Journal of Computational Physics **157,** 143–180 (2000)
doi:10.1006/jcph.1999.6369, available online at http://www.idealibrary.com on IDEAL®

A Cartesian Grid Finite-Volume Method for the Advection-Diffusion Equation in Irregular Geometries

Donna Calhoun*,‡ and Randall J. LeVeque*,†,§

*Department of Applied Mathematics and †Department of Mathematics, University of Washington, Box 352420, Seattle, Washington 98195-2420
E-mail: ‡calhoun@amath.washington.edu and §rjl@amath.washington.edu

Received September 10, 1998; revised April 2, 1999

We present a fully conservative, high-resolution, finite volume algorithm for advection-diffusion equations in irregular geometries. The algorithm uses a Cartesian grid in which some cells are cut by the embedded boundary. A novel feature is the use of a "capacity function" to model the fact that some cells are only partially available to the fluid. The advection portion then uses the explicit wave-propagation methods implemented in CLAWPACK, and is stable for Courant numbers up to 1. Diffusion is modelled with an implicit finite-volume algorithm. Results are shown for several geometries. Convergence is verified and the 1-norm order of accuracy is found to be-tween 1.2 and 2 depending on the geometry and Peclet number. Software is available on the web.

Key Words: Finite-volume; advection-diffusion; Cartesian grid; embedded boundary; high-resolution; software.

1. INTRODUCTION

We consider the advection-diffusion equation

$$\kappa q_t + \nabla \cdot (\mathbf{u}q) = \nabla \cdot (D \nabla q) \tag{1}$$

in two space dimensions, where $q(x, y, t)$ is a density or concentration, $\mathbf{u}(x, y, t)$ is the specified velocity field, $D(x, y)$ is the diffusion coefficient, and $\kappa(x, y)$ is a capacity function. This function could represent, for example, heat capacity if q is the temperature, or porosity in a porous medium if q is the concentration of a transported solute. A fundamental feature of the numerical method developed here is the use of a discrete version of κ which also incorporates information about the fraction of a computational grid cell which

0021-9991/00 $35.00

lies in the physical domain. This is used in order to apply the method on Cartesian grids with embedded boundaries. Our goal is to solve (1) is geometrically complex regions with impermeable boundaries where the fluid flow must be tangent to the physical domain and the flux of q is specified (e.g., no flux). Rather than attempting to determine a grid which conforms to the boundaries, we wish to use a uniform Cartesian grid in which the physical domain is embedded. The main advantages to using a Cartesian grid are that all difficulties associated with unstructured grid generation can be avoided, and standard finite volume or finite difference discretizations of the PDE on a uniform grid can be used, away from the boundaries at least.

This Cartesian grid approach is becoming increasingly popular for problems with complex geometry where grids conforming to the boundaries are difficult to generate. Some finite-volume approaches most closely related to ours can be found in [1, 8, 9, 12, 13, 17, 28–30]. Finite difference and finite element methods on Cartesian grids with embedded boundaries have also been developed for many different PDEs, for example, [10, 11, 14, 15, 24, 23, 22, 25–27, 31, 33, 34].

Here we develop a finite volume method which is essentially second order accurate for smooth q in very general geometries and which can also handle steep gradients in q (even discontinuities in q if $D \equiv 0$). This is accomplished by using the high-resolution shock-capturing methods of CLAWPACK for the advection portion of the method, coupled with an implicit finite-volume discretization of the diffusion equation.

The principle contribution of this paper is an algorithm with the following characteristics:

- Arbitrary geometry can be handled on a Cartesian grid, though we assume that the geometry is well resolved by a piecewise linear function on the underlying grid.
- After some preprocessing required to handle the irregular geometry, small irregular cells which are cut off by the boundary are updated by exactly the same formulas as regular cells away from the boundary.
- The advection portion of the algorithm is explicit and the time step is determined by the size of the regular grid cells, with no restriction caused by the small irregular cells.
- The accuracy and resolution in the irregular cells adjacent to the boundaries match that obtained in the regular cells and are essentially as good as would be obtained on similar problems with a grid conforming to the boundary.
- The algorithm is implemented using the CLAWPACK software and is freely available on the web, see

`http://www.amath.washington.edu/~rjl/clawpack/cartesian`

By assuming that the geometry is well resolved on the grid, we assume in particular that each finite volume cell is a standard Cartesian grid cell which is possibly sliced by a linear approximation to the boundary, as shown in Fig. 5, for example. Hence every cell is a simple polygon with at most five sides and the standard (i, j) labeling of grid cells can continue to be used. For geometry with more complications, for example, a thin trailing edge which cuts a cell into two distinct pieces, a more complicated data structure would be required. The methods proposed here could still be used but could not be implemented as directly in CLAWPACK. See [8] for one discussion of such geometries in the context of elliptic equations.

The diffusion and advection algorithms will first be described and tested separately and are then combined using a standard fractional step approach. The methods are described in

two space dimensions, but all of the essential ideas can be extended directly to three space dimensions. Some other extensions are discussed in Section 7.

2. NOTATION AND THE USE OF A CAPACITY FUNCTION

We consider finite volume methods for the general conservation law

$$\kappa(x, y)q_t(x, y, t) + f(x, y, t, q)_x + g(x, y, t, q)_y = 0, \tag{2}$$

where f and g are the flux function and κ is the capacity. In particular, we consider variable-coefficient linear equations including the *diffusion equation* in which

$$f = -D(x, y)q_x, \qquad g = -D(x, y)q_y \tag{3}$$

and the advection equation in which

$$f = u(x, y, t)q, \qquad g = v(x, y, t)q \tag{4}$$

and finally the advection diffusion equation which includes both advective and diffusive fluxes,

$$f = u(x, y, t)q - D(x, y)q_x, \qquad g = v(x, y, t)q - D(x, y)q_y. \tag{5}$$

Such equations arise from the integral form of the conservation law over an arbitrary region Ω

$$\frac{\partial}{\partial t}\int_\Omega \kappa(x, y)q(x, y, t)\,dx\,dy + \int_{\partial\Omega} \mathbf{n}\cdot\mathbf{f}\,ds = 0, \tag{6}$$

where $\mathbf{n}$ is the outward pointing normal at the boundary $\partial\Omega$ and $\mathbf{f} = (f, g)$ is the flux vector. The integral of κq is conserved up to fluxes through the boundary. The function κ represents the capacity of the medium as illustrated in the following two examples:

1. In heat conduction, if q is the temperature then (3) gives Fourier's Law of heat conduction for the flux of heat energy. The energy stored in the region Ω is $\int_\Omega \kappa q\,dx\,dy$ where κ is the heat capacity of the material and (6) arises from conservation of energy.

2. If q represents the concentration of a tracer in some fluid saturating a porous medium, and if $\kappa(x, y)$ is the porosity of the medium so that $\int_\Omega \kappa(x, y)\,dx\,dy$ is the volume of Ω which is available to the fluid, then $\int_\Omega \kappa q\,dx\,dy$ is the total mass of the tracer in the region Ω. If the fluid is flowing through the porous medium with the Darcy velocity (u, v) (defined below) then the conservation law (6) arises with fluxes (4). If the tracer also diffuses in the fluid then diffusive fluxes are also present.

Note that the integral $\int_\Omega \kappa q\,dx\,dy$ can also be interpreted as an integral of q over Ω with respect to the measure $d\mu = \kappa\,dx\,dy$. This interpretation may be useful in understanding our Cartesian grid finite volume integrals.

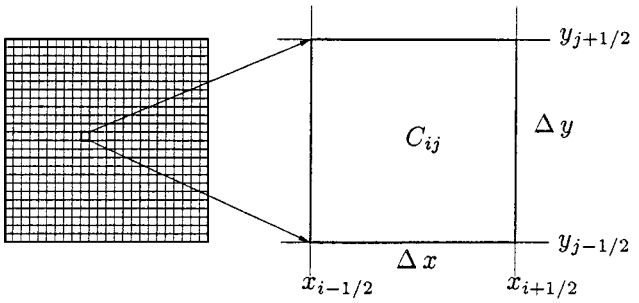

FIG. 1. Cartesian grid with cell C_{ij} shown in enlargement.

2.1. *Incorporating the Capacity Function into Finite Volume Methods on Regular Grids*

A finite volume method is based on dividing the physical region into finite volumes (i.e., grid cells) and representing the numerical approximation by cell averages over those cells. The cell average is updated in each time step by approximations to the flux through each edge of the cell. We are concerned here with a Cartesian grid of the form shown in Fig. 1. We start by considering a uniform grid with spacing Δx and Δy on a rectangular region, with no irregular geometry to handle.

The grid is indexed by (i, j) and Q_{ij}^n represents the approximation to the average values of q on cell (i, j) at time t_n with respect to the measure $\kappa\, dx\, dy$,

$$Q_{ij}^n \approx \frac{\int_{C_{ij}} \kappa(x, y) q(x, y, t_n)\, dx\, dy}{\int_{C_{ij}} \kappa(x, y)\, dx\, dy}. \tag{7}$$

Using the fact that the area of cell C_{ij} is $\Delta x \Delta y$, we define the average value of κ over cell C_{ij} as

$$\kappa_{ij} = \frac{1}{\Delta x \Delta y} \int_{C_{ij}} \kappa(x, y)\, dx\, dy. \tag{8}$$

Then we can write (7) as

$$Q_{ij}^n \approx \frac{1}{\kappa_{ij} \Delta x \Delta y} \int_{C_{ij}} \kappa q\, dx\, dy. \tag{9}$$

Note that if $\kappa \equiv 1$, then $\kappa_{ij} \equiv 1$ and (9) reduces to the standard cell average.

Let $F_{i-1/2,\,j}$ and $G_{i,\,j-1/2}$ represent the numerical fluxes at the left edge and bottom edge of the cell, respectively. These fluxes will be measured in units of q flowing normal to the edge per unit time per unit length. Hence the total flux through the left edge over the time step Δt, for example, is given by $\Delta t \Delta y F_{i-1/2,\,j}$. In porous media problems these fluxes are computed using the *Darcy* velocity and in heat transfer problems, they are computed using an *effective conductivity*. We explain both of these briefly now.

Figure 2 shows a cartoon of flow in a porous medium where the shaded regions represent rock or sand. The fluid velocity may be large in the regions where fluid is flowing, but it is zero in solid regions where there is no fluid. Hence the average horizontal velocity across a vertical line segment such as the one sketched in the figure is less than the fluid velocity at any particular point in the fluid. The u-component of the *Darcy velocity* at a point is this average velocity taken over an "infinitesimal" vertical segment centered at the point, which is short relative to the scale at which this average fluid velocity varies, but still long relative to the structure of the porous medium. Hence the Darcy velocity is roughly equal to the

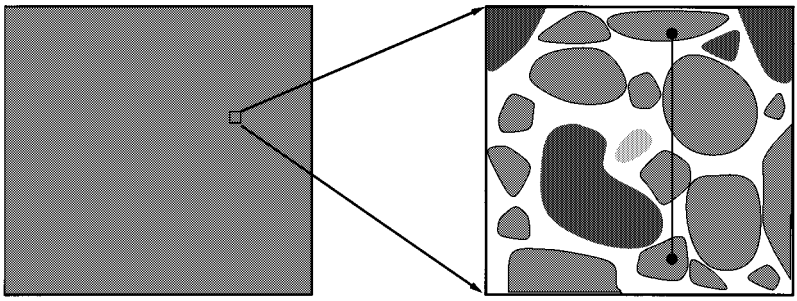

FIG. 2. Example of porous medium. Quantities such as velocity are averaged over an infinitesimal line segment, such as the one shown.

average fluid velocity multiplied by the fraction of the infinitesmal vertical line which lies within the fluid.

At the edge of a finite volume grid cell (which is assumed to be large relative to the scale of the porous medium), it is clearly the Darcy velocity and corresponding flux which determines the flux into the grid cell.

The porosity of a porous medium at a point is defined analogously as the fraction of an "infinitesimal" region Ω about the point which is occupied by the fluid, where Ω is again large relative to the porous structure. Our approach to handling Cartesian grids in complex geometries, which is described below, can be viewed as an extension of these ideas to the case where the solid is concentrated on some portion of the grid cell lying outside the fluid domain, rather than being distributed throughout the grid cell. As we show later, we can also define appropriate fluxes across edges which are only partially in the physical domain of interest.

In heat flow in composite materials, one can analogously define an *effective conductivity* as the average of the conductivity over an "infinitesmal" line segment which is short relative to the length over which the average conductivity varies, but long relative to any inhomogeneities in the composite material.

Using numerical fluxes based on Darcy velocities and effective conductivities, an explicit discretization of the conservation law (6) is given by

$$\frac{\kappa_{ij}\Delta x\Delta y Q_{ij}^{n+1} - \kappa_{ij}\Delta x\Delta y Q_{ij}^{n}}{\Delta t} + \Delta y\left[F_{i+1/2,j}^{n} - F_{i-1/2,j}^{n}\right] + \Delta x\left[G_{i,j+1/2}^{n} - G_{i,j-1/2}^{n}\right] = 0. \tag{10}$$

Rearranging terms, we can write an explicit formula for updating Q_{ij}^{n+1} as

$$Q_{ij}^{n+1} = Q_{ij}^{n} - \frac{\Delta t}{\kappa_{ij}\Delta x}\left[F_{i+1/2,j}^{n} - F_{i-1/2,j}^{n}\right] - \frac{\Delta t}{\kappa_{ij}\Delta y}\left[G_{i,j+1/2}^{n} - G_{i,j-1/2}^{n}\right]. \tag{11}$$

Note that (11) is in conservation form in the sense that

$$\Delta x\Delta y\sum_{ij}\kappa_{ij}Q_{ij}^{n} \tag{12}$$

will be conserved up to fluxes through the boundary of the computational domain. This type of "capacity form" differencing is discussed in more detail in the context of hyperbolic conservation laws in [21]. In many problems $\kappa(x, y) \equiv 1$. However, even for these problems the idea of introducing a capacity function becomes useful in handling complex geometries, as described in the next section.

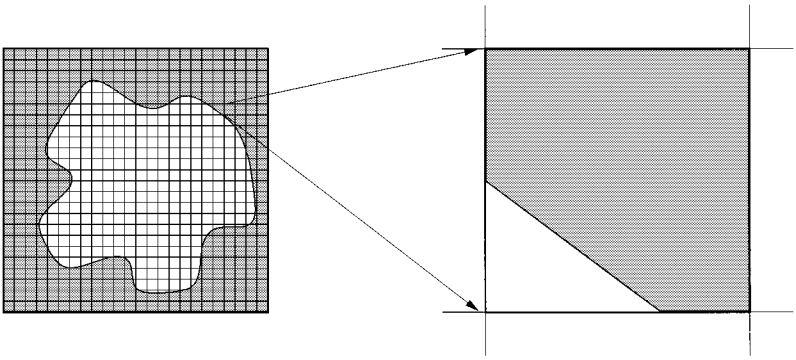

FIG. 3. Irregular flow domain (unshaded region) embedded in a larger computational domain. Enlargement shows an irregular grid cell cut by a piecewise linear boundary. The capacity of the irregular cell is the unshaded fraction of the cell in the irregular flow domain.

2.2. *Using the Capacity Function to Represent Irregular Regions*

In addition to physical interpretations of heat capacity and porosity, we can use the capacity function κ to embed an *irregular flow domain*, such as the one shown in Fig. 3, into a Cartesian computational domain. The natural extension of the interpretation of κ given in the preceding section is to define κ to be identically 0 everywhere outside of the irregular flow domain. This extension preserves the notion that the values of κ determine the *capacity* of the medium to hold the advected or diffused quantity q. Outside of the physical domain of interest, the medium has no capacity to hold q.

We now view the computational domain as the full rectangular region in Fig. 3, and refer to the original physical domain as the *flow domain* and to the portion which lies outside the original domain (the shaded region in Fig. 3) as the *no-flow domain*.

With this extended interpretation in mind, we now consider an irregular grid cell such as the one magnified in Fig. 3. We define κ_{ij} for this cell as in (8), after extending $\kappa(x, y)$ to be 0 in the no-flow domain. Note that $\kappa_{ij}\Delta x\Delta y$ gives the total capacity of the grid cell. In particular, in the simplest case where $\kappa(x, y) \equiv 1$ in the flow domain, $\kappa_{ij}\Delta x\Delta y$ is simply the area of the portion of this grid cell which lies in the flow domain (the unshaded portion).

