JOURNAL OF COMPUTATIONAL PHYSICS **136,** 522–545 (1997)
ARTICLE NO. CP975779

Adaptive Mesh Refinement and Multilevel Iteration for Flow in Porous Media

Richard D. Hornung[1] and John A. Trangenstein

Department of Mathematics, Duke University, Durham, North Carolina 27706

Received February 29, 1996; revised May 29, 1997

An adaptive local mesh refinement algorithm originally developed for unsteady gas dynamics by M. J. Berger is extended to incompressible flow in porous media. Multilevel iteration and domain decomposition methods are introduced to accommodate the elliptic/parabolic aspects of the flow equations. The algorithm is applied to a two-phase polymer flooding model consisting of a system of nonlinear hyperbolic mass conservation equations coupled to an elliptic pressure equation. While the various numerical methods used have been presented previously, our emphasis is on their consistent combination within the adaptive mesh refinement framework to treat important problems in porous media flow. To achieve efficient, easily maintainable code, we have exploited the features of object-oriented programming for the overall program structure and data management. Examples of algorithmic performance and computational results are provided.

1. INTRODUCTION

The numerical treatment of field-scale simulation of enhanced oil recovery and aquifer remediation processes is computationally expensive and may provide unsatisfactory results. There are several reasons that conventional simulations do not adequately resolve important flow features. First of all, much is unknown about the influence of fine-scale flow mechanisms on the macroscopic behavior approximated during field-scale simulations. Current information concerning the physical and chemical processes suggests complicated nonlinear relationships among a large number of dynamic flow variables. Often phase behavior models and reservoir geometry descriptions, to cite two examples, are simplified to allow simulations to be performed with reasonable computational expense. Second, when a multiphase fluid mixture is driven through a porous media, complicated fluid interface structures result. Standard computational methods have difficulty representing unstable fluid interfaces caused by subtle chemical and physical flow mechanisms and reservoir heterogeneity properly on a numerical mesh. In addition, the numerical dispersion and diffusion properties of conventional low-order methods can dominate the physical dispersion terms in the models. Third, porous media flow often involves important information on many physical scales. Representing physically meaningful data on various length and time scales efficiently during a numerical simulation is a formidable task.

To provide useful information for the development of recovery processes, field-scale simulations may need to resolve fine-scale localized flow behavior. Usually this means that the computational mesh must be sufficiently fine to resolve the length scales of important transient and static features. Consider that a single petroleum reservoir may be hundreds of meters thick and tens of kilometers in diameter and involve hundreds of wells. Also, fluid models used in enhanced oil recovery often encompass as many as 10 to 20 distinct chemical components. The magnitude of the systems of discrete equations resulting from the approximation of such large reservoir and fluid models severely restricts the possibility of using a computational mesh that allows sufficient resolution of all important flow features. To make simulation computationally viable, low resolution numerical methods are often combined with coarse numerical meshes. When these coarse numerical approximations do not resolve relevant flow features, the link between computational results and the actual flow in a hydrocarbon reservoir or aquifer is not clear. This reduces the usefulness of numerical simulation as a tool in the design of production processes.

In this paper, we demonstrate a combination of adaptive mesh refinement (AMR) and high resolution numerical discretization techniques. The use of AMR allows us to provide fine-scale resolution locally and to concentrate numerical effort near important flow features. Appropriately designed and implemented AMR algorithms have been shown to substantially reduce the computational expense needed to obtain a desired level of resolution in a variety of numerical simulation problems [5, 13, 14, 27, 43, 55, 77]. Due to the localized nature of fluid interfaces and other phenomena in subsurface simulation, AMR may

[1] Currently at Center for Applied Scientific Computing, Lawrence Livermore National Laboratory.

0021-9991/97 $25.00

provide similar benefits to field-scale simulation for petroleum recovery or aquifer remediation. In particular, industrial simulations using AMR would have some important advantages over uniformly fine mesh calculations. High resolution can be achieved locally with less computational expense allowing more simulations to be performed within a given computational budget. Also, computations employing AMR require less computer memory. Larger or more detailed simulations may be run than are possible with comparable uniformly fine meshes. Ideally, we would like to employ coarse-scale effective property models that provide sufficient resolution of important flow features and which make fine-scale simulation unnecessary. However, until such models are discovered, AMR is a tool by which local resolution can be achieved without overwhelming computational expense.

Specifically, we apply AMR to an incompressible, three-component, two-phase polymer flooding model. The polymer model presents several important mathematical and computational difficulties that complicate the application of AMR to more sophisticated models. In particular, the pressure equation and mass conservation equations are nonlinearly coupled in a fairly complicated manner. We use a sequential solution approach that separates the mixed hyperbolic/elliptic behavior in the flow equations [12]. While the individual methods used in the computational approach presented here need to be enhanced to treat practical problems, the combination of specialized numerical methods with adaptive mesh refinement is particularly challenging and is the primary focus of this paper. The algorithmic approach is sufficiently general that it extends to compressible flow problems involving complicated phase behavior models [78, 79]. This extension is the subject of ongoing work.

Key to achieving a successful adaptive approach is dealing with the communication between computational cells on a composite mesh in an efficient, mathematically, and physically meaningful manner. The subtle nature of this communication is reflected in the complexity of data structures and algorithms employed. To develop efficient, easily maintainable code, we have exploited the object-oriented features of C++ for the AMR program structure and data management, while numerically intensive routines are written in FORTRAN. In a typical two-dimensional problem, the overhead cost associated with AMR for interpatch communication and mesh adaptivity is approximately 20% of the cost of the entire computation. This is roughly twice the overhead cost reported in AMR computations for gas dynamics [13] and solid mechanics [77]. The additional expense is due to the communication costs associated with the iterative methods used to solve the elliptic parts of the flow equations.

In the remaining discussion, we first introduce the polymer flooding model in a form suitable for the sequential procedure we employ for its numerical solution. Next, we discuss the details of the numerical methods as they would be applied on a uniform mesh. The bulk of the paper is concerned with the application of AMR to systems of equations with mixed hyperbolic/elliptic character. We begin with an illustration of how the sequential solution procedure is carried out in the context of AMR. Then we address specific events in the AMR algorithm individually. These include temporal and spatial refinement of the computational mesh, interaction of data across mesh interfaces, the mesh adaptivity process, and mass conservation and consistency of the solution on the composite mesh. A description of the multilevel iteration process used to solve the pressure equation follows. We conclude with some numerical results, a discussion of computational timings for parts of the AMR algorithm, and a summary.

1.1. *Polymer Flooding Model*

The two-phase, immiscible polymer flooding model provides a simplified description of the flow of an oil and water mixture in which the viscosity of the water increases with the polymer concentration. We assume that there exist three fluid components (oil, water, and polymer) that flow in two phases, oleic (o) and aqueous (a). We assume no mass transfer between phases. The oil exists only in the oleic phase, whereas the water and polymer mixture forms the aqueous phase. For simplicity of discussion and to allow us to emphasize the numerical concerns in the application of adaptive mesh refinement, we assume that the flow is incompressible, and that adsorption, hysteresis, capillary pressure, and physical dispersion are all negligible. The basic equations of two-phase immiscible displacement are well known and treated in several sources [8, 9, 56, 59, 64]. The polymer flooding model is commonly presented as an extension of the classical two-phase Buckley–Leverett model [4, 65] and has been analyzed extensively from both mathematical and computational points of view [36, 40, 44, 47–50, 81].

We require that the mass of each of the fluid components is conserved subject to the constraint that the sum of the phase saturations is one:

$$s_{\mathrm{a}} + s_{\mathrm{o}} = 1. \tag{1}$$

Each phase saturation represents the ratio of phase volume to rock pore volume. The pore volume constraint combined with the assumption of no inter-phase mass transfer implies that a separate mass conservation equation for the oil component is redundant. Thus, the conservation of the masses of the three fluid components is completely described by Eq. (1) and the system of two conservation laws

$$\frac{\partial m^{\top}}{\partial t} + \nabla_x^{\top} F^{\top} = 0, \tag{2}$$

where

$$m \equiv \begin{bmatrix} s_a \\ c s_a \end{bmatrix} \phi,$$
$$F \equiv \begin{bmatrix} 1 \\ c \end{bmatrix} \left[\frac{\lambda_a}{\lambda_\top} (v_\top + \lambda_o(\rho_a - \rho_o) g \mathbf{K} \nabla_x d) \right]^\top. \quad (3)$$

The polymer concentration is c, ϕ is porosity, permeability is $\mathbf{K}$, and $\nabla_x d$ is the depth gradient. The permeability tensor and depth may be functions of the spatial position vector x. Incompressibility allows us to assume that the phase densities ρ_o, ρ_a are constant. The phase mobilities λ_o, λ_a and the total fluid mobility $\lambda_\top = \lambda_a + \lambda_o$ are functions of s_a and c.

Several empirical relationships are necessary to close the system of equations. Darcy's law relates the total volumetric flow rate $v_\top$ to the pressure gradient and gravity terms:

$$v_\top = -\mathbf{K}\lambda_\top [\nabla_x p - \lambda_\top^{-1}(\lambda_o \rho_o + \lambda_a \rho_a) g \nabla_x d]. \quad (4)$$

Incompressibility and the volume balance constraint give us a pressure equation:

$$\nabla_x^\top v_\top = 0. \quad (5)$$

The phase mobilities are defined as functions of s_a and c:

$$\lambda_o(s_a) = \frac{k_{ro}(s_a)}{\mu_o}, \quad \lambda_a(s_a, c) = \frac{k_{ra}(s_a)}{\mu_a(c)}.$$

The phase relative permeability model is $k_{ro}(s_a) = (1 - s_a)^2$, and $k_{ra}(s_a) = s_a^2$. The phase viscosities are given as $\mu_o = \mu_0$, and $\mu_a(c) = \mu_0(\mu_1 + c)$, where $\mu_0 = 0.35$, $\mu_1 = 0.50$. These mobilities and viscosities are not representative of any particular reservoir system. They are chosen because their functional form is used commonly in two-phase models. Also, they reveal certain mathematical and numerical difficulties in the flow equations. In particular, the given mobility functions imply that the hyperbolic system is not strictly hyperbolic and possesses local linear degeneracies [4, 42, 45].

In Section 5, we present numerical results from the simulation of a two-dimensional vertical cross-section between an injection well and a production well. The top and bottom of the reservoir are sealed. That is, $(v_\top)^\top \hat{n} = 0$ at any point along the top or bottom, where $\hat{n}$ is the outward unit normal to the reservoir. For the injection well, we specify the flow rate and the injected fluid composition. At the production well, we specify a single pressure value which represents pressure at the bottom of the well. The pressure at each depth in the producer is determined from the specified value and the assumption of gravity equilibrium.

To determine the total fluid velocity along the injection boundary, we assume that the velocity normal to the wellbore (i.e., along the boundary) is proportional to the transmissibility at the boundary. The proportionality constant is chosen so that the integral of the flow rate along the well is equal to the total prescribed flow rate. Since q is the specified rate of injection, $v_\top$ at height x_2 from the bottom of the well is

$$v_\top(x_2) = \frac{q}{\int_0^h e_1^\top(\lambda_\top \mathbf{K}) e_1 \, dz} \left(\frac{d}{dx_2} \int_0^{x_2} e_1^\top(\lambda_\top \mathbf{K}) e_1 \, dz \right) \Bigg|_{x_2}, \quad (6)$$

where h is the length of the injection well. Once the injection velocity is known, we can determine the injection pressure by using Eq. (4) along the boundary. Generally, this requires the solution of a linear system along the entire boundary as the well pressure values are coupled.

At the production well, we must determine the pressure along the boundary given the aforementioned "bottomhole" pressure. The gravity equilibrium assumption states that the sum of all fluid flow driving forces in the vertical direction (i.e., perpendicular to the direction of bulk flow) is zero. That is,

$$e_2^\top [\nabla_x p - \lambda_\top^{-1}(\lambda_o \rho_o + \lambda_a \rho_a) g \nabla_x d] = 0 \quad (7)$$

along the boundary. Note that the well boundary conditions are time-dependent in that the mobility and density terms along the boundary depend on the fluid in the wells and the fluid within the reservoir. For further details on wells and boundary conditions used in reservoir simulation; see Lake [56] and Peaceman [64].

2. NUMERICAL TREATMENT OF FLOW IN POROUS MEDIA

Multiphase flow in oil recovery and aquifer remediation processes exhibits behavior typical of the solutions to both elliptic/parabolic and hyperbolic partial differential equations. For example, pressure changes are felt quickly throughout the reservoir when the flow is incompressible or only slightly compressible. In contrast, fluid fronts tend to move with much slower speeds and can be fairly sharp. Generally, the equations of multiphase porous media flow can be written as a system of conservation equations for the masses of the fluid components, subject to a constraint that the fluid fills the void space in the rock. If the model describes incompressible flow, one obtains a pressure equation (recall Eq. (5)) by summing the mass conservation equations and applying the volume balance constraint [4, 41]. In the compressible case, it is often reasonable to

linearize the volume balance constraint in time to develop a nonlinear parabolic pressure equation [12, 78, 79]. In either case, the pressure equation relates the pressure of the fluid to its composition through phase behavior relationships and rock heterogeneity. Separating the flow equations allows us to develop a numerical approach suited to the specific character of each equation. For discussion of the effectiveness of the sequential approach for convection-dominated flow associated with enhanced oil recovery problems; see the work of Bell, Shubin, and Trangenstein [12, 78, 79].

As we saw in Section 1.1, the pressure equation (5) and the mass conservation equation (2) in the polymer model are nonlinearly coupled in several ways. The presence of the mobilities in the pressure equation implies that p and $v_\top$ depend on s_a and c. The conserved quantities depend on $v_\top$ (and p), since $v_\top$ appears in the flux matrix. The sequential formulation specifies that the conservation law and the pressure equation are alternately advanced as part of an overall discrete time integration procedure. This means that saturation and concentration are fixed when we solve the pressure equation. Thus, we can interpret the pressure equation as a linear elliptic partial differential equation. For the incompressible polymer model, the sequential method is essentially the IMPES (implicit pressure/explicit saturation) formulation commonly found in the numerical simulation of flow in porous media [8, 64]. The pressure equation is solved in an iterative fashion by applying an appropriate linear solver. The solution to the pressure equation is then used to compute the fluid velocity necessary to evaluate the flux in the conservation equations. Once the phase velocities are known, we can apply any of several numerical methods that integrate hyperbolic systems in a mass conservative fashion. Using a higher-order method with AMR is advantageous so that fluid interfaces can be resolved over a relatively small number of cells, thereby allowing mesh refinement to be concentrated in localized regions of the flow domain.

