An Efficient Discontinuous Galerkin Method for Advective Transport in Porous Media ⋆

Jostein R. Natvig [a,*] Knut–Andreas Lie [a,b] Birgitte Eikemo [c] Inga Berre [c]

[a] *SINTEF ICT, Department of Applied Mathematics, P.O. Box 124 Blindern, NO–0314 Oslo, Norway*

[b] *Centre of Mathematics for Applications, University of Oslo, P.O. Box 1053 Blindern, NO–0316 Oslo, Norway*

[c] *University of Bergen, Deptartment of Mathematics, Johs. Brunsgt. 12, NO-5008 Bergen, Norway*

Abstract

We consider a discontinuous Galerkin scheme for computing transport in heterogeneous media. An efficient solution of the resulting linear system of equations is possible by taking advantage of *a priori* knowledge of the direction of flow. By arranging the elements in a suitable sequence, one does not need to assemble the full system and may compute the solution in an element-by-element fashion. We demonstrate this procedure on boundary-value problems for tracer transport and time-of-flight.

Key words: transport in porous media, time-of-flight, discontinuous Galerkin discretization, linear solvers, directed asyclic graphs

⋆ The research was funded in part by the Research Council of Norway under grants no. 139144/431 and 158908/I30.

* Corresponding author.

Email addresses: Jostein.R.Natvig@sintef.no (Jostein R. Natvig), Knut-Andreas.Lie@sintef.no (Knut–Andreas Lie), birgitte@mi.uib.no (Birgitte Eikemo), ingab@mi.uib.no (Inga Berre).

1 Introduction

In this paper, we consider efficient and accurate methods for a class of linear equations on the form

$$\begin{aligned} \mathbf{v} \cdot \nabla u &= H(u, \mathbf{x}), && \text{for } \mathbf{x} \in \Omega, \\ u &= h(\mathbf{x}), && \text{for } \mathbf{x} \in \partial\Omega^-, \end{aligned} \tag{1}$$

where $\mathbf{v}$ is a given (divergence-free) vector field and $\partial\Omega^-$ denotes the inflow boundary of Ω. Our motivation for studying this equation comes from transport in porous media, where equations on this form are used as simple transport models or arise as the result of a semi-discretisation of a more complex transport equation. Accurate solution of (1) is of great importance in areas such as oil recovery and groundwater hydrology because (1) reveals the transport properties of $\mathbf{v}$. Solving (1) is rather easy for smooth $\mathbf{v}$, but becomes much harder when the vector field has large spatial variations and exhibits fine-scale details that are important for the global flow pattern.

In the following, we focus on *convective* transport in a porous medium completely filled with fluids of a single phase. To this end, we assume that the fluid velocity $\mathbf{v}$ is a time-independent function that is given as the result of a finite-volume or a (mixed) finite-element computation. In reservoir simulation, for instance, it is common to use a low-order method to compute the flux defined on a grid rather than the flow velocity $\mathbf{v}$, meaning that $\mathbf{v}$ will be given in terms of flux values that typically are constant on each element face in the grid.

To discretise (1), we will use a discontinuous Galerkin (dG) method for the operator $\mathbf{v} \cdot \nabla$ in combination with an upwind approximation of the flux. The discontinuous Galerkin method was introduced by Reed and Hill [17] for the problem of neutron transport. LeSaint and Raviart [14] analysed the method in this context and proved a rate of convergence of $\mathcal{O}(\Delta x^n)$ for smooth solutions on Cartesian grids. A number of researchers have made significant contributions since then. Among others, Lin and Zhou [13] proved convergence of the method for nonsmooth solutions. Moreover, Cockburn and Shu [2,3] analysed and extended the original discontinuous Galerkin method to systems of hyperbolic conservation laws and convection-dominated problems.

It is interesting to note that (1) can be interpreted as a stationary advection equation with a source term, and it is therefore close to the original application of Reed and Hill. However, the problem we address in this paper is the resolution of advective transport when $\mathbf{v}$ varies many orders of magnitude, whereas Reed and Hill consider a smooth velocity field.

The dG discretisation of (1) will lead to a large system of (non)linear equa-

tions. However, due to the directional derivative $\mathbf{v} \cdot \nabla$, the exact solution in each grid cell K only depends on the set of points on the upstream side of a bundle of streamlines passing through K and is independent of the solution elsewhere in the domain. Using an upwind flux in our dG discretisation preserves this one-sided domain of dependence, and it is therefore possible to compute the solution in one element at a time, from inflow to outflow boundaries, if we can find a sequence of elements so that element i appears after element j in the sequence if i depend on j. Using arguments from graph theory, we will show that such a sequence can be found in linear time by traversing the grid and visiting each grid cell once. This optimal sequence of elements can then be exploited to develop a very efficient (non)linear solver based on a reordering of the unknowns giving a upper block-triangular system. This way, the computational effort needed to solve (1) is reduced from solving a large sparse (non)linear system involving all degrees-of-freedom in the domain, to solving a sequence of smaller problems involving one or a few (non)linear equations. In the linear case, this gives a direct solver that is not only simple to implement, but also fast and inexpensive in terms of storage. In the nonlinear case, one has to apply an iterative solver for each subproblem. The resulting solver will generally have better convergence than a corresponding solver for the full system, since the iterations can be controlled independently in each subproblem. We note that these ideas are not new. The reduction of a matrix to block-triangular form by use of depth-first traversal of elements was described in Duff et al. [7]. Similarly, Dennis et al. [6] explore the use of block-triangular structures to construct effective Newton-type nonlinear solvers. As far as we know, however, these ideas have not been used to compute transport in porous media.

In this paper we will mainly focus on the case where $H(u)$ is a linear function of u. Extensions of the methodology to more general nonlinear equations of the form $\mathbf{v} \cdot \nabla F(u) = H(u, \mathbf{x})$ (arising e.g., from an implicit semi-discretization of multiphase-multicomponent transport models) are discussed in a separate paper [15]. In Section 2, we derive a few basic transport models on the form (1). Our motivating examples will be two linear boundary-value problems, one for the stationary distribution of tracers and one for the so-called time-of-flight. Isocontours of time-of-flight represent the time-lines in a reservoir and the corresponding differential equation exhibits all the difficulties seen in more complex transport models due to nonsmooth spatial variations in the forcing velocity field. The time-of-flight equation will therefore be our key example used to develop the methodology. Solutions of the stationary tracer equation have a much simpler structure and are only used herein as a means to delineate reservoirs with multiple wells into (nearly) independent flow regions. In Section 3, we introduce the discontinuous Galerkin method briefly and present the variational formulation and discretisation of (1). Then, in Section 4 we show how to solve the corresponding linear system efficiently using a reordering strategy. In Section 5, we show how to compute the distri-

bution of tracers from multiple wells in a single-phase reservoir. In Section 6, we present some numerical examples for computation of time-of-flight from (5) and compare the accuracy of our dG methods to highly resolved solutions obtained by pointwise integration of streamlines. Finally, Section 7 contains some concluding remarks.

2 Basic Transport Models

The flow of fluids through porous and heterogeneous media can be modelled as a set of balance laws for the conservation of mass for each fluid component. For a mixture of m fluid components separated into ℓ phases, we have

$$\sum_{i=1}^{\ell}\Big(\partial_t(\phi c_{\alpha i}\rho_i s_i) + \nabla\cdot(c_{\alpha i}\mathbf{v}_i\rho_i)\Big) = \sum_{i=1}^{\ell} c_{\alpha i}q_i, \qquad \alpha = 1,\ldots,m, \tag{2}$$

where ϕ is the porosity of the medium; ρ_i, s_i, $\mathbf{v}_i$, and q_i are the density, saturation (volume fraction), phase velocity, and volumetric source term of the i'th phase; and $c_{\alpha i}$ is the mass fraction of component α in phase i. In this model gravity and capillary effects have been neglected.

If all fluids are of the same phase (i.e., $\ell = 1$) and the flow is incompressible, we can write down the equation for the bulk motion of the fluid components in terms of the common fluid pressure p and the volumetric velocity field $\mathbf{v}$:

$$\nabla\cdot\mathbf{v} = q/\rho, \qquad \mathbf{v} = -\frac{\mathbf{K}}{\mu}\nabla p. \tag{3}$$

Here $\mathbf{K}$ is the permeability of the medium and μ is the viscosity. The linear relation between average fluid velocity and pressure gradients is called Darcy's law. For simplicity, we scale (3) such that $\mu = 1$ and assume that q consists of a set of point-sources modelling injection/production wells.

The indvidual distribution of the various components are now given in terms of *linear* transport equations, $\partial_t(\phi c_\alpha)+\nabla\cdot(c_\alpha\mathbf{v}) = c_\alpha q/\rho$. In other words, each fluid component is transported according to the volumetric velocity field $\mathbf{v}$. As our first example of such a transport model, we consider the stationary distribution of a set of passive tracers ($\alpha = 1,\ldots,m$) and assume incompressible flow. Equation (2) then simplifies to

$$\mathbf{v}\cdot\nabla c_\alpha = 0.$$

We remark that gravity can be included these simple transport equations by replacing Darcy's law in (3) by $\mathbf{v} = -\mathbf{K}(\nabla p - \rho\mathbf{g})/\mu$. In Section 5 we will use the stationary tracer tranport equation as a means for computing

connectivity between sites for injecting and producing fluids, thereby deriving reservoir compartmentalisation.

