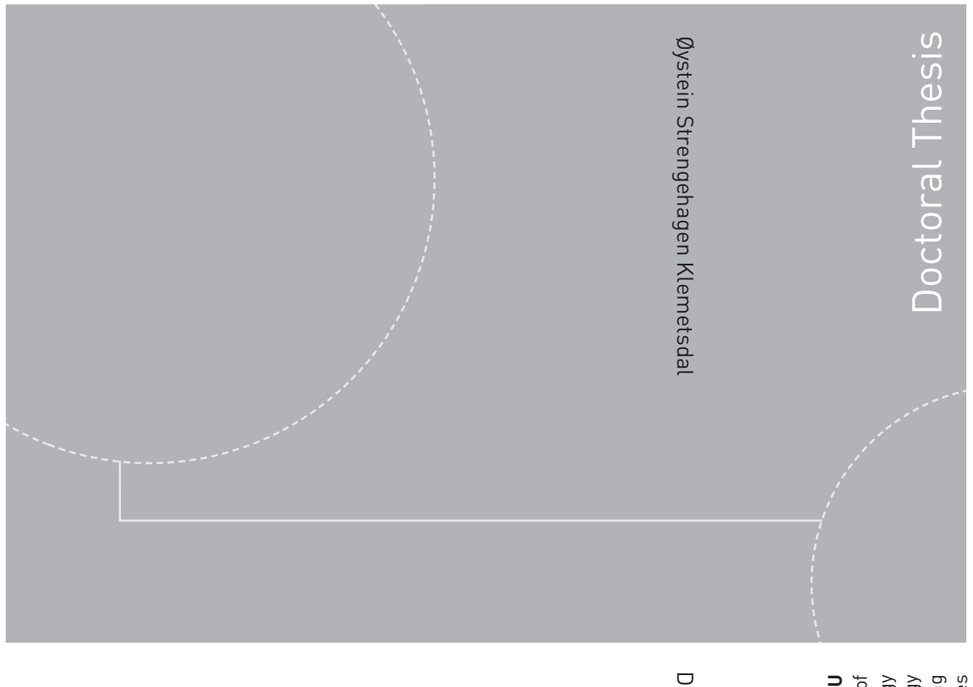
ISBN 978-82-326-4346-2 (printed version)
ISBN 978-82-326-4347-9 (electronic version)
ISSN 1503-8181




Doctoral theses at NTNU, 2019:374

Øystein Strengehagen Klemetsdal

# Efficient Solvers for Field-Scale Simulation of Flow and Transport in Porous Media

**NTNU**
Norwegian University of
Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Mathematical Sciences

NTNU
Norwegian University of
Science and Technology

Doctoral theses at NTNU, 2019:374

NTNU

NTNU
Norwegian University of
Science and Technology

Øystein Strengehagen Klemetsdal

# Efficient Solvers for Field-Scale Simulation of Flow and Transport in Porous Media

Thesis for the degree of Philosophiae Doctor

Trondheim, November 2019

Norwegian University of Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Mathematical Sciences

NTNU
Norwegian University of
Science and Technology

**NTNU**
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology
and Electrical Engineering
Department of Mathematical Sciences



ISBN 978-82-326-4346-2 (printed version)
ISBN 978-82-326-4347-9 (electronic version)
ISSN 1503-8181

Doctoral theses at NTNU, 2019:374

Printed by Skipnes Kommunikasjon as

## Abstract

The first part of the thesis serves as an introduction to modelling of flow and transport in porous media from the perspective of reservoir simulation. We discuss generation of computational grids and discretization strategies, and look at some of the main components that make up a reservoir simulator: discrete operators, linearization strategies, linear solvers, and nonlinear solvers. The second part consists of nine papers that present novel work on efficient solution strategies applicable to field-scale reservoir simulation. These strategies are primarily based on sequential splitting of the governing equations into a flow and transport subproblem.

In the first two papers, we consider novel techniques for generation of high-quality Voronoi grids that conform to 2D surface constraints representing geological features like faults and fractures, and 1D line constraints representing complex well trajectories. The flow problem has a strong elliptic character, and the first and third paper also study consistent discretization methods for elliptic (Poisson-type) flow equations with emphasis on numerical errors and computational efficiency. Flow equations can be solved efficiently using so-called multiscale methods. The fourth paper describes a dynamically adaptive, iterative multiscale method with improved convergence that uses additional coarse partitions to target features in the geological model and/or adapt to dynamic changes in the flow field.

The transport problem typically has a strong hyperbolic character. In the fifth and sixth papers, we use this to devise robust adaptive damping strategies for Newton's method that delineate different contraction regions in the residual function. In the seventh paper, we also exploit unidirectional flow properties to develop a local nonlinear solver that topologically sorts the grid cells according to the flow direction. By traversing the grid cells in this order, the nonlinear transport subproblems can be solved locally in a highly efficient manner. We apply this to accelerate the simulation of the widely used black-oil equations, discretized by first- and second-order discontinuous Galerkin methods. The eighth paper extends the method to compositional problems, and combines discontinuous Galerkin methods with a simple adaptive dynamic coarsening strategy to further accelerate the simulation of fine-scale transport equations. The last paper present a robust and efficient framework for adaptive dynamic coarsening, and combines this with our local nonlinear solvers.

# Contents

# Preface

This thesis is submitted in partial fulfillment of the requirements for the degree of philosophiae doctor (PhD) at the Norwegian University of Science and Technology (NTNU), Trondheim, Norway.

# Acknowledgments

I am grateful to many people for their support, scientific and otherwise, during the three years of working with this PhD thesis. First of all, I would like to thank my supervisors: Prof. Knut-Andreas Lie for giving me the opportunity to pursue a PhD, for shearing his insights on reservoir simulation, and for tremendous help and guidance in the writing process; and Dr. Olav Møyner for a productive collaboration, and for sharing his know-how on scientific software development. Fruitful discussions with Lie and Møyner have lead to many of the main results in this thesis. I would also like to thank my supervisor Prof. Xavier Raynuad for interesting discussions on discretizations, and for his keen and contagious interest in mathematics. Moreover, my supervisor Prof. Helge Holden at NTNU has always been supportive and helpful.

Furthermore, I would like to thank my coworkers at the Computational Geosciences group at SINTEF for welcoming me as part of the team, for sharing their impressive amount of knowledge, and for the numerous coffee breaks to either continue or escape from in-depth scientific discussions. Moreover, I would like to thank the people from SINTEF, the University of Geneva, and the University of Bergen whom I have co-authored papers with. Thanks also to Prof. Hamdi Tchelepi for hosting me at Stanford University from March – April 2019, which gave valuable new perspectives on my work. I am also grateful for financial support from the Research Council of Norway under grant no. 244361.

Finally, I would like to thank my family and friends for supporting me, and cheering me on. A special thanks to my parents for always encouraging me to do whatever I wanted in life, as long as I do my very best. And, of course, a big thanks to my fiancée, Jennie, for her limitless patience, support, and kindness. I love you.

Øystein Strengehagen Klemetsdal
Oslo, August 2019

# List of Papers

## Papers included in thesis

I: **Unstructured Gridding and Consistent Discretizations for Reservoirs With Faults and Complex Wells**
Øystein S. Klemetsdal, Runar Lie Berge, Knut-Andreas Lie, Halvor Møll Nilsen, Olav Møyner
*In proceedings of the 2017 SPE Reservoir Simulation Conference, Montgomery, Texas, USA*
*DOI: 10.2118/182666-MS*

II: **Unstructured Voronoi Grids Conforming to Lower-dimensional Objects**
Runar Lie Berge, Øystein S. Klemetsdal, Knut-Andreas Lie
*Computational Geosciences, volume 23, issue 1, pp. 169–188, 2019*
*DOI: 10.1007/s10596-018-9790-0*

III: **A Comparison of Consistent Discretizations for Elliptic Poisson-Type Problems on Unstructured Polyhedral Grids**
Øystein S. Klemetsdal, Olav Møyner, Xavier Raynaud, Knut-Andreas Lie
*Manuscript in preparation, 2019*

IV: **Accelerating Multiscale Simulation of Complex Geomodels by Use of Dynamically Adapted Basis Functions**
Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie
*Computational Geosciences, published ahead of print, 2019*
*DOI: 10.1007/s10596-019-9827-z*

V: **Non-linear Newton Solver for a Polymer Two-phase System Using Interface-localized Trust Regions**
Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie
*In proceedings of the 19th European Symposium on Improved Oil Recovery,*

*2017, Stavanger, Norway*
*DOI: 10.3997/2214-4609.201700356*

VI: **Robust Nonlinear Newton Solver with Adaptive Interface-Localized Trust Regions**
Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie
*SPE Journal, volume 24, issue 4, pp. 1576–1594, 2019*
*DOI: 10.2118/195682-PA*

VII: **Efficient Reordered Nonlinear Gauss-Seidel Solvers With Higher Order For Black-Oil Models**
Øystein S. Klemetsdal, Atgeirr Flø Rasmussen, Olav Møyner, Knut-Andreas Lie
*Computational Geosciences, published ahead of print, 2019*
*DOI: 10.1007/s10596-019-09844-5*

VIII: **Implicit High-resolution Compositional Simulation with Optimal Ordering of Unknowns and Adaptive Spatial Refinement**
Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie
*In proceedings of the 2019 SPE Reservoir Simulation Conference, Galveston, Texas, USA*
*DOI: 10.2118/193934-MS*

IX: **Dynamic Coarsening and Local Reordered Nonlinear Solvers for Simulating Transport in Porous Media**
Øystein S. Klemetsdal, Knut-Andreas Lie
*Manuscript in preparation, 2019*

## Papers not included in thesis

Four more papers were written during the period as a PhD student. Conference Papers X and XI were later reworked into peer-reviewed journal publications that are included in the thesis. Paper XII concerns simulation of $CO_2$ storage with mobility control, and Paper XIII modelling and simulation of geothermal energy storage, both outside the scope of this thesis.

