Fast Solvers for Computing Transport in Unfractured and Fractured Porous Media

PhD Thesis

Birgitte Eikemo

University of Bergen

August 2008

Birgitte Eikemo
Department of Mathematics
University of Bergen
Joh. Brunsgt. 12
NO-5008 Bergen
Norway
Birgitte.Eikemo@math.uib.no

University of Bergen, Norway
Printed by Allkopi, Bergen

Preface

The work in this thesis has been done as a part of a Ph.D. program in Applied Mathematics at the Department of Mathematics, University of Bergen (UiB), Norway. The Ph.D. is part of the GeoScale project at the Department of Applied Mathematics at SINTEF ICT in Oslo. This is a strategic research project for reservoir simulation on geological scale, and the collaboration includes Department of Mathematics, UiB.

Helge K. Dahle (UiB), Knut-Andreas Lie (SINTEF) and Geir Terje Eigestad (UiB) have been advisors for me, and the funding has been provided by The Research Council of Norway (Grant no.: 158908/130).

The period from September to mid December 2006 was spent at the Institute für Wasserbau at Universität Stuttgart in Germany. During my Ph.D. period I have also spent some time at the Department of the Applied Mathematics at SINTEF in Oslo.

Outline

This thesis is divided into two parts. The main focus of Part I is the theory behind the transport equations, and the discretisation principles used in the work together with a presentation of main methods and ideas. Many details have been left out in the papers presented in Part II, so the first part serves as a summary and gives background material to accompany published and submitted papers. Description of the developed and implemented methods, together with numerical results, are given in the research papers included in Part II. The main focus of these papers is on issues related to transport in unfractured and fractured porous media. In particular, we present a discontinuous Galerkin scheme, which combined with a numerical flux function, gives a fast, accurate and robust solution of advection dominated transport equations.

Part I

Chapter 1 gives an introduction to motivate the development of fast solvers for transport in unfractured and fractured porous media. Chapter 2 gives a brief overview of the physics and mathematics behind flow in porous media. In particular, the fundamental definitions needed for the mathematical model developed in Chapter 3 are explained. The numerical schemes for this mathematical model are derived in Chapter 4. In Chapter 5, the numerical scheme is applied to realistic examples for groundwater protection that are not presented in the included papers. Finally, in Chapter 6, we give a summary of the papers included in Part II.

Part II

The research papers included in Part II are:

Paper A: An Efficient Discontinous Galerkin Method for Advective Transport in Porous Media. J. R. Natvig, K.-A. Lie, B. Eikemo and I. Berre. *Advances in Water Resources*, Volume 30, Issue 12, pages 2424-2438, December 2007.

Paper B: Fast solvers for flow in porous media based on discontinuous Galerkin methods and optimal reordering. J. R. Natvig, K.-A. Lie and B. Eikemo. In *Proceedings of the XVI International Conference on Computational Methods in Water Resources*, Copenhagen, June 2006. Eds., P.J. Binning et al.

Paper C: A discontinuous Galerkin method for computing time-of-flight in discrete-fracture models. B. Eikemo, I. Berre, H. K. Dahle, K.-A. Lie and J. R. Natvig. In *Proceedings of the XVI International Conference on Computational Methods in Water Resources*, Copenhagen, June 2006. Eds., P.J. Binning et al.

Paper D: A discontinuous Galerkin method for transport in fractured media using unstructured triangular grids. B. Eikemo, K.-A. Lie, G. T. Eigestad and H. K. Dahle. Submitted.

As all the papers are results of cooperation, some remarks about my contributions are necessary.

In Papers A and B, we introduce an efficient discontinuous Galerkin (dG) method for advective transport in porous media. My contributions to the papers include parts of the method development and implementation of the dG schemes for time-of-flight and stationary tracer transport, mainly the last mentioned. I also took part in the convergence study in Paper A.

Paper C presents a discontinuous Galerkin method for computing time-of-flight in discrete-fracture models on Cartesian grids. My contributions to the paper include the implementation and investigation of different approaches for the dG discretisation in fractured regions.

In Paper D we continue the ideas from Paper C by extending the methodology to unstructured triangular grids such that more complicated grids may be handled. My contributions to the paper include the implementation of the dG scheme in combination with an optimal reordering of the elements on unstructured triangular grids. Additional to the convergence study of different perturbed grids, I investigated various approaches for the dG discretisation for discrete-fracture models.

Acknowledgements

When the project has come to an end there are a lot of people I wish to thank. I would like to express my deep gratitude to my advisors Helge K. Dahle, Knut-Andreas Lie and Geir Terje Eigestad for their guidance, good ideas and helpful advice. This work would never been completed without their help. I want to thank my main advisor Helge for his encouragement, support and guidance both before and during my time as a Ph.D. student. I also want to express my gratitude to my co-advisors Knut-Andreas and Geir Terje for their continuous support and enthusiasm. I appreciate our interesting discussions on research challenges and thank them for sharing their insights with me.

I would also like to thank Knut-Andreas and his group at the Department of Applied Mathematics at SINTEF ICT in Oslo, for letting me visit them and for making my stays there both pleasant and valuable by sharing knowledge with me.

A special thank to Rainer Helmig for hosting me at the Institut für Wasserbau at Universität Stuttgart for a period of three months during the Fall 2006, and for fruitful discussions and useful feedbacks. I want to thank Anozie Ebigbo and Yufei Cao, together with the rest of the group for taking good care of me and making my stay in Stuttgart memorable.

All the papers in my thesis are the result of collaboration with different people. In addition to my advisors, I would like to thank Jostein R. Natvig (Department of Applied Mathematics, SINTEF ICT) and Inga Berre (Department of Mathematics, UiB) for their contributions. A special thank to Jostein for many useful suggestions regarding our work and issues on implementation. Thank you, Inga, for interesting discussions and for useful feedbacks related to this work. I also want to thank my fellow student, Håkon Hægland, for his contribution related to streamlines.

A special thank to all my colleagues and friends at the Department of Mathe-

matics and the Centre for Integrated Petroleum Research for contributing to a very good work environment. Thank you, for interesting discussions, meals we shared and for other social events. All of you have been of great importance for making my Ph.D. period fun and interesting. No one mentioned, no one forgotten.

Finally, I want to express my gratitude to my family and friends for being there for me and giving me invaluable support and comfort in times of low spirit during my time as a Ph.D. student.

Birgitte Eikemo
Bergen, August 2008

Contents

Part I

Overview and Background Material

Chapter 1

Introduction

1.1 Reservoir Simulation

A *reservoir* is a trap in the subsurface where fluids, such as hydrocarbons (oil and gas) and water, have accumulated over millions of years. The reservoir rock is typically sedimentary in nature and consists of an interconnected porous network where fluids may flow, subject to forces such as fluid pressure, capillarity and gravity.

It should be clear that modelling reservoir flow is highly complex, involving different nonlinear physical effects and complex geometry. However, an understanding is necessary so that different production scenarios and reservoir interpretations may be tested. To some degree, this can be done in laboratory experiments. But these can only be performed on small scale samples of the reservoir (core plugs), and the experiments tend to be expensive to conduct. Instead, the use of *mathematical models* has become progressively more prominent in reservoir engineering. Using simple mathematical models with analytical solutions, engineers can provide basic performance predictions. However, for the more advanced models, analytical answers are not available. Instead, *numerical methods* for simulating the models have become popular, especially with the advent of fast computers.

Reservoir simulation is the study of how fluids flow and behave in a reservoir under different conditions. Unfortunately, obtaining an accurate prediction of reservoir flow scenarios is a difficult task. One of the reasons is that we can never get a complete and accurate characterisation of the rock parameters that influence the flow pattern down to the pore scale. And even if we did, we would not be able to run simulations that exploit all available information, since this would require a tremendous amount of computer resources that exceed by far the capabilities of modern multi-processor computers.

Moreover, reservoir modelling has become very advanced over the past decades. Advanced drilling techniques and enhanced seismic and geological characterisation of reservoirs have emerged and has resulted in more accurate geological information. Consequently, there is a substantial research activity that aims toward faster, more robust, and more accurate reservoir simulators. In this work we seek fast numerical schemes combined with suitable mathematical models for the reservoir.

1.2 Applications

It is of great importance to determine the flow characteristics of hydrocarbons in oil and gas reservoirs and groundwater in aquifers. Reservoir simulation has been an integrated part of oil-reservoir management for nearly half a century. A reservoir simulator can be used to decide the optimal operating policy, or it can be used to forecast future production. But such a tool can also be used within environmental studies, for instance by predicting impact of contaminant, performing groundwater management, or remediation of polluted soil.

1.2.1 Oil Recovery

A petroleum reservoir is a porous medium where hydrocarbons exist in the pore space. To recover the oil and gas, wells are drilled into the reservoir, some which produce oil and gas, and others which inject water and gas to provide pressure support. Since it is costly to drill and operate wells, it is desirable to optimise their number, placement and operation in the reservoir. For this to be done, a good understanding of the fluid dynamics in the reservoir is necessary.

In the oil industry, the goal is to determine how hydrocarbons and water behave in a reservoir under different conditions, and how local reservoir heterogenities affects the oil and gas recovery in reservoirs. Reservoir simulators are widely used to aid the planning and implementation of enhanced oil recovery strategies. The simulators can estimate production characteristics, calibrate reservoir parameters, visualise reservoir flow patterns, etc. The main purpose is to provide an information database supporting oil companies in positioning and managing wells to maximise the oil and gas recovery. See [2, 3] for details of petroleum reservoir simulation.

1.2.2 Groundwater Protection

Groundwater is an essential element of the hydrological cycle. A portion of the water that falls as precipitation infiltrates into the ground and becomes ground-

water. The water may stay in the groundwater reservoir for days or thousands of years. About 30% of freshwater on Earth is trapped in the sub-surface. Groundwater is the most important contribution to drinking water supply in many countries all around the world. Understanding how groundwater moves is important for predicting how quickly underground aquifers will be replenished when water is drawn up from wells drilled down from the surface. Sometimes, groundwater can become tainted with salt water, or even toxic compounds seeping from a contaminated source. In reality, impermeable rock and fractures can form pathways for the migration of contaminants. In these cases it is especially important to understand groundwater movement, in order to contain the spread and prevent contamination of aquifers used for drinking water or irrigation. See Figure 1.1 for a sketch of a groundwater case.

Groundwater discharges into lakes, ponds, streams, and wetlands. This usually occurs as underground seepage. It is important to understand that groundwater and surface water are connected, and what happens to one can affect the other. Pollution of groundwater resources can therefore seriously endanger lives. Thus, it is important to determine flow and transport paths in the subsurface. Since the material in the subsurface consists of porous media, mathematical and numerical models as used for oil reservoir simulation may also be used within groundwater flow.

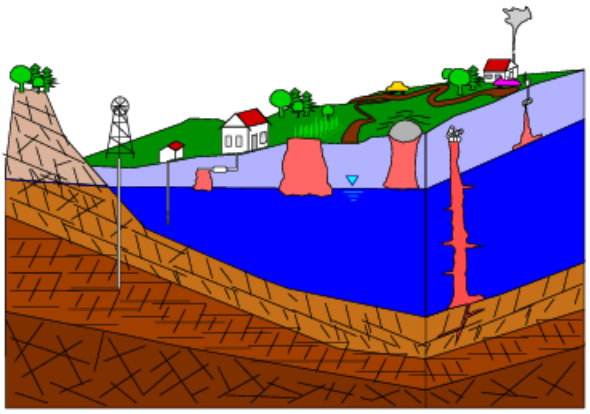


Figure 1.1: Sketch of a groundwater case. *Courtesy of R.Helmig, Universität Stuttgart.*

1.3 Naturally Fractured Medium

Modelling fractured reservoirs is an important issue in reservoir simulation. A naturally fractured reservoir can be defined as a reservoir containing planar discontinuities created by natural processes like diastrophism and volume shrinkage, see Figure 1.2 for illustrations of fractured media.

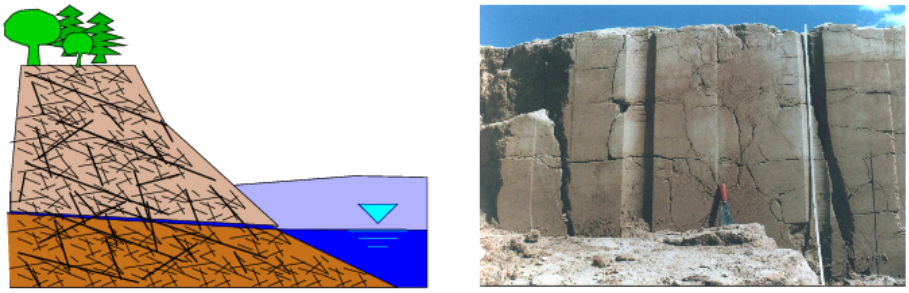


Figure 1.2: Illustrations of a fractured media. *Courtesy of R.Helmig, Universität Stuttgart.*

Fractured reservoirs are complex geological structures, where fractures have higher permeability and porosity than the surrounding rock (matrix). Although the aperture of fractures is very small compared to the dimensions of the reservoir, the fracture network often forms the primary pathway for fluid flow and mass transfer, and has a significant impact on the flow characteristics of the porous medium. Even though the porosity of the rock matrix is very small, the storage capacity of the rock matrix is higher than the storage capacity of the fracture system. This is due to the fact that the total volume of the rock matrix is in general much higher than the total volume of the fracture system. In summary, a fractured porous medium represents a system, principally consisting of two components with contrary properties; the fracture system has high permeability, high porosity (if one can assume porosity to open fractures), low total volume, and low storage capacity; the rock matrix has low permeability, low porosity, high total volume, and high storage capacity. Because of these significant differences, the requirements for a model of the fracture system are absolutely different from those for a model of the rock matrix. The discontinuities of the rock matrix resulting from the fractures can be regarded as a special case of heterogeneities, where the fractures have totally different properties than the rock matrix. Due to the complex geometries and potentially large variations in parameter values, fractures impact the flow pattern, and accurate representation of naturally fractured reservoirs represents a challenge for the characterisation, modelling, and simulation of petroleum and groundwater reservoirs, see [17, 45, 15, 39, 50].

Fracture-matrix models are in general described by a discrete or a multi-continua model (often denoted a multi-porosity model), see [45, 24, 49, 18, 19]. In a discrete model, the fractures are considered as discrete structures integrated in the surrounding rock matrix. With such a model, we have the possibility to model single- and multi-phase flow and transport processes very similar to nature. The fractures can be modelled equidimensionally (2D fractures in 2D space), or

as lower-dimensional elements (1D fractures in 2D space). Using multi-continua models, an representative elementary volume cannot only be obtained for porous medium, but also for the fractured system. The sub-domains is homogenised based on averaged parameters for rock matrix and fracture system. Thus, different flow and transport models can be used for the different continua. In these approaches, single-continuum, double-continua and multi-continua approaches are distinguished. The continua are coupled by exchange parameters. Exchange terms describing the interaction between the matrix system and the fracture system are very important using multi-continua models, see e.g.[38].

1.4 Fast Simulation of Transport in Porous Media

Reservoir simulation is based on mathematical models in the form of a coupled system of partial differential equations relating the dynamics of the trapped fluids and physical properties of the reservoir.

To compute the transport of fluids in a porous medium one, has to use numerical methods to solve the differential equations with corresponding boundary conditions for fluid pressure and velocity, and the equations for the transport of each fluid phase. Numerical solutions of the transport equations are often a bottleneck in current reservoir simulators. With the advances in numerical methods and computer technology, it is a continuous need of faster, more robust and more accurate reservoir simulators.

In this thesis we study a particular solution technique for a class of hyperbolic transport equations to approach the solution of incompressible flow of fluids in heterogeneous, unfractured, and fractured, porous media. Most of the work focus on *advective* transport in a porous medium completely filled with fluids of a single phase to describe quantities like time-of-flight, tracer concentration, contaminants concentration, etc. The velocity is given, time-independent and divergence free. We apply a higher order *discontinuous Galerkin* (dG), method combined with a prior optimal reordering, to the time-of-flight equation and the stationary tracer equation on different computational grids. Our motivation for studying these equations comes from transport in porous media, where equations of 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 these equations may predict the flow pattern in a reservoir and is therefore of great importance in areas such as oil recovery and groundwater hydrology.

Since the porous media considered in reservoir simulation commonly are strongly heterogeneous (and often fractured), the discontinuous Galerkin method is well suited to capture the discontinuities in the solution. The discontinuous Galerkin methods were introduced in 1973 by Reed and Hill [44] as a technique

to solve neutron transport problems. The methods quickly found use in a wide variety of applications; see e.g., [12, 25, 26, 10, 4, 46]. The first numerical analysis of the method for a linear advection equation was presented by Lesaint [30], while Lin and Zhou [31] provided convergence of the method for nonsmooth solutions. Cockburn and Shu [13, 14] analysed and extend the original dG method to systems of hyperbolic conservation laws and convection dominated problems. Recently, the technique has become popular as a method for solving fluid dynamics or electromagnetic problems. A general overview of the method is available in [11].

The dG method requires only a simple treatment of the boundary conditions to achieve uniformly high-order accuracy and has several appealing properties. The method is very well suited to handle complicated geometries and can easily handle irregular meshes with hanging nodes and approximations that have polynomials of different degrees in different elements. Hence, the dG methods can easily handle *hp*-adaptivity strategies since refinement or coarsening of the grid can be achieved without taking into account the continuity restrictions typical of conforming finite element methods.

Since the approximate solution of the dG method does not have to satisfy any inter-element continuity constraint, it produces mass matrices that are block-diagonal. This renders the method highly parallelisable. Other important properties of the dG methods are higher order convergence and local conservation of mass.

Due to their flexibility, dG methods are popular among the finite element community and they have been applied to a wide range of computational fluid problems.

An important distinction between the dG method and the usual finite-element method is that in the dG method the resulting equations are local to the generating element. The solution within each element is not reconstructed by looking to neighbouring elements. Its compact formulation can be applied near boundaries without special treatment, which greatly increases the robustness and accuracy of any boundary condition implementation; see [6, 7, 27, 51, 9] for details about general finite element methods.

A drawback of the method is the presence of small undershoots when solving problems using higher-order basis functions. The dG method has difficulty to capture shocks or sharp gradients in the solution without creating spurious oscillations. This problem may be handled by introducing a slope limiter. In our work, we do this by reducing the order of the basis functions used in the dG scheme together with grid refinement in the areas where the problems occur.

We restrict our attention to transport problems with positive characteristics, i.e., models where all waves in the solution travel along (integral curves of) the velocity field. By using numerical methods with compact stencils and upwind flux

approximations, the directness of the underlying differential equation is preserved at the discrete level. The dependency between two neighbouring grid blocks is directional such that the solution in one grid element can (and must) be computed before the solution in the other elements. By taking advantage of the directional dependency between elements we may reorder the elements in a suitable sequence, such that the global system becomes block triangular, see Duff and Reid [20].

In our numerical model for computing transport in porous media, we combine the dG method with an upwind numerical flux function. The numerical flux function creates a one-sided dependency by viewing the elements in the inter-element fluxes as vertices and edges in a directed graph. With topological sort of the graph, we produce an optimal ordering of the elements, allowing for the discrete global system to be decoupled in a sequence of linear problems. This way, it is not necessary to assemble the full system, but we may compute the solution in an element-by-element fashion. The optimal reordering algorithm can reduce the memory requirements, implementation complexity, and computational costs, thereby increasing the size of the problems one may solve (on a single computer).

The dG method's robustness with respect to strongly discontinuous coefficients renders it very attractive for porous medium flow and transport calculations. In this work we consider accurate and efficient solvers for computing transport in unfractured and fractured porous media. The results may predict the flow pattern in a reservoir. For reservoir simulation, the time-of-flight gives the timelines in the reservoir, whereas computing the tracer distribution can determine the spatial regions swept or drained by a fluid from a source or a sink. Within groundwater applications, the evaluation of the time-of-flight may be a important tool to visualize the spreading of contaminants and to help understanding the different transport processes.

Chapter 2

Physical Background and Mathematical Modelling

Simulation of the processes occurring in the reservoir incorporates elements from all main branches of science: geology, physics, chemistry, biology and mathematics. This work approaches reservoir simulation from a mathematical point of view, and some physical assumptions must be made to complete the mathematical approach.

The theory leads to the partial differential equations that describe the fluid flow in a porous medium. This set of equations is often the starting point for applied mathematics, and discretisation of the equations is a field that has received a lot of attention in the past decades. The partial differential equations arise from the principle of conservation and a physical description of how the velocity depends on the pressure (Darcy's law).

In this chapter, we present the principle of conservation, and briefly give the primary physical and geological parameters influencing fluid flow in porous media, before we introduce Darcy's law. With this we provide mathematical models for immiscible one-phase and multi-phase flow, assuming some common simplifications. Further theory details may be found in, for example, [3, 9, 23, 41].

2.1 Conservation Laws

Consider a conserved quantity u inside a volume V of a porous medium having the closed surface ∂V and outward normal vector $\mathbf{n}$; see Figure 2.1. The change of u inside V corresponds to the total flow in and out of V and the amount generated by sources or removed by sinks within V. On integral form, the conservation of u

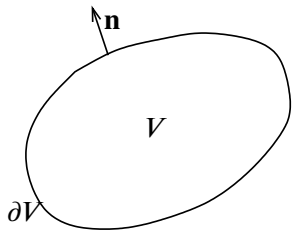


Figure 2.1: The volume V has a closed surface, ∂V. The surface normal vector, $\mathbf{n}$, is directed out of the volume.

is expressed by

$$\int_V \frac{\partial}{\partial t} u \, dV + \int_{\partial V} \mathbf{f} \cdot \mathbf{n} \, dS = \int_V q \, dV, \tag{2.1}$$

where $\mathbf{f}$ is the flux through V and q corresponds to source or sink terms. Using Gauss' divergence theorem, we may convert the boundary flux integral into a volume integral and obtain the equation,

$$\int_V \left(\frac{\partial}{\partial t} u + \nabla \cdot \mathbf{f} - q \right) dV = 0. \tag{2.2}$$

Since this must hold for any volume V inside some domain, and the change of u is assumed to be continuous, we may drop the integrals. Hence, we obtain the conservation law in differential form

$$\frac{\partial}{\partial t} u + \nabla \cdot \mathbf{f} = q. \tag{2.3}$$

This equation is denoted the *continuity equation*.

Starting from the equation (2.3), we can use a conservation law to describe how the fluids flow in porous medium. In reservoir flow modelling, the conserved quantity can be the mass density of a phase and we shall in the following chapters give expressions for u and its flux $\mathbf{f}$.

2.2 Model Parameters

2.2.1 Porosity

Porosity is a measure of the void spaces in a material; the more porous a material is, the greater amount of void space it contains. The total porosity of a rock volume is a measure of how much fluid the rock can contain. Porosity of a medium is defined by

$$\phi = \frac{V_{void}}{V_{bulk}}, \tag{2.4}$$

where V_{void} is the total void volume that exists between rock grains in a total or bulk volume V_{bulk} of rock.

In reality, the pore space is only partly available for flow due to dead ends and isolated pores. If V_{void} is the available (effective) pore volume, then (2.4) describes the effective porosity, that is, the volume fraction of the porous medium available for flow. Commonly, the term porosity refers to the effective porosity.

2.2.2 Permeability

Just as the porosity of soil affects how much water it can hold, it also affects how quickly water can flow through the soil. The permeability is a measure of the ability of a porous medium to transmit fluids. Since porosity measures the volume available to fluid flow, there is a strong correlation between porosity and permeability. The permeable regions of a rock correspond to available regions of pore space. In general, rocks with coarser grain sizes are more permeable than rocks with finer grains, because they contain larger and more interconnected pores.

The permeability is a property of the porous medium only, not the fluid. In naturally occurring materials, it ranges over many orders of magnitude. A common unit for permeability is Darcy, or more commonly milliDarcy (1 Darcy $\approx 0.987 \cdot 10^{-12}\ m^2$).

Permeability is a constant if and only if the medium is homogeneous. If permeability varies with spatial location, we say that the porous medium is heterogeneous. In an isotropic medium the resistance to fluid flow is equal in all directions and the permeability is described by $\mathbf{K} = k\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. If the medium is anisotropic, the resistance to flow may be directional dependent and the permeability is described by a symmetric tensor $\mathbf{K} = \{k_{ij}\}$.

Permeability is part of the proportionality constant in Darcy's law (see next section) which relates flow rate and fluid physical properties (e.g., viscosity), to a pressure gradient applied to the porous media. The proportionality constant specifically for the flow of water through a porous media is the hydraulic conductivity:

$$k = \frac{\mathbf{K}\rho g}{\mu}, \tag{2.5}$$

where μ is the viscosity, ρ is the density, and g is the gravitational constant.

2.3 Mathematical Models

2.3.1 Darcy's law

Darcy's law describes the flow of a fluid through a porous medium. The law was formulated by the French engineer Henry Darcy in the 19th century when he experimented with flow of water through different types of sand. In a range of experiments with flow vertically through beds of sand, he concluded that the flow through the sands was proportional to the pressure difference between the top and bottom pressure. For single phase flow and neglected gravity, the relation says that the volumetric flow velocity (Darcy velocity) $\mathbf{v}$ is proportional to the gradient of the pressure P, and the law reads

$$\mathbf{v} = -\frac{\mathbf{K}}{\mu}\nabla P, \tag{2.6}$$

where $\mathbf{K}$ is the permeability tensor and μ the viscosity. The equation states that the fluid will move from high pressure to regions of lower pressure, and the velocity is dependent on the medium and phase conductivities.

The extended Darcy's law including gravity effects is given by

$$\mathbf{v} = -\frac{\mathbf{K}}{\mu}(\nabla P + \rho g \nabla h), \tag{2.7}$$

for single phase flow, where ρ is the fluid density, g is the acceleration of gravity, and h is the reservoir height above some reference plane.

Assumptions needed for the derivation of Darcy's law are low flow velocities and significant friction between the fluids and the pore walls. Also, the porous medium is assumed to be rigid, and not compacted due to fluid flow.

The volumetric flow velocity is the effective flow velocity across a representative volume, and thus does not describe the pore velocity, which is typically larger. The pore velocity would be the velocity a passive particle would experience if carried by the fluid through the domain. The pore velocity is related to the volumetric flow velocity by the porosity. The velocity is divided by porosity to account for the fact that only a fraction of the total formation volume is available for flow. Hence, the pore velocity is expressed by

$$\mathbf{v}_p = \frac{\mathbf{v}}{\phi}. \tag{2.8}$$

2.3.2 One-Phase Flow

Consider any fixed volume V inside a (stationary) porous medium with porosity ϕ. For one-phase flow, the conserved density inside volume V is $u = \rho\phi$, where

ρ is the fluid density. The flux, $\mathbf{f}$, is simply the mass density times the velocity, $\mathbf{f} = \rho\mathbf{v}$. In this case we obtain the following continuity equation (2.3) for one-phase flow:

$$\frac{\partial}{\partial t}(\rho\phi) + \nabla \cdot (\rho\mathbf{v}) = q, \tag{2.9}$$

where $\mathbf{v}$ is the volumetric velocity given by (2.7).

Assume incompressible fluid, and constant ρ and ϕ. The bulk motion of the fluid in terms of the common pressure P and the volumetric velocity without gravity effect may be written

$$\nabla \cdot \mathbf{v} = \frac{q}{\rho}, \quad \mathbf{v} = -\frac{\mathbf{K}}{\mu}\nabla P. \tag{2.10}$$

When no sources or sinks are present, we obtain the following equations:

$$\nabla \cdot \mathbf{v} = 0, \quad \mathbf{v} = -\frac{\mathbf{K}}{\mu}\nabla P. \tag{2.11}$$

2.3.3 Multi-Phase Flow

When several phases (or components) are present in a porous medium, conservation must be posed for each of the phases (or components).

The efficient pore space is assumed to be filled with fluid at all times. Depending on the processes, the different phases will compete to occupy this space. The phase saturation, S_i, of phase i is the fraction of effective pore space occupied by the different fluids (phases):

$$S_i = \frac{V_i}{V_{void}}, \tag{2.12}$$

where V_i is the volume occupied by the specific phase, and V_{void} is the efficient pore space. Since the pores are always fully saturated, we have

$$\sum_i S_i = 1, \tag{2.13}$$

where we assume the summation to be over all the phases.

The conserved mass density for each phase is $\rho_i\phi S_i$, where ρ_i is the density and S_i is the saturation of fluid i. The flux can be written $\mathbf{f} = \rho_i\mathbf{v}_i$, where $\mathbf{v}_i$ denotes the volumetric flow velocity for each fluid. For immiscible flow, the conservation of mass for each phase (fluid) i is

$$\frac{\partial}{\partial t}(\rho_i\phi S_i) + \nabla \cdot (\rho_i\mathbf{v}_i) = q_i, \tag{2.14}$$

by using (2.3). Each phase may have distinct sources q_i and the volumetric flow velocity for each fluid is given by a generalised Darcy's law for multi-phase flow:

$$\mathbf{v}_i = -\mathbf{K}\lambda_i(\nabla P_i + \rho_i g \nabla h), \tag{2.15}$$

where P_i is the pressure of fluid phase i. The relative mobility λ_i of fluid phase i is defined by

$$\lambda_i = \frac{k_{ri}}{\mu_i}, \tag{2.16}$$

where μ_i is the fluid viscosity. Since the presence of more than one fluid generally inhibits flow, the relative permeability $k_{ri} = k_{ri}(S_i)$ accounts for the ability of the fluid to flow in the presence of the other phases.

For multiphase flow, we have phase pressures that typically are related through a quantity called the capillary pressure. The capillary pressure is defined as the difference between the local pressures of the fluids. The capillary pressure reflects the fact that the pressures at each side of a fluid-fluid interface differ due to interfacial tension. When only two phases are present, the capillary pressure is given by the pressure difference between the non-wetting and the wetting fluid:

$$P_c(S_w) = P_{nw} - P_w. \tag{2.17}$$

For multi-phase flow we get similar relations between the different phases.

2.4 Transport Models

As just presented, the flow of fluid 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

$$\frac{\partial}{\partial t}(c_{\alpha i}\rho_i \phi S_i) + \nabla \cdot (c_{\alpha i}\mathbf{v}_i \rho_i) = c_{\alpha_i} q_i, \quad \alpha = 1, ..., m, i = 1, ..., l. \tag{2.18}$$

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

Chapter 3

Hyperbolic Transport Equations

We will mainly focus on advective transport in a porous medium completely filled with fluids of a single phase, and propose fast procedures for solving a class of hyperbolic transport equations. The two equations we investigate are the stationary tracer equation and the time-of-flight equation. Our motivation for studying these equations comes from transport in porous media, where equations of 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 these equations may predict the flow pattern in a reservoir and is therefore of great importance both in petroleum engineering and groundwater management.

In the following, 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 or a unstructured triangular grid.

3.1 The Time-of-Flight Equation

Before introducing the time-of-flight equation, we will define streamlines.

Streamlines are a family of curves $s(\tau)$ that are instantaneously tangent to the velocity vector $\mathbf{v}$ at every point, that is,

$$\frac{ds}{d\tau} = \mathbf{v}. \tag{3.1}$$

Hence, all instantaneous transport occur along streamlines since the motion of the fluids is aligned with the velocity $\mathbf{v}$; see Figure 3.1. For a steady state velocity field the streamlines are traced out by a passive particle moving with the flow.

For incompressible flow, streamlines defined at a single instant do not cross each other. If they were to cross, this would indicate two different velocities at the same point, which is not physically possible. Hence, any particles of fluid starting on one streamline will stay on that same streamline.

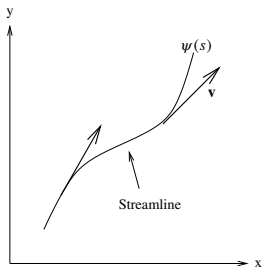


Figure 3.1: A streamline, $\psi(s)$, that is everywhere tangent to the flow field.

The *time-of-flight* $\tau(\mathbf{x})$ measures the time a particle needs to travel along the streamline from its initial position on the inflow boundary $\partial\Omega^-$ to a given point $\mathbf{x}$.

Isocontours of $\tau(\mathbf{x})$ are the time-lines in the porous medium and give information about the flow pattern, in particular for single phase flow. In a streamline setting, the time-of-flight $\tau(\mathbf{x})$ is usually given by the integral

$$\tau(\mathbf{x}) = \int_{\psi(s)} \frac{\phi}{|\mathbf{v}(\mathbf{x})|} ds, \tag{3.2}$$

evaluated along the streamline $\psi(s)$, where s denotes the distance along the streamline. The streamline $\psi(s)$ connects $\mathbf{x}$ to the inflow boundary $\partial\Omega^-$. The scaling with ϕ is necessary because the porosity of the medium will speed up the flow, and hence decrease $\tau(\mathbf{x})$.

Alternatively, we take the directional derivative of $\tau(\mathbf{x})$ along a streamline in the direction of the velocity field and obtain

$$\frac{d\tau}{ds} = \frac{\mathbf{v}}{|\mathbf{v}|} \cdot \nabla\tau = \frac{\phi}{|\mathbf{v}|}. \tag{3.3}$$

Hence, we get the time-of-flight equation as

$$\mathbf{v} \cdot \nabla\tau = \phi, \tag{3.4}$$

with boundary condition

$$\tau|_{\partial\Omega^-} = 0,$$

see e.g., [16].

