# Fast computation of arrival times in heterogeneous media

Inga Berre (inga.berre@mi.uib.no)
*Dept. of Mathematics, University of Bergen, Johs. Brunsgt. 12, N-5008 Bergen, Norway.*

Kenneth Hvistendal Karlsen (kenneth.karlsen@math.uio.no)
*Centre of Mathematics for Applications, University of Oslo, P.O. Box 1053 Blindern, N-0316, Norway.*

Knut–Andreas Lie (knut-andreas.lie@sintef.no)
*SINTEF ICT, Dept. of Applied Mathematics, P.O. Box 124 Blindern, N-0314 Oslo, Norway.*

Jostein R. Natvig (jostein.r.natvig@sintef.no)
*SINTEF ICT, Dept. of Applied Mathematics, P.O. Box 124 Blindern, N-0314 Oslo, Norway.*

**Abstract.** We continue the work that was initiated in [7] on a marching method for simulating two-phase incompressible immiscible flow of water and oil in a porous medium. We first present an alternative derivation of the marching method that reveals a strong connection to modern streamline methods. Then, through the study of three numerical test cases we present two deficiencies: (i) the original marching algorithm does not always compute the correct solution of the underlying difference equations, and (ii) the method gives largely inaccurate arrival times in the presence of large jumps within the upwind difference stencil. As a remedy of the first problem we present a new advancing-front method, which is faster than the original marching method and guarantees a correct solution of the underlying discrete linear system. To cure the second problem we present two adaptive strategies that avoid the use of finite-difference stencils containing large jumps in the arrival times.

The original marching method was introduced as a fast tool for simulating two-phase flow scenarios in heterogeneous formations. The new advancing-front method has limited applicability in this respect, but may rather be used as a fast and relatively accurate method for computing arrival times and derived quantities in heterogeneous media.

## 1. Introduction

The mathematical model for porous media flow considered herein describes the flow of two inviscid, incompressible and immiscible fluids in a domain $\Omega \subset \mathbf{R}^3$ filled with a porous medium. In reservoir simulation, these fluids are typically oil and water. Neglecting capillary pressure and gravity, the motion of the two fluids is governed by the scalar conservation law

$$\frac{\partial S}{\partial t} + \mathbf{v}(\mathbf{x}) \cdot \nabla f(S) = 0, \tag{1}$$

where $S$ is the saturation of water and $f$ is the fractional flow function accounting for different flow resistance of the two fluids. A complete model must also include an equation for the fluid velocity $\mathbf{v}$. The most commonly used model is a constitutive relation known as Darcy's law that relates $\mathbf{v}$ to the fluid pressure. The continuity equation can then be stated as

$$\nabla \cdot (\lambda_T \nabla P) = 0 \qquad \text{in } \Omega, \tag{2.1}$$

$$\mathbf{v} = -\lambda_T \nabla P, \tag{2.2}$$

where $P$ is the total fluid pressure and $\lambda_T$ is the total mobility, which depends on the absolute permeability as well as on $S$. We model injection and production of fluids with Neumann boundary conditions on $\Gamma^+ = \{\mathbf{v} \cdot \mathbf{n} > 0\}$ and $\Gamma^- = \{\mathbf{v} \cdot \mathbf{n} < 0\}$ respectively. The equations (2.1)–(2.2) yield a divergence free irrotational velocity field. We refer to [1, 3] for more general models.

The model (1)–(2.2) is solved most effectively by applying a specialized numerical method to each equation, which accurately resolves important features of the equation, see for example [5] and references therein. Recently, Karlsen, Lie and Risebro [7] proposed a new numerical method for the solution of (1) for a fixed velocity field $\mathbf{v}$. Their approach was an attempt to apply the level-set technology of Osher and Sethian [10], together with the fast-marching method [13, 16, 17, 6], to the propagation of curves of constant water saturation $S = \sigma$. In the absence of shocks, each point on these level curves moves with a characteristic speed $f'(\sigma)\mathbf{v}$. Thus, tracking a saturation contour for a fixed saturation $\sigma$ can be recast as tracking the zeroth level set of a function $\phi$ that is described by the linearized transport equation

$$\frac{\partial \phi}{\partial t} + f'(\sigma)\mathbf{v} \cdot \nabla \phi = 0. \tag{3}$$

In general, the solution obtained by tracking all saturation contours according to (3) will lead to a multivalued solution. To compute the

entropy solution of (1) we must modify the wave speed $f'(\sigma)\mathbf{v}$ when shocks form, i.e., when level sets for different $\sigma$ coincide. To avoid this complication we henceforth assume that the relative speeds of the level sets are monotone in $\sigma$. A simple way to ensure this is to consider only initial saturations formed by two constant states, say $\sigma^+$ in $\mathcal{W}$ and $\sigma^-$ in $\Omega \setminus \mathcal{W}$, separated by a contour at $\partial\mathcal{W}$. Then, all initial level sets of the saturation coincide with $\partial\mathcal{W}$. For this Riemann-like initial value problem, the relative speed of each contour is constant: If $\sigma^+ > \sigma^-$ (or $\sigma^+ < \sigma^-$), the relative speed is given as the derivative of the upper convex (or lower concave) envelope $f_c$ of $f$ between $\sigma^-$ and $\sigma^+$. The single-valued solution of (1) in $\Omega \setminus \mathcal{W}$ can then be obtained by solving

$$\frac{\partial \phi}{\partial t} + f_c'(\sigma)\mathbf{v} \cdot \nabla \phi = 0 \tag{4}$$

for all saturation contours. From a physical point of view, this corresponds to injection of a fixed liquid mixture into a reservoir containing another homogeneous fluid mixture. Note that we must have $\Gamma^+ \subseteq \mathcal{W}$ since fluid must flow out of $\mathcal{W}$. From here on, we will simply use $\mathcal{W} = \partial\mathcal{W} = \Gamma^+$.

Since the motion of the level sets are identical up to a constant in the wave speed, we should be able to compute them all in one go. If we require that $\mathbf{v} \cdot \mathbf{n} > 0$ on the initial contour $\partial\mathcal{W} = \Gamma+$, a level set will not pass any point in the domain more than once. Then we can set $\phi(\mathbf{x}, t) = T(\mathbf{x})/f_c'(\sigma) - t$ for some function $T$ independent of $\sigma$, and (4) is reduced to a linear boundary-value problem

$$\mathbf{v} \cdot \nabla T = 1 \text{ in } \Omega, \qquad T = 0 \text{ on } \Gamma^+. \tag{5}$$

In this case $T$ represents the motion of a level set with unit relative speed. Thus, we can interpret $T(\mathbf{x})$ as the time needed for a passive test particle to travel from some point on $\Gamma^+$ to $\mathbf{x}$. We will therefore denote this quantity by arrival time or time-of-flight and (5) as the arrival-time equation. Ultimately, it is the saturation that is to be computed. We recover the saturation by determining for which saturation the corresponding level set has reached a given point $\mathbf{x}$ at time $t$; that is, by solving $0 = T(\mathbf{x})/f_c'(\sigma) - t$ for $\sigma$. This is the well-known Buckley–Leverett profile, see [7] for more details. With this approach, the amount of work has been reduced to the solution of the boundary value problem (5) and the inversion of a function $f_c'$ (which can be done analytically). In [7], (5) was solved using a marching algorithm that has an asymptotic cost of $\mathcal{O}(N \log N)$ operations for $N$ unknowns. While the numerical examples in [7] were two-dimensional, some three-dimensional examples with the same method were presented in [8]. In [2] the method was extended to incorporate capillary forces

(i.e., viscous terms in the transport equation) through the use of the transport-collapse operator.

In this paper we will first give an alternative derivation of the marching method presented in [7] that will demonstrate a close connection between modern streamline methods and the marching approach. For an overview of the research on streamlines methods, we refer to [9] and the references therein. We then continue with an investigation of the marching method. In [7, 8], numerical tests showed that the method was fast and reliable and had acceptable accuracy. Unfortunately, the formulation of a boundary value problem for the arrival time and the subsequent fast-marching-like solution procedure is not as robust as first reported. A more detailed analysis of the discretization and the solution procedures reveals two problems. These problems are not apparent in the presentation of [7]. The first problem is related to the discretization of (5) for discontinuous media. The second problem is that the discretization of (5) lacks the causality principle that the fast-marching method hinges on. The purpose of the current paper is to point out and discuss these problems and to present possible improvements to avoid (or reduce) the defects of the method in [7]. Since these problems are related to the arrival-time equation (5), we will in the rest of this paper focus attention on arrival times and not on the computation of saturations. To simplify the presentation, we only consider arrival times in two spatial dimension. The extension to three spatial dimensions is straightforward.