X: **Use of Dynamically Adapted Basis Functions to Accelerate Multiscale Simulation of Complex Geomodels**
Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie
*In proceedings of the 16th European Conference on the Mathematics of Oil Recovery, 2018, Barcelona, Spain*
*DOI: 10.3997/2214-4609.201802251*

XI: **Nonlinear Gauss-Seidel solvers with Higher Order for Black-oil Models**
Øystein S. Klemetsdal, Atgeirr Flø Rasmussen, Olav Møyner, Knut-Andreas Lie
*In proceedings of the 16th European Conference on the Mathematics of Oil Recovery, 2018, Barcelona, Spain*
*DOI: 10.3997/2214-4609.201802130*

XII: **$CO_2$ Storage with Mobility Control**
Alv-Arne Grimstad, Per Eirik Strand Bergmo, Halvor Møll Nilsen, Øystein S. Klemetsdal
*In proceedings of the 14th Greenhouse Gas Control Technologies Conference Melbourne, 2018, Melbourne, Australia*

XIII: **Evaluating the thermal performance and perturbation of a high-temperature aquifer thermal energy storage (HT-ATES) system in the Greater Geneva Basin, Switzerland**
Marine Collignong, Øystein S. Klemetsdal, Olav Møyner, Marion Alcanié, Antonio Pio Rinaldi, Halvor Møll Nilsen, Matteo Lupi
*Geothermics, submitted for publication, 2019*

# Part I

# Background

# Chapter 1

# Introduction

This thesis focuses on the development of robust, efficient and accurate methods for solving partial differential equations describing flow and transport in porous media. In mathematics, solutions methods for such equations are typically studied with a generic perspective, with emphasis on properties like existence and uniqueness of solutions, stability, convergence proofs, error estimates, etc. This necessarily requires that we make simplifying assumptions that unfortunately tend to make the results less interesting from an application point of view.

The work herein therefore aims to develop new methods that can readily be implemented in commercial and open-source software used for practical engineering applications. Such methods must meet the special requirements of the application in order to be accepted and adopted for industry use, even if these may seem peculiar from a generic point of view. Methods should of course be mathematically sound, but theoretical proofs of asymptotic behavior and superior performance on idealized cases is much less important than performance observed on realistic model instances and for practically achievable discrete resolutions. Moreover, understanding the underlying physics of the model equations is crucial in order to devise new numerical methods that gain efficiency by utilizing causality, sparsity, and weak(er) couplings within the systems of model equations.

The introductory part of this thesis therefore tries to give an overview of simulation of flow in porous media, as seen from the perspective of a mathematician developing new numerical methods. We also try to outline challenges that are common for many porous media applications, as well as issues that are specific to reservoir simulation.

## 1.1 Flow in porous media

Research on flow in porous media has a wide range of applications. In Norway, the first that perhaps comes to mind is recovery of hydrocarbons from petroleum reservoirs, either for use as fossil fuels, or as key components in petrochemical products such as lubricants, fertilizers, and plastics. Understanding the physical processes involved in flow of oil and gas in a petroleum reservoir is crucial to ensure that the recovery is economically viable, and at the same time safe. The latter is particularly important when the reservoir is located underneath the seabed, where leakage of hydrocarbons into the sea may have widespread negative consequences for aquatic organisms.

Whereas the world is likely to depend on petroleum resources for many years to come, a shift towards renewable energy is considered important to mitigate the effects of climate change. Flow in porous media fits naturally into this development: Geothermal energy is an example of a renewable resource that can be exploited by injecting and extracting fluids in geothermally heated subsurface aquifers [11]. Capturing $CO_2$ from industrial processes for storage in subsurface reservoirs has been identified as a key technology in order to reduce emission to an acceptable level [86]. Moreover, approximately 30% of the world's freshwater is contained in underground aquifers, which amounts to more than 90% of our readily available freshwater resources [168]. A large amount of the world's population relies on exploitation of these resources for drinking water. A thorough understanding of subsurface processes is crucial for all of these technologies: The full potential of a geothermal energy system can only be exploited by understanding the dynamics of heat transfer in the aquifer; storing $CO_2$ underground only makes sense if we can make sure it will stay there for hundreds or thousands of years; whereas avoiding contamination and ensuring sustainable management of the world's freshwater reserves are imperative for the world's population.

## 1.2 Petroleum reservoirs

Sedimentary rocks are formed by the accumulation and deposition of particles in bodies of water covering the Earth's surface. This process is extremely slow – the buildup of particles amounts to a few vertical centimeters every hundred years. The particles originate either from erosion of mineral rocks, or from dead aquatic organisms, giving rise to different layers, or *strata*: Layers of mineral particles form clastic rocks such as sand-

stones and mudrocks, whereas skeletons from aquatic organisms break up into particles that form limestone and other carbonate rocks. The transition from layers of particles to a solid rock formation is called lithification, and is driven by pressure from the overlying body of water and heating from the center of the Earth. Different layers with different rock properties are referred to as beds. Geological activity forms fractures and faults across these beds, and stretching and breaking of the continental crust leads to the formation of sedimentary basins. It is in such basins we usually find petroleum reservoirs.

Alongside this geological activity, the organic constituents of the aquatic organisms, in particular plankton and algae, also deposit on the seabed. These are eventually trapped within the pores of the subsurface rock, where the high temperature from the center of the Earth and pressure of the overburden act together as a pressure cooker. The result is kerogen, which over the course of approximately one to a hundred million years forms hydrocarbons in the form of crude oil and natural gas. As the overburden pressure increases due to further deposition, the hydrocarbons migrate upwards in the porous rock formation, where they either escape, or get trapped under a layer of confining of rock. Figure 1.1 illustrates a reservoir with a fault and a pinch-out formed by geological activity. Brine, oil and gas are trapped in a layer of permeable sandstone, delineated by impermeable mudrock.

To extract hydrocarbons from a petroleum reservoir, wells are drilled into the porous rock formation. During primary production, the reservoir pressure is often so high that it pushes oil and gas to the surface through these wells. In some cases, a pump can be used at the surface to increase the pressure difference. However, as the reservoir pressure decreases, the flow of oil and gas out of the reservoir will decline and eventually stop. In a second phase, called secondary recovery, water or gas is injected to increase the reservoir pressure and further drive the oil and gas towards the producing wells.

Water injection, or water flooding, is particularly popular, especially in off-shore reservoirs where water is abundant. A common problem with this, is that injected water tends to follow high-flow zones through the reservoir to the nearest production well. In addition, the high viscosity of oil relative to water forces the injected water to form viscous fingers in the oil. As a result, water injection typically leaves large parts of the reservoir unswept, and consequently a large fraction of the reservoir oil and gas remains unexploited. A telling example is petroleum production on the Norwegian continental shelf, where oil production from secondary recov-

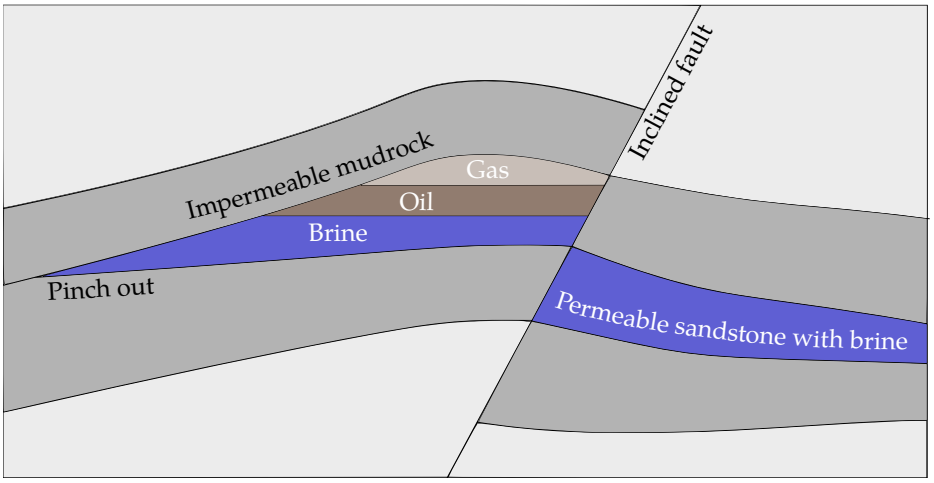



**Figure 1.1:** Reservoir with fault and a pinch out. Impermeable layers of mudrock delineate a region with permeable brine-filled sandstone. The upper mudrock layer acts as a cap under which oil and gas are trapped.

ery started to decline in 2000. The average recovery factor (i.e., recoverable amount of hydrocarbons) in this region is 46% [111], and approximately half of the petroleum resources is assumed to remain in the reservoir with standard water injection [182]. In comparison, this is still well above the world average of 22%.