As was demonstrated in Section 1.1, the separation of the governing equations into a hyperbolic part and an elliptic part is fairly simple for the polymer flooding model. The individual numerical methods that we use have been described in previous studies. The Godunov method and the mixed finite element method are discussed in [4, 12, 22, 84]. The success of the sequential solution approach for important problems in flow in porous media has also been demonstrated [8, 12, 41, 79, 78]. Thus, we focus on the incorporation of these methods into an AMR algorithm. The strong nonlinear dependencies among the unknowns in the equations and the nonlocal coupling between the pressure and total fluid velocity are the more substantial hurdles to overcome in applying AMR to flow in porous media. The portions of the AMR algorithm dealing with the hyperbolic equations are essentially the same as those in the work of Berger and Colella [13] and Trangenstein [77]. The algorithm is extended through the incorporation of multilevel iteration techniques to solve the elliptic pressure equation on the composite mesh that allows both temporal and spatial refinement locally within the problem domain.

2.1. *The Polymer Model Mass Conservation Equations*

It is important to approximate the time integration of the conservation equations (2) with a numerical method that is strictly conservative. Methods that are not conservative can produce nonphysical traveling waves [57] and prohibit the use of conservation-oriented scaling rules on an adaptive mesh. We employ the second-order unsplit Godunov method to integrate the hyperbolic system of mass conservation equations. By the term *unsplit,* we mean that the numerical approximation of the time evolution of the multidimensional conservation laws is not represented as a product of one-dimensional operators [22]. The method has been shown to provide superior resolution of complicated wave behavior associated with flow in porous media when compared to standard methods [10, 41, 55, 76, 78, 79]. A primary advantage of the Godunov approach is that it produces less numerical dispersion and dissipation than conventional methods, especially when applied to multidimensional problems [41, 73]. The grid orientation problems resulting from operator splitting and donor cell methods are reduced by combining higher order accuracy and appropriate transport in directions not aligned with coordinate directions induced by the computational mesh [22].

Recall the system of conservation laws for the polymer flooding model, Eq. (2),

$$\frac{\partial m^\top}{\partial t} + \nabla_x^\top F^\top = 0,$$

where the vector of conserved quantities and the flux matrix are given by Eq. (3). The Godunov method is based on the following conservative difference on a rectangular mesh:

$$\begin{aligned} m_{ij}^{n+1} = m_{ij}^n \\ - \Delta t^n \left[\frac{(Fe_1)_{i+1/2,j}^{n+1/2} - (Fe_1)_{i-1/2,j}^{n+1/2}}{(\Delta x_1)_i} \right. \\ \left. + \frac{(Fe_2)_{i,j+1/2}^{n+1/2} - (Fe_2)_{i,j-1/2}^{n+1/2}}{(\Delta x_2)_j} \right]. \end{aligned} \tag{8}$$

Here, (i, j) are the indices of the cell centers and n is the index of the timestep. The unit axis vector in the nth coordinate direction is represented by e_n. The following Courant–Friedrichs–Lewy (CFL) condition is required for numerical stability of the unsplit Godunov method [22]:

$$\max\left\{\max_{ij}\left\{\max_{k}\{|(\lambda_1^{(k)})_{ij}|\}\frac{1}{(\Delta x_1)_i}\right\},\right.$$
$$\left.\max_{ij}\left\{\max_{k}\{|(\lambda_2^{(k)})_{ij}|\}\frac{1}{(\Delta x_2)_j}\right\}\right\}\le\frac{1}{\Delta t^n}.$$

Here, we consider the maximum over all cells (i, j), and all eigenvalues $\lambda^{(k)}$ of the flux matrix F associated with each coordinate direction.

The computation of the numerical flux terms (Fe_1), (Fe_2) involves several steps. Formal second-order accuracy is facilitated in regions where the flow is smooth by constructing conservative piecewise-linear interpolants to each of the conserved variables within each cell [82]. The fluxes are evaluated by approximating the solutions to Riemann problems posed at cell boundaries. Initial data for the Riemann problems are constructed using the linear approximations, characteristic tracing, and transverse flux correlations to approximate the interaction of data in nearby cells. The transverse flux correction has a stabilizing effect on the numerical time integration that allows an improved CFL condition over other higher order approaches, such as TVD [75] or ENO [72] schemes, as well as the conventional first-order upstream weighting method [70]. We will not discuss the Godunov method in detail in this paper. The interested reader is encouraged to consult [10, 11, 37, 41, 63, 82] for further information. We outline the major features of the scheme below to show how it is applied to the polymer model.

For the application of the Godunov method, we use the vector of "flux variables" $w \equiv [s_a, c]^\top$, rather than the conserved variables m. Clearly, the conserved quantities are easily computed from the flux variables. We rewrite the conservation law in terms of the flux variables as

$$\left(\frac{\partial m}{\partial w}\bigg|_x\right)\frac{\partial w}{\partial t}+\sum_{n=1}^{2}\left\{\frac{\partial Fe_n}{\partial w}\bigg|_x\frac{\partial w}{\partial x_n}+\frac{\partial Fe_n}{\partial x_n}\bigg|_w\right\}=0.$$

The system is hyperbolic [4, 46], and we have

$$\left(\frac{\partial Fe_n}{\partial w}\bigg|_x\right)X_n=\left(\frac{\partial m}{\partial w}\bigg|_x\right)X_n\Lambda_n,$$

in either coordinate direction, $n \in \{1, 2\}$.

Taylor series expansions are used to determine approximate states at the cell edges at half time levels. For example, states at cell edges whose normal vector lies in the first-coordinate direction are approximated as

$$\begin{aligned}
&w\left(x_1\pm\frac{\Delta x_1}{2},x_2,t+\frac{\Delta t}{2}\right)\\
&\quad\approx w(x_1,x_2,t)\pm\frac{\Delta x_1}{2}\frac{\partial w}{\partial x_1}\bigg|_{(x_1,x_2,t)}+\frac{\Delta t}{2}\frac{\partial w}{\partial t}\bigg|_{(x_1,x_2,t)}\\
&\quad\approx w(x_1,x_2,t)\pm\frac{\Delta x_1}{2}\frac{\partial w}{\partial x_1}\bigg|_{(x_1,x_2,t)}-\frac{\Delta t}{2}\left[\left(\frac{\partial m}{\partial w}\bigg|_x\right)^{-1}\right.\\
&\qquad\left.\left\{\frac{\partial Fe_1}{\partial w}\bigg|_x\frac{\partial w}{\partial x_1}+\frac{\partial Fe_1}{\partial x_1}\bigg|_w+\frac{\partial Fe_2}{\partial x_2}\right\}\right]\bigg|_{(x_1,x_2,t)}\\
&\quad\approx w(x_1,x_2,t)\pm\left(X_1\left[I\mp\Lambda_1\frac{\Delta t}{\Delta x_1}\right]X_1^{-1}\frac{\partial w}{\partial x_1}\frac{\Delta x_1}{2}\right)\bigg|_{(x_1,x_2,t)}\\
&\qquad-\frac{\Delta t}{2}\left[\left(\frac{\partial m}{\partial w}\bigg|_w\right)^{-1}\left\{\frac{\partial Fe_1}{\partial x_1}\bigg|_w+\frac{\partial Fe_2}{\partial x_2}\right\}\right]\bigg|_{(x_1,x_2,t)}.
\end{aligned}$$

Then, the initial data for the Riemann problem that determines the flux at edge $(i + 1/2, j)$ is

$$w^L_{i+1/2,j}=w^{P,L}_{i+1/2,j}-\frac{\Delta t}{2}\left(\frac{\partial m}{\partial w}\bigg|_x\right)^{-1}_{ij}\left\{\frac{\partial Fe_1}{\partial x_1}\bigg|_w+\frac{\partial Fe_2}{\partial x_2}\right\}_{ij},$$
$$w^R_{i+1/2,j}=w^{P,R}_{i+1/2,j}-\frac{\Delta t}{2}\left(\frac{\partial m}{\partial w}\bigg|_x\right)^{-1}_{i+1,j}\left\{\frac{\partial Fe_1}{\partial x_1}\bigg|_w+\frac{\partial Fe_2}{\partial x_2}\right\}_{i+1,j},$$

where the terms involving the characteristic tracing are

$$w^{P,L}_{i+1/2,j}=w_{ij}+\left(X_1\left[I-\Lambda_1\frac{\Delta t}{\Delta x_1}\right]X_1^{-1}\right)_{i,j}\left(\frac{\partial w}{\partial x}\frac{\Delta x_1}{2}\right)_{ij},$$
$$w^{P,R}_{i+1/2,j}=w_{i+1,j}-\left(X_1\left[I+\Lambda_1\frac{\Delta t}{\Delta x_1}\right]X_1^{-1}\right)_{i+1,j}\left(\frac{\partial w}{\partial x}\frac{\Delta x_1}{2}\right)_{i+1,j}.$$

Analogous quantities corresponding to edges in the x_2-coordinate direction are computed similarly. The spatial derivatives of the flux variables corresponding to cell centers are approximated using the MUSCL slope-limiting approach of van Leer [82]. The terms involving the flux derivatives arise from the corner-transport upwind scheme [22]. Spatial derivatives of the flux terms are approximated by applying finite differences to preliminary fluxes. The initial data for the Riemann problems used to compute the preliminary fluxes are given by the $w^{P,L/R}$ terms. The transverse flux correction allows the improved CFL criterion alluded to earlier.

Although, the exact solution to the Riemann problem for the polymer model is well understood [36, 44, 47–50, 81], we approximate the solution in the interest of computational efficiency. The application of the Godunov method to the polymer model involves several modifications of the basic Godunov approach. First, there are bounds on the

aqueous phase saturation s_a and the polymer concentration c that must be enforced in the characteristic tracing and Riemann problem solution so that intermediate states are physically meaningful; namely, $0 \le s_A,\ c \le 1$. Second, the system exhibits a loss of strict hyperbolicity when the eigenvalues (corresponding to the Buckley–Leverett wave speed and the particle velocity) are equal. To overcome this difficulty when approximating the Riemann problem solutions, we resort to the first-order Rusanov flux [68] when this occurs in combination with a change in sign of the Buckley–Leverett wave speed. Essentially, this increases the local numerical diffusion of the scheme to prevent entropy violations [10]. Third, the flux is a function of the spatial variable x, in addition to the flux variables. As we can see from the formulae above, the transverse flux correction is easily adapted to this situation [41, 42].

2.2. *The Polymer Model Pressure Equation*

To approximate the pressure equation (5) in a manner consistent with the hyperbolic system, the discrete divergence operator must be the same in both. We accomplish this by using an approximation related to the mixed finite element with the lowest order Raviart–Thomas rectangular elements [66]. The mixed method produces the standard cell-centered finite difference approximation, common in industrial reservoir simulation, when particular numerical quadrature rules are used to evaluate the necessary integrals induced by the variational form of the equation [69, 84]. We will not describe the mixed method in detail; rather, we comment on the aspects of the method most relevant to our current application. The interested reader is encouraged to consult [1, 7, 34, 66] for a complete description of the method.

For simplicity, we consider diagonal tensor permeability. This results in the common finite-difference approximation of the pressure equation involving harmonic averaging of transmissibility terms across cell boundaries. In the lowest order Raviart–Thomas approximation spaces, the pressure and the divergence of the total fluid velocity are represented as piecewise-constant elements, constant on each rectangular mesh cell. Tensor products of continuous piecewise-linear and piecewise-constant elements are used for each component of the velocity vector. The Raviart–Thomas spaces model the continuity of $v_\top$ and the discontinuities possible in $\nabla_x p$ in a physically and mathematically appropriate manner. In addition, the method possesses super-convergence properties on irregular meshes [33, 84]. When the product $\mathbf{K}\lambda_\top$ is diagonal and sufficiently smooth, the pressure and velocity fields are second-order accurate in the L_2-norm [84].

The accuracy of the velocity field is not maintained on a composite mesh, however. Computational and theoretical results show that the method loses an order of accuracy near coarse–fine mesh interfaces [34, 42, 62]. Recent developments in the application of the mixed finite element method to more general mesh geometries and locally refined grids have incorporated Lagrange multipliers along mesh interfaces to improve flux approximation near such interfaces [87]. We have not yet considered the latest approaches to preserve flux continuity on composite meshes nor approaches to handle full tensor permeability fields (e.g., control-volume finite element methods [51]), although this will be a subject of future work. For random permeability fields, or permeability fields involving substantial discontinuities, elliptic regularity theory does not necessarily allow high-order accuracy. Also, shock-capturing methods, such as the Godunov method, are first-order at best near discontinuities in the solution. It remains to be explored whether new, more sophisticated approximation techniques can be made computationally viable in large-scale AMR calculations.

The approximation of the two-dimensional version of Eqs. (4) and (5) using the lowest order mixed method results in a discrete system with three unknowns per grid cell: the cell-centered pressure and the normal velocity at each cell edge. The linear system is nonsingular and symmetric, but not positive definite. When $\mathbf{K}$ is diagonal, the size of the linear system can be reduced by eliminating the velocities from the system [28]. The resulting discrete system has only the cell-centered pressure values as the unknowns; thus, one is left with only one unknown per cell, and a symmetric, positive definite linear system.

To discretize the normal component of $\mathrm{v}_\top$ at each cell edge on our rectangular mesh, we approximate the normal component of the pressure gradient at the cell edges as

$$(\delta_x p)^{(1)}_{i+1/2,j} = \frac{p_{i+1,j} - p_{i,j}}{(h_1)_{i+1/2}}, \quad (\delta_x p)^{(2)}_{i,j+1/2} = \frac{p_{i,j+1} - p_{i,j}}{(h_2)_{j+1/2}},$$

where the mesh increments are defined as

$$(h_1)_{i+1/2} = \tfrac{1}{2}[(\Delta x_1)_i + (\Delta x_1)_{i+1}],$$
$$(h_2)_{j+1/2} = \tfrac{1}{2}[(\Delta x_2)_j + (\Delta x_2)_{j+1}].$$

The components of the vector of gravity terms at each cell edge are approximated as

$$\gamma^{(k)}_{i+1/2,j} = \tfrac{1}{2}\{e_k^\top (\nabla_x d)_{i+1/2,j}((\rho_\top)_{i,j} + (\rho_\top)_{i+1,j})g\},$$
$$\gamma^{(k)}_{i,j+1/2} = \tfrac{1}{2}\{e_k^\top (\nabla_x d)_{i,j+1/2}((\rho_\top)_{i,j} + (\rho_\top)_{i,j+1})g\}.$$

Here, the mobility-weighted density terms are $(\rho_\top) = \lambda_\top^{-1}(\lambda_o\rho_o + \lambda_a\rho_a)$, and k corresponds to the coordinate direction. Finally, the normal components of the total fluid velocities (we omit the subscript $\top$) across the cell edges are

$$\begin{aligned} v_{i+1/2,j} &= -e_1^T(\mathbf{K}\lambda_T)_{i+1/2,j}e_1[(\delta_x p)^{(1)} - \gamma^{(1)}]_{i+1/2,j}, \\ v_{i,j+1/2} &= -e_2^T(\mathbf{K}\lambda_T)_{i,j+1/2}e_2[(\delta_x p)^{(2)} - \gamma^{(2)}]_{i,j+1/2}. \end{aligned} \quad (9)$$

The transmissibility term $\mathbf{K}\lambda_T$ at each cell edge is defined as the appropriate harmonic average of $\mathbf{K}\lambda_T$ associated with adjacent cell centers.

