell is captured by a level-set technique. Still, the algorithm constructs and maintains the grid on the contour by projecting nearby rectangular grid points onto the interface at each integration time step. The method employs finite-volume discretization with the grid becoming unstructured near the membrane. Because of non-conservative interpolation of displaced grid points, the method does not conserve mass locally, but global mass conservation is enforced on the assumption that the total mass inside the domain is known in advance.

This paper is focused on the description of a fully conservative discretization scheme for parabolic (diffusion-advection-reaction) equations with Rankine-Hugoniot boundary conditions imposed at moving interfaces. For simplicity, we defer the discussion of the front-tracking technique and assume that the position of a moving piecewise-linear interface is known at any time.

In our algorithm, spatial discretization is based on Voronoi decomposition [32] applied to a fixed set of regularly spaced gridpoints. Voronoi meshes were used previously in modeling cell migration by Bottino et al. [33] and in our algorithm for modeling diffusion on a surface coupled to diffusion in embedding volume [3]. Utilizing this meshing technique for purposes of the algorithm described in this article has a number of practical advantages. Voronoi polygons are well suited for conservative finite volume discretization. The procedure automatically generates control volumes of comparable sizes, and the CFL stability issue due to crossings of a moving front with the mesh does not arise. Thus, there is no need in our method to adjust positions of the grid points, nor it requires cell-merging or redistributing mass between control volumes. While the control volumes adjacent to a moving interface conform to the shape of the domain and are therefore irregular, they merge seamlessly with regular Voronoi cells in the interior of the domain. In fact, the procedure of Voronoi decomposition needs to be applied only to the gridpoints in the vicinity of the interface. Finally, the use of the standard meshing technique makes extension of the method to three dimensions conceptually straightforward.

Fully conservative formulation is achieved through finite-volume discretization and natural-neighbor interpolation [34] (interpolation is required to reinitialize unknowns at the nodes crossed by a propagating front and at their neighbors). The formulae of the natural-neighbor interpolation are derived from the requirement that mass be not relocated when boundaries between Voronoi cells are redrawn, thus preserving local mass conservation. Test cases with

two-dimensional geometries demonstrate exact mass conservation and indicate an order of convergence of the error between 1 and 2.

2. Mathematical problem

Consider a system defined in a spatial domain $\Omega(t) \in \mathbb{R}^n$ (n is the dimensionality of the space) whose boundary $\Omega(t)$ changes with time. The system is governed by the continuity (mass conservation) equation,

$$\frac{\partial u}{\partial t} = -\operatorname{div} \mathbf{J} + R, \quad (1)$$

where u is the concentration of a molecular species, $\mathbf{J} = -D\nabla u + \mathbf{v}u$ is the diffusive-advective flux density, and R is the reaction term. Eq (1) is subject to the Rankine-Hugoniot boundary condition at the moving boundary $\Omega(t)$, which in the case of the interface representing an impermeable membrane is written as

$$(\mathbf{J} - \mathbf{v}_b u) \mathbf{n} |_{\partial\Omega} = 0, \quad (2)$$

where $\mathbf{n}$ is the outward normal and $\mathbf{v}_b$ stands for the velocities of the boundary points. For simplicity, the diffusion coefficient, D , and the velocity field in the volume, $\mathbf{v}$, are assumed to be time-independent. We also assume that the membrane velocities, $\mathbf{v}_b$, are given at any time, so that the position of $\Omega(t)$ is either known exactly or can be computed with sufficiently high accuracy for any t .

3. Algorithm

Consider a box that would contain $\Omega(t)$ for any $t \in [0, T_{\max}]$. The boundary of the domain, $\Omega(t)$, is assumed to be piecewise-linear. A cell-centered, uniform orthogonal mesh of size h is applied to the box to generate a regular set of cell centers $\{\mathbf{r}_i\}$. Our algorithm approximates the solution of Eqs (1),(2) at cell centers inside $\Omega(t)$. Because $\Omega(t)$ changes with time, these cell centers comprise a time-dependent subset of grid points $S(t) = \{\mathbf{r}_i(t)\}$. Indeed, the time dependence of this subset is piecewise constant, because only at discrete times t_k ($k = 1, 2, \dots$) do one or more cell centers abruptly become interior or exterior points as the boundary crosses them, and in between those times, the set $S(t)$ remains unchanged. The set of the interior nodes in the time interval (t_{k-1}, t_k) will be denoted by S_k .

To ensure mass conservation, we apply the finite volume spatial discretization of Eqs (1), (2) [19]. In this discretization scheme, $\Omega(t)$ is decomposed into small control volumes $\{\omega_i(t)\}$, each including a single computational node $\mathbf{r}_i$. Eq (1) is then integrated over $\omega_i(t)$. Taking into account the Rankine-Hugoniot boundary conditions, Eq (2), imposed at the moving boundary, this integration yields

$$\frac{\partial}{\partial t} \int_{\omega_i} u dv = \int_{\partial\omega_i} \left(D \frac{\partial u}{\partial n} - \mathbf{v} \mathbf{n} u \right) ds + \int_{\omega_i} R dv, \quad (3)$$

where $\mathbf{n}$ is the outer normal to ω_i . In our method, the control volumes $\{\omega_i\}$ are obtained by means of Voronoi decomposition [32], a 2D example of which is shown in Figure 1. Note

that at the boundary, the Voronoi decomposition automatically generates a layer of irregular control volumes that conform to the geometry of the front and at the same time mesh seamlessly with neighboring regular cells. Later in this section, we describe two methods of constructing irregular control volumes, with exact and approximate matching of $\Omega(t)$.