To exploit as much as possible of the remaining hydrocarbon resources in a reservoir, one may turn to *tertiary recovery*. Commonly referred to as enhanced oil recovery (EOR) techniques, this includes injecting chemical and biological substances that alter the properties of the reservoir oil and gas, and the porous rock formation itself. Examples include injection of polymers to increase water viscosity, solvent gas to dilute and mobilize heavy and viscous oil, and surfactants to reduce the interface tension between the hydrocarbons and the porous rock.

EOR techniques have proven very effective for on-shore reservoirs. In off-shore applications, on the other hand, factors such as long distances between injection and production wells, limited storage space on the production facilities, and severe consequences of leakage into the sea, make the economic potential of EOR highly uncertain. To justify the expenses related to EOR techniques, it is therefore necessary to demonstrate its efficacy on a field scale through *reservoir simulation*.

## 1.3 Reservoir simulation

The ultimate goal of reservoir simulation is to predict how the aqueous and hydrocarbon fluids will flow through the reservoir with different injection strategies to determine an optimal recovery plan. In this setting, an optimal plan is typically the one that maximizes net present value, given a number of constraints such as regulatory requirements, drilling and injection costs, and environmental concerns. Schematically, a reservoir simulation model can be said to consist of three parts: (i) a geological model describing the subsurface reservoir; (ii) a flow model that describes how fluids flow in the porous media, including conservation laws and constitutive relations; and (iii) models for the wells and production facilities. Figure 1.2 illustrates the different parts. Together, these are the ingredients we use to translate the physical problem of flow in porous media into computer code.

Developing robust, efficient, and accurate reservoir simulation technology involves a number of disciplines, including geology, physics, mathematics, computer science, and of course petroleum engineering. The mathematical challenges, which are the focus of this thesis, are numerous. For example, generating suitable computational grids is seriously complicated by the fact that such grids need to adapt to orders-of-magnitude local variations in rock properties, complex geological layering with erosions and interbedding, structures such as fractures or faults, and possibly a large number of wells with deviated trajectories. Computational grids therefore tend to have skewed and irregular cell geometries, which makes it challenging to construct robust discretizations for the systems of partial differential equations that govern the fluid flow. The linear systems arising from these discretizations tend to be ill-conditioned for the same reasons. Moreover, the governing equations are typically coupled and nonlinear. This is particularly true for EOR simulations, for which the equations are often strongly coupled and highly nonlinear, and numerical diffusion tends to mask the true EOR effects in field-scale simulations.

This thesis covers a number of topics related to reservoir simulation, all with emphasis on fast an accurate simulation of EOR. Gridding, i.e., the process of constructing a computational grid to describe the reservoir and discretize the flow equations, is discussed in Paper I and Paper II; spatial discretizations for the pressure part of the model equations are covered in Paper I and Paper III, whereas spatial discretizations for the transport of fluid phases and components are discussed in Paper VII and Paper VIII; linear solvers are considered in Paper IV; and nonlinear solvers are discussed in Papers V – IX. The following chapters give a brief introduction to these different topics.

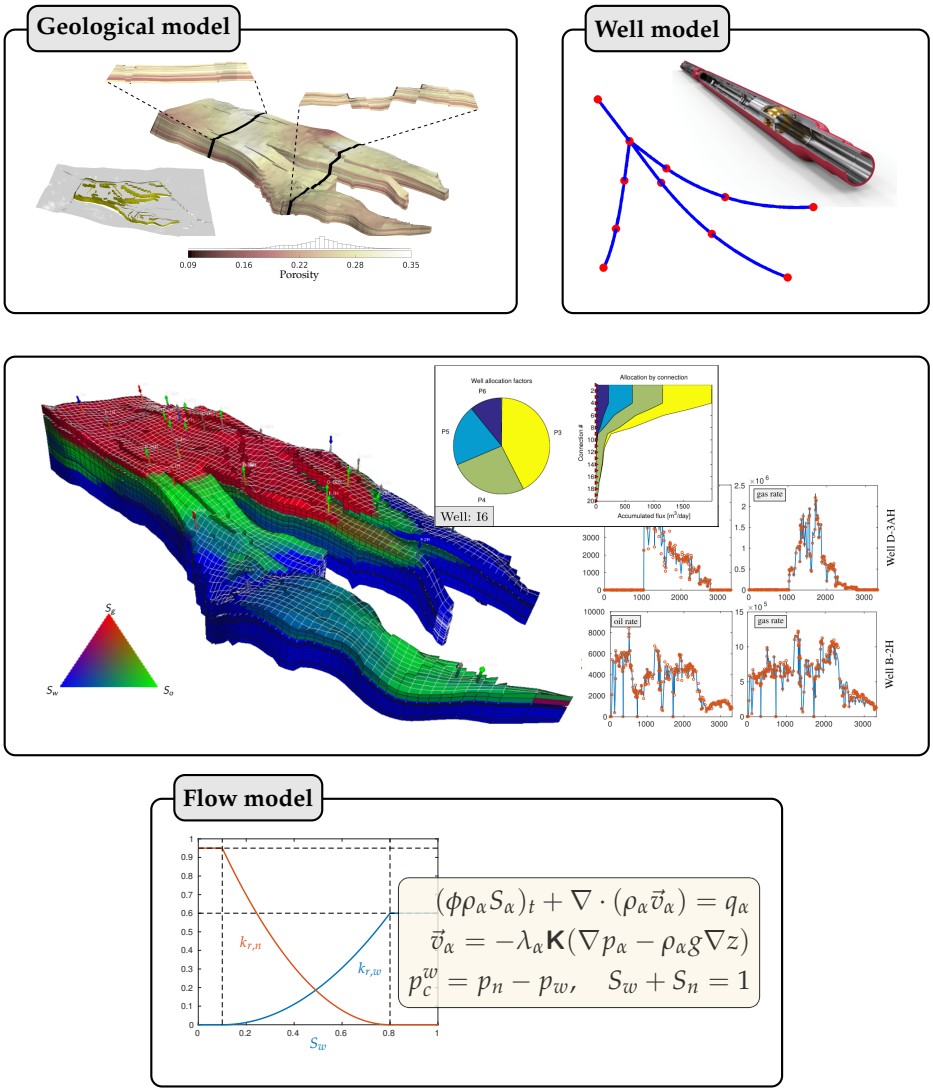



**Figure 1.2:** Components of a reservoir simulator: the geological model, the flow model, and the model for wells and production facilities. The middle figure and the well model contains figures from [111] and from Paper VII.

# Chapter 2

# Physical and Mathematical Aspects of Flow in Porous Media

Flow in porous media is a truly multiscale phenomenon in the sense that physical processes occur over a wide range of spatial and temporal scales: Properties of the subsurface rock vary with several orders of magnitude over a wide range of length scales. Moreover, pressure changes propagate quickly over large distances, at least in conventional reservoirs, whereas movement of fluids is typically slow and mostly local. In this chapter, we discuss some of the basic physical and mathematical aspects associated with flow in porous media. These represent core building blocks for our simulation model: the geological model, the flow model, and the well model.

## 2.1 Single-phase flow

Transport of fluids is governed by conservation of mass. Consider a fluid with density $\rho(\boldsymbol{x}, t)$ that moves with velocity $\vec{v}(\boldsymbol{x}, t)$ at a given time $t$ and coordinate $\boldsymbol{x}$, illustrated in Figure 2.1. In this figure, $\Omega(t)$ is a volume that encloses the same fluid particles at all times. The mass in this volume is

$$\int_{\Omega(t)} \rho(\boldsymbol{x}, t) \, \mathrm{d}\boldsymbol{x}.$$

The coordinate $\boldsymbol{x}(t)$ can be thought of as the position of a particle advected by the fluid. Since the volume follows the same fluid particles at all times,

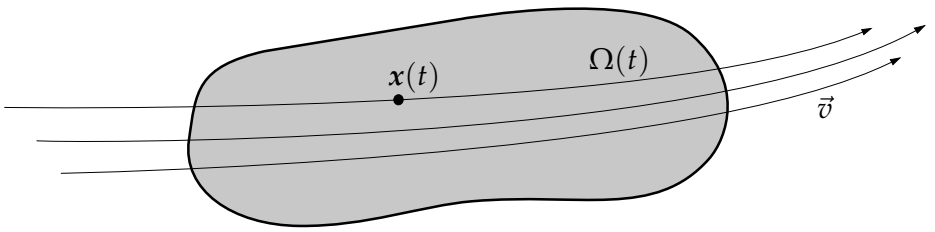



**Figure 2.1:** The volume $\Omega(t)$ encloses the same fluid particles at all times. The coordinate $\boldsymbol{x}(t)$ can be thought of as a passive particle that moves with the fluid at a velocity $\vec{v}$.

we must have that

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)}\rho(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x}=0. \tag{2.1}$$