To obtain a definition of the numerical fluxes $F_{i-1/2,j}$ and $G_{i,j-1/2}$ which is consistent with our previous use of these quantities in update formula (11), we first let $\hat{F}_{i-1/2,j}$ and $\hat{G}_{i,j-1/2}$ denote the average flux across that portion of each edge which is in the flow domain, as indicated in Fig. 4. Then we let $h_{i-1/2,j}$ and $h_{i,j-1/2}$ be the lengths of these edge portions

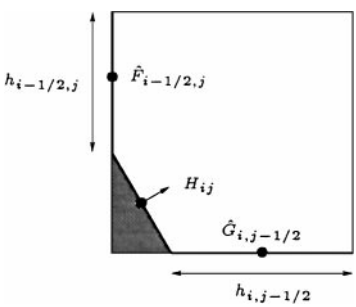

FIG. 4. Fluxes at edges of irregular cells.

to the left and below the cell C_{ij}, respectively. One or more of these may be zero. The total flux across each edge is then given by $h_{i-1/2,j}\hat{F}_{i-1/2,j}$ and $h_{i,j-1/2}\hat{G}_{i,j-1/2}$.

If we now define the fluxes

$$\begin{aligned} F_{i-1/2,j} &= \frac{h_{i-1/2,j}}{\Delta y}\hat{F}_{i-1/2,j} \\ G_{i,j-1/2} &= \frac{h_{i,j-1/2}}{\Delta x}\hat{G}_{i,j-1/2} \end{aligned} \tag{13}$$

we see that $\Delta y F_{i-1/2,j}$ and $\Delta x G_{i,j-1/2}$ are the total fluxes across the left edge and bottom edge, respectively, of cell C_{ij}. Hence, this definition of the fluxes on irregular cells is consistent with our definition on the regular cells.

To develop a general updating formula for irregular cells, we also introduce a flux H_{ij} across the boundary segment, as indicated in Fig. 4. Let the length of this boundary segment be denoted by l_{ij}. Then the general updating formula is

$$\begin{aligned} Q_{ij}^{n+1} = Q_{ij}^n &- \frac{\Delta t}{\kappa_{ij}\Delta x}\left[F_{i+1/2,j} - F_{i-1/2,j}\right] - \frac{\Delta t}{\kappa_{ij}\Delta y}\left[G_{i,j+1/2} - G_{i,j-1/2}\right] \\ &- \frac{\Delta t}{\kappa_{ij}\Delta x\Delta y}[l_{ij}H_{ij}]. \end{aligned} \tag{14}$$

For many problems $H_{ij} \equiv 0$. This is the correct boundary condition for tracer concentration in a flow bounded by a wall and also for heat conduction if the wall is insulated. If $H_{ij} = 0$, the update formula (14) reduces to exactly that given by (11).

For cells completely outside of the flow domain, we have $\kappa_{ij} = 0$ and the updating formula (11) appears to be invalid. However, if we note that fluxes $F_{i+1/2,j}$, $F_{i-1/2,j}$, $G_{i,j+1/2}$, and $G_{i,j-1/2}$ will always be zero for such cells, the earlier formula (10) will always be valid. For computational convenience, however, we want to apply the fomula (11) or (14) everywhere. To do this, we can set κ_{ij} to some arbitrary but nonzero value for cells entirely in the no-flow region, since the fact that the fluxes in this region will be zero will ensure that the value of Q_{ij}^n remains zero.

A key feature of our approach is that we have eliminated irregular grid cells from the geometric structure of our problem and instead take the viewpoint that some cells may have small capacity. This simplifies the structure of the computer programs and allows software packages such as CLAWPACK to be applied directly.

As we will later see, by shifting the small cell problem from one of cells with small volume to one of cells with small capacity, we also provide ourselves with a natural way to overcome the instability problems typically associated with the small cells. Although our capacity κ_{ij} is the same as the quantity Λ_{ij} introduced by Chern and Colella [7] in the context of front tracking, we use a different modification to the fluxes which is based more directly on wave propagation and which we believe gives better accuracy.

3. THE DIFFUSION EQUATION

We now develop a finite volume method for the diffusion equation in the flow domain. The problem we wish to solve numerically is given by

$$\kappa q_t = \nabla \cdot (D(x,y)\nabla q) \qquad \text{in } \Omega \tag{15}$$

subject to Neumann boundary conditions on the boundary of Ω which we use to define flux values H_{ij} used in Eq. (14). As described in the previous section, we define κ to be identically 0 for (x, y) in the no-flow domain. The fluxes are given by Eqs. (3).

As suggested in the previous discussion, it will be convenient to define numerical fluxes $F_{i-1/2,j}$ and $G_{i,j-1/2}$ so that the total flux across an edge of length Δx or Δy is given by $\Delta y F_{i-1/2,j}$ or $\Delta x G_{i,j-1/2}$. With this in mind, we approximate quantites q_x and q_y on regular cell edges by differencing across the edge and use these to approximate the fluxes as

$$F_{i-1/2,j} = -D_{i-1/2,j}\frac{Q_{ij} - Q_{i-1,j}}{\Delta x} \tag{16}$$

and

$$G_{i,j-1/2} = -D_{i,j-1/2}\frac{Q_{ij} - Q_{i,j-1}}{\Delta y}. \tag{17}$$

As before, these values are in units of q flowing normal to a cell edge per unit time per unit length.

On irregular cell edges, we approximate quantities $\hat{q}_x$ and $\hat{q}_y$ at the center of that portion of the edge which is in the flow domain by interpolating between values Q_k, $k = 1, \ldots, 4$, in four neighboring cells. We interpret these values Q_k as being located at centers of mass of the regular and irregular cells. After we have located the centers of four cells surrounding the partial edge at which we wish to compute the flux, we use the coordinates (ξ_k, η_k), $k = 1, \ldots, 4$, of these four centers and corresponding values Q_k to determine the coefficients a, b, c, d of the bilinear function

$$\hat{q}(x, y) = ax + by + cxy + d. \tag{18}$$

We then use these coefficients to approximate $\hat{q}_x$ or $\hat{q}_y$ at the center (x_e, y_e) of a partial edge. For example, we have

$$\hat{q}_x(x_e, y_e) \approx a + cy_e. \tag{19}$$

In Fig. 5, the values Q_{ij}, $Q_{i-1,j}$, $Q_{i-1,j+1}$, and $Q_{i,j+1}$ are used to approximate values $\hat{q}_x$ at $x_{i-1/2}$ and $\hat{q}_y$ at $y_{j+1/2}$.

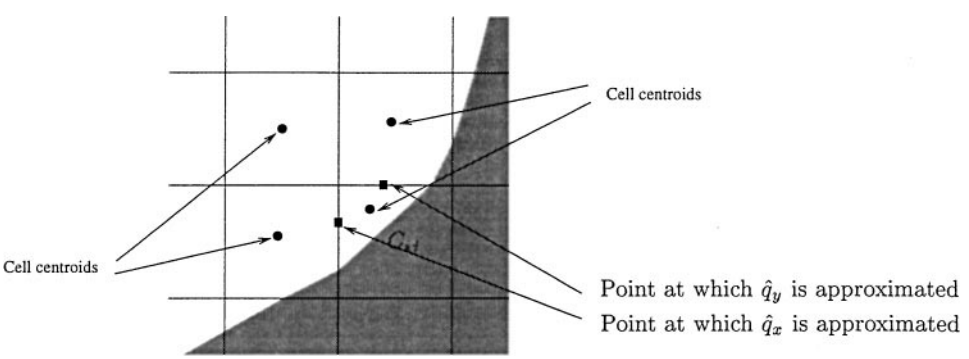

FIG. 5. Typical set of points, located at center of mass of irregular cells, used to approximate $\hat{q}_x$ at edge $x_{i-1/2}$ and $\hat{q}_y$ at $y_{j+1/2}$. The fluxes $\hat{F}_{i-1/2,j}$ and $\hat{G}_{i,j+1/2}$ are computed using these approximations.

Other researchers have proposed methods for approximating diffusive fluxes at partial cell edges to second order accuracy [18]. Although our method is only formally first order at the cell edges, we have found that we obtain results which are still globally second order.

We use the approximations given by (18) to compute values $\hat{F}_{i-1/2,j}$ and $\hat{G}_{i,j+1/2}$ using

$$\hat{F}_{i-1/2,j} = -D_{i-1/2,j}\hat{q}_x \tag{20}$$

$$\hat{G}_{i,j+1/2} = -D_{i,j+1/2}\hat{q}_y. \tag{21}$$

The numerical fluxes $F_{i-1/2,j}$ and $G_{i,j+1/2}$ are then defined as in (13).

To discretize the heat equation given by (15), we use a standard Crank–Nicolson method at all grid points. First, we define an approximation to the integral

$$-\int_{\partial C_{ij}} \mathbf{f}\cdot\mathbf{n}\,ds \tag{22}$$

in (6) as

$$W_{ij}^n = -\{\Delta y\big(F_{i+1/2,j}^n - F_{i-1/2,j}^n\big) + \Delta x\big(G_{i,j+1/2}^n - G_{i,j-1/2}^n\big)\}. \tag{23}$$

Then an implicit discretization of (6) is given by

$$\kappa_{ij}\Delta x\Delta y\frac{Q_{ij}^{n+1}-Q_{ij}^n}{\Delta t} = \frac{1}{2}\big(W_{ij}^n + W_{ij}^{n+1}\big) \tag{24}$$

which, upon rearrangement, is

$$Q_{ij}^{n+1} - \frac{\Delta t}{2\kappa_{ij}\Delta x\Delta y}W_{ij}^{n+1} = Q_{ij}^n + \frac{\Delta t}{2\kappa_{ij}\Delta x\Delta y}W_{ij}^n. \tag{25}$$

In the above, we assumed that there was no incoming flux from the boundary. If we assume that this flux, given again by H_{ij}, is non-zero and possibly time dependent, then our update formula looks like

$$Q_{ij}^{n+1} - \frac{\Delta t}{2\kappa_{ij}\Delta x\Delta y}W_{ij}^{n+1} = Q_{ij}^n + \frac{\Delta t}{2\kappa_{ij}\Delta x\Delta y}W_{ij}^n + \frac{\Delta t l_{ij}}{\kappa_{ij}\Delta x\Delta y}H_{ij}^{n+1/2}, \tag{26}$$

where again l_{ij} is the length of that portion of the boundary of the irregular flow domain which is in cell C_{ij}.

Because κ_{ij} varies from grid cell to grid cell, the matrix corresponding to the above system of linear equation is not symmetric. However, these equations are still easily solved using the iterative method designed for such systems. The one which was chosen here and which seems to give quite good results is the stabilized bi-conjugate gradient method (BiCGSTAB) developed by Van der Vorst [33].

As for the stability of the numerical scheme described above, we have observed that in practice, the presence of small cells near the boundary does not place any unreasonable restrictions on the size time step we can take to maintain stability. This to be expected, since Crank–Nicolson is an unconditionally stable scheme.

4. THE ADVECTION EQUATION

The advection portion of the algorithm is implemented using the CLAWPACK software, based on the high-resolution wave-propagation algorithm described in [22]. The special case of advection is discussed in detail in [21] and will be described briefly below.

On the grid cell (i, j) we require the average normal velocity $U_{i-1/2,j}$ along the left edge of the cell and $V_{i,j-1/2}$ along the bottom edge. For simplicity in two dimensions we assume that an incompressible velocity field is specified by means of a stream function $\psi(x, y)$. This is not necessary for our algorithm and would not be appropriate in three dimensions, where a different approach would be needed to determine the proper Darcy velocities. The velocity is determined from the stream function by

$$u(x, y) = \psi_y(x, y), \qquad v(x, y) = -\psi_x(x, y).$$

Then the average velocity along each edge is easily computed by differencing the stream function at the corners

$$\begin{aligned} U_{i-1/2,j} &= \frac{1}{\Delta y}\int_{y_{j-1/2}}^{y_{j+1/2}} \psi_y(x, y)\, dy \\ &= \frac{1}{\Delta y}\left[\psi\left(x_{i-1/2}, y_{j+1/2}\right) - \psi\left(x_{i-1/2}, y_{j-1/2}\right)\right] \end{aligned}$$

at the left edge, and

$$\begin{aligned} V_{i,j-1/2} &= -\frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} \psi_x(x, y)\, dy \\ &= -\frac{1}{\Delta x}\left[\psi\left(x_{i+1/2}, y_{j-1/2}\right) - \psi\left(x_{i-1/2}, y_{j-1/2}\right)\right] \end{aligned}$$

at the bottom edge. Note that this will automatically satisfy the discrete divergence-free condition

$$\left(U_{i+1/2,j} - U_{i-1/2,j}\right)/\Delta x + \left(V_{i,j+1/2} - V_{i,j-1/2}\right)/\Delta y = 0.$$

In the case of an irregular flow domain, ψ must be constant along any physical boundary and we assume that ψ is defined over the entire computational domain in such a way that ψ is identically constant in no-flow regions. Then the above formulas give zero average velocity along any cell edge that lies entirely in the no-flow domain. Moreover, the average velocity is computed correctly in cases where the edge is cut by the physical boundary, e.g., at the left and bottom edges of the cell shown in Fig. 3. This average velocity can be interpreted as analogous to the Darcy velocity described in Section 4. It is an effective velocity over the entire cell edge, rather than a pointwise approximation to the actual fluid velocity.

To explain the advection algorithm on irregular cells, it is illuminating to first consider the case shown in Fig. 6, where there is a horizontal wall cutting through some row of grid cells. Constant flow parallel to the wall is specified by

$$\psi(x, y) = \begin{cases} u_0 y & \text{if } y \ge y_0 \\ u_0 y_0 & \text{if } y \le y_0 \end{cases} \qquad u(x, y) = \begin{cases} u_0 & \text{if } y > y_0 \\ 0 & \text{if } y < y_0 \end{cases} \qquad v(x, y) \equiv 0.$$

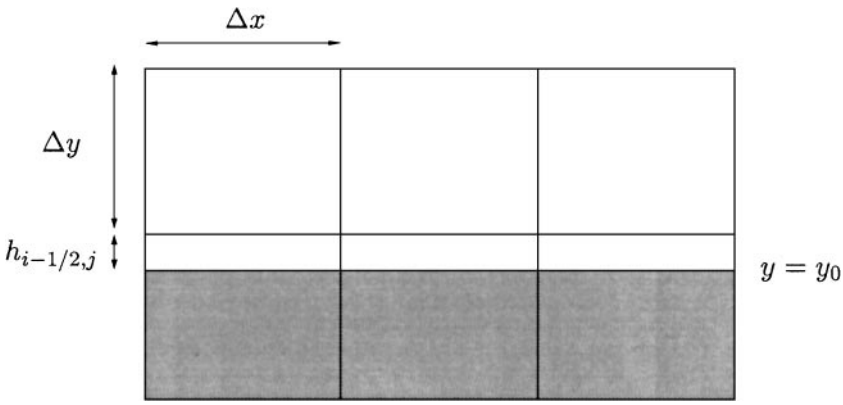

FIG. 6. Grid cells in the case where the boundary is parallel to the grid and cuts through one row of cells, creating thin sliver cells.

For grid cells entirely in the flow domain, $U_{i-1/2,j} = u_0$, but for the cut cells $U_{i-1/2,j} = u_0 h_{i-1/2,j}\,\Delta y$, where $h_{i-1/2,j}$ is the length of the left cell edge which lies in the flow domain (see Fig. 6). Consider the simple upwind method, in which case the fluxes will be given by

$$F_{i-1/2,j} = U_{i-1/2,j}\, Q_{i-1,j}, \qquad G_{i,j-1/2} = 0$$

and the update formula (11) reduces to

$$Q_{ij}^{n+1} = Q_{ij}^{n} - \frac{\Delta t}{\kappa_{ij}\,\Delta x\,\Delta y}\left[\Delta y\left(F_{i+1/2,j} - F_{i-1/2,j}\right)\right]. \tag{27}$$

Note that $\kappa_{ij}\,\Delta x\,\Delta y$ is the cell area.