The discrete version of Eq. (5) is completed by specifying that

$$(\Delta x_2)_j[v_{i+1/2,j} - v_{i-1/2,j}] + (\Delta x_1)_i[v_{i,j+1/2} - v_{i,j-1/2}] = 0 \quad (10)$$

must hold in each cell. Note that this discrete divergence operator is identical to that used in the conservative difference in Eq. (8). This is important to maintain the numerical consistency of the solutions between the two equations [42].

3. ADAPTIVE MESH REFINEMENT

3.1. *Overview of AMR for Flow in Porous Media*

Adaptive mesh refinement has proved to be a valuable computational technique when combined with high resolution shock-capturing methods for multidimensional simulations in a variety of problem areas, such as shock hydrodynamics [13, 27] and nonlinear solid mechanics [77]. The governing equations in these cited studies consisted primarily of nonlinear hyperbolic systems of partial differential equations. The development of an AMR strategy was driven by the need to accurately resolve unsteady flows with shocks in a computationally efficient manner. Accurately resolving discontinuities in the solution and maintaining global conservation were primary considerations in the development of the AMR approach. These issues are vital to the successful application of AMR to flow in porous media as well. However, there are additional considerations regarding porous media flow equations that complicate the incorporation of AMR.

During the past decade, a few AMR strategies have been developed to treat flow in porous media [24–26, 29, 71]. Commonly, the local mesh refinement was implemented by refining individual computational cells one at a time. We refer to this practice as *cell* refinement. The AMR paradigm that we employ is based on the ideas introduced by Berger and Oliger [15], and extended by Berger, Colella [13], and Trangenstein [77]. Once individual cells are selected for refinement, the algorithm clusters them to form a collection of logically rectangular *patches*. The refinement on any level other than the coarsest is defined so that the union of the patches is contained in the interior of the region determined by the union of the patches on the next coarser level. This is referred to as ''nested'' refinement.

The patch AMR approach has several important advantages over the cell refinement strategy. These advantages are significant for a wide range of computational problems including, but not limited to, flow in porous media. The cell refinement approach tends to produce refined regions that more tightly conform to the spatial structure of the features requiring mesh refinement since the patch approach refines additional cells to maintain rectangular regions. However, in the cell approach, the computational tasks are organized necessarily by operations performed on an individual cell. This leads to indirect addressing and irregular difference schemes to maintain communication among data associated with computational cells on the composite mesh. Moreover, the data structures maintaining the cell refinement need to be closely tied to the finite difference stencils used by the numerical methods. The irregular mesh configurations generated by the cell refinement strategy can present significant impediments to efficient linear algebra routines, parallelization, vectorization, and cache efficiency as well.

Patch refinement allows the intercell communication to be maintained in a more straightforward manner and is designed so that a small number of computationally rich tasks can be organized in a highly structured fashion. We are able to use integration and iteration routines developed for logically rectangular meshes on all patches on all levels in the adaptive mesh hierarchy. Therefore, the user interface in the code appears much like it does in a conventional simulator. Moreover, the implementation of efficient and accurate high resolution numerical routines is better understood on logically rectangular meshes [13, 77]. In addition, the numerical integration routines used to solve the equations on the adaptive grid are easily separated from the structure of the AMR algorithm. This greatly enhances code extendibility, maintenance, and generality. Finally, high resolution numerical methods and domain decomposition methods can be made very efficient on vector and parallel computers when the data is organized in a highly structured fashion that closely resembles the manner in which data is stored in computer memory [13].

It will become evident in the discussion that follows that the implementation of AMR requires substantially more sophisticated programming than conventional integration on regular meshes. The complexity cannot be avoided using any particular AMR paradigm, whether patch-oriented or cell-oriented. In the present discussion, program complexity is most prominent in the data structures representing the patches defining the various refinement levels, the communication between the patches, and the treatment of boundary conditions. The encapsulation and dynamic binding capabilities of C++ have allowed us to develop a very general adaptive mesh refinement code [74]. The majority of the code supporting all facets of the adaptive algorithm is independent of the fluid flow model and the number of spatial dimensions. The numerically intensive

routines for integration and patch communication are implemented in FORTRAN. This is advantageous since FORTRAN provides direct language support for efficient manipulation of logically rectangular data arrays and a much more sophisticated mathematical function library. As a result, the numerical routines appear much as they would in a conventional simulator written entirely in FORTRAN. The major difference is that the main time-stepping loop and the treatment of boundary data for the patches is handled by the C++ code controlling the AMR algorithm. For further details on the object-oriented implementation of AMR, see [23, 43, 77].

Before we commence discussion of the details of the AMR algorithm, we summarize the properties satisfied by the composite mesh configuration. The adaptive local mesh refinement process automatically and dynamically generates a hierarchy of levels of mesh refinement. The coarsest mesh level covers the entire computational domain. The region covered by the patches on each finer level is contained within the region covered by the patches associated with the next coarser level. Within each level, cells are grouped so that they are maintained as a list of logically rectangular patches. The boundaries of the patches on each finer level align with the boundaries of cells on the next coarser level. In other words, no cell is allowed to be refined partially. Our convention is to refer to the different mesh levels by a level number so that we know how the levels relate to each other when discussing the mesh configuration. In particular, we number the levels in increasing order from coarse to fine with the coarsest level numbered 0.

3.2. *The Sequential Solution Procedure in the Context of AMR*

The process of advancing the numerical solution over a given time interval when using the sequential method consists of two independent steps. In the first step, the hyperbolic conservation law is integrated in time. Second, the elliptic pressure equation is solved at the new time to provide a velocity field associated with the new time. This sequence of events is straightforward on a uniform mesh. Special care must be taken to treat the time integration process properly when using AMR.

Consider advancing the full system of equations over a time increment on a single level within the AMR hierarchy. The integration of the mass conservation equations uses an explicit conservative difference. Communication between cells associated with patches on the single mesh level is all that is needed once boundary values for cells bordering the union of patches on the level and the physical boundary are supplied. In contrast, the iterative method that solves the elliptic boundary value problem for pressure requires us to enforce communication globally within the computational domain. Thus, data on patches on different levels must interact repeatedly during the solution process.

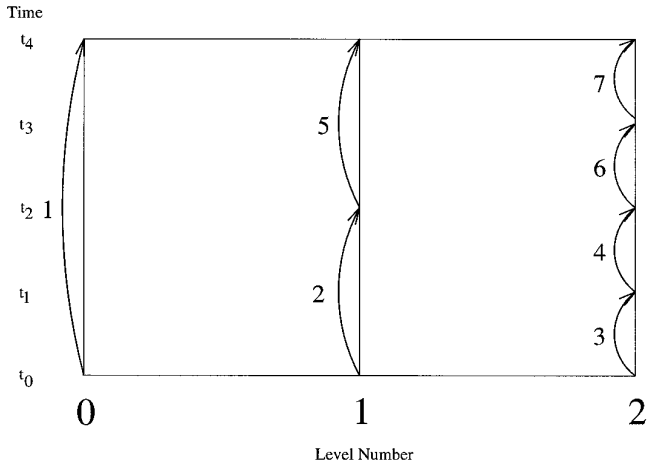

FIG. 1. Sample timestep sequence during AMR: three mesh levels, refinement ratio =2. The time levels corresponding to the solution on the finest level (level 2) are given on the vertical time axis at left. The order of integration on the different levels is illustrated by the numbers associated with the curved vertical arrows.

3.3. *The AMR Algorithm*

To motivate some of the subtle points in the AMR methodology, we address, by way of an illustrative example, some algorithmic issues that require special care. In particular, the value of any flow quantity at a point in the spatial and temporal computational domain may depend on the level of mesh one is observing. Special care is taken to ensure the consistency of the different flow variables on the adaptive mesh. In other words, the time integration of the flow equations on the different levels in the adaptive mesh hierarchy requires that the numerical solution on the different levels be synchronized when and where appropriate.

The process of advancing data on a level through a specified time increment involves several distinct tasks. First, the patches on the level are integrated over a specified time increment. Then, each finer level (if it exists) is advanced by a recursive invocation of the integration process on the finer level. Finally, when all data on the finer levels have been advanced to the time associated with the data on the current level, the data on the levels are synchronized. The order of the timesteps during this recursive process in a simple scenario involving three levels of mesh is illustrated in Fig. 1. Notice that each time increment on level 0 or 1 is partitioned into several smaller time intervals corresponding to the time increments on the next finer level. Thus, several timesteps must be taken on each level (finer than level 0) before it is appropriate to synchronize its data with that on a coarser level. Since we consider a refinement ratio of 2 in this example, the size of the timestep on a finer level is $\frac{1}{2}$ the size of the timestep

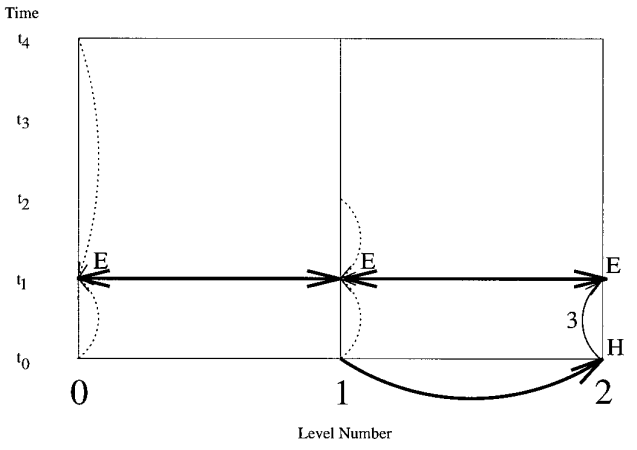

FIG. 2. In step 3, the conservation law is advanced to time $t = t_1$ on level 2, and then the pressure equation is solved on level 2 using all three levels. The curved horizontal arrow denotes the transfer of boundary data necessary to integrate the conservation equation, while the straight horizontal arrows denote the exchange of pressure equation data during the iterative solution process. The dotted, curved, vertical arrows indicate time interpolation for the pressure equation coefficients on levels 0 and 1 before we begin the pressure equation solution process. The H indicates where a hyperbolic advance occurs, and the E indicates where the elliptic equation is solved.

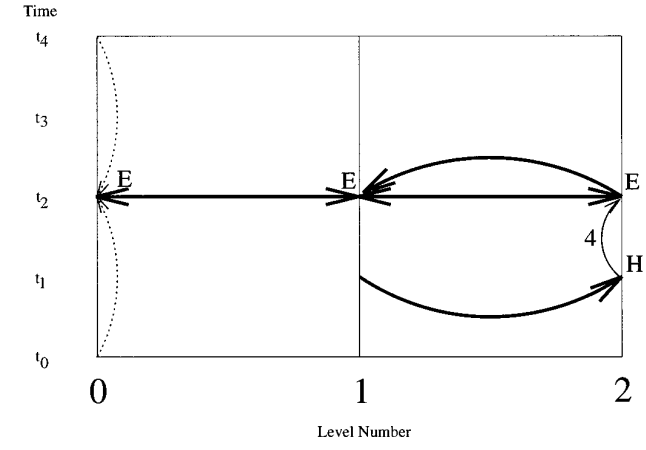

FIG. 3. In step 4, we advance the solution on level 2 to time $t = t_4$. Then, once the pressure equation is solved on all three levels, the data on levels 1 and 2 are synchronized. The curved horizontal arrow directed from right to left represents the data synchronization.

on the next coarser level. However, this is not the case in general as we explain in Section 3.4.

Let us consider the events associated with a few steps in the sequence in Fig. 1. Figure 2 shows the operations performed during the first fine timestep (step 3). First, we refine information from level 1 to level 2 at time $t = t_0$ to provide the boundary data required to advance the conservation law on level 2. This is indicated by the curved horizontal arrow directed from left to right in the figure. After advancing the conservation law to time $t = t_1$ (signified by the letter "H" for "hyperbolic" in the figure), we solve the pressure equation using all three levels (marked by the "E" for "elliptic" appearing on each level in the figure). The double arrows symbolize the transfer of information between levels 2 and 1, and 1 and 0 during the iterative solution process. The dotted curved vertical arrows represent the time interpolation of the pressure equation coefficients on the coarser levels before the iterative solution process is begun.

Figures 3 and 4 illustrate similar sequences of events for an intermediate and a final fine timestep. Figure 3 shows step 4 in which the solution on level 2 is advanced to time $t = t_2$ to synchronize with the solution on level 1. The procedure is similar to that which was carried out in step 3. However, a few differences are worth mentioning. After we integrate the mass conservation equations to time $t = t_2$, we synchronize the conserved quantities on levels 1 and 2 with the process described in Section 3.7. The curved horizontal arrow directed from level 2 to level 1 indicates the transfer of data during the synchronization process. Then, the pressure equation is solved at time $t = t_2$. However, since the pressure solve was deferred on level 2 earlier, we need to initiate the solve with level 2 as the finest level in the solution process. Following the completion of the pressure solve, the pressure and velocity fields on level 1 replace the values previously obtained after step 2. At the completion of step 4, all data on levels 1 and 2 are synchronized. In Fig. 4, we illustrate timestep 7, the last in the sequence. Once the conservation law is advanced on level 2 to time $t = t_4$, the conserved data is synchronized on all three levels. Then, the pressure equation is solved by iterating between all three levels once again. After the multilevel iteration is complete, the pressure and velocity

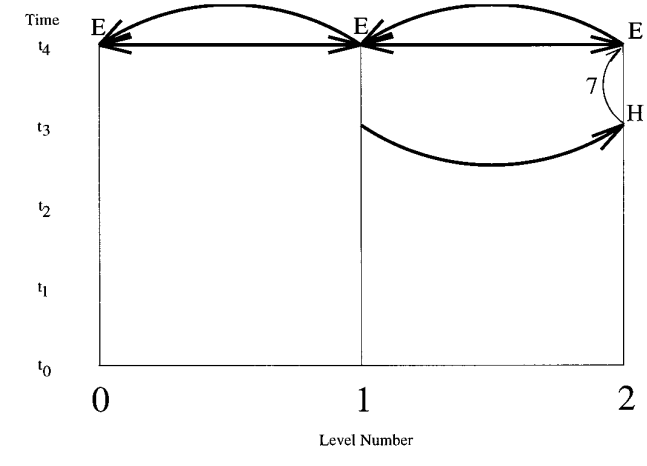

FIG. 4. After the final step (step 7), all data on all levels are synchronized at time $t = t_4$.

fields on all three levels are synchronized. Then, all data on all levels are synchronized at time $t = t_4$.