By a change of coordinates to a stationary reference frame and assuming the integrand is continuous, we can show that

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)}\rho(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x}=\int_{\Omega(t)}\left(\frac{\mathrm{d}\rho}{\mathrm{d}t}+\rho(\nabla\cdot\vec{v})\right)\,\mathrm{d}\boldsymbol{x}.$$

Since $\boldsymbol{x}$ is the position of a passive particle advected by the fluid, we have that $\partial_t\boldsymbol{x}=\vec{v}$, so that the first term in the integrand can be written

$$\frac{\mathrm{d}\rho}{\mathrm{d}t}=\frac{\partial\rho}{\partial t}+\frac{\partial\boldsymbol{x}}{\partial t}\cdot\nabla\rho=\frac{\partial\rho}{\partial t}+\vec{v}\cdot\nabla\rho.$$

Inserting this into the equation above gives us the famous Reynolds transport theorem:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)}\rho(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x}=\int_{\Omega(t)}\left(\frac{\partial\rho}{\partial t}+\nabla\cdot(\rho\vec{v})\right)\,\mathrm{d}\boldsymbol{x}. \tag{2.2}$$

We made no assumptions on the volume $\Omega(t)$ apart from that it follows the same fluid particles. Assuming the integrand in the right-hand side of (2.2) is positive at some point, it follows from continuity that there exists a domain $\Omega(t)$ so that the integral (2.1) is also positive. This is a contradiction, and we can conclude that the integrand must be zero everywhere. Therefore, conservation of mass is described by the following partial differential equation (PDE)

$$\frac{\partial\rho}{\partial t}+\nabla\cdot(\rho\vec{v})=\rho q. \tag{2.3}$$

Here, we have added the term $\rho q$ to the right-hand side to model fluid sources/sinks, where $q$ is the flow rate of the source/sink. Equation (2.3) is an example of a transport equation, which generally takes the form

$$\partial_t \mathcal{M} + \nabla \cdot \vec{\mathcal{F}} = \mathcal{Q}. \tag{2.4}$$

For (2.3), these terms read

$$\mathcal{M} = \rho, \quad \vec{\mathcal{F}} = \rho \vec{v}, \quad \mathcal{Q} = \rho q, \tag{2.5}$$

and we refer to them as the mass ($\mathcal{M}$), flux ($\vec{\mathcal{F}}$), and source/sink ($\mathcal{Q}$) terms, respectively. Note that if the fluid is incompressible, $\partial_t \mathcal{M} = 0$ and $\vec{\mathcal{F}} = \rho \nabla \cdot \vec{v}$, so that (2.3) divided by the constant density $\rho$ simplifies to

$$\nabla \cdot \vec{v} = q.$$

### Representative elementary volumes and Darcy's law

In addition to boundary and initial conditions, we need an equation describing the fluid velocity $\vec{v}$. As explained in Chapter 1, porous rock formations typically consist of small pores between grains of solid. The diameter of these pores depends on the rock type, and may vary from a few millimeters and down to as little as a few nanometers. As an example, accurate representation of a sandstone sample of 11.39 mm$^3$ from [55] requires a resolution of $300^3$ voxels. Direct numerical simulation of flow at the pore scale in this small sample would require in the order of 10 million unknowns. Pore-scale simulation of a real reservoir is therefore far out of reach for modern-day computers. Moreover, obtaining pore-scale data for an entire subsurface reservoir is impossible in practical applications. Instead, we consider a domain much larger than a typical pore, and let $\vec{v}$ be the apparent macroscopic velocity of a set of particles moving through the domain with the fluid, defined as the effective volumetric discharge per cross-sectional area. Figure 2.2 illustrates such a domain, which we refer to as a *representative elementary volume* (REV).

The apparent macroscopic velocity is also known as the *Darcy velocity*, named after the French hydraulic engineer Henry Darcy. By considering gravity-driven flow through a container filled with sand, he derived what we refer to as Darcy's law, which in modern notation reads

$$\vec{v} = -\frac{1}{\mu}\mathbf{K}(\nabla p - \rho g \nabla z). \tag{2.6}$$

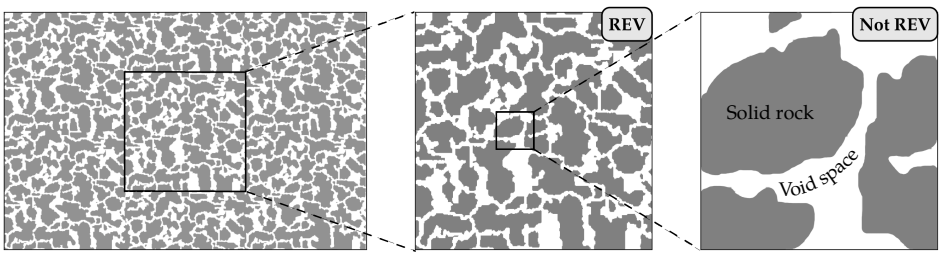



**Figure 2.2:** Pore network in a porous rock, with two levels of zoom. The first has a fairly uniform spatial distribution of pores and is therefore a representative elementary volume (REV) for the rock. The second, however, is so small that the resulting pore distribution is very heterogeneous, and is therefore not an REV. The pore network is inspired by an example from [125].

Here, $p$ is the fluid pressure, $\rho$ is the density, $g$ the gravitational acceleration, and $z$ the vertical coordinate. The absolute permeability $\mathbf{K}$ is a proportionality factor that measures the REV's ability to transmit a fluid. Generally, this is a tensor, and may be written in matrix form as

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{xx} & \mathbf{K}_{xy} & \mathbf{K}_{xz} \\ \mathbf{K}_{yx} & \mathbf{K}_{yy} & \mathbf{K}_{yz} \\ \mathbf{K}_{zx} & \mathbf{K}_{zy} & \mathbf{K}_{zz} \end{bmatrix}. \tag{2.7}$$

If all diagonal terms are equal, $\mathbf{K}_{xx} = \mathbf{K}_{yy} = \mathbf{K}_{zz}$, and all off-diagonal terms are zero, the absolute permeability is *isotropic*. In the general case, we refer to it as *anisotropic*. The permeability tensor will always be symmetric and positive definite [111, Chapter 2], and its SI unit is $\mathrm{m}^2$. It is, however, commonly given in units darcy (d), which equals approximately $0.987 \times 10^{-12}\mathrm{m}^2$, or in millidarcy (md), which is more appropriate in reservoir simulation. We also associate with the REV an effective porosity $\phi$ that measures the fraction of the bulk volume that can store and transmit fluids. Note that the term Darcy *velocity* is misleading: Since the macroscopic Darcy velocity is defined as volume per area occupied by the fluid per time, the macroscopic fluid velocity equals the Darcy velocity divided by $\phi$. Figure 2.3 illustrates permeability and porosity for the REV in Figure 2.2.

With these concepts, we can state a closed system of equations that governs single-phase flow in porous media:

$$\begin{aligned} (\phi\rho)_t + \nabla \cdot (\rho\vec{v}) &= \rho q \\ \vec{v} &= -\frac{1}{\mu}\mathbf{K}(\nabla p - \rho g \nabla z). \end{aligned} \tag{2.8}$$

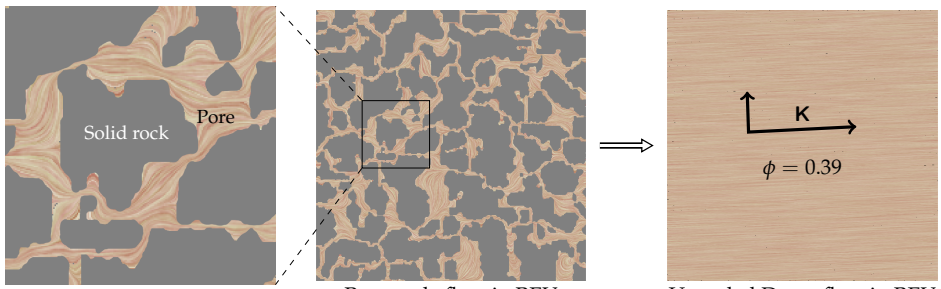



**Figure 2.3:** The pore network induces a complex flow pattern through the porous medium, here visualized by brown streamlines, but can be represented as a much simpler upscaled macroscopic *Darcy velocity* by considering a representative elementary volume (REV) with an effective permeability $\mathbf{K}$ and porosity $\phi$. The pore network is inspired by an example from [125].

Note that we have multiplied $\rho$ with $\phi$, since the fluid can only occupy the pore volume. This means that in this case, $\mathcal{M} = \phi\rho$. To close the system, we must also specify boundary conditions and the initial state. For incompressible flow, $\partial_t \mathcal{M} = 0$, and we may write (2.8) (omitting gravity effects) as

$$-\nabla \cdot \frac{1}{\mu}\mathbf{K}\nabla p = q, \tag{2.9}$$

which we recognize as the Poisson equation with variable coefficient.