For flows with more than one phase, one can derive a pressure equation of the form (3), where $\mathbf{K}/\mu$ is replaced by $\mathbf{K}\lambda(s)$ and $\lambda(s)$ is a nonlinear function accounting for the reduced mobility due to the presence of more than one fluid phase. As for single-phase flow, the motion of fluids is, in the absence of gravity, aligned with the velocity field $\mathbf{v}$; thus, all instantaneous transport occurs along integral curves Ψ of $\mathbf{v}$.

Integral curves, or streamlines, Ψ are everywhere tangent to the velocity field $\mathbf{v}$. If we introduce the bistream functions ξ, η, given such that $\mathbf{v} = \nabla\xi \times \nabla\eta$, the integral curves of $\mathbf{v}$ map to straight lines (ξ =const, η =const) in the so-called streamline coordinates (τ, ξ, η). Here τ takes the role of the spatial coordinate along streamlines and is called the time-of-flight coordinate. Moreover, we have the operator identity

$$\mathbf{v} \cdot \nabla = \phi \partial_\tau. \tag{4}$$

The appearance of ϕ in this relation is convenient since ϕ rescales streamline coordinate according to the pore volume the streamline passes through. For a homogeneous medium, however, τ equals the standard curve length along Ψ after an aproriate scaling (corresponding to setting $\phi = 1$).

The simplest possible model of the form (1) for convective transport induced by $\mathbf{v}$, is the following boundary-value problem for time-of-flight τ in Ω,

$$\mathbf{v} \cdot \nabla\tau = \phi, \qquad \tau|_{\mathbf{x}\in\partial\Omega^-} = 0. \tag{5}$$

The equation follows trivially by applying the operator identity (4) to the streamline coordinate τ. The time-of-flight $\tau(\mathbf{x})$ measures the time it takes a passive particle released at the closest point on the inflow boundary to travel to a given point $\mathbf{x}$. Isocontours of $\tau(\mathbf{x})$ are the time-lines in the porous medium and as such give vital information about the flow pattern, in particular for single-phase flow. The time-of-flight is also a cornerstone in modern streamline methods, see [5,11]. In a streamline setting, $\tau(\mathbf{x})$ is usually given by the integral

$$\tau(\mathbf{x}) = \int_\Psi \frac{\phi\, ds}{|\mathbf{v}|}, \tag{6}$$

evaluated along the streamline Ψ connecting $\mathbf{x}$ to the inflow boundary $\partial\Omega^-$.

Another interesting sub-case of (1) arises if we apply an implicit temporal discretisation to the transport equations (2), giving equations of the form

$$\frac{u^n - u^{n-1}}{\Delta t} + \nabla \cdot (\mathbf{v}u^n) = q. \tag{7}$$

By using the product rule on the term $\nabla \cdot (\mathbf{v}u^n)$ and a discontinuous Galerkin method for spatial discretisation of $\mathbf{v} \cdot \nabla u^n$, we will get essentially the same linear systems for each time step as for (5), but with a different right-hand side.

3 Discontinuous Galerkin Discretisation

The discretisation in a discontinuous Galerkin method starts with a variational formulation as in a standard Galerkin method, but allows for discontinuities over the element edges. To get the variational formulation of (1), we partition the domain into a collection of non-overlapping elements $\{K\}$. Let V be the space of arbitrarily smooth test functions. By multiplying (1) with a function $v \in V$ and integrating by parts over each element K, we get

$$-\int_K (u\mathbf{v}) \cdot \nabla v \, dx + \int_{\partial K} (u\mathbf{v}) \cdot \mathbf{n} \, v \, ds = \int_K H(u, \mathbf{x}) v \, dx \qquad \forall \, v \in V,$$

where $\mathbf{n}$ is the outer normal on the element boundary ∂K. We seek solutions in a finite-dimensional subspace $V_h \subset V$, so we replace the exact solution and the test function by $u_h \in V_h$ and $v_h \in V_h$, respectively. For V_h, we choose the space of piecewise smooth functions that may be discontinuous over element boundaries. Since u_h may be discontinuous over inter-element boundaries, we must replace the flux term $(u\mathbf{v} \cdot \mathbf{n})$ by a consistent and conservative numerical flux function $\hat{f}(a, b, \mathbf{v} \cdot \mathbf{n})$. This leads to the following discrete variational formulation: let

$$\begin{aligned} a_K(u_h, v_h) &= -\int_K (u_h \mathbf{v}) \cdot \nabla v_h \, dx + \int_{\partial K} \hat{f}(u_h, u_h^{ext}, \mathbf{v} \cdot \mathbf{n}) \, v_h \, ds, \\ b_K(u_h, v_h) &= \int_K H(u_h, \mathbf{x}) \, v_h \, dx, \end{aligned}$$

and find u_h such that

$$a_K(u_h, v_h) = b_K(u_h, v_h) \qquad \forall \, K, \quad \forall \, v_h \in V_h. \tag{8}$$

Here $\hat{f}$ is the upwind flux given by

$$\hat{f}(p, p^{ext}, \mathbf{v} \cdot \mathbf{n}) = p \max(\mathbf{v} \cdot \mathbf{n}, 0) + p^{ext} \min(\mathbf{v} \cdot \mathbf{n}, 0), \tag{9}$$

for inner and outer approximations p and p^{ext} at the element boundaries. The upwind flux preserves the directional dependency in the solution, which is crucial in our solution procedure.

To fix ideas, we assume, for simplicity of presentation, that $\Omega \subset \mathbb{R}^2$ and assume that the elements K are rectangles in a regular Cartesian grid. Let $\mathbb{Q}^n = \text{span}\{x^p y^q \, : \, 0 \leq p, q \leq n\}$ be the space of polynomials of degree at most

n in x and at most n in y, and let $V_h^{(n)} = \{v : v|_K \in \mathbb{Q}^n\}$. A convenient basis for this space is the tensor product of Legendre polynomials $L_k = \ell_r(x)\ell_s(y)$ for $r, s = 0, \ldots, n$. The approximate solution on an element K_i can then be written as

$$u_h(x, y) = \sum_{k=0}^{n^2} t_k^i \, L_k\Big(\frac{2(x - x^i)}{\Delta x^i}, \frac{2(y - y^i)}{\Delta y^i}\Big), \tag{10}$$

where (x^i, y^i) is the centre of element K^i. Thus, $V_h^{(0)}$ is the space of elementwise constant functions and yields a formally first-order accurate scheme, $V_h^{(1)}$ is the space of elementwise bilinear approximations and yields a formally second-order accurate scheme, etc. In the following, we use dG(n) to denote the discontinuous Galerkin approximation of polynomial order n. In other words, the error of a dG(n)-method will decay with order $n+1$ for smooth solutions. On nonsmooth solutions, slower convergence is to be expected. Finally, the degrees of freedom per element in a dG(n)-method is $m = (n+1)^d$ in d spatial dimensions.

4 Fast Solution by Reordering the Unknowns

In the following we will motivate and present the optimal reordering that allows us to solve (8) elementwise. To this end, we will use the time-of-flight equation (5), for which (8) simplifies to a *linear* system

$$a_K(u_h, v_h) = b_K(v_h) \qquad \forall\, K, \quad \forall\, v_h \in V_h. \tag{11}$$

Although (8) and (11) have different structure on a given element K, the two *global* systems will have a similar block structure. All ideas presented for the linear case (11) will therefore immediately carry over to the nonlinear case (8).

By substituting the approximate solution (10) and the tensor-product Legendre polynomials in the variational formulation (11), we get a set of linear equations for the degrees-of-freedom in each element. Let T denote the vector of unknown coefficients t_k^i in all of Ω, and let T_K be the vector of unknowns for element K. In element K, we have

$$A_K T = B_K, \qquad A_K T = -R_K\, T_K + F_K T$$

where $(A_K)_{ij} = a_K^h(L_i, L_j)$ and $(B_K)_i = b_K^h(L_i)$, and a_K^h and b_K^h are numerical approximations to the integrals in (11) using Gaussian quadrature. For convenience, we have split the coefficient matrix into the element stiffness matrix R and the coupling to other elements through the numerical flux integral F.

The coefficient matrix has a block-banded structure, where the size of each block is given by the number of unknowns in each element.

The properties of F are in general determined by the choice of numerical flux. The upwind flux (9) can be written as

$$F_K T = F_K^+ T_K + F_K^- T_{\mathcal{U}(K)}, \tag{12}$$

where F^+ denotes flux *out of* element K, F^- denotes flux *into* element K, and we write $\mathcal{U}(K)$ for the set of neighbouring elements on the upwind side of K, i.e., $\mathcal{U}(K) = \{E \in \Omega : \partial E \cap \partial K^- \neq \emptyset\}$. Thus, the system reads

$$-R_K\, T_K + F_K^+\, T_K = B_K - F_K^-\, T_{\mathcal{U}(K)}. \tag{13}$$

The split $F = (F^+ + F^-)$ is easy to motivate and understand if one assumes that $\mathbf{v} \cdot \mathbf{n}$ does not change sign on element interfaces, which we will do henceforth. If $\mathbf{v}$ is computed using a standard low-order discretisation method for (3) like the two-point flux-approximation method (i.e., the five-point method in 2D) or the lowest-order Raviart–Thomas mixed finite-element method, $\mathbf{v} \cdot \mathbf{n}$ will typically be constant on each element face. In this case, $\mathcal{U}(K)$ consists of all elements E such that $(\mathbf{v} \cdot \mathbf{n})|_{\partial E \cap \partial K} < 0$, where $\mathbf{n}$ is the outward-pointing normal to K.

