A FAST LEVEL SET METHOD FOR RESERVOIR SIMULATION

K. HVISTENDAHL KARLSEN, K.–A. LIE, AND N. H. RISEBRO

Abstract. We present a level set method for reservoir simulation based on a fractional flow formulation of two-phase, incompressible, immiscible flow in two or three space dimensions. The method uses a fast marching level set approach and is therefore considerably faster than conventional finite difference methods. The level set approach compares favourably with a front tracking method what regards both efficiency and accuracy, but maintains the advantage of being able to handle changing topologies of the front structure.

1. Introduction

The objective of oil reservoir simulation is to understand complex fluid flow processes in a reservoir and to optimize the recovery of hydrocarbons. In other words, one must be able to match production history and predict the flow pattern under various enhanced oil recovery strategies, e.g., water flooding, polymer flooding, thermal flooding, etc. To this end, accurate numerical simulation of appropriate mathematical models is a crucial task. Mathematical flow models typically consist of a strongly coupled system of nonlinear partial differential equations [3, 7, 28].

One such model for two-phase, incompressible, immiscible flow will be considered in this paper. In this model the basic unknowns are a fluid pressure and the saturation of the non-wetting phase. The fluid pressure is described by an elliptic equation and the saturation by a convection-diffusion equation. The equations are coupled through the total Darcy velocity. Enhanced recovery displacement processes are dominated by convective flow from injection to production wells and therefore mathematical models must have strong transport terms. Consequently, it is reasonable in many situations to neglect capillary forces to obtain a first-order hyperbolic equation for the saturation variable. A common strategy for solving such models is to decouple the equations, that is, first solve the pressure equation to generate a velocity field. Next, the velocity field is held fixed and the saturation is advanced forward a small time step. Then the pressure is recalculated, and so on. In this way, one can devise efficient numerical strategies that exploit the different mathematical properties of the model, thus taking properly care of the completely different nature of the equations in the system.

Due to the nonlinearity inherent in the saturation equation, a sharp fluid interface will arise between the injected fluid (water) and the resident fluid (oil). When the saturation is described by a hyperbolic equation, the interface will be a discontinuous shock front that develops even for smooth initial data. An important aspect of numerical simulations is to resolve the location and structure of the sharp fluid interface. The location of this interface indicates how much of and where the oil is left in the reservoir as a function of time. Knowledge of front location is crucial for determining infield drilling and new production strategies with the purpose of optimizing the oil recovery. In recent years, a variety of sophisticated numerical methods have been proposed which all have in common the ability to accurately represent such fronts, both for hyperbolic models and for more complex models involving nonlinear diffusion. We refer the reader to, e.g., [12, 22, 25] for a general introduction of modern numerical methods for nonlinear partial differential equations possessing solutions with large gradients (shocks).

For many of the numerical methods for hyperbolic equations currently in use, including some of the methods mentioned above, a sophisticated one-dimensional solver constitutes the core of the overall numerical method and extensions to several space dimensions is carried out by means of dimensional splitting. In most cases dimensional splitting works very well and it is well known that numerical methods based on dimensional splitting are very efficient, especially those based on one-dimensional, large time step solvers, see, e.g., [4, 5, 19, 24]. However, in some situations inaccuracies are introduced at shock fronts propagating obliquely to the splitting directions. Furthermore, for unstable displacements such instabilities can be magnified by the decoupling of the pressure and saturation equations and grow uncontrollably with time [16]. It is therefore natural to search for alternative ways to treat the multidimensional case. One particular approach that has received a lot of (renewed) attention in the petroleum community lately is the streamline method, which is well suited for problems without gravity. Equipped with one's favourite one-dimensional solver, this approach is based on integrating the saturation equation along the streamlines defined by the velocity field, see, e.g., [5, 6, 20] (and the references cited therein), thus avoiding the use of dimensional splitting. When gravity is present, the streamline method can be used as part of an operator splitting strategy, where the effect of gravitation is solved separately, see, e.g., [6].

In this paper we present a new numerical method for simulating two-phase flow in oil reservoirs. Our method is inspired by the level set idea of Osher and Sethian [27] (see Section 2 for more details) and is especially well suited to keep track of the front location in, e.g., a water flooding scenario. The method works in any number of dimensions, handles changing topologies of the front structure naturally, and is easy to program. We demonstrate that the level set approach compares favourably with a (large time step) front tracking method with respect to computational efficiency and accuracy. Our approach is to employ a sequential time stepping procedure to separate the elliptic pressure equation and the hyperbolic saturation equation. We then attack the saturation equation with a level set type approach; that is, we reformulate the saturation equation as a boundary value problem for a stationary Eikonal equation. The Eikonal equation is then solved by numerical methods based on the fast marching approach suggested by Sethian [29, 31].

The level set method proposed here can be viewed as a sort of streamline method. Streamline methods and the level set method are both based on one-dimensional solutions along streamlines (or approximate streamlines) of the total velocity field. The level set method is however much simpler to implement and consequently more robust.

Although different from our approach, we mention that Aslam [2] recently has proposed a level set algorithm for tracking discontinuities in hyperbolic conservation laws.

The outline of the paper is as follows. In Section 2 we briefly describe the original level set idea [27]. The reservoir model is described in Section 3 and our novel level set method is introduced in Section 4. The method is investigated numerically in Section 5 for several standard test problems. Moreover, the the efficiency and accuracy of our method is compared with a front tracking method based on dimensional splitting. Finally, we make some concluding remarks in Section 6.

2. The original level set approach

For completeness, we now describe the original level set method of Osher and Sethian [27] for tracking the evolution of an initial front Γ_0 as it propagates in a direction normal to itself with a given speed function F. The main idea is to match the one-parameter family of fronts $\{\Gamma_t\}_{t\geq 0}$, where Γ_t is the position of the front at time t, with a one-parameter family of moving surfaces in such a way that the zero level set of the surface always yields the moving front. To determine the front propagation, we then need to find and solve a partial differential equation for the motion of the evolving surface. To be more precise, let Γ_0 be an initial front in $\mathbb{R}^d$, $d \geq 2$ and assume that the so-called *level set function* $u : \mathbb{R}^d \times \mathbb{R}_+ \to \mathbb{R}$ is such that at time $t \geq 0$ the zero level set of $u(x,t)$ is the front Γ_t. We further assume that

$$u(x,0) = \pm d(x),$$

where $d(x)$ is the distance from x to the curve Γ_0. We use plus sign if x is inside γ_0 and minus if x is outside. Let each level set of u flow along its gradient field with speed F. This speed function should match the desired speed function for the zero level set of u. Now consider the motion of, e.g., the level set

$$\Big\{x \in \mathbb{R}^d : u(x,t) = 0\Big\}.$$

Let $x(t)$ be trajectory of a particle located at this level set so that

$$u(x(t),t) = 0.$$

The particle speed $\frac{\partial x}{\partial t}$ in the direction n normal to the level set is given by the speed function F, and hence

$$\frac{\partial x}{\partial t} \cdot n = F,$$

where the normal vector n is given by

$$n = -\frac{\nabla u}{|\nabla u|}.$$

This is a vector pointing outwards, giving our initialization of u. By the chain rule

$$\frac{\partial u}{\partial t} + \frac{\partial x}{\partial t} \cdot \nabla u = 0.$$

Therefore $u(x,t)$ satisfies the partial differential equation (the *level set equation*)

$$\frac{\partial u}{\partial t} - F|\nabla u| = 0, \tag{1}$$

and the initial condition

$$u(x,t=0) = \pm d(x).$$

This is called an Eulerian formulation of the front propagation problem because it is written in terms of a fixed coordinate system in the physical domain.