For cells entirely in the flow domain, we have $\kappa_{ij} = 1$, $U_{i-1/2,j} = u_0$, and this reduces to the one-dimensional upwind method

$$Q_{ij}^{n+1} = Q_{ij}^{n} - \frac{u_0 \Delta t}{\Delta x}(Q_{ij} - Q_{i-1_i,j}). \tag{28}$$

The cut cells have area $\kappa_{ij}\,\Delta x\,\Delta y = h_{i-1/2,j}\,\Delta x$, and so $\kappa_{ij} = h_{i-1/2,j}/\Delta y$ while $U_{i-1/2,j} = u_0 h_{i-1/2,j}/\Delta y$. In this case the formula (27) also reduces to (28). Hence the cut cells are updated by the same one-dimensional method as the full cells, and advection through the row of cut cells along the wall exactly matches the advection elsewhere in the flow domain. Note that this is true regardless of the size of the cut cells, even in the limit $h_{i-1/2,j} \to 0$, as the cell becomes vanishingly small. The Courant number restriction is given by

$$|u_0 \Delta t/\Delta x| \le 1$$

regardless of the size of the cut cells. Small cells cause no stability problems in this special case since flow is in the direction where the cell width is always Δx and fluxes are properly calculated to vanish along with $h_{i-1/2,j}$.

Now consider flow at some angle to the grid. Figure 7a shows the interpretation of the multi-dimensional wave-propagation algorithm as described in [20, 21] on a regular grid. The shaded region represents a wave propagating from the left cell edge with horizontal

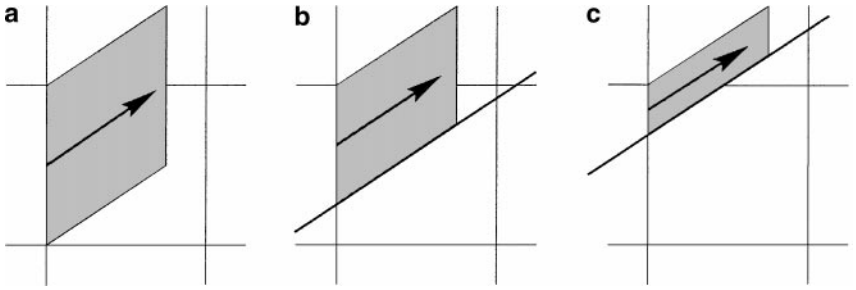

FIG. 7. (a) An advection wave propagating from the left edge moves at the fluid velocity into the adjacent cell and the cell above. (b) The wave in a cut cell at the boundary with the same velocity as in (a). In this case the cut cell is trapezoidal. (c) The situation when the cut cell is triangular, in which case more than half of the wave may leave the adjacent cell and go into the one above.

velocity $U_{i-1/2,j}$ and vertical velocity $V_{i,j+1/2}$, both taken to be positive in this illustration. The wave carries the jump $Q_{ij} - Q_{i-1,j}$ and leads to contributions to the two fluxes $F_{i-1/2,j}$ and $G_{i,j+1/2}$. The contribution to $F_{i-1/2,j}$ is due to the propagation of the rectangular region of area $\Delta y \Delta t U_{i-1/2,j}$ and jump $Q_{ij} - Q_{i-1,j}$ into cell (i, j), yielding

$$\text{contribution to } F_{i-1/2,j}: \qquad U_{i-1/2,j}(Q_{ij} - Q_{i-1,j}). \tag{29}$$

The contribution to $G_{i,j+1/2}$ results from the "transverse propagation" (see [20, 21]) of a triangular region of area $(\Delta t U_{i-1/2,j})(\Delta t V_{i,j+1/2})$ propagating out of cell (i, j) and into cell $(i, j+1)$. This yields

$$\text{contribution to } G_{i,j+1/2}: \qquad \frac{\Delta t}{\Delta y} U_{i-1/2,j} V_{i,j+1/2}(Q_{ij} - Q_{i-1,j}). \tag{30}$$

Each of these fluxes also receives contributions from other jumps as well. (See [20, 21] for more details, and also the mechanism by which second-order corrections with limiters are added.)

Figures 7b and 7c show two situations where the grid cell is cut by a physical boundary. Note that much of what flows into cell (i, j) from the left flows out again through the top edge of the cell. Stability is often a problem in small cells with explicit Cartesian grid methods since we do not want the time step to be limited by the size of the smallest cell that happens to appear. One way to maintain stability is to make sure that the fluxes $F_{i-1/2,j}$ and $G_{i,j-1/2}$ are consistent with one another, so that summing the fluxes around the cell edges will give suitable cancellation of inflow and outflow. This is critical since we divide by the cell area in the updating formula. For tiny cells the cancellation must be nearly complete. One approach to ensure this is the "h-box method" developed in [3, 4], but here we seek something easier for the relatively simple case of advection parallel to the wall.

First consider the situation shown in Fig. 7b, where at least one edge of the cell lies entirely in the flow domain (the top edge in this figure). In this case stability is achieved without special effort by the use of the multi-dimensional wave-propagation algorithm which includes transverse fluxes, as described above and indicated in the figure. The standard updating formulas work with the fluxes and κ_{ij} defined as above, even when the cell area is very small, e.g., a thin sliver as in Fig. 6. The portion which flows out through the top

edge is correctly computed by the transverse flux formulas, which correctly assume that the portion of the wave passing out through the top of the cell is triangular in shape.

Unfortunately this formula can lead to unstable behavior in the case illustrated in Fig. 7c, where the portion of the wave passing out the top edge is not triangular. This can happen if the small cell itself is triangular, with only two of the edge lengths $h_{i\pm1/2,j}$, $h_{i,j\pm1/2}$ nonzero and these both shorter than the edges of regular cells. In this case it may happen that more than half of what flows into the cell through the left edge should flow out through the top edge, as illustrated in Fig. 7c. This cannot be properly modelled by the formulas (29) and (30).

One way to restore stability would be to handle the geometry of the wave explicitly in the calculation of fluxes, carefully computing the area of the portion of the wave in Fig. 7c which passes through the top edge and using this to correctly define the contribution to the flux $G_{i,j+1/2}$. This approach was used in [2, 18, 19] for the Euler equations. We wish to avoid this complication here and instead use the "Darcy velocities" and capacities κ_{ij} directly in CLAWPACK.

Rather than modifying the definition of the fluxes, we have found that an extremely effective method is obtained by a simple modification of κ_{ij} in such cells. By increasing κ_{ij} and hence slightly enlarging the cell's capacity we have found that we can compensate for the possible mismatch in fluxes defined by (29) and (30). After explaining and motivating this modification we will discuss its physical implications.

The change is accomplished by defining the capacity in all cells by a modified version of (8),

$$\hat{\kappa}_{ij} = \frac{r_{ij}}{\Delta x \Delta y} \int_{C_{ij}} \kappa(x, y)\, dx\, dy \tag{31}$$

in which we define

$$r_{ij} = \min\left(\frac{\Delta x}{h_{i-1/2,j}}, \frac{\Delta x}{h_{i+1/2,j}}, \frac{\Delta y}{h_{i,j-1/2}}, \frac{\Delta y}{h_{i,j+1/2}}\right).$$

Note that $r_{ij} \geq 1$ and that $r_{ij} = 1$ if at least one of the cell edges is entirely in the flow domain, in which case $\hat{\kappa}_{ij}$ reduces to the usual definition of κ_{ij}. In the troublesome case of a small triangular cell, $r_{ij} > 1$ and the capacity is increased. This is equivalent to redefining the area of such cells based on a thin sliver rather than a small triangle, the sliver obtained by replacing the larger of the two lengths ($h_{i-1/2,j}$ or $h_{i,j+1/2}$, say) by the full size mesh width (Δy or Δx, respectively). However, we do not actually modify the geometry of the grid at all; we simply increase the capacity of the small cell.

For simplicity in discussing and analyzing this algorithm, we will assume that $\kappa(x, y) \equiv 1$ in the flow region, so that

$$\hat{\kappa}_{ij} = \frac{A_{ij} r_{ij}}{\Delta x \Delta y}, \tag{32}$$

where A_{ij} is the cell area.

The motivation for this form of r_{ij} is the following. The algorithm remains stable in a thin sliver of a cell because when two sides have small length $O(\epsilon)$ (e.g., $h_{i-1/2,j}$ and $h_{i+1/2,j}$ in Fig. 7b) while $h_{i,j+1/2} = \Delta x$ has normal length, then the area is $A_{ij} = O(\epsilon \Delta x)$ while the Darcy velocity $U_{i-1/2,j}$ is $O(\epsilon)$. For a fixed grid these two quantities vanish at the same

rate as $\epsilon \to 0$ and the cell update can remain bounded. In a small triangular cell, on the other hand, if $h_{i-1/2,j} = O(\epsilon)$ and $h_{i,j+1/2} = O(\epsilon)$ also, then the area A_{ij} is $O(\epsilon^2)$ while the Darcy velocities are still only $O(\epsilon)$. Hence the cell update is $O(\Delta t/\epsilon)$ and blows up as $\epsilon \to 0$. This results in instability unless we use very small time steps based on the size of these cells. Enlarging κ as in (31) eliminates this possible instability.

In practice we have also found it useful to place an absolute lower bound on κ, setting

$$\kappa_{ij} = \max(\hat{\kappa}_{ij}, \delta), \tag{33}$$

where $\delta = 0.01\Delta x \Delta y$, for example. This eliminates the chance of instability due to rounding errors in extremely small cells and also ensures that κ_{ij} is set to a nonzero value for cells entirely within the no-flow domain.

With this modified definition of κ_{ij}, we have found that we can use the standard Courant number restriction $\max_{i,j} \nu_{ij} \le 1$ in CLAWPACK to choose the timestep and the method will be stable. The cell Courant number in each grid cell is defined by

$$\nu_{ij} = \max\left(\left|\frac{U_{i\pm1/2,j}\Delta t}{\kappa_{ij}\Delta x}\right|, \left|\frac{V_{i,j\pm1/2}\Delta t}{\kappa_{ij}\Delta y}\right|\right). \tag{34}$$

In the calculations below we have always chosen the timestep by imposing $\max_{i,j} \nu_{ij} = 0.9$. Note that on a uniform grid the Courant number would be defined by

$$\nu = \max_{x,y}\left(\left|\frac{u(x,y)\Delta t}{\Delta x}\right|, \left|\frac{v(x,y)\Delta t}{\Delta y}\right|\right). \tag{35}$$

As discussed in the Appendix, depending on the geometry, we may be able to take the cell Courant number to be as large as 2 in some cases and still have $\nu \le 1$. With this larger Courant number, we have confirmed that the method is still stable numerically in several cases. This is justified in the Appendix. Hence the existence of irregular grid cells at the boundary has no impact on the allowable time step.

Modifying κ in small triangular cells gives an algorithm which is stable with a reasonable time step Δt. What is the physical effect of modifying κ in this way? First note that the method is still fully conservative in the sense that $\sum \kappa_{ij} Q_{ij}^n$ is conserved from one time step to the next except for fluxes through the boundary of the computational or physical domain. Hence the total mass of a tracer flowing through the region is conserved. All we have done is to increase the capacity of some cells to hold the tracer. Again we stress that we do not actually need to change the geometry or modify the Darcy velocities at the cell edges. Most importantly we do not need to modify the geometry of other nearby cells.

5. FRACTIONAL STEP METHODS FOR THE ADVECTION-DIFFUSION EQUATION

In the previous two sections, we have described in detail our algorithms for handling the advection and diffusion terms separately. These can be combined using standard fractional step methods to solve the coupled advection-diffusion equation. The advantage of using a fractional step approach is that we can still essentially treat the advection and diffusion operators separately. We only need to justify that the time accuracy of the solution obtained is acceptable.

In the fractional step approach, we first advect the tracer quantity Q^n to obtain an intermediate value Q^*. We then diffuse Q^* to obtain our approximate solution Q^{n+1}. The semi-discrete equations that we solve have the basic form

$$\frac{Q^* - Q^n}{\Delta t} = -(\mathbf{u} \cdot \nabla) Q^n \tag{36}$$

$$\frac{Q^{n+1} - Q^*}{\Delta t} = D\nabla^2 \left(\frac{Q^* + Q^{n+1}}{2} \right), \tag{37}$$

where the right hand side of each discrete equation is discretized as described in the previous sections. This is sometimes called the *Godunov* splitting, and while it is formally only first order accurate, we have found in practice that we can obtain much better than first order accuracy. To see why this might be so, we compare the Godunov splitting to the *Strang* splitting method, a method which is formally second order accurate. We write down the Strang splitting, using the notation from above, as

$$\frac{Q^* - Q^n}{\Delta t/2} = -(\mathbf{u} \cdot \nabla) Q^n \tag{38}$$

$$\frac{Q^{**} - Q^*}{\Delta t} = D\nabla^2 \left(\frac{Q^* + Q^{**}}{2} \right) \tag{39}$$

$$\frac{Q^{n+1} - Q^{**}}{\Delta t/2} = -(\mathbf{u} \cdot \nabla) Q^{**}. \tag{40}$$

In the Strang splitting, the half time step of advection is followed immeditely (in the next time step) by another half time step using the same operator. These two half steps can be combined into a single step of length Δt and in fact it is better to do so to reduce numerical diffusion and computational cost. Once this is done, the Strang splitting over many time steps is identical to the Godunov splitting except in the first and last time step, where a half step of advection is used rather than taking a full step only at the beginning. This change is typically negligible relative to other errors. Even though this error is formally first order, it has a very small constant relative to the overall error. For this reason we use the simpler Godunov splitting, but this could easily be changed in the implementation.

6. NUMERICAL RESULTS

In the following numerical examples, we demonstrate the performance of our proposed algorithm. We seek to demonstrate that using our capacity form differencing, we can obtain results that have the right physical behavior near irregular boundaries and that converge to the correct solution as the computational grid is refined. In the first set of examples, we look at advection and diffusion of a plane wave in a channel and compare the results to the exact solution. In the second set of examples, we test the diffusion and solid body rotation in an annulus and compare our finite volume solution to a reference solution computed in polar coordinates. In the final set of examples, we look at flow through a field of irregular objects and compute washout curves for the advection of a passive tracer through the field.

In all sets of examples, we test the advection algorithm using velocities computed from a stream function $\psi(x, y)$ which has been determined either analytically or numerically. Before the actual computations using CLAWPACK are done, the stream function is computed

at all grid nodes and the values are then differenced to compute values of u and v at all cell edges. These velocities then remain fixed for all time steps of the calculation.

To determine the numerical convergence rate of our solutions obtained in the following test problems, we compare our results to values of a reference solution, obtained either analytically or numerically. We assume that computed values Q_{ij} are located at the center of mass of regular and irregular cells. We then use this reference solution to compute the error e_{ij} in each cell. The weighted norm of this error over the entire computational grid is computed using

$$\|e\|_p = \left(\frac{1}{\text{Area}(\Omega)} \sum_{i,j} |e_{ij}|^p \kappa_{ij} \Delta x \Delta y \right)^{1/p} \tag{41}$$

$$= \left(\frac{\Delta x \Delta y}{\Delta x \Delta y \sum_{i,j} \kappa_{i,j}} \sum_{i,j} |e_{ij}|^p \kappa_{ij} \right)^{1/p} \tag{42}$$

$$= \left(\frac{1}{\sum_{i,j} \kappa_{ij}} \sum_{i,j} |e_{ij}|^p \kappa_{ij} \right)^{1/p}, \qquad 1 \le p < \infty. \tag{43}$$

The ∞-norm is defined in the standard way as

$$\|e\|_\infty = \max_{i,j} |e_{ij}|. \tag{44}$$

In all of the following mesh and contour plots, we only plot solution values in cells which are either completely or partially in the flow domain. Cells that lie completely in the no-flow domain are masked in all plots (using the NaN value in MATLAB).