The key to an efficient solution procedure is to take advantage of the fact that (1) has this one-sided domain of dependence; in other words, both the exact and the numerical solution in any element is determined by the solution on the upstream side(s). Thus, we can construct the solution in a given element once the solution is known in the element's immediate upstream neighbours. By careful inspection, we may therefore construct the solution *locally*, starting at sources or inflow boundaries and proceeding downstream. A similar approach was used in [17] in the context of neutron transport. To our knowledge, the idea has never been applied to transport in porous media before.

From a computational point of view, it is more convenient to look at this as a reordering of unknowns. Observe that we can solve (13) element by element if we can determine a sequence of elements such that i appears before j in the sequence if there is a flux from element K_i to element K_j. By processing the elements in such a sequence, the right-hand side of (13) is a known quantity in each step. Since the directions of the fluxes are determined solely by $\mathbf{v}$ (and not by T), this sequence can be computed as part of a preprocessing step before solving the system (13).

The idea of solving boundary-value problems for advective transport sequentially by a reordering of unknowns was also used in [1], but with a different spatial discretisation. In that paper, we used an algorithm to compute a suitable sequence based on physical arguments. The solution was constructed by

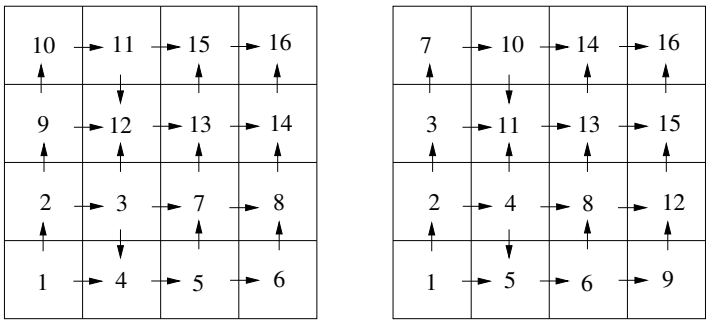

Fig. 1. (Left) Direction of flow and the order of computation generated by a depth–first traversal of the reversed flow field. (Right) The sequence that could have been computed from an advancing-front algorithm.

marching outwards from the inflow boundaries or sources. To do so, we needed to keep a list of candidate nodes for the next update(s). In each step of the algorithm, a suitable candidate node was sought in the list, the solution at this node was computed based entirely on known nodal values, and each of the node's downstream neighbours were added to the list.

In this paper, we choose a different approach. To find the sequence of elements, we observe that the elements and fluxes together form a directed graph, where the elements are the vertices and the fluxes are the directed edges; that is, if there is a flux from element i to element j, then there is a directed edge from vertex i to vertex j in the graph. Furthermore, if the desired sequence of elements exists, this graph is acyclic (DAG). In graph theory, the task of finding this sequence of vertices is known as a topological sort of the vertices, which can be accomplished by a depth-first traversal of the reversed DAG (see, e.g., [18]). The depth-first traversal takes $\mathcal{O}(N)$ operations for a graph of N vertices. In most cases, the depth-first traversal will produce a sequence of nodes that allows an elementwise computation of the solution. If the sequence of elements does not exist, it means that there are so-called strongly connected components in the graph, that is, groups of elements that are interdependent. For these groups of elements, we need to compute the solution simultaneously. This is discussed in the next subsection. However, strongly connected components of directed graphs can be found by one additional depth-first traversal. This means that finding a reordering and locating possibly connected components is altogether an $\mathcal{O}(N)$ operation. For the remaining part of the paper, we will assume that a topological sort can always be performed (or, as in our implementation, that the solution algorithm is capable of solving several cells simultaneously whenever connected cells are encountered). In Figure 1, we have illustrated the difference between the DAG algorithm and the advancing-front algorithm [1] for finding an adequate ordering of elements in a simple 2D case.

Standard linear solvers are usually assumed to have a computational complexity of n^α operations for n unknowns. Modern techniques like multigrid

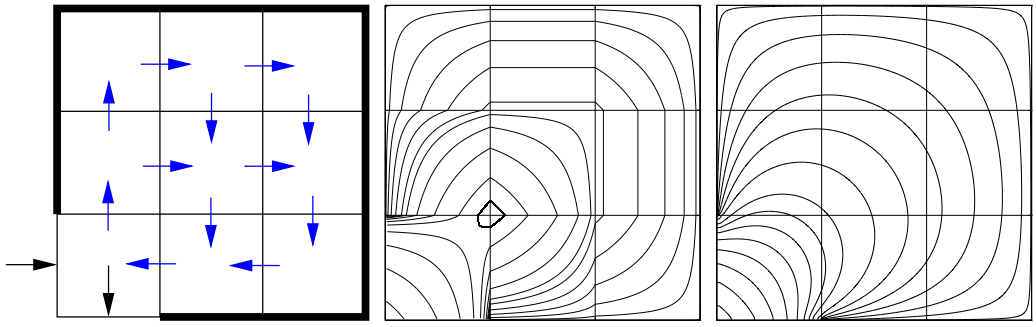

Fig. 2. (Right) A homogeneous domain with inflow and outflow in a single element and no-flow boundary elsewhere. The arrows indicate the sign of the flux on the faces of a 3×3 grid. (Middle) Streamlines of a mixed finite-element solution with 3×3 elements. (Right) Streamlines for a solution computed using 90×90 elements.

or domain decomposition can obtain (close to) linear complexity ($\alpha \sim 1.0$) for advection problems, but constructing such solvers efficiently is certainly non-trivial. Reordering the elements reduces the computational effort needed to solve (13) from $(Nm)^\alpha$ to Nm^α, where N is the number of elements and m is the number of degrees-of-freedom per element. In other words, instead of solving a large $(Nm) \times (Nm)$ system, we solve N small linear $m \times m$ systems, for which highly efficient solvers easily can be constructed. Thus, reordering is a simple way to obtain highly efficient linear solvers for large advection problems. See [15] for a discussion of how the same principle can be applied to decouple the solution of *nonlinear* systems arising in the implicit discretization of multiphase transport equations. Discontinuous Galerkin methods based on explicit temporal discretizations are discussed by Hoteit and Firoozabdi [9].

Strongly Connected Groups of Elements

As noted above, the graph defined by the grid and the fluxes may in some cases not be acyclic. For instance, if a higher-order method is used to discretise (3), there may be a few element faces over which the flux changes sign. Consider such a face, with neighbouring elements K_1 and K_2. If $\mathbf{v} \cdot \mathbf{n}$ takes both signs on $\partial K_1 \cap \partial K_2$, then $K_1 \in \mathcal{U}(K_2)$ *and* $K_2 \in \mathcal{U}(K_1)$ (see (12)), meaning that solutions in the two neighbouring elements depend on each other and must be computed simultaneously. A direct mapping of the corresponding fluxes to a graph results in a graph with two-way edges. To obtain a directed graph, each two-way edges must be replaced by two one-way edges. The corresponding cycle can be automatically detected as before.

Also when $\mathbf{v} \cdot \mathbf{n}$ is constant on each grid face, certain boundary conditions for the pressure equation (3) will produce cycles in the dependency graph. This is a reflection of the fact that although streamlines do not cross, they may pass through a grid cell more than once. An example of this is shown in Figure 2. The figure shows a velocity field computed using a mixed finite-

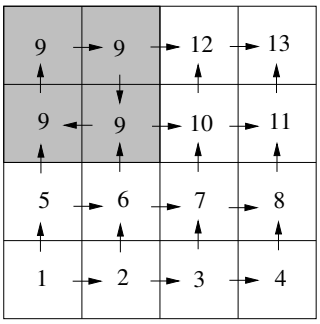

Fig. 3. The figure shows a suitable sequence of computations when a group of connected elements is encountered.

element method with lowest order Raviart–Thomas on a coarse and on a fine grid. The velocity field is the solution of (3) with the imposed boundary conditions shown. When the global inflow and outflow boundaries are edges of the same element, every streamline starts and ends in this element. Thus, the dependency graph of the elements will not be acyclic. In fact, for this case all the degrees-of-freedom in the domain must be computed simultaneously. Note also that a more accurate solution of (3) projected onto the 3×3 grid produces the same dependency graph.

The situation in Figure 2 is a worst-case scenario. A more likely distribution of fluxes is depicted in Figure 3, where a small subset of elements are strongly connected. In this situation, our reordering strategy still works and gives one larger linear system associated with the 2×2 block of interconnected cells in addition to the usual twelve linear systems associated with single cells.

Moreover, blocks of interconnected cells do not appear if one uses a two-point flux-approximation finite-volume scheme for the pressure equation (3), as has been the standard in the oil industry. A simple argument of decreasing pressure along streamlines rules out the possibility of a streamline re-entering a cell.

5 Approximation of Tracer Distribution

Determining the spatial region swept by a fluid source or an inflow boundary, or vice versa, the spatial region from which fluid is drained by a sink or an outflow boundary, is of practical importance both in groundwater management and petroleum engineering. In reservoir engineering, computing streamlines is well suited for predicting where fluid from different wells will eventually end up. For incompressible flow, any streamline in the domain connects two wells, an injector and a producer. Thus, by assigning a unique label to each well, streamlines will give information about well connectivity and areas affected by each injector or producer. The way one could obtain the same information in a finite-volume method is by computing the transport of tracers from each

injector. When the tracer transport becomes stationary, one would obtain information about well connectivity and affected areas.