The outline of the paper is as follows: In Section 2 we describe the concept of time-of-flight and how this can be used to split the saturation equation in two simpler problems. Then, in Section 2.1 we present the discretization of the arrival-time equation and demonstrate its shortcomings. In Section 2.2 we give a brief outline of the fast-marching method and explain why the marching method of [7] does not perform well for strongly varying velocity fields. In Section 3 we propose an alternative fast(er) solution method for the discretized equations that does not suffer from the same deficiency as the fast-marching-like algorithm employed in [7]. In Sections 3.2 and 3.3, we propose two ways to reduce the large discretization error we have observed. Finally, Section 4 is devoted to discussions relating our numerical approach to modern streamline methods.

## 2. Reformulation of the saturation equation

The marching method in [7] was motivated as a level-set/fast-marching formulation of the saturation equation (1). In this section we present

the method from a slightly different perspective, using the time-of-flight formalism [4], which has made a profound impact upon the development of modern streamline methods. We therefore start by introducing streamlines and the time-of-flight formalism in some detail. (A more thorough introduction is given in [9]).

A streamline $\Psi$ is the path traced out by a passive test particle moving with the flow given by a forcing velocity field $\mathbf{v}$ such that the vector $\mathbf{v}$ is tangential to $\Psi$ at every point. Along each streamline one can reparametrize the space coordinate by introducing the travel time, which is commonly referred to as the *time-of-flight*. In [7] this quantity is referred to as arrival time. The time-of-flight $T(\mathbf{x})$ measures the time it takes a passively particle to travel along a streamline from its initial position on $\Gamma^+$ to a given point $\mathbf{x}$. This travel time can be defined by the following integral along a streamline $\Psi$,

$$T(\mathbf{x}) = \int_{\Psi} \frac{ds}{|\mathbf{v}(\mathbf{x}(s))|}. \tag{6}$$

Thus, for our two-phase model (1)–(2.2), the arrival time gives a picture of the forcing velocity field, while not taking into account the nonlinear effects of $f$ caused by the relative mobilities of water and oil. Alternatively, the arrival time is given by the differential equation (5) (see [7, 9])

$$\mathbf{v} \cdot \nabla T = 1 \text{ in } \Omega, \qquad T = 0 \text{ on } \Gamma^+.$$

To derive this equation, we can consider an infinitesimal movement of the test particle. If $s$ denotes the distance in the direction given by the normalized velocity vector $\mathbf{n} = \mathbf{v}/|\mathbf{v}|$, we have by definition that $\partial s/\partial T = |\mathbf{v}|$, or in other words

$$1 = |\mathbf{v}| \frac{\partial T}{\partial s} = |\mathbf{v}| \left( \nabla T \cdot \frac{\mathbf{v}}{|\mathbf{v}|} \right) = \nabla T \cdot \mathbf{v}.$$

Changing perspective, $T(\cdot)$ maps each level set of $T$ to a positive real number. Thus, $T$ can be understood as the coordinate mapping that transforms (1) into a family of one-dimensional conservation laws, where $T$ takes the role of the space coordinate. To see this, let $T$ be the time-of-flight. If we consider the saturation equation along a streamline for $\mathbf{v}$, it takes the form

$$\frac{\partial \sigma}{\partial t} + |\mathbf{v}| \frac{\partial f(\sigma)}{\partial s} = 0.$$

Then we can substitute the equation for the time-of-flight along a streamline to get

$$\frac{\partial \sigma}{\partial t} + \frac{\partial f(\sigma)}{\partial T} = 0. \tag{7}$$

The data for these one-dimensional problems are derived from the data for (1). If we supply the following initial and boundary values for (1):

$$\begin{aligned} S(\mathbf{x},0) &= S_0(\mathbf{x}) \text{ in } \Omega, \\ S(\mathbf{x},t) &= S_D(\mathbf{x},t) \text{ on } \Gamma^+, \end{aligned}$$

the corresponding data for (7) are

$$\begin{aligned} \sigma(T,0) &= \sigma_0(T) = S_0(\Psi(T(\mathbf{x}))), \\ \sigma(0,t) &= \sigma_D(t) = S_D(\Psi(0),t). \end{aligned}$$

The method in [7] computes the map $T(\mathbf{x})$ for all $\mathbf{x}$, but does not solve for individual streamlines $\Psi$. In other words, the method does not explicitly associate a unique flow-path with each $\mathbf{x}$. Therefore all streamlines are treated as one when computing the saturation, and we must require that the data $\sigma_0$ and $\sigma_D$ are the same for all $\Psi$. This is a slight generalization from [7], where the data was restricted to constant injected fluid mixture $\sigma_D$ and constant initial saturation $\sigma_0(t) = \sigma_0$ in all of $\Omega$. In the present formulation it is evident that we can solve any initial-value problem that has equal data $\sigma_D$ and $\sigma_0$ for all streamlines. Another level-set approach that is capable of treating more general initial-value problems for the saturation equation is presented in [2].

### 2.1. The discrete arrival-time equation

To solve the boundary value problem (5) for the arrival time $T$, an upwind discretization can be constructed once a fluid velocity $\mathbf{v}$ is given by looking in the direction from which the information should be coming. In two spatial dimensions the main discretization used in [7, 8] can be constructed as follows. Let the domain be regularly partitioned into grid blocks of dimensions $\Delta x$ and $\Delta y$ and let the nodes for arrival time be located in the center of each grid block. Assume that we trace a streamline $\Psi$ from the center $\mathbf{x}_k$ of block $k$ to a point $\mathbf{y}$ in the upwind direction $-\mathbf{v}$. We can call the segment of the streamline from $\mathbf{x}_k$ to $\mathbf{y}$ for $\psi_k$. The upwind discretization of (5) can then be written as

$$T_k = T(\mathbf{x}_k) = T(\mathbf{y}) + \Delta T_k, \tag{8.1}$$

$$\Delta T_k = \int_{\psi_k} \frac{ds}{|\mathbf{v}(\mathbf{x}(s))|}. \tag{8.2}$$

To complete the scheme, we must specify how to compute $T(\mathbf{y})$ and how to evaluate the integral (8.2).

The point $\mathbf{y}$ is assumed to lie on the straight line connecting two neighboring nodes $\mathbf{x}_i$ and $\mathbf{x}_j$ of $\mathbf{x}_k$ and is therefore given as $\mathbf{y} = \alpha\mathbf{x}_i +$

$(1-\alpha)\mathbf{x}_j$ for some scalar $\alpha \in [0,1]$. To compute $T(\mathbf{y})$ we use a simple linear interpolation from the nodes $i$ and $j$ near $\mathbf{y}$,

$$T_k = \alpha T_i + (1-\alpha)T_j + \Delta T_k, \tag{9}$$

where $T_i$ and $T_j$ are the arrival times of node $i$ and $j$, respectively. In three dimensions we use a similar bilinear interpolation in the triangle spanned by three neighboring nodes in one of the axial planes.

The upwind discretization gives a consistent way to compute the arrival time in the whole domain. The difference scheme can be written as a linear system of equations

$$\sum_j A_{ij} T_j = \Delta T_i, \qquad i = 1 \ldots N, \tag{10}$$

where $A = \{A_{ij}\}$ contains the coefficients of (9).

Before we take a closer look at the marching algorithm from [7], we will examine how the difference scheme compares to a direct evaluation of (6). To evaluate the integral in (6), we use a standard approximate algorithm for integrating streamlines, as first reported by Pollock [12]. In each grid cell, Pollock's method assumes a piecewise linear approximation of the velocity $\mathbf{v}$ in each direction and uses this to compute streamlines analytically on a cell-by-cell basis. This approximation ensures that different streamline fragments never intersect, or in other words, that we have a consistent approximation of streamlines. For this reason, when evaluating the integral (8.2), we also replace the linear ray approximation of a streamline segment applied in [7, 8] by the more accurate streamline approximation due to Pollock.

We present three numerical test cases: one with a smooth velocity field, one with a mildly heterogeneous velocity field, and one with a nonsmooth velocity field. For all three cases, the solution of the linear system (10) is computed using an iterative solver. To have a measure of the discretization error associated with (9), we compute the residual

$$r = |AT_a - \Delta T|, \tag{11}$$

where $T_a$ is an approximation of (6) evaluated in each node in the grid. This will give us an idea of the accuracy we can expect from the difference scheme (8.1), which was the basis for the marching method in [7].