Hence, a simple model for advective transport in a porous medium completely filled with fluids of a single phase is the following linear boundary-value problem for the time-of-flight τ in Ω:

$$\mathbf{v} \cdot \nabla\tau = \phi, \quad \tau|_{x\in\partial\Omega^-} = 0, \tag{3.5}$$

where $\partial\Omega^-$ denotes the inflow boundary of the computational domain Ω.

3.2 The Stationary Tracer Equation

For incompressible flow, any streamline connects an injector and a producer, and we may find which part of the pore volume is connected to which injector and producer, and thereby deriving reservoir compartmentalisation. 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 obtains information about well connectivity and affected areas. Determining the spatial region swept by a fluid source (an inflow boundary), or vice versa, the spatial region from which fluid is drained by a sink (an outflow boundary), is of practical importance both in groundwater management and petroleum engineering.

To derive the stationary tracer equation, let c_α denote the volume fraction of the water phase occupied by the tracer substance α and consider mass conservation. Assuming single-phase flow (i.e., $l = 1$ in (2.18)) and incompressible flow outside sinks and sources, the individual distribution of the various components are given in terms of the *linear* (hyperbolic) transport equation,

$$\frac{\partial}{\partial t}(\phi c_\alpha) + \nabla \cdot (c_\alpha \mathbf{v}) = c_\alpha q. \tag{3.6}$$

Each tracer is transported according to the volumetric velocity field $\mathbf{v}$. For simplicity, we assume that $\mathbf{v} = \mathbf{v}(\mathbf{x})$ is given and time-independent, divergence free, and nearly irrotational. The evaluation of concentration c_α of tracer α is then governed by

$$\frac{\partial}{\partial t}(\phi c_\alpha) + \mathbf{v} \cdot \nabla c_\alpha = 0. \tag{3.7}$$

To derive the stationary tracer equation, we let $\frac{\partial(\phi c_\alpha)}{\partial t} = 0$, and thus

$$\mathbf{v} \cdot \nabla c_\alpha = 0. \tag{3.8}$$

If there are m injectors, the swept volumes of each injector may be revealed by successively launching tracer substance at each injector. This is accomplished by setting the tracer concentration to 1 in the particular injector, and 0 in the other injectors. Thus we get an equation for each injector,

$$\mathbf{v} \cdot \nabla c_\alpha = 0, \quad c_{\alpha|_{\partial\Omega_\alpha}} = 1, \quad \alpha = 1, ..., m, \tag{3.9}$$

where $\partial\Omega_\alpha$ is the injector launched with tracer substance α.

Notice that the stationary tracer equation is a special case of the time-of-flight equation where $\phi(\mathbf{x}) = 0$ everywhere in the reservoir.

3.3 Semi-Discrete Model of Multiphase Flow

Equation (2.14) and (2.15) are the building blocks in modelling nonlinear transport of l phases or components through a porous medium. By simple arguments and assuming divergence-free velocity field and incompressible medium, this can be rewritten as a system of nonlinear conservation laws for the saturation, or volume fractions, $u = (u_1, ..., u_{l-1})$:

$$\phi \frac{\partial}{\partial t} u_i + \mathbf{v} \cdot \nabla f_i(u) = q_i, \quad i = 1, ..., l-1. \tag{3.10}$$

Here f_i is the fractional flow function modelling the speed of each phase or component i relative to the mean (or total) velocity $\mathbf{v}$.

Using a implicit scheme, a temporal semi-discretised version of (3.10) is obtained

$$\phi u^n + \Delta t \mathbf{v} \cdot \nabla f(u^{n-1}) = \phi u^{n-1} + \Delta t q, \tag{3.11}$$

where u^{n-1} and u^n are approximations of $u(\cdot, t^{n-1})$ and $u(\cdot, t^n)$, respectively.

The methodology using implicit discontinuous Galerkin schemes for multiphase flow is briefly mentioned in Paper B. See Natvig and Lie [37, 36] for a thoroughly study of this solution procedure for multiphase flow.

Chapter 4

Numerical Methods

The transport equations presented in Chapter 3 can be written on the general form

$$\begin{aligned} \mathbf{v} \cdot \nabla u &= H(u, \mathbf{x}), && \text{for } \mathbf{x} \in \Omega \\ u &= h(\mathbf{x}, t), && \text{for } \mathbf{x} \in \partial\Omega^{-}, \end{aligned} \tag{4.1}$$

where $\mathbf{v}$ is a given (divergence-free) vector field and $\partial\Omega^{-}$ denotes the inflow boundary of a porous medium Ω. Solving (4.1) is rather easy for smooth $\mathbf{v}$, but becomes harder when the vector field has large spatial variations and exhibits fine-scale details that are important for the global flow pattern.

To obtain a numerical solution of (4.1), we use a higher order discontinuous Galerkin (dG) method. The dG method is feasible to capture discontinuities in the solution, which commonly appears due to strong heterogeneous characteristics of the porous media. Furthermore, the dG method is well suited for handling complicated geometries. In Paper D, we extend the methodology used on rectangular grids in Papers A, B and C, to unstructured triangular grids.

Discontinuous Galerkin methods in general lead to globally coupled systems of equations, which typical requires inversion of large band structured matrixes. On the other hand, a dG formulation in combination with an upwind flux approximation does not. The upwind flux preserves the directional dependency in the solution and we may find an optimal reordering of the elements that enables us to solve the equations locally element by element.

4.1 The Discontinuous Galerkin Formulation

The discretisation in a dG method starts with a variational formulation as in a standard Galerkin method, but allows for discontinuities over the element edges. To get a variational formulation of (4.1), we start by partitioning the domain Ω into a set of non-overlapping elements $\{K\}$. In the following, Ω is assumed to

be partitioned into elements such that a *face* refers to a line in $2D$ or a plane in $3D$. Faces are either boundary faces or shared by two neighbouring elements (inter-element faces).

In a dG formulation, one seeks an approximation to the solution in a space V of elementwise smooth functions that may be discontinuous over inter-element faces. We say that

$$V = \{\varphi \in L^2(\Omega) : \varphi|_K \text{ sufficiently smooth}\}, \tag{4.2}$$

where L^2 is the space of quadratic integrable functions.

To obtain a variational formulation of (4.1), we multiply (4.1) with a test function $\varphi \in V$ and integrate over each element $K \in \Omega$,

$$\int_K (\mathbf{v} \cdot \nabla u)\varphi d\mathbf{x} = \int_K H(u, \mathbf{x})\varphi d\mathbf{x} \quad \forall K, \forall \varphi \in V. \tag{4.3}$$

Integrating the left-hand side by parts, we obtain

$$-\int_K (u\mathbf{v}) \cdot \nabla\varphi d\mathbf{x} + \int_{\partial K} (u\mathbf{v}) \cdot \mathbf{n}\varphi ds = \int_K H(u, \mathbf{x})\varphi d\mathbf{x} \quad \forall K, \forall \varphi \in V, \tag{4.4}$$

where $\mathbf{n}$ denotes the unit outward normal on the element faces ∂K. The possible discontinuities over inter-element boundaries force us to use a consistent and conservative numerical flux function $\hat{f}(u, u^{ext}, \mathbf{v} \cdot \mathbf{n})$ to replace the flux term $(u\mathbf{v}) \cdot \mathbf{n}$ (we will come back to this later).

We seek solutions in a finite-dimensional subspace $V_h^{(n)} \subset V$, where $V_h^{(n)}$ is chosen as the space of piecewise smooth functions of order n that may be discontinuous over inter-element faces. We replace the exact solution u and the test function φ by $u_h \in V_h^{(n)}$ and $\varphi_h \in V_h^{(n)}$, respectively, and obtain a discrete variational formulation. The approximate solution u_h is determined as the unique solution of the following weak formulation: let

$$a_K(u_h, \varphi_h) = -\int_K (u_h\mathbf{v}) \cdot \nabla\varphi_h d\mathbf{x} + \int_{\partial K} \hat{f}(u_h, u_h^{ext}, \mathbf{v} \cdot \mathbf{n})\varphi_h ds, \tag{4.5}$$

$$b_K(H(u_h), \varphi_h) = \int_K H(u_h, \mathbf{x})\varphi_h d\mathbf{x}, \tag{4.6}$$

and find $u_h \in V_h^{(n)}$ such that

$$a_K(u_h, \varphi_h) = b_K(H(u_h), \varphi_h) \quad \forall K, \forall \varphi_h \in V_h^{(n)}. \tag{4.7}$$

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

$$\hat{f}(p, p^{ext}, \mathbf{v}\cdot\mathbf{n}) = p\max(\mathbf{v}\cdot\mathbf{n}, 0) + p^{ext}\min(\mathbf{v}\cdot\mathbf{n}, 0), \tag{4.8}$$

where p and p^{ext} denote the inner and outer approximations at the element faces, respectively. The upwind flux preserves the directional dependency in the solution, which is crucial for our solution procedure. With this flux function, the only neighbours affecting the solution on each element are the upwind neighbours.

To find a solution of (4.7), let $L^k = \{L_1^k, ..., L_m^k\}$ be some basis for $V_h^{(n)}$ on each element, where m is the number of *degrees-of-freedom* on K. The approximate solution u_h on an element K can be written as a linear expansion of basis functions,

$$u_h(K) = \sum_i^m u_i^k L_i^k, \quad \forall K, \tag{4.9}$$

where $\{u_i^k\}$ are the unknown coefficients of the solution in element K. We successively let $\varphi_h = L_j^k$ and substitute the approximate solution (4.9) into the variational formulation (4.7). We will approximate the resulting integrals with appropriate quadrature rules and get a set of linear equations for the degrees-of-freedom in each element K.

Let U denotes the vector of unknown coefficients u_i^k of all elements K in Ω, and let U_K be the vector of unknowns for element K. In element K, we have

$$A_K U = B_K, \tag{4.10}$$

where $(A_K)_{ij} = a_K^h(L_i^k, L_j^k)$ and $(B_K)_i = b_K^h(H(L_i^k), L_j^k)$, and a_K^h and b_K^h are numerical approximations to the integrals in (4.7) using Gaussian quadrature.

For convenience, we split the coefficient matrix of the unknowns into the element stiffness matrix R_K and the numerical flux integral F_K. The flux integral gives the coupling to the other elements. In element K, we have

$$A_K U = -R_K U_K + F_K U. \tag{4.11}$$

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_K are in general determined by the choice of numerical flux. We assume that $\mathbf{v}\cdot\mathbf{n}$ is constant on all element faces in Ω (i.e., that $\mathbf{v}\cdot\mathbf{n}$ does not change sign on element faces). The boundary integral $F_K = \int_{\partial K} \hat{f}(u_h, u_h^{ext}, \mathbf{v}\cdot\mathbf{n})\varphi_h\, ds$ may be split in an integral F_K^+ over the outflow faces ∂F_K^+, and an integral F_K^- over the inflow faces ∂F_K^- according to the definition of the numerical flux function. If $\mathbf{v}$ is computed using a standard low-order discretisation method for (2.11), 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 be constant on each element face.

Let $\mathcal{U}(K)$ be the set of neighbouring elements on the upwind side of K. Formally, $\mathcal{U}(K)$ consist of all elements E such that $(\mathbf{v}\cdot\mathbf{n}_K)|_{\partial E\cup\partial K} < 0$, where $\mathbf{n}_K$ is the outward-pointing normal to K. The upwind flux (4.8) can be written

$$F_K U = F_K^+ U_K + F_K^- U_{\mathcal{U}(K)}, \tag{4.12}$$

where F_K^+ denotes the flux *out* of element K, F_K^- denotes flux *into* element K, and $U_{\mathcal{U}(K)}$ is the vector of the degrees-of-freedom for all the neighbouring elements of K in the upwind direction.

Thus, we may write (4.10) as

$$A_K U = -R_K U_K + F_K^+ U_K + F_K^- U_{\mathcal{U}(K)} = B_K \quad \forall K. \tag{4.13}$$

The operators and the matrices are given by

$$(R_K)_{ij} = \int_K (L_i^k \mathbf{v})\cdot\nabla L_j^k d\mathbf{x}, \tag{4.14}$$

$$(F_K^+)_{ij} = \int_{\partial K^+} L_i^k L_j^k \mathbf{v}\cdot\mathbf{n} ds, \tag{4.15}$$

$$(F_K^-)_j = \int_{\partial K^-} L_i^{k_{ext}} L_j^k \mathbf{v}\cdot\mathbf{n} ds, \tag{4.16}$$

$$(B_K)_j = \int_K H(u_i^k L_i^k, \mathbf{x}) L_j^k d\mathbf{x} \tag{4.17}$$

Henceforth, we use dG(n) to denote the discontinuous Galerkin approximation of polynomial order n. If we want to find a first-order solution, meaning that u_h is element-wise constant and $A_K U$ is a linear system of equations, we get a $N_e \times N_e$ linear equation set, where N_e is the number of elements. Each row of the linear system corresponds to one unique element and has nonzero entries only on the diagonal and on the entries that correspond to the upwind neighbours. For dG(n) methods with a linear A we analogously get a block structured $N_e m \times N_e m$ system, where m is the number of degrees-of-freedom per element. Hence, $V_h^{(0)}$ is the space of elementwise constant functions and yields a scheme that is formally first-order accurate, $V_h^{(1)}$ is the space of elementwise linear approximations (bilinear approximations for Cartesian grid) and yields a scheme that is formally second-order accurate, etc. The results from Papers A and D show that the error of a dG(n)-method decays with order $n+1$ for smooth solutions. On nonsmooth solutions, slower convergence is to be expected.

If A is triangular, we may solve (4.7) one equation at a time. In the following section, we describe a procedure to find a column and row permutation of A such that A is block triangular, and therefore possible to solve in a sequence of smaller equation systems.

4.2 The Optimal Reordering

In the following, we consider the time-of-flight equation (3.5), which simplifies (4.7) to a linear system (since $H(u, \mathbf{x}) = \phi$):

$$a_K(u_h, \varphi_h) = b_K(\phi, \varphi_h) \quad \forall K, \forall \varphi_h \in V_h. \tag{4.18}$$

Although (4.7) and (4.18) have different structure on a given element K, the two global systems will have a similar block structure. The ideas presented for the linear case (4.18) will therefore immediately carry over to the nonlinear case (4.7).

An efficient solution procedure is obtained by taking advantage of the fact that (4.1) has one-sided domain of dependence. In other words, both the exact and the numerical solutions in any element are 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 [44] in the context of neutron transport. To our knowledge, Paper A was the first to apply the idea to transport in porous media.

From a computational point of view, it is more convenient to look at this as a reordering of unknowns. Observe that we can solve (4.13) element by element if there is a flux from element K_i to element K_j such that we can determine a sequence of elements such that i appears before j in the sequence. By processing the elements in such a sequence, equation (4.13) may be written

$$-R_K U_K + F_K^+ U_K = B_K - F_K^- U_{\mathcal{U}(K)}, \tag{4.19}$$

such that the right-hand side of (4.19) is a known quantity in each step. Since the directions of the fluxes are determined solely by $\mathbf{v}$ (and not by U), this sequence can be computed as part of a preprocessing step before solving the system (4.19).

The idea of solving boundary-value problems for advective transport sequentially was also used in [5], but with a different spatial discretisation. In that paper, they used an algorithm to compute a suitable sequence based on physical arguments. The solutions were constructed by marching outwards from the inflow boundaries or sources. To do so, one needs to keep a list of candidated 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 our work, we choose a different approach. To find the sequence of elements we let the elements and fluxes together form a directed graph, where the elements

are the vertices and the fluxes are the directed edges. Hence, 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 sequence of elements exists, we have a *directed acyclic graph*, also called DAG. A directed acyclic graph is a directed graph with no paths that start and end at the same vertex. Every DAG has a topological sort, an ordering of the vertices such that each vertex comes before all vertices it has edges to. Hence, topological sorting is the process of assigning a linear ordering to the vertices of a DAG so that if there is an path from vertex i to vertex j, then i appears before j in the linear ordering. In other words, a topological sort is a permutation P of the vertices of the directed acyclic graph such that an edge (i, j) implies that vertex i appears before vertex j in P. Every DAG has one or more topological sorts.

The task of finding this sequence of vertices can be accomplished by a depth-first traversal of the reversed DAG (see e.g., [48]). A depth-first search (DFS) is an algorithm for traversing or searching a tree, a tree structure, or a graph. Intuitively, one starts at the root (e.g., a sink) checking its neighbours (neighbour is often denoted by child), expanding the first node among the children, checking if that expanded node is our goal destination or if the node has no children, and if not, continue exploring more nodes. We explores as far as possible along each branch before backtracking and then starting off on the next node. When DFS backtracks from a vertex, all vertices reachable from this vertex have already been explored.

Running the DFS on the DAG, the output will be the vertices in reverse order of finishing time. The desired topological sorting is the reverse of these searches. That is, we can construct the ordering as a list of vertices, by adding each vertex to the start of the list at the time when the depth-first search is processing that vertex and has returned from processing all children of that vertex.

An example is presented in Figures 4.1 and 4.2. Figure 4.1 shows the direction of flow for a domain with a source in Element 1 and a sink in element 5. Topological sort can be viewed as placing all the vertices (elements) along a horisontal line so that all directed edges go from left to right. Figure 4.2 shows horisontal lines for the ordering of elements before and after reordering using DFS. The directed edges (arrows) represent the fluxes between the elements. The upper line shows the ordering before reordering, where the graph does not have a one-sided dependence. Hence, the numerical solution can not be solved in an element-by-element fashion. The lower line shows the ordering of the elements using the topological sort, where all directed edges go from left to right. Hence, the numerical solution in any element is determined by the solution in the upstream elements.

Since each edge and vertex are visited once, the algorithm runs in linear time. The depth-first traversal takes $\mathcal{O}(N_e)$ operations for a graph of N_e vertices (elements). In most cases, the depth-first traversal will produce a sequence of ver-

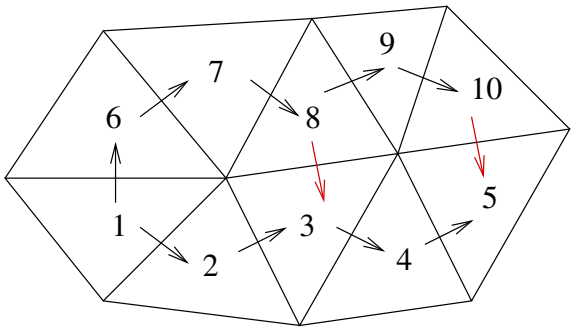


Figure 4.1: Direction of flow for a domain with a source in Element 1 and a sink in Element 5. The arrows show the fluxes between the elements.

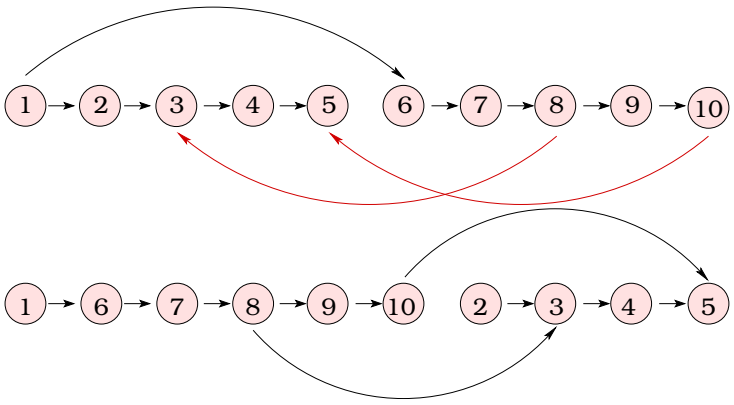


Figure 4.2: The ordering of elements and the dependency between the elements can be viewed by placing all the vertices(elements) along a horisontal line. Topological sort where a one-side dependency is created is viewed by all directed edges going from left to right, see lower ordering. The upper figure shows a ordering of the elements where the graph does not have a one-sided dependence.

tices that allows an elementwise computation of the solution. Reordering the elements reduces the computational effort needed to solve (4.19) from solving a large $(N_e m) \times (N_e m)$ system to solving N_e small linear $m \times m$ systems, where N_e is the number of elements, and m is the number of degrees-of-freedom per element. Thus, reordering is a simple way to obtain highly efficient linear solvers for large advection problems. See [37, 36] for discussions of how the same principle can be applied to decouple the solution of *nonlinear* systems arising in the implicit discretisation of multiphase transport equations.

If the velocity field $\mathbf{v}$ has a nonzero circulation, there are one or more *strongly connected components* in the graph, and the sequence of elements does not exist. Strongly connected components are groups of elements that are interdependent, and for these groups 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 $\mathcal{O}(N_e)$ operation. We assume that a topological sort can always be performed or that the solution algorithm is capable of solving elements simultaneously whenever connected elements are encountered.

4.2.1 Strongly Connected Groups of Elements

Search algorithms like DFS normally mark vertices they have already visited and do not visit them again. If they fail to do this, they may never terminate because they follow a cycle of edges forever. Each such cycle corresponds to strongly connected components, which is a subgraph where all vertices in the subgraph are reachable by all other vertices in the subgraph. Reachability between nodes is established by the existence of a path between nodes. A directed graph can be decomposed into strongly connected components by one extra depth-first traversal of the graph.

In the grid, cycles are groups of elements that are made mutually dependent by a nonzero circulation in the velocity field $\mathbf{v}$. The mutual dependence makes a topological sort impossible. For the discrete equations, this implies that the solution in such a collection of elements must be computed simultaneously.

By grouping elements corresponding to a strongly connected components in the dependency graph into a single vertex, we obtain an acyclic graph where each vertex corresponds to either a single element, or a single strongly connected component (i.e., a group of elements that form an irreducible block of degrees-of-freedoms). Thus, if our solver can compute the solution in groups of elements, we can still apply the sequential procedure.

A possible distribution of fluxes is depicted in Figure 4.3, where a small group of elements are strongly connected. In this situation, our reordering strategy still

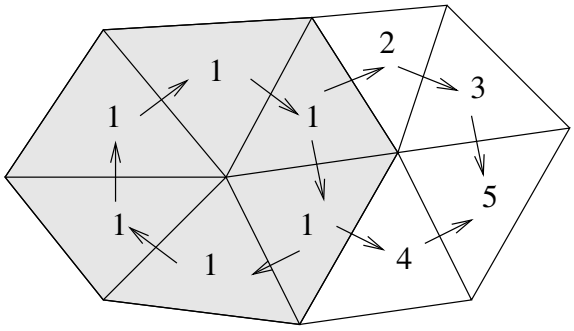


Figure 4.3: Direction of flow for a domain with a source in the lower left element and a sink in Element 5. A small group of elements are strongly connected and must be computed simultaneously.

works and gives one larger linear system associated with the six blocks of interconnected elements in addition to the usual four linear systems associated with single elements.

In general, the appearance of cycles in the dependency graph depends on both the modelling and on the numerical solver used to compute the velocity field $\mathbf{v}$. In most relevant cases of incompressible flow, the exact velocity field has zero circulation. Moreover, blocks of interconnected elements do not appear if one uses a two-point flux-approximation for the pressure equation (2.11). On the other hand, a mixed finite-element solution may produce velocity fields with nonzero circulation. In our experience, cycles that appear in velocity fields computed by mixed finite-element method are small and sparse for incompressible and weakly compressible flows.

The same idea can be applied when $\mathbf{v}$ is computed by a higher-order method and it may occur that the flux changes sign at element faces. This is a reflection of the fact that although streamlines do not cross, they may pass through a grid cell more than once. Consider 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 the 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 not acyclic graph with two-way edges, each two-way edges must be replaced with two one-way edges. The corresponding cycle can be automatically detected as before, and the neighbouring elements can easily be treated as on a cycle and be lumped together.

4.3 Representation of Computational Domains

An essential decision is how to represent the geometry (the shape of the region). Creating a mesh is the first step in a wide range of applications. To find an unstructured simplex mesh we need a choice of meshpoints (vertex nodes) and a partition of the computational domain Ω; see [40] for a simple mesh generator. The standard way of representing a mesh is by the node positions and the element indices. A N_n-by-d matrix X gives the coordinates of the nodes, and a N_e-by-N_{ne} matrix G specifies an element by its nodes with references to the rows in X. Here d is the spatial dimension, N_n is the total number of nodes, N_e is the number of elements and N_{ne} is the number of nodes per element. Two elements of space dimension d are neighbours if they have d nodes in common.

For implementation of the optimal ordering strategy for the partition of the computational domain Ω, the neighbouring relations between the elements are crucial. This information may be extracted from the matrixes X and G.

Papers A, B and C present a dG method for a regular Cartesian grid, while in Paper D, we extend the methodology to unstructured triangular grids. To simplify the integral evaluation using triangular grids, we use a linear mapping to unit triangles.

4.3.1 Cartesian Grid

For the case where the elements K are rectangles in a regular Cartesian grid, a convenient basis is the tensor product of Legendre polynomials $L_k(x,y) = l_r(x)l_s(y)$ for $r,s = 0,\ldots,n$. This basis functions are orthogonal to each other by nature and will simplifies the mass matrix structure. We assume for simplicity of presentation that $\Omega \subset \mathbb{R}^2$, and let $\mathbb{Q}^n = \text{span}\{x^p y^q : 0 \le p,q \le n\}$ be the space of polynomials of degree at most n in x and at most n in y, and let $V_h^{(n)} = \{\varphi : \varphi|_K \in \mathbb{Q}^n\}$. The approximate solution on an element K_i can then be written as

$$u_h(x,y) = \sum_{k=0}^{n^2} u_k^i L_k\Big(\frac{2(x-x^i)}{\Delta x^i}, \frac{2(y-y^i)}{\Delta y^i}\Big), \tag{4.20}$$

where (x^i, y^i) is the centre of element K^i. By applying this basis functions for a Cartesian grid, the degrees of freedom per element in a dG(n)-methods is $m = (n+1)^d$ in d spatial dimensions.

4.3.2 Triangular Grid

We now extend the strategy to unstructured triangular grids that adapt complicated geometries. We let $\mathbb{Q}^n = \text{span}\{x^p y^q : 0 \le p+q \le n\}$ be the space of polynomials of

degree at most n, and let $V_h^{(n)} = \{\varphi : \varphi|_K \in \mathbb{Q}^n\}$. Using this space of polynomials for triangular grid, the degrees of freedom per element in a dG(n)-methods is $m = \frac{(n+1)(n+2)}{2}$ in two spatial dimensions.

4.4 Linear Transformation

For general triangular grids, we evaluate the complicated integrals by transforming them into simpler ones using a linear transformation; see Figure 4.4. In the

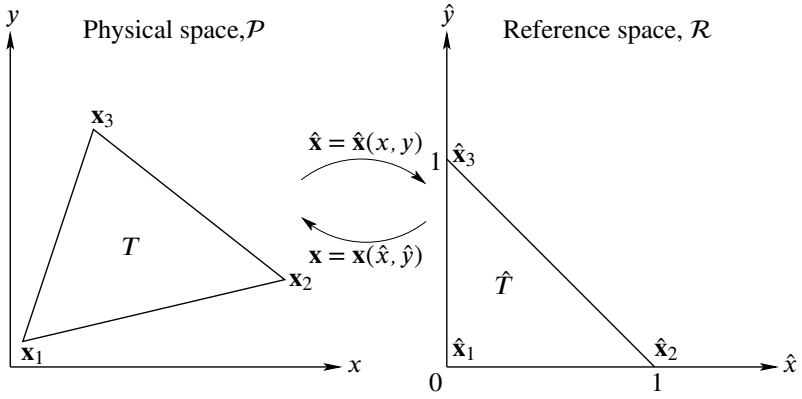


Figure 4.4: Triangles are mapped to a unit triangle through a linear transformation.

evaluation of the integrals, we use the change of variable formula and do a mapping to a unit triangle. In this reference space we find the approximation of the integrals, and transform it back to the physical space.

We denote the simulation grid containing the irregular triangle by the physical space $\mathcal{P}$ and the reference space by $\mathcal{R}$. The triangle in $\mathcal{P}$ corresponding to $\mathbf{x}_1, \mathbf{x}_2$, and $\mathbf{x}_3$ will be referred to as $T = T(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$, and the corresponding unit triangle in $\mathcal{R}$ will be $\hat{T}$. Variables in $\mathcal{R}$ will be denoted by a ' ˆ ', e.g., $\hat{T}$. The linear transformation from $\mathcal{R}$ to $\mathcal{P}$ is given by

$$\mathbf{x}(\hat{\mathbf{x}}) = \begin{bmatrix} x(\hat{x}, \hat{y}) \\ y(\hat{x}, \hat{y}) \end{bmatrix} = \mathbf{x}_1(1 - \hat{x} - \hat{y}) + \mathbf{x}_2\hat{x} + \mathbf{x}_3\hat{y} \tag{4.21}$$

$$= \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}(1 - \hat{x} - \hat{y}) + \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}\hat{x} \begin{bmatrix} x_3 \\ y_3 \end{bmatrix}\hat{y}, \tag{4.22}$$

where $\hat{\mathbf{x}} = (\hat{x}, \hat{y})$ are the local reference space coordinates such that $0 \leq \hat{x} \leq 1 - \hat{y}$ and $0 \leq \hat{y} \leq 1 - \hat{x}$.

4.4.1 Properties

The Jacobi matrix and the Jacobi determinant

The Jacobi matrix of the transformation (4.21) is constant and given by

$$J(\hat{\mathbf{x}}) = \frac{d\mathbf{x}}{d\hat{\mathbf{x}}} = \begin{bmatrix} \frac{\partial x}{\partial \hat{x}} & \frac{\partial x}{\partial \hat{y}} \\ \frac{\partial y}{\partial \hat{x}} & \frac{\partial y}{\partial \hat{y}} \end{bmatrix} = \begin{bmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{bmatrix}, \tag{4.23}$$

and the Jacobi determinant (or Jacobian) is

$$detJ(\hat{\mathbf{x}}) = \frac{\partial x}{\partial \hat{x}}\frac{\partial y}{\partial \hat{y}} - \frac{\partial x}{\partial \hat{y}}\frac{\partial y}{\partial \hat{x}} = (x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1). \tag{4.24}$$

The Jacobian is constant and equal to two times the area of the triangle

$$detJ = 2\{ \text{ Area of } K\}. \tag{4.25}$$

Provided $detJ \neq 0$, the inverse of the Jacobi matrix is given by

$$J^{-1}(\hat{\mathbf{x}}) = \frac{d\hat{\mathbf{x}}}{d\mathbf{x}} = \frac{1}{detJ}\begin{bmatrix} \frac{\partial y}{\partial \hat{y}} & -\frac{\partial x}{\partial \hat{y}} \\ -\frac{\partial y}{\partial \hat{x}} & \frac{\partial x}{\partial \hat{x}} \end{bmatrix}. \tag{4.26}$$

Change of variables in an integral

If $f(x, y)$ is integrable on a triangle T, and $f(x(\hat{x}, \hat{y}), y(\hat{x}, \hat{y}))$ is integrable on $\hat{T}$, the change of variables for double integrals is given by

$$\int_T f(x, y)dxdy = \int_{\hat{T}} f(x(\hat{x}, \hat{y}), y(\hat{x}, \hat{y}))detJ\, d\hat{x}d\hat{y}. \tag{4.27}$$

Transformation of a gradient

We will derive the transformation of a gradient. Let u be a function of $\mathbf{x}$ and define

$$\nabla u = \frac{du}{d\mathbf{x}} = \begin{bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{bmatrix} \tag{4.28}$$

and

$$\hat{\nabla} u = \frac{du}{d\hat{\mathbf{x}}} = \begin{bmatrix} \frac{\partial u}{\partial \hat{x}} \\ \frac{\partial u}{\partial \hat{y}} \end{bmatrix}. \tag{4.29}$$

From the chain rule, we have that

$$\frac{du}{d\mathbf{x}} = \frac{du}{d\hat{\mathbf{x}}}\frac{d\hat{\mathbf{x}}}{d\mathbf{x}} = \frac{du}{d\hat{\mathbf{x}}}J^{-1}. \tag{4.30}$$

By substituting (4.28) and (4.29) into (4.30), we obtain

$$\nabla u = J^{-T}\hat{\nabla}u. \tag{4.31}$$

Piola transformation

The Piola transformation is a vector transformation from $\mathcal{R}$ to $\mathcal{P}$ that conserves fluxes over any $\hat{x}$-curve or $\hat{y}$-curve. The transformation is given by

$$\mathbf{v}(\mathbf{x}(\hat{\mathbf{x}})) = \frac{1}{detJ}J\hat{\mathbf{v}}(\hat{\mathbf{x}}). \tag{4.32}$$

Here $\mathbf{v}(\mathbf{x}(\hat{\mathbf{x}}))$ is the vector in $\mathcal{P}$, $\hat{\mathbf{v}}(\hat{\mathbf{x}})$ is the vector in $\mathcal{R}$, $\mathbf{x}(\hat{\mathbf{x}})$ is the linear transformation (4.21). The Jacobi matrix J and the Jacobi determinant $detJ$ are denoted by (4.23) and (4.24), respectively.

Assuming that the transformation $\mathbf{x}(\hat{\mathbf{x}})$ is invertible, we can solve (4.32) for $\hat{\mathbf{v}}(\hat{\mathbf{x}})$. The inverse transformation is given by

$$\hat{\mathbf{v}}(\hat{\mathbf{x}}(\mathbf{x})) = detJ(\mathbf{x})J^{-1}(\mathbf{x})\mathbf{v}(\mathbf{x}). \tag{4.33}$$

By transforming the vector $\mathbf{v}$ to a vector $\hat{\mathbf{v}}$ in reference space using the Piola transformation, the corresponding flux in the two spaces will be equal. See [21, 1] for details about Piola transformation.