6.1. *Advection and Diffusion of a Plane Wave in a Channel*

For this set of examples, we look at the basic behavior of our solution in the presence of a planar boundary which is not aligned with the computational grid. The physical domain is a channel with parallel sides, as shown in Fig. 8. The channel is oriented at an arbitrary

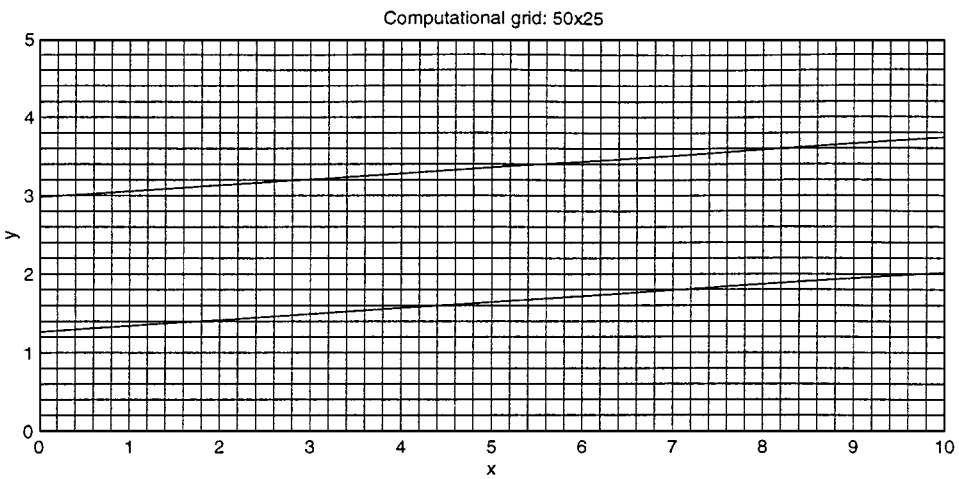

FIG. 8. Coarse grid (50×25) used for flow in the channel. The channel makes an angle of ≈ 0.0754 radians with bottom of computational domain and is ≈ 1.71 units wide.

angle (approximately 0.0754 radians) to the grid and the flow is advected at speed U_0, with $0 \le U_0 \le 1$ in a direction parallel to the walls. We impose no-normal-flow boundary conditions on the channel walls.

The computational grid used here is $N \times M$, where N is the number of cells in the horizontal direction and M is the number of cells in the vertical direction. For the channel problem, we have chosen $N = 2M$. Convergence rates are reported in terms of N.

The basic plane wave shape that we use to initialize the flow is given by

$$w(\xi) = 0.5\left\{\text{erf}\left(\frac{0.75-\xi}{\sqrt{4D}}\right) + \text{erf}\left(\frac{0.75+\xi}{\sqrt{4D}}\right)\right\}, \tag{45}$$

where $D = 0.01$ is the diffusion coefficient and ξ is the streamwise distance from the center of the channel. For problems involving advection, we initialize our flow with this wave shifted towards the inflow end of the channel. For a given velocity U_0, we have that the solution for the pure advection equation is

$$q_{adv}(\xi, t) \equiv w(\xi - U_0(t-3)). \tag{46}$$

The pure diffusion solution is

$$q_{diff}(\xi, t) = 0.5\left\{\text{erf}\left(\frac{0.75-\xi}{\sqrt{4D(1+t)}}\right) + \text{erf}\left(\frac{0.75+\xi}{\sqrt{4D(1+t)}}\right)\right\}. \tag{47}$$

The general advection-diffusion solution is

$$q_{adv\text{-}diff}(\xi, t) \equiv q_{diff}(\xi - U_0(t-3), t). \tag{48}$$

In Fig. 9 we plot the function $q_{adv\text{-}diff}(\xi, t)$ at four different times for $U_0 = 1$ and $D = 0.01$.

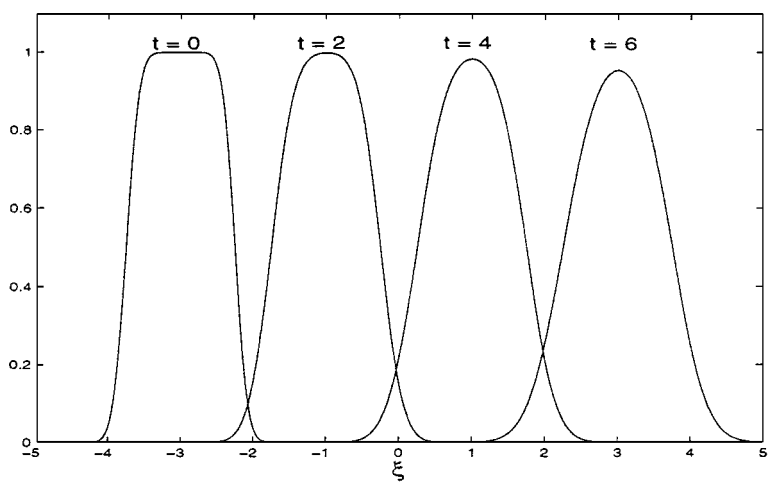

FIG. 9. Plot of exact solution $q_{adv\text{-}diff}(\xi, t)$ to plane wave advection-diffusion in channel. The velocity is set to $U_0 = 1$ and the diffusion coefficient is set to $D = 0.01$. The Peclet number is 100.

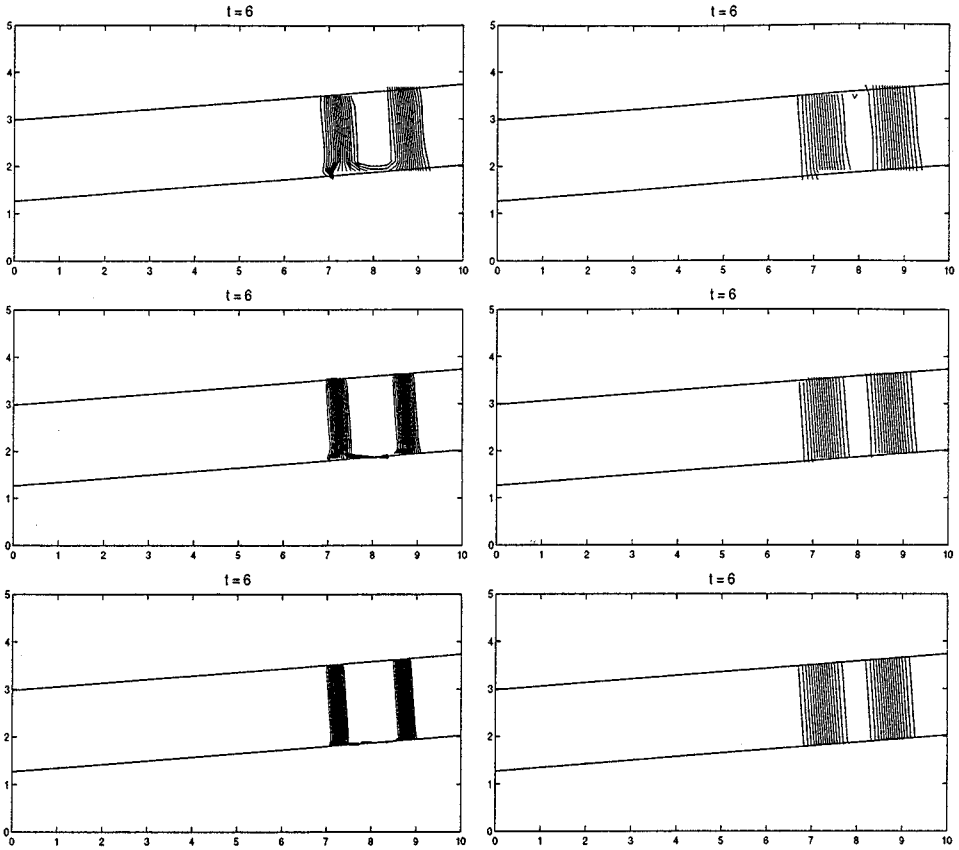

FIG. 10. Advection and diffusion of plane wave in the channel, computed on three different grids. The plots on the left show the results of pure advection and the plots on the right show the results of advection-diffusion. All plots are shown at time $t = 6$. From top to bottom the grid sizes are 50×25, 100×50, and 200×100.

6.1.1. Advection of a plane wave in the channel. To see how the advection portion of the algorithm behaves as the grid is refined, we computed the solution on three different grids. In Fig. 10, we show the results of this series of computations. As expected, the contours in each plot in Fig. 10 remain essentially normal to the wall.

The glitches in the plots are typically caused by inaccurate values in one or two very small cells cut by the boundary. To some extent this nonsmooth behavior is due to the use of MATLAB to plot contour lines. MATLAB takes the data as being at uniformly spaced points, which is really not correct in the small cells, where it is more properly viewed as located at centroids. Another reason may lie in the fact that small cells at the upper edge increase in size in the direction of the flow, whereas at the lower edge they decrease in size. We have not investigated this anomalous behavior in any detail, but do wish to point out that our scheme can produce slightly different results, depending on the orientation of the grid to the boundary, and on the order in which small cells appear as the flow passes through them.

While the algorithm is not perfect in these cells, it is notable that:

- The method remains stable and the incorrect values typically remain within the range of the nearby states.

• Such values only occur when a sharp front is passing by. Once it has passed the values rapidly settle down to the correct post-front state.

• These inaccuracies in the tiny cut cells typically have no impact on the solution in full cells nearby. (This is tested in a later example; see Fig. 17.)

• As observed below, when this advection algorithm is coupled with our diffusion algorithm for advection-diffusion equation, this anomolous behavior almost completely disappears as the implicit diffusion operator smooths out these spikes in the data.

Is the capacity function really necessary? As a final test of the advection in a channel, we demonstrate that by using "capacity form differencing" as described in the previous section, we can essentially eliminate smearing that would occur near the boundary if the numerical scheme did not take into account areas of small cells. For this test, we compute two solutions, one using the capacity function, modified as necessary to ensure stability, and the second without using the capacity function (i.e., setting $\kappa \equiv 1$ in all cells). In Fig. 11, we see that when the capacity function is not used, the results are smeared out considerably. Using the Darcy velocities at the cell interfaces gives a stable method, but the velocity is smaller at edges of the cut cells than elsewhere leading to slower advection along the walls. The smaller velocity is offset by the small area of the cells when our capacity algorithm is used.

6.1.2. Diffusion in the channel. To test the diffusion portion of the code, we looked at the diffusion of the initial pulse given in (45) and compare the results of our algorithm to the exact solution given in (47). While we don't show any plots of the diffusion alone, we can see in Fig. 10 that when a diffusion term is present, there is no noticable difference between the solution in small cells and that in neighboring full cells. In fact, the diffusion term tends to smooth out any spikes in the data produced by the advection algorithm. Figure 8 shows second order convergence in both the 1-norms and max-norms for the pure diffusion algorithm, and that the errors decrease quite smoothly as we refine the grid. As we note in our discussion of the convergence results in Subsection 6.1.4, this is not necessarily true when we include advection.

Performance of BiCGSTAB in the finite volume diffusion algorithm. To solve the non-symmetric linear system of equations arising from the implicit discretization of the diffusion equation, we used the linear iterative method BiCGSTAB. This iterative method has several

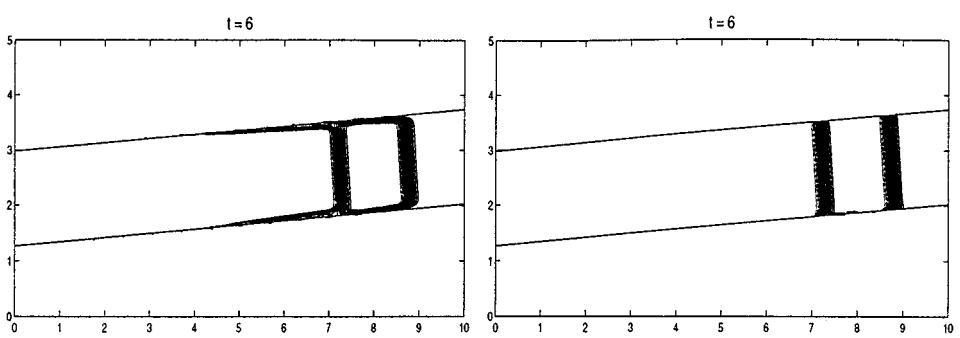

FIG. 11. The solution in the first plot was computed without using a capacity function (i.e., $\kappa_{ij} \equiv 1$). In the second plot, the capacity function was used to take into account areas of small cells. The computational grid is 200×100.

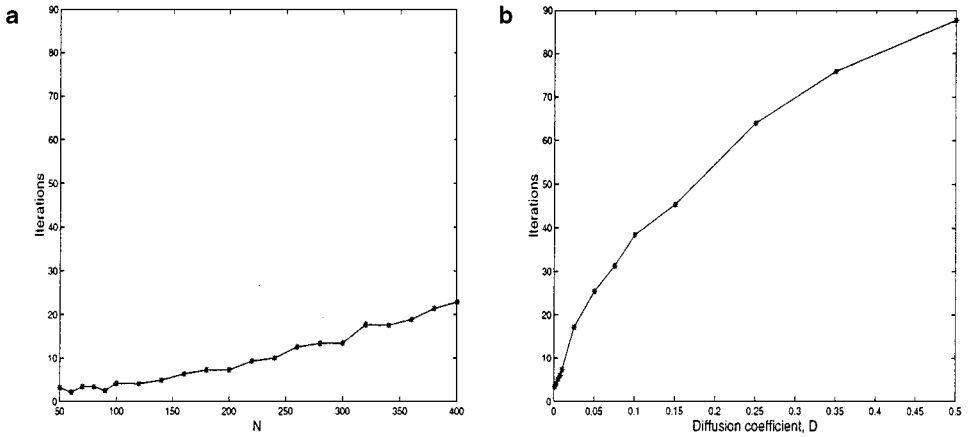

FIG. 12. Performance of BiCGSTAB, as measured by the average number of iterations required to advance the solution from t to $t + \Delta t$, for $\Delta t = 0.025$. (a) Number of iterations versus grid size, for fixed diffusion coefficient, $D = 0.01$. (b) Number of iterations versus diffusion coefficient, for fixed grid of size 200×100. In both plots, the time step was fixed at $\Delta t = 0.025$.

advantages over other methods for non-symmetric systems, and we have found its performance to be quite acceptable. Unlike GMRES, BiCGSTAB does not require additional storage for orthogonal vectors, and unlike the original BiCG, does not require a matrix *transpose* vector multiply. Furthermore, BiCGSTAB is straightforward to implement. One very clear presentation of the algorithm is given in [17].

In Fig. 12, we show how BiCGSTAB performs as a function of grid size and of diffusion coefficent. In both cases, we measure the performance by looking at number of iterations BiCGSTAB needs to reduce the residual of the linear system to the desired level (10^{-8}, in this test case). In the first test case, we look at how varying the size of the linear system affects the number of iterations BiCGSTAB requires. In this test, we varied the grid size and held the diffusion coefficient constant at $D = 0.01$. We then determined the average number of iterations required for BiCGSTAB to advance the solution from t to $t + \Delta t$, for $\Delta t = 0.025$. This average was computed over the 20 time steps needed to advance the solution to $t = 0.5$.

For the second test case, we held the number of grid points (and hence the order of the linear system) fixed at $N = 200$ and varied the diffusion coefficient between 10^{-3} and 0.5. Again, we held the time step fixed at $\Delta t = 0.025$ and computed the average number of iterations over the first 20 times steps. In both test cases, we solved only the diffusion problem.

What is interesting to note from the two plots is that increasing the diffusion coefficient has a much greater impact on the number of iterations BiCGSTAB requires than does increasing the grid size. A grid of 400×200 at $D = 0.01$ required about 7 iterations, whereas a grid of half that size for a diffusion coefficient of $D = 0.05$ required 25 iterations. One explanation for this may lie in the fact that the underlying matrix that BiCGSTAB is inverting differs from a symmetric matrix in only $O(N)$ rows. As the matrix increases in size, the ratio of the number of rows perturbing the symmetry of the matrix versus the total number of rows behaves like $O(N^{-1})$. As the grid size is increased, for fixed diffusion coefficient, the matrix behaves more like a symmetric matrix.