Summing up, the central mathematical idea is to view the moving front Γ_t as the zero level set of the higher-dimensional level set function $u(x,t)$. Depending on the form of the speed function F, the propagation of the level set function $u(x,t)$ is described by the initial value problem for a nonlinear Hamilton–Jacobi type partial differential equation (1) of first or second order [27, 30]. Because of the nonlinear nature of the governing partial differential equation (1), solutions are not smooth enough to satisfy this equation in the classical sense (the level set function is typically only Lipschitz). Furthermore, generalized solutions, i.e., Lipschitz continuous functions satisfying the equations almost everywhere, are not uniquely determined by their data and additional selection criteria (entropy conditions) are needed to pick out the (physically) correct generalized solutions. The correct framework for treating Hamilton-Jacobi type equations is provided by the notion of viscosity solutions [10, 9].

After its introduction, the level set approach has been successfully applied to a wide collection of problems that arise in geometry, fluid mechanics, computer vision, and manufacturing processes, see [30] for details. Numerous advances have been made to the original technique, including the adaptive narrow band methodology [1] and the fast marching method for solving the static Eikonal equation [29, 31]. For further details and summaries of level set techniques for numerical purposes, see [30, 31].

The mathematical theory of the level set approach, which is based on the theory of viscosity solutions [10, 9], was extensively developed independently by Evans and Spruck [13] for the motion by mean curvature and by Chen, Giga, and Goto [8] for more general geometric motions. Various generalizations were subsequently obtained by several authors, see the lecture notes [32] for an overview.

3. The reservoir flow model

We start our discussion by deriving the equations for a black oil reservoir model, containing two immiscible phases, denoted by $\mathbf{n}$ (non-wetting) and $\mathbf{w}$ (wetting). A more general formulation is given in, e.g., [3, 7, 28].

In the following all quantities are assumed to be functions of the spatial location $\mathbf{x}$, and some also of the time t, and ∇ denotes the gradient operator with respect to the spatial variables.

The velocity of each phase is assumed to obey the experimentally verified Darcy's law

$$v_i = -\lambda_i \left(\nabla P_i - \rho_i g \nabla D\right), \tag{2}$$

where the mobility of phase i is defined as

$$\lambda_i = K \frac{k_i}{\mu_i}.$$

Here, K denotes the absolute permeability (tensor) of the rock, k_i denotes the relative permeability of phase i, and μ_i the viscosity of phase i. Furthermore, ρ_i denotes the density of phase i, g the gravitational acceleration, and D measures vertical distance in the reservoir. The index i in (2) and subsequent equations is $\mathbf{n}$ and $\mathbf{w}$. Hereafter, we will ignore the capillary pressure and assume that $P = P_{\mathbf{w}} = P_{\mathbf{n}}$.

Conservation of mass for each phase now reads

$$-\nabla\left(\alpha \rho_i v_i\right) + \alpha q_i = \alpha \frac{\partial}{\partial t}\left(\phi \rho_i S_i\right), \tag{3}$$

where α denotes the cross section of the reservoir if the dimension is 1 or 2, and $\alpha = 1$ if we consider a three-dimensional model. The porosity, i.e., the available pore volume, is denoted by ϕ, and S_i denotes the saturation of phase i. The term q_i denotes sources or sinks present in the reservoir. The saturation of phase i is defined to be the percentage of the available pore volume occupied by this phase. Hence

$$S_{\mathbf{n}} + S_{\mathbf{w}} = 1. \tag{4}$$

Carrying out the differentiation in (3) yields

$$-\nabla\left(\alpha\rho_{\mathbf{n}}v_{\mathbf{n}}\right)+\alpha q_{\mathbf{n}}=\alpha\left(\rho_{\mathbf{n}}S_{\mathbf{n}}\frac{\partial\phi}{\partial t}+\phi S_{\mathbf{n}}\frac{\partial\rho_{\mathbf{n}}}{\partial t}+\phi\rho_{\mathbf{n}}\frac{\partial S_{\mathbf{n}}}{\partial t}\right) \tag{5}$$

and similarly for the wetting phase. If we divide the two equations by $\alpha\rho_i$ for $i=\mathbf{n}$, $\mathbf{w}$, and add the results using (4), we eliminate the saturations and are left with

$$-\frac{1}{\alpha\rho_{\mathbf{n}}}\nabla\left(\alpha\rho_{\mathbf{n}}v_{\mathbf{n}}\right)-\frac{1}{\alpha\rho_{\mathbf{w}}}\nabla\left(\alpha\rho_{\mathbf{w}}v_{\mathbf{w}}\right)+Q_T=\frac{\partial\phi}{\partial t}+\left(S_{\mathbf{n}}c_{\mathbf{n}}+S_{\mathbf{w}}c_{\mathbf{w}}\right)\phi\frac{\partial P}{\partial t}. \tag{6}$$

Here, $Q_T=q_{\mathbf{n}}/\rho_{\mathbf{n}}+q_{\mathbf{w}}/\rho_{\mathbf{w}}$ is the total volumetric injection or production rate, and the phase compressibilities c_i are given by

$$c_i=\frac{1}{\rho_i}\frac{d\rho_i}{dP}.$$

We now introduce

$$C_T=\frac{1}{\phi}\frac{d\phi}{dP}+S_{\mathbf{n}}c_{\mathbf{n}}+S_{\mathbf{w}}c_{\mathbf{w}}$$

and define the *total velocity* v_T by

$$v_T=v_{\mathbf{n}}+v_{\mathbf{w}}.$$

Using this notation (6) reads

$$-\nabla\left(\alpha v_T\right)+\alpha Q_T=\alpha\phi C_T\frac{\partial P}{\partial t}+\alpha\left(v_{\mathbf{n}}c_{\mathbf{n}}+v_{\mathbf{w}}c_{\mathbf{w}}\right)\nabla P. \tag{7}$$

Using Darcy's law (2) and the last equations we find

$$\nabla\left(\alpha\lambda_T\nabla P\right)+\alpha Q_T=\alpha\phi C_T\frac{\partial P}{\partial t}+\alpha\left(v_{\mathbf{n}}c_{\mathbf{n}}+v_{\mathbf{w}}c_{\mathbf{w}}\right)\nabla P+\nabla\left[\alpha\left(\lambda_{\mathbf{n}}\rho_{\mathbf{n}}+\lambda_{\mathbf{w}}\rho_{\mathbf{w}}\right)g\nabla D\right], \tag{8}$$

which is called the *pressure equation.* Here, we have introduced the total mobility $\lambda_T=\lambda_{\mathbf{n}}+\lambda_{\mathbf{w}}$. In this paper, we will concentrate on the incompressible flow, i.e., the densities and the porosities are assumed to be independent of the pressure. This assumption reduces the pressure equation considerably

$$\nabla\left(\alpha\lambda_T\nabla P\right)+\alpha Q_T=\nabla\left[\alpha\left(\lambda_{\mathbf{n}}\rho_{\mathbf{n}}+\lambda_{\mathbf{w}}\rho_{\mathbf{w}}\right)g\nabla D\right]. \tag{9}$$

Note that in this case the divergence of α times the total velocity is zero away from sources or sinks, i.e.,

$$\nabla(\alpha v_T)=0.$$

Adding and subtracting the two equations of (2) we find that

$$v_{\mathbf{n}}=f_{\mathbf{n}}\left(v_T+\lambda_{\mathbf{w}}\left(\rho_{\mathbf{n}}-\rho_{\mathbf{w}}\right)g\nabla D\right),$$
$$v_{\mathbf{w}}=f_{\mathbf{w}}\left(v_T+\lambda_{\mathbf{n}}\left(\rho_{\mathbf{w}}-\rho_{\mathbf{n}}\right)g\nabla D\right),$$

where f_i is the *fractional flow function* for phase i;

$$f_i=\frac{\lambda_i}{\lambda_{\mathbf{n}}+\lambda_{\mathbf{w}}}. \tag{10}$$

Using this in (3) gives the *saturation equation*

$$\alpha\phi\frac{\partial S_{\mathbf{n}}}{\partial t}+\nabla\big(\alpha F_{\mathbf{n}}(S_{\mathbf{n}})\big)=-q_{\mathbf{n}}\alpha, \tag{11}$$

where

$$F_{\mathbf{n}}=f_{\mathbf{n}}\left(v_T+\lambda_{\mathbf{w}}\left(\rho_{\mathbf{n}}-\rho_{\mathbf{w}}\right)g\nabla D\right).$$