4.4.2 Variational Formulation in the Reference Space

The variational formulation for the time-of-flight equation 3.5 in reference space may be written

$$A_{\hat{K}}U = -R_{\hat{K}}U_{\hat{K}} + F^{+}_{\hat{K}}U_{\hat{K}} + F^{-}_{\hat{K}}U_{\mathcal{U}(\hat{\mathcal{K}})} = B_{\hat{K}}, \quad \forall\hat{K}. \tag{4.34}$$

To evaluate the integrals in the unit triangle in the reference space, we use the properties discussed in the previous subsection. The area integrals are obtained by the following operators and matrixes

$$(R_{\hat{K}})_{ij} = \int_{\hat{K}} (L^k_i(\hat{\mathbf{x}})\hat{\mathbf{v}}) \cdot \hat{\nabla}L^k_j(\hat{\mathbf{x}})d\hat{\mathbf{x}}, \tag{4.35}$$

$$(B_{\hat{K}})_j = \int_{\hat{K}} \phi L^k_j(\hat{\mathbf{x}})detJ\,d\hat{\mathbf{x}}, \tag{4.36}$$

where $\hat{\mathbf{v}} = detJJ^{-1}\mathbf{v}$ and $\mathbf{v}$ is the velocities interpolated in the interior of the elements; see Equation (4.39) in next section.

Further, we have to evaluate the boundary integrals. Since $\mathbf{v}\cdot\mathbf{n}$ is assumed constant over each element face, we can write $\mathbf{v}\cdot\mathbf{n} = \frac{f_l}{|\partial K_l|}$, where $|\partial K_l|$ is the length of edge number l for element K, and f_l is the flux over the particular edge. The flux is conserved in the transformation, and we can directly replace $\mathbf{v}\cdot\mathbf{n}$ with $\frac{f_l}{|\partial K_l|}$ in the boundary integrals. Each boundary integral can be computed as line integrals using the Jacobi matrix (4.23) and a parametrisation $\beta_l(t)$ for each edge of the unit integral. The resulting integral of the boundaries can now be written as unit line integrals for each edge

$$(F^+_{\hat{K}})_{ij} = \sum_{l=1}^{d+1} \frac{\max(f_l,0)}{|\partial K_l|} \|J\beta_l'(t)\| \int_0^1 L_i^k(\beta_l(t)) L_j^k(\beta_l(t))dt, \tag{4.37}$$

$$(F^-_{\hat{K}})_{j} = \sum_{l=1}^{d+1} \frac{\min(f_l,0)}{|\partial K_l|} \|J\beta_l'(t)\| \int_0^1 L_i^{k_{ext}}(\beta_l(t)) L_j^k(\beta_l(t))dt. \tag{4.38}$$

4.5 The Velocity Field

For applications in porous media, the velocity field $\mathbf{v}$ is typically obtained by solving a pressure equation of the form (2.11). In this work, we assume that $\mathbf{v}$ is known and given in such a way that the flux $\mathbf{v}\cdot\mathbf{n}$ is constant over each element face in a Cartesian or triangular grid. To obtain this, the velocity field $\mathbf{v}$ is computed using a standard low-order discretisation method for (2.11), 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.

Since the velocity field is given as a flux on each face in all elements we have to interpolate these fluxes in the interior of the elements for the dG schemes. To do this we use the lowest order Raviart-Thomas vector basis function, see [43, 1]. For triangular elements, we do the interpolation of the velocity field using

$$\mathbf{v}_K = \sum_{i=1}^{d+1} f_i \mathbf{v}_i|_K, \quad \forall K, \tag{4.39}$$

where d is the spatial dimension, $\mathbf{v}_i|_K$ denotes the vector basis function on element K, and f_i is the Raviart-Thomas coefficient. For two spatial dimenstions, the Raviart-Thomas vector basis functions for triangles are

$$\mathbf{v}_i|_K = \frac{1}{2|K|}\begin{bmatrix} x-x_i \\ y-y_i \end{bmatrix}, \quad i = 1,2,3, \forall K \tag{4.40}$$

where (x_i, y_i) are the nodes of the triangles and $|K|$ is the area of the triangle. Equations (4.39) and (4.40) equip us with a connection between numbering of nodes, faces, and vector basis functions. Node i is defined to be the node opposite of face i, or, in other words, node i is the node that is not a part of face i. See left figure in Figure 4.5 for the numbering of nodes and edges in a triangle. The right figure shows the numbering of nodes in a rectangular element.

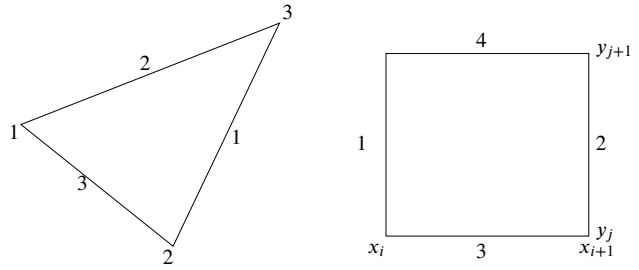


Figure 4.5: Numbering of edges and nodes in a triangle and the numbering of nodes in a rectangle.

In a similar manner, the interpolation function for Cartesian grid may be written

$$\mathbf{v}_K = \sum_{i=1}^{2d} f_i \mathbf{v}_i|_K, \quad \forall K. \tag{4.41}$$

The vector basis functions for $2D$ are defined as

$$\begin{aligned} \mathbf{v}_1|_K &= \frac{1}{|K|}\begin{bmatrix} x_{i+1}-x \\ 0 \end{bmatrix}, \quad \mathbf{v}_2|_K = \frac{1}{|K|}\begin{bmatrix} x-x_i \\ 0 \end{bmatrix}, \\ \mathbf{v}_3|_K &= \frac{1}{|K|}\begin{bmatrix} 0 \\ y_{j+1}-y \end{bmatrix}, \quad \mathbf{v}_4|_K = \frac{1}{|K|}\begin{bmatrix} 0 \\ y-y_j \end{bmatrix}, \end{aligned} \tag{4.42}$$

where x_i, x_{i+1}, y_j and y_{j+1} are the coordinates of the faces, as depicted in Figure 4.5.

The vector basis function $\mathbf{v}_i$ have the following property,

$$\mathbf{v}_i \cdot \mathbf{n}_j |\partial K_j| = \begin{cases} 0, & i \neq j, \\ 1, & i = j, \end{cases} \tag{4.43}$$

where $|\partial K_j|$ is face j. Hence $\mathbf{v} \cdot \mathbf{n}$ is constant on all faces, which is a prerequisite for the optimal ordering strategy.

4.6 Slope Limiting

For highly heterogeneous reservoirs with large variations in the porosity or strong shears in the velocity field, the time-of-flight will generally have low regular-

ity and exhibit very large variations. For instance, in regions where high-speed flow meets low-speed flow from nearly impermeable regions, the time-of-flight may oscillate with orders of magnitude over a few elements. This problem arises due to several types of reservoir heterogeneities: obstacles, shales, layers with large permeability ratios, fluvial reservoirs with high-permeable channels on a low-permeable background, large variations in pore volumes, etc. Use of higher-order polynomial approximations can easily result in spurious oscillations and non-physical time-of-flight values. Moreover, if these variations are not captured by the local approximation, (large) errors can propagate to neighbouring elements.

Spurious oscillations near discontinuities and kinks is a common problem in discontinous Galerkin methods using higher order basis functions. We can circumvent this problem by applying a (slope) limiter that reduces the local variation of each basis function by modifying the coefficients of higher order (one or more) polynomial terms.

Limiters are usually derived from the maximum principle or from a principle that limits the local variation. For the time-of-flight equation (3.5), the only principle available to us is the fact that $\tau(\mathbf{x})$ is strictly increasing along streamlines, which follows trivially from (3.2). We therefore propose to check that the time-of-flight is increasing from inflow to outflow in each element; that is,

$$\min \tau|_{\partial K^+} > (1-\epsilon) \max \tau|_{\partial K^-}, \quad 0 \leq \epsilon \ll 1. \tag{4.44}$$

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. 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. For rectangular elements, the number of new elements corresponds to the degrees-of-freedom in the original element to compensate for the reduced accuracy due to order reduction. Hence, for basis functions of order one, we split the element in two in each spatial direction, for basis functions of order two we split in three, etc. For triangular grids, a similar idea can be used.

To illustrate the difficulty of accurately resolving time-of-flights, we show the exact solution sampled in 2000×2000 evenly spaced points inside two grid cells taken from the fluvial Upper Ness formation in Layer 76 of the 10th SPE test case; see Figure 4.6, and Paper A for more details. 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.

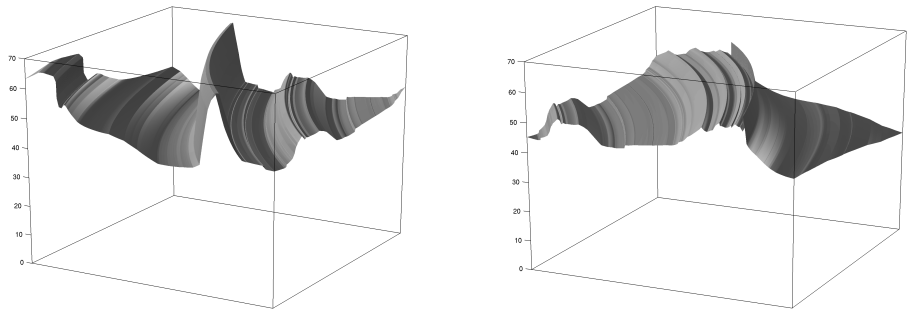


Figure 4.6: Time-of-flight in the two grid cells of Layer 76 in the 10th SPE test case sampled in 2000×2000 evenly distributed points inside each cell.

4.7 Multi-Phase Flow

An implicit discontinuous Galerkin scheme for purely advective multiphase flow in porous media in the absence of gravity and capillary forces is briefly presented in Paper B. We refer to Natvig and Lie [37, 36] for the complete presentation of implicit discontinuous Galerkin schemes for fast computation of multiphase flow in porous media.

An implicit discretisation of multiphase flow models is given by (3.11). To advance the solution a single time step, one must solve the discrete system of nonlinear equations. Assembling and solving a large nonlinear system is often very expensive, even for a simple first-order method, and using a higher-order spatial discretisation introduces extra couplings and increases the nonlinearity of the discretised equations. By applying a dG method in combination with an upwind flux approximation, the corresponding discrete system of nonlinear equations can be decoupled in a sequence of nonlinear problems that can be solved sequentially in one grid block at time using the standard Newton-Raphson algorithm. The use of an element-wise solutions procedure makes implicit higher-order schemes feasible. This scheme is robust and efficient which makes it possible to compute the solution in very large grids on a single desktop computer.

A multiphase model may produce discontinuities at interfaces between injected water and resident oil. In the presence of discontinuities or kinks in the solution, the higher order spatial discretisations tends to produce spurious oscillations, which can be suppressed by performing a post-processing with a nonlinear limiter function after each step. The gain in the nonlinear case is possibly even greater than in the linear case, because the Newton-Raphson iteration can be controlled seperately in each element; see [37, 36] for more details.

4.8 Streamline Tracing

As we compare grid-based solutions obtained by discontinuous Galerkin methods of varying order to highly resolved streamline solutions, we give a brief description of streamline tracing using Pollock's method.

Streamline-based methods may be applied to solve flow and transport problems in porous reservoirs; see e.g., [22, 29, 32, 33, 34]. Streamlines can visualise flow patterns and identify swept or drained areas for different wells. Currently, streamlines methods can be used on much larger geological models than finite-volume methods. The included papers demonstrate that similar capabilities are possible in a finite-element (or finite-volume) framework by using the assumptions of unidirectional flow.

To compute highly resolved reference solutions, a streamline tracing method due to Pollock [42] is used. Pollock's method builds a streamline as a series of (small) line segments that each cross a grid cell in physical space. The segments are constructed such that the exit point of the streamline in one cell is the entrance point in the next cell. Pollock's method uses an exact formula for streamlines through each element based upon a piecewise linear approximation of $\mathbf{v}$ in each direction. Given an entry point of a streamline into a grid cell, Pollock's method starts by mapping the grid cell onto the unit square (or unit cube in $3D$). Each component of the velocity field is then approximated in reference space by a linear function, and the streamline path in each direction is given as an exponential function of the travel time. To trace streamlines, Pollock's method determines the travel time through the grid block as the minimum time to exit in each spatial direction, which is given by a logarithmic expression. The travel time is then used to compute the exit point, and the exit point is mapped back into physical space to give the entry point into the next element, and so on. See [22, 42, 21] for a more detailed description of Pollock's method. The method is widely used in the petroleum industry to trace streamlines, even though it may become inaccurate for non-Cartesian grids, see [22].

4.9 Solution Procedure

Altogether, we have demonstrated that the dG schemes in combination with an optimal ordering of elements is a robust, accurate and efficient numerical approach for the solution of incompressible flow of fluids in porous media. Below, the solution procedure for our methodology is summerized. The velocity field is assumed given in such a way that the flux is constant over each elment face.

1. Find a representation of the geometry and *create a mesh*. The standard way

of representing a mesh is by the meshpoints (vertex nodes) and a partition of the computational domain.

2. For implementation of the optimal ordering strategy in Step 3, the neighbouring elements are crucial. This *relationships between elements* may be extracted from the information of node positions and the element indicies.

3. *Find a prior reordering* of elements such that the solution can be computed efficiently in an element-by-element fashion. To find the sequence we let the elements and fluxes together form a directed graph, where the elements are the vertices and the fluxes are the directed edges. The desired computational sequences can be obtained by a topological sort of the vertices in the graph.

   If the velocity field has a nonzero circulation, there are one or more *strongly connected components* in the graph, and these groups can be detected by one additional depth-first traversal of the graph. In such a group of elements, the solution must be computed simultaneously.

4. Rewrite the mathematical problem in a *variational form*.

5. *Seek solutions in a finite-dimensional subspace*. This space consists of piecewise smooth functions of order n that may be discontinuous over element interfaces.

6. The resulting linear system can be assembled and solved in an *element-by-element fashion*. The coefficients can be approximated by applying, e.g., *Gauss quadrature* of the integrals. The integrals are simplified by mapping them into a unit triangle using a *linear transformation*.

7. If spurious oscillations arise, we introduce a sort of (slope) *limiter* to stabilise the solution. Recompute the solution in the actual element by making a subdivision into a set of first-order elements.

Chapter 5

Applications in Groundwater Protection

In this section we will describe two applications related to groundwater protection using the dG methods introduced in the previous sections. They do not appear in the included papers, but are good examples that demonstrate the practical relevance of the dG methods; see [8].

A water-protection zone is fundamental in groundwater protection. Within this area, building houses is not permitted, nor have any industrial activities or aggressive agriculture that may contaminate the water.

In Germany, the water-protection zone is called a fifty-day line. This means that biological processes need at least fifty days to decompose and degrade the contaminants to an acceptable quantity of contamination. Hence, if a spill accident happens and contaminants infiltrate, it should take at least fifty days for the contaminants to travel from the spill point to the protected area. To build anything that can produce contaminations inside the fifty-day line is prohibited. To find this type of water-protection zone, we need a concept that allows us to calculate fifty-day time zones. Obviously, the time-of-flight calculations can give us directly the information that we need to find protection zones.

We present two cases where we calculate the protection zones around water supplies (e.g., a river, lake, etc.) using time-of-flight calculations. To compute time-of-flight, a third-order dG scheme is considered. From experience, we choose this order of the basis functions to obtain an accurate solution within a reasonable computation time (See Paper A). Of course, these results are strongly related to the heterogeneity of the permeability. For both cases, the viscosity μ in Equation (2.11) is constant equal to $1.14 \times 10^{-3} kg/ms$.

5.1 Case I

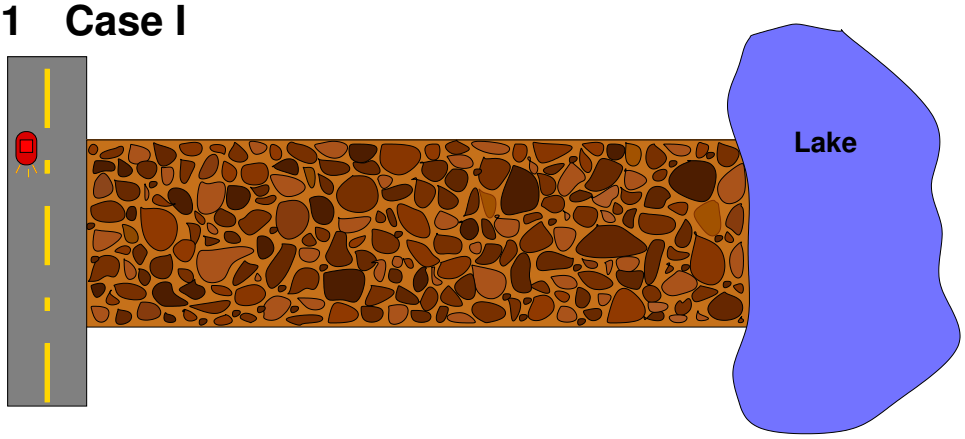


Figure 5.1: Sketch of case I

In the first example, we investigate a lake that needs to be protected from contamination. The lake is "connected" to a highway through the heterogeneous soil (computational domain); see Figure 5.1. We will investigate the following problem; Assume that an accident, which involves a contaminant spilling, happens on a certain position on the road. Which position will be most critical with respect to contamination of the lake? That means from which position on the highway will the contaminant use the shortest time to travel from the accident site on the highway to the lake. According to the calculation of the most dangerous position, restrictions can be set up on the road to avoid or reduce the risks of the fastest pollution case.

Boundary conditions for the computational domain Ω with dimensions 300 m $\times$ 75 m are given in Figure 5.2. We assume that the upper and lower boundaries are impermeable and defined as no-flow boundary conditions. Flow is generated by the difference of Dirichlet pressure boundary conditions between the accident site on the left boundary and the whole right boundary. As depicted in Figure 5.2, there is a no-flow boundary condition on the left side except for the injection point, where the spill accident takes place.

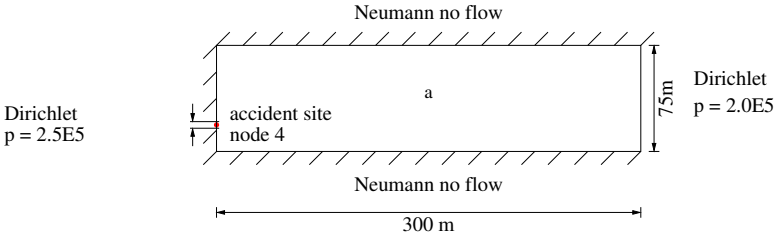


Figure 5.2: Boundary conditions for case I

The heterogeneous permeability field is obtained from synthetic data. However, the variation of the values in the permeability field is realistic. For our numerical simulation, a 64 × 16 rectangular grid is used. Hence, there are 15 nodes inside the left boundary, which are numbered in the sequence 1, 2, $\cdots$, 15 from bottom to top. In Figure 5.2, the injection point corresponds to node 4. To observe the effect of the heterogeneous porous medium on the results, we choose seven symmetric injection nodes 1, 4, 6, 8, 10, 12, 15 on the left boundary to calculate the time-of-flight. The time-of-flight results for the seven different accident sites are given in Figure 5.3.

The plots in Figure 5.3 are computed time-of-flights for the different injection points in the domain Ω. The first row is related to injection nodes 1 and 4, the next row 6 and 8 etc. It is easy to see that time-of-flights are non-symmetric for symmetric injection nodes due to the heterogeneity.

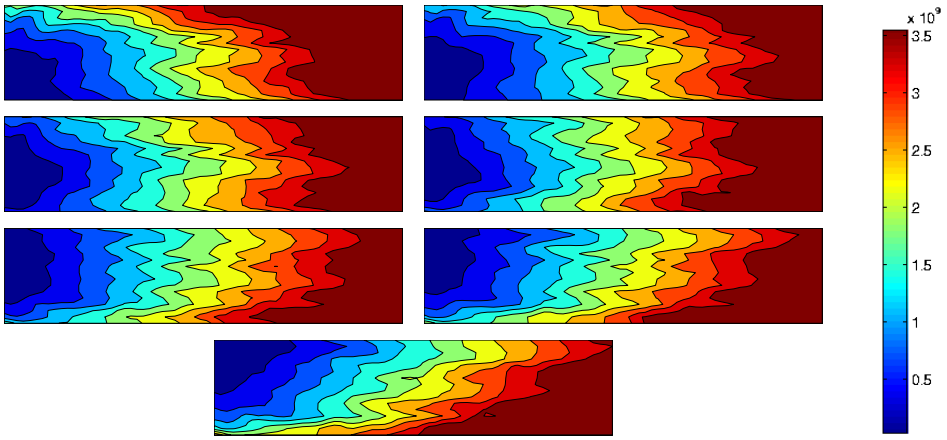


Figure 5.3: Time-of-flight [s] graphs for different injection points.

We will determine which position of the road that is the most critical regards pollution of the lake if an accident involving contamination spill should happen. Comparing the results in Figure 5.3, we can see that the breakthrough time at the lake is longest in the middle right figure, where we have injection in node 8. Here the exact breakthrough time is $4.4402 \times 10^6 s$ (≈ 51 days). The injection position for the shortest breakthrough time is node 15, (the lowest figure), where the breakthrough time is $3.5805 \times 10^6 s$ (≈ 41 days). If a fifty days water protection zone is establish around the lake, we have to set up restrictions on the road around the position associated with node 15.

5.2 Case II

In the second example, we consider a synthetic test case, where River 1 needs to be protected from the infiltration of contaminants from River 2, see Figure 5.4. The water table of River 2 changes with seasons and the weather, while the changes of the water table of River 1 are neglected. Hence, the pressure difference between the two rivers changes. In this case we will investigate the following scenario: if contaminants spill to River 2, how long will it take for the contaminants to infiltrate and arrive at River 1? The water-protection zone is a fifty-day line, and to guarantee this criteria we find the upper limit for the pressure gradient between the rivers. This means we must control the water table of River 2 such that the arrival time of potential contaminants to be longer than 50 days.

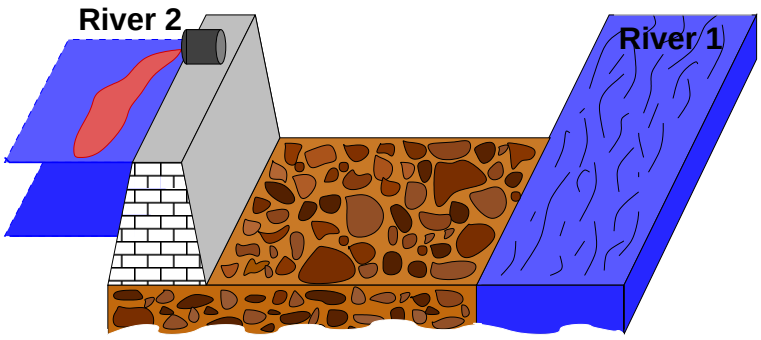


Figure 5.4: Sketch of case II

Assume that the porous medium between the rivers is heterogeneous in the horisontal direction and homogeneous in the vertical direction, so that the problem is reduced to a two-dimensional case.

Boundary conditions for the computational domain Ω with dimensions 100 m $\times$ 100 m are given in Figure 5.5. We assume that the upper and lower boundaries are impermeable and defined as no-flow boundary conditions. Flow is generated by the pressure difference between the left and right Dirichlet boundary conditions. To find the critical water table of River 2, different pressures on the left boundary are tried, e.g., p_1, p_2, p_3, p_4 in Figure 5.5. In this case, a 32 $\times$ 32 rectangular grid is used.

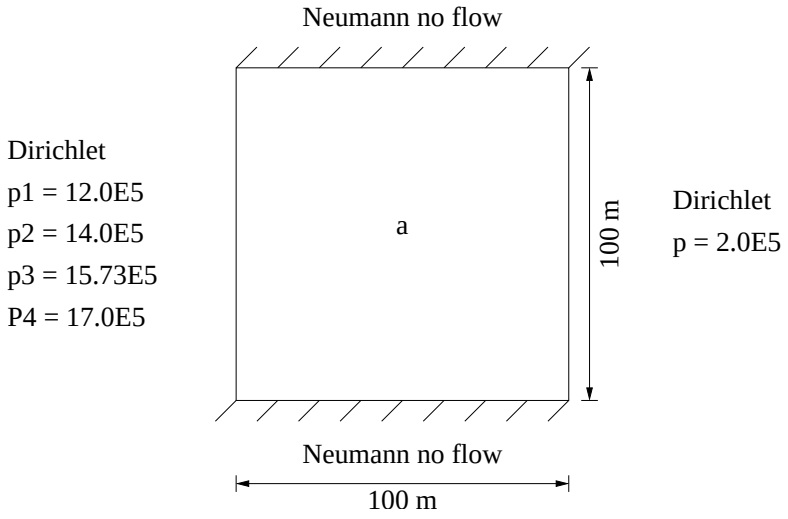


Figure 5.5: Boundary conditions for case II

Figure 5.6 shows the computed time-of-flights for different injection pressure heads in the domain Ω. To control the water table of River 2, we want to know how big the injection pressure head can be without contaminating River 1. To avoid polluting River 1, the time-of-flight of contaminants has to be longer than, or at least equal to 50 days ($= 4.32 \times 10^6 s$). By comparing the results in Figure 5.6, we see that for a pressure head of 17 bar the contaminants have arrived at River 1 in less than 50 days, which means that the pressure head in this case is too big (bigger than the allowed critical value). According to the results and by using manual bisection, the injection pressure head has to be smaller than, or at most equal to 15.73 bar, which corresponds to a certain water table such that the contaminants will use more than 50 days to travel from River 2 to River 1.

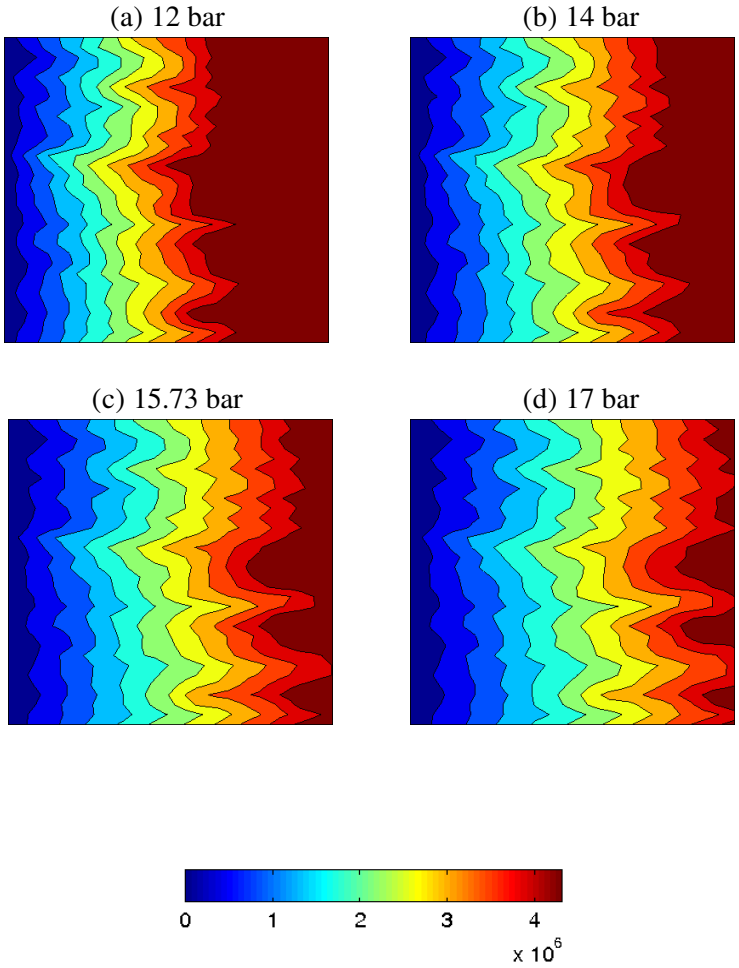


Figure 5.6: Time-of-flight [s] graphs for different injection pressure heads [bar]

Chapter 6

Summary of Papers

Summaries of the papers included in Part II of the thesis, will now be given. Details are also to be found in the previous sections.

6.1 Summary of Paper A

In Paper A, we focus on *advective* transport in a porous medium completely filled with fluids of a single phase and we assume that the fluid velocity $\mathbf{v}$ is a time-independent function. To this end, we consider a method for a class of linear equations on the form

$$\mathbf{v} \cdot \nabla u = H, \text{ in } \Omega, \qquad u = h, \text{ on } \partial\Omega^-, \tag{6.1}$$

where H and h are bounded functions, $\partial\Omega^-$ denotes the inflow boundary of the domain Ω and $\mathbf{v}$ is a given (divergence-free) vector field.

To accurately solve (6.1), we use a discontinuous Galerkin (dG) method in combination with an upwind flux approximation on a Cartesian grids. This dG discretisation will lead to a large system of linear equations that must be solved to obtain the approximate solution of (6.1). Since the characteristics of (6.1) are positive, all waves propagate in the same direction as $\mathbf{v}$ and the solution in each element K depends on the solution at the inflow boundary ∂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 the sign of the normal component of $\mathbf{v}$ at an element interface $K \cap E$ determines in what sequence the solution in element K and E must be computed. We can find a sequence of elements $p_1, p_2, ..., p_{N_e}$ such that element p_i appears after element p_j in the sequence if p_i depend on p_j. Then the solution of the boundary-value problem (6.1) can be computed element by element, from inflow to outflow boundaries.

In Paper A, we introduce a general procedure to obtain such sequences. To find the sequence we let the elements and fluxes together form a directed graph, where the elements are the vertices and the fluxes are the directed edges. The desired computational sequences can be obtained by a topological sort of the vertices in the graph. A topological sort can be computed in $\mathcal{O}(N_e)$ operations for N_e vertices using a depth-first traversal of the graph. This way, assembly of the full global system is avoided, reducing implementational complexity, computaional costs and memory requirements. The optimal sequence of elements is a simple way to obtain efficient linear solvers for large advective problems.

If the velocity field $\mathbf{v}$ has a nonzero circulation, there are one or more strongly connected components (cycles) in the graph. Strongly connected components are groups of elements that are interdependent, and for these groups we need to compute the solution simultaneously. Hence, a sequential procedure can still be achieved by replacing each strongly connected component with a single vertex in the graph.

The examples in Paper A demonstrate that the discontinuous Galerkin schemes in many cases can compute time-of-flight with an accuracy comparable to streamline approaches and superior to Berre et al's grid-based attempts for higher order polynomial approximations in [5]. For smooth media, higher-order versions of these schemes yield excellent results with resolution comparable to direct integration of streamlines. The optimal sequential solution procedure simplifies the implementation of higher-order schemes, and reduces the costs in terms of runtime and memory requirements.

For strongly heterogeneous cases, the results are quite inaccurate. The first-order discontinuous Galerkin scheme is, in fact, identical to the first-order five-point upwind scheme presented in [28]. This scheme gives results similar to the results obtained in [5]. Using higher-order polynomial approximations near high contrast features in the medium results in oscillations and unphysical time-of-flight values in these areas.

By applying a (slope) limiter in an adaptive scheme we have been able to improve the time-of-flight results for higher-order schemes. In elements where we detect oscillations in the approximation, we replace the higher-order approximation with a first-order approximation and refine the grid to compensate for the reduced accuracy associated with first-order elements. The number of subdivided elements corresponds to the degrees-of-freedom in the original element. Hence, for an n'th order method we replace a single element with a uniform $n \times n$ refinement. This approach yields pretty accurate approximations of time-of-flight and is only slightly more costly.

Another interesting observation in Paper A is that we efficiently can approximate swept and drained areas/volumes. By solving a boundary-value problem for stationary tracer transport of each injection well, i.e., (6.1) with $H = 0$ and $h = 1$

around a single injection well, we obtain the swept volumes. By reversing the velocity field, we can find the drained volumes. The higher-order discontinuous Galerkin scheme computes the stationary tracer transport problem with high accuracy for both smooth and strongly heterogeneous reservoirs. Moreover, low-order approximations seems to provide sufficient accuracy.

Comments: For fluvial and layered media, direct integration of streamlines gives a spatial resolution that is hard to match with other methods bases on grid points or cell volumes. One should therefore not expect grid-based methods to perform as well as back-tracked streamlines for all possible velocity fields (as was demonstrated in [5]). However, compared to streamline methods, our approach has a few advantages. First of all the discontinuous Galerkin scheme is conservative. Therefore, mass balance errors are not a problem. Secondly, whereas streamline methods need a method to distribute streamline such that the whole grid is covered, we only need to find a sequence of computations.

6.2 Summary of Paper B

In Paper B we summarise the development in Paper A for linear transport in porous media. In this paper we also briefly present an extension of the methodology to more general nonlinear equations of the form $\mathbf{v} \cdot \nabla f(u) = H(u, \mathbf{x})$; see Natvig and Lie [37, 36] for a throughout presentation of implicit discontinuous Galerkin schemes with an element-wise solver for multi-phase flow.

The non-linearity does not alter the dependency graph formed by elements and interfaces fluxes. By applying the same ideas as for linear cases, we can solve the system of nonlinear equations by solving a sequence of nonlinear subsystems. At each step in the sequence, the solution of a small system of nonlinear equations is computed using a Newton-Raphson method. Each such system involves the degrees-of-freedom of one element (or a few elements if the graph contains strongly connected components).