This brief illustrative description of the time integration program is intended to give the reader a sense of the arrangement of events that comprise the AMR algorithm. Although we have only discussed steps corresponding to the integration of the data on level 2, we remark that similar tasks are performed on levels 0 and 1. The treatment of the data on the coarser levels involves straightforward simplifications of the operations illustrated in the figures. Thus, we have omitted them. We now address the details of the various events individually.

3.4. *Selection of Time Increment*

In addition to the CFL condition (Section (2.1)), algorithmic considerations restrict the magnitude of the time increment over which we integrate. Once the stable time increment size is determined using the CFL condition, we reduce the increment, if necessary, to force an appropriate number of timesteps to be performed on the mesh level to reach a point of synchronization with the next coarser level. If we are at the finest mesh level and the level does not allow additional refinement, then there is no restriction on the size of the timestep aside from the CFL condition. On any level that allows refinement, we require that an even number of timesteps are performed on the level before synchronization with the next coarser level occurs. This requirement facilitates the regridding process described in Section 3.6.

It is important to note that we do not force the time increment on a finer level to be the size of a time increment on the next coarser level divided by the mesh refinement ratio. As a simulation evolves, changes in characteristic speeds cause changes in the stable timestep size determined by the CFL condition. Rapid changes in characteristic speeds can be resolved better on finer meshes. Commonly, we see that it is appropriate to choose a smaller timestep than that which is suggested by dividing the coarse time increment by the refinement ratio. This behavior is also observed by Trangenstein when applying AMR to nonlinear solid mechanics [77]. Thus, we allow the time intervals corresponding to the steps in the timestep loop on a single refinement level to vary. However, we do attempt to make the size of each timestep in the loop roughly the same.

3.5. *Boundary Data for a Patch*

In the discussion of the time integration in Section 3.3, we saw that coarse patches are integrated before fine patches. Therefore, coarse patches must provide boundary data to finer patches. Proper determination of boundary data for each patch is vital to the success of numerical integration on an adaptive mesh. The interpatch communication problem is facilitated by the notion of "ghost cells."

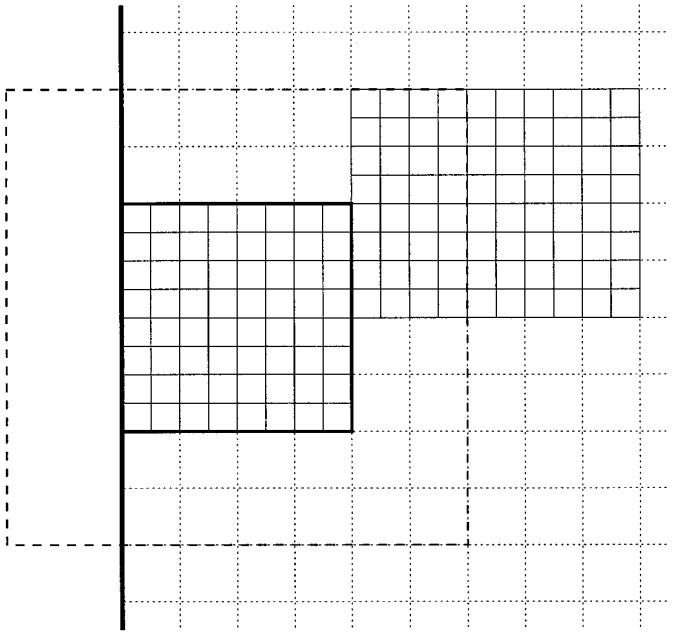

FIG. 5. Sources for ghost cell data during AMR. The boundary for the computational domain is the bold vertical line at left. The patch for which we seek ghost cell information is next to the physical boundary and has a boundary represented by line segments of intermediate thickness. The fine patch requires ghost cell information in cells inside the larger box whose boundary is indicated with dashed line segments.

Ghost cells correspond to the same index space as those cells associated with a patch interior, but lie outside of the patch boundary. If a ghost cell lies outside of the computational domain, the data is obtained from user-prescribed boundary conditions. If the ghost cell for a patch lies in the interior of an adjacent patch on the same refinement level, the data is copied from the neighboring patch interior. If some ghost cell data cannot be obtained in either of these two ways, then they are recursively sought by interpolating from coarser levels.

Figure 5 illustrates a possible scenario in which we must determine ghost cell data from each of these situations. The patch requiring ghost cell information is the left-most fine patch bounded by line segments of intermediate thickness in the figure. The region over which ghost cell data is sought lies between the boundary of the fine patch and the box whose boundary is indicated with dashed line segments. In this example, we need data in a region four fine cells wide around the fine patch. The ghost cells lying to the left of the bold vertical line are outside of the computational domain. Their data are determined by the user-prescribed boundary conditions. The data in the ghost cells intersecting the neighboring fine patch are simply copied from the appropriate cells located in the interior of that fine patch. The remainder of the ghost cell data are obtained by refining data from underlying coarse cells. Potentially, the

coarse cell data must be interpolated in time as well as space. Also, it is possible that we need to go to even coarser levels recursively to determine all ghost cell data for a patch.

The details of the process of determining ghost cell data depend on the location of the patch within the computational domain, the mesh level to which it corresponds, and the current state of the algorithm. The width (in number of cells) of the ghost cell region around a patch depends on the current state of the algorithm and the flow variables involved. When data are provided by coarser levels, the data interpolation procedure also depends on the flow variables involved. For instance, conservative linear interpolation is appropriate for the conserved variables in the conservation law. In contrast, the pressure values needed around patch boundaries during the pressure equation solve are obtained using multilinear interpolation.

3.6. *The Regridding Process*

When treating time-dependent problems involving transient phenomena, mesh refinement must move as a simulation evolves. At certain points during the calculation, a regridding process is invoked during which the algorithm adjusts the patch configuration on all levels finer than the mesh level from which the regridding was initiated. Note that if the time corresponding to the data on a level matches that of the data on a coarser level, we must be at a synchronization point. Then regridding may be deferred to some coarser level if it is also appropriate to invoke the regridding process on the coarser level at that time.

There are three main design principles enforced during the regrid process. First, new fine mesh should be located only where it is needed to provide adequate resolution or accuracy. In other words, computational efficiency dictates that we avoid unnecessarily large refined regions. Second, the nesting of the mesh levels must be maintained. Third, the mesh is not altered after every timestep on each level since the costs associated with the determination of refined cells and the movement of data as patches are created and destroyed are potentially substantial. The primary aim is to keep the cost of the regridding process as small as is reasonable to attain desired resolution.

In previous AMR work [13, 42, 55, 77], an estimate of the global truncation error was used to identify cells for refinement. The goal was to guarantee a specified level of global accuracy throughout a simulation with nearly minimal computational cost. The procedure combined Richardson extrapolation with a simple appraisal of the total number of timesteps needed on a given mesh level to complete the simulation. There are several important advantages to this process, the most important of which is that the method is fairly general. But it can produce larger refined regions than may be desired for some applications and it is more expensive than more simple gradient detection strategies. The nonconvex form of the flux function in the polymer model implies juxtaposed waves of different type may appear in the solution to the Riemann problem. For example, we commonly have a contact discontinuity connected to a rarefaction wave. Within the rarefaction wave the solution is smooth, and the contact is not self-sharpening. The global truncation error estimation procedure typically selects many cells for refinement in the rarefaction wave. Since the flow is smooth there, more refinement may be provided than is desired for reasonable resolution of the contact discontinuity. In industrial petroleum reservoir simulation, it can be more important to resolve the leading edge of a fluid interface and accurately predict how it will advance rather than refine portions of the flow domain over which the fronts have already passed.

In our implementation, we have the option to use either Richardson extrapolation or some user-specified gradient detection scheme. In the numerical results presented later, we use a modified version of a sharp discontinuity detector presented by Colella [80]. That is, we refine cells based on significant jumps in the flux variables. Let us illustrate the application of the gradient detection strategy in one dimension. For the polymer model, we select cell i for refinement if the following three inequalities are satisfied:

$$\frac{|c_{i+1} - c_{i-1}|}{|c_{i+2} - c_{i-2}|} \geq \alpha, \quad |c_{i+1} - c_{i-1}| > \beta \max_j |c_j|,$$

$$\max\{|s_{i+1} - s_i|, \quad |s_i - s_{i-1}|\} > \beta \max_j |s_j|.$$

The extension to more spatial dimensions is straightforward. Notice that each of the comparisons is dimensionless since it considers the relative magnitudes of the quantities involved. The first two inequalities are designed to capture sharp contact discontinuities in the polymer concentration. The last inequality accounts for significant jumps in the saturation (such as near Buckley–Leverett shocks), while limiting refinement in the rarefaction wave. We use the values $\alpha = 0.85$ and $\beta = 0.05$. Our code design is such that the criteria for mesh refinement is not part of the general AMR code; rather, it is supplied by the user. If one can identify the conditions under which she desires refinement of the mesh, then that procedure is easily incorporated into the AMR algorithm.

Finally, we address one other important mesh refinement issue. Given a particular problem, it must be decided which quantities are monitored during the regridding process. Currently, in our polymer model simulations, we consider each of the flux variables so that we are assured of refining the mesh sufficiently near contact discontinuities in the polymer concentration and Buckley–Leverett shocks in the aqueous phase saturation. For some prob-

lems, it may be appropriate to consider additional quantities such as pressure or total fluid velocity as well. However, recent analytical and numerical results suggest that, in incompressible flow problems involving a heterogeneous reservoir, the changes in pressure and velocity can be estimated in terms of suitably chosen norms of mobility perturbations [21]. To this point, we have monitored only the flux variables during the regridding process. Whether this is appropriate for more complicated models or compressible flow is the subject of ongoing work.

3.7. *Mass Conservation*

The final AMR issue we will discuss in this section is mass conservation on an adaptive mesh. We have noted previously that boundary data for fine patches are obtained from coarser patches. However, fine patches typically produce more accurate numerical results. Therefore, at times when it is appropriate for patches on different levels to be synchronized, we use information obtained on fine patches to improve data on coarser patches. The refluxing process enforcing mass conservation requires extra data to be stored during the numerical time integration process. If a level can be refined, we store the numerical flux integrals associated with centers of the space-time sides of each cell on the level. If a level is not the coarsest, we accumulate the time and space integral of the flux around the boundary of the finer patches during the timestep loop.

Recall the discrete conservative difference (Eq. (8)) that approximates mass conservation. The refluxing process for the conservation law is a three step process. First, we replace the coarse fluxes at coarse cell edges intersecting the boundaries of fine patches with the time and boundary integral of fine fluxes accumulated during integration on the finer level. That is, approximate coarse flux integrals

$$\Delta t_c^n (Fe_1)_{I+1/2,J}^{n+1/2}, \quad \Delta t_c^n (Fe_2)_{I,J+1/2}^{n+1/2}$$

are replaced by sums of approximate fine flux integrals

$$\sum_k \left\{ \sum_{(i+1/2,j)\in(I+1/2,J)} \Delta t_f^k (Fe_1)_{i+1/2,j}^{k+1/2} \right\},$$

$$\sum_k \left\{ \sum_{(i,j+1/2)\in(I,J+1/2)} \Delta t_f^k (Fe_2)_{i,j+1/2}^{k+1/2} \right\},$$

respectively. The cell edge summation is over fine cell edges intersecting coarse cell edges whose union forms fine patch boundaries. The time increment summation is over the time increments in the fine timestep loop associated with the coarse timestep Δt_c^n. Second, we repeat the conservative difference of Eq. (8) on the coarse patches that intersect fine patches. Third, we replace the mass in coarse cells underlying fine cells with a volume-weighted average of the fine cell masses:

$$m_{IJ} = \frac{\sum_{ij\in IJ} m_{ij}(\Delta x_1)_i(\Delta x_2)_j}{\sum_{ij\in IJ}(\Delta x_1)_i(\Delta x_2)_j}.$$

Here, the sum is over fine cells (i, j) contained in coarse cell (I, J). At the completion of the recursive timestepping process invoked from the coarsest mesh level, the time integration is mass conservative.

4. THE ITERATIVE SOLUTION OF THE PRESSURE EQUATION

Iteration schemes based on the multigrid method have been employed to solve boundary value problems on locally refined grids in a variety of ways [20, 61, 67]. Some studies partition the computational mesh into local subregions. Solution methods are applied to each subregion independently to precondition the linear system associated with the large globally defined problem or to achieve a fast solution method through parallelization [16–18]. Other approaches couple the grid adaptivity to the iterative solution process. Local refinement is used to increase accuracy in the approximate solution and to speed the reduction of the residual in certain subregions of the problem domain [67].

The multilevel iteration method we use is based on the full approximation storage (FAS) algorithm originally proposed by Brandt [20]. The FAS method generalizes the common linear multigrid method in several ways. In particular, the FAS algorithm is appropriate for nonlinear problems and provides a natural setting for an iterative treatment of problems approximated on a locally refined mesh. As in the multigrid method, the FAS algorithm reduces the magnitude of nonsmooth error modes, which have a short coupling range on the numerical mesh, by employing a suitable relaxation scheme. The smooth error modes, which are coupled over many mesh cells, are reduced by solving a system of equations corresponding to a related problem on a coarser mesh.

The process of employing a sequence of coarser mesh levels to solve a discrete problem associated with a fine mesh provides a complementary operation to classical iterative methods. The spectral radius of the error reduction operator associated with the Gauss–Seidel method, for example, is known to approach one rapidly as the mesh is refined and the size of the linear system increases [39, 83]. On the other hand, classical iterative methods are proficient at rapidly reducing the amplitude of high frequency error components in a linear problem with a smooth solution. In other words, these iterative methods

are good at smoothing the error, rather than eliminating it entirely. Thus, classical relaxation schemes can be used in a multigrid setting without the overall iterative solution process suffering from the poor convergence rates typical of the relaxation schemes alone. The convergence of multigrid-type iterative methods has been studied extensively; see Hackbusch [38] and McCormick [60] for the classical theory. Unfortunately, rigorous analysis has been completed only in specialized cases. However, the robustness of the multigrid approach has been demonstrated by its incorporation into the numerical solution of a wide variety of physical and mathematical problems [2, 30, 61, 67, 86].

In the current application, we are primarily concerned with capturing fluid interfaces whose time evolution is modeled by the coupling between hyperbolic mass conservation equations and an elliptic pressure equation. The role of the multilevel iteration is to solve the pressure equation on the adaptive mesh configuration determined by the location of fluid interfaces. Since mesh refinement is placed near important fluid interfaces, the pressure and velocity fields are more highly resolved near the fronts. The multilevel iteration must be able to treat a fairly general mesh configuration as well as reasonably general behavior in the system of discrete equations representing the pressure equation. The individual components of the multilevel iteration (e.g., relaxation scheme, intergrid communication operators, domain decomposition) that we employ are fairly standard and are only well suited to smooth elliptic boundary value problems. To this point, our emphasis has been on algorithm and data structure development. Sophisticated state-of-the-art methods need to be explored and incorporated so that AMR may be applied to more complicated, realistic porous media flow problems.