*Case 1* To demonstrate that the scheme (10) works well for smoothly varying velocity fields $\mathbf{v}$, we first compute $T$ for a homogeneous quarter-five spot. We set $\Omega = [0,1] \times [0,1]$ with no-flow boundaries. In the lower-left corner we place an injector and in the upper-right corner we

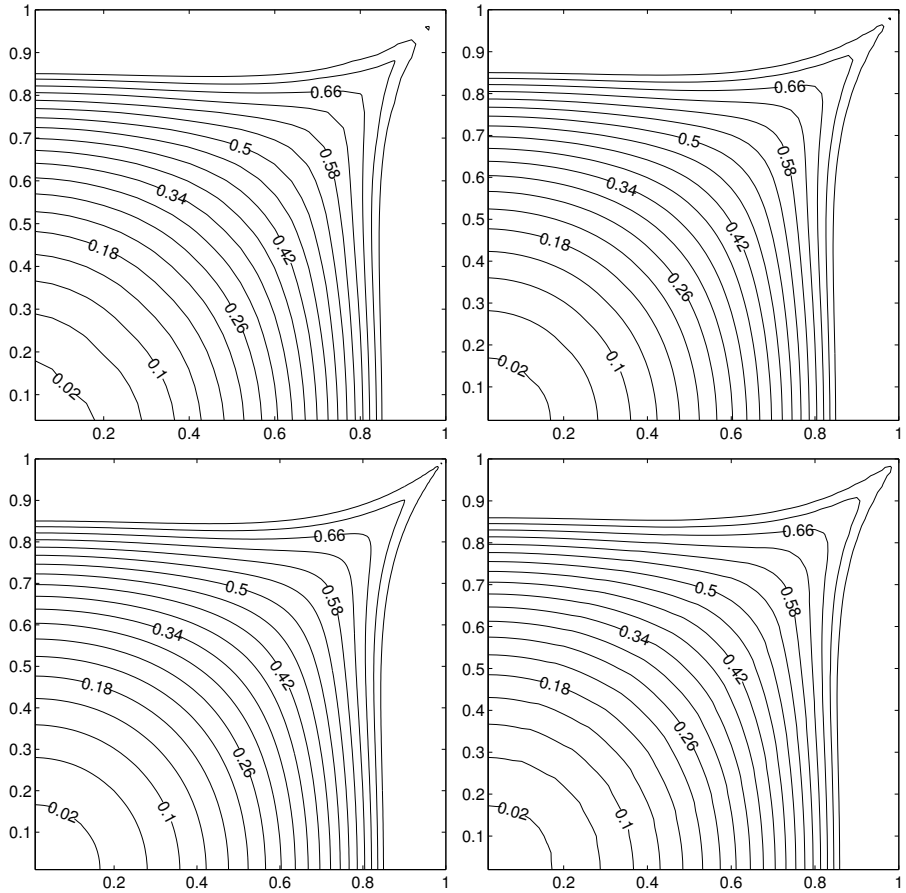


*Figure 1.* Convergence study for the homogeneous quarter-five spot (Case 1) with mesh spacing $h = 0.04$ (upper left), $h = 0.02$ (upper right), and $h = 0.01$ (lower left). The lower-right plot shows the arrival time computed by direct evaluation of (6) with $h = 0.02$. Contours are only drawn for $T \leq 0.74$.

place a producer. These are modeled as point sources with intensity $+1.0$ and $-1.0$ respectively. This means that one pore volume of fluid is injected per time unit. The corresponding data for the arrival-time equation (5) is a homogeneous Dirichlet boundary $\Gamma^+$ around the injector. The velocity field $\mathbf{v}$ is computed using Darcy's law (2.2). To discretize (5) we have used mesh spacing $\Delta x = \Delta y = h$. Figure 1 shows the solution of the linear system (10) for three different grid resolutions: $h = 0.04$, $h = 0.02$, and $h = 0.01$. For comparison, we have also computed the time-of-flight by direct evaluation of (6). The streamlines $\Psi$ have been approximated using Pollock's method [12] with mesh spacing $h = 0.02$. As we can see from the right column in Figure 1, the two solution strategies are almost visually indistinguishable

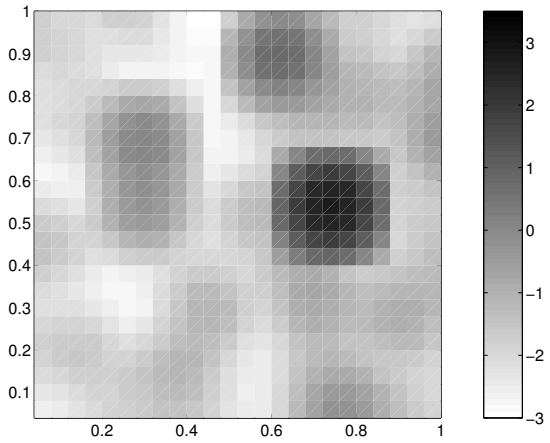


*Figure 2.* Logarithm of the permeability field for Case 2. The permeability is given on a grid with $h = 0.04$. This permeability field is used for the finer grids as well.

as expected. From the three plots, we observe that the discretization (8.2)–(9) seems to converge as the grid is refined. In Figure 5 we have plotted the discretization error (11) for $h = 0.02$. As we can see, the error is small throughout the domain.

*Case 2* In our second test case we consider a quarter-five spot problem as in Case 1, but now with a heterogeneous permeability field as shown in Figure 2. The permeability varies between 40 mD and 15 D. Figure 3 shows solutions of (10) on three different grids compared with an approximation obtained by direct evaluation of the streamline integrals (6). Again, we observe that our upwind scheme seems to converge as the grid is refined. However, the convergence is a bit slower than in Case 1. In Figure 5 we have plotted the discretization error (11) for $h = 0.02$. The error is clearly larger than for Case 1, but still of order $h$.

*Case 3* Our third test case is taken from [7] and describes flow in a porous cross-shaped channel defined on $\Omega = (0, 1) \times (0, 1)$. The permeability is equal 100 mD in the beams of the cross and 0.01 mD outside. Injectors are placed in the lower and left-hand beams and producers are placed in the upper and right-hand beams. The corresponding flow field is almost discontinuous along the walls of the channel and smooth inside. We compute $T$ on three uniform grids with spacing $h = 0.02$, $h = 0.01$, and $h = 0.005$, respectively, by solving the linear system (10) and compare the results with an approximation to (6) as before. Figure 4 shows contour plots of the computed time-of-flights. As we can

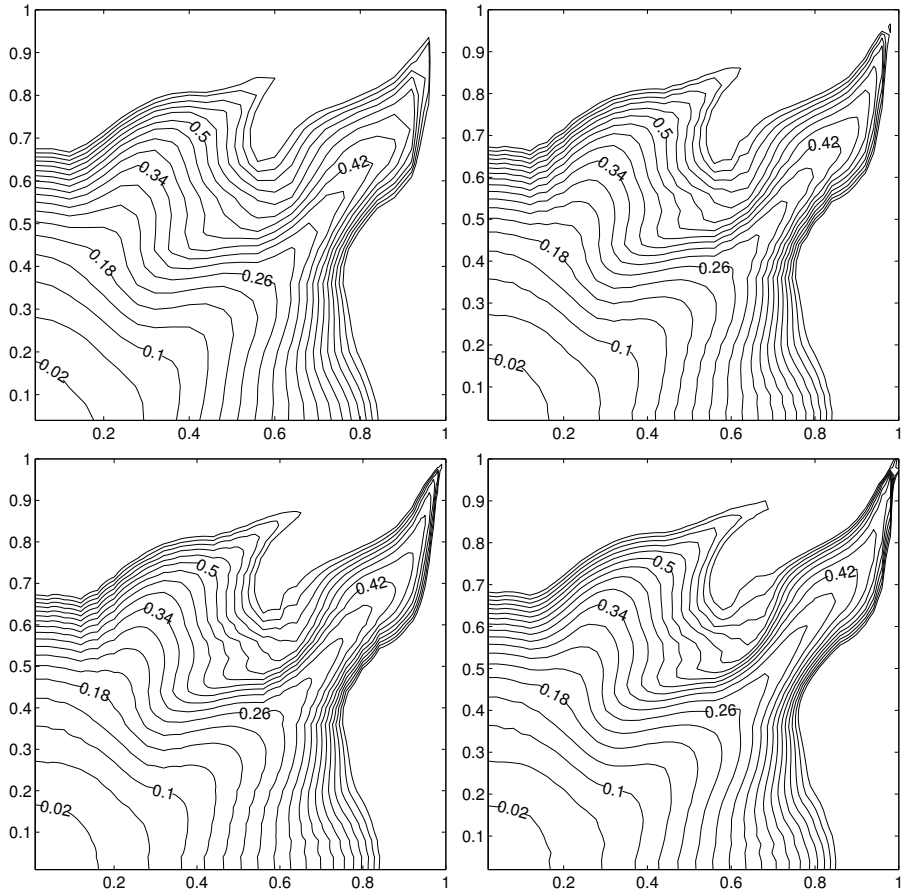


*Figure 3.* Convergence study for the heterogeneous quarter-five spot (Case 2) with $h = 0.04$ (upper left), $h = 0.02$ (upper right), and $h = 0.01$ (lower left). The lower-right plot shows the arrival time computed by direct evaluation of (6) with $h = 0.02$. Contours are only drawn for $T \leq 0.70$.

see, the solution computed using the difference scheme is reasonably accurate in some parts of the channel, and wrong in others.