The discretized form of Eq (3) yielded by this meshing scheme is

$$\frac{d(u_i(t)V_i(t))}{dt} \approx \sum_{j \in G(i)} \left(D \frac{u_j(t) - u_i(t)}{d_{ij}} - \mathbf{n}_{ij} \cdot \frac{\mathbf{v}_j + \mathbf{v}_i}{2} \cdot \frac{u_j + u_i}{2} \right) \cdot s_{ij}(t), i \in S(t), \quad (4)$$

where u_i and u_j are the concentration values at the corresponding grid points, $V_i = |\omega_i|$ is the volume (area in 2D) of the i th Voronoi cell, d_{ij} is the distance between the i th and j th grid points, s_{ij} is the area (length) of the shared face (edge) separating the cells i and j ,

$\mathbf{n}_{ij} = \frac{\mathbf{r}_j - \mathbf{r}_i}{|\mathbf{r}_j - \mathbf{r}_i|}$ is the outer normal to s_{ij} , and $G(i)$ is the set of nodes in $S(t)$ that correspond to the Voronoi cells adjacent to the i th cell. The reaction term, whose spatial discretization is trivial, is omitted in Eq (4) for simplicity. While the discretization of the flux terms in the right-hand side of Eq (4) is second-order accurate at the interior grid points representing regular Voronoi cells, the accuracy is lower for irregular control volumes, because the fluxes in this case are approximated by our algorithm at the points that are different from the face centroids. As a result, the overall order of accuracy in space is between 1 and 2 (see the results of convergence studies and the discussion of the accuracy of the method in Section 4).

We now describe two methods of constructing irregular control volumes adjacent to the propagating front. By definition, a Voronoi polygon is a locus of points that are closer to a given node than to any other node. Based on this definition, our method of generating irregular control volumes that exactly conform to the domain shape includes two steps. First, Voronoi polygons are constructed for all grid points in S_k , assuming (theoretically) that the space they are in is infinite. Second, the intersection of the polygons obtained in step 1 and $\Omega(t)$ is computed. While theoretically some of the polygons obtained in step 1 will not be closed, in practice they are closed by boundaries of the box containing $\Omega(t)$ (see the beginning of this section). This is implemented by adding ‘ghost’ points to S_k , which are created in this method by reflecting in the box boundaries the nodes in S_k close to $\Omega(t)$. The intersection of polygons in step 2 is performed using the Vatti clipping algorithm [35].

The second method, illustrated in Figure 1, right, approximates a domain boundary with the Voronoi convex polygons (polyhedrons). It also employs ‘ghost’ points for closing irregular Voronoi polygons, but in this case, the “ghost” points (shown in red) are obtained by reflecting the gridpoints in $\Omega(t)$. This method is computationally less expensive but, unlike the first one, introduces additional errors due to local curvature, which are $O(h/\rho)$, where h is the mesh size and ρ is the radius of curvature.

Eqs (4) are integrated over time using first-order fixed time step methods. For regular ω_i , which are not adjacent to $\Omega(t)$ and therefore remain regular throughout the time step, V_i and s_{ij} are independent of time, and standard time discretization schemes [19], explicit or

implicit, can be applied to the corresponding grid points. For the Voronoi cells adjacent to $\Omega(t)$, the integration within the periods (t_{k-1}, t_k) , for which $S(t) \equiv S_k$ remains unchanged, is not much different: the only distinction is that V_i and s_{ij} of the irregular Voronoi cells are time-dependent and need to be updated for each integration step t . In particular, the time discretization of the left-hand side of Eqs (4) for the corresponding grid points is $t^{-1} (u_i(t + \Delta t) V_i(t + \Delta t) - u_i(t) V_i(t))$.

Our algorithm assumes that the irregular cells do not disappear within one time step due to the repositioning of the front. This imposes a constraint on the integration time step,

$\Delta t < \frac{h}{2 \max_{\partial\Omega} |\mathbf{v}_b \cdot \mathbf{n}|}$, mathematically resembling the regular CFL condition. Note that unlike the CFL stability issue that usually arises in hybrid methods, the upper bound for the time step in our method is finite. This is because the sizes of the Voronoi cells centered around uniformly spaced grid points are comparable by construction.

Special care is required for integrating across $t = t_k$, when the set of unknowns $\{u_i\}$ changes because of the changing position of $\Omega(t)$. At those times, Eqs (4) must be properly re-initialized for the nodes that become interior and the nodes whose natural neighbors change. The structure and parameters of Eqs (4) should be adjusted correspondingly, and the new initial values of the unknowns at the affected nodes must be computed. The reinitialization is an interpolation procedure, in which the values of $u(\mathbf{x})$ at the affected nodes are approximated on the basis of known values of u at surrounding gridpoints. In our algorithm, this is done by means of the natural-neighbor interpolation [34] that ensures local mass conservation. In what follows, we use the superscripts “−” and “+” to denote the values of variables at $t = t_k - 0$ and $t = t_k + 0$, respectively, so that $S_k \equiv S^-$, $S_{k+1} \equiv S^+$, $u(\mathbf{r}, t_k - 0) \equiv u^-(\mathbf{r})$, $u(\mathbf{r}, t_k + 0) \equiv u^+(\mathbf{r})$, etc.