We shall be primarily interested in the case where the reservoir is horizontal, or the two densities are equal, in this case the flux function reduces to $F_{\mathbf{n}} = f_{\mathbf{n}} V_T$, where $V_T = \alpha v_T$. Summing up, we have arrived at the following model

$$\phi \frac{\partial S_{\mathbf{n}}}{\partial t} + V_T \nabla f_{\mathbf{n}}(S_{\mathbf{n}}) = 0, \tag{12}$$

$$\nabla(\alpha\lambda_T \nabla P) + Q_T = \begin{cases} \nabla(\alpha\lambda_T g\rho\nabla D) & \text{if } \rho_{\mathbf{n}} = \rho_{\mathbf{w}} = \rho, \\ 0 & \text{if } \nabla D = 0. \end{cases} \tag{13}$$

In this case the total velocity V_T is given by

$$V_T = \begin{cases} -\alpha\lambda_T(\nabla P - g\rho\nabla D) & \text{if } \rho_{\mathbf{n}} = \rho_{\mathbf{w}} = \rho, \\ -\alpha\lambda_T \nabla P & \text{if } \nabla D = 0. \end{cases} \tag{14}$$

In applications, these equations are coupled with boundary and initial conditions. We will concentrate on *water injection*. This is a process of injection of water at some locations in the reservoir, in order to maintain the pressure, thereby forcing more oil out. This situation is usually modelled by setting the initial saturation $S_{\mathbf{n}}$ (if water is the non-wetting phase) to

$$S(\mathbf{x}, 0) = \begin{cases} 1 & \text{for } |\mathbf{x}_{\text{inj}} - \mathbf{x}| < r_0, \\ 0 & \text{otherwise}, \end{cases} \tag{15}$$

where $\mathbf{x}_{\text{inj}}$ are the locations of the water injection, and r_0 is some (small) radius.

4. The numerical algorithm

The governing equations (12)–(14) constitute a coupled system of nonlinear partial differential equations. A sequential time stepping procedure is used to decouple the equations, which essentially consists of solving one equation at the time, starting with the pressure equation to generate a velocity field. Subsequently, this velocity field is used as input in the saturation equation, and so on. This strategy reflects the different nature of the elliptic pressure equation and the convection dominated parabolic saturation equation.

Let T_s be the final computing time, and choose sequential time steps Δt_n and a positive integer N such that $\sum_{m=1}^{N} \Delta t_m = T_s$. Let (P^n, V_T^n, S^n) denote the approximate solution of the reservoir flow model (12)–(14) at time $t_n = \sum_{m=1}^{n} \Delta t_m$, for some $n = 0, \dots, N-1$. The approximate solution at the next time level is computed in the following two steps:

1. *Pressure:* We use the saturation field from the previous time level in the coefficients of the pressure-velocity equation (13)-(14). Let now $\left(P^{n+1}, V_T^{n+1}\right)$ be the approximate solution of the following pressure-velocity equations:

$$\nabla\left(\alpha\lambda_T(S^n)\nabla P^{n+1}\right) + Q_T = \begin{cases} \nabla(\lambda_T(S^n) g\rho\nabla D) & \text{if } \rho_{\mathbf{n}} = \rho_{\mathbf{w}} = \rho, \\ 0 & \text{if } \nabla D = 0, \end{cases}$$

$$V_T^{n+1} = \begin{cases} -\alpha\lambda_T(S^n)\left(\nabla P^{n+1} - g\rho\nabla D\right) & \text{if } \rho_{\mathbf{n}} = \rho_{\mathbf{w}} = \rho, \\ -\alpha\lambda_T(S^n)\nabla P^{n+1} & \text{if } \nabla D = 0. \end{cases}$$

   The pressure equation is solved by a Galerkin method with piecewise linear (on triangles in the numerical grid) elements. Hence the velocity derived from the Darcy equation is piecewise constant on triangles.

2. *Saturation:* Equipped with the velocity V_T^{n+1} calculated in the previous step, let S^{n+1} be an approximate solution of the saturation equation

$$\phi \frac{\partial S^{n+1}}{\partial t} + V_T^{n+1} \nabla f_{\mathbf{n}} \left(S^{n+1}\right) = 0, \qquad S^{n+1}\left(x, t_n\right) = S^n(x).$$

A good treatment of the saturation equation is essential for obtaining an accurate solution of the reservoir flow model (12)–(14). We propose to use a level set method to solve the saturation equation (12) numerically. In what follows, we present the algorithm in detail.

4.1. **The level set method.** As a result of the sequential solution strategy outlined above, we need only consider the case where the velocity is stationary (in time) and independent of the saturation. In this case the saturation equation is a conservation law of type

$$\phi(\mathbf{x}) \frac{\partial u}{\partial t} + \mathbf{v}(\mathbf{x}) \nabla f(u) = 0, \tag{16}$$

where $u(\mathbf{x}, t)$ is the unknown function, ϕ is strictly positive, and the divergence of $\mathbf{v}$ is zero. We are interested in the initial value problem where $u(\mathbf{x}, 0)$ is given. In general, (16) possesses discontinuous solutions and must thus be interpreted in the weak sense. Furthermore, as is well known, weak solutions are not uniquely determined by their initial data and an entropy condition is used to pick out the physically correct weak solution. In the following, we use the term *entropy weak solutions* when referring to solutions of the initial value problem for (16) defined in the sense of Kružkov [21], see also Oleĭnik [26].

We first show that in an important special case, the initial value problem for the conservation law (16) can be reformulated as an Eikonal equation. If the solution of (16) is smooth, then it can be found by the method of characteristics, i.e., let $\mathbf{x}(\tau)$ and $t(\tau)$ be solutions of the ordinary differential equations

$$\begin{aligned} &\dot{\mathbf{x}} = \mathbf{v}(\mathbf{x}) f'(u(\mathbf{x}, t)), \quad \mathbf{x}(0) = \mathbf{x}_0 \\ &\dot{t} = \phi(\mathbf{x}), \quad t(0) = 0, \end{aligned} \tag{17}$$

where $\dot{}$ denotes $d/d\tau$. In this case

$$\frac{d}{d\tau} u(\mathbf{x}, t) = \phi \frac{\partial u}{\partial t} + \mathbf{v} f'(u) \nabla u = \phi \frac{\partial u}{\partial t} + \mathbf{v} \cdot \nabla f(u) = 0.$$

Consequently, $u(\mathbf{x}, t) = u(\mathbf{x}_0)$. Consider now the contour given by $u_0(\mathbf{x}) = k$, and let the 'front' $\Xi_k(t)$ be defined as

$$\Xi_k(t) = \left\{\mathbf{x} \,|\, u(\mathbf{x}, t) = k\right\}. \tag{18}$$

Now the above calculations imply that $\Xi_k(t)$ will move with a speed given by

$$\mathbf{V}(x) = \mathbf{v}(\mathbf{x}) \frac{f'(k)}{\phi(\mathbf{x})}.$$

Let $T_k(\mathbf{x})$ be the time $\Xi_k(t)$ crosses the point $\mathbf{x}$. The crossing time satisfies (in the viscosity solution sense [10]) the Eikonal equation

$$|\nabla T_k| F(\mathbf{x}) = 1, \tag{19}$$

where $F(\mathbf{x})$ is the outward normal velocity of the propagating front, which in our case reads $F(\mathbf{x}) = (\mathbf{V} \cdot \mathbf{n})(\mathbf{x})$. The normal vector of the front is given by $\mathbf{n} = \nabla T_k / |\nabla T_k|$. We now assume that $f'(k) \geq 0$ for all k, which is the case in the reservoir model (12). Then the Eikonal equation for the unknown T_k reads

$$\nabla T_k(\mathbf{x}) \cdot \mathbf{v}(\mathbf{x}) = \frac{\phi(\mathbf{x})}{f'(k)}, \qquad f'(k) > 0. \tag{20}$$

If $f'(k) = 0$, then $T_k(x) := \infty$.