Additional to sharp contrast in the features in the media, a multi-phase model will generally produce discontinuities at interfaces between the different phases. Near discontinuities, the dG scheme will tend to produce spurious oscillations. This can be suppressed by performing a post-processing with a nonlinear limiter function after each step; see [37, 36, 35] for more details. This scheme makes it possible to compute the solution in very large grids on a single desktop computer. Furthermore, the use of an element-wise solution procedure makes implicit higher-order schemes feasible.

Comments: An observation from the examples done by Natvig and Lie in [37] is that the decreasing of numerical diffusion in the spatial discretisation from first to second order is very noticeable. Third or higher-order schemes do not

improve the solution as much because of the use of a limiter effectively reduces the order at the fronts. In addition, the overall accuracy of scheme also depends on the size of the time step.

Paper B was presented at the conference on Computational Methods in Water Resources in June 2006.

6.3 Summary of Paper C

The numerical results in papers A and B show that the discontinuous Galerkin method is efficient and accurate for computing transport in porous media. In Paper C, we investigate dG schemes in naturally fractured media.

Due to the high contrast and different length scales of the rock matrix and the fractures, naturally fractured media represent a challange for reservoir characterisation, modelling, and simulation of petroleum and groundwater reservoirs. The aim of Paper C is to investigate how the dG discretisation of the time-of-flight equation is able to handle the geometries and sharp variations in rock properties of naturally fractured reservoirs. For simplicity, we only consider models in two space dimensions consisting of regular Cartesian grid. The optimal reordering of unknowns based on prior information of the direction of flow is used in the same manner as in Papers A and B. The result is an efficient method, which can be a grid-based alternative to streamline methods.

Since fractures exist on a much smaller spatial scale than the characteristic length scale of the matrix, we assume that we have a discrete fracture model, where the fractures are modelled as one-dimensional in a two-dimensional reservoir model. However, for the numerical calculations, we model the fractures as two-dimensional objects and approximate the solution in fracture elements in the same manner as for the matrix elements. This can be motivated based on the fact that the transport changes rapidly in the fractured regions.

Due to the complex geometries and potentially large variations in parameter values, fractures will often have a significant impact on the flow characteristics of a porous medium, and in this paper we investigate two simplified grid models to examine various approaches for the dG discretisation in fractured regions of the porous medium. Since it is assumed that the width of the fractures is negligible compared to characteristic length scales of the reservoir, we first consider a modification of the discretisation for the fracture elements, by assuming that there is no variation in time-of-flight across the fracture. The fractures are resolved with one element in the latitudinal direction of the fractures, thus reflecting the fact that the fractures initially are modelled as one-dimensional. The numerical results in Paper C demonstrate that this scheme does not give accurate approximations. Motivated by the fact that the time-of-flight has large variations across the fractures,

it is natural to consider a finer grid resolution in the latitudinal direction of the fractures (as opposed to only one element). Comparing the numerical results of the dG approximation with those from a streamline simulator, we reveal the importance of a sufficient grid resoultion in the latitudinal direction of the fractures, even though the widths of the fractures are very small compared to typical length scales of the unfractured parts of the reservoir.

The grid-resolution is necessary since the time-of-flight is an integrated quantity that exhibits fine-scale details and contains large spatial variation within and close to fractures (even though these may have been modelled as lower-dimensional objects in the original model). To avoid instabilities in the solution, we consider order reduction of the dG approximation in the latitudinal direction of the fractures. This also results in more efficient scheme due to a reduced number of unknowns for fracture elements.

Paper C was presented at the conference on Computational Methods in Water Resources in June 2006. Extension of the methodology to unstructured triangular grids is done in Paper D.

6.4 Summary of Paper D

The research presented in Paper D are a continuation of the ideas from C. In Paper C, we only considered Cartesian grids and this restricted the orientation of the fractures to be horizontal or vertical. Realistic fractured media lead to complicated domains which demand the use of adapted grids, and in Paper D we extend the methodology used in Paper C to unstructured triangular grids in 2D. The extension to tetrahedral elements in 3D is straightforward. A dG approximation on unstructured grids using lower order basis functions is presented by Røe in [47].

We consider single-phase flow in semi-realistic models of fractured reservoirs, where the fractures themselves are represented explicitly as volumetric cells with small width and high permeability. Explicit modelling of complex fracture networks will give rise to very complex structures, and using unstructured triangular (tetrahedral) grids, at least locally, may be necessary to accurately model realistic cases.

Numerical examples in Paper D for unfractured and fractured media illustrate the efficiency and robustness of the proposed numerical model. For triangular grids, the dG method is convergent for smooth solutions, but loses accuracy near discontinuities. The numerical results show how the nature of random perturbated grids impact the accuracy, leading to reduced convergence rates for rough grids.

Overall, the dG approximations give solutions that are qualitatively good—the schemes predict the actual flow patterns even for coarse grid resolutions. Some of the examples indicate that increasing the order of the basis functions is more

important than increasing the grid resolution (provided the flux is resolved with sufficient accuracy). 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. However, strongly discontinuities in the reservoir may give oscillations and reduce the accuracy of the solution. To avoid this we may introduce a limiter as presented in Paper A, where we reduce the order of the basis functions and refine the grid in areas with high media contrasts. In Paper D, we confirm the observation in Paper C about the importance of having a sufficient grid resolution in the latitudinal direction of the fractures to accurately compute time-of-flight in fractured porous media. This is necessary to capture the fine-scale features of the solution and reduce local discretisation errors that would otherwise tend to destroy the solution downstream of the fracture.

The framework is used to compute the stationary tracer equation in order to find an approximations to the well connectivity in a reservoir. The numerical experiments show that low-order approximations provide sufficient accuracy to produce a reasonable delineation of the reservoir volume.

Bibliography

[1] I. Aavatsmark. Bevarelsesmetoder for elliptiske differensialligninger. Lecture notes, University of Bergen, 2007.

[2] J. H. Abou-Kassem, S. M. Farouq Ali, and M. R. Islam. *Petroleum reservoir simulation - A Basic Approach*. Gulf Publishing Company, 2006.

[3] K. Aziz and A. Settari. *Petroleum Reservoir Simulation*. Elsevier Applied Science Publishers, London, 1979.

[4] P. Bastian and B. Rivière. Discontinuous galerkin methods for two-phase flow in porous media. Technical Report 2004-28, IWR (SFB 359), University of Heidelberg, 2004.

[5] 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.

[6] D. Braess. *Finite elements*. Cambridge, 2002. Second Edition.

[7] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*. Springer, 2002.

[8] Y. Cao, B. Eikemo, and R. Helmig. Fractional flow formulation for two-phase flow in porous media. Report, Universität Stuttgart, 2007.

[9] G. Chavent and J. Jaffre. *Mathematical Models and Finite Elements for Reservoir Simulation*. Elsevier, Amsterdam, 1978.

[10] B. Cockburn. An introduction to the discontinuous galerkin method for convection-dominated problems. In *in: B. Cockburn, C. Johnson, C.-W. Shu, E. Tadmor (Eds.), Advanced Numerical Approximation of Nonlinear Hyperbolic Equations, in: A. Quarteroni (Ed.)*, volume 1697, pages 151–268. Springer, New York, 1998.

[11] B. Cockburn. Discontinuous galerkin methods for convection dominated problems. In T. Barth and H. Reconink, editors, *High-Order Methods for Computational Physics*, volume 9 of *Lecture Notes in Computational Science and Engineering*, pages 69–224. Springer Verlag, 1999.

[12] B. Cockburn, G. E. Karniadakis, and C.-W. Shu. *Discontinuous Galerkin Methods: Theory, Computation and Applications*. Springer, December 1999. Lecture Notes in Computational Science and Engineering.

[13] B Cockburn and S.-W. Shu. Tvb runge-kutta local projection discontinuous galerkin finite element method for scalar conservation laws: General framework. *Math. of Comp.*, 52(186):411–435, 1989.

[14] B Cockburn and S.-W. Shu. The runge-kutta local projection p-discontinuous galerkin method for scalar conservation laws. 25:337–361, 1991.

[15] National Research Council, editor. *Rock fractures and fluid flow*. National Academy Press, 1996.

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

[17] P. Dietrich, R. Helmig, M. Sauter, H. Hötzl, J. Köngeter, and G. Teutsch. *Flow and Transport in Fractured Porous Media*. Springer, 2005.

[18] J. Douglas and T. Arbogast. *Dual Porosity Models for Flow in Naturally Fractured Reservoirs*, chapter VII. Dynamics of fluids in hierarchical porous media.

[19] J. Douglas Jr., M. Kischinhevsky, P. J. Paes-Leme, and A. M. Spagnuolo. A multiple-porosity model for a single-phase flow through naturally-fractured porous media. *Comp. Appl. Math.*, 17:19–48, 1998.

[20] 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.

[21] H. Hægland. Streamline tracing on irregular grids. Master's thesis, Department of Mathematics, University of Bergen, Cand.Scient Thesis in Computational Science, December 2003.

[22] 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.

[23] B.-O. Heimsund. *Mathematical and Numerical Methods for Reservoir Fluid Flow Simulation*. PhD thesis, Dept. of Mathematics, University of Bergen, March 2005.

[24] R. Hinkelmann. *Efficient Numerical Methods and Information-Processing Techniques for Modeling Hydro- and Environmental Systems*, volume 21. Springer, 2005.

[25] H. Hoteit and A. Firoozabadi. Multicomponent fluid flow by discontinuous galerkin method and mixed methods in unfractured and fractured media. *Water Resour. Res.,*, 41, 2005.

[26] H. Hoteit and A. Firoozabadi. An efficient numerical model for incompressible two-phase flow in fractured media. *Advances in Water Resources*, 31:891–905, 2008.

[27] C. Johnson. *Numerical solution of partial differential equations by the finite element method*. Cambridge, 1987.

[28] K. H. Karlsen, K.-A. Lie, and N. H. Risebro. A fast marching method for reservoir simulation. *Comput. Geosci.*, 4(2):185–206, 2000.

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

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

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

[32] S. F. Matringe and M. G. Gerritsen. On accurate tracing of streamlines. In *Proceedings of the SPE Annual Technical Conference and Exhibition*, Houston, Texas, 26-29 September 2004.

[33] S. F. Matringe, R. Juanes, and H. A. Tchelepi. Streamline tracing on general triangular and quadrilateral grids. In *Proceedings of the SPE Annual Technical Conference and Exhibition*, Dallas, Texas, 2005.

[34] S. F. Matringe, R. Juanes, and H. A. Tchelepi. Robust streamline tracing for the simulation of porous media flow on general triangular and quadrilateral grids. *J. Comp. Phys.*, 219(2):992–1012, December 2006.

[35] J. R. Natvig. *High-Resolution Methods for Conservation Laws in the Geosciences*. PhD thesis, Faculty of Mathematics and Natural Sciences, University of Oslo, 2006.

[36] J. R. Natvig and K.-A. Lie. On efficient implicit upwind scheme. In *Proceedings of ECMOR XI*, Bergen, Norway, 8-11 September 2008. `http://folk.uio.no/kalie/papers/ecmorxi-ro.pdf`.

[37] 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. *J. Comput. Phys.*, to appear. `http://folk.uio.no/kalie/papers/dg-multiphase.pdf`.

[38] L. Neunhäuserer, A. Hemminger, and R. Helmig. Influence of fracture - matrix - interaction on flow and transport processes and the resulting effective parameters in fractured porous systems. In *28. IAHR Congress: Hydraulic Engineering for Sustainable Water Resources Management at the Turn of the Millennium*, Graz, Austria, 22. - 27. August 1999.

[39] E. Øian. *Modeling of Flow in Faulted and Fractured Media*. PhD thesis, Dept. of Mathematics, University of Bergen, March 2004.

[40] P.-O. Persson and G. Strang. A simple mesh generator in matlab. *SIAM Review*, 46:329–345, 2004.

[41] Ø. Pettersen. Grunnkurs i reservoarmekanikk., 1990. Lecture notes, University of Bergen.

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

[43] P. A. Raviart and J. M. Thomas. A mixed finite element method for 2nd order elliptic problems. In *Lecture Notes in Mathematics 606*, pages 292–315. 1977.

[44] W. H. Reed and T. R. Hill. Triangular mesh methods for the neutron transport equation. Technical report, Los Alamos Sci. Lab., 1973.

[45] V. Reichenberger, R. Helmig, H. Jakobs, P. Bastian, and J. Niessner. Complex gas-water processes in discrete fracture-matrix systems: Upscaling, mass-conservative discretization and efficient multilevel solution. Institut für Wasserbau, Universität Stuttgart, 2004.

[46] B. Rivière, M.F. Wheeler, and K. Banas. Part ii. discontinuous galerkin method applied to a single phase flow in porous media. *Comp. Geosci.*, 4:337–349, 2000.

[47] O.I. Røe. Discontinuous galerkin methods with optimal ordering for fast reservoir simulation on general grids. Master's thesis, Norwegian University of Science and Technology, 2006.

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

[49] S. P. Van, L. Stadler, and R. Hinkelmann. Comparison of a micro-scale and a meso-scale model concept for two-phase flow in fractured-porous media. In Philip J. Binning, Peter K. Engesgaard, Helge K. Dahle, George F. Pinder, and William G. Gray, editors, *Proceedings of the XVI International Conference on Computational Methods in Water Resources*, Copenhagen, Denmark, June 2006. `http://proceedings.cmwr-xvi.org/`.

[50] J. E. Warren and P. J. Root. The behavior of naturally fractured reservoirs. *Society of Petroleum Engineers Journal*, 3:245–255, September 1963.

[51] O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu. *The Finite Element Methods: Its Basis and Fundamentals*. Elsevier, 2005. Sixth Edition.

Part II

Published and Submitted Papers

Paper A

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

* *Advances in Water Resources*, 2007, Volume 30, Issue 12, pages 2424-2438

Available online at www.sciencedirect.com
ScienceDirect
Advances in Water Resources 30 (2007) 2424–2438
Advances in Water Resources
www.elsevier.com/locate/advwatres

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, Department of Mathematics, Johs. Brunsgt. 12, NO-5008 Bergen, Norway

Received 19 February 2007; received in revised form 16 May 2007; accepted 16 May 2007
Available online 15 June 2007

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.

Keywords: Transport in porous media; Time-of-flight; Discontinuous Galerkin discretization; Linear solvers; Directed acyclic graphs

1. Introduction

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

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

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 of 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 non-smooth solutions. Moreover, Cockburn and Shu [2,3] analysed

* Corresponding author. Tel.: +47 22067323.
E-mail addresses: Jostein.R.Natvig@sintef.no, jrn@sintef.no (J.R. Natvig), Knut-Andreas.Lie@sintef.no (K. Lie), birgitte@mi.uib.no (B. Eikemo), ingab@mi.uib.no (I. Berre).

0309-1708/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.advwatres.2007.05.015

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 nonlinear equations. 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 nonlinear solver based on a reordering of the unknowns giving an upper block-triangular system. This way, the computational effort needed to solve (1) is reduced from solving a large sparse nonlinear system involving all degrees-of-freedom in the domain, to solving a sequence of smaller problems involving one or a few nonlinear 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 non-smooth 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 distribution 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

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

where ϕ is the porosity of the medium; $\rho_i, s_i, \mathbf{v}_i$, and q_i are the density, saturation (volume fraction), phase velocity, and volumetric source term of the ith 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 individual 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. Eq. (2) then simplifies to

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

We remark that gravity can be included in 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 transport 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 ($\xi = \text{const}$, $\eta = \text{const}$) 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 appropriate 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

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

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 Eq. (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

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

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

$$a_K(u_h,v_h) = -\int_K (u_h\mathbf{v})\cdot\nabla v_h\,\mathrm{d}x + \int_{\partial K}\hat{f}(u_h,u_h^{\mathrm{ext}},\mathbf{v}\cdot\mathbf{n})v_h\,\mathrm{d}s,$$
$$b_K(u_h,v_h) = \int_K H(u_h,\mathbf{x})v_h\,\mathrm{d}x,$$

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

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

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 $\mathbb{Q}^n = \mathrm{span}\{x^p y^q : 0\leqslant p,q\leqslant n\}$ 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

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

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 non-smooth 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 Eq. (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

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

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

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

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, $\mathscr{U}(K)$ 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 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 $\mathscr{O}(N)$ 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 (Section 4.1). 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 Fig. 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 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].

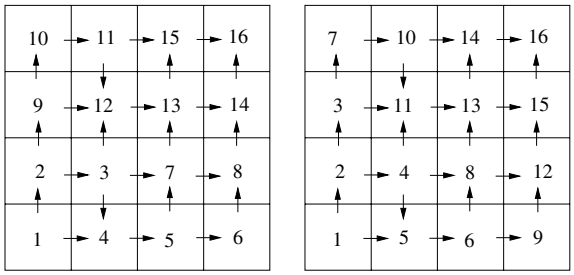


Fig. 1. (Left) Direction of flow and the order of computation generated by a depth-first traversal of the reversed flow field. (Right) The sequence that could have been computed from an advancing-front algorithm.

4.1. 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 $K_1 \in \mathscr{U}(K_2)$ and $K_2 \in \mathscr{U}(K_1)$ (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 Eq. (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 Fig. 2. The figure shows a velocity field computed using a mixed finite-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 Fig. 2 is a worst-case scenario. A more likely distribution of fluxes is depicted in Fig. 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 inter-

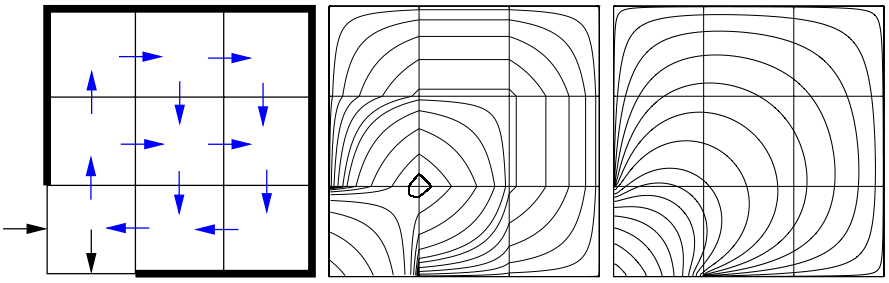


Fig. 2. (Left) A homogeneous domain with inflow and outflow in a single element and no-flow boundary elsewhere. The arrows indicate the sign of the flux on the faces of a 3×3 grid. (Middle) Streamlines of a mixed finite-element solution with 3×3 elements. (Right) Streamlines for a solution computed using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

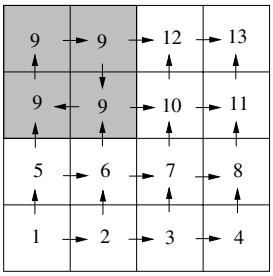


Fig. 3. The figure shows a suitable sequence of computations when a group of connected elements is encountered.

connected 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 Eq. (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

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

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 Eq. (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 $C_{i,\mathscr{U}(K)}$ in the neighbouring upwind elements. Then, (15) may be written as

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

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 Eq. (15), we are effectively computing a *weighted colouring* of the graph.

5.1. 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 assess the performance of the method, we will use geological data from Model 2 of the 10th SPE Comparative Solution Project [4]. The model contains $60\times220\times85$ 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 Fig. 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.

Fig. 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. Fig. 5 shows swept volumes computed for the 15 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. 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

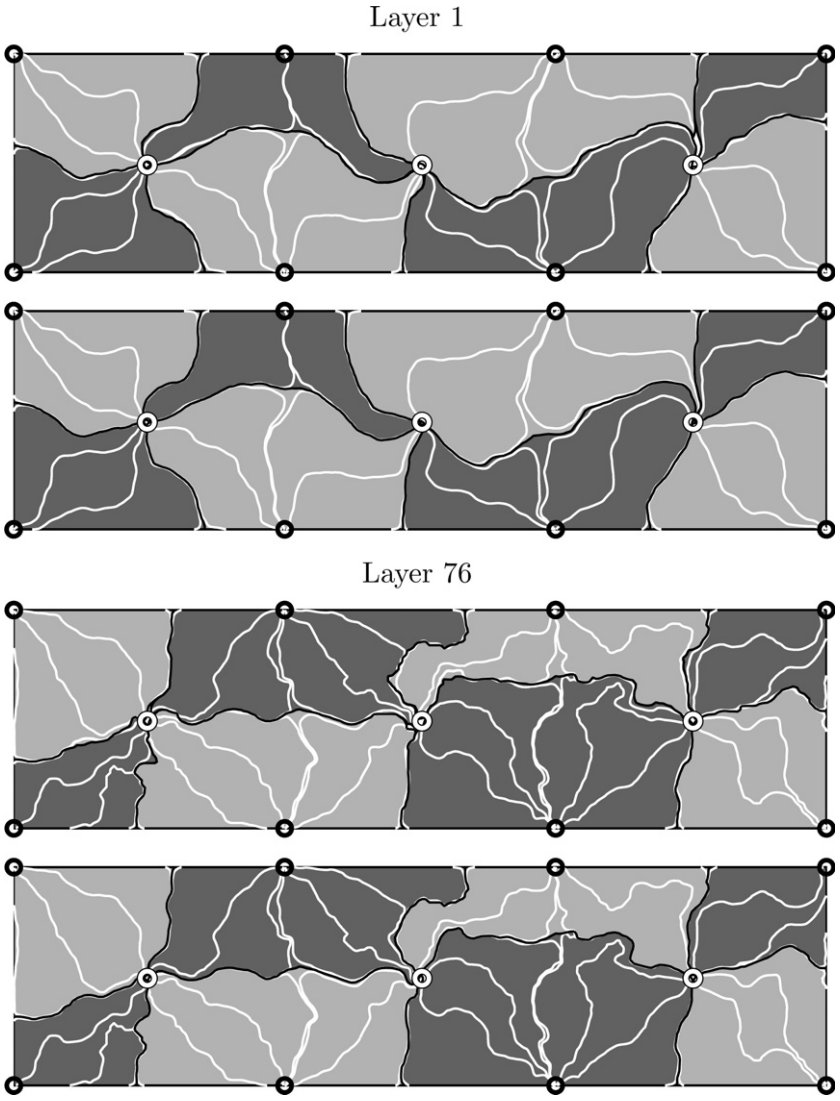


Fig. 4. The plots show the tracer distribution for two layers of the SPE 10 test case. The solution is computed using the dG(0) (upper) and dG(1) (lower).

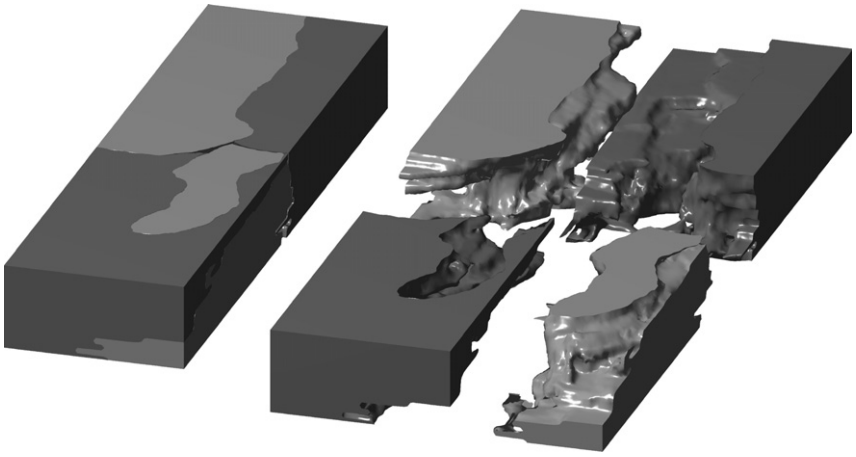


Fig. 5. Tracer distribution for a subsample from the smooth Tarbert formation in the SPE 10 test case. The velocity is computed using a two-point flux approximation.

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

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

Runtimes are measured on a single core on an AMD Athlon X2 4400+ processor.

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 less than 10 s and a third-order approximation in less than 1.5 min. (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 Eq. (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 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

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

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

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

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

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 $[1, 2] \times [1, 2]$.

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.

Fig. 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 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 Fig. 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.

Fig. 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.

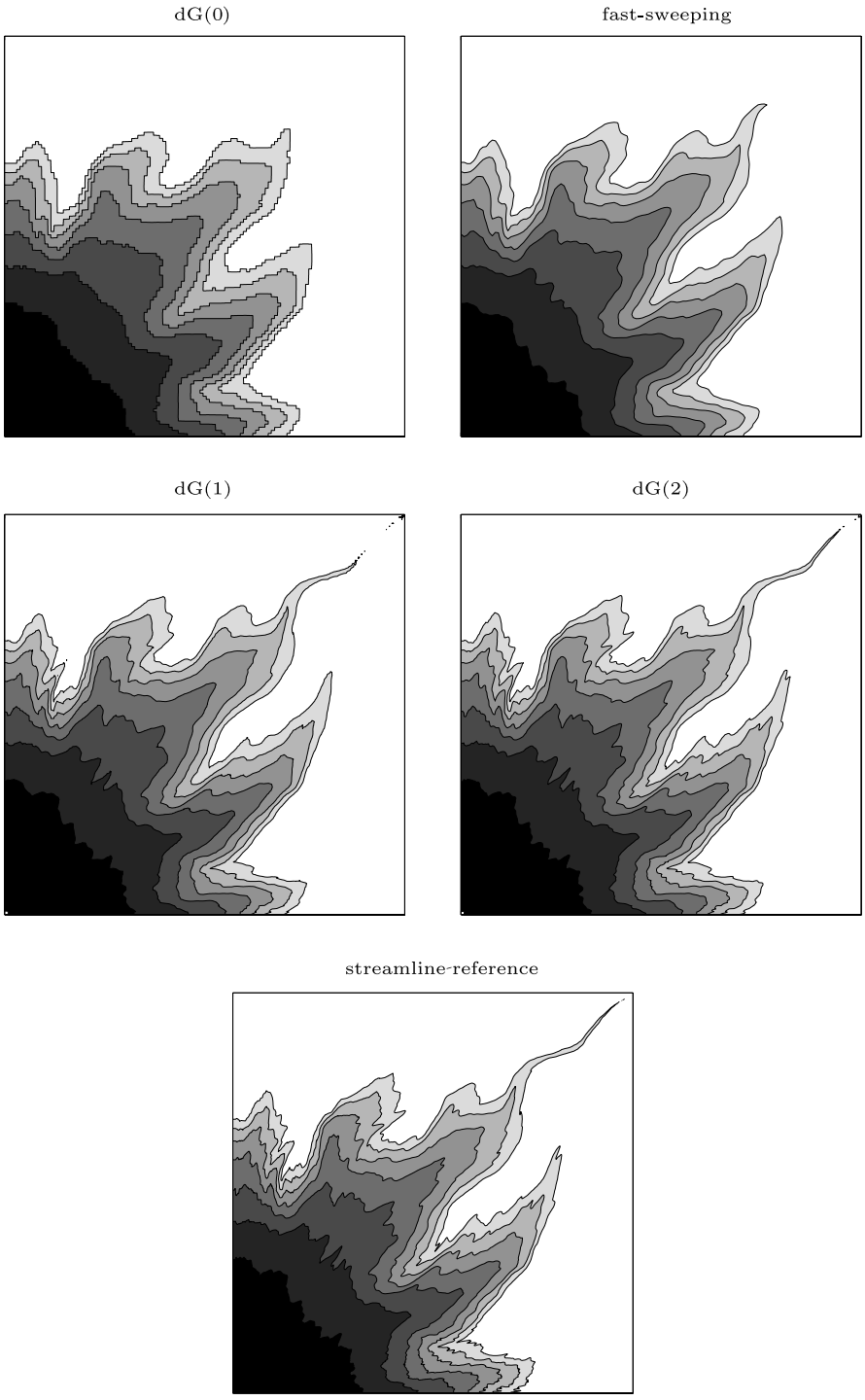


Fig. 6. Time-of-flights for Case 2 computed using dG(n) for $n = 0, 1, 2$, the fast-sweeping method, and direct streamline integration. The contours in the plots are $T = 0.07, \ldots, 0.49$ PVI in steps of 0.07.

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 magnitude.

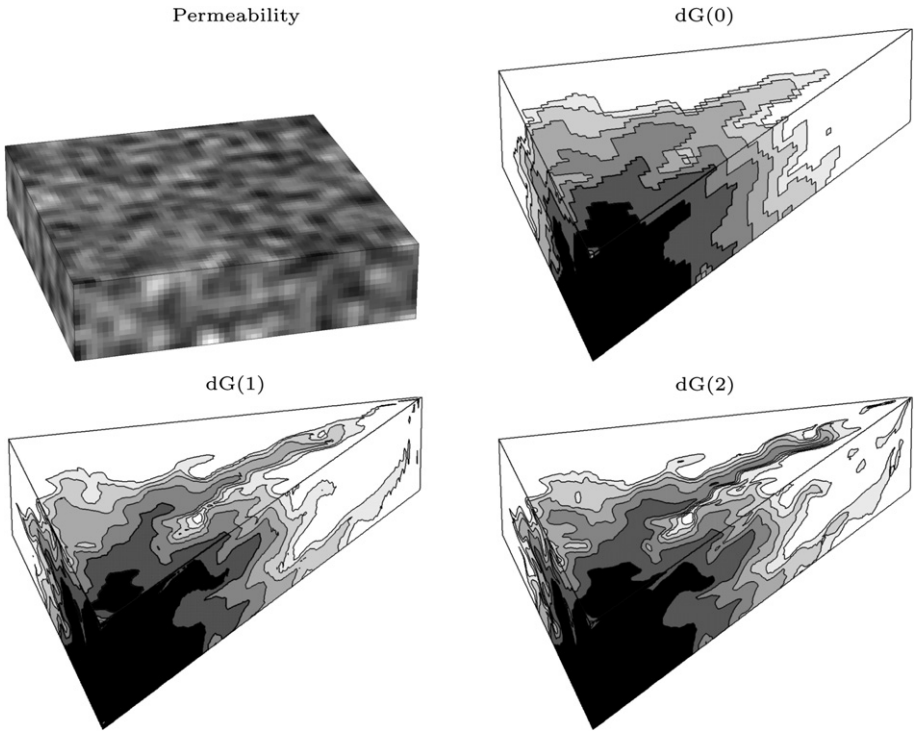


Fig. 7. Lognormal permeability field for Case 3 and corresponding time-of-flights computed using dG(n) for $n = 0, 1, 2$. The contours shown in the slice plots are at $T = 0.1, \ldots, 0.6$ PVI in steps of 0.1.

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.

6.1. 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 Eq. (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,

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

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 Fig. 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)-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 generally not computationally feasible. Instead, modern streamline methods [11] rely on tracing a set of *representative* streamlines launched from 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. Fig. 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

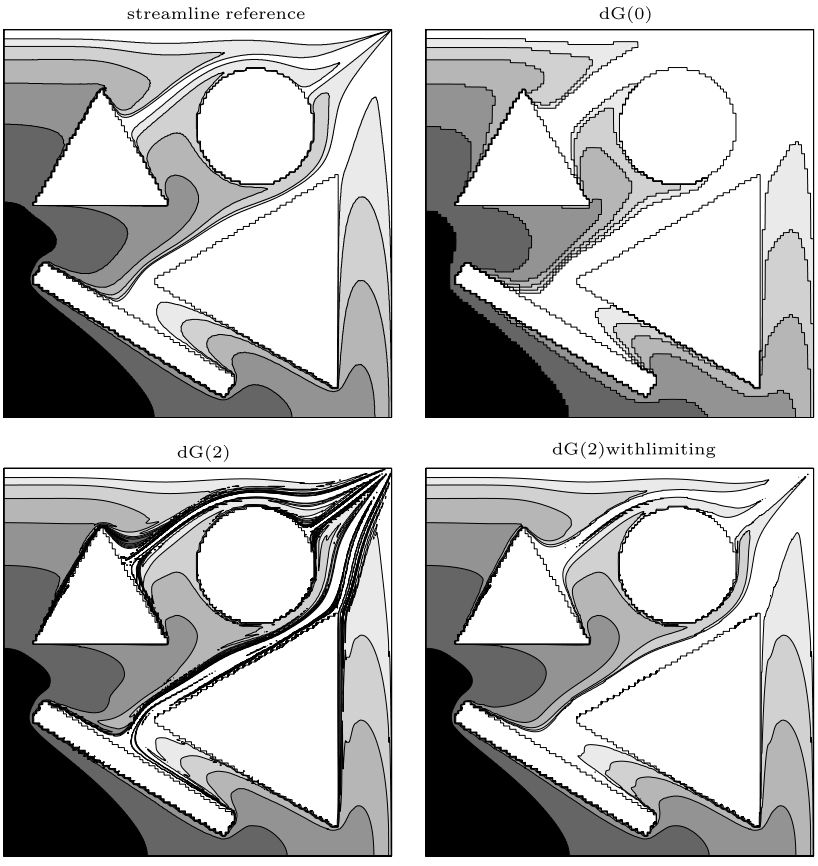


Fig. 8. Transport past obstacles for Case 4 computed by dG(0) and dG(2) with and without order-reduction/subdivision.