The initial conditions for the new set of variables are derived from the requirement that mass be conserved locally while the borders of the Voronoi cells are redrawn [34]. Thus, the mass within $\omega_i^+ (i \in S^+)$, $m^+(\omega_i^+) = u_i^+ V_i^+$, must equal the mass that existed in the same volume before the new borders between the Voronoi cells were drawn. The latter is

$\sum_{j \in S^-} u_j^- |\omega_i^+ \cap \omega_j^-|$ because ω_i^+ is composed of the parts of the control volumes $\omega_j^- (j \in S^-)$ which existed before the “redistricting”, i.e. $\omega_i^+ = \bigcup_{j \in S^-} (\omega_i^+ \cap \omega_j^-)$. Therefore,

$$m^+(\omega_i^+) = \sum_{j \in S^-} u_j^- |\omega_i^+ \cap \omega_j^-|. \quad (5)$$

The initial values for the new set of variables, $u_i^+ = \frac{m^+(\omega_i^+)}{V_i^+}$, then follow from Eqs (5):

$$u_i^+ = \frac{1}{V_i^+} \sum_{j \in S^-} u_j^- |\omega_i^+ \cap \omega_j^-|. \quad (6)$$

Because equations (5) and (6) are based on local mass conservation, the total mass in Ω is conserved as well. This can be verified by summing Eqs (5) over all $i \in S^+$ and taking into

account that $V_j^- = \sum_{i \in S^+} |\omega_i^+ \cap \omega_j^-|$, which yields $\sum_{i \in S^+} u_i^+ V_i^+ = \sum_{j \in S^-} u_j^- V_j^-$.

Note that the reinitialization of the concentration is not required for the nodes whose natural neighbors remain unchanged during the transition $S^- \rightarrow S^+$. Indeed, in this case,

$$\sum_{j \in S^-} u_j^- |\omega_i^+ \cap \omega_j^-| = \sum_{j \in S^+} u_j^- |\omega_i^+| \delta_{ij} = V_i^+ \sum_{j \in S^+} u_j^- \delta_{ij} \quad (\delta_{ij} \text{ is the Kronecker delta}) \text{ and from Eqs (6), } u_i^+ = u_i^-.$$

We now use Eqs (6) to derive re-initialization formulae for situations where a node is added to or lost from S^- as a result of the transition $S^- \rightarrow S^+$. If a new node i is added to S^- at $t = t_k$ (Figure 2), then, using Eqs (6), the corresponding concentration u_i^+ is

$$u_i^+ = \frac{1}{V_i^+} \sum_{j \in G(i)} u_j^- |\omega_i^+ \cap \omega_j^-| = \frac{1}{V_i^+} \sum_{j \in G(i)} u_j^- (V_j^- - V_j^+). \quad (7)$$

Note that $(V_j^- - V_j^+)$ in Eqs (7) are guaranteed to be positive because $\omega_j^+ \subset \omega_j^-$ ($j \in G(i)$).

For the neighboring nodes $j \in G(i)$, Eqs (6) yield $u_j^+ = u_j^-$, again due to $\omega_j^+ \subset \omega_j^-$.

For the situation where a node i is lost from S_- at $t = t_k$ (Figure 3), Eqs (6) yield the following new initial values u_j^+ at the neighboring nodes $j \in G(i)$:

$$u_j^+ = \frac{1}{V_j^+} (u_i^- |\omega_i^- \cap \omega_j^+| + u_j^- |\omega_j^- \cap \omega_j^+|) = \frac{1}{V_j^+} (u_i^- V_i^- \phi_j + u_j^- V_j^-), \quad (8)$$

where $\phi_j = (V_j^+ - V_j^-) [\sum_{j \in G(i)} (V_j^+ - V_j^-)]^{-1}$ is the volume fraction of the part of ω_i^-

included into ω_j^+ (Figure 3, left panel). Note that $\sum_{j \in G(i)} \phi_j = 1$ and $\phi_j > 0$ for all j because in this case, $\omega_j^+ \supset \omega_j^-$.

In Figures 2 and 3, the added (lost) node is located exactly at the interface, consistently with the definition of t_k . It is easy to see, however, that Eqs (7) and (8) and their derivation would not change if the update of the Voronoi decomposition for the nodes adjacent to $\Omega(t)$ were done at a slightly different time with the added (lost) node being a little behind (or ahead of) the front. This means that one does not have to look for the exact values of t_k but rather can check for changes in the subset of the adjacent nodes at each time step.

Note that Eqs (7) and (8) were derived for cases where nodes crossed by a moving boundary within one time step are far from each other (or they are crossed one at a time). If the time step is insufficiently small, several nodes that are close to one another might be crossed within the same time step, a situation that is not covered by Eqs (7) and (8). There are

various ways to deal with this type of events. In a simple approach employed in our algorithm, all nodes that change their status from interior to exterior or vice versa within a given time step are tagged for reinitialization, and Eq (7) or (8) is applied to them one at a time in the order they were listed in the array $\{\mathbf{r}_i\}$. Importantly, the set of interior nodes S_k must also be updated consecutively each time Eq (7) or (8) has been used. The method may introduce slight asymmetry due to the ordering of gridpoints. But because the situations in which neighboring nodes change their status simultaneously become rarer as the time step decreases, the possible asymmetry is not a factor for the overall convergence, as demonstrated by the convergence studies described in the next section. An alternative approach would be to apply the natural-neighbor interpolation simultaneously to all gridpoints changing their status [35]. For this, one should compute an intersection of the new decomposition and the old mesh to determine all non-overlapping clips that change their association with the nodes, and Eqs (7) and (8) should be appropriately generalized.