Our ideas lend themselves naturally to compute the transport of tracer effectively. The stationary distribution of tracers is given by an equation of the form

$$\mathbf{v} \cdot \nabla c = 0, \qquad c|_{\mathbf{x} \in \partial\Omega^-} \text{ given.} \tag{14}$$

Since c is constant along streamlines, the solution to this equation can be determined in each point by tracing a streamline backward to the inflow boundary.

To determine the reservoir volume connected to a particular injector, we would solve (14) by setting the concentration of tracer component i to 1 in well i and 0 in the $m-1$ other wells and compute the tracer distribution in the non-well blocks. For an upwind discontinuous Galerkin discretisation of equation (14), the linear equations for element K are

$$(-R_K + F_K^+ + F_K^-)C_i = 0, \quad i = 1, \ldots, m, \tag{15}$$

where C_i is the vector of unknowns for tracer i. As before, we may split the vector of unknowns C_i in the unknowns $C_{i,\,K}$ for element K and the unknowns $C_{i,\,\mathcal{U}(K)}$ in the neighbouring upwind elements. Then, (15) may be written as

$$-R_K C_{i,K} + F_K^+ C_{i,K} = -F_K^- C_{i,\,\mathcal{U}(K)}, \quad i = 1, \ldots, m.$$

By solving this equation for C in one element at a time, we compute the distribution of tracers in the domain. Note that the tracer components are independent so only one matrix factorisation is needed for the solution of an m-tracer problem. The same idea can easily be extended to compressible flows, for which (14) is replaced by $\mathbf{v} \cdot \nabla c = -c\nabla \cdot \mathbf{v}$.

From the tracer distribution, we can approximate the swept areas of each injection well. The simplest approach is to draw the 0.5 contour (isosurface) of each tracer concentration. To obtain the drained areas for each production well, we simply reverse the velocity field. To get the well connectivity, we can combine the two calculations to uniquely determine the part of the domain affected by a given injector-producer pair.

At this point it might be tempting to ask why one could not replace (14) by a simple graph colouring algorithm to assign a colour to all nodes influenced by a particular injector (or more generally, a particular part of the inflow boundary). Such an approach is indeed possible, but would in general lead to multi-labelled nodes. Due to the fluxes, our directed graph is a *weighted* graph. By solving the tracer equation (15), we are effectively computing a *weighted colouring* of the graph.

Swept Areas/Volumes

We will now present three test cases, in which we use the above idea to delineate reservoirs in 2D and 3D, respectively. To this end, we compute the stationary distribution of one tracer launched from each injector. In the figures, we show the swept areas/volumes, which are distinguished by different shading. In 2D, the boundaries of the regions are marked by black and correspond to the 0.5 contour of each tracer concentration.

We first show how this idea works in two space dimensions. To asses the performance of the method, we will use geological data from Model 2 of the 10th SPE Comparative Solution Project [4]. The model contains $60 \times 220 \times 85$ cells and consists of two formations: a shallow-marine Tarbert formation in the top 35 layers, where the permeability is relatively smooth, and a fluivial Upper-Ness permeability in the bottom 50 layers. Both formations are characterised by large permeability variations, 8–12 orders of magnitude, but are qualitatively different; see Figure 9 for plots of the corresponding permeabilities. We compute the swept areas of eight injectors placed on the boundary of two rectangular reservoirs corresponding to Layers 1 and 76 Three production wells are placed inside the domain so that the wells form three five-spot patterns. The production wells are sources with rate -2.0, the injection wells in the corners have rate +0.5, and the other injection wells have rate +1.0. In the figures, the production wells are marked by white and injection wells by black circles.

Figure 4 shows the swept areas for Layers 1 and 76, respectively, computed using the dG(0) and dG(1) methods. To illustrate the flow directions, a few streamlines are plotted in the domain. In the figures, these are drawn in white. The streamlines close to the boundaries between the swept areas make it possible to evaluate the quality of the approximations for different orders of the dG discretisation.

The permeability in Layer 1 is relatively smooth, and the differences between the dG(0) and dG(1) discretisations are minor. The permeability in Layer 76, has a strongly heterogeneous structure with intertwined high-permeable channels on a low-permeable background. Using the dG(0) method, we can observe some streamlines crossing the boundaries between the swept areas, whereas the dG(1) method seems to have captured the areas correctly. Increasing the order in the dG discretization further did not produce any observable differences in the swept areas.

The same idea can also be applied to three-dimensional problems. Figure 5 shows swept volumes computed for the fifteen upper layers of the SPE 10 test case. The injection wells are located in the upper-left and upper-right corners of the back plane and in the lower-left and lower-right corner in the front plane.

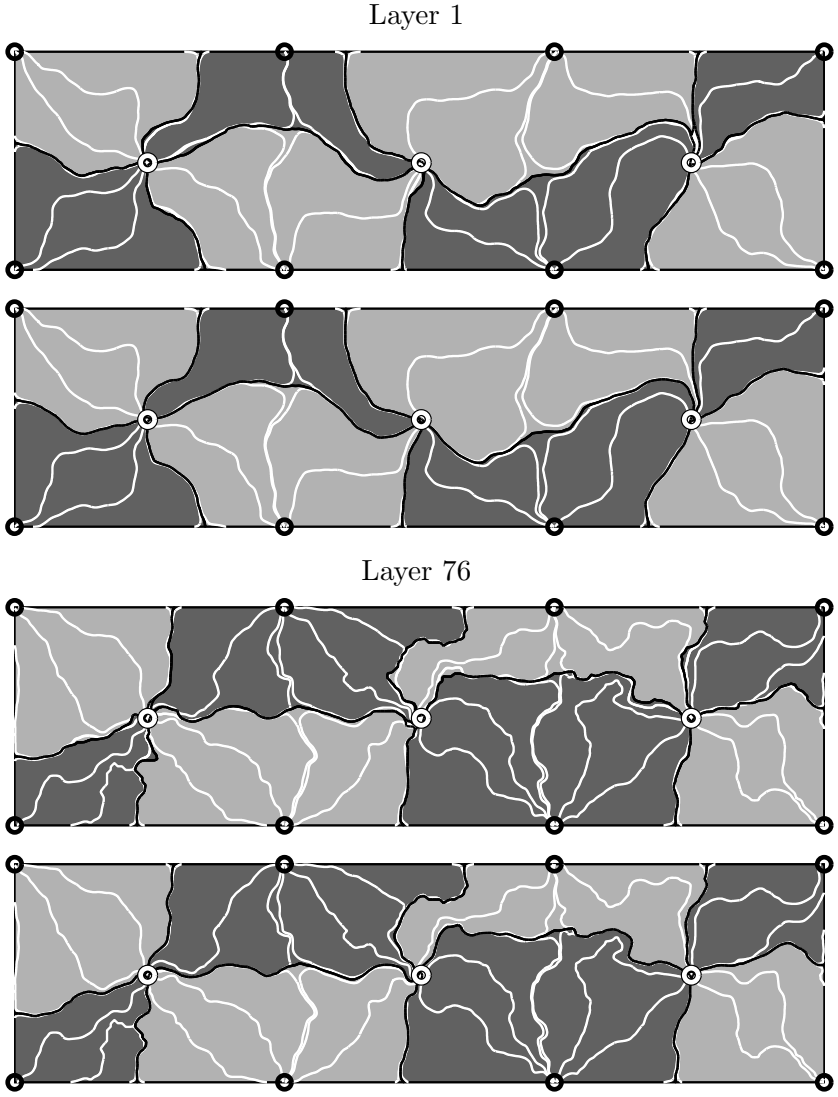

Fig. 4. The plots show the tracer distribution for two layers of the SPE 10 test case. The solution is computed using the dG(0) (upper) and dG(1) (lower).

The production well is placed in the centre of the domain. To distinguish the swept regions for each tracer, we have applied different shadings.