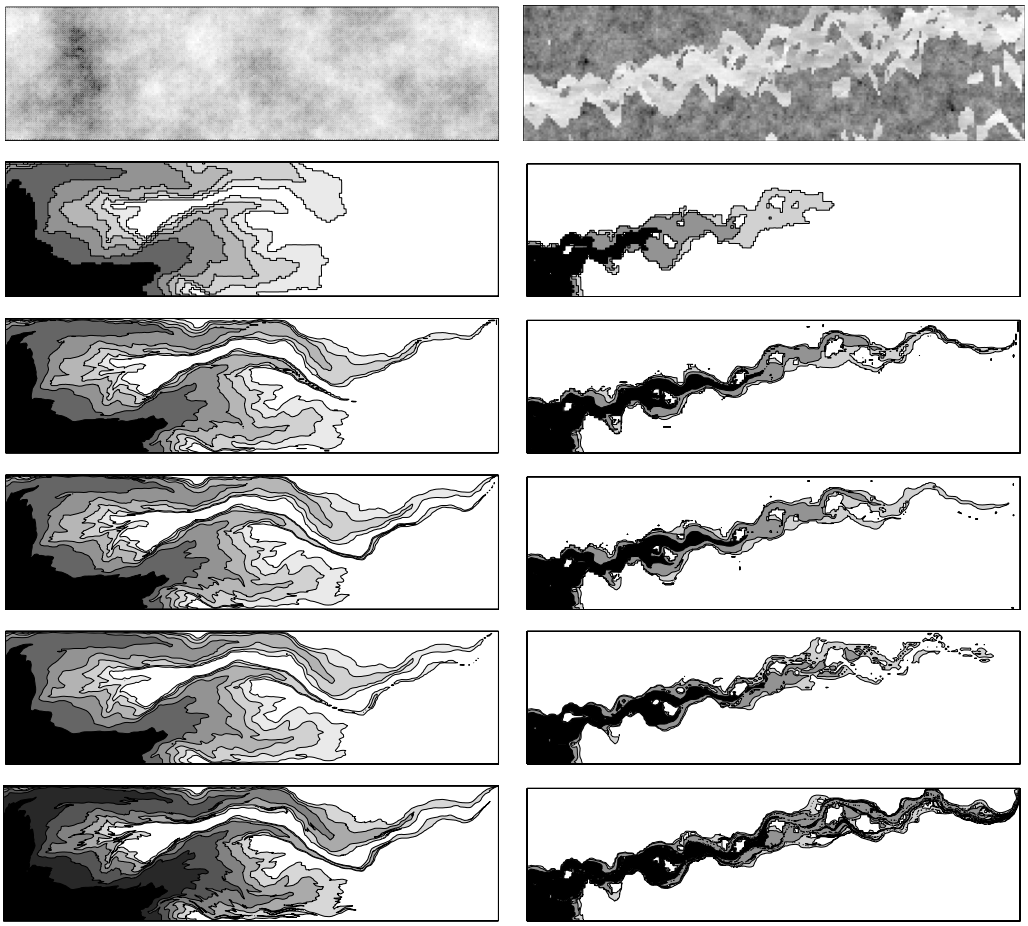


Fig. 9. Permeability field and time-of-flight for Layers 1 (left) and 76 (right) of the SPE 10 test case computed with dG(n) for $n = 0, 1, 2$ (order increasing downwards), streamline solution with 1500 streamlines, and a streamline reference solution (bottom). The contours shown in the plots are at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

fingering. The accuracy of the standard streamline method is somewhere between that of dG(1) and dG(2).

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. Fig. 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, Fig. 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

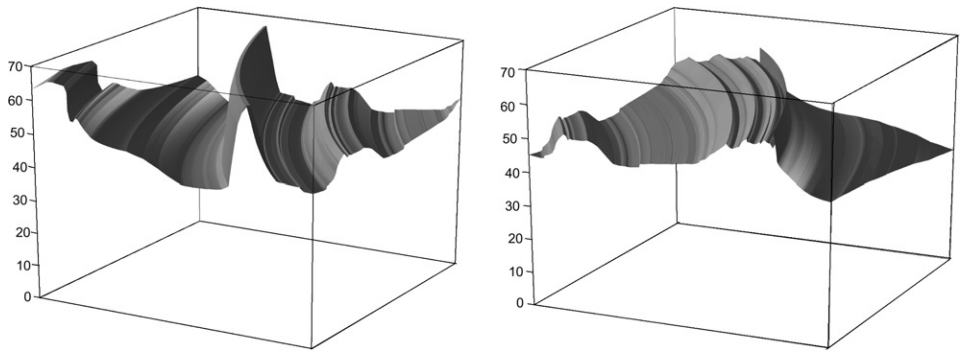


Fig. 10. Time-of-flight in the two grid cells (200, 36) and (200, 37) of Layer 76 in the 10th SPE test case sampled in 2000×2000 evenly distributed points inside each cell.

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 Fig. 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 transport 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.

Acknowledgements

The research was funded in part by the Research Council of Norway under Grant Nos. 139144/431 and 158908/I30.

References

[1] Berre I, Karlsen KH, Lie K–A, Natvig JR. Fast computation of arrival times in heterogeneous media. Comput Geosci 2005;9(4):179–201.
[2] Cockburn B, Shu S–W. TVB Runge–Kutta local projection discontinuous Galerkin finite element method for scalar conservation laws: general framework. Math Comp 1989;52:411–35.
[3] Cockburn B, Shu S-W. The Runge–Kutta local projection P^1-discontinuous Galerkin method for scalar conservation laws. M^2AN 1991;25:337–61.
[4] Christie MA, Blunt MJ. Tenth SPE comparative solution project: a comparison of upscaling techniques. SPE Reservoir Eval Eng 2001;4(4):308–17. http://www.spe.org/csp.
[5] Datta–Gupta A, King MJ. A semianalytic approach to tracer flow modeling in heterogeneous permeable media. Adv Water Resour 1995;18(1).
[6] Dennis Jr JE, Martínez JM, Zhang X. Triangular decomposition methods for solving reducible nonlinear systems of equations. SIAM J Opt 1994;4(2):358–82.
[7] Duff IS, Reid JK. An implementation of Tarjans algorithm for block triangularization of a matrix. ACM Trans Math Softw 1978;4(2):137–47.
[8] Eikemo B, Berre I, Dahle HK, Lie K-A, Natvig JR. A discontinuous Galerkin method for computing time-of-flight in discrete-fracture models. In: Binning PJ, et al., (editors). Proceedings of the XVI international conference on computational methods in water resources, Copenhagen, Denmark, June, 2006, <http://proceedings.cmwr-xvi.org/>.
[9] Hoteit H, Firoozabadi A. Multicomponent fluid flow by discontinuous Galerkin and mixed methods in unfractured and fractured media, Water Resour Res 41, W11412, doi:10.1029/2005WR004339.
[10] Hægland H, Dahle HK, Eigestad GT, Lie K-A, Aavatsmark I. Improved streamlines and time-of-flight for streamline simulation on irregular grids. Adv Water Resour 2007;30(4):1027–45. doi:10.1016/j.advwatres.2006.09.00.
[11] King MJ, Datta-Gupta A. Streamline simulation: a current perspective. In Situ 1998;22(1):91–140.
[12] Kippe V, Hægland H, Lie K-A. 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] Lin Q, Zhou A-H. Convergence of the discontinuous Galerkin method for a scalar hyperbolic equation. Acta Math Sci 1993;13:207–10.
[14] LeSaint P, Raviart PA. On a finite element method for solving the neutron transport equation. In: de Boor D, editor. Mathematical aspects of finite elements in partial differential equations. Academic Press; 1974. p. 89–145.

[15] Natvig JR, Lie K-A. Fast computation of multiphase flow in porous media by implicit discontinuous Galerkin schemes with optimal ordering of elements, in press.
[16] Pollock DW. Semi-analytical computation of path lines for finite difference models. Ground Water 1998;26(6):743–50.
[17] Reed WH, Hill TR. Triangular mesh methods for the neutron transport equation, Tech. Report, LA-UR-73-479, Los Alamos Sci. Lab., 1973.
[18] Sedgewick R. Algorithms in C. Addison-Wesley; 1990.

Paper B

Fast Solvers for Flow in Porous Media by Implicit Discontinuous Galerkin Schemes with Optimal Ordering of Elements *

* *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/

FAST SOLVERS FOR FLOW IN POROUS MEDIA BASED ON DISCONTINUOUS GALERKIN METHODS AND OPTIMAL REORDERING

JOSTEIN R. NATVIG[1], KNUT–ANDREAS LIE[1,2], AND BIRGITTE EIKEMO[3]

[1] SINTEF ICT, Dept. Applied Math, P.O. Box 124 Blindern, N-0314 Oslo, Norway,
[2] Centre of Mathematics for Applications (CMA), University of Oslo, Norway
[3] University of Bergen, Dept. of Mathematics, Johs. Brunsgt. 12, N-5008 Bergen, Norway

Abstract

We present a family of efficient solvers for hyperbolic transport equations modelling flow in porous media. The solvers are based on discontinuous Galerkin spatial discretisations and implicit temporal discretisation. By applying an optimal reordering algorithm, the corresponding discrete system of (non)linear equations can be solved in one grid-block at a time. This way, we avoid assembly of a full (non)linear system. Our approach allows us to handle large numbers of grid blocks with modest requirements on memory.

1. INTRODUCTION

In this paper we present efficient and accurate solution procedures for a class of linear and nonlinear boundary-value problems of the form

$$\begin{aligned} \alpha u + \nabla \cdot \big(\mathbf{v} F(u)\big) &= \beta, \qquad \mathbf{x} \in \Omega, \\ u &= h(\mathbf{x}), \qquad \mathbf{x} \in \partial\Omega^+. \end{aligned} \tag{1}$$

Here $F(u)$ is a flux function with positive characteristics, $\mathbf{v}$ is a given (nearly) curl-free vector field, and $\partial\Omega^+$ denotes the inflow boundary on which $\mathbf{v}\cdot\mathbf{n} < 0$. Equations of this form arise either as simple models for single-phase flow, like e.g., the time-of-flight equation,

$$\mathbf{v}\cdot\nabla\tau = \phi, \tag{2}$$

or as the result of an implicit semi-discretisation of systems of hyperbolic conservation laws for multiphase and multicomponent flow of the form

$$\phi\partial_t u_i + \nabla\cdot\big(\mathbf{v}F_i(u)\big) = q_i, \qquad i = 1,\ldots,\ell-1. \tag{3}$$

To discretise (1) we use a discontinuous Galerkin (dG) formulation. By this approach, we can easily achieve high-order accuracy with local, compact stencils where the only coupling is between elements sharing a common element face. This yields systems of (non)linear equations with predictable structure: Each (non)linear equation describes the interaction between the degrees-of-freedom of one element and its immediate neighbours sharing a common element face. If the normal velocity $\mathbf{n}\cdot\mathbf{v}$ is constant on each face, this structure can be greatly simplified by using an upwind approximation of the fluxes across element interfaces. In fact, in the linear case the upwind discretisation yields *reducible* systems of equations, for which we can find symmetric permutations that map the global systems to block-triangular systems, where each block involves the

degrees-of-freedom of one or a few elements. In the nonlinear case, the permutation of equations and unknowns yields a block-triangular Jacobian matrix. Finding the permutation (or reordering) is quite easy if we view the fluxes across element interfaces as edges in a directed graph and rephrase the permutation as a topological sort. From elementary graph theory it follows that the reordering can be found by using a depth-first traversal of the grid, in which each cell is visited only once.

A key point in our approach is to exploit this optimal reordering to develop very efficient (non)linear solvers. In the linear case [7], we use a direct solver to factor the small diagonal blocks in the triangular system and thereby obtain a very efficient direct solver. In the nonlinear case (see [6]), the nonlinear subsystems can be solved one-by-one according to the reordering, using for instance a Newton–Raphson method. In both cases the computational effort is reduced significantly from solving a large sparse (non)linear system for all degrees-of-freedom in the domain to solving a sequence of block problems involving a few (non)linear equations for each element. Moreover, in the nonlinear case, we may control the iterations separately for each subsystem and this will generally give better convergence than for the corresponding global nonlinear iteration. Finally, by using the optimal reordering one avoids assembling the global system.

The reordering idea is not new and has been described previously by [3]. Similarly, [2] explore the use of block-triangular structures to construct effective Newton-type nonlinear solvers. However, to the best of our knowledge, these ideas have not previously been used to compute transport in porous media, even though the idea is quite natural and can easily be motivated by the underlying physics: The triangular structure of the equations reflects the directional dependence of the continuous equation (1) that previously has been exploited in streamline methods, see [5]. Due to the positive characteristics of F, the exact solution in each element K will only depend on the upstream points of all streamlines passing through K and be independent of the solution elsewhere in the domain. Using an upwind flux in our dG formulation preserves this one-sided domain-of-dependence, which is a prerequisite for the reordering approach.

2. DISCONTINUOUS GALERKIN DISCRETISATION

A discontinuous Galerkin method starts with a variational formulation. We thus partition the domain into non-overlapping elements $\{K\}$, multiply (1) with a function v from the space of arbitrary piecewise smooth functions V, and integrate by parts over K to get

$$\int_K (\alpha u - \beta) v \, dx - \int_K F(u)\mathbf{v} \cdot \nabla v \, dx + \int_{\partial K} v F(u)\, \mathbf{v} \cdot \mathbf{n} \, ds = 0, \qquad \forall v \in V.$$

We seek a solution in a finite-dimensional subspace $V_h \subset V$ consisting of functions that are smooth inside each element, but may be discontinuous over the element boundaries. Due to the possible discontinuities, we must replace the flux $F(u_h)\mathbf{v}\cdot\mathbf{n}$ with 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

$$a_K(u_h, v_h) = \int_K (\alpha u_h - \beta) v_h \, dx - \int_K F(u_h)\mathbf{v} \cdot \nabla v_h \, dx + \int_{\partial K} \hat{F}\big(u_h, u_h^{\text{ext}}, \mathbf{v} \cdot \mathbf{n}\big) v_h \, ds \quad (4)$$

and find u_h such that

$$a_K(u_h, v_h) = 0, \qquad \forall K, \ \forall v_h \in V_h. \quad (5)$$

For the numerical flux $\hat{F}$ we use an upwind approximation

$$\hat{F}(p, p^{ext}) = F(p)\max(\mathbf{v}\cdot\mathbf{n}, 0) + F(p^{\text{ext}})\min(\mathbf{v}\cdot\mathbf{n}, 0), \tag{6}$$

for inner and outer values p and p^{ext} at element boundaries. Note that there are other consistent flux approximations are also consistent, but they may not preserve the directional dependency we rely on to compute a permutation of the unknowns. For instance, the well-known Lax-Friedrichs flux yields a consistent approximation of the inter-element fluxes, but creates a bidirectional dependence between all elements.

In the following we assume that our elements K are hexahedrals in a regular Cartesian grid and choose $V_h^{(n)} = \{v : v|_K \in \mathbb{Q}^{n-1}\}$, where $\mathbb{Q}^n = \text{span}\{x^p y^q z^r, 0 \leq p, q, r \leq n\}$. A simple basis for this space is the tensor-product of Legendre polynomials L_k. Thus $V_h^{(1)}$ is the space of elementwise constant functions giving a scheme that is formally first order; $V_h^{(2)}$ is the space of elementwise trilinear functions, giving a second-order scheme; etc. Henceforth, dG(n) will denote the discontinuous Galerkin scheme of formal order n having $m = n^d$ unknowns per element in d spatial dimensions.

Substituting the tensor-product Legendre basis functions into (5) and using an appropriate Gaussian quadrature rule to approximate the integrals, we end up with a system of nonlinear equations for the unknown degrees-of-freedom U

$$G_K(U) := a_K^h(u_h, L_k) = 0, \quad \forall K. \tag{7}$$

By writing $U|_K$ for the unknowns in element K and $U|_{\Omega\setminus K}$ for the unknowns outside K and separating G_K into the different terms of (4) and (6), we can write (7) as

$$M_K U_K - B_K + R_K(U|_K) + F_K^+(U|_K) + F^-(U_{\Omega\setminus K}) = G_K^+(U|_K) + G^-(U|_{\Omega\setminus K}) = 0.$$

If we reorder the unknowns, all degrees-of-freedom on the upwind side of element K will be known, meaning that $G^-(U|_{\Omega\setminus K})$ is a known quantity. The only unknowns in (7) for each K are therefore the degrees-of-freedom in K.

3. SEQUENTIAL SOLUTION

As explained in the introduction, the key to obtaining a fast (non)linear solver is to find a reordering $p = (p_1, \ldots, p_N)$ of the elements that renders the system of equations (7) into a block-triangular form

$$\begin{aligned} &G_{p_1}^+(U_{p_1}) && = 0, \\ &G_{p_2}^-(U_{p_1}) + G_{p_2}^+(U_{p_2}) && = 0, \\ &\vdots && \vdots \\ &G_{p_N}^-(U_{p_1}, \ldots, U_{p_{N-1}}) + G_{p_N}^+(U_{p_N}) && = 0. \end{aligned} \tag{8}$$

We therefore consider the directed graph defined by assigning a vertex to each element K_i and a directed edge for each flux $(\mathbf{v}\cdot\mathbf{n})|_{\partial K_i \cap \partial K_j}$ between elements. Thus, an edge from vertex i to vertex j implies that the solution in K_j depends on the solution in K_i. The task of arranging vertices in a sequence according to their position in a directed graph is called a topological sort. To see how a suitable sequence can be constructed, note that $p_i < p_j$ for any vertex i that can be reached from vertex j by going backwards in the graph. By traversing the graph backwards in a depth-first manner, adding vertex j to the sequence when the search backwards from j has been completed, we obtain a topologically sorted sequence. Since a depth-first search only visits each vertex once, the topological sort of a directed graph can be obtained in $\mathcal{O}(N)$ time for N vertices.

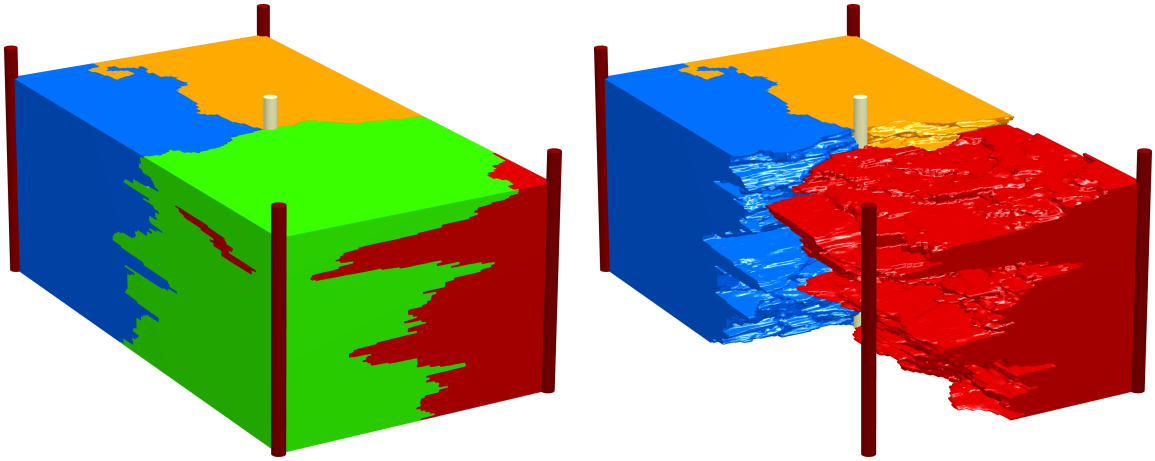


FIGURE 1. Stationary tracer distributions used to delineate the SPE 10 test case into independent flow regions. (Left) The four swept volumes in this scenario are shown in different colours. (Right) By removing one tracer, the intricate surfaces separating the swept volumes are revealed.

If the sequence cannot be found by a single depth-first traversal, the graph has at least one cycle of vertices that can reach any other vertex in the same cycle. Cycles correspond to groups of elements that are made mutually dependent by a nonzero circulation in the velocity field $\mathbf{v}$. The degrees-of-freedom in such a group of elements correspond to a irreducible diagonal block and must be computed simultaneously. Fortunately, cycles can be detected automatically by performing a *forward* depth-first search. By lumping all elements in a cycle into a single vertex, we obtain an acyclic graph where each vertex corresponds to one or a few elements that form an irreducible block of degrees-of-freedoms.

For flow in porous media, the velocity field is typically computed by solving a pressure equation. For incompressible flow, the exact velocity field has zero circulation. A simple argument shows that the same is true for an approximate velocity field computed by a two-point pressure solver. A mixed finite-element solution, on the other hand, may give a velocity field with nonzero circulation. In compressible flow, we may also get nonzero circulation. In our experience, cycles that appear in velocity fields computed by the mixed finite-element method are small and sparse for incompressible and weakly compressible flows.

4. NUMERICAL EXAMPLES

In this section we present a few examples to demonstrate that our dG approach gives efficient and accurate solvers for single-phase and multiphase flow in porous media.

4.1. **Steady-State Tracer Distribution.**

As our first example, we consider the stationary distribution of a set of tracers being continuously injected into a reservoir, modelled by the simple equation

$$\mathbf{v} \cdot \nabla c_\alpha = 0, \qquad c_\alpha|_{\partial\Omega^+} \text{ given.}$$

The reservoir model is taken from [1], has $220 \times 60 \times 85$ grid cells, and consists of a smooth shallow-marine Tarbert formation on-top of a fluvial Upper Ness formation. A vertical injection well is located in each of the four corner and a producer is located in the middle. By launching different tracers in each of the four injectors, we may

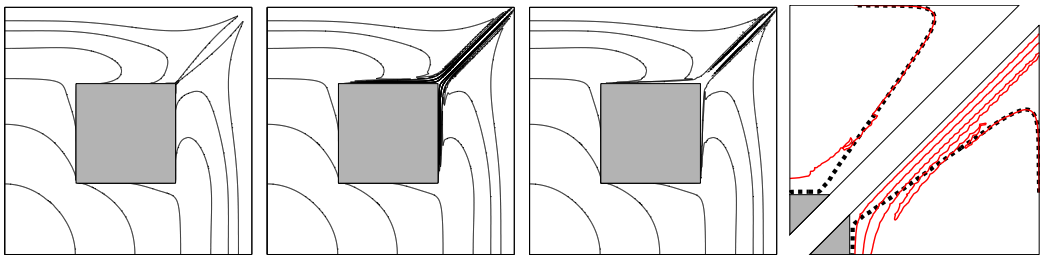


FIGURE 2. Time-of-flight in quarter five-spot with low-permeable region. Contours correspond to [0.075 : 0.15 : 0.875] pore volumes injected. The solution is computed using (from left to right) streamline integration, dG(3), and adaptive dG(3). The figure on the right zooms on the 0.65 contour of dG(3) (red) in the lower half and the adaptive dG(3) (red) in the upper half, both with the streamline solution shown as a dashed line.

determine the volumes swept by each injector and use this to delineate the reservoir into independent flow regions, as shown in Figure 1. The velocity field of this scenario was computed with a two-point flux approximation, with pressure-driven injection. The stationary tracer distributions were computed with a second-order discontinuous Galerkin scheme, and the boundaries between the swept volumes were computed as the 0.5 isosurface of the different tracer saturations.

4.2. **Time-of-Flight.** In our second example, we show time-of-flight computed in a quarter five-spot test case in the unit square. The flow is driven by point sources in the lower left and upper right corners. The permeability and porosity equal 10^{-6} in $(0.3, 0.7)\times(0.3, 0.7)$ and 1.0 elsewhere. The solutions computed on 100×100 elements with the dG-method are shown in Figure 2 together with a reference solution computed by direct integration of time-of-flight along streamlines. To generate the plots the solutions have been sampled using 10×10 points in each element.

This seemingly innocent example turns out to be quite difficult to compute using any finite-difference or finite-volume method. The reason is that the time-of-flight solution has a large gradient downstream from the impermeable region. This rapid variation is impossible to capture accurately with polynomial elements, and will generate oscillations for elements with order higher that one.

Since our solution procedure computes the solution in one element at a time, we are able to implement a simple adaptive procedure. Clearly non-physical solutions can be detected by checking if the solution is increasing from inflow to outflow in each element K. If this is not the case (large) errors can propagate to the next element. To avoid this situation, we recompute the solution in K with a first-order method on a refined grid of $n\times n$ sub-elements. A computation based on this approach is shown in Figure 2.

4.3. **Two-Phase Flow.** In our next example we consider (3) for $\ell = 2$, modelling an oil-water system. The primary unknown is the water saturation s and the flux function is given by $f(s) = s^2/(s^2+(1-s)^2)$. Applying a backward Euler temporal discretisation to (3), we get an equation of the form (1) for each time-step

$$\frac{\phi}{\Delta t}s^n + \nabla\cdot\big(\mathbf{v}f(s^n)\big) = \frac{\phi s^{n-1}}{\Delta t}.$$

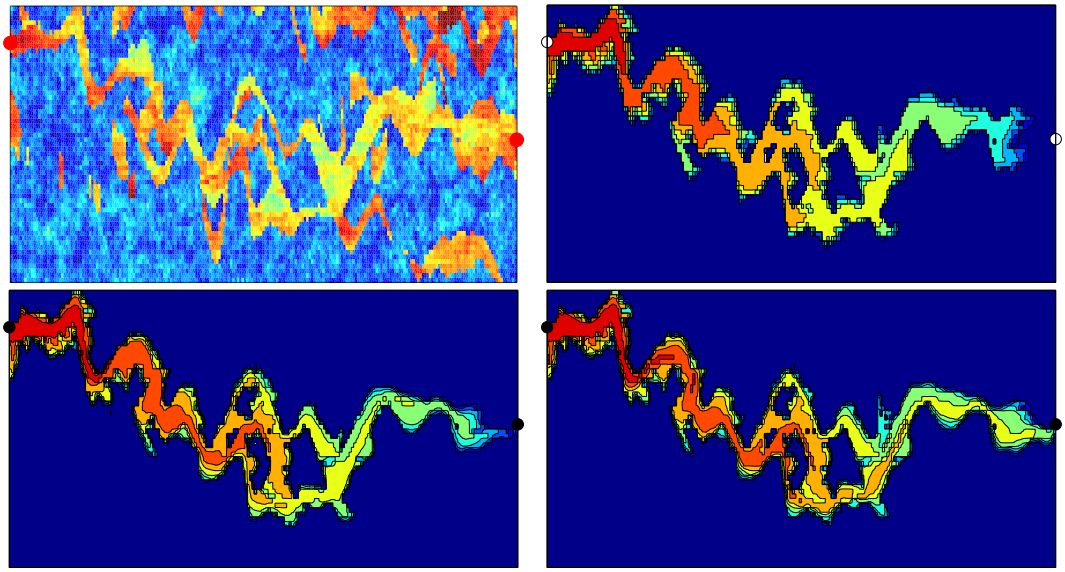


FIGURE 3. Two-phase flow in the fluvial Upper Ness formation of the SPE 10 test case. The figure shows (from top-left to bottom-right) the permeability field, the solutions at time $t = 0.2$PVI computed using dG(1), dG(2), and dG(3).

The two-phase model will generally produce discontinuities at interfaces between injected water and resident oil. Near discontinuities, the dG scheme will tend to produce spurious oscillations, which can be suppressed by performing a post-processing with a nonlinear limiter function after each step, see [6] for more details. Figure 3 shows a solution of a water injection scenario in Layer 37 of the model from [1]. This layer is part of the fluvial Upper Ness formation characterised by high contrast and complex channel patterns. The computations are performed with a fixed velocity field computed with a mixed finite-element method and saturations have been computed with dG(1), dG(2) and dG(3) using very long time steps corresponding to a CFL-number of 2500. The resolution of thin fingers improve with increased order of accuracy as expected except at the front, where the nonlinear limiter function reduce the accuracy to second order.

4.4. **Three-Phase WAG Injection.** In our last example, we consider a water-alternating-gas scenario in a quarter five-spot. Here the primary unknowns are the water and gas saturations, s_w and s_g. To define the flux functions, we introduce the phase mobilities

$$\lambda_w(s_w) = s_w^2/\mu_w, \quad \lambda_g(s_g) = (0.1s_g + 0.9s_g^2)/\mu_g,$$
$$\lambda_o(s_w, s_g) = (1 - s_w - s_g)(1 - s_w)(1 - s_g)/\mu_o,$$

where $\mu_w = 0.35$, $\mu_g = 0.012$ and $\mu_o = 0.8$. The two components of the flux function are $f_\alpha = \lambda_\alpha/(\sum_\alpha \lambda_\alpha)$ for $\alpha = w, g$. This system has only positive characteristics and is strictly hyperbolic except for the single point of 100% gas saturation, where the eigenvalues coincide [4]. Figure 4 shows the time evolution of a WAG injection cycle starting with the injection of 0.05 pore volumes of water, then 0.05 pore volumes of gas, etc. Before each injection step, the pressure and velocity fields are recomputed to

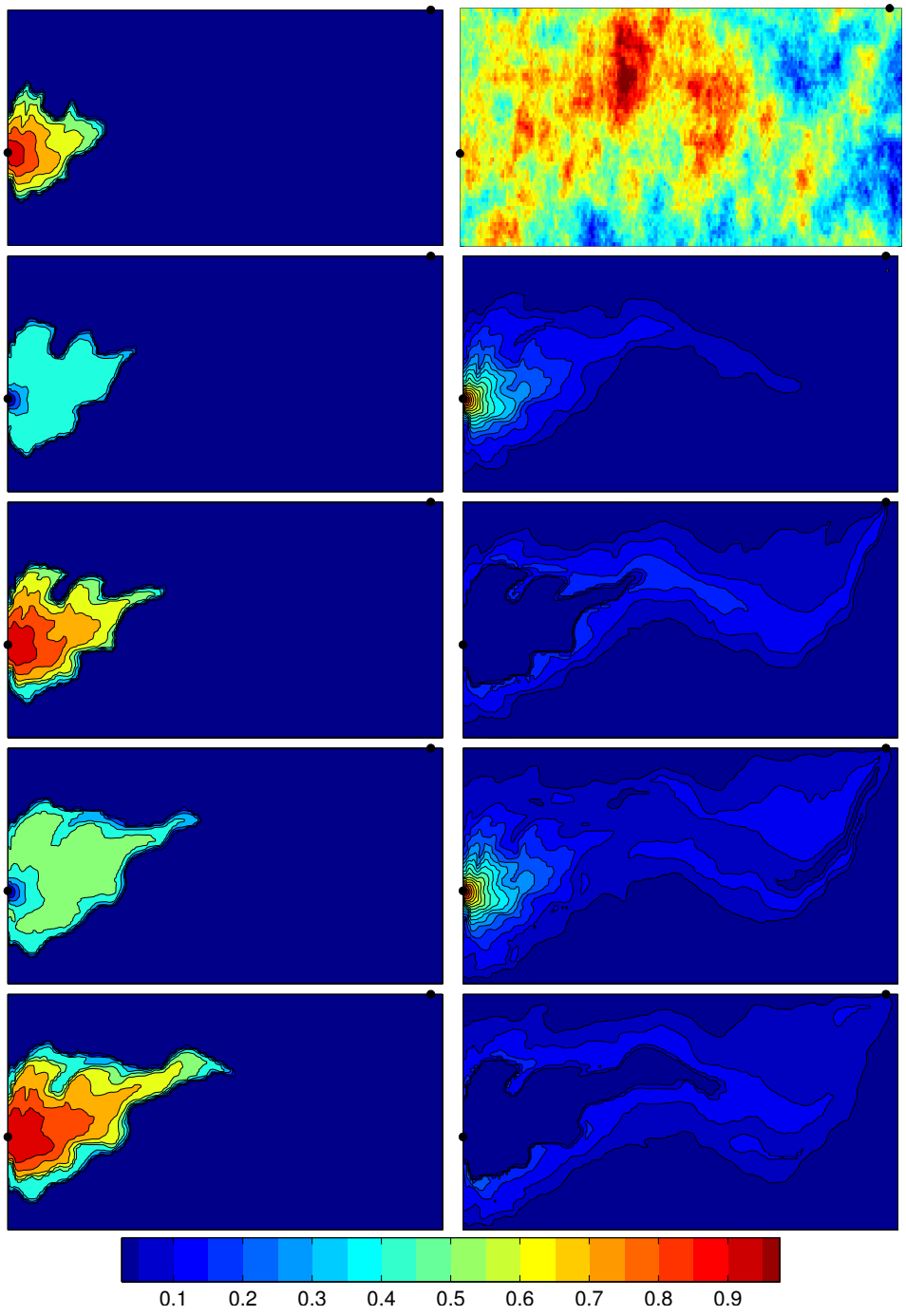


FIGURE 4. Three-phase flow in the smooth Tarbert formation in the 6th layer of the SPE 10 test case. The figures show (top, right) the permeability field with high permeability indicated by light shading, (left column) the water saturation and (right column) the gas saturation in five steps of a WAG cycle. The plotted contours are $[0\!:\!0.1\!:\!1]$ for the water saturation and $[0\!:\!0.05\!:\!1]$ for the gas saturation. The permeability field span values from $1e-6$ to $1e-12$.

account for the change in total mobility. The transport is computed with dG(2) using a CFL-number of 5000.

5. CONCLUDING REMARKS

In this paper we have demonstrated the capabilities of an implicit discontinuous Galerkin discretisation for linear and nonlinear transport in porous media. A sequential solution procedure based on reordering the equations yields a very fast method with the attractive feature that the runtime scales linearly with the number of elements. In the linear case, this offers a competitive alternative to streamline methods for delineating reservoirs. For nonlinear time-dependent problems this scheme yields a nonlinear solver that allows implicit time-stepping in large multiphase flow computations on desktop computers.