On the other hand, increasing the diffusion coefficient affects the diagonal dominance of the underlying matrix, since at least away from the interface, the operator we are inverting is

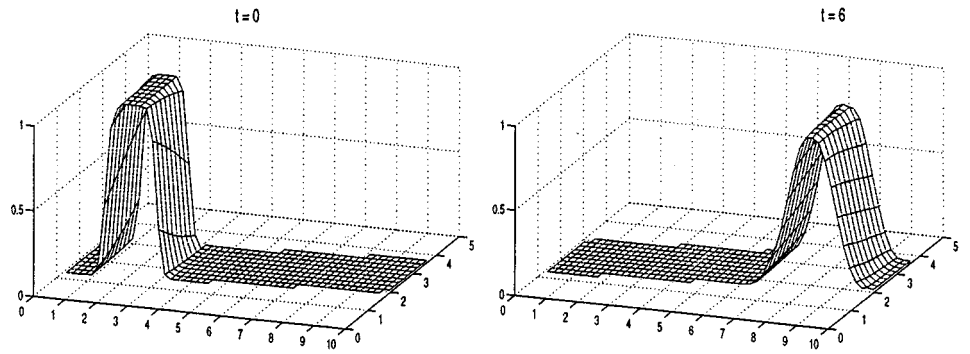

FIG. 13. Advection-diffusion of a plane wave down the channel. Mesh plots of the initial data and solution at time $t = 6$ on the coarse grid, 50×25.

$(1 + \Delta t D \nabla^2)$. As we increase D, the matrix looks more like the discrete Laplacian, which requires many more iterations to invert than its parabolic counterpart.

These results are characteristic of the behavior of BiCGSTAB for all problem geometries investigated, and so are not reported for other test problems.

6.1.3. Advection and diffusion in the channel. Finally, we combine the advection and diffusion schemes using the fractional step approach described in Section 5. Contour plots showing the results for this test problem are shown on the right side of Fig. 10. Note that the contour lines are now perpendicular to both sides of the channel. The implicit diffusion algorithm smooths out the incorrect values in very small cells and corrects the anomalous behavior seen at the boundary in the pure advection case. Figure 13 shows a mesh plot of the results on the coarsest grid.

6.1.4. Convergence results for the channel for different Peclet numbers. In Fig. 14, we show the convergence rate for the solution of the advection-diffusion equation in the channel for different values of the non-dimensional parameter $Pe = U_0 l / D$, known as the Peclet number. Here, l is the length scale of the problem, and for our purposes, is taken to be exactly 1. For $0 < Pe < \infty$, we varied Pe by varying the velocity and holding the diffusion coefficient fixed at $D = 0.01$. For $Pe = 0$, we carry out only the diffusion step of the fractional step scheme, using $D = 0.01$ and for $Pe = \infty$, we carry out only the advection step of the splitting scheme, using $U_0 = 1$.

For a given Peclet number, we compute a single convergence rate r for our algorithm by assuming that the truncation error in our discretization has the form $\epsilon(h) = Ch^r$ for some constant C. By computing the error on several grids, we can then approxmimate the convergence rate by fitting a line through the log of the computed errors.

To compute the errors, we use as a reference solution the exact solution given in (48). The errors were all computed at time $t = 6$, roughly the time it takes for the wave moving at the largest velocity we tested ($U_0 = 1$) to traverse the length of the channel. For problems computed at slower velocities, the wave did not travel as far, but in all cases which involved diffusion ($Pe < \infty$), the total effect of diffusion was the same. The errors, computed using the 1-norm and ∞-norm, along with the rates of convergence are shown in the log–log plots in Fig. 8.

We use so many different grid spacings because the exact manner in which the geometry intersects the Cartesian grid naturally has an impact on the accuracy on any particular grid. As a consequence, the error may not be smoothly varying as the grid spacing is

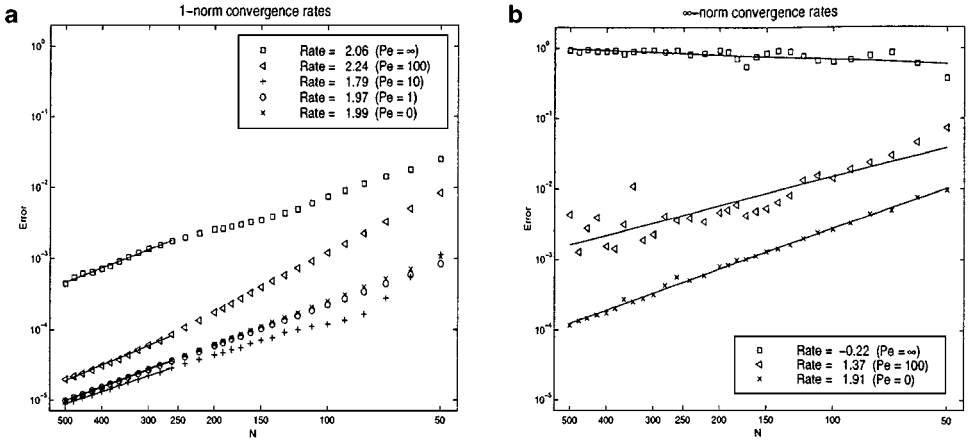

FIG. 14. (a) The 1-norm convergence rates and (b) ∞-norm convergence rates for advection-diffusion in the channel, for varying Peclet numbers. Solid line is the best-fit line used to compute convergence rate for each case.

decreased. From the plots, though, it is clear that the error (at least in the 1-norm) fluctuates about lines with slopes indicating second-order accuracy. The 1-norm errors are dominanted by errors in the full cells, and so do in fact vary smoothly, The max-norm errors, however, are dominated by the errors in small cells and so show less smooth behavior (Fig. 14).

Because the diffusion tends to smooth out any errors produced in small cells by the advection scheme, we expect that the lower the Peclet number (and, hence, the more dominant the diffusion term), the smaller the magnitude of the computed errors. We see in Fig. 8 that this is in fact that case. The largest errors produced by the pure advection ($Pe = \infty$) case, and the smallest errors are produced by the pure diffusion case. In the max-norm, we see that the pure advection may not even converge, but with even just a small amount of diffusion, we get convergence rates that approach second order as we decrease the Peclet number. In the second plot of Fig. 8, we have plotted max-norm errors for 3 for the 5 Peclet numbers ($Pe = \infty$, $Pe = 100$, and $Pe = 0$). The convergence rates for the remaining two cases were 0.90 ($Pe = 10$) and 1.47 ($Pe = 1$). The magnitude of the errors was very close to those cases plotted and so to avoid cluttering the graph, these errors are not shown.

6.2. *Advection and Diffusion in an Annulus*

For this set of problems, we look at diffusion and solid body rotation inside of an annulus composed of two concentric circles embedded in a square computational grid. The goal here is to test how well our advection-diffusion scheme works in more complicated geometry. The annulus is a convenienent choice because we can compare our finite volume solution to a reference solution computed in polar coordinates.

Figure 15 illustrates, on a 30×30 grid, the geometry used for this test case. The radius of the inner circle of the annulus is $R_1 = 0.25$. The outer radius is $R_2 = 1.25$. The center of each circle is at $\approx$(1.509, 1.521). The initial profile of our solution is a function of θ only and is given in terms of the wave profile similar to that used for the channel. This wave profile is

$$w(\theta) = 0.5\left\{\text{erf}\left(\frac{\pi/6 - \theta}{\sqrt{4D}}\right) + \text{erf}\left(\frac{\pi/6 + \theta}{\sqrt{4D}}\right)\right\}. \tag{49}$$

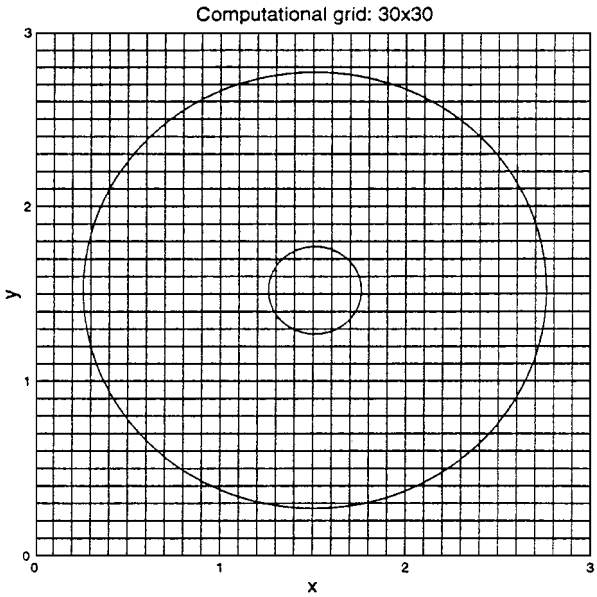

FIG. 15. Geometry for annulus test problem solver. The inner radius of the annulus is $R_1 = 0.25$. The outer radius is $R_2 = 1.25$ and the center is at $\approx(1.509, 1.521)$.

In the interior of the annulus, the solution was initialized using

$$q_0(\theta) \equiv w(\theta - \pi/2). \tag{50}$$

We imposed no-flow boundary conditions at the two boundaries of the annulus.

6.2.1. Solid body rotation in an annulus. To test our proposed numerical scheme for the advection solver, we considered advection under solid body rotation. The velocities for solid body rotation can be determined from the stream function

$$\psi(x, y) = \begin{cases} 0, & r > R_2 \\ 0.2\pi\left(R_2^2 - r^2\right), & R_1 \le r \le R_2 \\ 0.2\pi\left(R_2^2 - R_1^2\right), & r < R_1, \end{cases} \tag{51}$$

where $r = ((x - 1.509)^2 + (y - 1.521)^2)^{1/2}$ is the distance of a particular point (x, y) to the center of the annulus. For this particular choice of ψ, the flow makes one complete revolution in 5 time units.

Figure 16 shows contour plots of the computed solution as it rotates around the annulus. The solution is shown at a sequence of times on two different grids. As in the channel flow case, the flow remains essentially parallel to both boundaries of the annulus, although there is evidence of numerical diffusion in the coarse grid solution.

As a second validation of the solid body rotation, we looked at the computed solution as a function of θ at different values of r, after one and two full rotations, and compared these values with the initial conditions. These comparisons are presented in Fig. 17 for a 60×60 grid. In Fig. 17a, we plot the value of the solution in the partial cells cut by the outer boundary of the annulus versus θ. The solution stays bounded and stable in these cells, but some inaccuracy in the smallest cells is observed as the front propagates

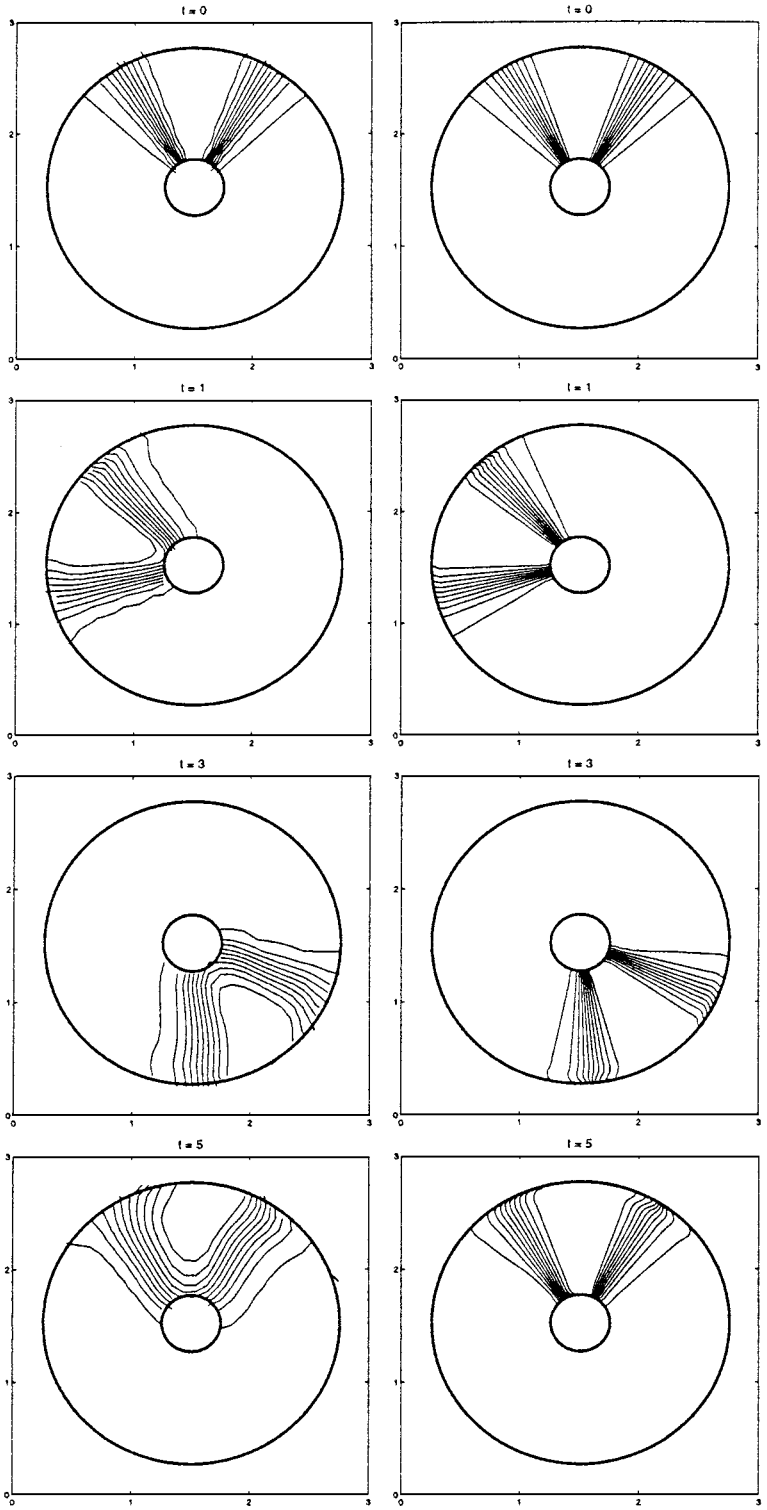

FIG. 16. Contour plots on the coarse (30×30) and fine (150×150) grid showing results of solid body rotation at various times during one full rotation. Contour levels shown are $(0.1, 0.2, \ldots, 0.9)$.

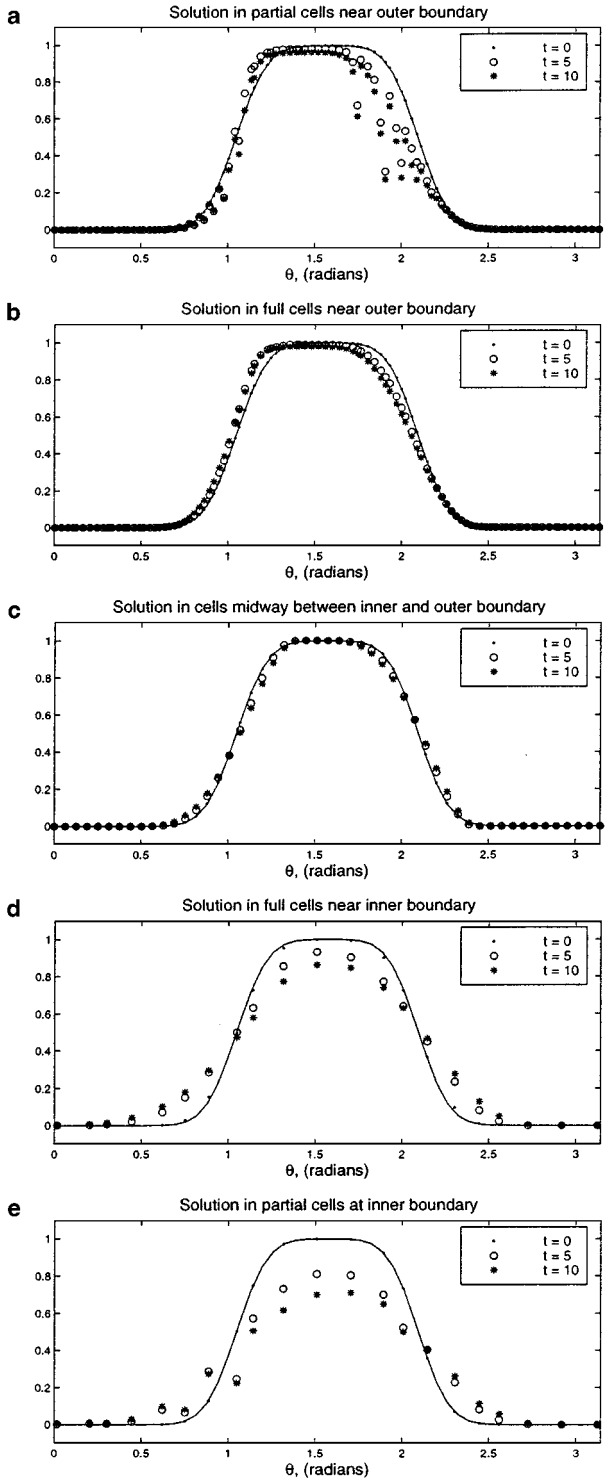

FIG. 17. Computed solution to the pure advection equation in the annulus, shown as a function of θ for five different values of r. The numerical solution was computed on a 60×60 grid and is shown at two different times $t = 5$ (after one revolution) and $t = 10$ (after two revolutions). The true solution is shown as a solid line: (a) at $r = R_2$, along the outer boundary; (b) at $r = R_2^-$, just inside the outer boundary; (c) at $r = (R_1 + R_2)/2$, the middle of the annulus; (d) at $r = R_1^+$, just inside the inner boundary; (e) at $r = R_1$, along the inner boundary.

past. However, this loss of accuracy in the cut cells does not appear to affect the solution away from the boundary. In Fig. 17b, we show the solution a small distance inside the boundary, as interpolated from the full cell values to points (R_2^-, θ_i), where $R_2^- = R_2 - \sqrt{2}\Delta x$ is a radial value slightly inside of the outer edge of the annulus, and the θ_i values are equally spaced values of θ in $[-\pi, \pi]$. Here the solution agrees quite well with the expected behavior.