Table 1 reports runtimes for a similar partitioning of the full SPE 10 model with 1 122 000 cells. The runtimes have been split into time used to reorder and time used for solving the local dense $m \times m$ systems with LAPACK. For completeness we have also included corresponding timings for dG with tri-linear and tri-quadratic basis functions (P-basis). By using the first-order dG(0) discretization, the whole 1.1 million reservoir model is delineated in only a few seconds, which means that the method has a big potential for use in interactive user-exploration of large geomodels. If more accuracy is required for the swept volumes, the corresponding runtime will of course increase significantly. However, by using the P-basis, a second-order approximation is computed in

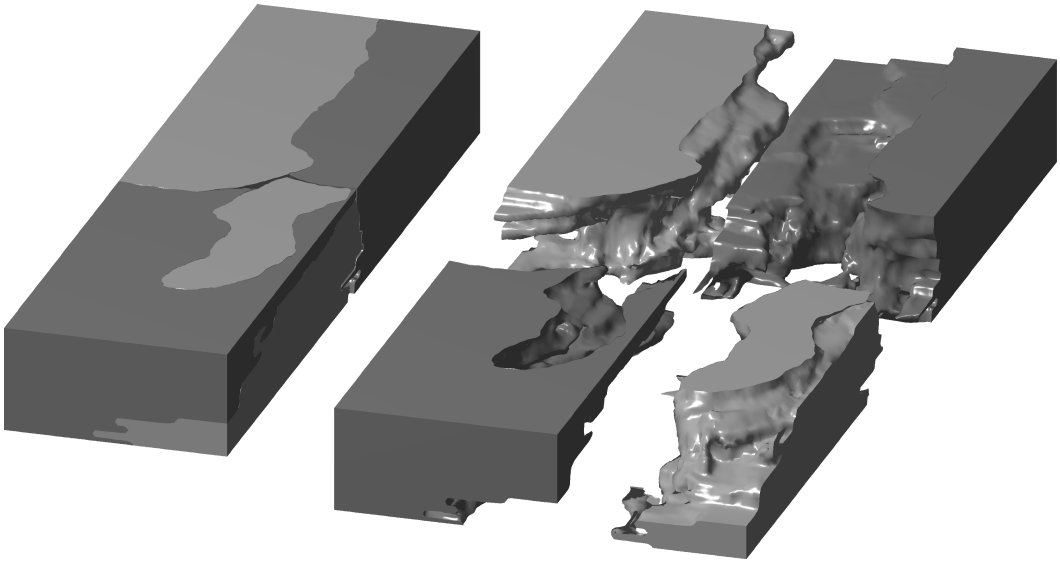

Fig. 5. Tracer distribution for a subsample from the smooth Tarbert formation in the SPE 10 test case. The velocity is computed using a two-point flux approximation.

Table 1
CPU time in seconds used to reorder and to solve the tracer boundary-value problem for the SPE 10 five-spot reservoir using dG(n) with m degrees of freedoms per cell. Runtimes are measured on a single core on an AMD Athlon X2 4400+ processor.

n	basis	m	reorder	solve	total
0	—	1	1.24	1.87	3.11
1	P-basis	4	1.21	8.65	9.86
1	Q-basis	8	1.20	25.22	26.41
2	P-basis	10	1.21	85.28	86.48
2	Q-basis	27	1.20	582.34	583.53

less than ten seconds and a third-order approximation in less than 1.5 minutes. (Notice also that for dG(2), the total number of unknowns is more than 30 millions).

6 Approximation of Time-of-Flight

In this section we will discuss the approximation of the time-of-flight equation (5) through a series of test cases with increasing difficulty. Although the equation has a simple form, the solutions are useful in many applications of transport in porous media. In ground-water flow, for instance, the time-of-flight can be used to identify the areas affected by a contamination. Moreover, as the examples in this section will clearly demonstrate, time-of-flight holds much of

Table 2
The L_2-errors and convergence rates for a grid refinement study of the discontinuous Galerkin scheme with increasing approximation order on a series of $N \times N$ grids. In the upper half, the L_2 error is measured over the smooth domain $[1, 1.3] \times [1, 1.3]$ and in the lower half over the square $[1, 2] \times [1, 2]$.

N	dG(0)		dG(1)		dG(2)		dG(3)	
10	3.36e-03	—	3.13e-05	—	1.74e-07	—	2.77e-09	—
20	1.52e-03	1.15	7.42e-06	2.08	2.24e-08	2.96	1.45e-10	4.25
40	8.01e-04	0.92	1.95e-06	1.93	2.90e-09	2.95	9.58e-12	3.92
80	4.14e-04	0.95	5.02e-07	1.96	3.69e-10	2.97	6.22e-13	3.94
160	2.05e-04	1.01	1.25e-07	2.01	4.60e-11	3.01	3.84e-14	4.02
320	1.02e-04	1.01	3.10e-08	2.01	5.73e-12	3.00	2.39e-15	4.01
10	2.83e-02	—	2.06e-03	—	6.16e-04	—	3.07e-04	—
20	1.72e-02	0.72	7.59e-04	1.44	2.07e-04	1.57	9.81e-05	1.64
40	1.01e-02	0.76	2.75e-04	1.47	6.80e-05	1.61	3.07e-05	1.68
80	5.79e-03	0.80	9.90e-05	1.47	2.23e-05	1.61	9.54e-06	1.68
160	3.23e-03	0.84	3.54e-05	1.48	7.25e-06	1.62	2.94e-06	1.70
320	1.76e-03	0.87	1.26e-05	1.49	2.34e-06	1.63	9.00e-07	1.71

the spatial complexity present in solutions of multicomponent and multiphase models from reservoir simulation. Therefore, the following test cases show not only the correctness of our solution strategy, but also the spatial resolution, or the lack thereof, one can expect to get for more complex transport models.

We start by verifying the accuracy and convergence rates of our discontinuous Galerkin schemes. For this purpose we use a simple rotating velocity field, for which the exact time-of-flight can be computed analytically.

Case 1 (Convergence Study) *Consider* (5) *with* $(u, v) = (y, -x)$ *for* $(x, y) \in [1, 2] \times [1, 2]$. *This makes* $x = 1$ *and* $y = 2$ *inflow boundaries and the remaining boundaries outflow boundaries. By setting* $T = 0$ *on the inflow boundaries, the exact time-of-flight can be computed as*

$$T(x, y) = \arctan\left(\frac{y}{x}\right) - \arctan\left(\frac{\min(\sqrt{r(x,y)^2 - 1}, 2)}{\max(\sqrt{\max(r(x,y)^2 - 4, 0)}, 1)}\right),$$

where $r(x, y) = \sqrt{x^2 + y^2}$. *In Table 2, we have computed the* L_2*-errors of the discontinuous Galerkin scheme for different orders and grid resolutions. The upper half of the table shows* L_2*-errors in a smooth part* $(1, 1.3) \times (1, 1.3)$ *of the domain, while in the lower half the error is integrated over the whole domain.*

The dG methods yield the expected order of accuracy in smooth regions, but due to the kink in the solution along the circular arc $r = \sqrt{5}$, we get reduced convergence rates for the whole domain.

For applications in porous media the velocity field $\mathbf{v}$ is typically obtained by solving a pressure equation of the form (3). In the remaining examples of this section we will compare grid-based solutions obtained by discontinuous Galerkin methods of varying order to highly resolved streamline solutions obtained by back-tracing streamlines from a set of 10×10 uniformly distributed points within each element. Unless stated otherwise, the same subresolution is used in all the following plots to evaluate the piecewise polynomial dG-solutions within each element.

In all examples, we assume that $\mathbf{v}$ is known and given in a such way that the flux $\mathbf{v} \cdot \mathbf{n}$ is constant over each element face in a Cartesian grid. We can then use a streamline tracing method due to Pollock [16] to compute highly resolved reference solutions. Pollock's method uses an exact formula for the streamline through each element based upon a piecewise linear approximation of $\mathbf{v}$ in each direction. The method is widely used in the petroleum industry to trace streamlines, even though it may become highly inaccurate for non-Cartesian grids, see [10].

The first example is a standard test case in oil reservoir simulation, called a quarter-five spot:

Case 2 (Heterogeneous Quarter Five-Spot) *Consider a reservoir consisting of the unit square with no-flow boundaries and with a source placed in the lower-left corner and a sink in the upper-right corner. The synthetic permeability field is lognormal and isotropic and spans six orders of magnitude from the smallest to the largest value and the porosity is assumed to be constant equal unity. The corresponding (single-phase) velocity field is computed using a mixed finite-element method with first-order Raviart–Thomas basis on* 129×129 *elements.*

Figure 6 compares solutions computed by the dG(n) scheme for $n = 0, 1, 2$ with the solution obtained by the node-based fast-sweeping scheme of [1] (for which no subsampling was used in the plotting). In addition, we have included a reference solution obtained by tracing streamlines with Pollock's method [16] from 10×10 *uniformly distributed points inside each element. Pollock's method reproduces the exact time-of-flight since the velocity field computed by the lowest-order Raviart–Thomas approximation is consistent with the velocity approximation used in the tracing algorithm. The fast-sweeping method gives a resolution that is slightly better than dG(0) and slightly worse than dG(1). The differences are easily explained if we momentarily interpret dG(0) as a first-order* finite-difference *scheme. Whereas dG(0) uses a first-order upwind*