In the model considered here, the flux function f is of the form

$$f(u) = \frac{\lambda_1(u)}{\lambda_2(u) + \lambda_1(u)}, \tag{21}$$

where $\lambda_1(u)$ is a non-decreasing and concave function such that $\lambda_1(0) = 0$, and $\lambda_1(1) = 1$. Similarly λ_2 is convex with $\lambda_2(0) = 1$ and $\lambda_2(1) = 0$. The prototypes for these functions are $\lambda_1(u) = u^2$ and $\lambda_2(u) = (1-u)^2$, leading to the flux function

$$f(u) = \frac{u^2}{u^2 + (1-u)^2}. \tag{22}$$

In general, the properties of λ_i ensure that the flux function is s-shaped, i.e., non-decreasing with one inflection point and $f(0) = 0$, $f(1) = 1$.

We now assume that the velocity field $\mathbf{v}$ is given by the solution of (13) via (14), and that Q_T is a sum of localized Dirac masses, i.e.,

$$Q_T(\mathbf{x}) = \sum_j c_j \delta\left(\mathbf{x}_j\right), \tag{23}$$

where $\delta(\mathbf{x})$ denotes the Dirac mass localized at $\mathbf{x}$. This is commonly used to model injection wells located at those $\mathbf{x}_j$ where $c_j > 0$ and production wells where $c_j < 0$. In this case $\mathbf{v}$ is such that the characteristic curves $\mathbf{x}(\tau)$ given by (17) connect $\mathbf{x}_l$ and $\mathbf{x}_m$, where $c_m > 0$ and $c_l < 0$.

Now consider (16) and the special initial value (15) i.e.,

$$u(\mathbf{x}, 0) = \begin{cases} 1 & \text{if } |\mathbf{x}_j - \mathbf{x}| < r_0 \text{ and } c_j > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{24}$$

In this case the solution will not be smooth, and to use (20) we must solve the Riemann problem

$$\frac{\partial v}{\partial t} + \frac{\partial f(v)}{\partial x} = 0, \quad v(x,0) = \begin{cases} 1 & \text{for } x < 0, \\ 0 & \text{for } x \geq 0, \end{cases} \tag{25}$$

where f is given by (22). The solution is found by taking the upper convex envelope of f between 0 and 1. Since f is s-shaped, the solution is of the form

$$v(x,t) = \begin{cases} \left(\tilde{f}'\right)^{-1}(x/t) & \text{for } x/t < \tilde{f}'\left(\bar{u}\right), \\ 0 & \text{otherwise,} \end{cases} \tag{26}$$

where $\tilde{f}$ denotes the upper convex envelope, and $(\tilde{f}')^{-1}$ the inverse of its derivative. Furthermore, $\bar{u}$ is the solution of

$$f'\left(\bar{u}\right) = \frac{f\left(\bar{u}\right)}{\bar{u}}.$$

If f is given by (22), then $\bar{u} = \sqrt{2}/2 \approx 0.707$.

Due to the special form of the initial value function (15) and the velocity field $\mathbf{v}$, the solution u will take values in the set $[\bar{u}, 1] \bigcup \{0\}$. Let now $T_k(\mathbf{x})$ be the time that $u(\mathbf{x}, t) = k$ for $k \in [\bar{u}, 1]$. Since $\Xi_k(0) = \{\mathbf{x} \mid |\mathbf{x} - \mathbf{x}_j| = r_0 \text{ and } c_j > 0\}$ for all such k, T_k equals $\tilde{T}/f'(k)$, where $\tilde{T}$ solves the equation

$$\nabla T \cdot \mathbf{v} = \phi \tag{27}$$

with the boundary condition $T = 0$ for $\mathbf{x} \in \Xi_k(0)$. Consequently, the unique weak solution of (16) and (24) is given by

$$u(\mathbf{x}, t) = \begin{cases} 0 & \text{if } t < T_{\bar{u}}(\mathbf{x}), \\ k & \text{if } T_k(\mathbf{x}) = t, \end{cases} = \begin{cases} 0 & \text{if } t f'(\bar{u}) < T(\mathbf{x}), \\ \left(\tilde{f}'\right)^{-1} \left(\frac{t}{T(\mathbf{x})}\right) & \text{otherwise.} \end{cases} \tag{28}$$

Hence, solving the Eikonal equation (27) is equivalent to solving the initial value problem (24) for (16).

4.2. **The fast marching method.** So far, we have defined a semi-discrete approximation of (16), where (27) is solved exactly. The next step is then to compute (27) numerically. To this end, we use a *fast marching method.* For simplicity, we use a regular grid with $m \times n$ cells in this study. Our implementation of this fast marching method is taken from Sethian [29, 30, 31]. The basic observation underlying the fast marching method is that all waves have finite speed of propagation. Since the flow is directed out from injection wells and towards production wells (unless wells are shut off), information will flow from regions with smaller arrival times towards regions with higher arrival times. In other words, the arrival time T, cf. (27), at a certain point in the grid depends only on points having smaller values.

Rather than solving equation (27) simultaneously at all points of the domain, we can use an iterative approach based on the above observations in which we gradually march the solution outwards from the injection wells. To this end, we divide the nodes into three categories: alive nodes, narrow-band nodes, and far-away nodes. Assume that the solution has been computed for all alive nodes. The narrow-band nodes consist of all nodes lying within a certain distance in time from the alive nodes. At each narrow-band node an estimate of the arrival time has been computed during the previous steps. The far-away nodes consist of the remaining nodes in the grid. To march the solution one step forward, we pick the node in the narrow band having the lowest arrival time and update its value using an upwind discretization of the Eikonal equation. Since the calculation at the node point uses only nodes with lesser arrival times, the arrival time at the current node can not increase. The node is then tagged as alive and removed from the narrow band, and we update the arrival time of all neighbouring nodes that are not alive. If a neighbour is a far-away node, the node is added to the narrow band. We continue the algorithm until either all nodes are visited or a certain prescribed maximum arrival time is reached.

The points in the narrow band are organized in a *complete binary three.* Hence retrieving the node with the smallest T-value is trivial, and inserting new nodes is an $\mathcal{O}(\log N)$ operation, where N is the number of nodes in the narrow band. Typically, N is of the same order as the number of grid blocks in one dimension, i.e., of order m or n. Solving the Eikonal equation for the whole grid usually means 'marching' the narrow band across the entire grid. Consequently, solving the Eikonal equation by the fast marching method, storing the narrow-band points in a binary three, is an $\mathcal{O}(N \log N)$ operation. This means that it will be much faster than other methods for solving the Eikonal equation which are typically $\mathcal{O}\left(N^2\right)$, see, e.g., [15]. The algorithm is described in more detail in Sethian [29, 30, 31].

To discretize the Eikonal equation, we can use one of several upwind methods. One choice is to square the equation (27)

$$\left(\nabla T \cdot \mathbf{v}\right)^2 = \phi^2. \tag{29}$$

Then we can use the five-point difference stencil

$$\Big[\big(\max(D_x^- T, 0) + \min(D_x^+ T, 0)\big)\, v_x + \big(\max(D_y^- T, 0) + \min(D_y^+ T, 0)\big)\, v_y\Big]^2 = \phi^2, \tag{30}$$

where

$$D_x^\pm = \pm\frac{T_{i\pm1,j} - T_{i,j}}{\Delta x}, \qquad D_y^\pm = \pm\frac{T_{i,j\pm1} - T_{i,j}}{\Delta y}. \tag{31}$$

Inserting the discretization into (30) gives a nonlinear equation for $T_{i,j}$ in terms of its four closest neighbours $T_{i\pm1,j}, T_{i,j\pm1}$ that can be solved by, e.g., Newton iteration. The advantage of this method is that it is very fast, typically we only require two or three Newton iterations until numerical convergence. However, squaring the equation means that we no longer differ between 'uphill' and 'downhill'. Hence this scheme works only in the case where $\operatorname{sign}(\nabla T \cdot \mathbf{v})$ is constant. Since there are potentially five points in the stencil, we call this scheme a 'five-point scheme'.