## 2.2 Multiphase and multicomponent flow

Single-phase flow equations are hardly ever sufficient to model real hydrocarbon reservoirs, which generally comprise a multitude of hydrocarbon components that may exist in e.g., liquid ($\ell$) or vapor ($v$) phase, as well as smaller amounts of non-hydrocarbon natural gases such as nitrogen, hydrogen sulfide, and helium. The formation also contains brine water, represented as an aqueous ($a$) phase. Hydrocarbon components may dissolve into the brine, and the water may evaporate into steam at high temperatures. Moreover, unless the reservoir is chemically inert, the liquid and vapor phases can undergo geochemical reactions with each other and/or the rock surface. To further complicate matters, enhanced oil recovery involves injection of chemical species like $CO_2$, polymers, surfactants, solvents, and other chemical or biological agents that alter properties of the rock formation, the resident fluids, and their interaction with each other, and possibly

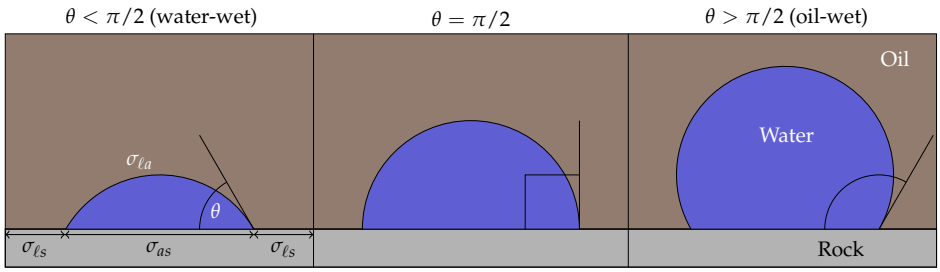



**Figure 2.4:** The wettability of a system is determined by the fluid/rock contact angle $\theta$. The different quantities in (2.10) are shown to the left, with water as the aqueous phase $(a)$, and oil as the liquid phase $(\ell)$.

also create new fluid phases (gels, foams, (micro)emulsions) and precipitation of solids. In other words, a proper multiphase and multicomponent flow formulation is usually required to properly describe flow and transport in subsurface rocks. In this section, we first consider a few multiphase concepts, before briefly discussing multicomponent formulations.

## Wettability and capillary pressure

If we consider two immiscible fluid phases at the pore scale, the cohesive forces between two molecules within the same phase will be greater than the adhesive forces between two molecules in different phases. Effectively, molecules at the interface between the two phases form a surface. The force required to change the shape of this surface is measured by the surface tension. Similarly, molecules in a fluid phase will also experience adhesive forces toward the rock surface. The phase with stronger attraction to the rock surface is called the *wetting phase*, whereas the other phase is referred to as the *non-wetting phase*. Young's equation (see, e.g. [30]) gives the relationship between the liquid–water interface energy $\sigma_{\ell a}$, the contact angle $\theta$, and the interface energy $\sigma_{\alpha s}$ between phase $\alpha$ and the solid:

$$\sigma_{\ell a} \cos\theta = \sigma_{\ell s} - \sigma_{as}. \tag{2.10}$$

This contact angle is a measure of the wettability of the system: The system is called water-wet if $0 \leq \theta < \pi/2$, and oil-wet if $\pi/2 \leq \theta < \pi$. The quantities are shown in Figure 2.4.

The surface tension leads to a pressure difference across the interface between the two phases:

$$p_c = p_n - p_w,$$

where subscripts $w$ and $n$ denote the wetting- and non-wetting phase, respectively. This pressure difference is called the *capillary pressure*. In reservoir simulation, capillary pressure is typically modelled as a function of the fractions of pore volume that the different phases occupy. We refer to these fractions as *saturations*, and denote the saturation of phase $\alpha$ by $S_\alpha$. We assume that reservoir fluids exist in either the liquid, vapor or aqueous phase, and that they occupy the entire accessible pore volume, so that

$$S_a + S_\ell + S_v = 1.$$

During the formation of a petroleum reservoir, hydrocarbons migrate upwards in the porous rock formation and displace resident water. Because of the surface tension, the migrating oil and gas will typically not displace all the in-place water, but leave behind an irreducible water saturation $S_{ar}$. Generally, the residual saturation $S_{\alpha r}$ of phase $\alpha$ is a function of the pore-space topology, and the phases' affinity to wet the rock relative to the other phases.

For three-phase fluid systems, is common to model capillary pressure separately for the liquid-water and liquid-vapor contact as

$$p_c^a(S_a) = p_\ell - p_a, \quad p_c^v(S_v) = p_v - p_\ell.$$

Capillary pressure plays a key role in allowing or preventing hydrocarbons to migrate into a porous rock formation and displace the in-place formation water. However, capillary forces are often weak compared to viscous and buoyancy forces for conventional reservoirs. Figure 2.5 shows capillary pressure curves for the Norne oil and gas field [87] and the SPE9 Benchmark [96]. Notice that capillary pressure $p_c^v$ between the liquid and vapor phase increases with increasing vapor saturation $S_v$, since the liquid phase is the wetting phase of the two.

### Relative permeability

From (2.6), we see that the rate at which a single fluid phase flows through a porous medium is governed by the permeability $\mathbf{K}$. Intuitively, interfacial tension between fluid phases means that the flow rate of one phase should also depend on the other phases present in the pores. This is modelled by introducing *relative permeabilities* (see, e.g., [136]):

$$\mathbf{K}_\alpha^{\text{eff}} = k_{r,\alpha}\mathbf{K}.$$

In most cases, the interfacial tension between different phases results in a reduction in the effective permeability, so that $k_{r,\alpha} \in [0, 1]$. In cases where

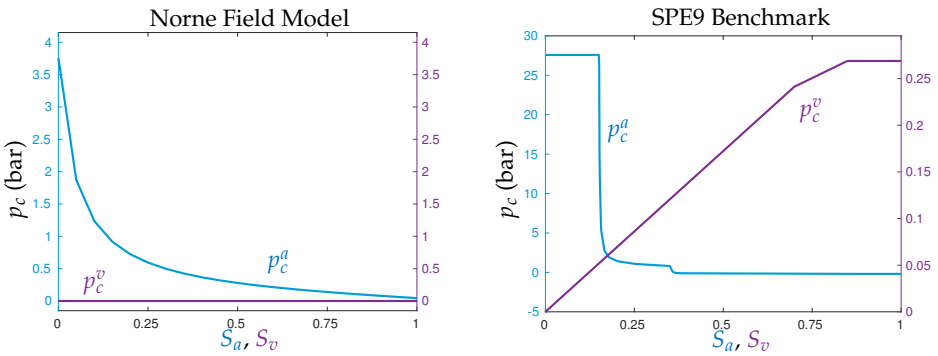



**Figure 2.5:** Capillary pressure curves for the Norne oil and gas field (left), and the SPE9 benchmark (right). The liquid-vapor capillary pressure is identically equal to zero in the Norne fluid model, and slightly greater than zero in the SPE9 fluid model (notice the different y-axes). Note that the capillary pressure curve in the Norne fluid model is used match real data from the reservoir by scaling it differently in different parts of the model.

the non-wetting phase is more viscous, however, the wetting phase may act as a lubricant, so that $k_{r,n}$ may take on values greater than one [147].

Since $S_{\alpha r}$ represents the minimum saturation at which phase $\alpha$ becomes mobile, it follows that $k_{r,\alpha}(S_\alpha) = 0$ for $S_\alpha \leq S_{\alpha r}$. The relative permeability of a phase is typically assumed to be a monotone function of saturation. Whereas relative permeability can be experimentally measured by analyzing core samples, analytic functions remain popular due to significant uncertainties related to experimental data [30]. The most common is the Brooks-Corey power law

$$k_{r,\alpha}(S_\alpha) = k_{r,\alpha}^{\max} \left( \frac{S_\alpha - S_{\alpha r}}{1 - \sum_{\beta \neq \alpha} S_{\beta r}} \right)^{n_\alpha}, \tag{2.11}$$

where exponent $n_\alpha$ and the maximum relative permeability $k_{r,\alpha}^{\max}$ are used to fit experimental data. The model gets its name from the similarity to the expressions derived by Brooks and Corey [35] for two-phase liquid-water flow. Figure 2.6 shows two examples of Brooks-Corey relative permeabilities with exponents $n_w = n_n = 2$, but with different residual saturations and maximum relative permeabilities.