4.1. *Domain Decomposition*

Since the union of the patches on the level may not cover the entire computational domain, the coarser levels in the mesh hierarchy must be employed to enforce the global communication of the data described by the elliptic pressure equation. The process of constructing the solution by iterating between fine and coarse levels in the multilevel algorithm acts as a Schwarz-like domain decomposition process [53, 58]. That is, the solution on the composite mesh is generated by applying a product of linear operators to some initial guess at the solution. The classical application of the Schwarz alternating method suffers from a linear convergence rate. During the multilevel iteration, ghost cell data facilitates the residual smoothing process on finer levels, and the overlapping hierarchical mesh structure allows faster global data communication.

Recall Eq. (4) relating the total fluid velocity to the pressure gradient and gravity, and the divergence-free equation satisfied by the velocity (5). Substituting expression (4) into Eq. (5), we obtain an elliptic equation for pressure,

$$\nabla_x^\top(\mathbf{K}\lambda_\top \nabla_x p) = \nabla_x^\top[\mathbf{K}(\lambda_o\rho_o + \lambda_a\rho_a)g\,\nabla_x d]. \qquad (11)$$

For ease of notation in the following discussion, we represent the discrete weak form of this problem (and appropriate boundary conditions) as

$$\mathbf{L}u = b, \qquad (12)$$

where u corresponds to the pressure unknown.

Let Ω be the rectangular domain representing the entire reservoir problem. We aim to solve Eq. (11) on Ω with suitable boundary conditions specified on $\partial\Omega$. The iterative solution process consists of successively treating a sequence of boundary value problems associated with Eq. (11) posed on the composite mesh. Consider the overlapping subdomains of Ω represented by two consecutive mesh levels. We label the subdomains Ω_c and Ω_f, and assume $\Omega_f \subset \Omega_c$. On Ω_f, we solve $\mathbf{L}_f u_f = b_f$ which represents Eq. (12) restricted to Ω_f, subject to the boundary condition $u_f = \mathbf{P}u_c$ on $\partial\Omega_f$. The operator $\mathbf{P}$ is the prolongation operator mapping functions defined on the coarse mesh in Ω_c so that they are defined on the fine mesh in Ω_f. The coarse solution u_c satisfies $\mathbf{L}_c u_c = b_c$ on $\Omega_c\backslash\Omega_f$ and $\mathbf{L}_c u_c = \mathbf{L}\mathbf{R}u_f$ on Ω_f. The restriction operator $\mathbf{R}$ coarsens functions defined on the fine mesh corresponding to Ω_f so that they are defined on the coarse mesh corresponding to Ω_c. The boundary values for u_c on $\partial\Omega_c$ are provided by the solution to a problem posed on a subdomain of Ω containing Ω_c and corresponding to a coarser mesh level if such a level exists. During the multilevel iteration associated with AMR, each of these subdomains (corresponding to a single mesh level) is partitioned further into a collection of subdomains each of which represents the region covered by a single rectangular patch in the mesh hierarchy. Similar boundary value problems to those described above are posed on each rectangular subregion with boundary values provided by neighboring subdomains (in either Ω_f or $\Omega_c\backslash\Omega_f$) in the obvious manner.

4.2. *Multilevel Iteration*

The multilevel iteration carries out the successive treatment of the various boundary value problems described in the previous section. We present the iteration process in a recursive form in terms of two consecutive levels in the mesh hierarchy. The collection of computational cells associated with the finer mesh level is denoted as G_f, and G_c represents the mesh on the next coarser level. In reference to the discussion in Section 4.1, the regions covered

by G_f and G_c are Ω_f and Ω_c, respectively. Since $\Omega_f \subset \Omega_c$, it is appropriate to partition G_c into two subsets:

$$G_c = (G_c \backslash G_c^f) \cup G_c^f.$$

Here, G_c^f represents the collection of cells in G_c such that the interior of each cell intersects the interior of some cell in G_f. In other words, G_c^f is the set of cells in G_c that are refined.

Let $\mathcal{U}_k$ denote the set of real-valued grid functions defined on G_k representing the pressure unknowns on mesh level k, where $k \in \{f, c\}$. We assume that we have a smoothing (relaxation) operator defined on each level

$$\mathbf{S}_k : \mathcal{U}_k \times \mathcal{U}_k \to \mathcal{U}_k, \quad k \in \{f, c\}.$$

The expression

$$u_k^{(new)} = \mathbf{S}_k^{(\nu)}(u_k^{(old)}, b_k)$$

symbolizes the process of producing a new approximation $u_k^{(new)}$ to u_k (satisfying $\mathbf{L}_k u_k = b_k$) by performing ν iterations of the relaxation method, where we use the approximation $u_k^{(old)}$ as the initial iterate. We also assume that there exist linear operators that map discrete functions defined on each mesh level to the other mesh level:

$$\mathbf{P} : \mathcal{U}_c \to \mathcal{U}_f, \quad \mathbf{R} : \mathcal{U}_f \to \mathcal{U}_c.$$

Here $\mathbf{P}$ is a prolongation (refining) operator and $\mathbf{R}$ is a restriction (coarsening) operator.

Suppose that we have been provided with an initial guess at the solution on each mesh level. Denote the initial guess on the finest level as $u_f^{(0)}$. The adaptive multilevel iteration transforms $u_f^{(0)}$ by cycling through the mesh levels in the V-cycle pattern adopted from the multigrid methodology. A single V-cycle can be described in recursive pseudocode form as follows:

(0) **begin** V-cycle to solve $\mathbf{L}_f u_f = g_f$.

if (level f is the coarsest level in mesh hierarchy) **then**

(1) Solve system: $u_f = \mathbf{L}_f^{-1} g_f$.

else

(2) Perform smoothing relaxations on level f until convergence slows:

$u_f^{(1)} = S_f^{(\cdot)}(u_f^{(0)}, g_f)$.

(3) Adjust the solution on the next coarser level, $u_c = \mathbf{R} u_f^{(1)}$.

Also, set $u_c^{(0)} = u_c$.

(4) Set the right-hand side of the linear system on the next coarser level:

$$g_c = \begin{cases} b_c & \text{in } (G_c \backslash G_c^f) \\ b_c + \tau_c^f & \text{in } G_c^f \end{cases}$$

(5) Recursive call: start at step (0) on the next coarser level c.

(6) Correct the approximation on the current level:

$$u_f^{(2)} = u_f^{(1)} + \mathbf{P}(u_c - u_c^{(0)}).$$

(7) Perform post-correction smoothing relaxations until convergence slows:

$u_f = S_f^{(\cdot)}(u_f^{(2)}, g_f)$.

endif

end

Here, g_k represents the right-hand side of the linear system on level k, $k \in \{f, c\}$. The term τ_c^f appearing in the definition of g_c is motivated by the multigrid approach. We will address the details of the various parts of this iterative solution process in the sections that follow.

4.2.1. *Prolongation and Restriction.* A proper choice of prolongation and restriction operators is important to the success of the multilevel iteration. The prolongation operator transfers the error approximation from a coarse level to a finer level. Prolongation also impacts the accuracy of the fine approximation since the boundary values for the fine patches are determined by the manner in which we fill the ghost cells from a coarser level. The restriction operator also affects the correction process since it transfers the fine solution approximation to a coarser level.

In the cell-centered mesh structure that we employ, fine cells are constructed by partitioning each coarse cell. On the composite mesh, it is meaningful to transfer data between two levels only in regions where there exist cells corresponding to each level. Thus, the domain and range of $\mathbf{R}$ and $\mathbf{P}$ are limited to the cells existing on a given level. The restriction operator $\mathbf{R}$ that we use employs a cell volume-weighted average from fine cells within a coarse cell to the underlying coarse cell. The prolongation operator $\mathbf{P}$ is piecewise-bilinear interpolation. We note that the piecewise-constant restriction operator is not the adjoint of the bilinear interpolation operator. However, it is the adjoint of a piecewise-constant prolongation operator. The bilinear interpolation operator interpolates first degree polynomials exactly, while the piecewise-bilinear restriction operator interpolates constant polynomials exactly. Classical multigrid theory suggests that these characteristics are necessary for convergence of the iteration [38, 85, 86]. Although we have achieved some success with the

straightforward application of these simple mesh transfer operators, the complexity and nonsmooth nature of porous media flow equations suggests that we should consider more elaborate, physically-motivated approaches [2, 30, 38] in subsequent work.

4.2.2. *The Coarse-Mesh Correction.* The algorithmic form of the sequence of steps (3)–(6) in which the fine solution is corrected by adding a term obtained on the coarser level is similar to the correction stage in the multigrid method. The difference is that, in the multilevel iteration described above, the mesh transfer operators and the definition of the coarse mesh linear system are constructed for the locally refined mesh case. The mesh transfer operators are defined only in regions where the mesh exists on a particular level. The right-hand side of the coarse mesh linear system accounts for the possibility that fine data do not exist on the entire domain associated with the coarse mesh.

When treating the composite mesh configuration, we define the right-hand side of the coarse mesh linear system as

$$g_c = \begin{cases} b_c & \text{in } (G_c \backslash G_c^f) \\ b_c + \tau_c^f & \text{in } G_c^f, \end{cases}$$

where

$$\tau_c^f = \mathbf{R}(b_f - \mathbf{L}_f u_f) - (b_c - \mathbf{L}_c \mathbf{R} u_f).$$

That is, τ_c^f is a difference between residuals on two levels. The key to the success of the FAS form of the multigrid algorithm is that the current approximation to the solution to the discrete linear system is maintained on each mesh level. The standard linear multigrid method usually maintains only the correction to the solution on the next finer level since the fine and coarse meshes region cover the same part of the domain. In that case, the right-hand side for the coarse level is simply the fine residual coarsened to the coarse mesh.

The term τ_c^f appearing on the right-hand of the coarse-mesh equation is suitable to make the coarse-mesh solution coincide with the fine-mesh solution in the sense that the restriction operator is defined

$$u_c = \mathbf{R} u_f.$$

It is easy to see that when the fine linear system is satisfied, the coarse linear system reduces to

$$\mathbf{L}_c u_c = \mathbf{L}_c \mathbf{R} u_f.$$

Since $\mathbf{L}_c$ is a nonsingular linear operator, we have $u_c = \mathbf{R} u_f$. The term τ_c^f is referred to as the relative discretization error. It is the fine-mesh approximation to the coarse-mesh truncation error [20, 67]. Therefore, each mesh level plays two roles potentially. Aside from the finest and the coarsest levels, each level serves as both a coarser level and a finer level. In a region where a particular mesh level is the finest, the mesh associated with the level is used to approximate the actual differential equations to be solved. In a region where there exists further refinement, the mesh is a device used to compute a correction to the solution on a finer level. Indeed, the multigrid algorithm can be interpreted as a scheme for determining the τ term on each level [67].

4.3. *The Coarse-Mesh Coefficient Matrix*

Recall the discrete equations (9) and (10) that determine the approximate pressure p and total fluid velocity $\mathrm{v}_\top$. When the difference equations (9) for the components of $\mathrm{v}_\top$ are substituted into the discrete version of the divergence-free equation (10), we obtain the system of difference equations associated with the elliptic pressure equation (11). This system corresponds to the linear system $\mathbf{L}u = b$ appearing in the discussion of the multilevel iteration. The discrete differential operator on the finest level in the multilevel iteration cycle arises directly from the difference equations applied to each patch on the level. For coarser levels, the elements of the matrix corresponding to refined cells are upscaled so that coarse and fine linear systems are consistent in the sense that total fluid volume is conserved. The total fluid velocity represents the flux of the total fluid volume. So our primary concern when solving the pressure equation is conserving this flux across cell boundaries on the composite mesh to produce a discrete total fluid velocity field that is divergence-free. In the following discussion, we omit the subscript $\top$ when referring to the total fluid velocity.

Consider a coarse cell $\mathscr{R}_{IJ}$ indexed by (I, J). On the composite mesh, $\mathscr{R}_{IJ}$ is the union of four fine cells $\mathscr{R}_{ij}$ indexed by (i, j). At any point in the multilevel iteration, we would like the integral form of equation (5) to hold over each cell:

$$\int_{\partial \mathscr{R}_{IJ}} v^\top \hat{n}\, ds = \int_{\mathscr{R}_{IJ}} q dA = \sum_{\mathscr{R}_{ij} \subset \mathscr{R}_{IJ}} \int_{\mathscr{R}_{ij}} q dA = \sum_{\mathscr{R}_{ij} \subset \mathscr{R}_{IJ}} \int_{\partial \mathscr{R}_{ij}} \mathrm{v}^\top \hat{n}\, ds.$$

Here, q corresponds to nonzero source terms appearing on the right-hand side of the coarse-mesh linear system. The desired integral relationship will hold in a discrete form if we define the coarse-mesh coefficient matrix so that the discrete integral of the coarse fluid velocity normal to each coarse cell edge is equal to the sum of the discrete integrals of the fine normal velocity across the fine cell edges whose union is the coarse cell edge. For example, along the right edge of coarse cell $\mathscr{R}_{IJ}$, we want

$$(\Delta x_2)_J \mathrm{v}_{I+1/2,J} = \sum_{(i+1/2,)\in(I+1/2,J)} (\Delta x_2)_j \mathrm{v}_{i+1/2,j}$$

to hold.

The upscaling of the fluid velocities is performed in a manner consistent with the extension of the lowest order mixed finite element method described in Section 2.2. On any refinement level, the discrete integral of the velocity associated with the x_1-coordinate direction has the form

$$(\Delta x_2)_j \mathrm{v}_{i+1/2,j} = (\Delta x_2)_j [\Gamma^{(1)}_{i+1/2,j} - \mathrm{T}^{(1)}_{i+1/2,j} (\delta_x p)^{(1)}_{i+1/2,j}].$$

The Γ term represents the product of transmissibility and gravity terms at the cell edge, while the T terms represent entries in the transmissibility matrix associated with the cell edge. Thus, the gravity terms are upscaled as

$$\Gamma^{(1)}_{I+1/2,J} = \frac{1}{(\Delta x_2)_J} \left\{ \sum_{(i+1/2,j)\in(I+1/2,J)} (\Delta x_2)_j \Gamma^{(1)}_{i+1/2,j} \right\}.$$

The transmissibility terms associated with the normal direction are upscaled as

$$\mathrm{T}^{(1)}_{I+1/2,J} = \frac{(h_1)_{i+1/2}}{(h_1)_{I+1/2}} \left\{ \sum_{(i+1/2,j)\in(I+1/2,J)} \mathrm{T}^{(1)}_{i+1/2,j} \right\}.$$

Similar formulae are used for velocity components normal to the x_2-coordinate direction. For a more detailed discussion of the upscaling issues involved in the solution of the pressure equation, see [43].