A better alternative is to use (27) directly, and compute local streamlines around each point $\mathbf{x}_{i,j} = (i\Delta x, j\Delta y)$, and use these to update $T_{i,j}$. These approximate streamlines can be defined in several ways; we use the following simple strategy. Let $\omega_{i,j}$ consist of the points around $\mathbf{x}_{i,j}$, i.e.,

$$\omega_{i,j} = \left\{(x, y) \mid x = (i \pm 1)\Delta x,\ y = (j \pm 1)\Delta y\right\}.$$

Assume that the calculated velocity at $\mathbf{x}_{i,j}$ is $\mathbf{v}_{i,j}$. Then let $\boldsymbol{\theta}_{i,j}$ be the point on $\omega_{i,j}$ such that

$$\boldsymbol{\theta}_{i,j} = \mathbf{x}_{i,j} + \theta_{i,j}\mathbf{v}_{i,j},$$

for some positive $\theta_{i,j}$. Then let $T_{\theta_{i,j}}$ be defined by linear interpolation between the points $T_{k,l}$ on $\omega_{i,j}$. Now we can discretize (27) as

$$\frac{T_{i,j} - T_{\theta_{i,j}}}{|\mathbf{x}_{i,j} - \boldsymbol{\theta}_{i,j}|}\,|\mathbf{v}_{i,j}| = \phi_{i,j},$$

where $\phi_{i,j}$ is the porosity at $\mathbf{x}_{i,j}$. Hence

$$T_{i,j} = T_{\theta_{i,j}} + \theta_{i,j}\phi_{i,j}. \tag{32}$$

Since there are eight neighbours that can contribute to $T_{i,j}$ (but at most two actually will), we call this scheme a 'nine-point scheme'.

Presumably, higher-order approximate streamlines, using the velocity field and T values from more grid points would give better results, but we found that this simple approach worked well. Also, as we used a finite element method with piecewise linear elements for the pressure equation, this results in a piecewise constant velocity field, so it fits with our discrete local streamlines. If one uses a higher-order method for the pressure equation, one should modify the local streamline computation accordingly.

Note that the values $\boldsymbol{\theta}_{i,j}$ can be computed at the beginning of the calculation, once the velocity field is known.

4.3. **Restarting.** The initial data will not generally be of the simple form (24). This form applies only for the first step in a sequential time stepping procedure for (12)–(14). For subsequent steps, the initial data will be given by the saturation at the end of the previous time step, i.e.,

$$s_{i,j} = \begin{cases} 0 & \text{if } T_{i,j} \geq T_\ell, \\ \left(\tilde{f}'\right)^{-1}\left(\frac{t_\ell}{T_{i,j}}\right) & \text{otherwise,} \end{cases} \tag{33}$$

where $T_\ell = t_\ell f'(\bar{u})$, and t_ℓ is the time after ℓ time steps. This saturation is then used as coefficients in the pressure equation (13) and the velocity for the next time interval is computed. To solve the saturation equation again, we could fix some small number Δs and make an initial narrow-band where $s_{i,j}$ in the interval $[\bar{u}, \bar{u} + \Delta s]$, and tag as far-away points and alive points those where $s_{i,j} = 0$ and $s_{i,j} > \bar{u} + \Delta s$ respectively. This would update the region around the discontinuity in s. To update the rest of the saturations we could define narrow bands in intervals $[\bar{u} + (k-1)\Delta s, \bar{u} + k\Delta s]$ for k such that $\bar{u} + k\Delta s \leq 1$, and update the saturation in those intervals.

It is however more convenient to use the T values directly. Fix some small number ΔT and tag as narrow-band those points where $T_{i,j}$ is in the interval $[T_\ell - k\Delta T, T_\ell - (k-1)\Delta T]$. The far-away points and the alive point are those where $T_{i,j} > T_\ell - (k-1)\Delta T$ and $T_{i,j} < T_\ell - k\Delta T$. The solution with this as initial values is stopped when the smallest largest alive point has a T value of $T_{\ell+1} - (k-1)\Delta T$. This is repeated until $T_\ell - k\Delta T = 0$. Then we can set $\ell = \ell + 1$ and use (33) to update the saturation for the next time step.

This means that we have to solve the Eikonal equation $K = T_\ell/\Delta T$ times each time we restart. But each narrow band will typically pass only a correspondingly small region of the grid. Therefore solving K times does not take longer time than solving once with a larger 'time step'.

5. Numerical examples

In this section we present four numerical examples that highlight the features of the method. The first two examples are quarter five-spots with homogeneous and heterogeneous permeability, respectively. The third example describes flow in a channel system. The last example studies flow around a low-permeable barrier. In all three examples, we use the fractional flow function defined in (10) with $\lambda_{\mathbf{n}}(S) = S^2/\mu_{\mathbf{n}}$ and $\lambda_{\mathbf{w}}(S) = (1-S)^2/\mu_{\mathbf{w}}$.

The level set method is compared with a large time step, front tracking method based on dimensional splitting with Dafermos method [11, 18, 17] for each one-dimensional problem. The efficiency of this method has been documented in previous studies [4, 5, 19, 23, 24].

Regarding the pressure equation, wells are represented as point sources (23), and we use homogenous Neumann boundary conditions at the boundaries of the reservoir. As mentioned above, we use a first-order element method to solve the pressure equation (13), where the basis functions are piecewise linear on triangles. To solve the resulting linear system of equations we use a conjugate gradient method.

5.1. **Homogeneous quarter five-spot.** The first example is the well-known quarter five-spot test case. The test case consists of a repeated pattern of squares. In each square there is an injection well at the origin and production wells at the corners $(\pm 1, \pm 1)$. All wells have rates equal unity and we use mobility ratio equal one. Since the velocity field is slowly varying in this case, we use only one (initial) pressure update.

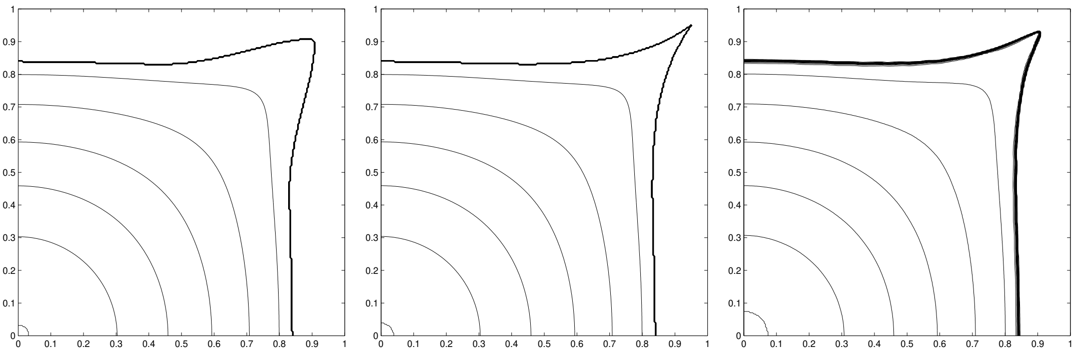

FIGURE 1. Saturation profiles computed by the level set method with five-point (left) and nine-point scheme (middle) and by the front tracking method (right).