Figure 5 gives a plot of the associated residual (11) for $h = 0.1$. The residual is dominated by large discretization errors on the diagonal and in six narrow layers of grid blocks along the walls of the channel. These layers coincide with the part of the wall where the difference formula (9) includes a node both inside and outside the high-permeability channel, or in other words, where the difference scheme has a stencil that covers a region where $T$ has a large jump! Thus the discretization error increases from $\mathcal{O}(h)$ to $\mathcal{O}(1)$ in a layer of cells along the wall. The large discrepancy observed in Figure 4 between the difference scheme and

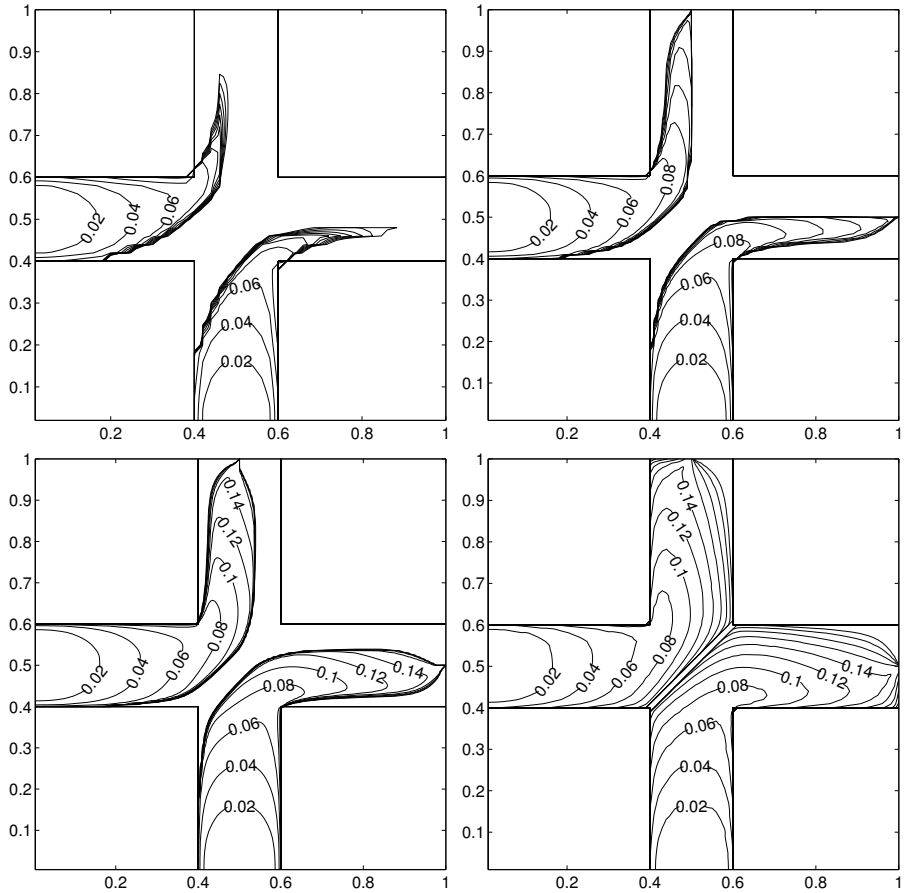


*Figure 4.* Convergence study for the channel problem (Case 3) with $h = 0.02$ (upper left), $h = 0.01$ (upper right), and $h = 0.005$ (lower left). The lower-right plot shows the arrival time computed by direct evaluation of (6) with $h = 0.01$. Contours are only drawn for $T \leq 0.24$.

the direct evaluation of (6) is caused by the discretization errors along the walls, and these errors are convected in the direction of the flow.

The numerical results for Cases 1 to 3 demonstrate that while the difference scheme is accurate for smooth data, it has low accuracy for layered permeability fields with high permeability contrasts. In cases where the fluid velocity is nearly parallel to the layers, there will be a large shear in the velocity that makes interpolation of time-of-flight between adjacent streamlines inaccurate. In addition, any errors that occur in the computation accumulate in the direction of flow. Thus the arrival time will be more and more smeared out downstream. Note also that if these arrival times were to be used to compute saturations, the resulting computations would have large mass balance errors. Case 3

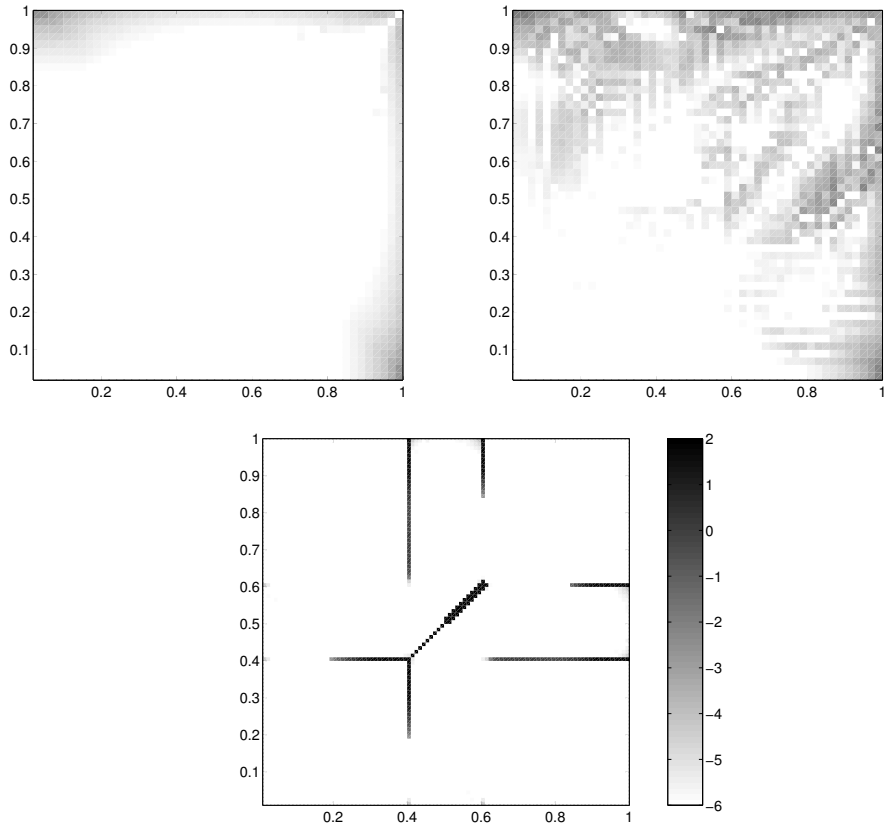

*Figure 5.* Logarithm of the residual residual (11) for the three test cases: Case 1 with $h = 0.02$ (upper left); Case 2 with $h = 0.02$ (upper right); and Case 3 with $h = 0.01$ (bottom). For Case 3, the residual is only plotted inside the channel.

therefore demonstrates that for certain boundary value problems with large jumps in the velocity, the method (10) is inferior to methods that approximate (6) directly. A fast method based on reordering of unknowns will, if consistent, give no better results than a direct solution of (10), for the discretization (8.1).

At the time of publication of [7, 8] these problems were unknown to the authors. In fact, the early results obtained by the new method looked very promising. We will come back to the reason why the problems went unnoticed at the end of the next section, where we discuss another fundamental problem with the approach in [7].

## 2.2. The fast-marching method

In [7] the difference scheme (9) was solved using a marching method similar to the standard fast-marching method [13, 16, 17, 6]. In this

section, we explain why this method method may be inappropriate for solving the arrival-time equation (5).