Figure 17c shows the solution near the middle of the annulus, at $r = (R_1 + R_2)/2$. Figure 17d shows the solution just inside the inner boundary, at $R_1^+ = R_1 + \sqrt{2}\Delta x$, again interpolated from the full grid cells. Finally, Fig. 17e shows the values in the partial cells cut by the inner boundary. Note that near the inner boundary the solution is poorly resolved because there are very few grid cells around this circle (see Fig. 15). In all cases there is very little degradation of the solution between the first and second revolution.

6.2.2. Diffusion in an annulus. To test diffusion in the annulus, we allowed the initial data to diffuse over time (with no advection). The results we obtained agreed well with a reference solution computed by solving the diffusion equation in polar coordinates:

$$\frac{\partial q}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial q}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2 q}{\partial \theta^2}, \qquad R_1 \le r \le R_2, \quad -\pi \le \theta \le \pi. \tag{52}$$

We obtained a spectrally accurate solution to (52) using a Fourier expansion in the azimuthal direction and a pseudospectral method based on Chebyshev polynomials in the radial direction. We used 256 points in θ and 32 points in the r-direction (where the solution varies more smoothly). This gives a highly accurate reference solution, which was then interpolated to the cartesian grid using Fourier and Chebyshev interpolation.

Figure 18 shows contour plots comparing the computed and reference solution at four different times. Even near the boundaries of the annulus it is evident that the no-flux boundary condition is satisfied. To show how our computed solution behaves as the grid is refined, a fine grid solution and corresponding reference solution are also plotted in Fig. 18.

6.2.3. Solid body rotation coupled with diffusion in an annulus. Contour plots showing the results of combining the diffusion and advection in an annulus are shown in Fig. 19. Even the coarse grid solution agrees reasonably well with the reference solution. In Fig. 20 we show a mesh plot of the same set of results at two different times.

6.2.4. Convergence results for the annulus for different Peclet numbers. In Fig. 8, we show 1-norm and max-norm convergence results for the annulus test problem. As we did for the channel, we investigated how the convergence rates varied with the Peclet number. For this test, however, we held the velocity fixed (except in the $Pe = 0$ case, where there is no rotation), and varied the diffusion coefficient. The Peclet numbers we tested were $Pe = 0, 20, 100, 500, \infty$. For $0 < Pe < \infty$, we computed the diffusion coefficient for a given Peclet number as $D = U_0 l/Pe$, where for this problem, $l = 1$ and U_0 was taken to be the tangential velocity on the outer boundary of the annulus. For the stream-function we are using, this velocity is given by $U_0 = 2\pi R_2/5 = 0.5\pi$. For $Pe = 0$, the diffusion coefficient was set to $D = 0.01$. For $Pe = \infty$, we use a diffusion coefficient of $D = 0.01$ only to initialize the flow, using (50).

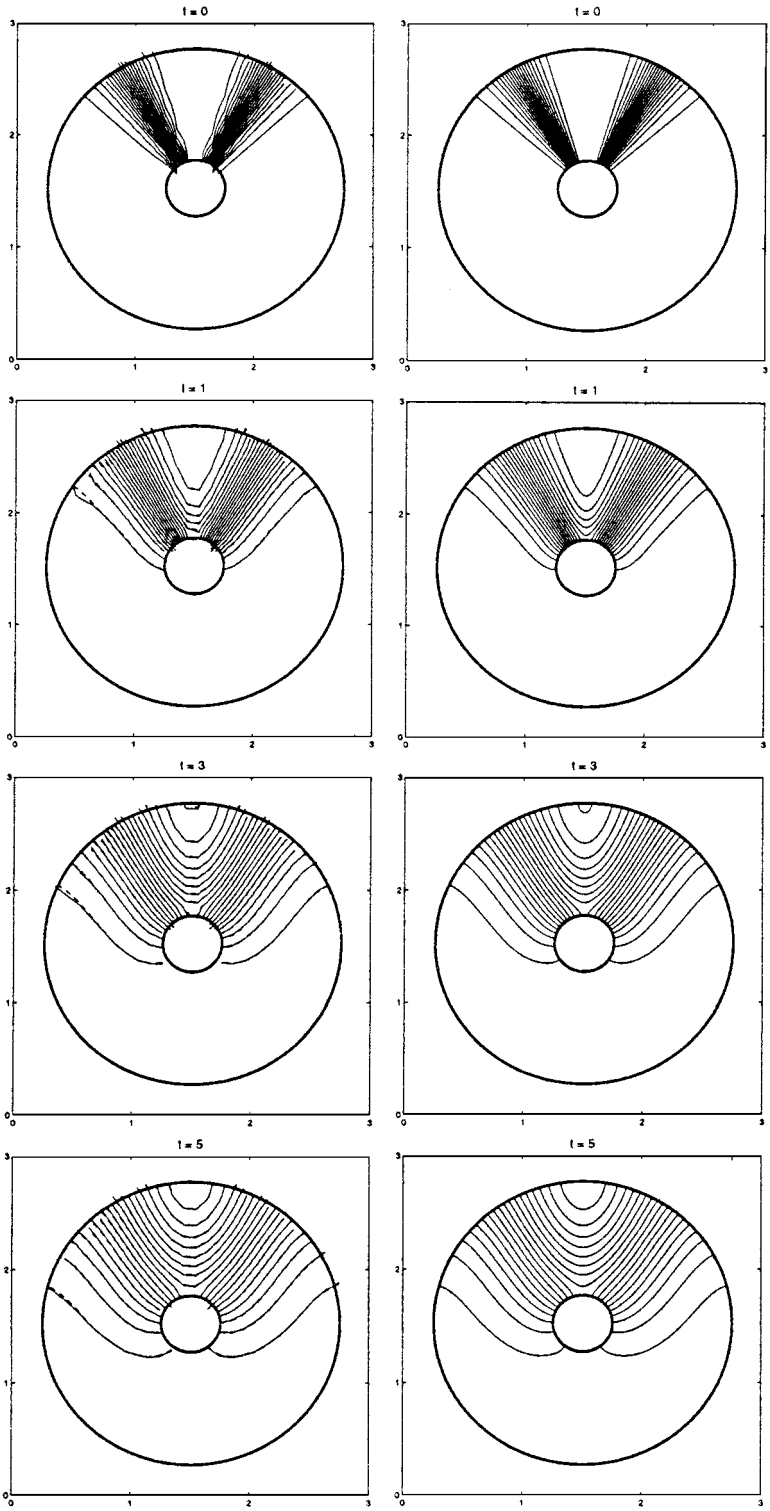

FIG. 18. Contour plots on the coarse (30×30) and fine (150×150) grid showing results of diffusion in the annulus at various times. The solid lines represent results of finite volume scheme, and dashed lines are the reference solution, interpolated to appropriate grid. Contour levels shown are (0.01, 0.05, 0.1, 0.15, . . . , 1.0).

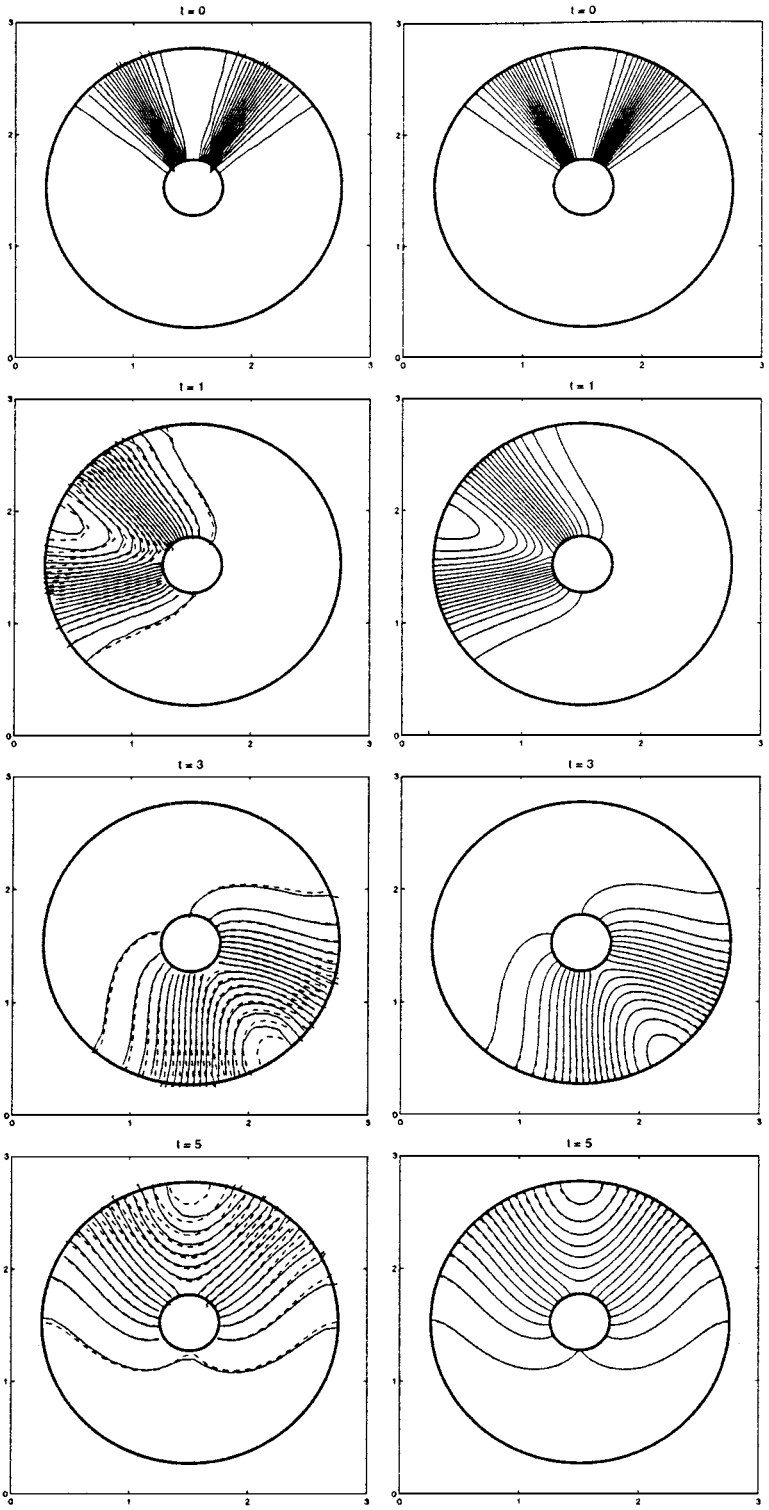

FIG. 19. Contour plots on the coarse (30×30) and fine (150×150) grid showing results of solid body rotation coupled with diffusion at $Pe = 100$ in the annulus at various times over one full rotation. The solid lines represent results of finite volume scheme, and dashed lines are the reference solution, interpolated to the Cartesian grid. Contour levels show are (0.01, 0.05, 01, 0.15, . . . , 1.0).

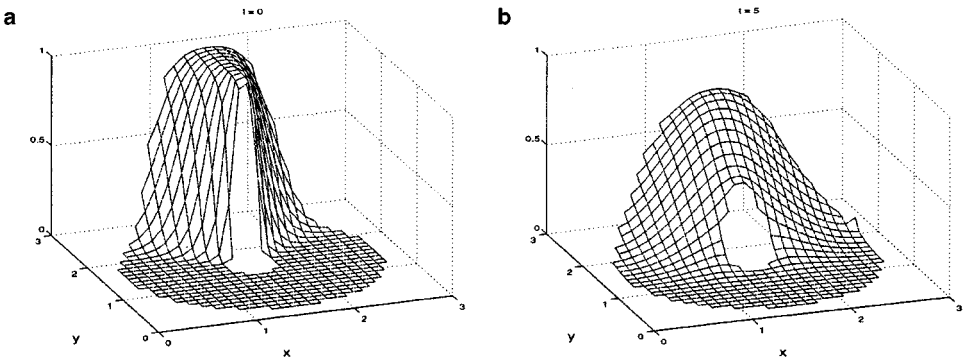

FIG. 20. Mesh plots on coarse grid (30×30) for advection and diffusion in the annulus with $Pe = 100$. (a) Initial data. (b) Computed solution after one complete revolution.

Our choice of Peclet numbers differs from the channel case in that here, the total effect of the advection was kept fixed over the range of Peclet numbers chosen, whereas before, we held the effects of diffusion fixed. All errors were computed at $t = 5$, after the initial wave had made one complete revolution in the cases with advection. The errors vary more smoothly with decreasing grid size in this case than was observed in the channel. This may be because the annulus cuts through cells in a wider variety of random ways regardless of where it is located, while the channel leads to a periodically repeating pattern of cut cells, and hence may be more sensitive to its location, or the grid spacing. As seen in Fig. 21, the errors for the annulus all lie very close to the best-fit line for both the 1-norm and the max-norm.

The rates of convergence for this test problem were not as close to second order as they were in the channel case, but we do see that the rates generally improve as the Peclet number decreases (Fig. 21). We consider this to be a challenging test case because of the poor resolution of the rapidly varying solution near the inner boundary, as seen in Fig. 17 e, for example. Even on the finest grids there are relatively few grid points around the inner boundary.

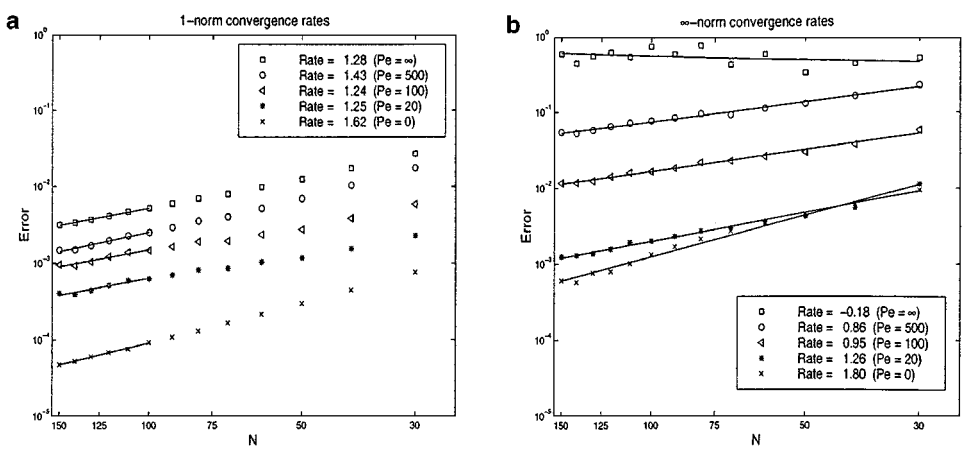

FIG. 21. (a) The 1-norm convergence rates and (b) ∞-norm convergence rates for advection-diffusion in the annulus, for varying Peclet numbers. Solid line is the best-fit line used to compute convergence rate for each case.