Most models for three-phase relative permeability are based on combining relative permeability models for two-phase oil-water ($k_{r,\ell a}$) and oil-

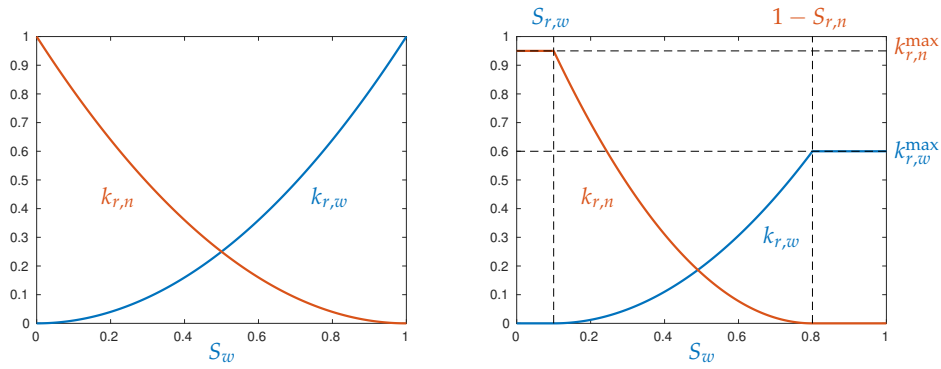



**Figure 2.6:** Brooks-Corey relative permeabilities with $n_w = n_n = 2$. Relative permeabilities in the right figure have residual saturations $S_{r,\alpha} > 0$ and maximum relative permeabilities $k_{r,\alpha}^{\max} < 1$. The different quantities are indicated in the figure.

gas ($k_{r,\ell v}$) systems, e.g.,

$$
\begin{aligned}
k_{r,\ell}(S_a, S_\ell, S_v) &= \omega_a(S_a, S_v) k_{r,\ell a}(S_\ell) + \omega_v(S_a, S_v) k_{r,\ell v}(S_\ell), \\
\omega_a(S_a, S_v) &= \frac{S_a - S_{ar}}{S_a + S_v - (S_{ar} + S_{vr})}, \\
\omega_v(S_a, S_v) &= 1 - \omega_a(S_a, S_v).
\end{aligned}
$$

See [16] for a review. Figure 2.7 shows three-phase relative permeabilities from the Norne field model and the SPE9 Benchmark.

## Hysteresis

The process when a non-wetting fluid migrates upwards into a porous rock formation and displaces the resident wetting phase is known as *drainage*. In an REV of this formation, the residual wetting-phase saturation will then equal $S_{wr}$. The reverse process, i.e., when the wetting phase displaces the non-wetting phase, is known as *imbibition*. During this process, some of the non-wetting phase tends to be trapped in small pores as a result of interfacial tension. In effect, the residual saturation of the non-wetting phase is different during drainage and imbibition. This, and other pore-scale phenomena such as variations in wettability between fluid phases, effectively means that the relative-permeability and capillary-pressure functions will be different during drainage and imbibition. This is known as *hysteresis*, and is a direct effect of that we are explaining microscopic and pore-scale effects through upscaled constitutive relations valid for REVs.

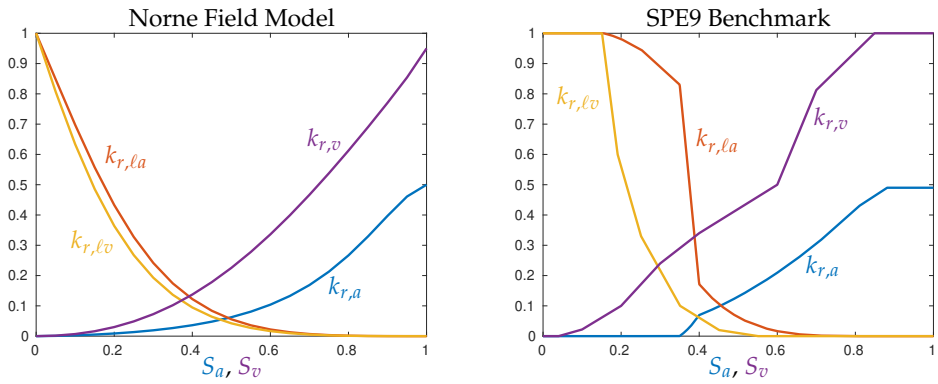



**Figure 2.7:** Relative permeabilities from one real and one realistic model. The left plot shows (unscaled) relative permeabilities from the Norne field model, whereas the right plot shows relative permeabilities from the SPE9 benchmark. Relative permeabilities for the Norne filed model have different endpoint scaling in different parts of the reservoir.

## Governing equations for multiphase flow

We have now introduced the basic concepts of multiphase flow in porous media, most notably permeability, porosity, and saturation. We commonly refer to these and related properties as *petrophysical properties*. With these concepts, we can state the governing equations for imisscible multiphase flow at the macro-scale:

$$\partial_t \mathcal{M}_\alpha + \nabla \cdot \vec{\mathcal{F}}_\alpha = \mathcal{Q}_\alpha, \quad \alpha = a, \ell, v, \tag{2.12}$$

where

$$\mathcal{M}_\alpha = \phi \rho_\alpha S_\alpha, \quad \vec{\mathcal{F}}_\alpha = \rho_\alpha \vec{v}_\alpha, \quad \mathcal{Q}_\alpha = \rho_\alpha q_\alpha, \tag{2.13}$$

and the macroscopic Darcy velocity is given by

$$\vec{v}_\alpha = -\lambda_\alpha \mathbf{K}(\nabla p_\alpha - \rho_\alpha g \nabla z), \quad \lambda_\alpha = \frac{k_{r,\alpha}}{\mu_\alpha}. \tag{2.14}$$

Here, we have introduced the *mobility* $\lambda_\alpha$ of phase $\alpha$. Finally, we have a set of closure relations for phase saturations and pressures,

$$S_a + S_\ell + S_v = 1, \quad p_c^\alpha = p_\ell - p_\alpha, \quad \alpha = a, v.$$

### *Compositional flow*

As mentioned in the beginning of this section, reservoir fluids usually consist of a number of hydrocarbon components that, at least in principle, can

be distributed among all three fluid phases. Compositional flow can be modeled by introducing phase mass fractions $X_{\alpha,c}$ that denote the fraction of component $c$ present in phase $\alpha$. If $M_{\alpha,c}$ is the mass of component $c$ in phase $\alpha$, the corresponding mass fraction is

$$X_{\alpha,c} = \frac{M_{\alpha,c}}{\sum_{i=1}^{m} M_{\alpha,i}},$$

where $m$ is the number of components. The mass, flux, and source terms for component $c$ in phase $\alpha$ read

$$\mathcal{M}_{\alpha,c} = \phi \rho_\alpha X_{\alpha,c} S_\alpha, \quad \vec{\mathcal{F}}_{\alpha,c} = \rho_\alpha X_{\alpha,c} \vec{v}_\alpha, \quad \mathcal{Q}_{\alpha,c} = \rho_\alpha X_{\alpha,c} q_\alpha.$$

The component mass fractions and phase saturations can be determined by imposing thermodynamic equilibrium, as predicted by a chosen equation of state. Since we have interphase mass transfer, the conserved quantity is now *component mass*

$$\partial_t \mathcal{M}_c + \nabla \cdot \vec{\mathcal{F}}_c = \mathcal{Q}_c,$$

where

$$\mathcal{M}_c = \sum_{\alpha=a,\ell,v} \mathcal{M}_{\alpha,c}, \quad \vec{\mathcal{F}}_c = \sum_{\alpha=a,\ell,v} \vec{\mathcal{F}}_{\alpha,c}, \quad \mathcal{Q}_c = \sum_{\alpha=a,\ell,v} \mathcal{Q}_{\alpha,c}.$$

With $m$ number of components, we get $m$ such equations. The mass fractions of all components in a phase must sum to unity, giving an additional closure relation for each phase,

$$\sum_{c=1}^{m} X_{\alpha,c} = 1, \quad \alpha = a, \ell, v.$$

For many applications, it may be sufficient to lump together components with similar properties into pseudo-components. A particularly popular model that does this is the *black-oil model*, which gathers the hydrocarbon molecules into two pseudo-components that appear as oil and gas at surface conditions, respectively. Developed by and for the petroleum industry, the model uses equations of state that are defined through formation-volume factors $B_c$, or the reciprocal shrinkage factors $b_c = 1/B_c$, and the solubility ratios $r_s$ and $r_v$. The shrinkage factor $b_c$ is defined as the ratio of volume occupied by the same number of molecules of (pseudo)-component $c$ at surface conditions relative to the volume at reservoir conditions. The solubility of gas in oil is modeled by $r_s$. This is defined as

the volume of gas at standard conditions that at reservoir conditions can be dissolved into the amount of oil that forms a unit volume at so-called stock-tank conditions. Similarly, vaporization of oil into the vaporous gas phase is modeled by $r_v$, defined as the volume of surface condensate oil that can be vaporized in a unit volume of surface gas at reservoir conditions; see, e.g., [111, Chapter 11] for a thorough explanation to these concepts. In the black-oil model, the aqueous phase is assumed to consist of the water component only, so that the mass, flux, and source terms for this phase are given by (2.13) with $\alpha = w$. The mass, flux, and source terms for the oil ($o$) and gas ($g$) pseudo-components read

$$\begin{aligned} \mathcal{M}_o &= \phi(b_o S_o + b_g r_v S_g), & \mathcal{M}_g &= \phi(b_g S_g + b_o r_s S_o), \\ \vec{\mathcal{F}}_o &= b_o \vec{v}_o + b_g r_v \vec{v}_g, & \vec{\mathcal{F}}_g &= b_g \vec{v}_g + b_o r_s \vec{v}_o, \\ \mathcal{Q}_o &= b_o q_o + b_g r_v q_g, & \mathcal{Q}_g &= b_g q_g + b_o r_s q_o. \end{aligned} \tag{2.15}$$

Note that an accurate description of the fluids in a petroleum reservoir should also account for thermal effects, diffusion of components due to concentration gradients, and chemical reactions. Treatment of these phenomena is outside the scope of this thesis.