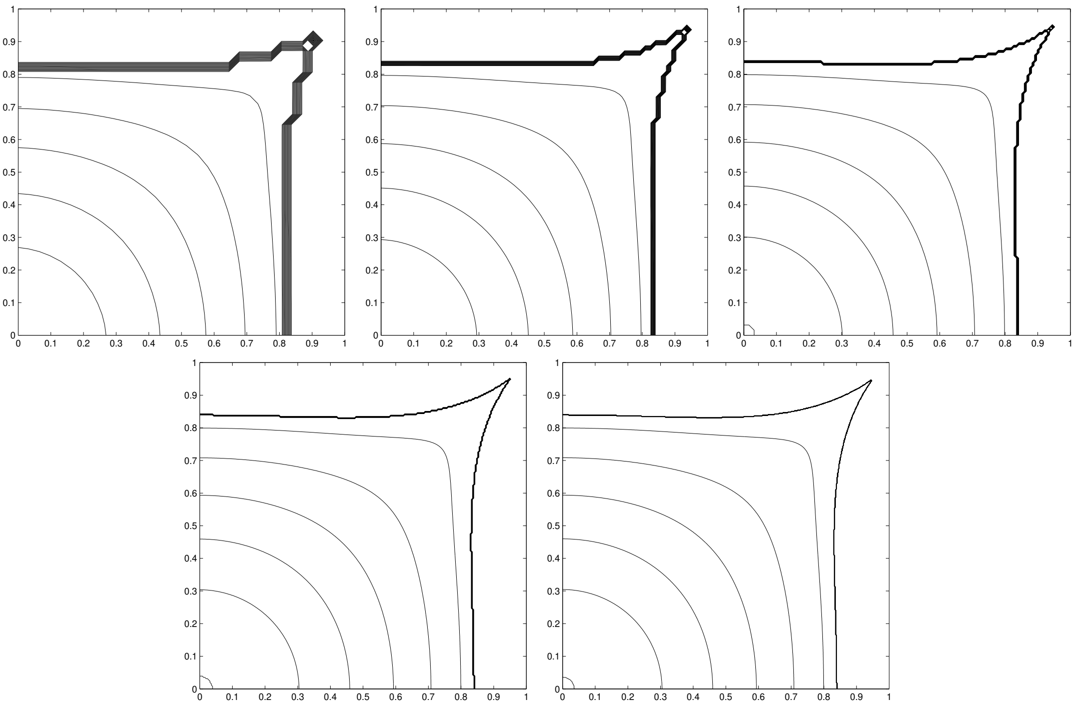

FIGURE 2. Saturation profiles for the nine-point scheme on $2^k \times 2^k$ grids for $k = 5, \ldots, 9$.

Figure 1 shows the saturation profile at time $t = 0.59$ computed on a 256×256 grid by the level set method with the five-point and the nine-point scheme and by the front tracking method. The front tracking method was run with CFL number 32.0 up to time 0.5 and then 4.0 afterwards. The five-point scheme is obviously the most diffusive, giving a much too broad finger. The front tracking method gives accurate resolution of the finger, but has some numerical diffusion along the leading front due to repeated projections onto a uniform grid. The sharpest resolution is obtained by the nine-point scheme which has *very* little numerical diffusion. Table 1 gives the mass balance errors and the runtimes measured on a dual 400

TABLE 1. *Relative mass balance error and runtime for homogeneous quarter five-spot.*

	five-point	nine-point	front tracking
runtime [s]	1.8	2.6	24.6
mass error [%]	0.84	-0.29	-0.19

TABLE 2. *Runtimes, mass error, and self-convergence for a grid refinement study in Figure 2. The discrete L^1-errors are measured relative to the solution computed on the 512×512 grid and normalized by the corresponding L^1-norm.*

N	runtime [s]	mass [%]	rate	L^1 error	rate
32	0.02	-5.77	—	0.0521	—
64	0.10	-1.93	1.58	0.0176	1.56
128	0.48	-0.85	1.19	0.0076	1.21
256	2.58	-0.29	1.56	0.0028	1.41
512	15.43	-0.11	1.45	—	—

MHz Pentium II processor. We remark that the runtimes reported here are for the saturation equation only. The mass balance error is defined as

$$\Delta x \Delta y \sum_{ij} (s_{ij}(t) - s_{ij}(0)) - t,$$

where the time t equals the number of injected pore volumes. Notice that compared with front tracking, the level set method uses only around 1/10th of the runtime. With a CFL number 1.0 for front tracking, the factor becomes 1/20.

Figure 2 and Table 2 display the result of a grid refinement study of the saturation profile at time $t = 0.59$ for the nine-point scheme. The convergence is of order one both with respect to mass balance error and L^1 error. The runtime increases with an exponent 2.4–2.6 in the number of grid blocks in one direction.

5.2. **Heterogeneous quarter five-spot.** In the next example we add a stochastically generated permeability field to the above case and change the viscosity ratio to $\mu_{\mathbf{n}} : \mu_{\mathbf{w}} =$ 4:1. The permeability is realized from a log-Gaussian distribution, with values in the range from 4.8 mD to 4.2 D. Figure 3 shows saturation profiles at time $t = 0.35$ computed by the nine-point scheme and front tracking (with CFL number 16). The fingers are very sharply represented and are almost identical for both methods. However, notice the small oscillations in the front tracking plot. For lower CFL numbers in the front tracking method, these oscillations disappear and the fingers become slightly longer due to added numerical diffusion. The runtimes for the methods are 3.0 seconds for the nine-point scheme and 25.0 seconds for the front tracking method. Both methods used a 256×256 grid.

5.3. **A channel problem.** To investigate further preservation of symmetries and dissipation properties of the level set scheme, we consider flow in a channel system in the form of a cross. The permeability is set equal 0.1D in the beams of the cross and 0.01mD outside. Water is injected at a uniform rate at the bottom and on the left, and oil is produced at the top and on the right. Due to the symmetry of the problem (about a diagonal from the lower left to the upper right corner), the advancing water fronts should not intersect, but can come arbitrary close along the diagonal as time increases, see Figure 4. For comparison, Figure 5

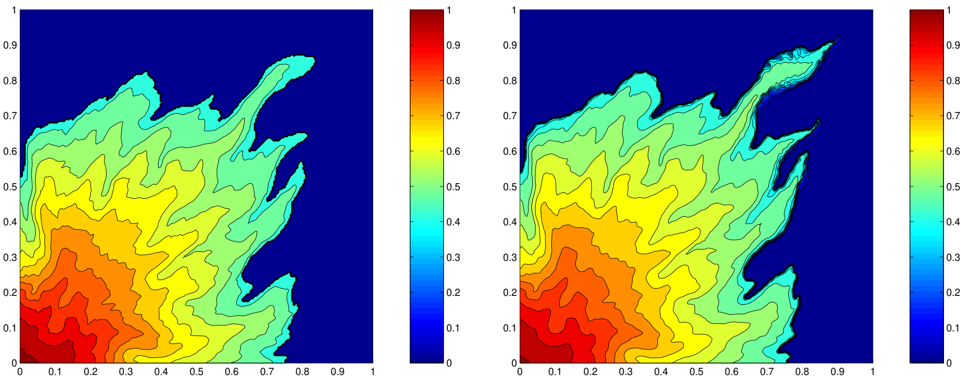

FIGURE 3. Saturation profiles computed by the nine-point scheme (left) and front tracking (right).

shows the solution computed by front tracking with CFL number 8.0 on a 513×513 grid. Even on this fine grid, the two water fronts collapse into a single front due to the numerical diffusion introduced by the projection in that scheme. The fact that the two water fronts will not intersect in the level set method can easily be seen from the $T_{i,j}$, which is plotted for the 100×100 grid in Figure 6. In the corresponding simulation, the fast marching method was run only until time $T = 1.0/f'(\bar{u})$.

Figure 7 gives the result of a grid refinement study for the channel flow at time $t = 0.15$. On the coarsest grids, some grid blocks along the diagonal have been partially flooded by both injection wells, as can be seen from the wiggles in the contour lines. Still, the two water fronts are clearly separated on all grids. Continuing the refinement to a 400×400 grid gave no visual changes compared with the 200×200 grid.