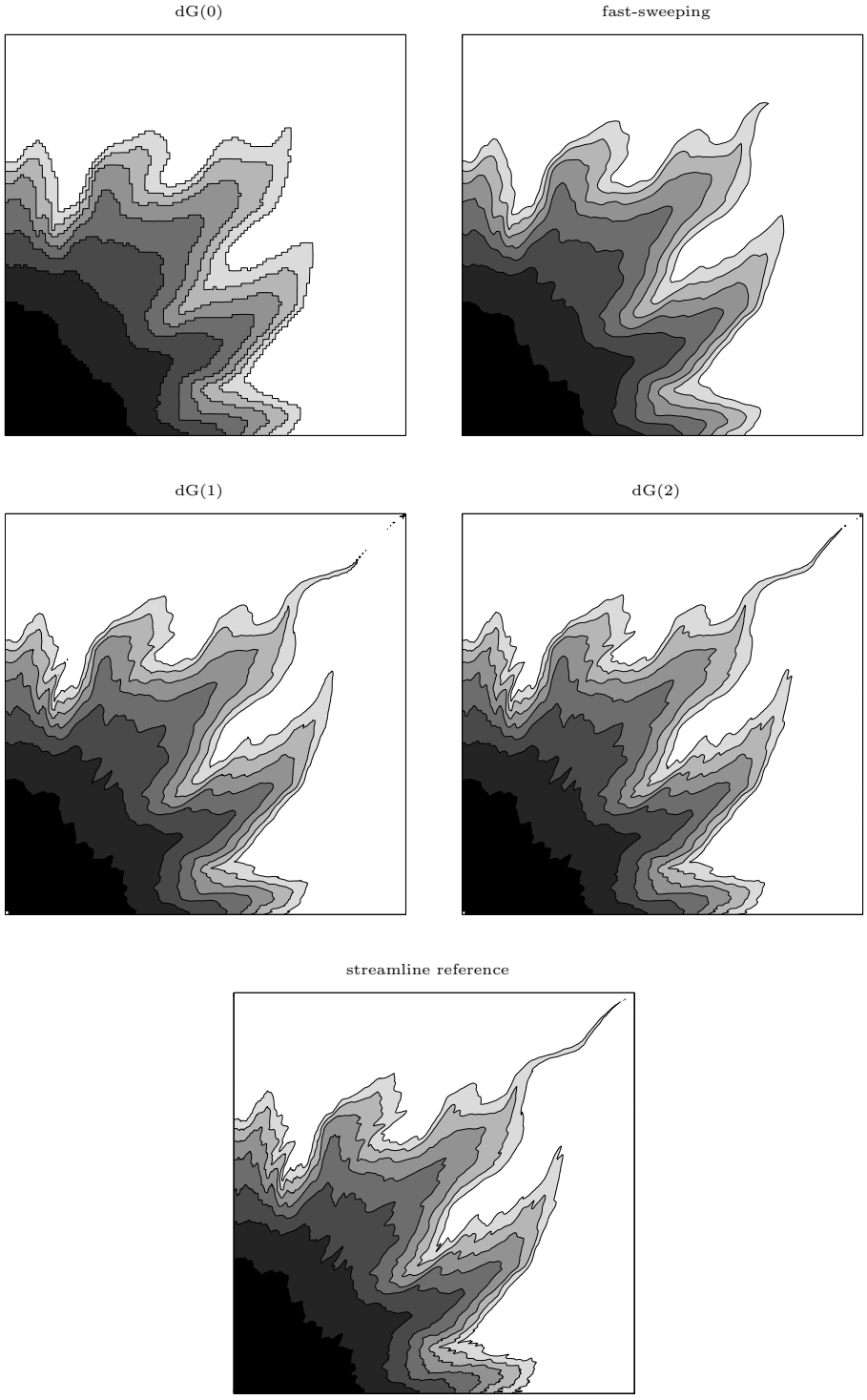

Fig. 6. Time-of-flights for Case 2 computed using dG(n) for $n = 0, 1, 2$, the fast-sweeping method, and direct streamline integration. The contours in the plots are $T = 0.07, \ldots, 0.49$ PVI in steps of 0.07.

discretisation in each coordinate direction corresponding to a five-point stencil, the fast-sweeping method uses a first-order upwind discretisation along local streamlines, which corresponds to a nine-point stencil and should therefore be more accurate. In the plots, it also appears to be smoother than dG(0), but this is a plotting artifact due to the linear interpolation inherent in the contouring algorithm. (For dG(0) we effectively use a piecewise constant interpolation due to the 10×10 *subsampling).*

Finally, we notice that the third-order method agrees remarkably well with the highly-resolved streamline reference solution.

In our second reservoir example we consider a three-dimensional case with similar heterogeneity as in Case 2.

Case 3 *We consider a reservoir model consisting of* $64 \times 64 \times 16$ *grid cells with unit porosity and a smoothed, lognormally distributed permeability field with values spanning five orders of magnitude, see Figure 7. An injector is located in the lower-left corner of the front face, a producer is located in the upper-right corner of the back face, and no-flow conditions are specified at the boundaries.*

Figure 7 shows time-of-flights computed by dG(n) for $n = 0, 1, 2$*. As in Case 2, dG(0) resolves the main features of the heterogeneous flow field, but underestimates the penetration of sharp fluid fingers. By increasing the polynomial order in the dG-basis functions, we allow for sub-cell variation in the time-of-flight and thereby improve the resolution of the viscous fingering, which in a sense is a sub-grid phenomenon.*

In the absence of gravity effects, (hyperbolic) models for multiphase and multicomponent transport will typically have only positive characteristics. This means that time-of-flight carries important information about the temporal development of complex spatial structures in the solution and $\tau(\mathbf{x})$ can thus be used to infer much about flow patterns for convection-dominated transport. In other words, by solving for $\tau(\mathbf{x})$ one can therefore learn much about the fluid motion without having to compute all time steps of a full fluid simulation.

From a computational point-of-view, computing the time-of-flight is in a certain sense more difficult than computing one time-step of a transport problem like e.g., (7). For transport problems, data are given in the whole spatial domain, and the domain of dependence for a single point (or grid cell) is therefore limited by Δt times the maximum wave speed associated by the corresponding continuous equation, and the variation in phase saturations or component concentrations are typically limited to the interval $[0, 1]$. Time-of-flight, on the other hand, has a global domain of dependence in the sense that $\tau(\mathbf{x})$ depends on all points along the streamline from $\mathbf{x}$ and back to the inflow boundary; see (6). Moreover, the time-of-flight values may easily span several orders of

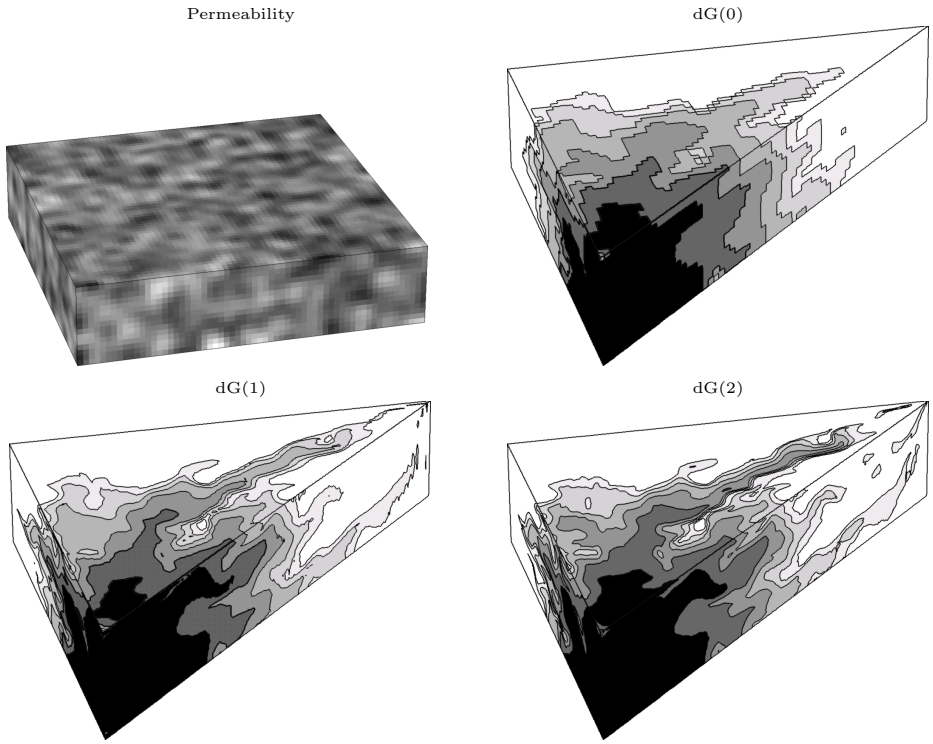

Fig. 7. Lognormal permeability field for Case 3 and corresponding time-of-flights computed using dG(n) for $n = 0, 1, 2$. The contours shown in the slice plots are at $T = 0.1, \ldots, 0.6$ PVI in steps of 0.1.

magnitude.

In the two examples above, the reservoir heterogeneity was mild due to unit porosity and relatively smooth spatial variation in $\mathbf{v}$. As a result, $\tau(\mathbf{x})$ had relatively smooth variation even though it contained the characteristic viscous fingers, and we were able to obtain good resolution by choosing a uniform sufficiently high order for the dG basis functions.

For highly heterogeneous reservoirs with large variations in the porosity or strong shears in the velocity field, τ will generally have low regularity and exhibit very large variations. For instance, in regions where high-speed flow meets low-speed flow from nearly impermeable regions such as channel walls or obstacles, the time-of-flight may oscillate with orders of magnitude over a few elements. Use of higher-order polynomial approximations can therefore easily result in oscillations and unphysical time-of-flight values, as will be demonstrated in Case 4. Moreover, if these variations are not captured by the local approximation, the error along the outflow edges will be propagated to the neighbour elements.

Slope Limiting

Spurious oscillations is a common problem in many discontinuous Galerkin methods and is usually circumvented by applying a (slope) limiter that reduces the local variation of each basis function by modifying the coefficients of polynomial terms of order two and higher. Limiters are usually derived from a maximum principle or from a principle that limits the local variation.

For the time-of-flight equation (5), the only principle available to us is the fact that $\tau(\mathbf{x})$ is strictly increasing along streamlines, which follows trivially from (6). We therefore propose to check that the time-of-flight is higher on the outflow edges than on the inflow edges of each element; that is,

$$\min \tau|_{\partial K^+} > (1-\varepsilon) \max \tau|_{\partial K^-}, \qquad 0 \leq \varepsilon \ll 1.$$

If this is not the case, we recompute the solution in this element by making a uniform subdivision into a set of first-order elements such that the number of new elements corresponds to the degrees-of-freedom in the original element. That is, for dG(1) we split the element in two in each spatial direction, for dG(2) we split in three, etc. By reducing the order to one, we expect to reduce possible oscillations, and by subdividing, we try to compensate for the reduced accuracy associated with first-order elements.