In summary, our algorithm includes the following steps:

Initializing the problem:

- (i) given $\Omega(0)$, identify the set of interior nodes $S(0)$ and the subset of nodes adjacent to $\Omega(0)$;
- (ii) perform Voronoi decomposition for the initial subset of nodes adjacent to $\Omega(0)$ (for other grid points in $S(0)$, the control volumes are regular orthogonal boxes);
- (iii) compute $d_{ij}(0)$ (distances between Voronoi neighbors), $s_{ij}(0)$ (lengths of the edges (areas of the faces) of the irregular Voronoi cells), and $V_i(0)$ (areas (volumes) of the irregular Voronoi cells). Parameters of the regular cells are precomputed, given the mesh size h .

Advancing the solution in time:

- (iv) find $\Omega(t + \Delta t)$;
- (v) update the Voronoi decomposition for the nodes adjacent to $\Omega(t + \Delta t)$;
- (vi) compute $s_{ij}(t + \Delta t)$ and $V_i(t + \Delta t)$ for the irregular Voronoi cells (as in step (iii));
- (vii) solve Eqs (4) to find $u_i(t + \Delta t)$; (Note: steps (iv)-(vii) are performed for $S(t)$; if $S(t + \Delta t)$ is different from $S(t)$, the results of step (vii) are used in step (ix).)
- (viii) identify the set of interior nodes $S(t + \Delta t)$ by updating the subset of the nodes adjacent to $\Omega(t + \Delta t)$;
- (ix) if during the time step a computational node changes its status from exterior to interior (from interior to exterior), adjust Eqs (4) and use Eqs (7) (Eqs (8)) to update $u_i(t + \Delta t)$ for this node and its natural neighbors. This involves computing the corresponding new distances $d_{ij}(t + \Delta t)$ and areas

(lengths) $s_{ij}(t + \Delta t)$. If there are several such nodes in one time step, treat them one by one in an arbitrary order.

4. Convergence studies and tests of mass conservation

Accuracy of the proposed algorithm is affected by several factors. As mentioned in the previous section, the finite volume discretization is second-order accurate only for regular control volumes in the bulk of the domain. For irregular Voronoi cells adjacent to the boundary, approximation of fluxes at the points that are not centroids of the faces lowers the order of convergence in space by approximately $1/2$ (see results below). In addition, the reinitialization of unknowns at grid points in the vicinity of a moving front is a first-order procedure, which in multiple dimensions¹ results in L_∞ -norm error close to $O(h)$. However, the error due to reinitialization occurs only at few nodes at a time and does not affect significantly other nodes. This is because it dissipates quickly to its spatial average, which is zero due to exact local mass conservation built in the algorithm. Indeed, the solution error averaged over the layer of irregular points remains to be $O(h^{1.4})$ or higher, whereas convergence in the L_2 -norm is closer to the second order. Overall, the order of convergence of our algorithm in space is between 1 and 2.

The convergence of the algorithm was studied using a number of test problems whose numerical solutions obtained with the algorithm were tested either against exact analytical solutions or precise numerical solutions computed by alternative methods. Total mass inside moving domains has been monitored in all tests and indicates exact mass conservation.

We evaluated several versions of the algorithm that differed by their time discretization scheme (explicit or implicit) and/or by a method of constructing irregular cells near the boundary: the method in which $\Omega(t)$ is approximated by irregular Voronoi polygons (illustrated in Figure 1) or the method where irregular control volumes are obtained by exact intersection of $\Omega(t)$ and (theoretically) open Voronoi polygons. While various versions of the algorithm have similar orders of convergence, the magnitude of the error was significantly lower for the solutions obtained with irregular control volumes that conform to $\Omega(t)$ precisely. The results presented below were obtained using the explicit time discretization and precise construction of irregular control volumes.

4.1. Diffusion inside a moving box: steady-state solutions

A problem of diffusion in a steadily moving box can be solved exactly and therefore provides a convenient test for our algorithm. Consider a gas in an $a \times a$ box $\Omega'(t) = \{x'(t), y'(t) : |x' - v_b t| \leq a/2, |y'| \leq a/2\}$ that moves with a constant velocity, $\mathbf{v}_b = (v_b, 0)$. By changing to a frame of reference moving with the same velocity $\mathbf{v}_b$, the problem reduces to that in a fixed domain,

$$\partial_t u = \text{div}(D \nabla u + \mathbf{v}_b u), (D \nabla u + \mathbf{v}_b u) \mathbf{n} \Big|_{\partial \Omega} = 0, \quad (9)$$

where u is now defined in $\Omega = \{x, y : |x| \leq a/2, |y| \leq a/2\}$.

¹2D and 3D; in 1D, the reinitialization is a rare event with no significant effect on the L_∞ -norm error.

Finding the steady state of the system is particularly straightforward. The problem becomes one-dimensional, with the density $u(x)$ satisfying the following equations: $D \frac{\partial^2 u}{\partial x^2} + v_b \frac{\partial u}{\partial x} = 0$, $(D \frac{\partial u}{\partial x} + v_b u)|_{x=\pm a/2} = 0$. The exact solution of these equations is $u(x) = u_0 \exp(-kx)$, where $k = v_b / D$ and the pre-exponential factor $u_0 = u(0)$ that can be expressed in terms of a given average gas concentration, $\bar{u} = \iint_{\Omega} u dx dy$. For $\bar{u} = 1$, $u_0 = (ka/2)(\sinh(ka/2))^{-1}$. The corresponding traveling wave solution in the laboratory frame of reference, $u(x', t) = u_0 \exp(-k(x' - v_b t))$, was used for evaluating errors of the numerical solutions obtained with the algorithm.