Originally, the fast-marching method was developed as an efficient method to compute the solution of the Eikonal equation,

$$|\nabla\phi| = \frac{1}{c(\mathbf{x})}, \text{ in } \Omega \setminus \Omega_0, \qquad \phi = 0 \text{ on } \partial\Omega_0, \tag{12}$$

where $\Omega_0$ is some reasonable subset of $\Omega$ and $c(\mathbf{x})$ is the local wave speed. The Eikonal equation is the first-order term in a geometrical-optics approximation to the wave equation. It characterizes the motion of wave fronts originating from $\partial\Omega_0$. At any time $t \geq 0$, the shape and position of the wave front is given by the contour $\{\phi = t\}$. Equation (12) states that the wave front propagates in its normal direction with the spatially varying speed $c$. The front may develop corners and cusps even for smooth wave-speeds $c$ and initial domains $\partial\Omega_0$. The characteristics of the Eikonal equation are perpendicular to the level sets of $\phi$, and $\phi$ is strictly increasing along the characteristics. Thus, consider a characteristic $\psi$ within the simplex spanned by three points, say $\mathbf{x}_{ij}$, $\mathbf{x}_{i-1,j}$, $\mathbf{x}_{i,j+1}$. If the wave speed $c$ is constant within the simplex, then $\psi$ will be a straight line. It follows that if $\psi$ passes through $\mathbf{x}_{ij}$ and $\phi$ is increasing along $\psi$ toward $\mathbf{x}_{ij}$, then

$$\phi_{ij} \geq \max(\phi_{i-1,j}, \phi_{i,j+1}). \tag{13}$$

This property is referred to as *causality* in the fast-marching terminology, and depends only on the assumption that each such simplex in the grid has no obtuse angles. To solve equation (12) on a Cartesian grid $\Lambda = \{i\Delta x, j\Delta y\}$, we set $\phi_{ij} = \phi(i\Delta x, j\Delta y)$ and use the following first-order upwind-discretization

$$\sqrt{\max(D_{ij}^{-x}\phi, -D_{ij}^{+x}\phi, 0)^2 + \max(D_{ij}^{-y}\phi, -D_{ij}^{+y}\phi, 0)^2} = \frac{1}{c_{ij}}, \tag{14}$$

where $D_{ij}^{+x}\phi = (\phi_{i+1,j} - \phi_{ij})/\Delta x$, $D_{ij}^{-x}\phi = (\phi_{ij} - \phi_{i-1,j})/\Delta x$, etc. The extension to three spatial dimensions is straightforward. A simple method to solve this set of equations would be to apply a fixed-point iteration. However, based on the above observations one can do better.

The causality property ensures that the solution of the upwind difference scheme can be computed in one node at a time by applying (14) to the unknowns $\{\phi_{ij}\}$ in the order of increasing values of $\phi_{ij}$. In other words, we can compute the solution of (14) in $N$ operations if we can determine an optimal ordering of the unknowns. This optimal ordering is characterized by increasing values of $\phi_{ij}$. The fast-marching method hinges on this idea to construct the solution as an advancing front.

Introducing a narrow-band strategy, the solution can be computed in $\mathcal{O}(N \log N)$ operations, where $\log N$ is the cost of the ordering. For a throughout description of the fast-marching method and many of its applications, we refer to the books [11, 15].

In [7], the apparent similarities between the arrival equation (5) and the Eikonal equation (12) led the authors to apply the fast-marching method to (5). To justify this, the arrival-time equation was written as

$$|\nabla T| = \frac{|\nabla T|}{\mathbf{v}(\mathbf{x}) \cdot \nabla T} = \frac{1}{F(\mathbf{x})}.$$

This reformulation hides an important fact: Whereas the speed function $c$ in (12) is isotropic, i.e., independent of orientation, $F$ depends on $T$ and is therefore anisotropic. Furthermore, $F$ is not bounded away from zero and $\nabla T$ can be almost perpendicular to $\mathbf{v}$. If this is the case, the correct value of $T_k$ may be smaller than the maximum of the two values $T_i$ and $T_j$ used to compute $T_k$ according to (9). In other words, $T_k$ may depend on values that are yet to be computed by the marching algorithm. Thus, the causality property does not in general hold for the discretization (9) of (5), and we cannot assume that the upwind scheme is decoupled by arranging the unknowns in the order of increasing $T$. Notice that the arrival-time equation (5) *has* a causality principle along streamlines, but not across streamlines. This rules out the use of the fast-marching method to solve (10). On reasonably smooth velocity fields, this defect in [7] does not severely affect the fast-marching solution of (10) as the results in [7] show. However, as the next example shows, for nonsmooth velocities, the error is significant.

*Case 1 to 3, revisited* We apply the marching method in [7] to Case 1 to 3 in Section 2.1. If the method is correct, we would expect to get identical results as with a direct solution of the linear system (10). Contour plot of the arrival times computed with the marching method, together with a direct solution of the linear system (10), are shown in Figure 6 for the three cases. As one can see, the method produces wrong solutions for all three tests, and the errors become more pronounced for the heterogeneous and layered Case 2 and Case 3.

The reason for these errors is, as has been pointed out, that the *causality* that the fast-marching method is based on, does not hold for the discretized arrival-time equation (5). The causality breaks down when the fluid velocity is nearly perpendicular to $\nabla T$. When this happens, the nodes $i$ and $j$ in the difference formula (9) of $T_k$ may have larger arrival time than $T_k$. Thus, $T_k$ may be computed based on values $T_i$ and $T_j$ that are not yet correct. To illustrate this in Figure 6,

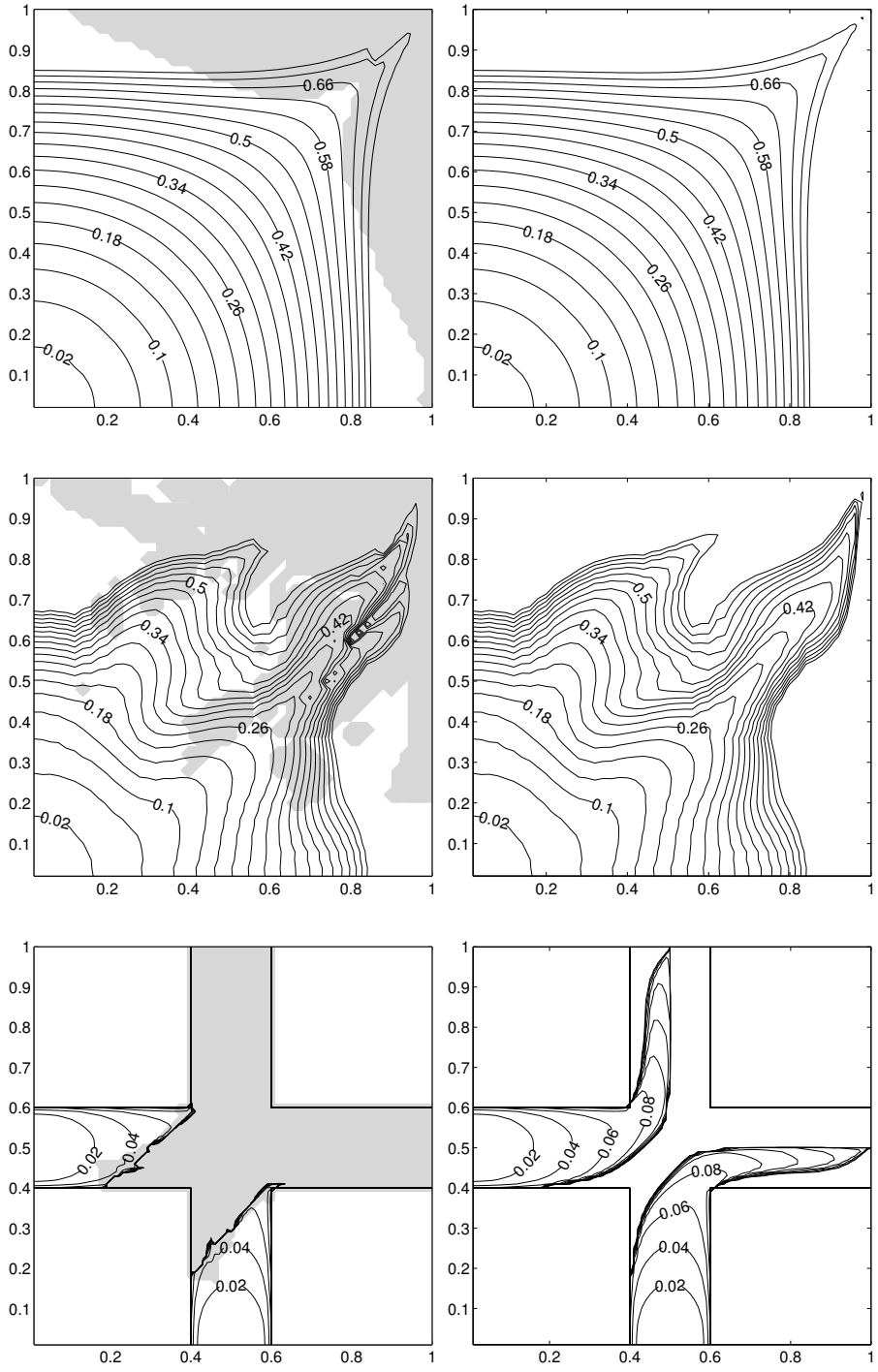


*Figure 6.* Arrival times computed with the marching method (left) and by direct solution of (10) (right). From top to bottom, the plots correspond to Cases 1–3. The shaded regions correspond to nodes where the causality property (13) is violated.

we have shaded the regions where the causality principle (13) is not fulfilled, i.e., where $T_k < \max(T_i, T_j)$.