4.4. *The Smoothing Operation and Stopping Criteria*

Since the multilevel iteration involves a discrete system of equations on many mesh levels potentially, we force the residuals on all levels to be small before we terminate the iteration. Our heuristic, based on discussion presented in Rüde [67], is that once an appropriately chosen norm of the residual on each level is reduced sufficiently, then a related norm of the global error, defined on the composite mesh, is below some desired tolerance. We remark that the coarsest grid used in the multilevel iteration may be much coarser than the global coarsest grid appear in the level hierarchy. Since the coarsest grid in the multilevel sequence contains very few grid points, we solve the linear system to machine precision with a conjugate-gradient iteration.

The relaxation method that we employ on each rectangular patch on each finer mesh level is the classical Gauss–Seidel method with the checkerboard ordering of the mesh points. Much is known about the smoothing properties of many standard iterative methods when the problem to be solved behaves like Poisson's equation [20, 38, 52, 86]. We have chosen the checkerboard Gauss–Seidel method since it is known (through both rigorous analysis and computational experiments) to possess suitable smoothing properties for many problems [38]. The checkerboard ordering also reduces the directional bias of the standard line-ordered Gauss–Seidel method.

Following the lead of multigrid theory and computational experience, it is most efficient to change the iteration from one level to the next at the point when the convergence rate of the relaxation process begins to decrease. Instead of using a fixed value for the number of iterations in the smoothing process on each level, we allow the current state of the solution to guide the process of cycling through the levels. In particular, we discontinue the relaxation process on a level when

$$\frac{\|\bar{r}^{(n)}\|_\infty}{\|\bar{r}^{(n-1)}\|_\infty} \geq \eta,$$

where η is a user-specified tolerance. The fraction represents the ratio between the L_∞-norms of scaled residuals after successive iterations in the relaxation process. The scaled residual $\bar{r}$ is defined to be

$$\bar{r} = \mathbf{D}^{-1}(b - \mathbf{L}u).$$

The matrix $\mathbf{D}$ represents the diagonal entries of $\mathbf{L}$. The component of the scaled residual vector corresponding to a particular mesh cell is precisely the change in the approximate solution obtained by applying one iteration of the Gauss–Seidel method at that point. When the Gauss–Seidel iterates are not changing much and the ratio of successive scaled residual norms is near one, the smoothing operation carried out by the Gauss–Seidel method is not contributing significantly to the solution process. Thus, the scaled residual is the correct quantity to monitor to avoid wasted computational expense in the multilevel process. See Rüde [67] for additional discussion regarding the role of the scaled residual in multilevel iteration methods. As in the case of the prolongation and restriction operators, the choice of the Gauss–Seidel method as a relaxation scheme was made to gain computational experience with the combination of AMR and multilevel iteration. Further analytical and computational work must be done to understand the effect that substantial variations and discontinuities in the coefficients of partial differential equations have on multilevel iterative solution methods. Recent studies suggest that the difficulties in applying multilevel iteration methods to ''interface'' problems center on the deficiencies in the prolongation and restriction operators and the relaxation schemes [2, 3, 19, 52, 54], as well as the fundamental discretization of the underlying partial differential equations [51].

5. NUMERICAL RESULTS

In this section, we present computational results illustrating some numerical properties of the methods described in previous sections. First, we discuss convergence behavior of the multilevel iteration procedure on a locally refined grid. Then, we discuss convergence behavior of a sequential calculation involving a coupled system of equations as described in Section 1.1. Finally, we compare an AMR simulation with a uniform grid simulation. We include an execution timing comparison between the adaptive computation and the fixed mesh computation. The timing comparisons demonstrate the primary benefit (i.e., computational savings) of using AMR and show where the computational effort is being expended in our calculations.

5.1. *Multilevel Iteration*

First, we demonstrate the convergence properties of the approximate solution to Poisson's equation using the multilevel iteration on a locally refined grid. On a mesh with no local refinement, the discretization is equivalent to the lowest order mixed finite element. On a mesh with local refinement, the discretization on each patch on each level is also equivalent to the mixed finite element discretization. However, the boundary data associated with each patch during the multilevel iterative solution process are determined through communication with coarser mesh patches. For a discussion of convergence results for mixed finite element methods; see [31, 34, 35, 66, 84, 87]. We show that we achieve similar results.

We approximate the pressure p and the velocities $-\partial p/\partial x$, $-\partial p/\partial y$ for the model problem

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = b(x, y) \quad \text{in } \Omega$$

$$p(x, y) = f(x, y) \quad \text{on } \partial\Omega,$$

where $b(x, y)$ and $f(x, y)$ are chosen so that the solution is

$$p(x, y) = \sin(\pi x)\sin(\pi y) + x^2 y.$$

The computational domain is $\Omega = (0, 1) \times (0, 1)$. The solution is chosen so that p has a simple analytic form with no obvious symmetries in x or y. In all computations, the multilevel iterations were continued until the residual was on the order of the machine precision on each level used by the multilevel process. The iteration process is performed on a fixed locally refined mesh configuration illustrated in Fig. 6. We also compare calculations on refinements of this grid configuration, where the coarse grid is 40 by 40 and 80 by 80.

Standard continuous L_2 error norms comparing the integral of the squared difference between the analytic solution and the finite element representation of the approximate solution yield the optimal first-order convergence reported in [35, 66]. In the interest of brevity, we present results using discrete error norms only. In Table I, discrete L_2-norms of the error arising in the quadrature formulae for the mixed finite element method [84] are used. The results illustrate that we achieve nearly the second-order superconvergence rate of the mixed method for the pressure values on the locally refined mesh. However, the overall accuracy is worse for local refinement. The primary contribution to the error arises near the coarse–fine interface indicating that our discretization process needs to be im-

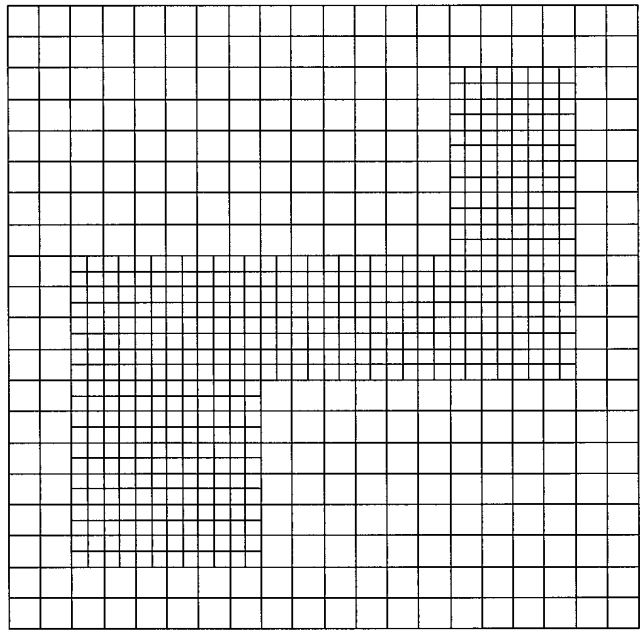

FIG. 6. The mesh configuration for the Poisson's equation test problem: coarse grid is 20 by 20, refinement ratio = 2.

TABLE I

Discrete L_2 Error Norms Computed over Entire Computational Domain

Global errors in the discrete L_2-norm			
Refinement	Coarse grid	$\|\varepsilon_p\|_M$	$\|\varepsilon_v\|_{TM}$
None	20×20	0.0001847	0.002625
	40×40	0.0000464	0.000672
	80×80	0.0000126	0.000172
Local	20×20	0.000545	0.008603
	40×40	0.000133	0.002727
	80×80	0.000035	0.000880

Note. Subscript *TM* signifies trapezoidal-midpoint quadrature, while subscript *M* indicates midpoint quadrature.

proved. Note also that the convergence rate in $\|\varepsilon_v\|_{TM}$ is significantly less than second-order for local refinement. This result was also reported for a similar problem in [31, 32, 34]. We remark that the results shown above are not generally achieved for more complicated problems involving discontinuities and anisotropies in the coefficients, problems that we are ultimately interested in.

5.2. *Sequential Solution Method and AMR*

We are primarily interested in developing a computational approach that will efficiently resolve porous media flows with interesting fluid interfaces. The problem for which we present results involves a two-dimensional vertical cross section between an injection well and a production well. The reservoir is filled initially with a 10% aqueous phase mixture containing a 10% polymer concentration. The reservoir porosity is constant, but we allow permeability variation in horizontal layers throughout the reservoir. We have also included the influence of gravity in the computation ($g\nabla_x d = [0, -5 \times 10^{-4}]^\top$ in Eq. (3)). A 100% aqueous phase mixture with a 90% polymer concentration is injected at a constant rate in the injector along the left boundary. The flow into the producer along the right boundary is determined from the specified production pressure and gravity equilibrium in the producer. The top and bottom of the reservoir are sealed; that is flow normal to those boundaries is not permitted. The numerical treatment of no flow boundary conditions is straightforward in that the normal component of the total fluid velocity is simply set to zero at each cell edge along the top and bottom of the reservoir. The well boundary conditions are discussed in Section 1.1. At the injector, we must conserve the total flow rate along the boundary. We make sure that the integral of the normal component of the flux along any portion of the injector is independent of the level of mesh one is considering. Special care is given to assure that this holds across coarse–fine mesh interfaces along the boundary as well. Vertical equilibrium at the producer requires that pressure at any point along the boundary is also independent of the level of mesh one is observing. Further details of the AMR implementation of well boundary conditions are presented in [43].

First, we consider a numerical estimation of the global error incurred in integrating such a problem on a series of uniform meshes and a series of adaptive meshes. The global error is the difference in the conserved mass computed on the uniform and adaptive meshes and a very fine uniform mesh (384 by 160) at a fixed integration time. The computed L_1 error $\|\varepsilon_m\|_1$ is summed over the components of the conserved mass vector m in Eq. (3) and summed over all computational cells in the composite mesh at the given time. The coarsest mesh used has 24 cells in the horizontal direction, 10 cells in the vertical direction. Finer uniform mesh computations are performed on uniform refinements of the coarsest mesh, where the ratio between refinement levels is 2. The adaptive mesh calculations are performed using 2, 3, and 4 levels of mesh, or 1, 2, and 3 levels of refinement of the coarsest level. The errors in Table II can then be compared by considering the refinement associated with the finest level of mesh present. We achieve nearly the same rate of convergence and magnitude of accuracy on the series of uniform meshes as on the adaptive meshes. We remark that these sorts of results are problem-dependent when using a gradient-detection refinement strategy as described in Section 3.6 (see [43]). Note also that the rate of convergence is first order at best. There are a couple of important reasons for this. First, the major contribution to the error arises near the fluid interfaces in all cases. The Godunov method is only first-order accurate at best near these discontinuities. Second, the sequential time integration method described in Section 2 is at most first-order accurate in time when the pressure and velocity field depend on the fluid composition and vice versa. Neither of these considerations necessarily eliminates the need for a high-resolution method such as the Godunov method. Lower resolution methods often underresolve complicated fluid interfaces spatially due to substantially increased numerical diffusion [41]. Recently developed sequential integration approaches for coupled equations have achieved higher order accuracy on adaptive meshes; see [6], for example. We are exploring similar enhancements for flow in porous media currently.

TABLE II

L_1 Error Norms in Conserved Mass Vector Computed over Entire Computational Domain

Global mass errors: $\|\varepsilon_m\|_1$		
	Uniform mesh	Adaptive mesh
24×10	0.01408	0.01408
48×20	0.008409	0.008865
96×40	0.004999	0.005591
192×80	0.002610	0.002946

Finally, in Figs. 7–9, we compare pictorially the uniform and adaptive mesh results for 192×80 case above. Each plot contains 25 evenly spaced contours of the aqueous phase saturation and the polymer concentration. Recall that the values of both s_a and c lie in the interval between zero and one. Thus, the maximum variation between successive contours is at most 4%. The rectangular boxes in the AMR plots correspond to the boundaries of the patches on the different levels of mesh refinement. An early stage of the simulation (0.06 pore volumes injected) is illustrated in Fig. 7. The impact of the permeability variation on the

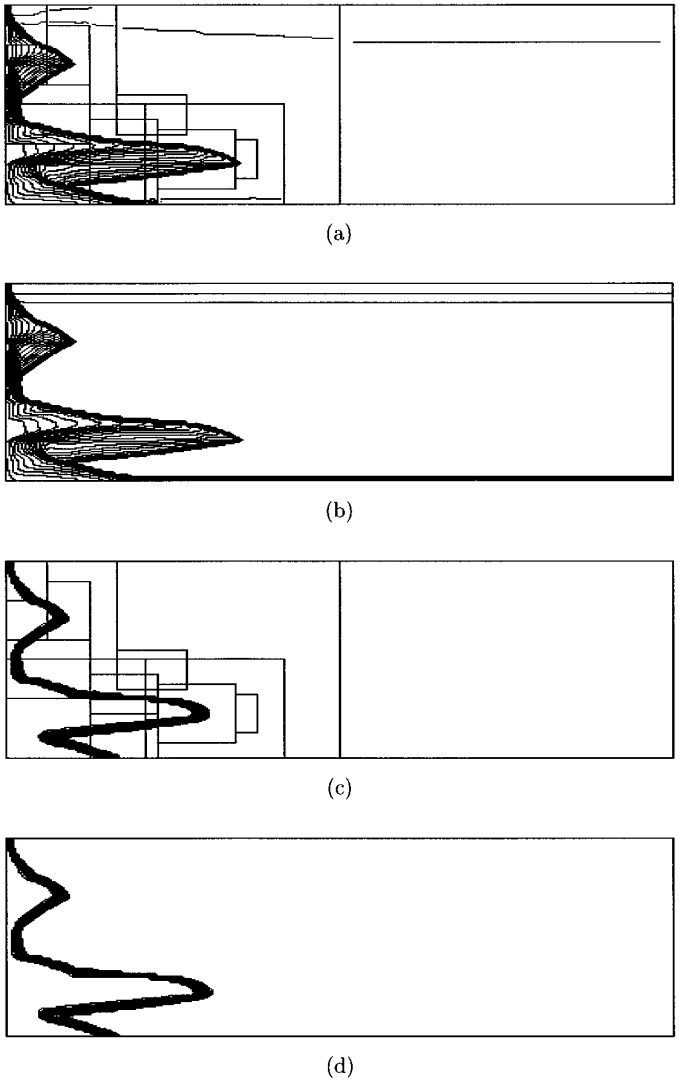

FIG. 7. Cross-section polymer flooding example at approximately 0.06 pore volumes injected. AMR results shown with uniform fine grid results for comparison. The boxes in the AMR contour plots correspond to the boundaries of the rectangular patches on the adaptive mesh. Shown are (a) aqueous phase saturation (AMR), (b) aqueous phase saturation (uniform grid), (c) polymer concentration (AMR), (d) polymer concentration (uniform grid).

formation of fluid interfaces is evident in the irregular shape of the contours. At this point, the contact discontinuity and the Buckley–Leverett shock have not separated substantially yet. In the saturation plot, we see the rarefaction wave beginning to form behind the leading edge of the front.