In the test, numerical solutions of Eqs (1), (2) were obtained for $D = 1$, $\mathbf{v} = 0$, and $R = 0$ in a square box with $a = 2$ moving with the speed $v_b = 1$. Given the initial condition $u(x', 0) = u_0 \exp(-x')$, the solver quickly settles on a numerical steady state. The snapshot in Figure 4 (upper left) illustrates a numerical solution at $t = 0.5$ obtained with $h = 0.048$ and $\tau = 5.76 \times 10^{-4}$. The upper right panel of Figure 4 shows a horizontal line scan of the snapshot (dots) superimposed on the corresponding exact solution (solid curve).

The lower panel of Figure 4 illustrates orders of convergence of the L_{∞} - and L_2 - norms of the solution error, E_{∞} and E_2 . Because the problem is essentially one-dimensional, the low-order reinitialization procedure is performed rarely and therefore does not affect E_{∞} appreciably. Thus, the order of convergence of E_{∞} in this test largely reflects the effect of irregularity of the control volumes adjacent to the boundary.

Interestingly, the traveling wave solution, $u(x', t) = u_0 \exp(-k(x' - v_b t))$, holds true regardless of the domain shape, provided the shape remains fixed during the motion and the problem is solved for the zero-flux boundary conditions. We therefore conducted a similar test for a circular domain where the problem is fundamentally two-dimensional. The results indicate that in multiple dimensions, the order of convergence of the L_{∞} -norm error, E_{∞} , is indeed affected by the first-order reinitialization error (Figure 5, left). This error, however, is local in nature and does not affect the accuracy of the solution even within the boundary layer of irregular Voronoi cells. Indeed, when averaged over irregular control volumes, the error exhibits convergence of $O(h^{1.39})$, comparable to that of E_{∞} in 1D. The convergence of E_2 for the circular domain is similar to that for the square domain (Figure 5, right).

4.2. Diffusion inside a moving circle: time-varying solutions

Here we test the accuracy of time-varying solutions to the problem of diffusion in a circular domain moving with velocity $\mathbf{v}_b = (1, 0)$, $\Omega'(t) = \{x'(t), y'(t) \mid (x' - v_b t)^2 + y'^2 \leq 1\}$, by comparing them with precise numerical solutions obtained by different methods. To find baseline numerical solutions, the problem is reformulated, similarly to the previous case, as an equivalent diffusion-advection problem in a fixed domain $\Omega(t) = \{x, y \mid x^2 + y^2 \leq 1\}$ (see Eqs (9)). The accurate numerical solutions are then obtained by solving the equivalent problem with high precision using two “conventional” PDE solvers, a VCell diffusion-advection solver [1] and a COMSOL MULTIPHYSICS® PDE solver (<http://www.comsol.com/products/multiphysics/>).

Test results are presented in Figure 6. The snapshot in the upper left pictures a solution of the moving boundary problem at $t = 0.2$. The solution was obtained with the initial

concentrations $u(\mathbf{r},0) = 1$, using the following discretization parameters: $h = 0.044$, $t = 4.84 \times 10^{-4}$. In the upper right of Figure 6, the results scanned along the equator of the snapshot (dots) are superimposed on the baseline solution obtained with the VCell diffusion-advection solver (solid curve); the latter was computed with $h = 0.0015$ and the absolute and relative tolerances set at 10^{-9} and 10^{-7} , respectively, which yielded absolute error $\sim 10^{-5}$, estimated by Richardson extrapolation.

The solution of the test problem over the entire simulation time was recorded in Supplementary Movie 1, that also provides comparison with the VCell baseline solution over time. In addition, the movie includes a third panel that demonstrates exact mass conservation by displaying the total mass inside the moving domain as a function of time.

The lower panel of Figure 6 illustrates the convergence in space of the L_∞ - and L_2 - norms of the solution error at $t = 0.2$. The dependencies $E_\infty(h)$ and $E_2(h)$, obtained in this case against an accurate solution computed with COMSOL MULTIPHYSICS®, are similar to the ones observed in the previous test (Figure 5). The error averaged over irregular control volumes was $O(h^{1.45})$, also comparable to the result of the previous test.

In computing the baseline solution with COMSOL MULTIPHYSICS®, the absolute and relative tolerances were set at 10^{-9} and 10^{-8} , respectively, and the mesh was refined to yield at least 7 accurate digits after the decimal point. The error of the baseline solution was estimated by computing system's steady state, for which the exact solution is available (see section 4.1).

4.3. Diffusion-advection in a moving circle

In living cells, diffusion of macromolecules occurs in cytosol (intracellular fluid), and in a moving cell, the cytosol moves as well. The realistic models of migrating cells will therefore require solving the diffusion-advection equations in moving domains. Here we evaluate the accuracy of the algorithm when it is applied to diffusion-advection systems, using a simple test problem with an analytical solution.