These problems went unnoticed in [7, 8] due to an unforeseen side-effect of a trick used to speed up the computations of saturations. In [7] the saturation computation was viewed as the numerical evolution of a level set. The authors observed that in order to compute the saturation at a given point in time $t$, one only needs to compute the arrival times for nodes that have been passed by the saturation front; that is, nodes that have time-of-flight less than or equal to $T_{max} = t \max f'$. For the other (unflooded) nodes, the saturation is given by the initial saturation distribution. Thus, some nodes were left undefined in the algorithm (as they were not needed). The algorithm in [7] filled these undefined nodes with a value $T_{\max} + \epsilon$, where $\epsilon$ is an arbitrary small number. If the flooded nodes had fulfilled the causality principle, that is, had they only depended on values with lesser $T$, this fill-inn would have had no effect. However, although the causality principle is true for the *continuous* problem, the discretization stencil frequently involves undefined nodes corresponding to larger time-of-flights, as seen in Figure 6.

The effect of the fill-in was that the magnitudes of the largest (non-computed) arrival times were significantly reduced. This corresponds almost to a one-sided interpolation of $T(\mathbf{y})$ from the "correct" side. Therefore the results of the example in Section 5.3 of [7] look seemingly reasonable. However, the results reported in [7] are *not* solutions of the linear system (10).

## 3. An improved method

The question remains: Is it possible to repair the deficiencies in the marching method from [7] ? The answer is yes and no. The problem with lack of causality in the fast-marching method can be circumvented. In the current section we present a new advancing-front method for the linear system (10), which will in fact be faster than the original marching method.

The inaccuracies in the interpolation scheme in the presence of large jumps in the arrival times are harder to eliminated in a good manner. In Section 3.2 we present one alternative based upon tracking longer local streamlines in the upwind direction. The method is accurate, but expensive. Then in Section 3.3 we present an alternative method based upon using a zeroth-order interpolation in the presence of large jumps in the local interpolation stencil. This approach gives no extra work, but reduces the formal accuracy locally.

### 3.1. Advancing front solution

To improve the results from the previous sections, we have implemented a different strategy for solving (5). First, as discussed in Section 2.1, we replace the linear ray approximation of a streamline segment by a more accurate streamline approximation due to Pollock [12].

Since the upwind scheme (9) can be constructed a priori, the corresponding discretization of (5) can be written as a system of linear equations. We know that the velocity field $\mathbf{v}$ is divergence free and irrotational, from which it follows that no nodes are mutually dependent. For instance, if $T_k$ is computed from $T_i$ and $T_j$, then neither $T_i$ nor $T_j$ will depend on $T_k$. From this we can deduce that it will be possible to compute the solution $T$ one node at a time; that is, there must exists a reordering of the unknowns that renders the system of linear equations (10) triangular. If we can determine this ordering, we can construct the solution of (10) directly.

The appropriate reordering can be found by traversing the nodes in the Cartesian grid $\Lambda$ as an advancing front. Assume that we know the correct time-of-flight in a subset $\mathcal{A} \subset \Lambda$ of nodes. Initially, this set is the set of nodes lying on the Dirichlet boundary $\Gamma^+$. In addition, there will be some nodes that depend on nodes in $\mathcal{A}$; this set of nodes will be denoted $\mathcal{F}$. In $\mathcal{F}$ there are nodes that depend *only* on nodes in $\mathcal{A}$. If we apply upwind update (9) to these nodes, we get the correct solution of (10), and we can consider the values in these nodes known. This is the basic idea that we apply to construct the solution in $N$ steps.

To be more precise, in each step of the algorithm we pick a node $k \in \mathcal{F}$ that depends only on nodes in $\mathcal{A}$ through (9). In other word, if $T_k$ depends on $T_i$ and $T_j$ in (9), then $\{i, j\} \subseteq \mathcal{A}$. Thus, if we apply (9) to node $k$, we can be certain that the computed value $T_k$ is the $k$'th component of the solution of (10). To update $\mathcal{F}$ and $\mathcal{A}$ we then move $k$ from $\mathcal{F}$ to $\mathcal{A}$ and add to $\mathcal{F}$ any nodes that depend on $k$. If we proceed in this manner, we can construct the solution in all of $\Lambda$ in $\mathcal{O}(N)$ operations.

In a sense, this advancing front approach is a combination of a reordering algorithm and a back-substitution. Note that the work required for this approach is considerably reduced compared with the $\mathcal{O}(N \log N)$ operations required by the marching algorithm in [7]. In this algorithm, the $\mathcal{F}$-nodes are sorted in ascending order with respect to estimated arrival times, and the next node processed in the algorithm is always the one with the lowest estimated arrival time. This sorting requires in general $\mathcal{O}(\log N)$ operations. In the advancing front algorithm, no sorting is required since we pick the next node at random.

The advancing-front algorithm reads:

$\mathcal{A} = \{\text{Nodes on the boundary } \Gamma^+\}$
$\mathcal{F} = \{\text{All nodes in } \mathcal{A}^C \text{ that depend on nodes in } \mathcal{A}\}$
**repeat**
  **repeat**
    pick some (random) $k \in \mathcal{F}$
    $\mathcal{S} = \{\text{all nodes in stencil of } k\}$
  **until** $\mathcal{S} \subseteq \mathcal{A}$
  $\mathcal{F} = \mathcal{F} \setminus \{k\}$
  $\mathcal{A} = \mathcal{A} \cup \{k\}$
  apply (9) to $T_k$
  $\mathcal{F} = \mathcal{F} \cup \{\text{nodes that depend on } k\}$
**until** $\mathcal{F} = \emptyset$

The algorithm will terminate if the velocity field has zero rotation and sufficient boundary data is described. At every step in the algorithm, $\mathcal{A}$ is the set of nodes for which the solution is computed, while unknowns in $\mathcal{A}^C$ have not been computed yet. The set $\mathcal{F} \subseteq \mathcal{A}^C$ depends on $\mathcal{A}$ through (9). By construction, this algorithm computes the correct solution of (10) in $N$ operations. The reason for this is that the "pick" operation in the fifth line can be performed in $\mathcal{O}(1)$ operations. Note also that if we order the unknowns in the order that they are processed in the outer loop, we get a triangular system.

*Case 3, revisited* We can now recompute the solution of (10) for the channel-flow problem with the advancing-front algorithm instead of the iterative linear solver. On all three grids, the advancing-front algorithm gives identical results with no discrepancy in the sup norm.

### 3.2. Longer streamline fragments

In the previous section we introduced a new advancing-front algorithm that computes the correct solution to the linear system (10) in $\mathcal{O}(N)$. We have thus cured one of the deficiencies of the fast-marching-like method of [7]. The remaining problem related to the discretization error is not as easy to cure. This problem is essentially a problem of too low grid resolution. Our first attempt to improve the method is based on adaptivity: We wish to trace streamline segments $\psi$ backward until a sufficiently accurate interpolation can be made. Since the time-of-flight is actually an integrated quantity, the solution does not only depend on local properties of the medium, but also on properties 'upstream' along a streamline. Therefore $T$ can have large jumps transverse to the streamlines; over one grid block $T$ can be discontinuous to grid resolution even if the permeabilities locally is continuous. Since we have

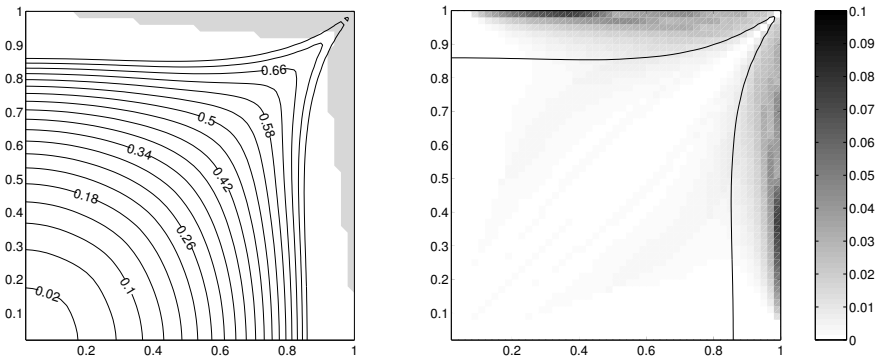


*Figure 7.* Arrival times for the homogeneous quarter-five spot (Case 1). The left plot shows the solution computed by the adaptive algorithm with $C = 0.1$. Contours are drawn for $T \leq 0.74$. The shaded region shows nodes that have been updated by using longer streamline fragments. The right plot shows the difference in solutions computed by the adaptive scheme and by direct evaluation of (6), together with a plot of the contour $T = 0.74$ for the direct evaluation. The mesh spacing is $h = 0.02$.

no precise *a posteriori* error measure for the difference scheme (9) the adaptive approach must necessarily involve some ad hoc features.