For very large problems, domain decomposition may be used to circumvent the memory limitations of a single computer. For time-dependent problems, the sequential solution procedure allows many time steps to be computed in parallel. Neither of these options have been tested yet.

References

[1] 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. url: www.spe.org/csp.

[2] 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.

[3] 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.

[4] R. Juanes and T. W. Patzek. Analytical solution to the Riemann problem of three-phase flow in porous media. *Transp. Porous Media*, 55(1):47–70, 2004.

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

[6] J. R. Natvig and K.-A. Lie. Fast computation of multiphase flow in porous media using discontinuous Galerkin and optimal ordering. submitted, mar 2006.

[7] J. R. Natvig, K.-A. Lie, B. Eikemo, and I. Berre. A discontinuous Galerkin method for computing single-phase flow in porous media. submitted, feb 2006.

Paper C

A Discontinuous Galerkin Method for Computing Time-of-Flight in Discrete-Fracture Models *

* *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/

A DISCONTINUOUS GALERKIN METHOD FOR COMPUTING TIME-OF-FLIGHT IN DISCRETE-FRACTURE MODELS

B. EIKEMO[1], I. BERRE[1], H.K. DAHLE[1], K.-A. LIE[2,3], AND J. R. NATVIG[2]

[1]Dept. Mathematics, University of Bergen, Johs. Brunsgt. 12, N-5008 Bergen, Norway
[2]SINTEF ICT, Applied Mathematics, P.O. Box 124 Blindern, N-0314 Oslo, Norway
[3]Centre of Mathematics for Applications (CMA), University of Oslo, Norway

Abstract

Discrete fracture models, in which fractures are represented individually as lower-dimensional objects, are beginning to appear in simulators for porous media flow. Here we present a discontinuous Galerkin method for computing time-of-flight in discrete-fracture models of fracture-fault systems. Isocontours of time-of-flight are time-lines of porous-media flow and give information about flow patterns, in particular for single-phase flow.

Recent numerical results show that the discontinuous Galerkin (dG) method is efficient and accurate for solving the time-of-flight equation. In this paper, we use two simplified grid models to examine various approaches for the dG discretisation in fractured regions of the porous medium. Comparing the numerical results of the dG approximation with those from a streamline simulator, we demonstrate the importance of a sufficient grid resolution across the fractures, even though the widths of the fractures are very small compared to typical length scales of the unfractured parts of the reservoir.

1. INTRODUCTION

In recent years, advanced drilling techniques and enhances in seismic and geological characterisation of petroleum reservoirs have emerged. Consequently, there is an increased need for more detailed understanding of how local reservoir heterogeneities, such as fractures, affect the oil and gas recovery. A naturally fractured reservoir can be defined as a reservoir containing planar discontinuities created by natural processes like diastrophism and volume shrinkage. Due to the complex geometries and potentially large variations in parameter values, fractures will often have a significant impact on the flow characteristics of a porous medium, and fractured reservoirs represent a challenge for reservoir characterisation, modelling, and simulation.

The traditional way of simulating flow in a fractured medium is by the use of dual-porosity models, where the matrix (unfractured rock) and fractures are treated as two co-existing porous media. Although such models are efficient in some cases, they generally fail to deliver sufficient resolution of the complex flow patterns that develop when a fractured medium is produced. In recent years, several approaches have been taken to describe fracture-fault systems more accurately. These approaches rely a discrete description of individual fractures, using complex (unstructured) gridding schemes in which each fracture is represented explicitly by lower-dimensional objects at cell faces.

In a recent paper (Natvig et al., 2006), we presented a discontinuous Galerkin (dG) method for computing single-phase transport in porous media. Here we present the first step towards extending this method to discrete fracture systems. For simplicity, we only consider conceptual 2D models consisting of a regular Cartesian grid representing the matrix and extra lines at cell edges representing straight fractures. The aim of this first step is to investigate how the dG discretisation is able to handle the geometries and sharp variations in rock properties of fractured fields. In general, we use the same solution procedure as in (Natvig et al., 2006), but due to the high contrasts and different length scales of the rock matrix and the fractures, we investigate different dG approximation strategies for the model equation in the fractures. A key point in our approach is an efficient solution procedure for the resulting system of discrete flow equations. By exploiting *a priori* knowledge of the directions of flow, we may arranging the elements in a suitable sequence such that one does not need to assemble the full system and can compute the solution extremely fast in an element-by-element fashion.

The outline of the paper is as follows: In Section 2, we introduce the time-of-flight formalism as a model for single-phase transport in porous media. In Section 3, we give a brief outline of the dG method and present the variational formulation and discretisation of our model problem, distinguishing between the discretisation in the rock matrix and in the fractures. We also show how to solve the resulting linear system using an a prior reordering of the elements. Numerical examples are presented in Section 4; here, we compare the accuracy of the solutions computed by the dG method to highly resolved solutions obtained by pointwise integration of streamlines. Finally, we draw some conclusions and indicate further work.

2. GOVERNING EQUATIONS

We consider single-phase transport in porous media. The velocity field $\mathbf{v}$ is governed by Darcy's law, and for convenience we assume that $\mathbf{v} = \mathbf{v}(\mathbf{x})$ is given and is nearly irrotational and divergence free. The motion of the fluid is aligned with the velocity field $\mathbf{v}$; thus, all instantaneous transport occurs along integral curves (streamlines). A streamline Ψ is the path traced out by a passive particle moving with the flow given by a velocity field $\mathbf{v}$ such that the vector $\mathbf{v}$ is tangential to Ψ at every point. The time-of-flight $\tau(x)$ is the time needed for a passive particle to travel along a streamline from the inflow boundary to a given point x. Isocontours of $\tau(x)$ are the time-lines in the porous medium and give information about the flow patterns, in particular for single-phase flow. The time-of-flight can be defined by the following integral along a streamline Ψ:

$$\tau(x) = \int_{\Psi} \frac{\phi \, \mathrm{d}s}{|\mathbf{v}(x(s))|}, \tag{1}$$

where ϕ is the porosity of the porous medium. Hence, a simple model for convective transport in $\mathbf{v}$ is the boundary-value problem for time-of-flight τ in Ω:

$$\mathbf{v} \cdot \nabla \tau = \phi, \quad \tau = 0 \text{ in } \partial\Omega^{+}; \tag{2}$$

see (Datta-Gupta and King, 1995). Here, $\partial\Omega^{+}$ denotes the inflow boundary of the fluid. Accurate solution of (2) is rather easy for smooth velocities, but the equation becomes harder to solve when the vector field has large spatial variations and fine-scale details. In

this paper, we will show the efficiency and accuracy of the dG method to simulate single-phase transport in fractured porous media as described by (2); however, the solution strategy also applies for slightly more general models of the same type; see (Natvig et al., 2006). We refer to (Natvig and Lie, 2006) for an extension of the dG methodology in to multiphase and multicomponent flow.

3. DISCONTINUOUS GALERKIN METHOD

The physical domain Ω consists of matrix and fractures. Since fractures exist on a much smaller geometrical scale than the characteristic length scale of the matrix, we assume that we have a discrete fracture model, where the fractures are modelled as one-dimensional curves in a two-dimensional reservoir model. However, for the numerical calculations, we let the fractures have a small width ϵ, so that both the matrix and the fracture are two dimensional.

The domain is partitioned into a regular quadrilateral grid of N elements $\{E_i\}_{i=1}^N$. More precisely, we denote the M elements corresponding to the matrix by $\{K_i\}_{i=1}^M$ and the $N-M$ elements describing the fractures by $\{I_i\}_{i=1}^{N-M}$. As quadrilateral corner-point grids can be transformed to regular grids (Prévost et al., 2002), the method can be extended to also handle more general partitions.

In the following, we describe the discretisation of the time-of-flight equation (2) using a discontinuous Galerkin method (Reed and Hill, 1973), distinguishing between the discretisation in the matrix and in the fracture elements. Thereafter, we explain the numerical solution procedure. It is assumed that the fluid velocity $\mathbf{v}$ is a time-independent function that is given in terms of fluxes across the element edges.

3.1. Approximation in the Matrix.

Let V be the space of arbitrarily smooth test functions. By multiplying (2) with a function $\varphi \in V$ and integrating by parts over each matrix element K, we obtain

$$-\int_K T\mathbf{v}\cdot\nabla\varphi\,\mathrm{d}\mathbf{x}+\int_{\partial K} T\mathbf{v}\cdot\mathbf{n}\varphi\,\mathrm{d}s=\int_K \phi\varphi\,\mathrm{d}\mathbf{x}\quad\forall\varphi\in V, \tag{3}$$

where $\mathbf{n}$ is the outer normal on the element boundary ∂K. We seek a solution in a finite-dimensional subspace $V_h\subset V$, so we replace the exact solution and the test function by $T_h\in V_h$ and $\varphi_h\in V_h$, respectively. The space V_h consists of functions that are smooth inside each element, but may be discontinuous over the element boundaries. Since T_h may be discontinuous over the element boundaries, we must replace the flux term, $T\mathbf{v}\cdot\mathbf{n}$, by a consistent and conservative numerical flux function $\hat{f}$. This leads to the following discrete variational formulation: Find T_h such that

$$-\int_K (T_h\mathbf{v})\cdot\nabla\varphi_h\,\mathrm{d}\mathbf{x}+\int_{\partial K}\hat{f}(T_h,T_h^{ext},\mathbf{v}\cdot\mathbf{n})\varphi_h\,\mathrm{d}s=\int_K\phi\varphi_h\,\mathrm{d}\mathbf{x}\quad\forall K,\quad\forall\varphi_h\in V_h. \tag{4}$$

For inner and outer approximations T_h and T_h^{ext} at the boundaries, the numerical flux $\hat{f}$ is approximated by the upwind flux given by

$$\hat{f}(T_h,T_h^{ext},\mathbf{v}\cdot\mathbf{n})=T_h\max(\mathbf{v}\cdot\mathbf{n},0)+T_h^{ext}\min(\mathbf{v}\cdot\mathbf{n},0). \tag{5}$$

The upwind approximation of the flux preserves the directional dependency that we will later exploit to compute the solution element by element.

The restriction $\varphi|_K$ of a function $\varphi \in V_h$ on an element K is defined by $\varphi|_K \in \mathbb{Q}^{n-1}$, where $\mathbb{Q}^n = \text{span}\{x^p y^q : 0 \leq p, q, \leq n\}$. Hence, for $n = 1$, $V_h^{(1)}$ is the space of functions that are elementwise constant in the unfractured domain and yields a scheme that is formally first order; for $n = 2$, $V_h^{(2)}$ is the space of elementwise bilinear functions on the unfractured domain and yields a formally second order accurate scheme; and so on.

3.2. **Approximation in the Fractures.** Since it is assumed that the width ϵ of the fractures is negligible compared to characteristic length scales of the reservoir, we first consider a modification of the discretisation described above for the fracture elements, $\{I_i\}$, by assuming that there is no variation in time-of-flight across the fracture. This is consistent with the initial reservoir model, which assumes that the fractures have zero width. Depending on the position of the fractures, the thin fracture elements $\{I_i\}$ are placed in either the x_1-direction or the x_2-direction.

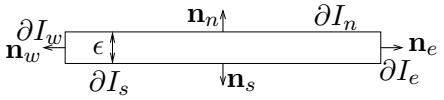


FIGURE 1. Fracture element of width ϵ.

Assuming a constant solution across the fracture, and using the same framework as for the discretisation in the matrix, we obtain the following discrete variational formulation for a fracture element I_i placed in the x_1-direction: Find T_h such that for all $\varphi_h \in V_h$

$$-\epsilon \int_{I_{x_1}} (T_h \mathbf{v}) \cdot \nabla \varphi \, \mathrm{d}x_1 + \epsilon T_h \varphi_h \mathbf{v} \cdot \mathbf{n}_w + \int_{\partial I_s} T_h \varphi_h \mathbf{v} \cdot \mathbf{n}_s \, \mathrm{d}x_1 + \epsilon T_h \varphi_h \mathbf{v} \cdot \mathbf{n}_e + \int_{\partial I_n} T_h \varphi_h \mathbf{v} \cdot \mathbf{n}_n \, \mathrm{d}x_1 = \epsilon \int_{I_{x_1}} \varphi_h \, \mathrm{d}x_1. \tag{6}$$

For fracture elements located in the x_2-direction, a discretisation is obtained in the same manner, using that the solution is constant across the fractures in the x_1-direction. To compute the boundary integral we use the upwind flux function (5), where $\mathbf{n}_w$, $\mathbf{n}_s$, $\mathbf{n}_e$, and $\mathbf{n}_n$ denote the outer normals at each element boundaries. Additionally, we use that $\partial I = \partial I_w \cup \partial I_s \cup \partial I_e \cup \partial I_n$; see Figure 1. Since the order of the scheme is reduced to one in the direction across the fractures, the discretisation is simplified compared with the discretisation (4) in the matrix elements.

As an alternative, we may model the fractures as fully two-dimensional objects and approximate the solution in fracture elements in exactly the same manner as for the matrix elements. This can be motivated based on the fact that the flow changes rapidly in the fractured regions. For this reason, it is also natural to consider a finer grid resolution across the fracture (as opposed to only one element). The two alternative discretisations are discussed further in Section 4 by the means of two numerical examples. See (Hoteit and Firoozabadi, 2005) for a different dG approach.

3.3. **Numerical Solution Procedure.** The approximate solution and the test function on an element E_i can be written as a linear expansion of basis functions. By substituting this into the variational formulations (4) for the matrix elements and (6) for the fracture elements and approximating the integrals using Gaussian quadrature, we get a set of

linear equations for the degrees-of-freedom in each element. Let T_i denote the vector of unknowns for element E_i. If n denotes the order of the scheme, the number of unknowns per element using a dG method is n^2 for matrix elements, and n for fracture elements if the order reduction in the dG approximation across the fractures is applied.

Let us now examine the structure of the linear system. For convenience, we split the coefficient matrix into the element stiffness matrix R_i and the coupling to the other elements through the numerical flux integral F_i. The *exact* solution in E_i depends only on the upwind points of the bundle of streamlines passing through E_i and is independent of the solution elsewhere in the domain (Berre et al., 2005; Natvig et al., 2006). Using the upwind flux (5) preserves this one-side domain of dependence. In other words, the solution in E_i will only be influenced by elements that are intermediate neighbours in the upwind direction. Let $U(i) = \{j \mid \mathbf{v} \cdot \mathbf{n} < 0 \text{ on } \partial E_i \cap E_j\}$ denote the indices of these elements. Then, if F_i^+ denotes the flux *out of* element E_i and F_i^- the flux *into* element E_i, we have

$$-R_i T_i + F_i^+ T_i = B_i - F_i^- T_{U(i)}, \tag{7}$$

where $T_{U(i)}$ are the degrees-of-freedom for all neighbouring elements of E_i in the upwind direction.

The key to obtaining a fast linear solver is to find an a prior reordering of the elements that renders the system of equations (7) in block-triangular form. In other words, we seek a reordering $(p_1, \ldots, p_N)$ of the N elements such that $p_j < p_i$ if $j \in U(i)$, which means that it is possible to start at the inflow boundary and compute the solution element by element. Such a reordering can be found in N operators if it exists. If a reordering does not exist, there must be streamlines that pass through a grid cell more than once. If this occurs, the mutually connected elements must be solved for simultaneously. Nevertheless, the reordering still applies; the only difference is that we locally get a larger linear system associated with the interconnected elements; see (Natvig et al., 2006).

4. NUMERICAL EXPERIMENTS

We now present numerical examples for two different test cases and discuss the dG approximations for computing time-of-flight in fractured porous media.

We consider two test cases. For both, we assume no-flow boundaries and an injector placed in the lower-left corner and a producer in the upper-right corner of the unit square $\Omega = [0\ 1] \times [0\ 1]$. The fractures are of permeability 10^6 D and are located in an elsewhere homogeneous reservoir of permeability 1D as illustrated in Figure 2. The fracture width is set to 0.0001 length units. The velocity field $\mathbf{v}$ is given such that the flux $\mathbf{v} \cdot \mathbf{n}$ is constant over each element face.

We compare solutions obtained by the dG methods with a highly resolved streamline (SL) reference solution. We have also calculated a "reference" solution using the dG approximation of 7th order. For both reference solutions, we have used a grid consisting of 320×320 standard matrix elements in addition to the elements that result from discretising the fractures with a resolution of eight elements in the direction across the fractures. The streamline solutions are obtained by back-tracking streamlines from the cell centre of each element.

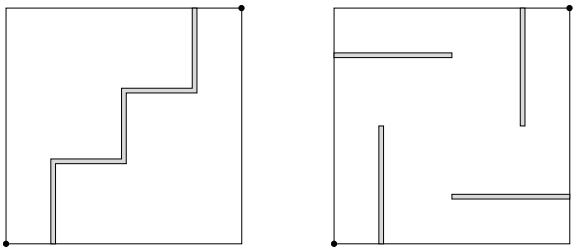


FIGURE 2. Fracture distribution in Case 1 (left) and Case 2 (right).

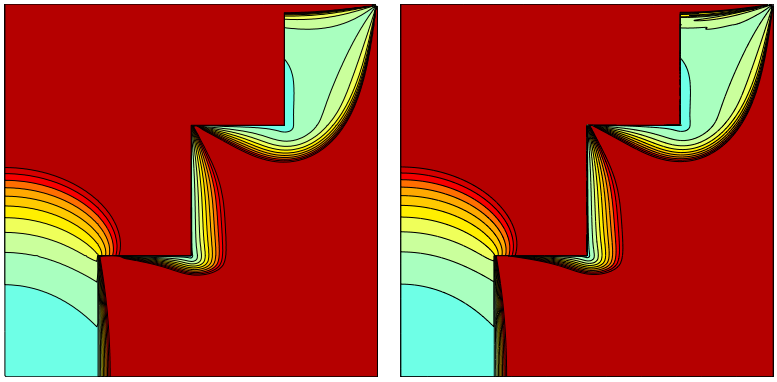
Case 1. In our first test case, a single fracture forms a staircase structure in the flow direction in a elsewhere homogeneous reservoir; see Figure 2. The reference solutions for this test case are shown in Figures 3(a) and 3(b). For all test examples, we have considered a second-order dG approximation.

Figure 3(c) displays the result of the dG approximation with order reduction across the fracture as described in Section 3.2. For this example, the fracture is resolved with one element in the direction across the fracture, thus reflecting the fact that the fracture initially is modelled as one-dimensional. As we can see, the breakthrough time at the production well is highly inaccurate. Figure 3(f) shows the result for the same test example, but without order reduction; that is, the same dG approximation is applied both for the matrix and the fracture elements and in this respect, the fractures are considered as fully two-dimensional. With this approximation, we observe instabilities in the solution, resulting in negative time-of-flights in some regions. On the other hand, the breakthrough-time is more correct than for the approximation displayed in Figure 3(c). To sum up, we see that we faced with two problems: either we get a highly erroneous breakthrough time at the producer, or we get negative values of time-of-flight.

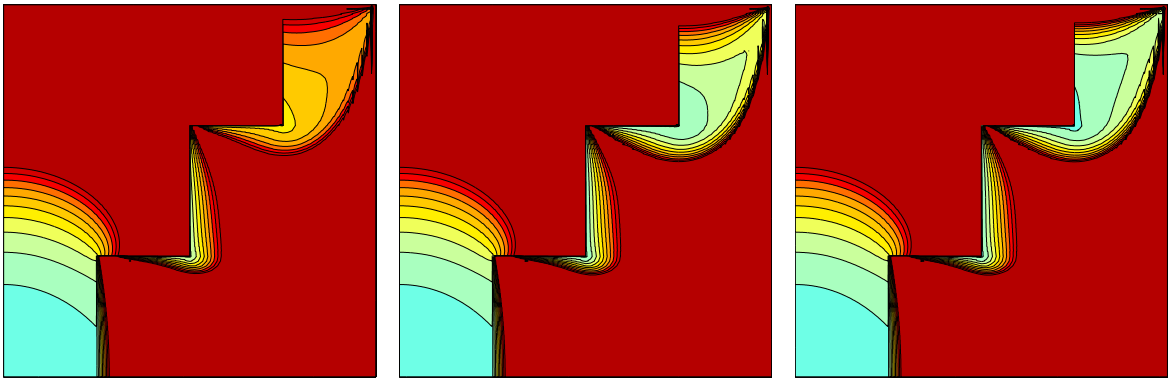
Motivated by the fact that the transport is very rapid in the fractures, and our previous experience with dG methods for obstacle problems (Natvig et al., 2006), we try to increase the grid-resolution in the fractures. However, to avoid instabilities in the solution, we first consider examples where the order reduction of the dG approximation in the fracture elements is kept in the direction across the fracture. The results are depicted in Figures 3(d) and 3(e) for a resolution of four and eight elements across the fractures. Clearly, the results are significantly improved. In Figures 3(g) and 3(h), we display the results for refined fracture resolutions, but without order reduction of the dG approximation in the fractures. For eight elements across the fractures, we obtain a solution that resembles the SL reference solution, although one can still see small signs of instabilities.

Case 2. The fracture distribution in the second test case consists of four fractures in an elsewhere homogeneous reservoir as depicted in Figure 2. Figures 4(a) and 4(b) show the streamline reference solution and the solution obtained with the seventh-order dG method on a fine grid.

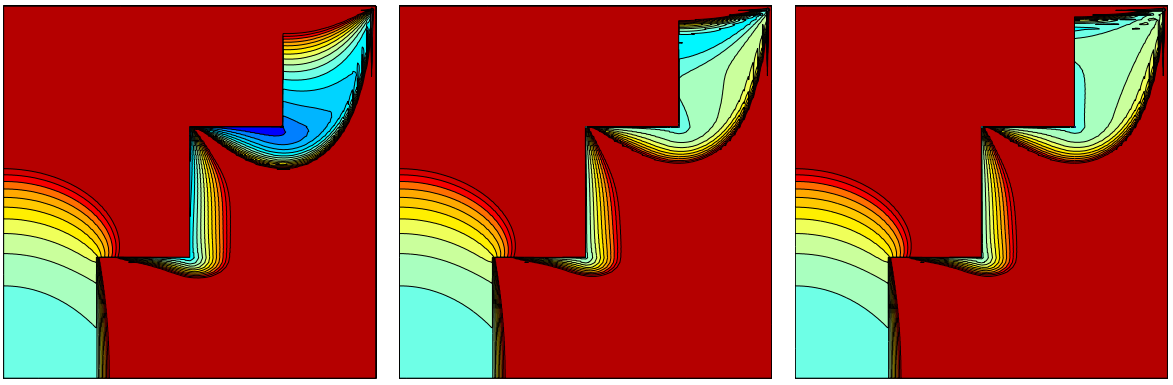
Figures 4(c) and 4(d) show results for a second and fourth-order dG approximation with order reduction and a resolution of eight elements across the fracture when order-reduction is applied. As we can see, the solutions are stable, but fail to completely capture the complex flow in the region near the upper-right fracture. Results for the same example,



(a) SL reference solution. (b) dG reference solution.



(c) Order red., $n_f = 1$. (d) Order red., $n_f = 4$. (e) Order red., $n_f = 8$.



(f) No order red., $n_f = 1$. (g) No order red., $n_f = 4$. (h) No order red., $n_f = 8$.

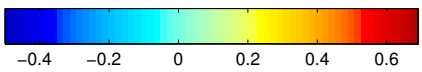
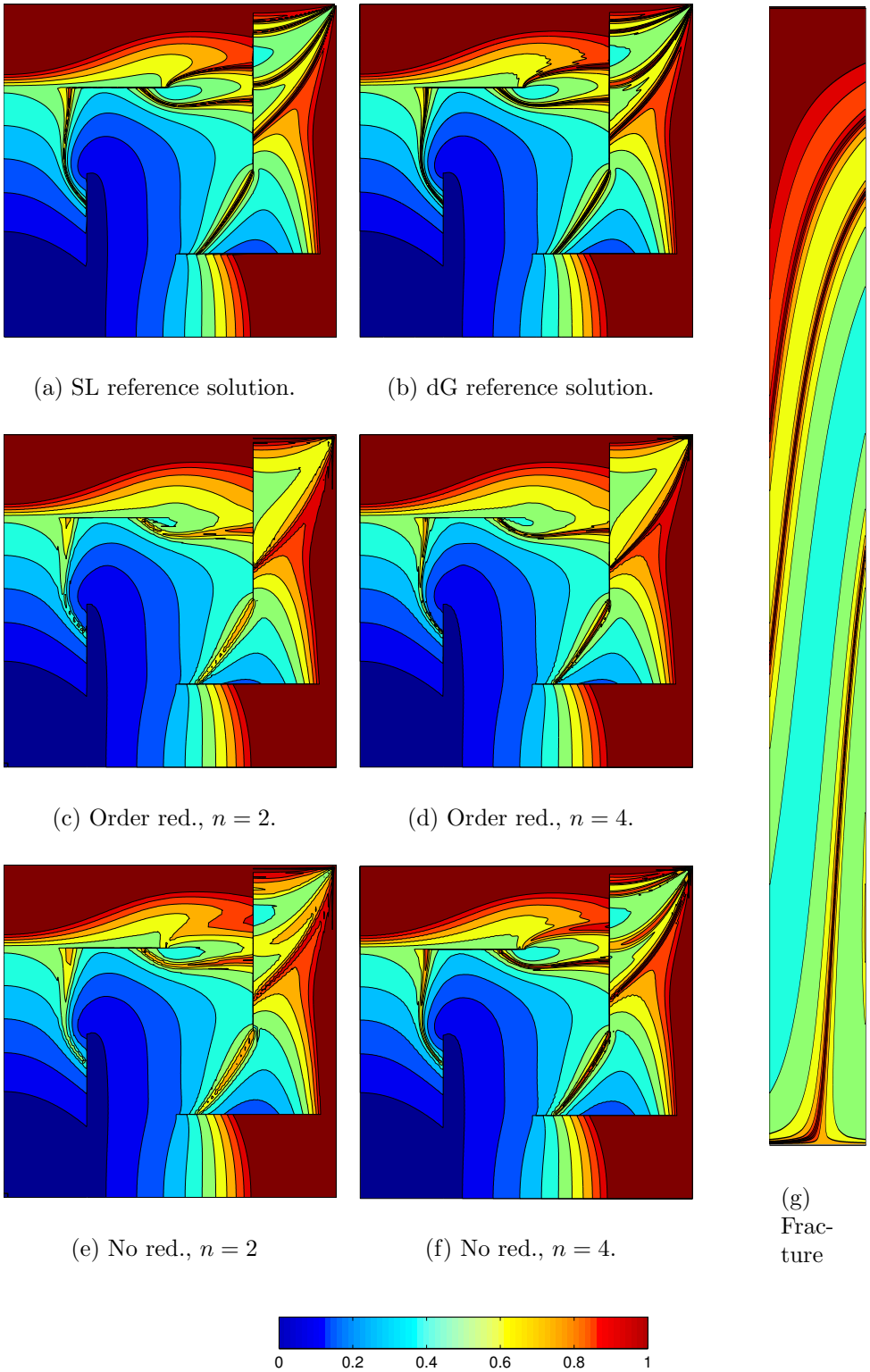


FIGURE 3. Second-order dG approximations with 80×80 standard matrix elements in addition to the elements that result from discretising the fracture by n_f elements in the fracture width, with and without order reduction.



(a) SL reference solution.

(b) dG reference solution.

(c) Order red., $n = 2$.

(d) Order red., $n = 4$.

(e) No red., $n = 2$

(f) No red., $n = 4$.

(g) Fracture

FIGURE 4. Second and fourth-order dG approximations with and without order reduction on a grid with 80×80 standard matrix elements in the fracture width in addition to the elements that result from discretising the fracture by eight elements in the fracture width. The right plot shows the approximation with a subresolution of 25×25 points in each element for the fracture from the upper-right part of the reservoir.

but without order reduction, are displayed in Figures 4(e) and 4(f). Here, the solutions are improved, but, as for Case 1, we can observe signs of instabilities in the solutions.

To further illustrate the necessity of a sufficient grid-resolution across the fractures, we have plotted the time-of-flight computed with the streamline simulator in the upper-right fracture element of Case 2 in Figure 4(g). This plot clearly demonstrates the complex flow pattern in the fracture.

5. Concluding Remarks

In this paper, we have investigated an efficient discontinuous Galerkin method for computing time-of-flight in fractured porous media. In particular, we have revealed the importance of a sufficient grid-resolution across the fractures. This is necessary since the time-of-flight is an integrated quantity that exhibits fine-scale details and contains large spatial variation within the fractures (even though these may have been modelled as lower-dimensional objects in the original grid model). To assure a stable solution, one can apply order reduction of the dG approximation in the direction across the fractures. This also results in a more efficient scheme due to a reduced number of unknowns for fracture elements.

A crucial part of the methodology is an optimal reordering of unknowns, which is based on prior information of the direction of flow. The result is an efficient method, which can be a grid-based alternative to streamline methods. Extension of the methodology to triangular and unstructured grids is currently in progress.

References

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

Datta-Gupta A., and M. J. King (1995), A semianalaytic approach to tracer flow modeling in heterogeneous permeable media, *Adv. Water Resour.*, 18(1).

Natvig, J. R., and K.-A. Lie (2006), Fast computation of multiphase flow in porous media using discontinuous Galerkin and optimal ordering, preprint.

Natvig, J. R., K.-A. Lie, B. Eikemo and I. Berre (2006), A discontinuous Galerkin method for computing single-phase flow in porous media, preprint.

Prévost, M., M. G. Edwards, and M. J. Blunt (2002), Streamline tracing on curvilinear structured and unstructured grids, *SPE Journal*: 139–148.

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

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

Paper D

A Discontinuous Galerkin Method for Transport in Fractured Media using Unstructured Triangular Grids *

* Submitted.

A DISCONTINUOUS GALERKIN METHOD FOR TRANSPORT IN FRACTURED MEDIA USING UNSTRUCTURED TRIANGULAR GRIDS

B. EIKEMO, K.-A. LIE, G. T. EIGESTAD, AND H. K. DAHLE

Abstract. The possibility to couple discrete (fractures, shear zones) and continuous (rock matrix) model elements is a prerequisite for simulating flow and transport processes in fractured rocks. The method described in this paper uses unstructured triangular grids to explicitly represent the fractures and matrix rock as a single continuum in which one can compute the transport using a higher-order discontinuous Galerkin method. By modelling the complex fracture networks explicitly, very complex structures can be modelled and using unstructured triangular grids may be necessary to accurately model realistic cases. Herein we consider single-phase equations for advective transport, which have an inherent causality in the sense that information propagates along streamlines. Our discontinuous Galerkin discretization preserves this causality. We can therefore use a simple topological sort of the graph of discrete fluxes to reorder the degrees-of-freedom such that the discretised linear system gets a lower block-triangular form, from which the solution can be computed very efficiently using a single-pass forward block substitution. The accuracy and utility of the resulting transport solver is illustrated through several numerical experiments.

1. Introduction

Accurate representation of fractured reservoirs represents a challenge for the characterisation, modelling, and simulation of petroleum and groundwater reservoirs, see [4, 12, 3]. Fractured reservoirs are complex geological structures, where fractures (cracks and joints created by rock stress) have higher permeability and porosity than the surrounding rock (matrix). Although the aperture of fractures is very small compared with the dimensions of the reservoir, the fracture network often forms the primary pathway for fluid flow and mass transfer and has a significant impact on the flow characteristics of the porous medium. The matrix blocks between the conducting fractures, on the other hand, can significantly increase the storage capacity of the rock.

Models for fractured media have traditionally been of two general types: discrete or multi-continua (porosity) models. In a discrete model, the fractures are considered as discrete structures integrated in the surrounding rock matrix. With such a model we have the possibility to model single- and multiphase flow and transport processes accurately. Using multi-continua models, we have to make assumption that an representative elementary volume cannot only be obtained for porous medium–the rock matrix–but also for the fractured system.

Date: Draft August 26, 2008.

In a dual-porosity model, for instance, the rock is characterised as two overlapping continua, which are both treated as porous media, meaning that also the matrix blocks are assigned a value of porosity greater than zero.

It is principally possible to use different flow and transport models for the different continua. Exchange terms describing the interaction between the matrix system and the fracture system are very important using multi-continua models, see [11]. For a rock mass with large porous blocks between the conducting fractures, multi-continua models have been used to account for the release of fluid from storage in the matrix blocks into the fracture network. The primary advantage of multi-continua flow models is that they provide a mechanism to account for the delay in the hydraulic response of the rock caused by fluid that is resident in less permeable matrix blocks.

The interaction of fracture and matrix porosities and permeabilities is very complex and often makes simple models highly inaccurate. Indeed, it is widely recognised that state-of-the-art simulation methods based upon multi-continua descriptions are not able to deliver sufficient resolution of the complex flow patterns that develop when a fractured reservoir is produced. Several approaches have therefore been taken to accurately describe fracture-fault systems on a grid-cell scale, that is, based upon complex gridding schemes in which fractures are represented explicitly as lower-dimensional objects at the cell faces. Herein we consider an even more ambitions modelling approach that has increased in popularity lately; in this approach fractures are represented explicitly as thin volumetric cells in a highly detailed geological model. In the following we consider single-phase flow in semi-realistic 2D models of fractured reservoirs and use unstructured, conforming triangular grids, where the fractures themselves are represented explicitly as cells with small width and high permeability (and porosity). This will lead to models with highly contrasting reservoir properties and very complex hydraulic conductivities. To accurately model the flow and transport in regions characterised by high contrast in permeability between the fractures and the matrix, we will also briefly investigate the use of local adaptive refinement.