Consider a situation where the fluid is stationary with respect to the domain. Then, in a frame of reference that moves with the domain, the problem reduces to solving a diffusion equation in a fixed domain with zero-flux boundary conditions, a problem that has an

analytical solution for simple domain shapes. Specifically, we solve Eq (1) with $R = 0$, $D = \frac{1}{4}$ and $\mathbf{v} = (1,0)$ in a unit circle $\Omega(t)$ moving with the velocity $\mathbf{v}_b = \mathbf{v}$. For certain initial conditions, the problem has closed-form solutions. In particular, one can verify that $u(x, y, t) = \exp(-\lambda^2 D t) x J_1(\lambda((x - v_b t)^2 + y^2)^{1/2}) + J_1(\lambda)$, where $J_1(z)$ is the first-order Bessel function of the first kind and $\lambda = 1.841183781340659$, is an exact solution of the original problem in $\Omega(t) = \{(x, y) : (x - v_b t)^2 + y^2 = 1\}$. The corresponding initial condition is $u(x, y, 0) = x J_1(\lambda(x^2 + y^2)^{1/2}) + J_1(\lambda)$, which was used in the numerical test.

The error analysis was performed for solutions at $t = 0.2$ with varying h . One such solution is shown in Figure 7, along with the plots of $E_\infty(h)$ (circles) and $E_2(h)$ (triangles). The enhanced convergence of E_∞ and an essentially second order of E_2 are likely because of the

smallness of gradients of the solution near the boundary, which is in part, due to the zero-flux boundary condition. In particular, the error due to reinitialization, that would normally dominate E_∞ , is significantly reduced in the absence of steep gradients at the boundary.

4.4. Mass conservation in domains with varying shape and velocities

Exact mass conservation was observed in all tests described in the previous sections. In addition, mass conservation was also tested using problems for which we did not have exact solutions, such as diffusion in domains with changing shapes and non-constant translational velocities. The solutions of these problems obtained with our algorithm are demonstrated in Supplementary Movies 2 and 3.

In Movie 2, a circular domain abruptly changes direction of its motion. A color code is used to illustrate changes in the density of a diffusing gas from low (blue) to high (red). A similar color code is used in Movie 3 to demonstrate a solution of a diffusion problem in a cell that changes its shape. The vectors shown in the movie depict velocities of the points of the cell boundary. The solutions demonstrate physically reasonable dynamics and conservation of mass within machine precision.

5. Concluding remarks

In this article, we described and analyzed a conservative algorithm for solving reaction-diffusion-advection systems in domains with moving boundaries. The algorithm was developed for applications in cell biology, where ensuring rigorous conservation of mass is of particular importance.

The proposed algorithm is a hybrid method where dynamic boundaries move relative to a fixed regular grid. The CFL stability issue that arises in hybrid methods due to intersections of a moving front with the underlying mesh is avoided by applying Voronoi decomposition. Control volumes produced by this technique in the vicinity of a front conform to the shape of the domain and merge seamlessly with regular control volumes in the interior of the domain. Local mass conservation is achieved through the built-in finite-volume discretization and by applying the mass-conserving natural-neighbor interpolation to nodes affected by crossings with a propagating front.

The algorithm was tested against exact analytical solutions as well as precise numerical solutions obtained by different algorithms. The results indicate the order of convergence in space between 1 and 2. The exact, within machine precision, conservation of mass was observed in all tests, including those for which we did not have exact solutions. The latter involved domains with changing shapes and domains moving with non-constant velocities.

Certain limitations must be relaxed before the described algorithm can be integrated into a general-purpose computational tool. The method is currently implemented and tested for 2D geometries. Because it employs standard meshing techniques, extension to 3D is conceptually straightforward. Still, for a 3D version of the algorithm to be practical, additional optimization of the method will likely be necessary.

The test problems of the previous section are solved in domains with explicitly defined geometries and given kinematics. In actual applications, the algorithm should be able to handle cases where the positions of moving boundaries are not known *a priori*, but rather determined based on given or computed velocity fields, and with geometries defined more generally, e.g. by implicit functions. It therefore must work in conjunction with robust front-tracking (capturing) techniques. We evaluated several existing methods, by testing them against “exact” kinematics, and found a recent version of FronTier [28] particularly appealing. It combines advantages of front-tracking and front-capturing techniques and maintains surface grids. The latter is of particular importance, given that applications to cell biology often include variables defined on moving interfaces.

Front-tracking with FronTier is illustrated by Supplementary Movies 4 and 5. The examples used in the movies mimic dynamic changes of cell membranes. In Movie 4, the solid curve represents a dynamic shape of a 2D “cell”, with the “exact” kinematics $\Omega(t) = \{x(t), y(t)\}$ described by $x = f(\phi, t) \cos \phi$, $y = f(\phi, t) \sin \phi$, where $f(\phi, t) = 1 + 0.1 \cos(5\phi + 5t) + 0.2 \cos(7\phi + 3.5t)$ and $\phi \in [0, 2\pi]$, $t \in [0, 25]$. The dots shown in the movie are the boundary points computed by FronTier on the basis of the input velocities $x'(t)$, $y'(t)$.