## 2.3 Geological models

It is clear from the discussion in Section 2.1 that porous rock formations show large local variations in their ability to transmit fluids. Reservoirs are in many cases also delineated into different flow compartments by faults or other sealing mechanisms. The first step in accurate description of flow in a subsurface reservoir is therefore an accurate description of the reservoir geology.

Geological data from a subsurface reservoir are hard to obtain, especially when the reservoir is located off-shore and buried hundreds or thousands of meters underneath the seabed. Available information then typically consists of data from geophysical (seismic, electromagnetic) surveys, as well as logs and core samples extracted from a small number of wells drilled from the porous rock at strategical points. These data, along with a conceptual understanding of the geological history of the reservoir rock, are used by the geomodeller to determine the stratigraphic layering and sedimentology of the reservoir. Since data points are scarce, computational tools like geostatistical techniques and/or process simulations are used to fill in plausible missing pieces to obtain a *geological model* that represents the full reservoir volume. This model consists of a collection of small REVs,

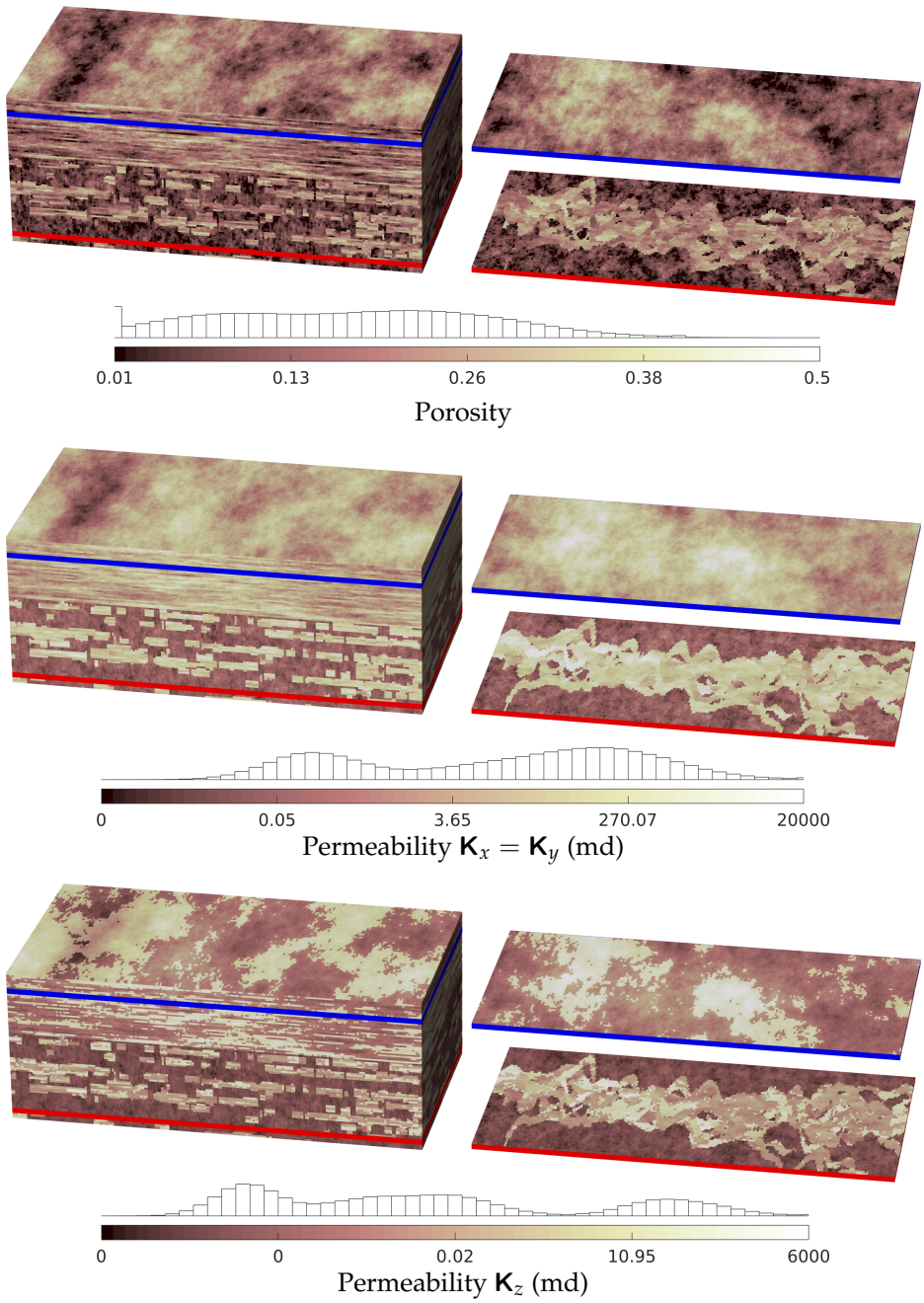



**Figure 2.8:** Porosity and permeability from Model 2 of the SPE10 benchmark study. The blue line shows Layer 10, which is part of the Tarbert formation, whereas the red line shows Layer 80, which is part of the Upper Ness formation.

commonly referred to as cells, with distinct *petrophysical properties* like permeability and porosity.

Figure 2.8 shows Model 2 from the 10th SPE Comparative Solution Project [41], which is an artificial geological model that is widely used as a benchmark for reservoir simulation. This model consists of $60 \times 220 \times 85$ rectangular cells of $20 \times 10 \times 2$ ft$^3$, and is a geostatistical realization of the Jurrasic Brent formations in the northern North Sea region. The top 35 horizontal layers have properties from the Tarbert formation, which is made up of shallow-marine sandstone and coal beds, whereas the next 50 layers have properties from the Upper Ness formation, consisting of an intertwined pattern of fluvial sandstone channels on a background of low-permeable mudrock and coal beds. The petrophysical properties are highly heterogeneous and has permeability values spanning eight orders of magnitude. Note that the ratio of horizontal to vertical permeability is exaggerated in this benchmark in order to make it challenging to *upscale*, which we will discuss in the next section.

Figure 2.9 shows a geological model of the Norne oil and gas field from the Norwegian Sea, first made openly available by the NTNU IO-Center [87] and later by the Open Porous Media (OPM) initiative [143]. The inset shows a number of faults, which together delineate a sealed compartment in the full model. This compartment makes up the oil and gas reservoir, with porosity ranging from 0.09 to 0.35. The figure also shows two different vertical slices of the model. Notice how geological activity has shifted different parts of the model in the vertical direction, creating faults. Figure 2.10 shows the permeability, which spans four orders of magnitude.

While not part of the geological model, resident fluid properties are also part of the reservoir description, since different regions of a reservoir usually also have different relative permeability and capillary pressure curves. A more detailed discussion of geological models aimed at mathematicians, computer scientists, and other non-geologists can be found in e.g., [111, Chapter 2]

## Upscaling

Even though a geological model is significantly less detailed than a pore network description, spatial resolution of contemporary geological models will nonetheless tend to be too high to allow simulations to be performed directly on the model. Moreover, there are usually large uncertainties associated with a geological model, so that we often have a number of equally probable model realizations of the same reservoir. Therefore, even

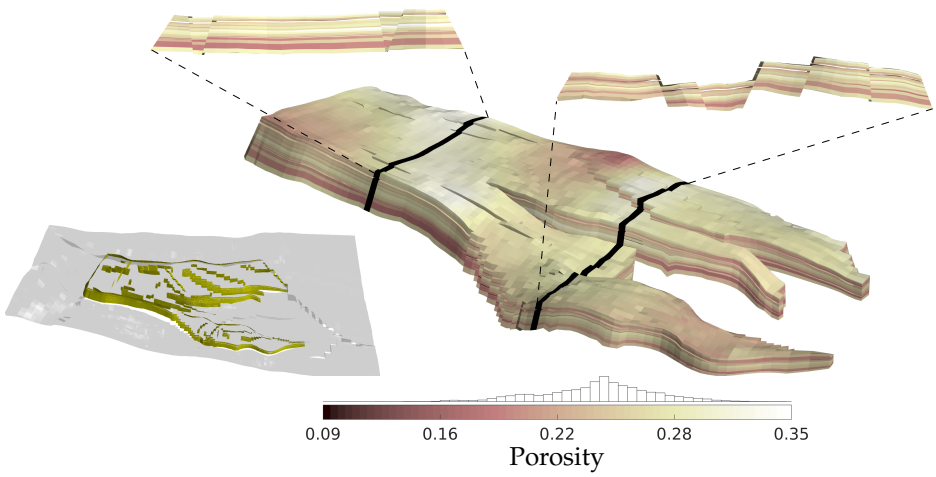



**Figure 2.9:** Geological model of the Norne oil and gas field in the Norwegian Sea. The inset on the left shows a number of faults that delineates the active part of the model. The porosity of this part is indicated by color, with two highlighted slices.