In Fig. 8, approximately 0.14 pore volumes have been injected. The flow has evolved to the point where the separation of the shock and contact is evident in the saturation profile. The separation is most evident below the two higher permeability layers, where gravity is causing the denser water phase to flow downward. The location of the contact discontinuity in the saturation is more clear if one mentally super-imposes the polymer contours onto the saturation plot. It is also significant to note the sharpness of the polymer contours, even in the presence of permeability heterogeneity. Also, the leading edge of the Buckley–Leverett wave in the saturation is very sharp although the front is not aligned with the mesh. This high resolution is primarily the result of the ability of the Godunov scheme to treat diagonal transport properly.

In Fig. 9, the aqueous phase has broken through to the producer. The leading shock in the saturation profile is still sharply resolved. The influence of gravity segregation

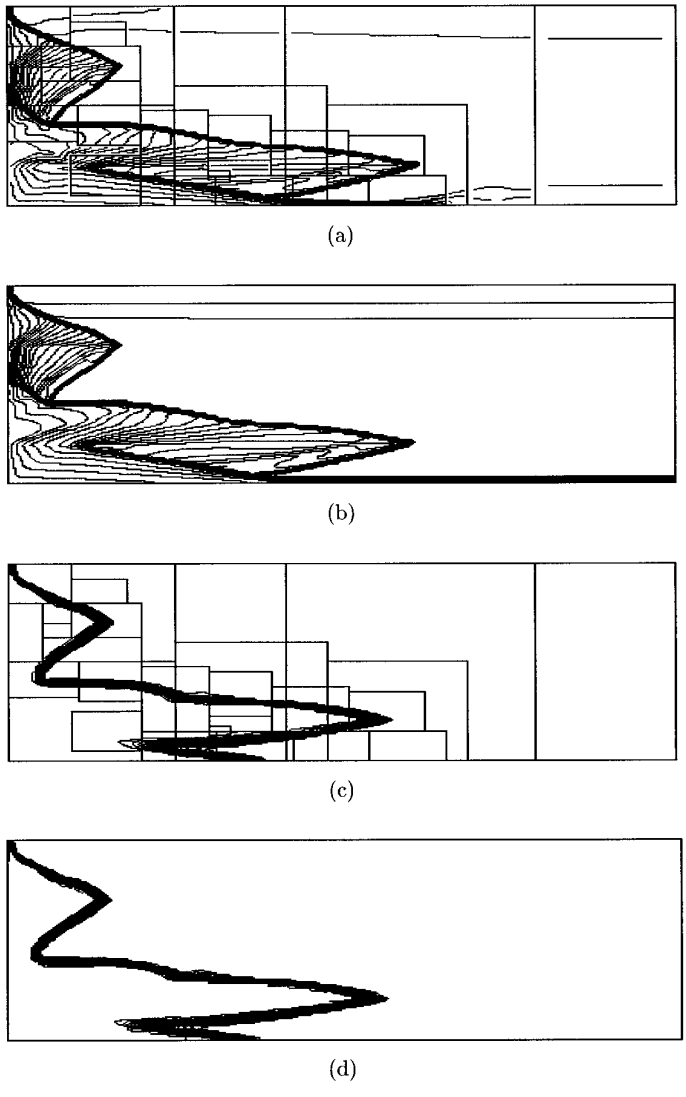

FIG. 8. Cross-section polymer flooding example at approximately 0.14 pore volumes injected. Shown are (a) aqueous phase saturation (AMR), (b) aqueous phase saturation (uniform grid), (c) polymer concentration (AMR), (d) polymer concentration (uniform grid).

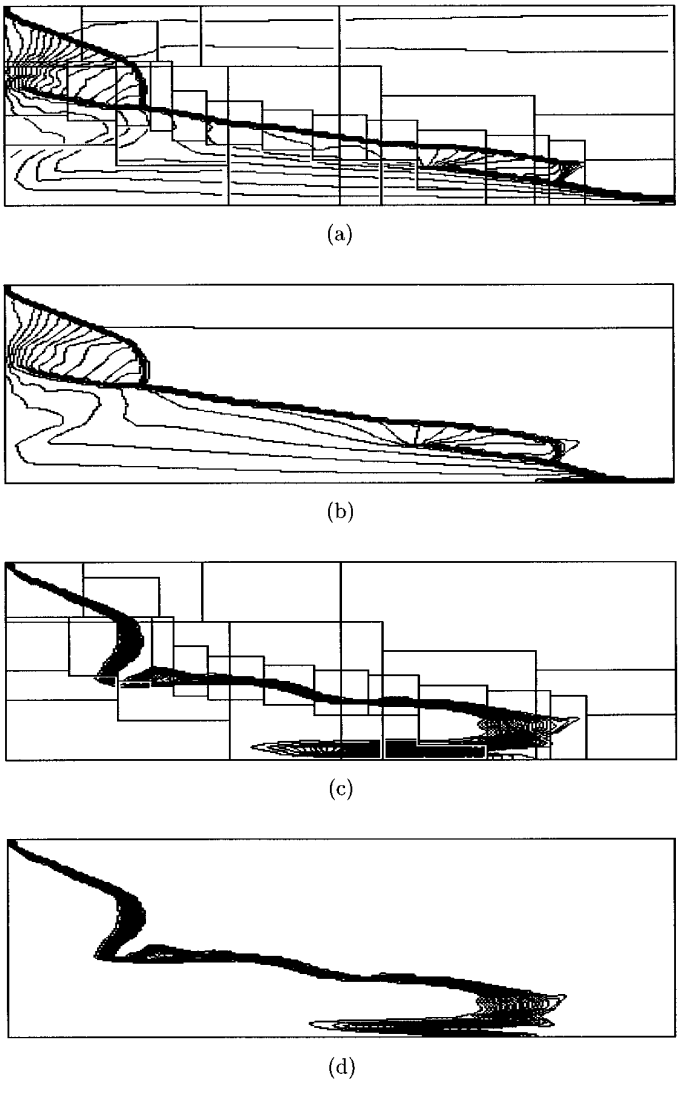

FIG. 9. Cross-section polymer flooding examples at approximately 0.30 pore volumes injected. Shown are (a) aqueous phase saturation (AMR), (b) aqueous phase saturation (uniform grid), (c) polymer concentration (AMR), (d) polymer concentration (uniform grid).

and the layered permeability variation has caused the contact to spread in the regions below the high permeability layers. Overall the fluid interfaces are still quite clear. Notice that the mesh has de-refined in the region of the rarefaction wave behind the saturation discontinuity. This mesh coarsening is the primary cause of noticeable differences in saturation contours between the uniform and adaptive mesh calculations. Essentially, information in the rarefaction profile is lost as the mesh is coarsened.

In Table III we have outlined a comparison of computational timings between the adaptive mesh refinement polymer flooding simulation and the uniform mesh calculation. The timing comparison shows some interesting results. Most prominent is that the AMR simulation ran around 4 times faster than the fine mesh calculation. Also, the simulation time associated with the pressure equation solve is much greater than that required to advance the conservation law. We can see that just over 20% of total computational time in the AMR calculation was spent on mesh adaptation and refluxing operations. This is roughly twice the overhead cost reported in AMR computations for gas dynamics [13] and solid mechanics [77]. The additional expense for flow in porous media calculations arises from the need to re-solve the pressure equation after the mesh has been reconfigured. The mesh adaptation times reported in the table include the cost of the multilevel iteration for this purpose. We also note that timings for the mesh adaptation on levels 0 and 1 include the expense of regridding finer levels. The reason for this is that at certain points in the algorithm, invocation of the mesh movement process on a level is deferred to a coarser level if the coarser level should also allow mesh movement at that time. Therefore, most of the time indicated for mesh adaptation on coarser levels was in fact spent during the recursive application of the regridding procedure on finer levels.

6. SUMMARY

We have described the extension of the adaptive mesh refinement methodology originally developed by Berger and Colella [13] to two-dimensional incompressible multiphase flow in porous media. As in the original AMR approach, we employ the high resolution Godunov method for the time integration of the hyperbolic system of mass conservation equations. The extension of the algorithm

TABLE III

Computational Time Comparison for Polymer Flooding Problem Using Adaptive and Uniform Meshes for Polymer Flood Problem

Computation time (in seconds)					
Computational task	Uniform mesh	Adaptive mesh			
		L0	L1	L2	L3
Conservation law	1283.74	5.11	42.46	333.82	1613.49
Pressure equation	53440.06	10.34	131.40	956.77	6384.62
Mesh adaptation		446.71	837.92	1195.27	
Refluxing			2.07	26.30	199.99
Time on level	54723.80	462.16	1013.85	2512.02	8198.10
Total time	54723.80	12186.13			

Note. Times are in seconds on an SGI Indigo R4000 workstation.

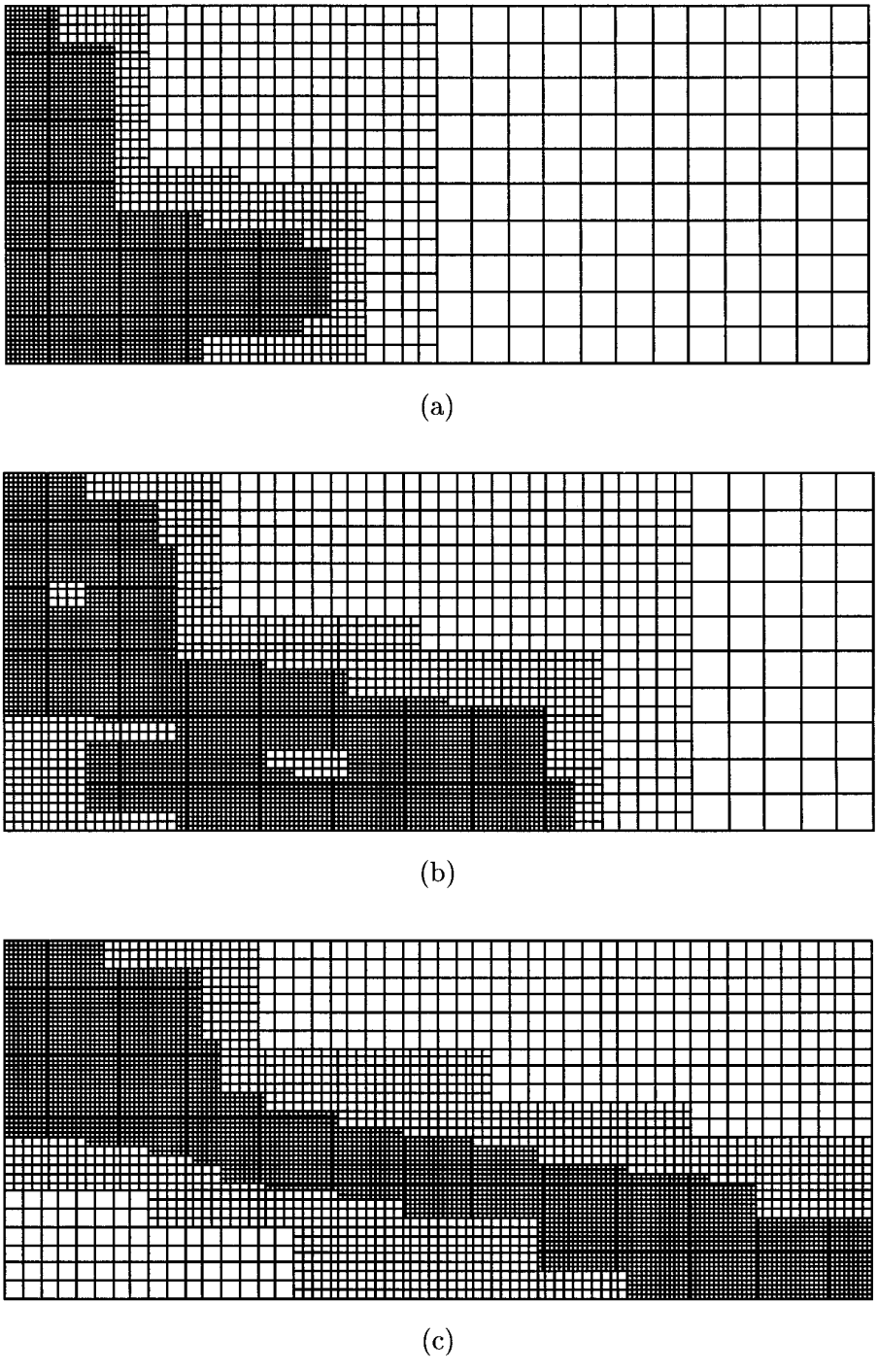

FIG. 10. Adaptive mesh configurations at (a) 0.06 pore volumes injected, (b) 0.14 pore volumes injected, (c) 0.30 pore volumes injected.

centers on the numerical treatment of the elliptic/parabolic aspects of the flow equations in the AMR setting. In particular, multilevel iteration and domain decomposition methods are introduced to solve the elliptic pressure equation. The AMR algorithm allows specialized numerical methods to treat different behavior presented by systems of equations with mixed hyperbolic/elliptic character to be combined in a consistent fashion. Other extensions to the original AMR algorithm follow the work of Trangenstein and his application of AMR to solid mechanics problems [77]. The algorithm allows significant computational savings with an overhead cost of slightly more than

20% for mesh adaptation and data synchronization. The various numerical components of the multilevel iteration presented here are not necessarily state-of-the-art. Our emphasis has been on algorithm and data structure development to provide a framework for numerical experimentation. The extension of the numerical methods and AMR algorithm to more complicated models is the subject of ongoing research.

ACKNOWLEDGMENTS

The research of the first author was sponsored by NSF Grant DMS-9407029. The second author was supported by NSF Grant DMS-9201361 and DOE Grant DE-FG05-92-ER25245, and partially supported by NSF Grant DMS-9201034. The authors also thank the reservoir engineering group at Mobil Exploration and Technical Center, Dallas, Texas, for their support.