Similarly, the “exact” kinematics of a 3D “cell”, $\Omega(t) = \{x(t), y(t), z(t)\}$, described by $x(t) = f(\phi, \theta, t) \cos \phi \cos \theta$, $y(t) = f(\phi, \theta, t) \sin \phi \cos \theta$, $z(t) = f(\phi, \theta, t) \sin \theta$, where $f(\phi, \theta, t) = 1 + 0.9 \cos(10t + 4.8) Y_4^2(\phi, \theta) + 0.9 \cos(3t + 1.44) Y_7^3(\phi, \theta)$, $Y_4^2(\phi, \theta)$ and $Y_7^3(\phi, \theta)$ are the spherical harmonics, and $\phi \in [0, 2\pi]$, $\theta \in [0, \pi]$, $t \in [0, 8]$ (Movie 5a), were used to compute velocities $x'(t)$, $y'(t)$, $z'(t)$ which were an input for FronTier. Based on this input, FronTier computed the dynamics of boundary points shown in Movie 5b for a cross-section $z = 0$, along with the cross-section of the exact surface (solid curve).

Upon the implementation of the algorithm in VCell, the users will be able to solve problems where the velocities of the membrane points are either known or can be evaluated as functions of state variables. Because velocities are determined by forces exerted on, and generated by, the cell, the algorithm will ultimately be integrated with cell mechanics. The latter typically includes forces generated by molecular motors and by polymerization of the cytoskeleton, viscoelasticity of the cytoskeleton and its adhesion to extracellular matrix [36].

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

We thank Leslie Loew for continuing support and stimulating discussions and James Schaff and Diana Resasco for numerous helpful discussions and suggestions. Fei Gao performed tests with FronTier and helped coding the algorithm. Xiaolin Li provided guidance in using latest versions of FronTier. The work was supported by the National Institutes of Health, through grants from the National Center for Research Resources (2P41RR013186-15) and the National Institute of General Medical Sciences (9P41GM103313-15 and U54GM64346).

References

1. Resasco DC, Gao F, Morgan F, Novak IL, Schaff JC, Slepchenko BM. Virtual Cell: computational tools for modeling in cell biology. *WIREs Syst. Biol. Med.* 2012; 4:129–140.

2. Moraru II, Schaff JC, Slepchenko BM, Blinov M, Morgan F, Lakshminarayana A, Gao F, Li Y, Loew LM. Virtual Cell modelling and simulation environment. *IET Syst. Biol.* 2008; 2:352–362. [PubMed: 19045830]
3. Novak IL, Gao F, Choi YS, Resasco D, Schaff JC, Slepchenko BM. Diffusion on a Curved Surface Coupled to Diffusion in the Volume: Application to Cell Biology. *J Comput. Phys.* 2007; 226:1271–1290. [PubMed: 18836520]
4. Slepchenko BM, Schaff JC, Choi YS. Numerical Approach to Fast Reactions in Reaction-Diffusion Systems: Application to Buffered Calcium Waves in Bistable Models. *J. Comput. Phys.* 2000; 162:186–218.
5. Schaff J, Fink CC, Slepchenko B, Carson JH, Loew LM. A general computational framework for modeling cellular structure and function. *Biophys. J.* 1997; 73:1135–1146. [PubMed: 9284281]
6. Lauffenburger DA, Horwitz AF. Cell migration: a physically integrated molecular process. *Cell.* 1996; 84:359–369. [PubMed: 8608589]
7. Mogilner A. Mathematics of cell motility: have we got its number? *J. Math. Biol.* 2009; 58:105–134. [PubMed: 18461331]
8. Dembo M. The mechanics of motility in dissociated cytoplasm. *Biophys. J.* 1986; 50:1165–1183. [PubMed: 3801576]
9. Dembo M, Harlow F. Cell motion, contractile networks, and the physics of interpenetrating reactive flow. *Biophys. J.* 1986; 50:109–121. [PubMed: 3730497]
10. Dembo M, Maltrud M, Harlow F. Numerical studies of unreactive contractile networks. *Biophys. J.* 1986; 50:123–137. [PubMed: 3755362]
11. DiMilla PA, Barbee K, Lauffenburger DA. Mathematical model for the effects of adhesion and mechanics on cell migration speed. *Biophys. J.* 1991; 60:15–37. [PubMed: 1883934]
12. Mogilner A, Oster G. Cell motility driven by actin polymerization. *Biophys. J.* 1996; 71:3030–3045. [PubMed: 8968574]
13. Mogilner A, Verzi D. A Simple 1-D Physical Model for the Crawling Nematode Sperm Cell. *J. Stat. Phys.* 2003; 110:1169–1189. [PubMed: 19277214]
14. Rubinstein B, Jacobson K, Mogilner A. Multiscale two-dimensional modeling of a motile simple-shaped cell. *Multiscale Model. Simul.* 2005; 3:413–439. [PubMed: 19116671]
15. Novak IL, Slepchenko BM, Mogilner A. Quantitative analysis of G-actin transport in motile cells. *Biophys. J.* 2008; 95:1627–1638. [PubMed: 18502800]
16. Zajac M, Dacanay B, Mohler WA, Wolgemuth CW. Depolymerization-driven flow in nematode spermatozoa relates crawling speed to size and shape. *Biophys. J.* 2008; 94:3810–3823. [PubMed: 18227129]
17. Herant M, Dembo M. Form and function in cell motility: from fibroblasts to keratocytes. *Biophys. J.* 2010; 98:1408–1417. [PubMed: 20409459]
18. Herant M, Dembo M. Cytopede: a three-dimensional tool for modeling cell motility on a flat surface. *J. Comput. Biol.* 2010; 17:1639–1677. [PubMed: 20958108]
19. Ferziger, JH.; Peric, M. *Computational Methods for Fluid Dynamics*. third ed.. Springer; 2002.
20. Baines MJ, Hubbard ME, Jimack PK. A moving mesh finite element algorithm for the adaptive solution of time-dependent partial differential equations with moving boundaries. *Appl. Num. Math.* 2005; 54:450–469.
21. Osher S, Sethian JA. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.* 1988; 79:12–49.
22. Johansen H, Colella P. A Cartesian Grid Embedded Boundary Method for Poisson's Equation on Irregular Domains. *J. Comput. Phys.* 1998; 147:60–85.
23. Glimm J, Li X, Liu Y, Xu Z, Zhao N. Conservative front tracking with improved accuracy. *SIAM J. Numer. Anal.* 2003; 41:1926–1947.
24. Colella P, Graves DT, Keen BJ, Modiano D. A Cartesian grid embedded boundary method for hyperbolic conservation laws. *J. Comput. Phys.* 2006; 211:347–366.
25. Gatti-Bono C, Colella P, Trebovitch D. A second-order accurate conservative front-tracking method in one dimension. *SIAM J. Sci. Comput.* 2010; 31:4795–4813.