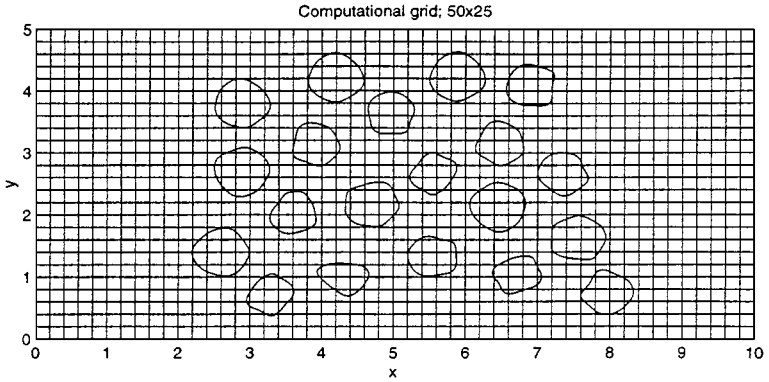

FIG. 22. Coarse grid (50×25) with embedded objects.

6.3. *Advection through a Field of Irregular Objects*

In this set of examples, we constructed a field of irregularly shaped inclusions and simulated the advection of a passive tracer through this field. One can imagine that this field represents, for example, rocks in a porous medium or the cross section of pipes in a heat exchanger. The purpose of this set of examples is threefold. First, we illustrate that our reliance on the stream-function for obtaining our velocity field does not limit us to objects for which we know a stream-function analytically. Second, we show that our proposed advection-diffusion algorthim is not particularly sensitive to geometry chosen. Third, we demonstrate that our capacity-form differencing with the modified capacity function is effective at reducing smearing even when the stream-function is only known approximately. Figure 22 shows the field of irregular objects, embedded in the coarsest grid used for this example.

To compute the stream-function in the multiply connected domain, we solved an elliptic equation in the domain using the Immersed Interface Method [23] and a procedure similar to that developed by Yang [35] for this problem. This Cartesian grid method produces second-order accurate values of the stream-function at all grid points, which were chosen to be the corners of our finite volume cells. Differencing these values gives the required Darcy velocities at cell edges. Figure 23 shows a contour plot of the stream-function used for this set of test problems, computed on a very fine grid.

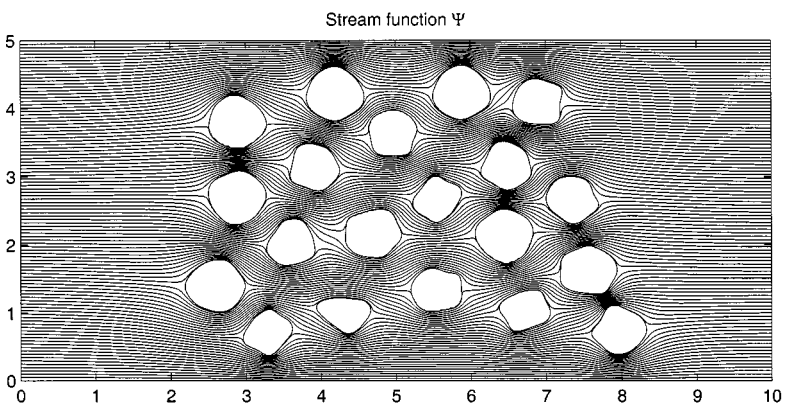

FIG. 23. Stream function used for flow through irregular objects.

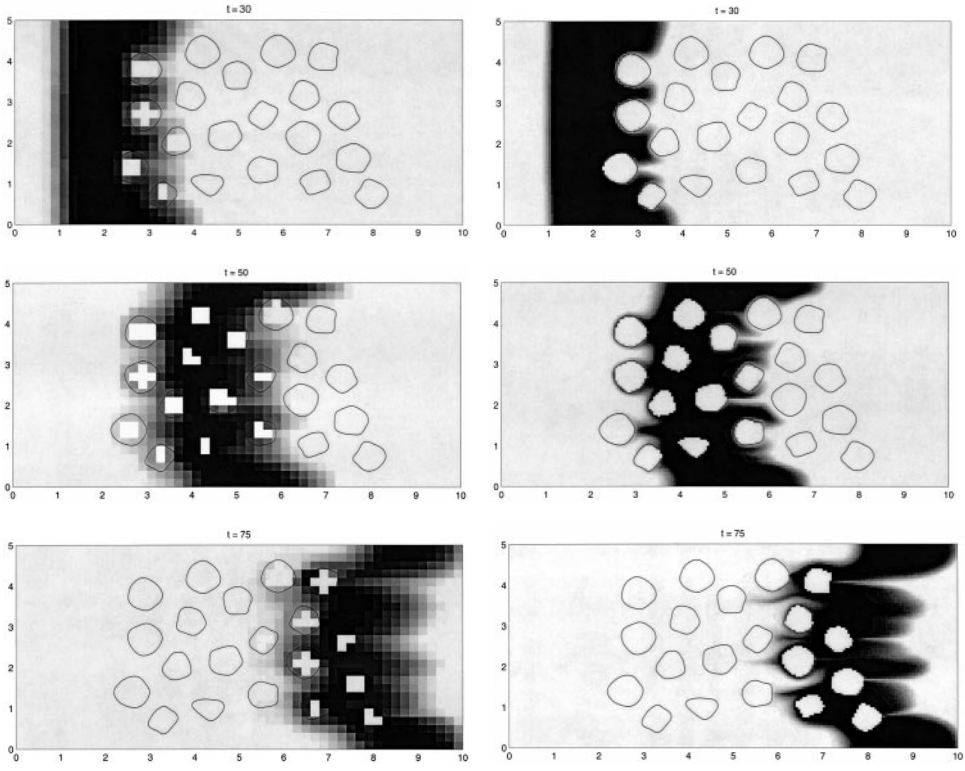

FIG. 24. Advection of passive tracer through field of irregularly shaped objects. Coarse grid (left side) is 50×25 and fine grid (right side) is 200×100.

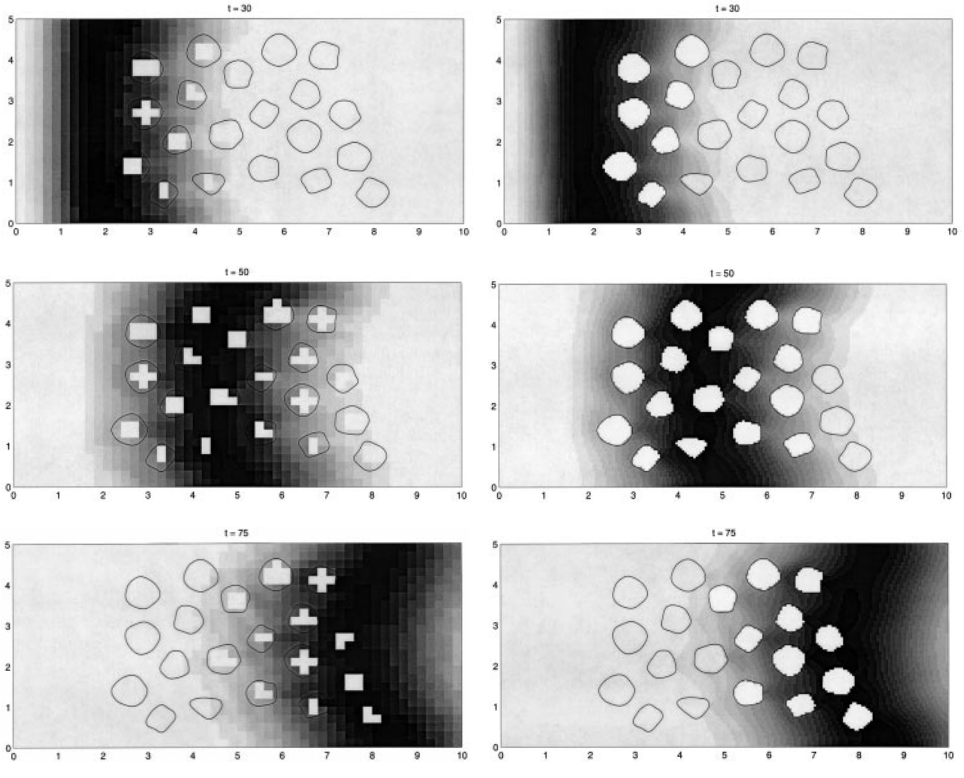

FIG. 25. Advection and diffusion of passive tracer through field of irregularly shaped objects. Coarse grid (left side) is 50×25 and fine grid (right side) is 200×100.

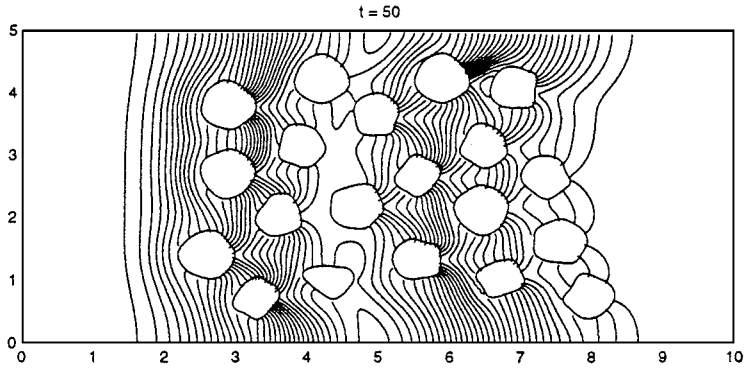

FIG. 26. Contour plot of advection-diffusion results through field of irregular objects for the fine grid calculation at time $t = 50$. Note that the contour lines are roughly perpendicular to the boundary of each object.

We use initial data $q(x, y, 0) \equiv 0$ and a boundary condition $q(0, y, t) = 1$ up to time 20 (and zero thereafter), which corresponds to the injection of a tracer at the left (inflow) boundary over this time interval. Figure 24 shows plots of the solution at different times.

In order to compare the results more quantitatively as the grid is refined, we also computed "washout" curves $w(t)$ which measure the total mass passing through the right edge of the domain as a function of time. This is computed by

$$w(t_n) = \sum_{j=1,N} Q_{M,j} \Delta y \approx \int_0^5 q(10, y, t)\, dy \tag{53}$$

which sums the values of Q along the rightmost row of grid cells on an $M \times N$ grid. This gives the curves shown in Fig. 27. The general behavior of flow through the field of irregular objects is preserved as the grid is refined. Even with the coarsest grid, where the fine-scale structure of the flow is poorly resolved in Fig. 24, the washout curve shows quite good agreement. For many applications, e.g., groundwater flow simulations, this is of great importance.

6.4. *Advection and Diffusion through a Field of Irregular Objects*

In this test, we simulated a tracer advecting and diffusing through the same field of irregular objects. Figure 25 shows the results of this simulation at 3 different times. To see that the normal derivative of q is close to zero near the boundaries, as expected from the Neumann boundary conditions with diffusion, we also show a contour plot of this solution in Fig. 26.

Washout curves are again shown in Fig. 27. When diffusion is added, there is virtually no difference in the washout curves at different grid resolutions. Even the under-resolved grid of Fig. 22 gives excellent results.

7. CONCLUSIONS AND EXTENSIONS

We have demonstrated that our capacity from differencing, coupled with a new approach towards handling the small cell stability problem, can be used to solve the advection-diffusion equation in an irregular physical domain embedded in a Cartesian grid. The key contributions of this paper are summarized in Section 1. Several interesting extensions to this work are also possible, a few of which we describe briefly here.

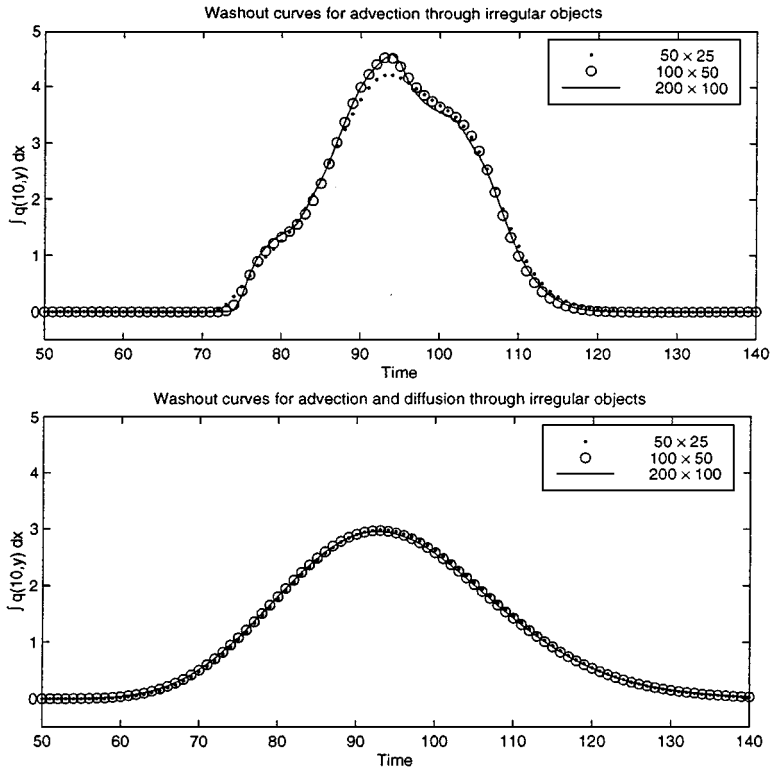

FIG. 27. Washout curves computed on three different grids. The upper plot is for advection only. The lower plot is for the advection-diffusion equation.

The CLAWPACK software has been extended to AMRCLAW, an adaptive mesh refinement version of the code, developed by Berger and LeVeque [5]. This software also uses capacity-form differencing and is freely available on the web [6]. The advection portion of the algorithm presented in this paper, originally implemented using CLAWPACK, carries over directly to AMRCLAW with almost no modification. Figure 28 shows the solution at two times when AMRCLAW is used for the same problem as illustrated in Fig. 24. The finest grid level in this computation would correspond to a uniform 512×256 grid, and hence has much better resolution than the computation on the right in Fig. 24. Computing on a uniform 200×100 grid up to time $t = 75$ requires advancing a total of about 14 million grid cells over the entire computation with the time step we used. Computing on a

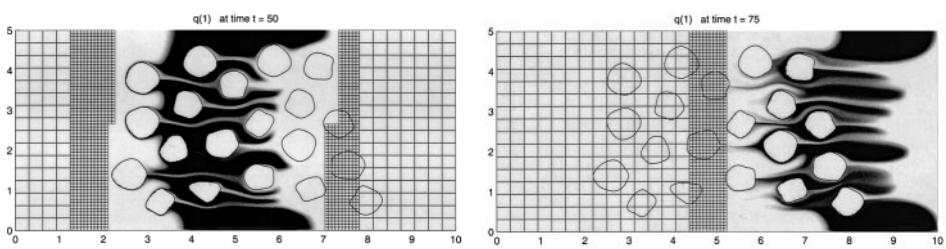

FIG. 28. Advection of passive tracer through field of irregularly shaped objects, using the adaptive mesh refinement algorithm of AMRCLAW. The coarsest Level 1 grid is 32×16. This is refined by a factor of 4 to the Level 2 grids and by an additional factor of 4 to the Level 3 grids. The grid lines are shown only on Levels 1 and 2.

uniform 512×256 grid would require advancing a total of 232 million cells in time. The adaptive code advanced about 71 million cells, plus another 1 million for error estimation. Here the front was quite complex so that a large fraction of the domain was ultimately refined. For other computations the advantages of adaptive refinement can be even more significant.

For the adaptive code, the stream-function was computed first on a 512×256 grid and used at each grid resolution to define the correct Darcy velocity at the edge of all cells. The capacity function was also first determined everywhere on the finest grid. For a cell at a coarser level, the values of κ_{ij} in all fine grid cells covered by the coarse cell are averaged to compute κ in the coarse cell. Except for the stability modification to κ, this simply corresponds to taking the area of the flow-domain within the coarse cell to be the sum of the areas within all underlying fine cells. The stability modification is applied to κ on the fine grid before doing the averaging to coarser levels, and this appears to maintain stability at all levels with no additional corrections.