The proposed algorithm is essentially the same as the advancing front approach, with one exception: For each unknown that is to be computed, we trace a streamline backwards until the interpolation seems to be reasonably accurate. A very rough indicator of whether the discretization error is large or small is to compare $\Delta T_k$ of (8.2) to $|T_i - T_j|$ in (9). If $\Delta T_k > C|T_i - T_j|$ for some fixed $C$ we accept the the update formula based on the nodes $i$ and $j$. Otherwise, we trace the streamline $\psi_k$ further back. We emphasize that this indicator for the interpolation error is ad hoc and does not have a rigorous basis.

*Cases 1–3, revisited* We apply the improved method of Sections 3.1 and 3.2 to our three test problems. For all the three cases we let $C = 0.1$, and, as an additional test, we also try $C = 1.0$ for the heterogeneous quarter-five spot (Case 2). The left columns in Figures 7, 8 and 9 show contour plots of corresponding approximate arrival times. For extra illustration, we have shaded the regions where nodes have been updated using longer streamline fragments. In the right columns in Figures 7–9, we have plotted the difference between the solutions computed by the adaptive scheme and by direct evaluation of (6). In the regions where the difference between the two solutions is larger than $\mathcal{O}(h)$, the nodes generally have large arrival times.

The results demonstrate that the adaptive scheme gives reasonably good results, both for the channel problem and for the heterogeneous

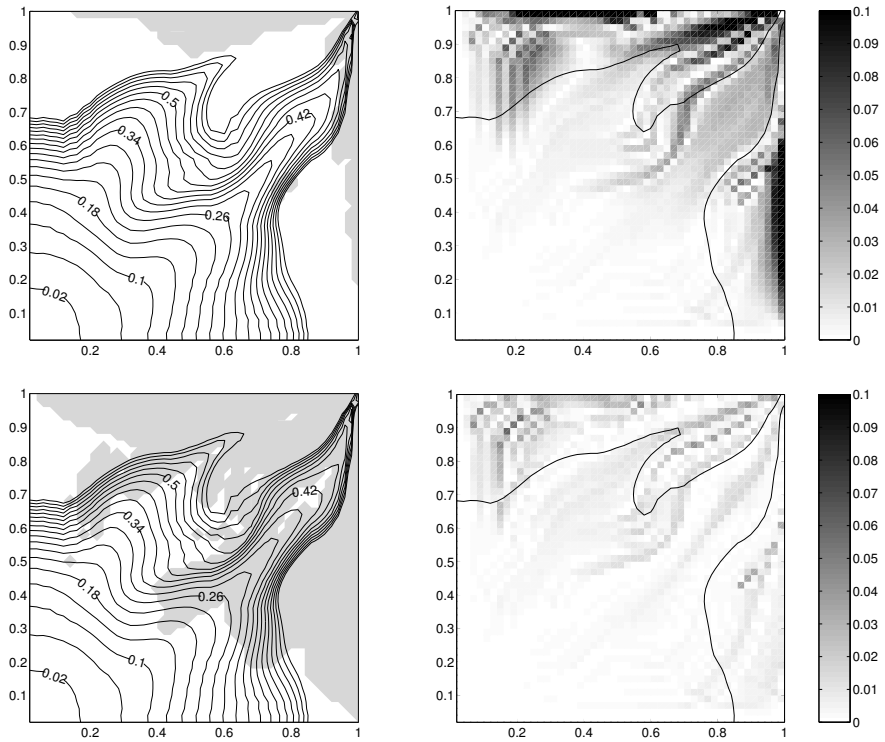


*Figure 8.* Arrival times for the heterogeneous quarter-five spot (Case 2). The plots in the left column show solutions computed by the adaptive algorithm. Contours are drawn for $T \leq 0.70$. The shaded regions show nodes that have been updated by using longer streamline fragments. The plots in the right column show the difference in solutions computed by the adaptive scheme and by direct evaluation of (6), together with a plot of the contour $T = 0.70$ for the direct evaluation. The mesh spacing is $h = 0.02$. The threshold parameters are $C = 0.1$ (upper row) and $C = 1.0$ (lower row).

medium. However, the added computational cost of the adaptive algorithm is quite high. As expected, tracking of longer streamlines is mostly initiated in areas where $\nabla T$ is almost perpendicular to $\mathbf{v}$. In other words, the added work is spent in regions where there is a large shear in the flow. These areas seem to be of little significance in the early stages of a reservoir production, since the corresponding nodes are flooded long after water breakthrough. Another problem with this method is that the parameter $C$ is hard to specify in advance. For a specific example, one may adjust the accuracy by adjusting the parameter, but there is no guarantee that this $C$-value will yield good results for a different example. Thus, the $C$-parameter may have to be tuned

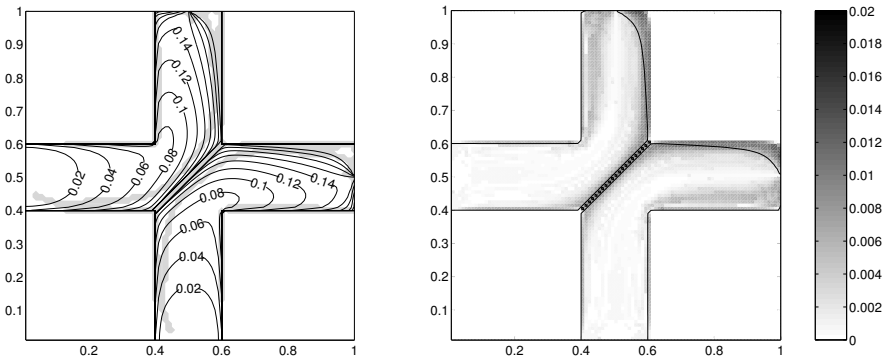


*Figure 9.* Arrival times for the channel-flow problem (Case 3). The left plot shows the solution computed by the adaptive algorithm with $C = 0.1$. Contours are drawn for $T \leq 0.24$. The shaded region shows nodes that have been updated by using longer streamline fragments. The right plot shows the difference in solution computed by the adaptive scheme and by direct evaluation of (6) together with a plot of the contour $T = 0.24$ for the direct evaluation. The mesh spacing is $h = 0.01$.

on a case-to-case basis. To be practical, the method must include a more precise error measure that is inexpensive to compute.

### 3.3. Local zeroth order interpolation

As we have seen earlier, it is possible that a node is assigned a smaller arrival time than one of the nodes in the interpolation scheme when arrival times are computed with the advancing-front algorithm from Section 3.1. Even though the causality that the algorithm is based on is not broken, the physical "upwinding" for the flow is. However, this will not result in large interpolation errors unless $\mathbf{v}$ is close to perpendicular to $\nabla T$. These regions can be detected by our previously given error measure.

As an attempt to cure the problem of discretization error in a manner that will not affect the efficiency of the algorithm, we can use local interpolation of zeroth order for nodes where the error measure for the interpolation is large.

*Case 3, revisited* Figure 10 shows results obtained using the adaptive algorithm with zeroth order interpolation for the channel-flow problem. The results demonstrate that even though we locally reduce the order of accuracy for the interpolation, the resulting scheme gives visually good results even for the most challenging of our test cases.