26. Wolgemuth CW, Zajac M. The moving boundary node method: A level set-based, finite volume algorithm with applications to cell motility. *J. Comput. Phys.* 2010; 229:7287–7308. [PubMed: 20689723]
27. Glimm J, Grove J, Lindquist B, McBryan OA, Tryggvason G. The bifurcation of tracked scalar waves. *SIAM J. Sci. Stat. Comput.* 1988; 9:61–79.
28. Du J, Fix B, Glimm J, Jia X, Li X, Li Y, Wu L. A simple package for front tracking. *J. Comput. Phys.* 2006; 213:613–628.
29. Udaykumar HS, Shyy W. Simulation of interfacial instabilities during solidification – I. Conduction and capillarity effects. *Int. J. Heat Mass Transfer.* 1995; 38:2057–2073.
30. Udaykumar HS, Mittal R, Rampunggoon P, Khanna A. A Sharp Interface Cartesian Grid Method for Simulating Flows with Complex Moving Boundaries. *J. Comput. Phys.* 2001; 174:345–380.
31. Hirt CW, Nichols BD. Volume of fluid (VOF) method for dynamics of free boundaries. *J. Comput. Phys.* 1981; 14:201–225.
32. Okabe, A.; Boots, B.; Sugihara, K.; Chiu, SN. *Spatial Tessellations – Concepts and Applications of Voronoi Diagrams*. 2nd ed.. John Wiley & Sons; 2000.
33. Bottino D, Mogilner A, Roberts T, Stewart M, Oster G. How nematode sperm crawl. *J. Cell Sci.* 2002; 115:367–384. [PubMed: 11839788]
34. Sibson, R. A brief description of natural neighbor interpolation. In: Barnett, V., editor. *Interpreting Multivariate Data*. Chichester: John Wiley & Sons; 1981. p. 21–36.
35. Vatti BR. A generic solution to polygon clipping. *Commun ACM.* 1992; 35:56–63.
36. Abu Shah E, Keren K. Mechanical forces and feedbacks in cell motility. *Curr. Opin. Cell Biol.* 2013; 25:550–557. [PubMed: 23860439]

Figure 1. Generation of control volumes by Voronoi decomposition. The blow-up of the delineated square at the right includes meshing parameters and illustrates a simple way of constructing a boundary layer of irregular Voronoi cells. The “ghost” points (shown in red), obtained by reflecting the adjacent grid points in the front, are used to “close” a Voronoi cell.

Figure 2.

Creation of an additional Voronoi cell ω_0 around the node r_0 that becomes an interior point of a moving domain. Because the masses previously assigned to cells ω_1 , ω_2 , and ω_3 are not reallocated, the corresponding unknown u_0 is initialized according to Eq (7), whereas the unknowns, u_1 , u_2 , u_3 remain unchanged.

Figure 3.

Voronoi cells ω_1 , ω_2 , and ω_3 are redrawn when the node r_0 becomes an exterior point, as the cell 0 that has included r_0 is eliminated. The mass, however, is not reallocated, and the unknowns u_1 , u_2 , u_3 are reinitialized according to Eqs (8).

Figure 4.

Diffusion in a moving square box. A numerical solution, upper left, obtained for $D=1$, $v_b=1$, $a=2$, is compared with the corresponding exact solution, upper right. In the lower panel, plots of L_∞ - (left) and L_2 - (right) norms of absolute error as functions of the mesh size h illustrate convergence of the algorithm. All results are based on solutions at $t = 0.5$.

Figure 5. Diffusion in a moving circular domain: testing against exact steady state. Convergence of L_∞ - norm error, E_∞ , is affected by reinitialization of unknowns at the boundary, but L_2 - norm error, E_2 , is $O(h^{1.79})$, which is similar to E_2 for diffusion in a moving square. All results are based on solutions at $t = 0.2$.

Figure 6.

Testing time-varying solutions. Solution at $t = 0.2$ (upper left) is compared, along the equator, with a precise numerical solution obtained by a regular VCell solver (right). Lower panel: convergence of E_∞ and E_2 ; errors of the solutions at $t = 0.2$ are computed against a precise numerical solution obtained with COMSOL®.

Figure 7.

A diffusion-advection system in a moving circle. A snapshot of a numerical solution at $t = 0.2$, left, and plots of $E_\infty(h)$ (circles) and $E_2(h)$ (triangles), right, computed for solutions at $t = 0.2$.