A simple single-phase model is often sufficient to reveal the major displacement patterns in a fractured medium (e.g., if represented as a single continuum with fractures as volumetric objects). Computing single-phase flow essentially amounts to solving an elliptic pressure equation. However, to further understand the flow mechanisms one can consider various derived quantities like timelines, influence regions, reservoir partitioning, tracer profiles, well pairs, etc., that may be more visual and intuitive than pressure values and discrete fluxes. One particular quantity of interest is the time-of-flight, which can be used to identify areas affected by contaminations in groundwater flow or to determine drainage and flooded volumes in petroleum reservoirs.

Most of such derived quantities are often associated with, and computed by, streamlines methods. However, since they all can be described by (steady-state) transport equations, one could equally well use a grid-based method: the purpose of our paper is to develop a finite-volume method for solving time-of-flight type equations to characterise flow patterns and to compute fluid transport for highly detailed models with explicit fracture modelling. To discretize the

time-of-flight equation, we will use a higher-order discontinuous Galerkin (dG) method, which results in a linear system having a block structure where each block corresponds to the degrees-of-freedom in a single cell (which we sometimes will refer to as an element). Blocks corresponding to neighbouring cells in the grid are coupled through the numerical flux function used to approximate the physical flux over cell interfaces. By introducing an upwind flux approximation, the elements can be ordered to ensure that the linear system has a lower block-triangular form, where each block corresponds to the degrees-of-freedom in a single cell or in a collection of cells having circular dependence due to rotation in the velocity field. Given the triangular form, the linear system can be decomposed to a set of small problems, one for each block and solved using a forward block substitution. This solution procedure is very efficient and has very low memory requirements: once the elements have been reordered, the linear system can be assembled and solved in a local block-by-block fashion. For more details on the efficiency of the reordering method, we refer the reader to [8, 7], in which the same ideas are applied to multiphase flow. The ideas presented herein are a continuation of the research in [5], where we presented a family of discontinuous Galerkin schemes for simulating flow in idealised fractured media using rectangular grids. In this representation, the orientation of the fractures are restricted to being horizontal or vertical.

The rest of this paper is organised as follows: In Section 2 the equations used to model single-phase flow are described in detail. Next, Section 3 introduces the discontinuous Galerkin method used to discretize the fluid transport equations. Then, numerical results for single-phase transport in fractured 2D media are given in Section 4. We also verify the accuracy and convergence rates of our schemes using a simple unfractured case with known analytical solution. Finally, in Section 5 we summarise and give main conclusions.

2. Single-Phase Flow Models

Single-phase flow in an incompressible porous medium is typically modelled by a mass-balance equation in combination with Darcy's law. If we assume gravity to be negligible, the governing equations can be written

$$\nabla \cdot \mathbf{v} = f, \qquad \mathbf{v} = -\frac{1}{\mu}\mathbf{K}\nabla p, \qquad \mathbf{x} \in \Omega. \tag{1}$$

This system can be solved to compute the pressure p and the volumetric flow velocity $\mathbf{v}$ if given a specification of the fluid sources f, the rock permeability $\mathbf{K}$, the fluid viscosity μ, and proper conditions at the boundary $\partial\Omega$ of the physical domain Ω. Alternatively, the system can be written as a second-order elliptic equation for the pressure. To simplify the presentation, we assume that there are no internal fluid sources or sinks and that the flow governed by (1) is driven entirely by conditions set on the inflow and outflow boundaries, denoted $\partial\Omega^-$ and $\partial\Omega^+$, respectively.

For many purposes, (1) does not give a sufficient description of the flow patterns and it is therefore customary to introduce additional transport equations to describe quantities like tracers, contaminants, etc. that are passively advected with the single-phase flow. This paper focuses on such transport equations. For

simplicity, we will henceforth assume that $\mathbf{v} = \mathbf{v}(\mathbf{x})$ is given and is divergence free and irrotational. (Later we will also assume that $\mathbf{v}$ is given implicitly in the form of fluxes that are constant on each element interface.) Given a fixed flow velocity, the concentration q of a passively advected quantity evolves according to the linear hyperbolic equation

$$\phi q_t + \mathbf{v} \cdot \nabla q = 0, \qquad q|_{\partial\Omega^-} = q^-(\mathbf{x}, t), \tag{2}$$

where ϕ is the porosity of the medium. The steady-state version of (2),

$$\mathbf{v} \cdot \nabla q = 0, \qquad q|_{\partial\Omega^-} = q^-(\mathbf{x}), \tag{3}$$

describes the stationary distribution of a tracer that is injected into a reservoir at the inflow boundary $\partial\Omega^-$. This equation can, for instance, be used to determine the spatial region influenced by an inflow boundary (or a fluid source), or by reversing the sign of $\mathbf{v}$, the region influencing an outflow boundary (or drained by a fluid sink). Within reservoir simulation, this could typically be used to compute the swept region of an injector or the drainage region of a producer (or combinations thereof).

Another quantity of interest is the time-of-flight $\tau = \tau(\mathbf{x})$, which is defined as the time needed for a passive particle to travel from a point on the inflow boundary to a given point $\mathbf{x}$. Iso-contours of τ define natural timelines in a reservoir. To define τ, we introduce streamlines, which are a family of curves that at any point are tangential to the velocity vector $\mathbf{v}$ of the flow. For a steady velocity (as considered herein), streamlines coincide with the path traced out by a passive particle moving with the flow field. The time-of-flight τ is defined as

$$\tau(\mathbf{x}) = \int_{\psi} \frac{\phi(r)\, dr}{|\mathbf{v}(\mathbf{x}(r))|}, \tag{4}$$

where ψ denotes the streamline that connects $\mathbf{x}$ to an inflow boundary (or fluid source) and r denotes the arclength along the streamline. Note that modern streamline methods use the time-of-flight τ rather than the arclength r as spatial coordinates. Equation (4) may alternatively be written in differential form as,

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

The transport equations (3) and (5) are special cases of the more general equation

$$\mathbf{v} \cdot \nabla q = H(q, \mathbf{x}), \qquad q|_{\Omega^-} = h(\mathbf{x}, t). \tag{6}$$

Similarly, (2) comes on the form (6) if we introduce an appropriate semi-discretization in time. Accurate solution of (6) is important in areas such as oil recovery and groundwater hydrology to reveal the transport properties of $\mathbf{v}$. Solving (6) is rather easy for smooth velocities, but becomes harder when $\mathbf{v}$ has large spatial variations and exhibits fine-scale details that are important for the global flow pattern.

In the following we present an efficient strategy for solving transport equations on the form (6) on unstructured triangular grids where we combine higher-order discontinuous Galerkin (dG) spatial discretizations with an upwind numerical flux function that creates a one-sided dependency between the elements

in the grid and ensures that we can find a reordering of the elements such that the resulting system becomes lower block-triangular and can be solved block-by-block. We have previously studied the dG-reordering method for rectangular grids [10, 5, 9], for which it proved to be both accurate and highly efficient. In [8], we demonstrated that the same technique can be applied to semi-discrete nonlinear transport equations of the form $\mathbf{v} \cdot \nabla \mathbf{F}(\mathbf{q}) = \mathbf{H}(\mathbf{q}, \mathbf{x}, t)$ that describe multiphase and multicomponent flow when gravity, capillarity, and dispersivity are neglected.

3. Discontinuous Galerkin Schemes with Optimal Ordering

To develop higher-order discontinuous Galerkin methods, we start with a variational formulation of (6). We then partition the solution domain Ω into an unstructured grid consisting of non-overlapping triangular elements (cells) $\{T_k\}$, and seek solutions in a finite-dimensional space V_h consisting of piecewise smooth functions that may be discontinuous over element interfaces. 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, and let $V_h^{(n)} = \{\varphi : \varphi|_{T_k} \in \mathbb{Q}^n\}$. Thus, $V_h^{(0)}$ is the space of elementwise constant functions, which will give a scheme that is formally first-order accurate. Similarly, $V_h^{(1)}$ is the space of elementwise linear functions giving a formally second-order accurate scheme, and so forth. Henceforth, we use dG(n) to denote the discontinuous Galerkin approximation of polynomial order n. Inside each element T_k, the discrete solution q_h can be written

$$q_h(T_k) = \sum_{i=1}^{m_k} q_i^k L_i^k, \qquad \forall T_k. \tag{7}$$

where $\{L_i^k\}$ is some basis for $V_h^{(n)}$ on T_k and m_k is the number of associated degrees-of-freedom. The unknown coefficients $\{q_i^k\}$ are collected in the vector Q for the whole domain and in (sub)vector Q_T for element T.

The approximate solution q_h is determined as the unique solution of the following weak formulation of (6)

$$a_T^h(q_h, \varphi_h) = b_T^h(q_h, \varphi_h) \quad \forall T, \quad \forall \varphi_h \in V_h^{(n)}, \tag{8}$$

where

$$\begin{aligned} a_T^h(q_h, \varphi_h) &= -\int_T (q_h \mathbf{v}) \cdot \nabla \varphi_h d\mathbf{x} + \int_{\partial T} \mathbf{v} \cdot \mathbf{n}\, q_h \varphi_h ds, \\ b_T^h(q_h, \varphi_h) &= \int_T H(q_h, \mathbf{x}) \varphi_h d\mathbf{x}. \end{aligned} \tag{9}$$

Since the solution is discontinuous over element interfaces, we will use an upwind flux to approximate the integrand of the second integral in $a_T^h(\cdot, \cdot)$,

$$\begin{aligned} \mathbf{v} \cdot \mathbf{n}\, q_h &\approx \hat{f}(q_h, q_h^{ext}, \mathbf{v} \cdot \mathbf{n}) \\ &= q_h \max(\mathbf{v} \cdot \mathbf{n}, 0) + \max(q_h^{ext}, 0) \min(\mathbf{v} \cdot \mathbf{n}, 0). \end{aligned} \tag{10}$$

Here q_h and q_h^{ext} are the inner and outer approximations at the element interfaces. The upwind approximation of the flux preserves the directional dependency of the underlying continuous equation (6). In other words, the solution

on T will only be influenced by elements $\mathcal{U}(T)$ that are intermediate neighbours in the upwind direction, which we later exploit to compute the solution in a blockwise fashion. Formally, $\mathcal{U}(T)$ consists of all elements E such that $(\mathbf{v}\cdot\mathbf{n}_T)|_{\partial E\cap\partial T} < 0$, where $\mathbf{n}_T$ is the outward-pointing normal to T. Notice that compared with a standard upwind approximation, we have introduced an additional clipping, $\max(q_h^{ext}, 0)$, to prevent negative values from propagating downstream. Negative values are unphysical, but may arise when using high-order polynomials.

To find a solution to (8), we choose trial functions $\varphi_h = L_i^k$ and evaluate (9) using appropriate quadrature rules. This gives a set of linear equations for the degrees-of-freedom in each element,

$$A_T Q = B_T, \qquad (A_T)_{ij} = a_T^h(L_i, L_j), \quad (B_T)_i = b_T^h(L_i, L_j).$$

For convenience, we split the coefficient matrix into the element stiffness matrix R_T and the coupling to other elements through the numerical flux integral $F_T(Q)$. Given the upwind approximation of the flux (10), we can split the flux integral in two parts. Let $F_T^+ Q_T$ denote the flux out of element T and $F_T^- Q_{\mathcal{U}(T)}$ denote the flux into element T. Hence, the following system of linear equations is obtained

$$\big(-R_T + F_T^+\big)Q_T + F_T^- Q_{\mathcal{U}(T)} = B_T, \qquad \forall T. \tag{11}$$

The coefficient matrix has a block-banded structure, where the size of each block is given by the number of degrees-of-freedom in each element or connected collection of elements, see [8] for a more detailed discussion.

A fast linear solver can now be constructed by observing that the solution in each element can be computed by inverting $(-R_T + F_T^+)$ once the solution is known in all upstream neighbours of T. We may therefore construct the solution locally, starting at inflow boundaries (or fluid sources) and proceeding element by element downstream. From a computational point of view, it is more convenient to look at this as an optimal ordering of unknowns that renders the system of equations (11) in lower block-triangular form. If N_e is the number of elements, such an ordering can be found in N_e operations if it exists.

If the reordering of elements does no exist, there must be circular dependence among some of the elements and these mutually dependent elements must be solved for simultaneously. Nevertheless, the reordering still applies, the only difference is that we locally get a block system associated with a set of interconnected elements instead of a single element. More details are found in [10, 8].

4. Numerical Examples

In [5], we presented a dG scheme for computing time-of-flight in fractured porous media represented on rectangular grids. The use of rectangular grid restricts the orientation of fractures to be either horizontal or vertical. In this section we will consider more realistic fracture distributions modelled on triangular grids present results from selected numerical experiments using higher-order dG schemes and optimal ordering of triangular elements. For each example, the forcing velocity field will either be given by an analytical expression or be

computed by a standard conservative method for solving the first-order system of flow equations (1) or its second-order counterpart, in which case the velocity will be divergence free and nearly irrotational. We will mainly apply the dG schemes to compute time-of-flight in semi-realistic examples of fractured media. Because of the localised nature of the dG formulation, using hybrid grids consisting of both rectangular and triangular cells are within reach, but is not considered herein. However, we show one example of adaptively refined grids.

Case 1 (Convergence Study). We start by verifying the accuracy and convergence rates of discontinuous Galerkin schemes on triangular elements. To this end, we consider a rotating velocity field $\mathbf{v} = (y, -x)$ in the domain $[1, 2] \times [1, 2]$. Let $T = 0$ on the inflow boundaries ($x = 1$ and $y = 2$), then the exact time-of-flight is given by

$$T(x, y) = \tan^{-1}\left(\frac{y}{x}\right) - \tan^{-1}\left(\frac{\min(\sqrt{x^2 + y^2 - 1}, 2)}{\max(\sqrt{\max(x^2 + y^2 - 4, 0)}, 1)}\right). \tag{12}$$

Tables 1 and 2 present L_2-errors and convergence rates for a grid-refinement study performed by increasing the order n in dG(n) on four grid types with increasing roughness (see Figure 1):

Grid 1: triangulation of a uniform $N \times N$ Cartesian grid.

Grid 2: uniform refinement and triangulation of a 10×10 base grid where each internal node has been given a random perturbation up to 20% in each spatial direction.

Grid 3: same as Grid 1, but with a perturbation up to 20% of *all* inner nodes on the $2N \times N$ grid.

Grid 4: same as Grid 2, but with a perturbation up to 20% of *all new* nodes on the $2N \times N$ grid for each refinement.

In Table 1 the L_2-errors are measured in a smooth part of the domain, $[1, 1.3] \times [1, 1.3]$, while in Table 2 the error is integrated over the whole domain. The tables indicate how different roughness[1] in the refined grids impacts the L_2-errors and the convergence rates. For the perturbed grids the rates of convergence are computed by comparing to two different mesh sizes: the average maximum mesh size, which is the average of the maximum cell edges of each element, and by the maximum mesh size, which is the largest cell edge in the domain.

Grids 1 and 2 are refined such that the elements approach half of parallelograms in the asymptotic limit, and hence we observe the expected order of accuracy in smooth regions. For the whole domain, however, we get reduced convergence rates because of the kink in the solution along the circular arc $x^2 + y^2 = 5$. This agrees with the results in [10] for rectangular elements. Note that the kink may impact the regularity of the analytical solution, such that the decays of convergence rates are expected.

[1]Rough grids are defined in the literature (see e.g., [6]) as quadrilateral grids that do not approach parallelograms as the grids are refined. Here, the triangular grids are constructed by dividing each quadrilateral of a quadrilateral grid into two triangles which have one common edge. By this definition, Grids 3 and 4 are rough grids. Similar convergence studies have been performed in [1], where in general a decay in convergence rates may be seen.

TABLE 1. L_2-errors and convergence rates over a smooth part of the domain, $[1, 1.3] \times [1, 1.3]$, for a grid refinement study with dG(n) on a series of $2N \times N$ grids for Grids 1 to 4. Convergence rates on Grids 3 and 4 are computed in terms of the average maximum and maximum mesh size, respectively.

	dG(0)		dG(1)		dG(2)		dG(3)	
N	Error	Rate	Error	Rate	Error	Rate	Error	Rate
10	2.17e-03	—	2.14e-05	—	4.52e-07	—	7.67e-09	—
20	1.09e-03	1.00	5.47e-06	1.97	5.88e-08	2.94	4.81e-10	4.00
40	5.47e-04	1.00	1.37e-06	1.99	7.38e-09	2.99	3.01e-11	4.00
80	2.74e-04	1.00	3.45e-07	1.99	9.12e-10	3.02	1.88e-12	4.00
160	1.37e-04	1.00	8.70e-08	1.99	1.12e-10	3.02	1.18e-13	4.00
10	2.05e-03	—	2.16e-05	—	4.50e-07	—	8.62e-09	—
20	1.03e-03	1.00	5.43e-06	1.99	5.55e-08	3.02	5.51e-10	3.97
40	5.15e-04	1.00	1.36e-06	2.00	6.91e-09	3.01	3.48e-11	3.99
80	2.58e-04	1.00	3.42e-07	1.99	8.50e-10	3.02	2.16e-12	4.01
160	1.29e-04	1.00	8.62e-08	1.99	1.06e-10	3.00	1.35e-13	4.01
10	2.28e-03	—/—	2.22e-05	— /—	4.99e-07	—/—	1.01e-08	—/—
20	1.13e-03	1.01/1.12	5.86e-06	1.92/2.18	6.68e-08	2.90/3.22	6.76e-10	3.90/4.52
40	5.67e-04	0.99/1.04	1.50e-06	1.96/2.05	8.74e-09	2.94/3.03	4.86e-11	3.80/4.03
80	2.85e-04	0.99/1.03	3.84e-07	1.97/2.03	1.15e-09	2.93/3.12	3.25e-12	3.90/3.92
160	1.43e-04	1.00/1.03	9.76e-08	1.98/2.04	1.52e-10	2.92/2.97	2.20e-13	3.88/4.01
10	2.05e-03	—/—	2.16e-05	—/—	4.50e-07	—/—	8.62e-09	—/—
20	1.12e-03	0.90/0.91	6.36e-06	1.81/1.83	7.19e-08	2.71/2.74	8.32e-10	3.46/3.49
40	6.09e-04	0.90/0.93	1.96e-06	1.75/1.80	1.26e-08	2.59/2.67	9.09e-11	3.30/3.40
80	3.35e-04	0.88/0.90	6.25e-07	1.69/1.73	2.28e-09	2.53/2.59	1.07e-11	3.16/3.24
160	1.94e-04	0.80/0.82	2.02e-07	1.67/1.71	4.66e-10	2.35/2.39	1.42e-12	2.99/3.05

TABLE 2. Same as Table 1, but with the L_2-errors and convergence rates measured over the whole domain, $[1, 2] \times [1, 2]$.

	dG(0)		dG(1)		dG(2)		dG(3)	
N	Error	Rate	Error	Rate	Error	Rate	Error	Rate
10	1.92e-02	—	8.95e-04	—	2.67e-04	—	1.49e-04	—
20	1.01e-02	0.92	3.06e-04	1.55	1.00e-04	1.42	5.34e-05	1.48
40	5.34e-03	0.92	1.10e-04	1.47	3.23e-05	1.63	1.68e-05	1.67
80	2.81e-03	0.92	3.97e-05	1.48	1.07e-05	1.60	5.46e-06	1.62
160	1.47e-03	0.93	1.42e-05	1.48	3.48e-06	1.61	1.69e-06	1.69
10	1.96e-02	—	1.08e-03	—	3.41e-04	—	2.07e-04	—
20	1.05e-02	0.90	3.65e-04	1.57	1.13e-04	1.60	6.20e-05	1.74
40	5.62e-03	0.90	1.28e-04	1.51	3.61e-05	1.64	1.98e-05	1.64
80	3.00e-03	0.91	4.58e-05	1.48	1.22e-05	1.57	6.15e-06	1.69
160	1.59e-03	0.92	1.65e-05	1.47	4.01e-06	1.60	1.89e-06	1.70
10	1.95e-02	—/—	9.97e-04	—/—	2.91e-04	—/—	1.73e-04	—/—
20	1.05e-02	0.89/0.97	3.25e-04	1.62/1.62	1.15e-04	1.35/1.45	6.29e-05	1.46/1.56
40	5.51e-03	0.93/0.93	1.19e-04	1.45/1.50	3.88e-05	1.56/1.61	1.86e-05	1.76/1.82
80	2.91e-03	0.92/0.95	4.27e-05	1.48/1.50	1.22e-05	1.67/1.71	6.50e-06	1.51/1.52
160	1.53e-03	0.93/0.95	1.54e-05	1.47/1.50	3.99e-06	1.61/1.65	1.89e-06	1.78/1.82
10	1.96e-02	—/—	1.08e-03	—/—	3.41e-04	—/—	2.07e-04	—/—
20	1.13e-02	0.82/0.83	3.98e-04	1.48/1.50	1.44e-04	1.28/1.30	6.97e-05	1.61/1.63
40	6.58e-03	0.80/0.82	1.61e-04	1.34/1.37	4.77e-05	1.64/1.67	2.45e-05	1.55/1.58
80	3.93e-03	0.77/0.78	6.91e-05	1.26/1.29	1.97e-05	1.31/1.34	1.00e-05	1.32/1.36
160	2.35e-03	0.76/0.78	2.85e-05	1.31/1.34	7.51e-06	1.43/1.46	3.81e-06	1.44/1.47

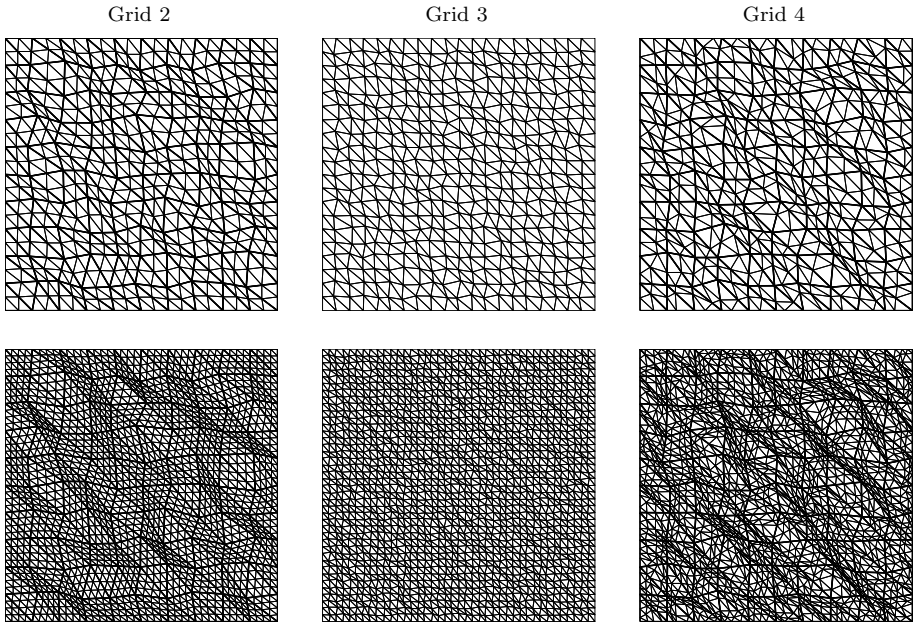


FIGURE 1. Two refinement levels for Grids 2, 3 and 4.

For the two rough grids, Grids 3 and 4, the errors increase on each refinement level. Grid 3 seems to yield marginally lower convergence rates than the formal order of the method when comparing to the average maximum mesh size. Note, however, that when using the maximum mesh size to compute rates, they fluctuate above and below the formal convergence rates, which may indicate that the average maximum mesh size is a somewhat non-conclusive mesh size to compare convergence against. This may be explained by the nature of random grid perturbations and the impact this has on the numerically calculated solutions on each refinement level. A long cell edge may impact the shapes of the neighbouring triangles and may yield larger errors at nodes associated with these.

Grid 4 experiences loss of convergence orders; this is observed when comparing to both the average maximum mesh size and the maximum mesh size. This may be explained by the diminishing grid quality such perturbations lead to. In Figure 2, the histograms for the mesh sizes of the triangular grids have been plotted for both Grids 3 and 4 for the grid size $N = 80$. Grid 3 has a normal distribution of the measured mesh size h for each element, whereas the mesh distribution in Grid 4 is skewed with a long tail in the interval corresponding to triangular grid cells with longer edges. Note that by the definition of the grid perturbation, the first refinement level of Grid 4 is a normal distribution. As the grids are refined, the mesh distribution becomes more and more skewed, with a significant tail to the right. Because of this kind of distribution (short triangle edges combined with longer triangle edges), the mesh quality diminishes as the grid is refined, which introduces an opposite effect to the pure

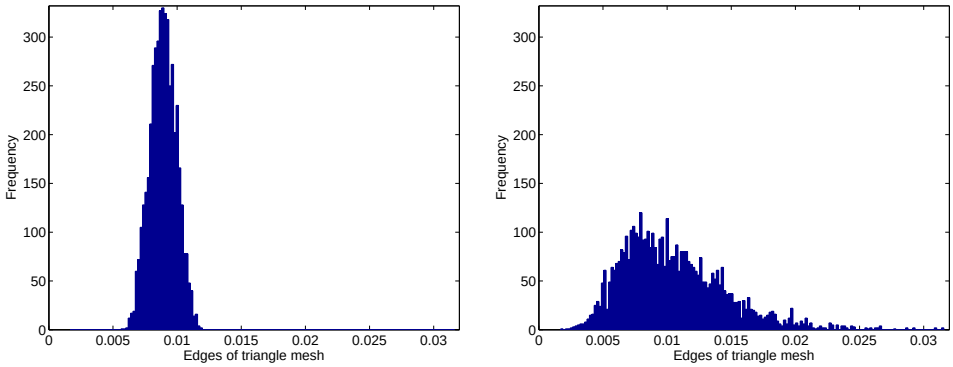


FIGURE 2. Histogram of edges of triangle meshes for Grid 3 (left) and Grid 4 (right).

reduction of triangle edge sizes the other grids experience. The observed order of dG(n) is significantly lower than $n+1$ for the Grid 4 refinement and also seems to decrease as the grids are refined, indicating a stronger effect of the skewed mesh distribution for increasing numbers of grid cells. Note also that the decay increases with the order of the basis functions.

These results should be compared with the theoretical results obtained in [6] and [1], where the convergence of the pressure equation is studied for general permeability description and irregular geometry. When a transformation to a computational space is performed for the pressure equation on general quadrilateral grids with general permeability, the evaluation of a quantity which may be viewed as *the computational space permeability* depending on the Piola mapping, becomes important. Different evaluations of the computational space permeability may have a very different behaviour on rough grids, and convergence may be lost entirely for rough grids that do not handle this evaluation properly.

In the next example, we consider a case with strongly heterogeneous media properties.

Case 2 (A Fluvial Medium). Consider a 2D quarter five-spot case with permeability and porosity data from Layer 77 of Model 2 in the 10th SPE Comparative Solution Project [2]. This layer contains sharp contrasts in permeability (and porosity) between the low-permeable background and a set of intertwined high-permeable channels. The strongly heterogeneous structure in this permeability field is shown in the upper plot in Figure 3; the permeability variation is up to eleven orders of magnitude. The right column in the figure shows the computed time-of-flights on triangular elements for dG(n), $n = 0, 1, 3$, and 5. For comparison, the left column shows the corresponding solutions using the dG scheme on rectangular elements, see [10]. The grid size is 220×60 for the rectangular grid, while the triangular grid is created by dividing each rectangular element into two triangles. The plots were created by sampling the polynomial patches in 10×10 uniformly distributed points inside each rectangular elements. In the

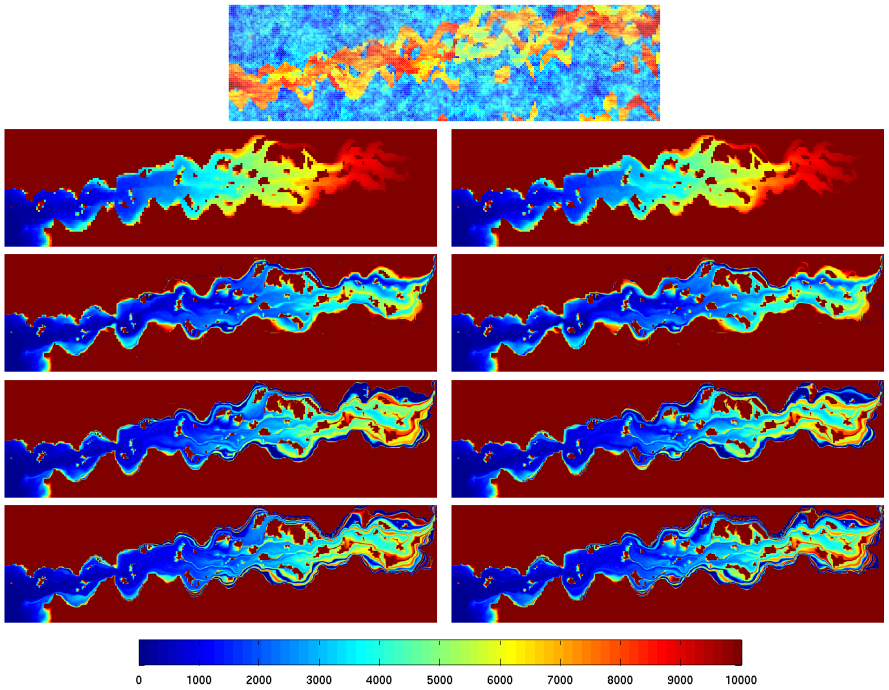


FIGURE 3. Time-of-flights for Layer 77 from the SPE10 test case computed using dG(n), $n = 0, 1, 3$, and 5 on the original rectangular grid (left column) and on a triangular grid (right column) created by splitting each rectangular cell in two.

TABLE 3. The relative L_1-errors of the computed time-of-flights for different vertical cross sections. See Figure 4 for the computed time-of-flights.

x	dG(0)	dG(1)	dG(2)	dG(3)	dG(4)	dG(5)
55	2.0719e-01	1.8535e-01	1.7421e-10	1.5946e-01	1.3955e-01	1.2579e-01
110	4.3507e-01	3.3024e-01	2.9480e-01	2.5810e-01	2.2044e-01	2.0217e-01
165	6.1523e-01	5.2704e-01	5.0089e-01	4.7045e-01	4.3190e-01	3.8929e-01
220	3.2763e-01	1.8338e-01	1.6834e-01	1.4444e-01	1.3316e-01	1.2459e-01

visual norm, the accuracy is approximately the same on the triangular and on the rectangular grid.

For this case we investigate the computed time-of-flights for four different vertical cross sections at $x = 55, 110, 165, 220$. The red graphs in Figure 4 show the dG(5) solution for the different cross sections and the blue graphs show the solution obtained by back-tracing approximately 8 000 streamlines. The corresponding relative L_1-errors for time-of-flights computed in different cross sections are present in Table 4. Altogether, we observe that strong heterogeneities in the permeability field influence the accuracy of the computed time-of-flights.

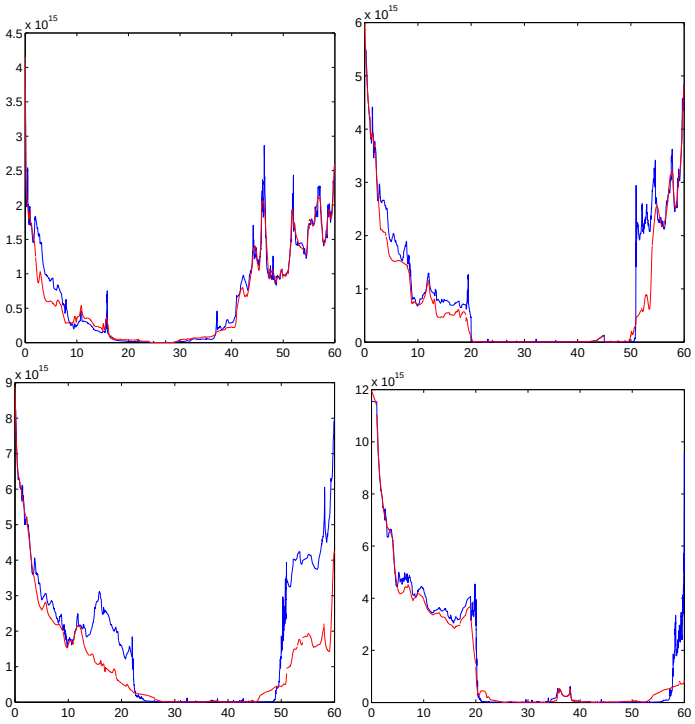


FIGURE 4. The computed time-of-flights along vertical cross sections at $x = 55, 110, 165, 220$. The red graphs give the dG(5) solution and the blue graphs give the solutions computed using approximately 8 000 streamlines.

Case 3 (Discrete Fracture Model). In this example, we consider a case with three high-permeable fractures inside the unit square. We impose no-flow boundaries at bottom and top, inflow at the left boundary, and outflow at the right boundary. Two cases are considered with the fractures having a permeability of 10^3 and 10^5, respectively, relative to the homogeneous and isotropic background field. The aperture of the fracture is 10^{-4} length units.