5.4. **Reservoir with a barrier.** In the next example we consider a reservoir with a horizontal low-permeability barrier with a narrow passage at each end. The barrier is centred around $(0.5, 0.5)$ and has width 0.9 and height 0.1. Inside the barrier the permeability is 0.01 mD, in the left passage it is 0.5 D and 1.0 D elsewhere. The injection well is in the lower left corner and the production well is in the upper right corner.

Figure 8 shows saturation profiles computed by the level set and the front tracking method on a 129×129 grid. In the left column, the pressure was computed only once and in the right column it was computed 9 times. For equal number of pressure updates, the solutions computed by the two methods are quite similar. The front tracking solutions are more diffusive, while the level set method has sharper fingers. This is particularly evident in the right column and is in correspondence with the observations for the homogeneous quarter five-spot simulations.

6. Concluding remarks

The level set method presented above is very accurate and efficient for numerical reservoir simulation. In the level set formulation, the saturation equation is recast to a set of stationary Eikonal equations that can be solved by a fast marching method. This gives very high efficiency of the computer code and extensions to three dimensions (disregarding gravity) is straightforward.

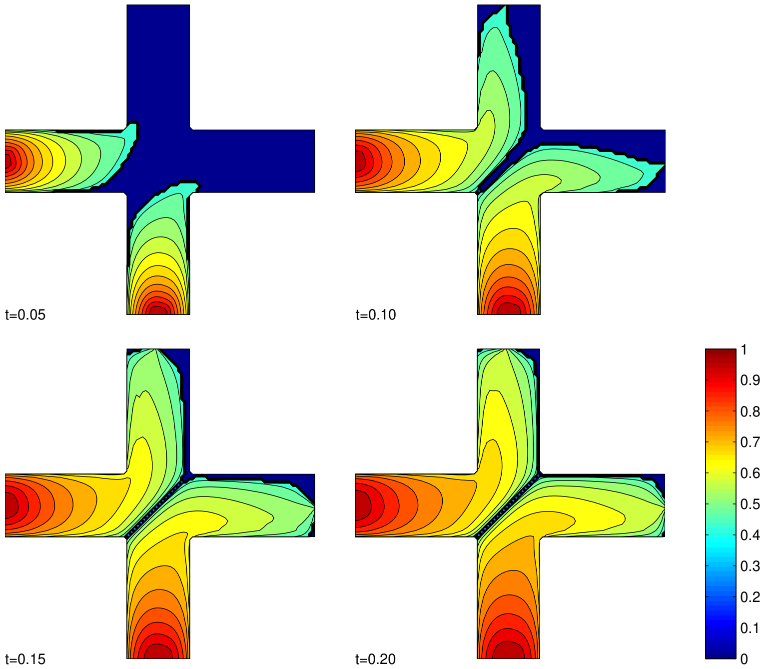

FIGURE 4. Advancing water fronts computed by the level set method on a 100×100 grid.

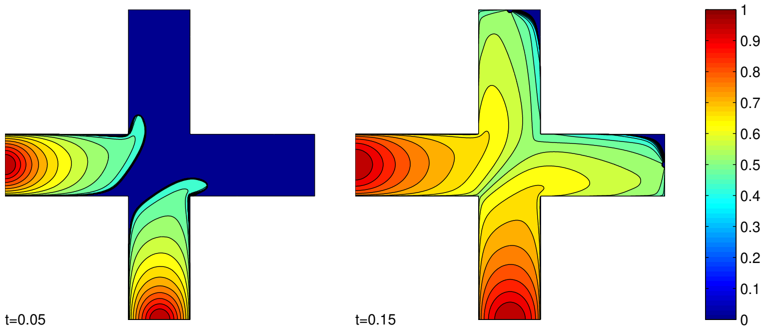

FIGURE 5. Advancing water fronts computed by front tracking on a 513×513 grid.

Even though the method can be viewed as a streamline method, it operates on a grid and does not explicitly compute streamlines. Thus, the method is easy to program and avoids most of the numerical difficulties associated with streamline methods.

Furthermore, it is straightforward to extend the level set method to more general models where one knows the solution of the one-dimensional Riemann problem for the saturation equation, e.g., polymer flow. Other obvious extensions are computations of tracer injection and drainage and seepage areas.

Also, the level set method discussed here can be used as one ingredient in numerical methods for solving related reservoir models with capillary pressure. A simple way to do this would be to use operator splitting as in [14].

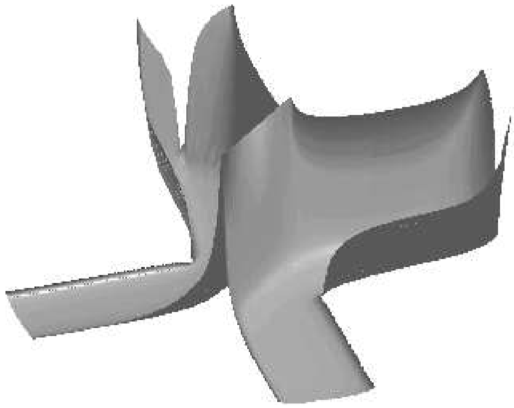

FIGURE 6. $T_{i,j}$ computed on a 100×100 grid. The height represents the time a water front with unit velocity takes to reach the point (x, y).

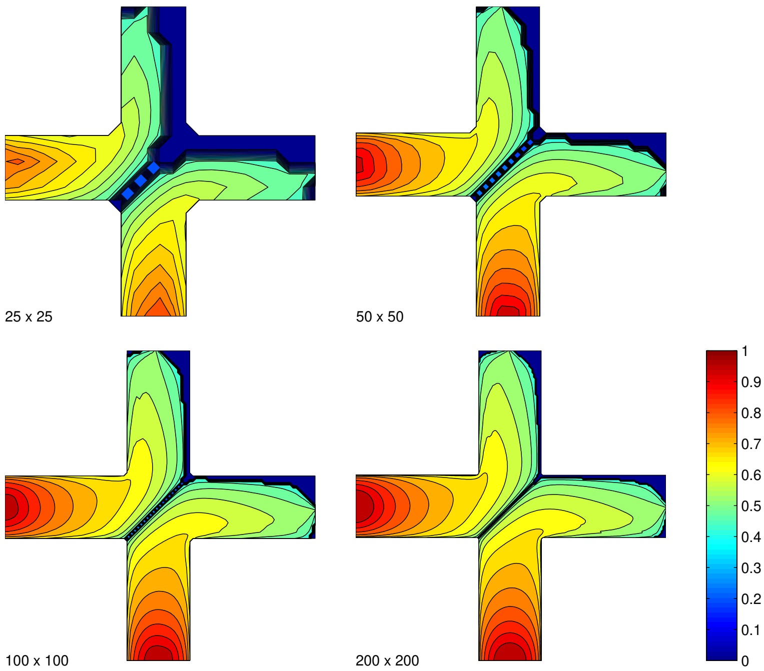

FIGURE 7. The result of a convergence study for the channel problem.

References

[1] D. Adalsteinsson and J. A. Sethian. A fast level set method for propagating interfaces. *J. Comput. Phys.*, 118(2):269–277, 1995.

[2] T. D. Aslam. A level set algorithm for tracking discontinuities in hyperbolic conservation laws i: Scalar equations. Preprint 28, UCLA Computational and Applied Mathematics, 1998.