REFERENCES

1. T. J. Abrogast and Z. Chen, On the implementation of mixed methods as nonconforming methods for second-order elliptic problems, *Math. Comp.* **64,** 943 (1995).
2. R. E. Alcouffe, A. Brandt, J. E. Dendy, and J. Painter, The multigrid method for the diffusion equation with strongly discontinuous coefficients, *SIAM J. Sci. Stat. Comp.* **2**(4), 430 (1981).
3. M. B. Allen, R. E. Ewing, and P. Lu, Well-conditioned iterative schemes for mixed finite-element models of porous media flows, *SIAM J. Sci. Stat. Comp.* **13**(3), 794 (1992).
4. M. B. Allen III, A. Behie, and J. A. Trangenstein, *Multi-Phase Flow in Porous Media: Mechanics, Mathematics and Numerics,* Springer-Verlag, New York/Berlin, 1988. [*Lecture Notes in Engineering,* Vol. 34].
5. A. S. Almgren, J. B. Bell, P. Colella, and L. H. Howell, An adaptive projection method for the incompressible Euler equations, in *Proceedings, 11th AIAA Computational Fluid Dynamics Conference, 1993.*
6. A. S. Almgren, J. B. Bell, P. Colella, L. H. Howell, and M. L. Welcome, *A Conservative Adaptive Projection Method for the Variable Density Incompressible Navier–Stokes Equations,* Technical Report LBNL-39075, Lawrence Berkeley National Laboratory, 1996.
7. D. N. Arnold and F. Brezzi, Mixed and nonconforming finite element methods: Implementation, postprocessing, and error estimates, *RAIRO Modél. Math. Anal. Numér.* **19,** 7 (1985).
8. K. Aziz and A. Settari, *Petroleum Reservoir Simulation,* Appl. Sci., Braking, Essex, 1979.
9. J. Bear, *Dynamics of Fluids in Porous Media.* Dover, New York, 1988.
10. J. B. Bell, P. Colella, and J. A. Trangenstein, Higher-order Godunov methods for general systems of hyperbolic conservation laws, *J. Comp. Phys.* **82,** 362 (1989).
11. J. B. Bell, C. N. Dawson, and G. R. Shubin, An unsplit, higher order Godunov method for scalar conservation laws in multiple dimensions, *J. Comp. Phys.* **74,** 1 (1988).
12. J. B. Bell, G. R. Shubin, and J. A. Trangenstein. A method for reducing numerical dispersion in two-phase black-oil reservoir simulation, *J. Comp. Phys.* **65,** 71 (1986).
13. M. J. Berger and P. Colella, Local adaptive mesh refinement for shock hydrodynamics, *J. Comput. Phys.* **82,** 64 (1989).
14. M. J. Berger and J. S. Saltzman, Amr on the cm-2, *Appl. Numer. Math.* **14,** 239 (1994).
15. M. J. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations, *J. Comp. Phys.* **53,** 484 (1984).
16. J. H. Bramble, J. E. Pasciak, and A. H. Schatz, An iterative method for elliptic problems on regions partitioned into substructures, *Math. Comp.* **46**(174), 361 (1986).
17. J. H. Bramble, J. E. Pasciak, and A. H. Schatz, The construction of preconditioners for elliptic problems by substructuring, ii, *Math. Comp.* **49**(179), 1 (1987).
18. J. H. Bramble, J. E. Pasciak, and A. H. Schatz, The construction of preconditioners for elliptic problems by substructuring, iii, *Math. Comp.* **51**(184), 415 (1988).
19. J. H. Bramble, J. E. Pasciak, J. Wang, and J. Xu, Convergence estimates for multigrid algorithms without regularity assumptions, *Math. Comp.* **57**(195), 23 (1991).
20. A. Brandt, Multi-level adaptive solutions to boundary-value problems, *Math. Comp.* **31**(138), 333 (1977).
21. A. M. Bruaset and B. F. Nielsen, On the stability of pressure and velocity computations for heterogeneous reservoirs. Technical report, SINTEF Applied Mathematics, Oslo, 1994. Submitted to SIAM J. Appl. Math.
22. P. Colella, Multidimensional unsplit methods for hyperbolic conservation laws, *J. Comp. Phys.* **87,** 171 (1990).
23. W. Y. Crutchfield, Parallel adaptive mesh refinement: An example of parallel data encapsulation. LLNL report UCRL-JC-107680, 1991.
24. H. K. Dahle, *Adaptive Characteristic Operator Splitting Techniques for Convection-Dominated Diffusion Problems in One and Two Space Dimensions,* Ph.D. thesis, University of Bergen, 1988.
25. H. K. Dahle, M. S. Espedal, R. E. Ewing, and O. Sævereid, Characteristic adaptive subdomain methods for reservoir flow problems, *Numer. Methods Partial Diff. Equations* **6,** 279 (1990).
26. H. K. Dahle, M. S. Espedal, and O. Sævereid, Characteristic, local grid refinement techniques for reservoir flow problems, *Inter. J. Numer. Engrg.* **34,** 1051 (1992).
27. K. D. Devine and J. E. Flaherty, Parallel adaptive hp-refinement techniques for conservation laws, *Appl. Numer. Maths.* **16,** 1 (1996).
28. J. Douglas, R. E. Ewing, and M. F. Wheeler, The approximation of the pressure by a mixed method in the simulation of miscible displacement, *RAIRO Anal. Num.* **17**(1), 17 (1983).
29. M. G. Edwards and M. A. Christie, Dynamically adaptive Godunov schemes with renormalization in reservoir simulation, in *Proceedings of the 12th SPE Symposium on Reservoir Simulation,* 1993. [SPE 25268]
30. M. G. Edwards and C. F. Rogers, Multigrid and renormalization for reservoir simulation, in *Proceedings, 4th European Multigrid Conference, Amsterdam, 1993,* edited by P. W. Hemker and P. Wesseling, p. 190.
31. R. E. Ewing, R. D. Lazarov, T. F. Russell, and P. S. Vassilevski, Local refinement via domain decomposition techniques for mixed finite element methods with rectangular Raviart–Thomas elements, in *Third International Symposium on Domain Decomposition for Partial Differential Equation,* edited by T. F. Chan, *et al.,* SIAM, Philadelphia, 1989.
32. R. E. Ewing, R. D. Lazarov, and P. S. Vassilevski, Local refinement techniques for elliptic problems on cell-centered grids i: Error analysis, *Math. Comp.* **56**(194), 437 (1991).
33. R. E. Ewing, R. D. Lazarov, and J. Wang, Superconvergence of the velocity along the gauss lines in mixed finite element methods, *SIAM J. Num. Anal.* **28**(4), 1015 (1991).
34. R. E. Ewing and J. Wang, Analysis of mixed finite element methods on locally refined grids, *Numer. Math.* **63,** 183 (1992).

35. R. S. Falk and J. E. Osborn, Error estimates for mixed methods, *R.A.I.R.O. Analyse Numerique* **14**(3), 249 (1980).
36. K. Furati, The Riemann problem for polymer flooding with hysteresis, in *Proceedings of the Fifth International Conference on Hyperbolic Problems: Theory, Numerics, and Applications, 1994*.
37. S. K. Godunov, Difference methods for the numerical calculation of the equations of fluid dynamics, *Mat. Sb.* **47,** 271 (1959).
38. W. Hackbusch, *Multi-Grid Methods and Applications,* Springer-Verlag, New York/Berlin, 1985. [Comput. Math., Vol. 4]
39. L. A. Hageman and D. M. Young, *Applied Iterative Methods.* Computer Science and Applied Mathematics, Academic Press, 1981.
40. K. Holing, *A Conservative Front-Tracking Method for Two-Dimension Polymer Flooding,* PhD thesis, Norges Tekniske Høgskole Trondheim, 1990.
41. K. Holing, J. Alfestad, and J. A. Trangenstein, The use of second-order Godunov-type methods for simulating eor processes in realistic reservoir models, in unknown, editor, *Third European Conference on the Mathematics of Oil Recovery,* unknown, 1990.
42. R. D. Hornung, *Adaptive Mesh Refinement and Multi-level Iteration Techniques,* PhD thesis, Department of Mathematics, Duke University, 1994.
43. R. D. Hornung, S. A. Khan, and J. A. Trangenstein, Adaptive mesh refinement and upscaling for multicomponent flow in porous media, Submitted to *Comput. Geosci.* (1996).
44. E. Isaacson, D. Marchesin, and B. Plohr, The structure of the Riemann solution for nonstrictly hyperbolic conservation laws, in *Proceedings of the Second International Conference on Hyperbolic Problems, Aachen, 1988,* p. 269.
45. E. Isaacson, D. Marchesin, B. Plohr, and J. B. Temple, The classification of solutions of quadratic Riemann problems I, *SIAM J. Appl. Math.* **48,** 1009 (1988).
46. E. L. Isaacson, Global solution of a Riemann problem for a nonstrictly hyperbolic system of conservation laws arising in enhanced oil recovery, Rockefeller University preprint.
47. T. Johansen, *Riemann Problems for Hyperbolic Systems of Conservation Laws Modeling Multicomponent, Two-Phase Flow Through Porous Media,* PhD thesis, University of Oslo, 1992.
48. T. Johansen, A. Tveito, and R. Winther, A Riemann solver for two-phase multicomponent process, *SIAM J. Sci. Stat. Comput.* **10,** 846 (1989).
49. T. Johansen and R. Winther, The solution of the Riemann problem for a hyperbolic system of conservation laws modeling polymer flooding, *SIAM J. Math. Anal.* **19,** 541 (1988).
50. T. Johansen and R. Winther, The Riemann problem for multicomponent polymer flooding, *SIAM J. Math. Anal.* **20,** 909 (1989).
51. J. E. Jones, *A Mixed Finite Volume Element Method for Accurate Computation of Fluid Velocities in Porous Media,* PhD thesis, University of Colorado at Denver, 1995.
52. R. Kettler, Analysis and comparison of relaxation schemes in robust multigrid and preconditioned conjugate gradient methods, in *Proceedings of the Conference on Multigrid Methods,* Springer-Verlag, 1982. [*Lecture Notes in Mathematics,* Vol. 960, p. 502.]
53. D. E. Keyes, Domain decomposition: A bridge between nature and parallel computers, in unknown, editor, *Proceedings of the Symposium on Adaptive, Multilevel and Hierarchical Computational Strategies,* ASME, 1992. ASME Winter Annual Meeting, 1992.
54. M. Khalil and P. Wesseling, Vertex-centered and cell-centered multigrid for interface problems, *J. Comp. Phys.,* p. 1 (1992).
55. S. A. Khan, J. A. Trangenstein, R. D. Hornung, K. Holing, and B. E. R. Schilling, Application of adaptive mesh refinement with a new higher-order method in simulation of a north sea micellar/polymer flood, in *Proceedings of the 13th SPE Symposium on Reservoir Simulation, 1995.* SPE 29145.
56. L. W. Lake, *Enhanced Oil Recovery,* Prentice–Hall, Englewood Cliffs, NJ, 1989.
57. R. J. LeVeque, *Numerical Methods for Conservation Laws,* Birkhäuser, Basel, 1990. [*Lectures in Mathematics,* ETH Zürich]
58. P. L. Lions, On the schwarz alternating method, in R. Glowinski, G. E. Golub, G. A. Meurant, and J. Périaux, editors, *First International Symposium on Domain Decomposition Methods for Partial Differential Equations,* SIAM, 1988. Paris, France.
59. C. M. Marle, *Multiphase Flow in Porous Media,* Gulf Pub., Houston, 1981.
60. S. F. McCormick, *Multigrid Methods,* Vol. 3 of *Frontiers in Applied Mathematics,* SIAM, 1987.
61. S. F. McCormick, *Multilevel Adaptive Methods for Partial Differential Equations,* SIAM, Philadelphia, 1989. [*Frontiers in Applied Mathematics,* Vol. 6].
62. M. Minion, *Two Methods for the Study of Vortex Patch Evolution on Locally-Refined Grids,* Ph.D. thesis, University of California at Berkeley, 1994.
63. S. Osher, Convergence of generalized MUSCL schemes, *SIAM J. Num. Anal.,* **22,** 947 (1985).
64. D. W. Peaceman, *Fundamentals of Numerical Reservoir Simulation,* Elsevier Scientific, Amsterdam, 1977.
65. G. A. Pope, The application of fractional flow theory to enhanced oil recovery, *Society of Petroleum Engineers J.* **20,** 191 (1980).
66. P. A. Raviart and J. M. Thomas, A mixed finite element method for 2nd order elliptic problems, in *Mathematical Aspects of Finite Element Methods,* edited by I. Galligani and E. Magenes, Springer-Verlag, 1977. [*Lecture Notes in Mathematics,* Vol. 606, p. 292].
67. U. Rüde, *Mathematical and Computational Techniques for Multilevel Adaptive Methods,* volume 13 of *Frontiers in Applied Mathematics,* SIAM, 1993.
68. V. V. Rusanov, The calculation of the interaction of non-stationary shock waves with barriers, *J. Comp. Math. Math. Phys. USSR,* **1,** 267 (1961).
69. T. F. Russell and M. F. Wheeler, Finite element and finite difference methods for continuous flows in porous media, in *The Mathematics of Reservoir Simulation,* edited by R. E. Ewing, chap. 2, p. 35, Society for Industrial and Applied Mathematics, 1983.
70. P. H. Sammon, An analysis of upstream differencing, *Soc. Pet. Eng. J. Res. Engrg.* **3,** 1053 (1988).
71. G. H. Schmidt and F. J. Jacobs, Adaptive local grid refinement and multi-grid in numerical reservoir simulation, *J. Comp. Phys.* **77,** 140 (1988).
72. C. W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock capturing schemes ii. *J. Comp. Phys.* **83,** 32 (1989).
73. G. R. Shubin and J. B. Bell, An analysis of the grid orientation effect in numerical simulation of miscible displacement, *Comput. Meth. in Appl. Mech. and Engr.* **47,** 47 (1984).
74. B. Stroustrup, *The C++ Programming Language,* Addison-Wesley, 1992.
75. P. K. Sweby, High resolution schemes using flux limiters for hyperbolic conservation laws, *SIAM J. Num. Anal.* **21**(5), 995 (1984).
76. J. A. Trangenstein, A second-order Godunov algorithm for two-dimensional solid mechanics, *Comput. Mech.* **13,** 343 (1994).
77. J. A. Trangenstein, Adaptive mesh refinement for wave propagation in nonlinear solids, *SIAM J. Sci. Stat. Comput.* **16**(4) 819 (1995).

78. J. A. Trangenstein and J. B. Bell, Mathematical structure of compositional reservoir simulation, *SIAM J. Sci. Stat. Comput.* **10,** 817 (1989).
79. J. A. Trangenstein and J. B. Bell, Mathematical structure of the black-oil model for petroleum reservoir simulation, *SIAM J. Appl. Math.* **49,** 749 (1989).
80. J. A. Trangenstein and R. B. Pember, Numerical algorithms for strong discontinuities in elastic-plastic solids, *J. Comp. Phys.* **103,** 63 (1992).
81. A. Tveito and R. Winther, The solution of nonstrictly hyperbolic conservation laws may be hard to compute, *SIAM J. Sci. Stat. Comput.* **16,** 320 (1995).
82. B. van Leer, Towards the ultimate conservative difference scheme. V. a second-order sequel to Godunov's method, *J. Comp. Phys.* **32,** 101 (1979).
83. R. S. Varga, *Matrix Iterative Analysis,* Prentice-Hall, 1962.
84. A. Weiser and M. F. Wheeler, On the convergence of block-centered finite differences for elliptic problems, *SIAM J. Num. Anal.* **25**(2), 351 (1988).
85. P. Wesseling, Cell-centered multigrid for interface problems, *J. Comp. Phys.* **79,** 85 (1988).
86. P. Wesseling, *An Introduction to Multigrid Methods,* John Wiley and Sons Ltd., 1992.
87. I. Yotov, *Mixed Finite Element Methods for Flow in Porous Media,* Ph.D. thesis, The University of Texas at Austin, 1996.