The diffusion algorithm could also, in principle, be extended to an adaptive refinement code. However, since it is an implicit algorithm this is more complicated and has not yet been implemented.

The extension of our method to three space dimensions should be straightforward, and the three-dimensional version of CLAWPACK is already set up to handle a capacity function. To modify the capacity function to ensure the stability of the advection scheme, we believe that simple analogs of the two-dimensional approach will work in three dimensions. To approximate the diffusive fluxes at the partial *faces* of a three-dimensional grid cell, we expect to be able to extend the bilinear interpolaton directly to three dimensions. However, these extensions have not yet been implemented or tested.

One possible application of this work is to the computation of two-dimensional incompressible flow by solving the stream-function vorticity equations. Our approach can be used to develop an embedded-boundary Cartesian-grid finite-volume method for this problem, by solving an advection-diffusion equation

$$\omega_t + \nabla \cdot (\mathbf{u}\omega) = \nabla \cdot (\nu \nabla \omega),$$

where the vorticity ω is now the conserved quantity and velocities are obtained from a stream-function ψ, as described in this paper. The stream-function must be determined in each time step by solving a Poisson problem with the vorticity appearing on the right hand side. The main new challenge is to determine the correct flux of vorticity at the boundaries in order to impose the no-slip boundary condition. Calhoun [7] demonstrates that for low Reynolds number flows the immersed interface method can be used to determine this flux. This work will also be presented elsewhere in the future.

Advection-diffusion equations in complex geometry also arise in many other contexts in heat transfer or fluid dynamics. The approach outlined here should be useful for a wide variety of practical problems.

APPENDIX: STABILITY OF THE ADVECTION ALGORITHM

We do not have a complete proof of stability of the advection algorithm with modified capacities κ_{ij} in cut cells. However, we can offer some justification and intuition for why the timestep can be chosen based on $\nu \leq 1$ with ν defined by (35). We wish to show that

stability will still be maintained in cut cells such as those in Figs. 7b and 7c. Recall that we are assuming $\kappa \equiv 1$ in the flow domain for simplicity. This stability result rests on three claims, which will be justified but not fully proved:

Claim 1. The general advection algorithm (11) with fluxes determined by the multi-dimensional approach (29) and (30) on any grid is stable provided that

$$\max_{i,j} \nu_{ij} \leq 1, \tag{54}$$

where

$$\nu_{ij} = \max\left(\left| \frac{U_{i\pm 1/2,\, j}\, \Delta t}{\kappa_{ij}\, \Delta x} \right|, \left| \frac{V_{i,\, j\pm 1/2} \Delta t}{\kappa_{ij}\, \Delta y} \right| \right). \tag{55}$$

Claim 2. With the definition (33) for κ_{ij}, the bound $\nu \leq 1$ (using (35)) guarantees that $\nu_{ij} \leq 2$ in all cells.

Claim 3. If $\nu \leq 1$ then $\nu_{ij} \leq 1$ in regular cells and for cut cells stability is still maintained even though (54) may be violated with $1 < \nu_{ij} \leq 2$.

Claim 1 has not been proved in general, though for the problem considered here this is the standard definition of the Courant number that is used in the CLAWPACK code. The code has been found to be stable on a wide variety of problems and is typically used with automatic timestep selection which seeks to keep this Courant number at a value of about 0.9. One could satisfy (54) without the modification of κ by (31) by using a sufficiently small timestep, but this is what we wish to avoid.

Claim 2 is the heart of the matter, as it shows that our modified κ_{ij} give a stable algorithm with Δt chosen to be at worst half the value required on the portion of the domain where the grid is uniform, regardless of the geometry of the small cut cells. Claim 3 allows us to eliminate this factor of $1/2$ as well.

To justify Claim 3, consider the special case shown in Fig. 29 comparing a full cell and a half-size triangular cell when the velocity is at 45 degrees to the grid. Figure 29a illustrates why the advection algorithm with transverse propagation is stable for Courant numbers up to 1, as described more fully in [20]. The waves from the left and bottom interfaces both

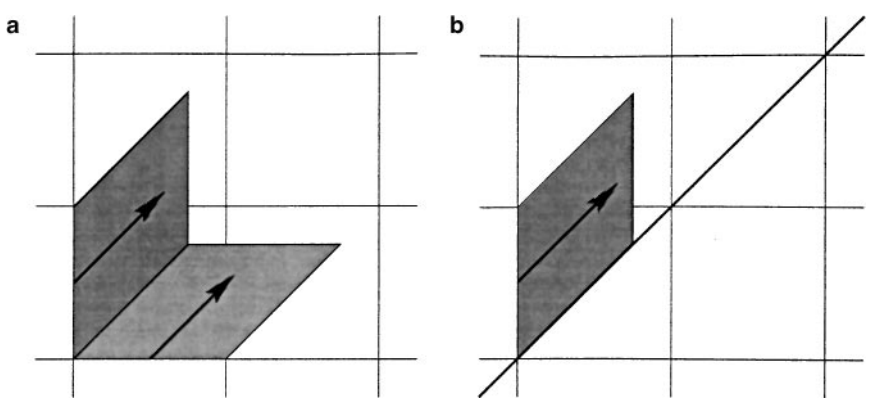

FIG. 29. Advection at 45 degrees to the grid (a) in a full cell; (b) in a half cell cut by the boundary. With transverse propagation the same size timestep can be used in each case.

move in the direction of the flow and only half of each wave affects the cell value, since transverse fluxes transfer the other half to adjacent cells. The Donor Cell upwind method, by contrast, does not include this transverse propagation, the two waves would both affect only cell (i, j), and the Donor Cell method is only stable for Courant number $1/2$ or less in this case.

Figure 29b shows the same situation in a triangular cell cut by the wall. In this case one of the two waves from Fig. 29a is missing, but the algorithm is clearly still stable with the same Δt as in Fig. 29a. With this value of Δt the Courant number will be twice as large, however, since the area of the triangular cell is $\frac{1}{2}\Delta x\,\Delta y$. Hence in this case a Courant number up to 2 is allowed. Similar analysis applies to other cut cells, justifying Claim 3.

Claim 2 will be justified for the two situations shown in Figs. 7b and 7c. Other configurations follow by similar reasoning. In particular, we suppose $U_{i-1/2,j} > 0$, $V_{i,j+1/2} \geq 0$ and write

$$U_{i-1/2,j} = \hat{u} h_{i-1/2,j}/\Delta y, \qquad V_{i,j+1/2} = \hat{v} h_{i,j+1/2}/\Delta x,$$

where $\hat{u} \leq \max|u(x, y)|$ and $\hat{v} \leq \max|v(x, y)|$ are average pointwise velocities along each side. Then we can rewrite (55) as

$$\nu_{ij} = \max\left(\left|\frac{\hat{u} h_{i-1/2,j}\Delta t}{\kappa_{ij}\Delta x\Delta y}\right|, \left|\frac{\hat{v} h_{i,j+1/2}\Delta t}{\kappa_{ij}\Delta x\Delta y}\right|\right). \tag{56}$$

We wish to show that $\nu_{ij} \leq 2$ if $\nu \leq 1$.

First consider the case in Fig. 7b, where the cell is trapezoidal with $h_{i+1/2,j} = \Delta x$, so that the cell area is

$$\kappa_{ij}\Delta x\Delta y = \frac{1}{2}\left(h_{i-1/2,j} + h_{i+1/2,j}\right)\Delta x \geq \frac{1}{2} h_{i-1/2,j}\Delta x.$$

Then we can bound (56) by

$$\nu_{ij} \leq \frac{2\Delta t}{h_{i-1/2,j}\Delta x}\max\left(\left|\hat{u} h_{i-1/2,j}\right|, \left|\hat{v} h_{i,j+1/2}\right|\right). \tag{57}$$

Now assume that u and v are roughly constant in this cell, which is reasonable for smooth flow on a sufficiently fine grid. Then the slope of the wall segment in Fig. 7b is given by $\hat{v}/\hat{u}$, and so

$$\hat{v} h_{i,j+1/2} = \hat{v}\Delta x \leq \hat{u} h_{i-1/2,j}.$$

Hence the first term is the larger in the max of (57) and cancelling the factor of $h_{i-1/2,j}$ gives

$$\nu_{ij} \leq \frac{2\hat{u}\Delta t}{\Delta x} \leq 2\nu \leq 2$$

as desired.

Now consider the triangular case in Fig. 7c. Expression (56) still holds, but now κ_{ij} has been modified using Eq. (31) so that $\kappa_{ij}\Delta x\Delta y$ is larger than the cell area $A_{ij} = \frac{1}{2} h_{i-1/2,j} h_{i,j+1/2}$. (Using the unmodified κ_{ij} could clearly lead to arbitrarily large values of ν_{ij} in (56).) The modification (31) in this case reduces to

$$\kappa_{ij}\Delta x\Delta y = \frac{1}{2}\min\left(h_{i-1/2,j}\Delta x, h_{i,j+1/2}\Delta y\right) \tag{58}$$

since the larger cell side is replaced by the full cell length in the formula for the triangle's area.

If the first term in (56) is larger, then we can use

$$\kappa_{ij}\,\Delta x\,\Delta y \geq \frac{1}{2} h_{i-1/2,\,j}\,\Delta x$$

(which follows from (58)) to obtain

$$\nu_{ij} \leq \left| \frac{2\hat{u}\,\Delta t}{\Delta x} \right| \leq 2\nu \leq 2.$$

Or, if the second term in (56) is larger, then we can use

$$\kappa_{ij}\,\Delta x\,\Delta y \geq \frac{1}{2} h_{i,\,j+1/2}\,\Delta y$$

(which also follows from (58)) to obtain

$$\nu_{ij} \leq \left| \frac{2\hat{u}\,\Delta t}{\Delta y} \right| \leq 2\nu \leq 2.$$

Either way, we have obtained the desired bound.

ACKNOWLEDGMENTS

This work was supported in part by NSF Grants DMS-9505021, DMS-96226645, and DOE Grant DE-FG03-96ER25292.

REFERENCES

1. A. S. Almgren, J. B. Bell, P. Colella, and T. Marthaler, A Cartesian grid projection method for the incompressible Euler equations in complex geometries, *SIAM J. Sci. Comput.* **18**, 1289 (1997).
2. M. Berger and R. J. LeVeque, *An Adaptive Cartesian Mesh Algorithm for the Euler Equations in Arbitrary Geometries*, AIAA paper, AIAA-89-1930, 1989.
3. M. Berger and R. J. LeVeque, Cartesian meshes and adaptive mesh refinement for hyperbolic partial differential equations, in *Proc. Third Int'l Conf. Hyperbolic Problems, Uppsala, 1990.*
4. M. Berger and R. J. LeVeque, Stable boundary conditions for Cartesian grid calculations, *Comput. Systems Eng.* **1**, 305 (1990).
5. M. J. Berger and R. J. LeVeque, AMRCLAW software, available on the Web at the URL `http://www.amath.washington.edu/~rjl/amrclaw/`.
6. M. J. Berger and R. J. LeVeque, Adaptive mesh refinement using wave-propagation algorithms for hyperbolic systems, *SIAM J. Numer. Anal.* **35**, 2298 (1998).
7. D. Calhoun, *A Cartesian Grid Method for Solving the Streamfunction–Vorticity Equations in Irregular Geometries*, Ph.D. thesis, University of Washington, 1999.
8. I. Chern and P. Colella, *A Conservative Front Tracking Method for Hyperbolic Conservation Laws*, Report UCRL-97200, LLNL, 1987.
9. M. S. Day, P. Colella, M. J. Lijewski, C. A. Rendleman, and D. L. Marcus, Embedded boundary alogirthms for solving the Poisson equation on complex domains, preprint, LBNL-41811, Lawrence Berkeley Lab, 1998.
10. D. De Zeeuw and K. Powell, An adaptively-refined Cartesian mesh solver for the Euler equations, *J. Comput. Phys.* **104**, 56 (1993).

11. A. Fogelson and J. Keener, Immersed interface methods for Neumann and related problems in two and three dimensions, preprint, 1997.
12. A. L. Fogelson, A mathematical model and numerical method for studying platelet adhesion and aggregation during blood clotting, *J. Comput. Phys.* **56**, 111 (1984).
13. H. Forrer, *Boundary Treatments for Cartesian-Grid Methods*, Ph.D. thesis, ETH-Zurich, 1997.
14. H. Forrer and R. Jeltsch, A higher-order boundary treatment for Cartesian-grid methods, *J. Comput. Phys.* **140**, 259 (1998).
15. R. Glowinski, T.-S. Pan, and J. Periaux, A fictitious domain method for Dirichlet problem and applications, *Comput. Methods Appl. Mech. Eng.* **111**, 283 (1994).
16. R. Glowinski, T.-S. Pan, and J. Periaux, A fictitious domain method for external incompressible viscous flow modeled by Navier–Stokes equations, *Comput. Methods Appl. Mech. Eng.* **112**, 133 (1994).
17. A. Greenbaum, *Iterative Methods* (SIAM, Philadelphia, 1997).
18. H. Johansen and P. Colella, A Cartesian grid embedded boundary method for Poisson's equation on irregular domains, *J. Comput. Phys.* **147**, 60 (1998).
19. R. J. LeVeque, Cartesian grid methods for flow in irregular regions, in *Num. Meth. Fl. Dyn. III*, edited by K. W. Morton and M. J. Baines (Clarendon, Oxford, 1988), p. 375.
20. R. J. LeVeque, High resolution finite volume methods on arbitrary grids via wave propagation, *J. Comput. Phys.* **78**, 36 (1988).
21. R. J. LeVeque, High resolution conservative algorithms for advection in incompressible flow, *SIAM J. Numer. Anal.* **33**, 627 (1996).
22. R. J. LeVeque, Wave propagation algorithms for multi-dimensional hyperbolic systems, *J. Comput. Phys.* **131**, 327 (1997).
23. R. J. LeVeque and Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, *SIAM J. Numer. Anal.* **31**, 1019 (1994).
24. R. J. LeVeque and Z. Li, Immersed interface methods for Stokes flow with elastic boundaries or surface tension, *SIAM J. Sci. Comput.* **18**, 709 (1997).
25. Z. Li and A. Mayo, ADI methods for heat equations with discontinuties along an arbitrary interface, *Proc. Symp. Appl. Math.* **48**, 311 (1994).
26. A. Mayo, The fast solution of Poisson's and the biharmonic equations on irregular regions, *SIAM J. Numer. Anal.* **21**, 285 (1984).
27. A. Mayo and A. Greenbaum, Fast parallel iterative solution of Poisson's and the biharmonic equations on irregular regions, *SIAM J. Sci. Stat. Comput.* **13**, 101 (1992).
28. C. S. Peskin, Numerical analysis of blood flow in the heart, *J. Comput. Phys.* **25**, 220 (1977).
29. K. Powell, *Solution of the Euler and Magnetohydrodynamic Equations on Solution-Adaptive Cartesian Grids*, Von Karman Institute for Fluid Dynamics Lecture Series, 1996.
30. J. J. Quirk, An alternative to unstructured grids for computing gas-dynamic flow around arbitrarily complex 2-dimensional bodies, *Comput. Fluids* **23**, 125 (1994).
31. J. J. Quirk, *A Cartesian Grid Approach with Hierarchical Refinement for Compressible Flows*, ICASE Rep. No. TR-94-51, NASA Langley Research Center, 1994.
32. S. S. Samant, J. E. Bussoletti, F. T. Johnson. R. H. Burkhart, B. L. Everson, and R. G. Melvin, *TRANAIR: A Computer Code for Transonic Analyses of Arbitrary Configurations*, AIAA Paper 87-0034, Reno, Nevada, 1987.
33. H. Van der Vorst, Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, *SIAM J. Sci. Stat. Comput.* **13**, 631 (1992).
34. A. Wiegmann, *The Explicit Jump Immersed Interface Method and Interface Problems for Differential Equations*, Ph.D. thesis, University of Washington, 1998.
35. Z. Yang, *A Cartesian Grid Method for Elliptic Boundary Value Problems in Irregular Regions*, Ph.D. thesis, University of Washington, 1996.