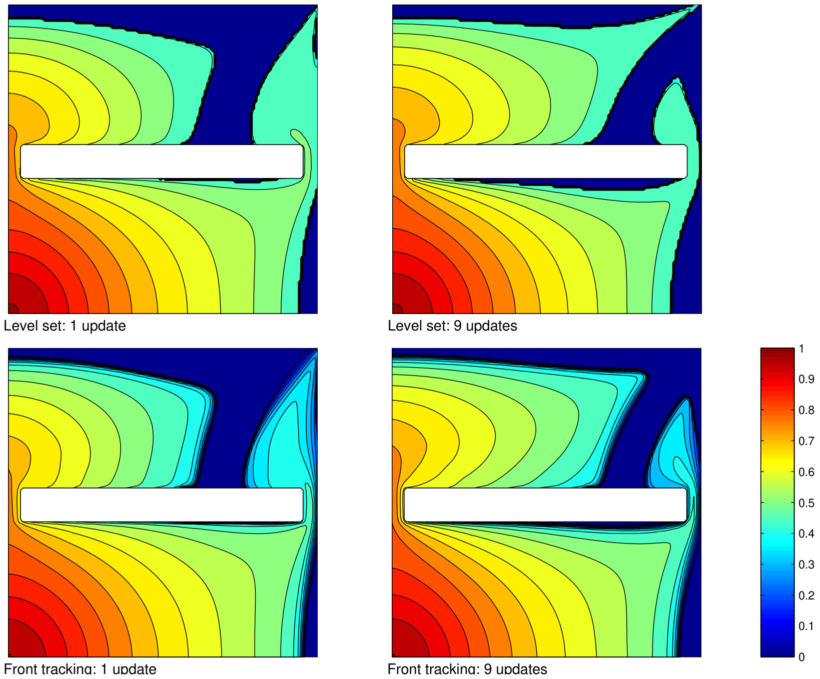

FIGURE 8. Saturation profiles at time $t = 0.45$ computed by the nine-point scheme (upper row) and front tracking (lower row).

[3] K. Aziz and A. Settari. *Petroleum reservoir simulation*. Elsevier Applied Science Publishers, Essex, England, 1979.
[4] F. Bratvedt, K. Bratvedt, C. F. Buchholz, T. Gimse, H. Holden, L. Holden, R. Olufsen, and N. H. Risebro. Three-dimensional reservoir simulation based on front tracking. In *North Sea Oil and Gas Reservoirs – III*, pages 247–257. Kluwer Academic Publishers, 1994.
[5] F. Bratvedt, K. Bratvedt, C. F. Buchholz, T. Gimse, H. Holden, L. Holden, and N. H. Risebro. FRONTLINE and FRONTSIM: two full scale, two-phase, black oil reservoir simulators based on front tracking. *Surveys Math. Indust.*, 3(3):185–215, 1993.
[6] F. Bratvedt, T. Gimse, and C. Tegnander. Streamline computations for porous media flow including gravity. *Transport in Porous Media*, 25:63–78, 1996.
[7] G. Chavent and J. Jaffre. *Mathematical models and finite elements for reservoir simulation*, volume 17 of *Studies in mathematics and its applications*. North Holland, Amsterdam, 1986.
[8] Y. G. Chen, Y. Giga, and S. Goto. Uniqueness and existence of viscosity solutions of generalized mean curvature flow equations. *J. Differential Geom.*, 33(3):749–786, 1991.
[9] M. G. Crandall, H. Ishii, and P.-L. Lions. User's guide to viscosity solutions of second order partial differential equations. *Bull. Amer. Math. Soc. (N.S.)*, 27(1):1–67, 1992.
[10] M. G. Crandall and P.-L. Lions. Viscosity solutions of Hamilton-Jacobi equations. *Trans. Amer. Math. Soc.*, 277(1):1–42, 1983.
[11] C. M. Dafermos. Polygonal approximation of solutions of the initial value problem for a conservation law. *J. Math. Anal. Appl.*, 38:33–41, 1972.
[12] M. S. Espedal and K. H. Karlsen. Numerical solution of reservoir flow models based on large time step operator splitting algorithms. In A. Fasano and H. van Duijn, editors, *Filtration in Porous Media and Industrial Applications*, Lecture Notes in Mathematics. Springer. To appear.
[13] L. C. Evans and J. Spruck. Motion of level sets by mean curvature. I. *J. Differential Geom.*, 33(3):635–681, 1991.
[14] S. Evje, K. H. Karlsen, K. A. Lie and N. H. Risebro. Front tracking and operator splitting for non-linear degenerate convection-diffusion equations. *Institut Mittag-Leffler Report*, 1997, Stockholm, Sweden

[15] M. Falcone, T. Giorgi and P. Loretti. Level sets of viscosity solutions: Some applications to fronts and rendez-vous problems. *SIAM J. Appl. Math.* 54(5):1335:1354, 1994.
[16] V. Haugse, K. H. Karlsen, K.-A. Lie, and J. Natvig. Numerical solution of the polymer system by front tracking. Preprint, 1999.
[17] H. Holden and L. Holden. On scalar conservation laws in one-dimension. In S. Albeverio, J. E. Fenstad, H. Holden, and T. Lindstrøm, editors, *Ideas and Methods in Mathematics and Physics*, pages 480–509. Cambridge University Press, Cambridge, 1988.
[18] H. Holden, L. Holden, and R. Høegh-Krohn. A numerical method for first order nonlinear scalar conservation laws in one-dimension. *Comput. Math. Applic.*, 15(6–8):595–602, 1988.
[19] H. Holden and N. H. Risebro. A method of fractional steps for scalar conservation laws without the CFL condition. *Math. Comp.*, 60(201):221–232, 1993.
[20] M. J. King and A. D. Datta-Gupta. Streamline simulation: a current perspective. *In Situ (Special Issue on Reservoir Simulation)*, 22(1):91–140, 1998.
[21] S. N. Kružkov. First order quasi-linear equations in several independent variables. *Math. USSR Sbornik*, 10(2):217–243, 1970.
[22] R. J. LeVeque. *Numerical methods for conservation laws.* Birkhäuser Verlag, Basel, second edition, 1992.
[23] K.-A. Lie. A dimensional splitting method for nonlinear equations with variable coefficients. Preprint. Mathematics No. 17, Norwegian University of Science and Technology, Trondheim, Norway, 1997.
[24] K.-A. Lie, V. Haugse, and K. H. Karlsen. Dimensional splitting with front tracking and adaptive grid refinement. *Numer. Methods for Partial Differential Equations*, 14(5):627–648, 1998.
[25] K. W. Morton. *Numerical solution of convection-diffusion problems.* Chapman & Hall, London, 1996.
[26] O. A. Oleĭnik. Discontinuous solutions of non-linear differential equations. *Amer. Math. Soc Transl. Ser. 2*, 26:95–172, 1963.
[27] S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.*, 79(1):12–49, 1988.
[28] D. W. Peaceman. *Fundamentals of Numerical Reservoir Simulation.* Elsevier, 1977.
[29] J. A. Sethian. A fast marching level set method for monotonically advancing fronts. *Proc. Nat. Acad. Sci. U.S.A.*, 93(4):1591–1595, 1996.
[30] J. A. Sethian. *Level set methods.* Cambridge University Press, Cambridge, 1996. Evolving interfaces in geometry, fluid mechanics, computer vision, and materials science.
[31] J. A. Sethian. Theory, algorithms, and applications of level set methods for propagating interfaces. In *Acta numerica, 1996*, pages 309–395. Cambridge Univ. Press, Cambridge, 1996.
[32] P. E. Souganidis. Front propagation: theory and applications. In *Viscosity solutions and applications (Montecatini Terme, 1995)*, pages 186–242. Springer, Berlin, 1997.

(Kenneth Hvistendahl Karlsen)
Department of Mathematics, University of Bergen, Johs. Bruunsgt. 12, N–5008 Bergen, Norway
E-mail address: `kennethk@math.uib.no`
URL: `www.mi.uib.no/~kennethk/`

(Knut–Andreas Lie)
Department of Informatics, University of Oslo, P.0. Box 1080 Blindern, N–0316 Oslo, Norway
E-mail address: `kalie@ifi.uio.no`
URL: `www.ifi.uio.no/~kalie/`

(Nils Henrik Risebro)
Department of Mathematics, University of Oslo, P.O. Box 1053 Blindern, N–0316 Oslo, Norway
E-mail address: `nilshr@math.uio.no`
URL: `www.math.uio.no/~nilshr/`