We compare the computed time-of-flights on a triangular grid with 5 048 elements that are adapted around and along the fractures with results on a coarser grid with 437 elements, a finer grid with 23 463 elements, and a grid with 5 301 elements but without adaptivity. The four grids are depicted in the top row of Figure 5. The permeability ratio between matrix and the fractures is $1 : 10^5$. From the plots, we observe three qualitative tendencies: (i) for the same number of unknowns (see Table 4), the solution is better for the grid *without* adaptivity; (ii) increased polynomial order is more important than increased grid resolution; and (iii) the improvements obtained by using finer grid resolutions decays with the polynomial order of the scheme. Finally, we observe that all the dG solutions establish the qualitative structures of the flow pattern.

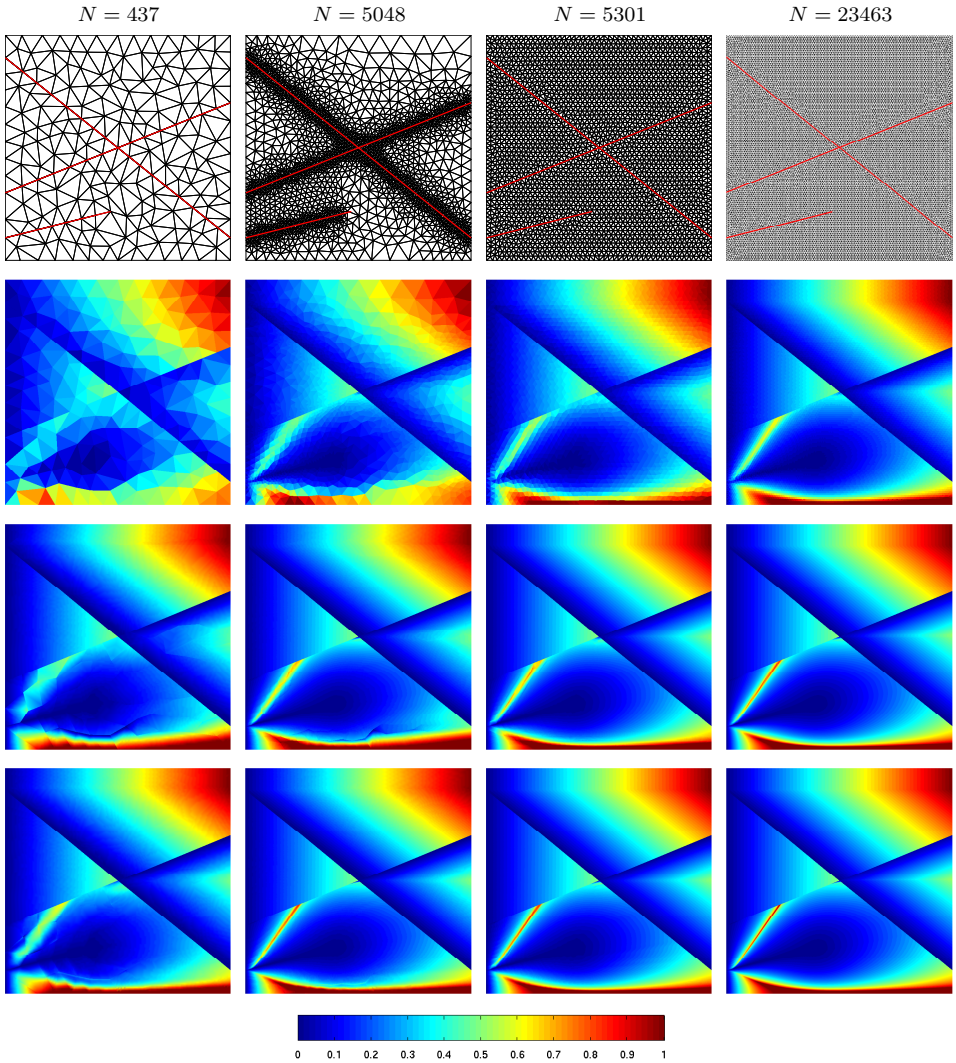


FIGURE 5. Case 3 with ratio between matrix and fracture permeability equal $1 : 10^5$. The rows present the computed time-of-flights using dG(0), dG(1), and dG(3).

TABLE 4. Degrees-of-freedom for different order and grid resolution.

N	dG(0)	dG(1)	dG(2)	dG(3)	dG(4)	dG(5)
437	437	1311	2622	4370	6555	9177
5048	5048	15144	30288	50480	75720	106008
5301	5301	15903	31806	53010	79515	111321
23463	23463	70389	140778	234630	351945	492723

Next, we consider the pointwise accuracy at the outflow boundary compared with a highly resolved solution computed by back-tracing approximately 16 000 streamlines. Table 5 shows the discrete relative L_1-errors for the time-of-flight

TABLE 5. Discrete relative L_1-errors in time-of-flight (upper half) and mass flow (lower half) at the outflow boundary for Case 3 with ratio K_m:K_f between the matrix and fracture permeability. The solutions are compared with solutions computed by tracing approximately 16 000 streamlines.

K_m:K_f	N	dG(0)	dG(1)	dG(2)	dG(3)	dG(4)	dG(5)
$1:10^3$	437	9.1575e-02	3.5080e-02	1.3979e-02	8.4314e-03	6.2851e-03	5.2484e-03
	5048	7.9320e-02	2.0271e-02	7.3588e-03	5.4863e-03	3.9867e-03	3.1843e-03
	5301	7.0833e-02	1.9132e-02	7.2066e-03	5.4397e-03	4.1306e-03	3.7196e-03
	23463	6.4046e-02	1.4980e-02	6.6697e-03	4.4689e-03	3.4315e-03	2.6675e-03
$1:10^5$	437	1.2607e-01	4.6378e-02	6.7521e-02	1.6426e-02	6.9459e-02	4.2250e-02
	5048	9.4916e-02	3.7634e-02	6.3852e-02	1.5811e-02	6.8247e-02	4.3420e-02
	5301	6.1381e-02	1.3697e-02	5.3138e-02	9.5423e-03	6.1411e-02	3.5569e-02
	23463	3.9014e-02	1.2143e-02	5.6907e-02	1.5259e-02	1.6766e-02	6.2730e-03
$1:10^3$	437	1.1240e-00	5.1969e-01	1.7987e-01	1.1910e-01	7.5825e-02	5.8945e-02
	5048	9.7077e-01	2.9547e-01	1.1434e-01	6.8977e-02	4.9809e-02	3.8912e-02
	5301	8.3155e-01	2.8433e-01	1.0473e-01	6.8073e-02	4.7003e-02	4.0898e-02
	23463	7.6335e-01	2.2220e-01	1.1358e-01	6.7778e-02	4.9492e-02	4.0935e-02
$1:10^5$	437	1.1102e-00	1.0192e-00	9.0023e-01	9.0149e-01	8.7923e-01	8.1843e-01
	5048	1.0885e-00	1.0394e-00	9.2523e-01	9.1535e-01	9.2921e-01	8.5429e-01
	5301	1.0577e-00	1.0305e-00	9.1703e-01	9.1194e-01	9.1910e-01	8.4378e-01
	23463	1.0514e-00	1.0723e-00	1.0027e-00	9.4507e-01	9.2882e-01	7.5827e-01

and the mass flux across the outflow boundary. Figure 6 shows the time-of-flight at the boundary for permeability ratio $1:10^3$. Similarly, Figure 7 shows the tracer production curve (average tracer concentration at the outflow boundary versus time) that results from injecting a tracer slug in the time interval $t \in [0, 0.05]$. As above, we observe that high polynomial order is more important than high grid resolution. In particular, Figure 7 shows that using dG(3) gives the same qualitative structures for all grid resolutions, whereas dG(0) fails to compute the correct tracer production on all grids. We also observe from Table 5 that the error increases with increasing ratio between the matrix and fracture permeability. This observation agrees with the results in [10].

When increasing the grid resolution in the example above, the grid inside the thin fractures only increased resolution in the longitudinal direction. For the simple Cartesian grids studied in [5], we observed that it was more important to increase the grid resolution in the *latitudinal* direction of the fractures to accurately resolve sharp transitions in time-of-flight arising when the flow changes from matrix to fracture and vice versa. In the next example, we therefore also consider refinement in the latitudinal direction of the fractures.

Case 4 (Latitudinal Refinement in Fractures). Consider a unit square with flow from left to right and no-flow boundaries at bottom and top. The fracture network consists of five horizontal fractures and a skew vertical fracture extending from top to bottom. The aperture of the fractures is 10^{-4} unit lengths and the permeability ratio is $1:10^5$. Figure 8 shows time-of-flight computed with dG(n) for $n = 0$, 1, and 3. The upper row shows the time-of-flights computed on a grid where each fracture is represented with one rectangular element divided into two triangle elements in the latitudinal direction. The lower row shows the

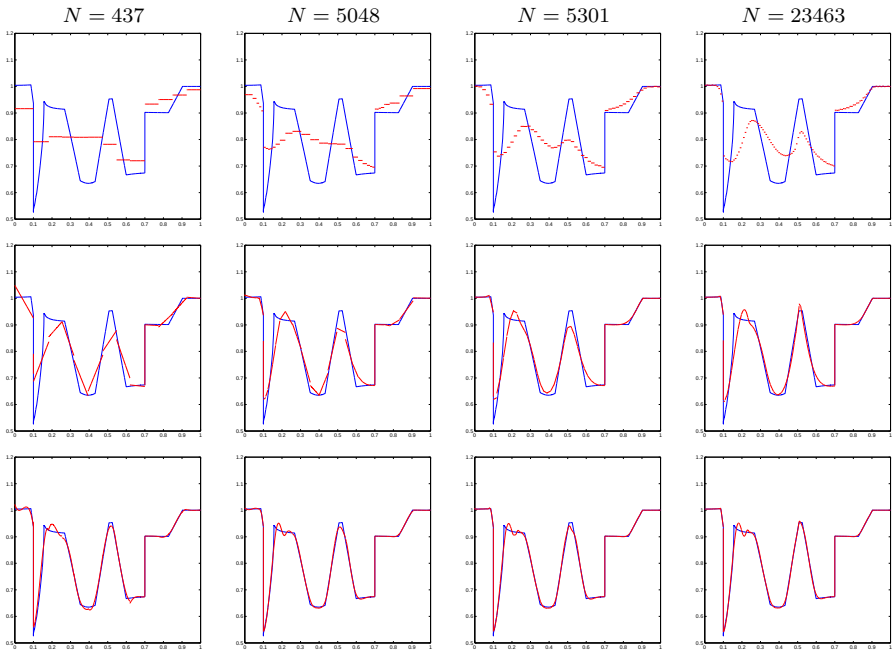


FIGURE 6. Time-of-flight for Case 3 with ratio between matrix and fracture permeability equal $1 : 10^3$. The red graphs show the dG(n) solutions for $n = 0$, 1, and 3 (from top to bottom) and the blue graphs are solutions computed by tracing approximately 16 000 streamlines.

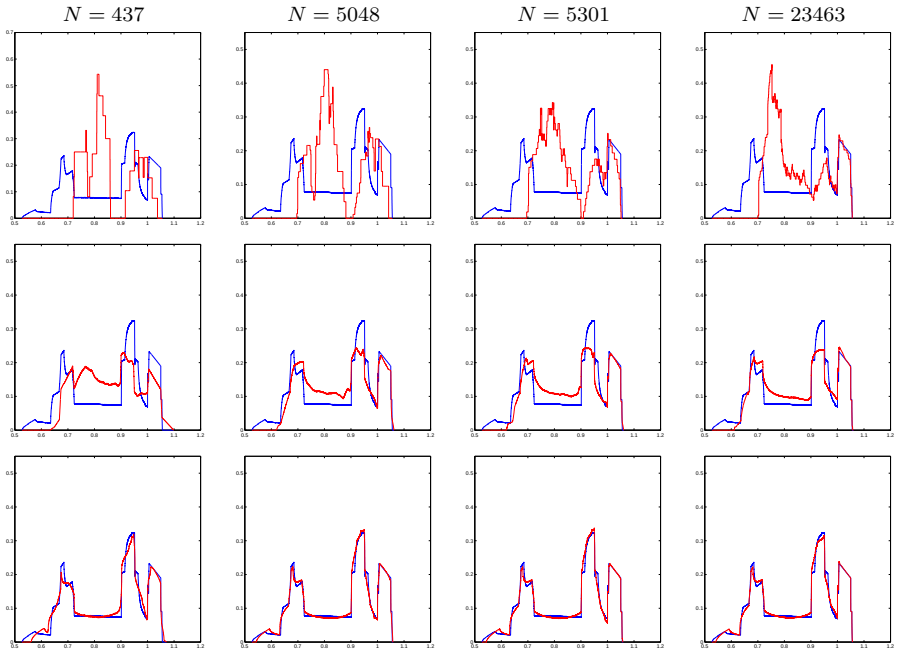


FIGURE 7. Average tracer concentration over the outflow boundary as a function of time for the simulations shown in Figure 6.

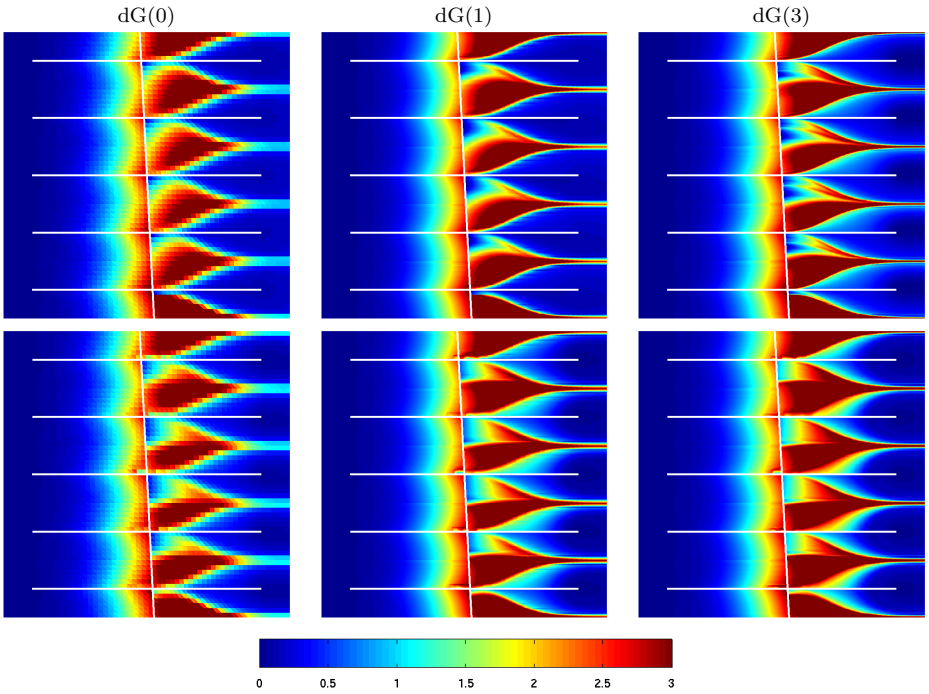


FIGURE 8. Time-of-flight for Case 4 with ratio between matrix and fracture permeability equal $1 : 10^5$. The upper row shows the results using no latitudinal grid refinement, while the lower row shows the solutions for a refinement with eight rectangular (divided in sixteen triangular) elements in the latitudinal direction of the fractures. The distribution of fractures is depicted as white lines.

solutions with eight rectangular elements (sixteen triangular elements) in the latitudinal direction.

Table 6 reports the time-of-flights and mass flow computed at the outflow boundary for the permeability ratios $1 : 10^3$ and $1 : 10^5$. We compare the computed time-of-flights with a reference solution obtained by back-tracking streamlines from uniformly distributed points inside each element at the outflow boundary for a refined grid. With one exception, the errors decrease when refining the grid in the latitudinal direction inside the fractures. These results agree with the results in [5], where we observed the importance of sufficient latitudinal grid resolution to correctly capture large spatial variations inside the fractures. Capturing these variations is necessary since the time-of-flight is an integrated quantity that is strongly affected globally by local discretization errors.

Criteria to guide the choice between single and multi-continua (porosity) formulations in site-specific applications are not easily defined. A simple method is to consider by measuring the (outflow) concentration of some species present in a reservoir model during some predefined time interval. Here we consider breakthrough curves resulting from the injection of a tracer slug/pulse. If the

TABLE 6. Discrete relative L_1-errors in time-of-flight (upper half) and mass flow (lower half) at the outflow boundary for Case 4 using M elements across the fractures. The solutions are compared to solutions computed by tracing approximately 4 000 streamlines.

K_m:K_f	M	dG(0)	dG(1)	dG(2)	dG(3)	dG(4)	dG(5)
$1:10^3$	1 (2)	1.5772e-01	6.7707e-02	3.8263e-02	2.8205e-02	2.0743e-02	1.9426e-02
	8 (16)	1.5260e-01	5.7438e-02	3.2888e-02	2.3043e-02	1.7009e-02	1.6180e-02
$1:10^5$	1 (2)	7.7989e-01	5.8586e-01	3.9670e-01	3.5901e-01	3.2230e-01	2.9336e-01
	8 (16)	7.4916e-01	4.6600e-01	2.1120e-01	1.5121e-01	1.3237e-01	1.0724e-01
$1:10^3$	1 (2)	1.1149e-00	7.5882e-01	6.4524e-01	5.5346e-01	4.3380e-01	4.3060e-01
	8 (16)	1.1992e-00	5.9138e-01	3.8487e-01	2.9969e-01	2.3405e-01	2.4164e-01
$1:10^5$	1 (2)	1.4736e-00	1.2881e-00	6.2057e-01	6.3658e-01	3.9217e-01	3.7723e-01
	8 (16)	8.3593e-01	2.6399e-01	2.5384e-01	2.6574e-01	1.8156e-01	1.9228e-01

curve has two peaks, there are two distinct transport mechanisms corresponding to flow in fractures and matrix. On the other hand, if the curve has a single peak, the medium can be modelled using a discrete model.

In the next example we demonstrate that our dG scheme can provide a fast and easy method for evaluating tracer-breakthrough curves for flow in fractured porous media.

Case 5 (Discrete model versus multi-continua model). Consider the same test example as in Case 4, now with grid refinement in the latitudinal direction of the fractures. We measure the concentration over the outflow boundaries. The tracer is a pulse injection for a short time; in our case for $t \in [0, 0.05]$. Figure 9 shows the mass flow over the outflow boundary computed using dG(n) for $n = 0$, 1, and 4 compared with a highly resolved streamline simulation on a refined grid. For permeability ratio $1 : 10^3$, shown in the upper row, we obtain multiple peaks, where the first peak represents the tracer going through the fractures and the next peak represent tracer flowing through the lower-permeable rock matrix. The results for permeability ratio $1 : 10^5$ only has a single peak that breaks through very early, meaning that the tracer goes straight through the fractures and that this is the predominant transport mechanism. Thus, for the first case it is necessary to use a multi-continua model, while for the second case a discrete model may be appropriate.

The previous example demonstrates that the time-of-flight formalism can be used to find breakthrough curves for highly resolved small-scale models where fractures are represented explicitly as volumetric object. This may be used as a guide when choosing an appropriate conceptual model to be used on a larger scale. Hence, our method may serve as a technical guide for the choice of single and multi-continua formulation in fractured rocks.

In the next example, we demonstrate how our dG methods can be used to delineate the reservoir by determining swept and drainage volumes and well connectivities. To this end, we will solve the steady tracer-concentration equation (3) rather than the time-of-flight equation. The stationary tracer equation describes the steady concentration arising if we continuously inject tracer at a certain part of the inflow boundary. Hence, if the tracer concentration is positive at a point, the point is influenced by the part of the inflow boundary on

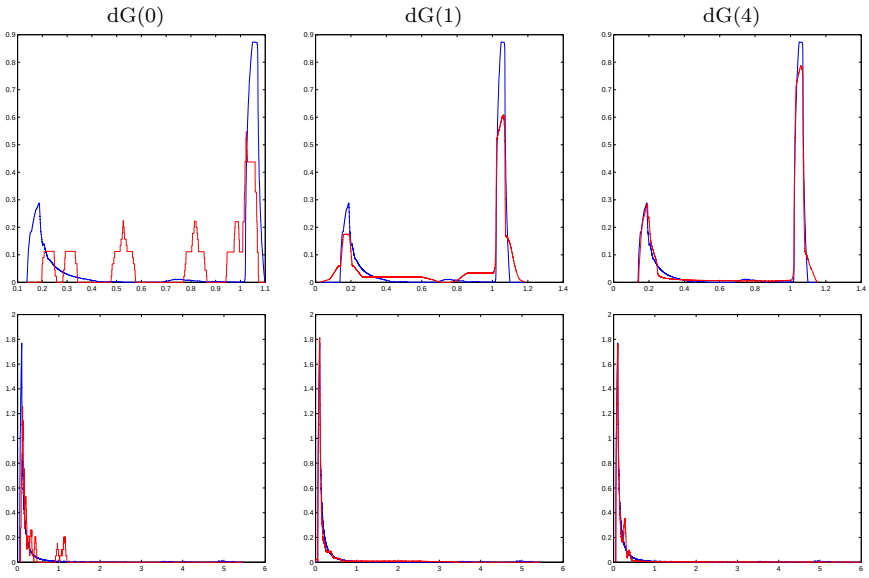


FIGURE 9. Computed mass flow over the outflow boundary with permeability ratio between matrix and the fractures equal $1 : 10^3$ in the upper row and $1 : 10^5$ in the lower row. The red graphs give the dG(n) solutions and the blue graphs give solutions computed by back-tracing approximately 4 000 streamlines.

which we inject tracer. To partition a reservoir, we define the swept/drained volumes as the volumes having a concentration larger than 0.5. Notice in particular that due to the efficient sequential solution procedure, computing each drainage volume is a single-sweep computations that can be performed with high order accuracy and modest demands on storage and computing power.

Case 6 (Approximation of Stationary Tracer Distribution). We consider the stationary tracer distribution for a fractured reservoir shown in Figure 10. The permeability ratio between the the matrix and the fractures is $1 : 10^5$ and the the aperture of the fractures is 10^{-4} length units. Four injection wells are located in each corner and two production wells are located inside the domain. Figure 10 shows the tracer distribution for each injector computed using basis functions of increasing order. The different swept areas are shown in different colours/shading, and the boundaries between the swept areas correspond to the 0.5 contour of the different tracer concentrations. The figure illustrates that low-order approximations do, in general, provide sufficient accuracy to delineate the reservoir. This was also observed in [10].

Figure 11 shows the stationary tracer distribution in a more challenging fractured reservoir. The distribution of the fractures is depicted in the figures, and the permeability ratio between matrix and the fractures is $1 : 10^3$. One producer is located in the lower left corner and three injectors are located in the three other corners. Each row in Figure 11 shows the swept areas for the

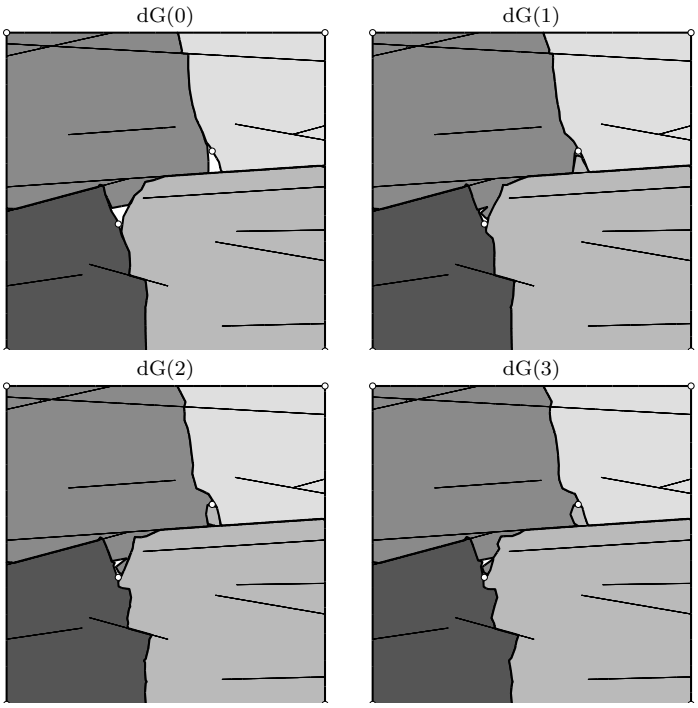


FIGURE 10. Stationary tracer distribution for four injectors placed one in each corner and two producers are placed inside the domain.

three different injectors computed using dG(0) in the first column and dG(2) in the second column.

5. Final Remarks

We have previously shown that the combination of a discontinuous Galerkin spatial discretization and a optimal ordering of cells is a robust, accurate, and efficient numerical approach for the solution of incompressible flow of fluids in porous media, see [10, 8]. For multiphase flow [8, 7] and single-phase flow in media with mild heterogeneity, our experience indicates that a low-order dG method (the standard upwind method, dG(0), or the second-order dG(1)) is sufficient to accurately capture the fluid transport. For single-phase flow in strongly heterogeneous media, one may need to increase the order to accurately capture integrated quantities like time-of-flight and steady tracer concentration.

For fractured media, all our results have so far been presented for Cartesian grids. However, explicit modelling of complex fracture networks will give rise to very complex structures, and using unstructured triangular (tetrahedral) grids, at least locally, may be necessary to accurately model realistic cases. In this paper, we have made the first steps toward extending our dG methodology to unstructured grids by presenting results for triangular elements in 2D, from which the extension to tetrahedral elements in 3D is straightforward.

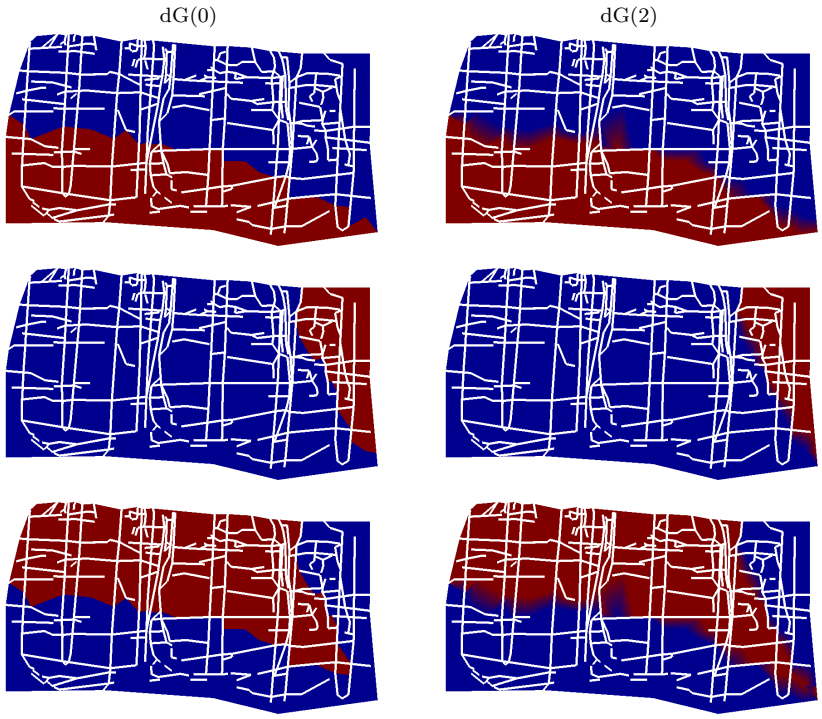


FIGURE 11. Tracer distribution for three injectors placed in three of the corners and one producers placed in the lower left corner.

There are two features with our methodology that may prove very useful when attacking complex 3D models. First of all, using a discontinuous Galerkin discretization in combination with an upwind flux, we localise the degrees-of-freedom (and their assembly) and simplify the coupling of different element types. Secondly, we use an optimal ordering of the unknowns that allows us to compute the solutions in an element-by-element fashion. This method is quite general and applies to *any* grid where the inter-element dependence can be described by a graph

For triangular grids, the dG method is convergent for smooth solutions, but loses accuracy near discontinuities. Case 1 in Section 4 shows how the roughness of randomly perturbed grids impact the accuracy, leading to reduced convergence rates for rough grids. Considering polynomial degree versus grid resolution, some of the other examples indicate that increasing the order of the basis functions is more important than increasing the grid resolution (provided the flux is resolved with sufficient accuracy). Our experience is that a dG discretisation of sufficiently high order is a relative robust alternative to streamlines that performs well in a wide range of realistic cases. However, high permeability contrasts reduce the accuracy of the solution. This may be countermanded by introducing a sort of a slope limiter as used in [10], where we reduce the order of the basis functions and refine the grid in areas with high media contrasts. Finally, to accurately compute time-of-flight in fractured porous media, it is

important having a sufficient grid resolution in the latitudinal direction of the fractures. This is necessary since the time-of-flight is an integrated quantity that is very sensitive to small-scale variations in media properties and contains large spatial variation, in particular within and close to fractures.

We have also demonstrated how the framework can be used to compute accurate approximations to the stationary tracer distribution in a reservoir. Two test cases indicate that low-order approximations have sufficient accuracy to produce reasonable delineations of a reservoir volume.

Altogether, we have demonstrated that the dG schemes in most cases can accurately compute time-of-flight and stationary tracer distribution. These quantities are of practical importance for applications in petroleum reservoir simulation and groundwater modelling. For petroleum reservoir simulation, the time-of-flight gives the timelines in the reservoir, whereas computing the tracer distribution can determine the spatial regions swept or drained by a fluid from a source or a sink. Within groundwater applications, the evaluation of the time-of-flight may be a important tool to visualise the spreading of contaminants and to help understanding the different transport processes.

Acknowledgements

We would like to thank Håkon Hægland for the streamline simulations, and Rainer Helmig for useful feedback on fractured model issues. The research is founded in part by the Research Council of Norway through the GeoScale project with grant 158908/130.

References

[1] I. Aavatsmark, G. Eigestad, R. Klausen, M. Wheeler, and I. Yotov. Convergence of a symmetric MPFA method on quadrilateral grids. *Comput. Geosci.*, 11:333–345, 2007. doi:10.1007/s10596-007-9056-8.

[2] 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`.

[3] N. R. Council, editor. *Rock fractures and fluid flow*. National Academy Press, 1996.

[4] P. Dietrich, R. Helmig, M. Sauter, H. Hötzl, J. Köngeter, and G. Teutsch. *Flow and Transport in Fractured Porous Media*. Springer, 2005.

[5] B. Eikemo, I. Berre, H. K. Dahle, K.-A. Lie, and J. R. Natvig. A discontinouos Galerkin method for computing time-of-flight in discrete-fracture models. In P. J. Binning, P. K. Engesgaard, H. K. Dahle, G. F. Pinder, and W. G. Gray, editors, *Proceedings of the XVI International Conference on Computational Methods in Water Resources*, Copenhagen, Denmark, June 2006. `http://proceedings.cmwr-xvi.org/`.

[6] R. Klausen and R. Winther. Robust convergence of multi point flux approximation on rough grids. *Numer. Math.*, 104:317–337, 2006. doi:10.1007/s00211-006-0023-4.

[7] J. Natvig and K.-A. Lie. On efficient implicit upwind schemes. In *Proceedings of ECMOR XI*, Bergen, Norway, 8-11 September 2008. `http://folk.uio.no/kalie/papers/ecmorxi-ro.pdf`.

[8] J. Natvig and K.-A. Lie. Fast computation of multiphase flow in porous media by implicit discontinuous Galerkin schemes with optimal ordering of elements. *J. Comput. Phys.*, to appear. `http://folk.uio.no/kalie/papers/dg-multiphase.pdf`.

[9] J. Natvig, K.-A. Lie, and B. Eikemo. Fast solvers for flow in porous media based on discontinuous Galerkin methods and optimal reordering. In P. J. Binning, P. K. Engesgaard, H. K. Dahle, G. F. Pinder, and W. G. Gray, editors, *Proceedings of the XVI International*

Conference on Computational Methods in Water Resources, Copenhagen, Denmark, June 2006. http://proceedings.cmwr-xvi.org/.

[10] J. Natvig, K.-A. Lie, B. Eikemo, and I. Berre. An efficient discontinuous Galerkin method for advective transport in porous media. *Adv. Water Resour.*, 30(12):2424–2438, 2007.

[11] L. Neunhäuserer, A. Hemminger, and R. Helmig. Influence of fracture - matrix - interaction on flow and transport processes and the resulting effective parameters in fractured porous systems. In *Hydraulic Engineering for sustainable Water Resources Management at the Turn of the Millennium*, Graz, Austria, 22-27 August 1999. XXVIII IAHR Congress.

[12] V. Reichenberger, R. Helmig, H. Jakobs, P. Bastian, and J. Niessner. Complex gas-water processes in discrete fracture-matrix systems: Upscaling, mass-conservative discretization and efficient multilevel solution. Institut für Wasserbau, Universität Stuttgart, 2004.

(Birgitte Eikemo) University of Bergen, Dept. of Mathematics,
Johs. Brunsgt. 12, NO-5008 Bergen, Norway
E-mail address: birgitte@math.uib.no

(Knut–Andreas Lie) SINTEF ICT, Applied Mathematics,
P.O. Box 124, Blindern, NO–0314 Oslo, Norway
E-mail address: Knut-Andreas.Lie@sintef.no

(Knut–Andreas Lie) University of Bergen, Dept. of Mathematics,
Johs. Brunsgt. 12, NO-5008 Bergen, Norway

(Geir Terje Eigestad) University of Bergen, Dept. of Mathematics,
Johs. Brunsgt. 12, NO-5008 Bergen, Norway
E-mail address: geirte@math.uib.no

(Helge K. Dahle) University of Bergen, Dept. of Mathematics,
Johs. Brunsgt. 12, NO-5008 Bergen, Norway
E-mail address: helge.dahle@math.uib.no