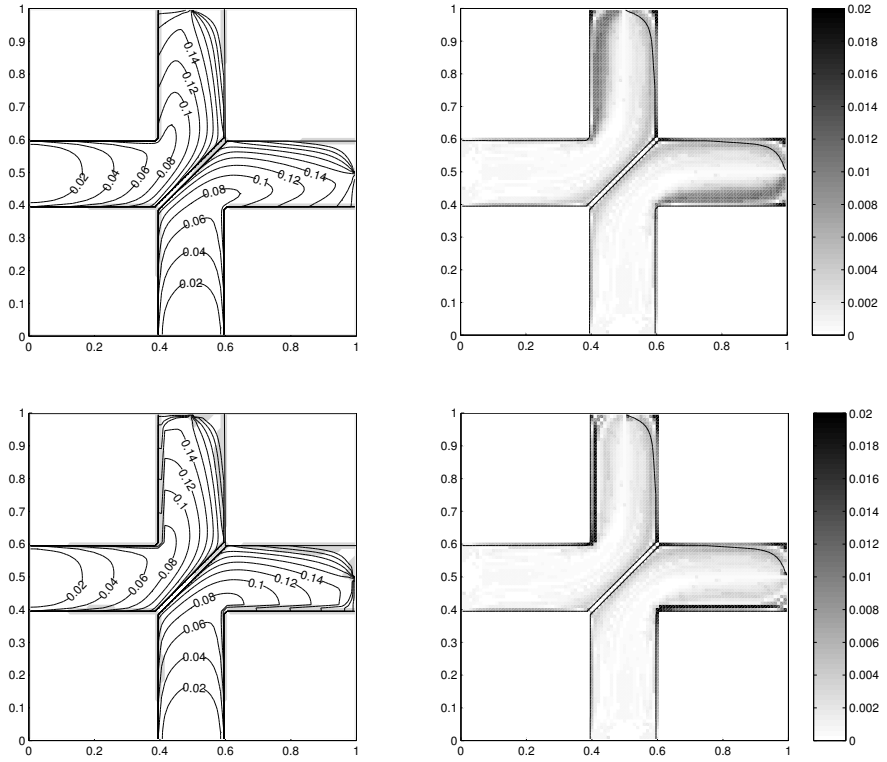

*Figure 10.* Arrival times for the channel-flow problem (Case 3). The left column shows solutions computed by the adaptive algorithm with zeroth order interpolation and error threshold $C = 0.01$ (upper row) and $C = 0.1$ (lower row). Contours are drawn for $T \leq 0.24$. The shaded regions show nodes that have been updated using zeroth order interpolation. (right) Difference between the adaptive scheme and the direct evaluation of (6) together with a plot of the contour $T = 0.24$ for the direct evaluation. The mesh spacing is $h = 0.01$.

## 4. Discussion and relation to streamline methods

In this paper we have carefully investigated and extended the fast saturation solver presented in [7]. In our analysis, we have interpreted the method in terms of the time-of-flight formalism. It is evident from this interpretation that there is a clear relation between streamline methods and our marching method(s). Therefore, it is natural to compare the possibilities and limitations of the marching methods presented herein with those of streamline methods.

The original idea in [7] was to develop a fast saturation solver for immiscible, incompressible, two-phase flow. In this respect, the marching and advancing-front method have limited applicability. The reason is

that the splitting technique applied in [7] and in this paper is limited to initial-boundary-value problems with an inherent (radial) symmetry in terms of the time-of-flight coordinate. This symmetry ensures that the three-dimensional initial-boundary-value problem for the saturation equation can be decomposed into a single one-dimensional conservation law for the behavior along any streamline and the computation of an appropriate coordinate transform. This coordinate transform allows us to map the one-dimensional solution into three-dimensional space. The approach presented herein differs from streamline methods mainly in the calculation of the coordinate transform. For streamline methods the transform is obtained by tracing individual streamlines explicitly. In the current paper, the coordinate transform is found by solving the arrival-time equation. As demonstrated, it is possible to construct special-purpose solvers for this equation, which seem to be fast and reliable for permeability fields that are either homogeneous or vary relatively smoothly. However, for layered media with high permeability contrasts and flow parallel to the layers, the marching method produces inaccurate solutions. The reason for this is that the time-of-flight is less smooth for these problems resulting in large discretization errors.

As these problems are caused by discretization errors, we have proposed an adaptive scheme that changes the difference formula in the presence of large variations in time-of-flight. The adaptivity reduces the problems with the discretization errors by avoiding interpolation in regions where an interpolation will result in large discretization errors, typically where the gradients in time-of-flight are nearly perpendicular to the fluid velocity. The adaptive scheme based upon tracing streamlines longer into the upstream direction seems to be able to better compute time-of-flight than a fixed difference scheme, but the adaptivity is quite expensive. Most of the extra work is spent in regions with high arrival times due to the rough error measure used here. Because of the extra work, the speed of the method is reduced, and is probably comparable to, if not slower than other methods like streamline methods. A more accurate error measure is therefore needed to make the method practical. The adaptive method based upon zeroth order interpolation is as fast as the fixed-stencil method, and although the formal accuracy of the interpolation may be reduced locally, the effect on the overall accuracy seems to be small. The method may therefore be used as a fast grid-based method for computing time-of-flight and derived quantities (volumetric sweeps, drainage volumes, reservoir compartmentalisation, etc) in heterogeneous porous media.

Streamline methods share many properties with the advancing-front method and the method presented in [7], but do not suffer from the same limitations. To be more specific, streamline methods map one-

dimensional solutions back to three-dimensional space by computing particle paths (streamlines). These computations make streamline methods more expensive than the fast finite-difference solver presented in Section 3.1. However, the computation of streamlines is well-behaved for all the examples presented in the paper, and streamline methods will therefore resolve the time-of-flight accurately in all examples. Furthermore, the tracking of individual streamlines leaves room for more general problems where a different one-dimensional solution must be computed for each path. Thus, if one is interested in simulating flow of more than one phase, methods based on integration along streamlines are more robust and general than the new advancing-front method. If one, on the other hand, is merely interested in computing time-of-flight (and derived quantities), the question is more open.

## Acknowledgment

The research of Karlsen, Lie, and Natvig was supported by the BeMatA program of the Research Council of Norway. Karlsen was also supported in part by the European network HYKE, funded by the EC as contract HPRN-CT-2002-00282

## References

1. K. Aziz and A. Settari. *Petroleum reservoir simulation*. Elsevier Applied Science Publishers, Essex, England, 1979.
2. I. Berre, H. K. Dahle, K. H. Karlsen, K.-A. Lie, and J. R. Natvig. Time-of-flight + fast marching + transport collapse: an alternative to streamlines for two-phase porous media flow with capillary forces? *Proc. CMWR XIV*, Volume 2, Elsevier, Amsterdam (2002), pages 995–1002.
3. G. Chavent and J. Jaffre. *Mathematical models and finite elements for reservoir simulation*, vol. 17 of *Studies in mathematics and its applications*. North Holland, Amsterdam, 1986.
4. A. Datta-Gupta and M. J. King. A semi-analytic approach to tracer flow modeling in heterogenous permeable media. *Advances in Water Resources*. Res.,18(1):9–24, 1995.
5. M. S. Espedal and K. H. Karlsen. Numerical solution of reservoir flow models based on large time step operator splitting algorithms. In volume 1734 of *Lecture Notes in Mathematics*, pages 9–77. Springer, Berlin, 2000.
6. J. Helmsen, E. G. Puckett, P. Colella, and M. Dorr. Two new methods for simulating photolithography development in 3d. *Proc. SPIE*, 2726:253–261, 1996.
7. K. H. Karlsen, K.-A. Lie, and N. H. Risebro. A fast marching method for reservoir simulation. *Comp. Geo.*, 4(2):185–206, 2000.
8. K. H. Karlsen, K.-A. Lie, J. R. Natvig, and N. H. Risebro. A fast marching method for 3D reservoir simulation. *Proc. NSCM-13*, Mechanics and Applied

Mathematics Series, No. 7, pp. 147–150, University of Oslo, Norway, October 2000.
9. 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.
10. 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.
11. S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer Verlag, New York, 2002.
12. D. W. Pollock. Semianalytical computation of path lines for finite difference models. *Ground Water* 26(6):743–750, 1988.
13. 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.
14. J. A. Sethian. Fast marching level set methods for three-dimensional photolithography development. *Proc. SPIE*, 2726:261–272, 1996.
15. J. A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, Cambridge, 1999.
16. J. Tsitsiklis. Efficient algorithms for globally optimal trajectories. In *Proceedings of IEEE 33rd Conference on Decision and Control, Lake Buena Vista, Fl, Dec. (1994)*, pages 1368–1373.
17. J. N. Tsitsiklis. Efficient algorithms for globally optimal trajectories. *IEEE Transactions on Automatic Control*, 40(9):1528–1538, 1995.