To clearly demonstrate the problems caused by shear in the velocity field and the effect of our order-reduction/subdivision strategy, we consider an artificial transport problem with four large impermeable geometrical obstacles.

Case 4 *We consider a quarter five-spot in a square domain with an injector in the lower-left corner and a producer in the upper right. The permeability field consists of a homogeneous background into which we have inserted four nearly impermeable obstacles—two triangles, a circle, and a rectangle—each having a permeability* 10^{-6} *relative to the background. The corresponding velocity field is computed using a mixed finite-element method with the lowest-order Raviart–Thomas basis.*

As observed in [1], transport past obstacles and through channels is very challenging since the time-of-flight field will have extreme gradients downstream from the obstacles. In Figure 8 we compare two approximate solutions computed using dG(0) and dG(2) with the exact streamline solution. As above, the first-order scheme fails to capture the leading viscous fingers, in particular those creeping around the impermeable circle. Similarly, the dG(2) solution contains strong oscillatory pollution that arise along impermeable boundaries and propagate in the downstream direction. By applying the order-reduction/subdivision strategy devised above with $\varepsilon = 0.005$*, almost all the oscillations are removed and the exact solution is reproduced quite accurately. By only using order reduction and no subdivision, the corresponding dG(2)-*

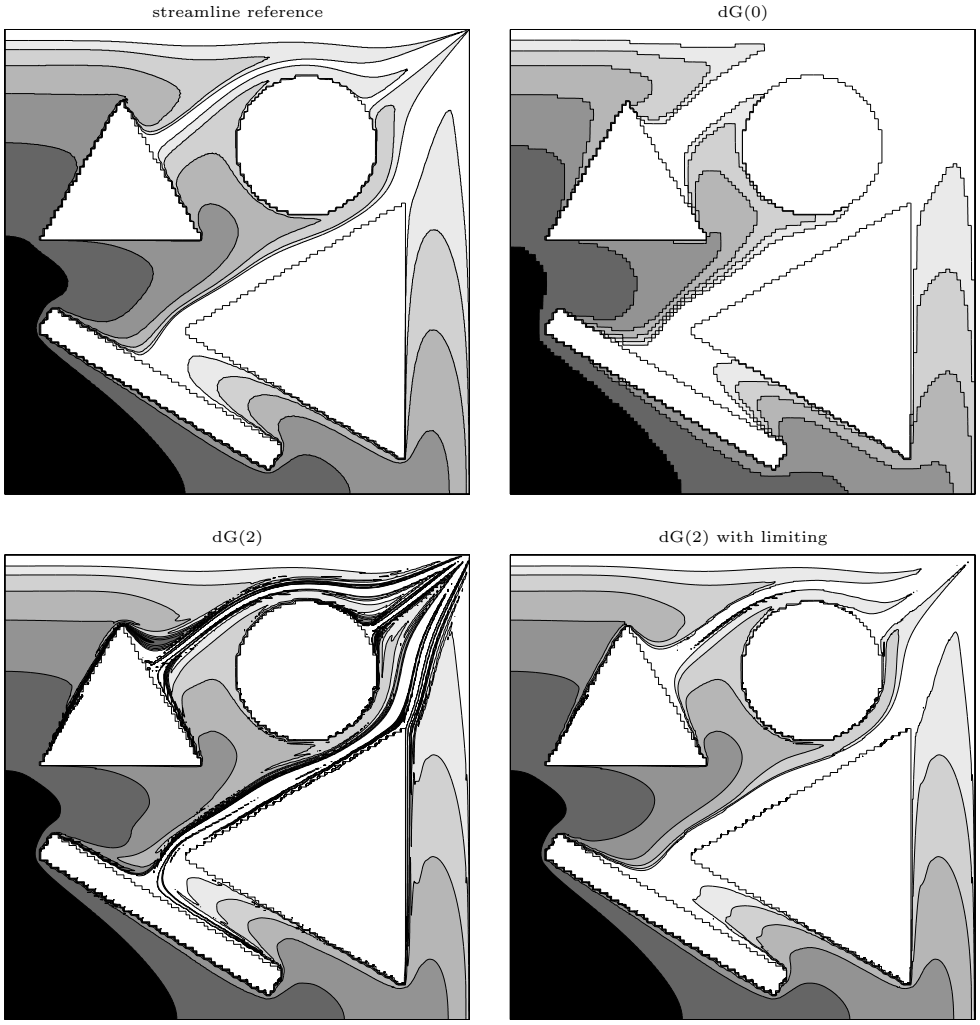

Fig. 8. Transport past obstacles for Case 4 computed by dG(0) and dG(2) with and without order-reduction/subdivision.

solution has comparable accuracy to that of dG(0).

In the above example we used a somewhat extreme case to demonstrate the problems caused by large shears in the velocity field. Similar problems will arise due to several types of strong reservoir heterogeneities: impermeable blocks, shales, layers with large permeability ratios, fluvial reservoirs with high-permeable channels on a low-permeable background, large variations in pore volumes. These difficulties will be partly demonstrated in our final example, in which we revisit Model 2 from the 10th SPE Comparative Solution Project [4].

So far, our reference solutions have been obtained by back-tracing a large number of streamlines inside each grid cell. For large models (in 3D), this approach is is generally not computationally feasible. Instead, modern streamline methods [11] rely on tracing a set of *representative* streamlines launched from

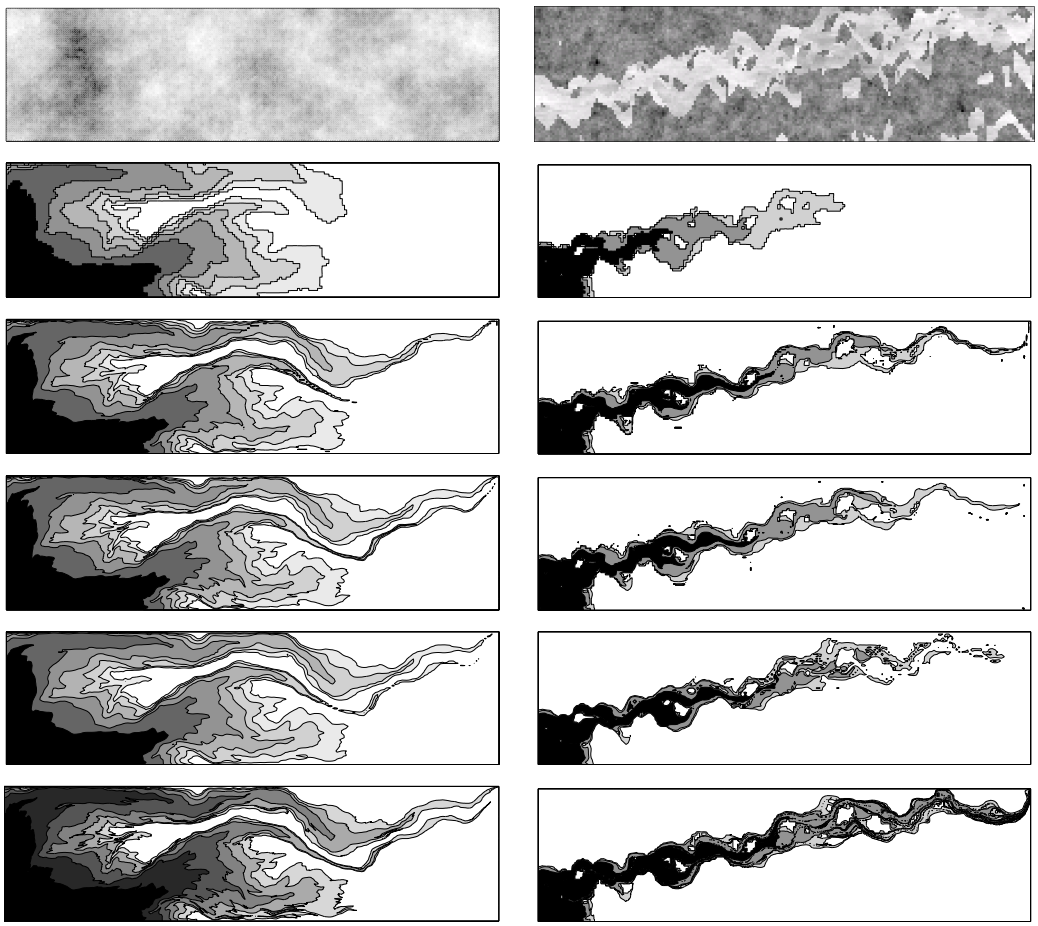

Fig. 9. Permeability field and time-of-flight for Layers 1 (left) and 76 (right) of the SPE 10 test case computed with dG(n) for $n = 0, 1, 2$ (order increasing downwards), streamline solution with 1500 streamlines, and a streamline reference solution (bottom). The contours shown in the plots are at $T = 0.1, \ldots, 0.6$ PVI in steps of 0.1 for Layer 1 and $T = 0.05, 0.1, 0.15$ PVI for Layer 76.

injectors and/or producers; see e.g., [12]. Cell-values for time-of-flight can then be computed by averaging all streamlines passing through or in the neighbourhood of each cell. This approach reduces the spatial accuracy unless one uses a sophisticated scheme for obtaining sufficient streamline coverage.

Case 5 *In this example we consider two 2D quarter five-spot cases with permeability and porosity data taken from Layers 1 and 76, respectively, of Model 2 in the SPE 10 test case. Figure 9 shows the permeability and the corresponding time-of-flights computed by dG(n) for* $n = 0, 1, 2$. *For comparison we also show solutions obtained by tracing 1500 streamlines initiated uniformly from the well block. For the Tarbert formation in Layer 1, the variation in permeability and porosity is relatively smooth. As in Case 2, dG(1) and dG(2) reproduce the qualitative behaviour of the solution, whereas dG(0) underestimates the viscous fingering. The accuracy of the standard streamline method is somewhere between that of dG(1) and dG(2).*

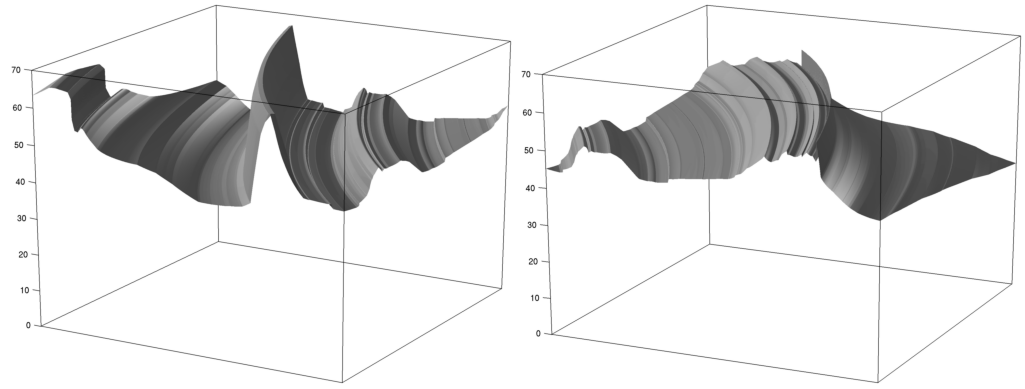

Fig. 10. Time-of-flight in the two grid cells $(200, 36)$ and $(200, 37)$ of Layer 76 in the 10th SPE test case sampled in 2000×2000 evenly distributed points inside each cell.

The fluvial Upper Ness formation in Layer 76 contains sharp contrasts in permeability (and porosity) between the low-permeable background and a set of intertwined high-permeable channels. For the higher-order dG methods we have therefore applied our order-reduction/subdivision strategy. Figure 9 shows that although dG(1) and dG(2) have quantitative errors, they are able to capture most of the qualitative behaviour of the time-of-flight, which should be of most interest to a reservoir engineer. Moreover, the higher-order dG solutions are at least as accurate as the solution obtained by the standard streamline method.

To illustrate the difficulty of accurately resolving the time-of-flight in Layer 76, Figure 10 shows the exact solution sampled in 2000×2000 *evenly spaced points inside two grid cells. Since time-of-flight is an integrated quantity, it is generally not sufficient to capture the complex spatial behaviour inside each grid cell in an averaged sense. The variations in time-of-flight over a grid cell may be quite large relative to an average value or a few representative point values. Any method based on either a low-order polynomial (as in dG(1) and dG(2)), or a few representative streamlines, is therefore bound to give quantitative errors, as observed in Figure 9.*

7 Final Remarks

The purpose of this paper has been to explore the efficiency and accuracy of a discontinuous Galerkin scheme applied to a class of boundary-value problems for advective transport. A unique feature of our methodology is the use of an optimal ordering of the unknowns that allows us to compute the solutions in an element-by-element fashion.

We have demonstrated how one can use the framework to compute accurate approximations to the stationary tracer distribution in a reservoir. This can

be used to compute so-called swept and drained areas/volumes and well connectivities. These quantities are usually computed using streamline methods and have proved to be useful tools in, e.g., ranking and history matching. Due to the efficient sequential solution procedure presented in this paper, this is a one-sweep computation that can be performed with high order accuracy and modest demands on storage and computing power. Moreover, as our numerical test cases illustrate, low-order approximations do, in general, provide sufficient accuracy.

We have also demonstrated that the discontinuous Galerkin schemes in many cases can compute the time-of-flight with an accuracy comparable to streamline approaches and superior to our earlier grid-based attempts [1]. For strongly heterogeneous cases, direct integration of streamlines gives a spatial resolution that is hard to match with other methods based on grid points or cell volumes. One should therefore not expect grid-based methods to perform as well as back-traced streamlines for all possible velocity fields, as was demonstrated in [1]. Indeed, we observe reduced accuracy for tranport past (and through) barriers and through channels as in Cases 4 and 5. For simple 2D cases one can always argue that better results can be obtained by grid refinement or by tracing more streamlines, but this is less feasible, e.g., for the full SPE 10 model containing $60 \times 220 \times 85 = 1\,122\,000$ grid cells, even if one is able to use the reordering algorithm to solve for the time-of-flight separately in each cell.

Our experience is that a dG discretisation of sufficiently high order is a relatively robust alternative (to streamlines) that performs well in a wide range of realistic cases. The computational efficiency of our methodology makes it a candidate for applications where one needs to establish the qualitative structures of the flow pattern. Prime examples of such applications are the calibration of reservoir models to production data and validation of upscaling of geological models. In [8] the dG-methodology was used to study simple 2D models of discrete fracture networks. Extensions of the dG/reordering methodology to multiphase and multicomponent transport will be discussed in a forthcoming paper [15]. Finally, although the method was presented for uniform Cartesian grids, the reordering idea is equally applicable to unstructured and irregular grids.

References

[1] I. Berre, K. H. Karlsen, K.–A. Lie, and J. R. Natvig. Fast computation of arrival times in heterogeneous media. *Comput. Geosci.*, 9(4):179–201, 2005.

[2] B. Cockburn and S.–W. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for scalar conservation laws: General framework.

Math. Comp., 52:411–435, 1989.

[3] B. Cockburn and S.–W. Shu. The Runge-Kutta local projection P^1-discontinuous Galerkin method for scalar conservation laws. *M^2AN.*, 25:337–361, 1991.

[4] M. A. Christie and M. J. Blunt. Tenth SPE comparative solution project: A comparison of upscaling techniques. *SPE Reservoir Eval. Eng.*, 4(4):308–317, 2001, `http://www.spe.org/csp`.

[5] A. Datta–Gupta and M. J. King. A semianalytic approach to tracer flow modeling in heterogeneous permeable media. *Adv. Water Resour.*, 18(1), 1995.

[6] J. E. Dennis JR., J. M. Martínez, and X. Zhang. Triangular decomposition methods for solving reducible nonlinear systems of equations, *SIAM J. Opt.*, 4(2):358–382, 1994.

[7] I. S. Duff and J. K. Reid. An implementation of Tarjans algorithm for block triangularization of a matrix, *ACM Trans. Math. Softw.*, 4(2):137–147, 1978.

[8] B. Eikemo, I. Berre, H. K. Dahle, K.-A. Lie, and J. R. Natvig. A discontinuous Galerkin method for computing time-of-flight in discrete-fracture models. In "Proceedings of the XVI International Conference on Computational Methods in Water Resources, Copenhagen, Denmark, June, 2006", Eds., P.J. Binning et al. `http://proceedings.cmwr-xvi.org/`

[9] H. Hoteit and A. Firoozabadi. Multicomponent fluid flow by discontinuous Galerkin and mixed methods in unfractured and fractured media. *Water Resour. Res.*, 41, W11412, doi:10.1029/2005WR004339.

[10] H. Hægland, H. K. Dahle, G. T. Eigestad, K.–A. Lie, and I. Aavatsmark. Improved streamlines and time-of-flight for streamline simulation on irregular grids. *Adv. Water Resour.*, 30(4):1027–1045, 2007. doi:10.1016/j.advwatres.2006.09.00.

[11] M.J. King and A. Datta-Gupta. Streamline simulation: A current perspective. *In Situ*, 22(1):91–140, 1998.

[12] V. Kippe, H. Hægland, and K.-A. Lie. A method to improve the mass-balance in streamline methods. SPE 106250. 2007 SPE Reservoir Simulation Symposium, Houston, Texas U.S.A., Feb. 26–28, 2007.

[13] Q. Lin and A.–H. Zhou. Convergence of the discontinous Galerkin method for a scalar hyperbolic equation. *Acta Math. Sci.*, 13:207–210, 1993.

[14] P. LeSaint and P. A. Raviart. On a finite element method for solving the neutron transport equation. In D. de Boor, editor, *Mathematical aspects of finite elements in partial differential equations*, pp. 89–145. Academic Press, 1974.

[15] J.R. Natvig and K.–A. Lie, Fast computation of multiphase flow in porous media by implicit discontinuous Galerkin schemes with optimal ordering of elements. Submitted, 2007.

[16] D. W. Pollock. Semi-analytical computation of path lines for finite difference models. *Ground Water* 26(6):743–750, 1988.

[17] W. H. Reed and T. R. Hill, Triangular mesh methods for the neutron transport equation, *Tech. Report*, LA-UR-73-479, Los Alamos Sci. Lab., 1973.

[18] R. Sedgewick. Algorithms in C. Addison-Wesley, 1990.