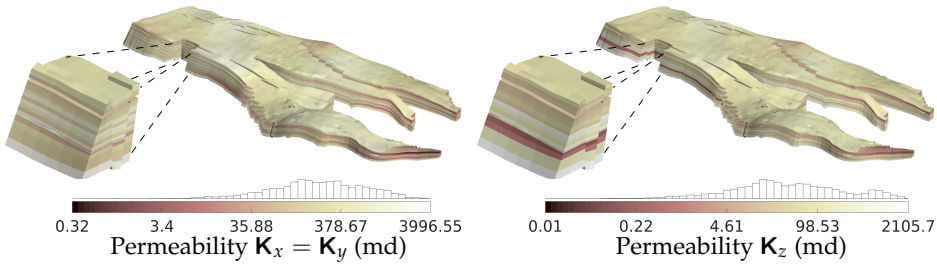



**Figure 2.10:** Permeability of the Norne oil and gas field. The cut-out shows that the model has large local variations in permeability, especially in the vertical direction.

with sufficient computing power available, it is more common to perform many simulations of different realizations of the same reservoir at a somewhat coarser resolution instead of performing a single direct simulation of a highly resolved geological model.

The process of obtaining a coarse version of a geological model is called upscaling, and amounts to propagating properties from the fine geological model to a model with a lower number of cells. Petrophysical properties of a cell in the upscaled model are often computed by averaging the properties of cells in the fine model that comprise the coarse cell. A review of different upscaling techniques can be found in e.g., [151, 65, 68, 18, 40]. An

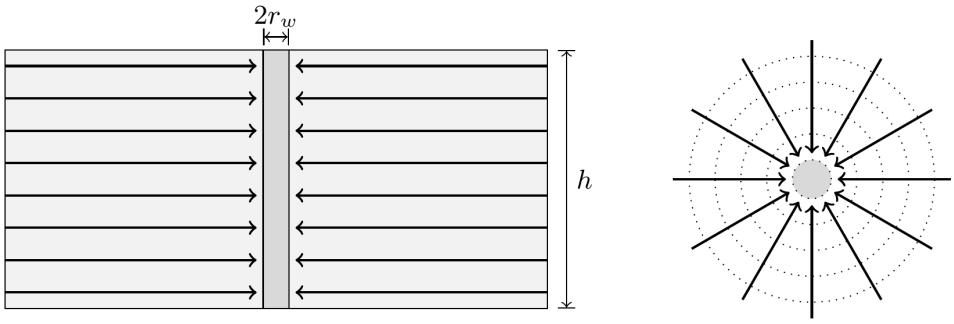



**Figure 2.11:** Radial flow into a vertical wellbore of diameter $r_w$ and height $h$. Figure from [111].

important result of the upscaling process is that cells tend to get anisotropic and full-tensor permeabilities, which make the subsequent flow simulations more demanding than for models with isotropic properties.

Upscaling methods are not discussed much further in this thesis, but form an important background for our work on multiscale methods in Paper IV and adaptive coarsening methods in Paper IX.

## 2.4 Wells

Wells are drilled into a reservoir to extract fluids (typically a mixture of hydrocarbons and formation water) out of the reservoir and inject other fluids into the reservoir (typically water or gas, possibly with chemical and/or biological substances) to maintain pressure and/or displace the resident fluids. In its simplest form, a well is a hole with a radius from 5 to 40 inches, but in most cases, the hole is cased with steel and cement and equipped with perforations or more advanced valve systems that allow fluids to flow between the reservoir and wellbore, i.e., the void space inside the well that is connected to the surface. Injection wells are often operated at a constant injection rate, whereas production wells are usually operated at a constant *bottom-hole pressure*, defined as the pressure at a certain depth (usually the lowest perforation) inside the wellbore. In a real production scenario, wells can also be controlled by a number of parameters such as maximum bottom-hole pressure; oil-, gas-, and water-rates; total volumetric rate; etc.

Flow near the wellbore usually takes place on a smaller scale compared with flow in the rest of the reservoir. It is therefore common to introduce some kind of analytical or semi-analytical model to determine the flow

rates in/out of wells and the pressure drop in the near-well region. If we consider steady-state, single-phase, radial flow into a vertical wellbore, as depicted in Figure 2.11, the mass term $\mathcal{M}$ is constant, and we can rewrite (2.8) in cylinder coordinates as

$$\frac{1}{r}\frac{\partial}{\partial r}(r\rho v) = 0, \tag{2.16}$$

where $v = v(r)$ is the radial Darcy velocity. We obtain $v = C/\rho r$ for some constant $C$. With this formula and a given injection rate $q$, we can integrate over a small cylinder with radius $r$ around the wellbore to get

$$q = \int_0^h \int_0^{2\pi} r\rho v(r)\,\mathrm{d}\theta\,\mathrm{d}z = 2\pi h C. \tag{2.17}$$

This gives $C = \frac{q}{2\pi h}$, which we insert into Darcy's law to get

$$v = \frac{q}{2\pi\rho h} = -\frac{K}{\mu}\frac{\mathrm{d}p}{\mathrm{d}r}.$$

Rearranging and integrating from the wellbore radius $r_w$ to an arbitrary radius $r > r_w$, this reads

$$\frac{q}{2\pi h K}\int_{r_w}^{r}\frac{1}{r}\,\mathrm{d}r = \int_{r_w}^{r}\frac{\rho}{\mu}\frac{\mathrm{d}p}{\mathrm{d}r}\,\mathrm{d}r = \left(\frac{\rho}{\mu}p\right)\Big|_{r_w}^{r} + \int_{r_w}^{r} p\frac{\partial}{\partial r}\left(\frac{\rho}{\mu}\right)\,\mathrm{d}r.$$

Density and viscosity will generally both increase with pressure, and the fraction $\rho/\mu$ will therefore typically vary much less than the pressure $p$ with respect to $r$. Assuming $\rho/\mu$ to be constant, we get the expression

$$p(r) = \frac{q\mu}{2\pi h K\rho}\ln\left(\frac{r}{r_w}\right) + p_w. \tag{2.18}$$

Expressions for idealized scenarios such as (2.18) are hardly representative for realistic scenarios: the flow is rarely single phase and perfectly radial for a variety of reasons. The permeability is often altered nonuniformly in the near-well region during drilling and completion, the flow is affected by buoyant forces, resident and injected fluids exist in multiple phases and comprise a multitude of components, and the wellbore is hardly ever fully vertical. In fact, the characteristic "pancake-topology" of petroleum reservoirs means that *horizontal* well trajectories with bends and branches are much more effective. Figure 2.12 illustrates the complexity of a real well with multiple branches from the Troll oil and gas field in the North Sea by superimposing it on the streets of Manhattan.

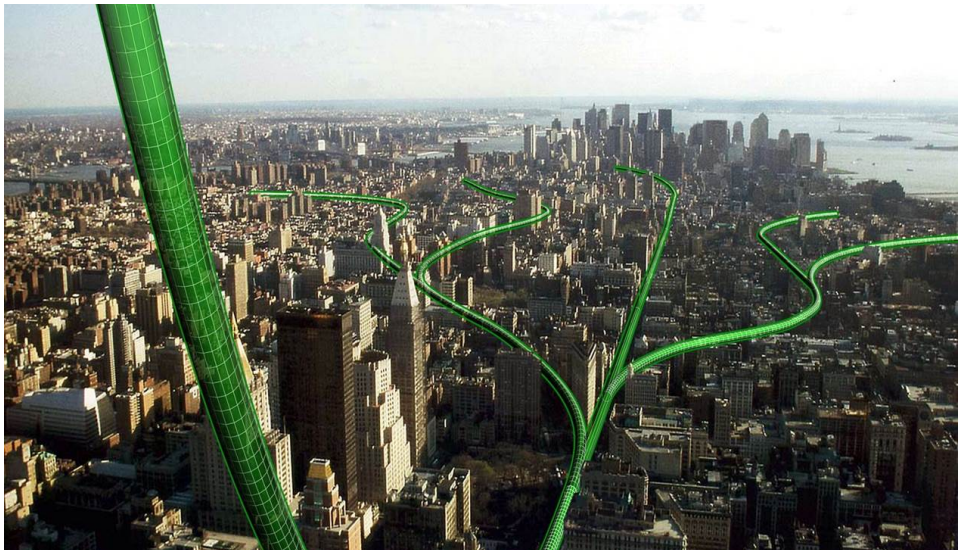


**Figure 2.12:** Horizontal well from the Troll oil and gas field in the North Sea, superimposed on Manhattan, New York for perspective. Figure by Baker Hughes Inc, published under the creative commons CC-BY-SA-4.0 license.

Needless to say, closed-form expressions that take all these effects into account are generally not possible to obtain, and it is more common to use subscale models that represent the subscale variations in the near-well zone in an averaged sense. One example is Peaceman-type well models, which are widely used in reservoir simulations. First derived by Muskat and Wyckoff [136], and developed further by Peaceman [145, 144] by using analytical solutions to an infinitely repeated pattern of producers and injectors in a so-called five-spot configuration, such models relate the average cell pressure to the pressure at the wellbore using expressions of the form (2.18). A more in-depth discussion of such well models can be found in [111, Chapter 4].

Wells are in most cases the primary controls that determine flow inside a reservoir, which has lead to development of a number of models that aim to represent more complex wells and near-well flow. This includes mechanistic models to represent changes of the permeability in the near-well zone because of hydraulic fracturing and filter-cake buildup, as well as specialized solution algorithms to handle cross-flow, where fluids flow into one part of the wellbore, and out of another part [155]. So-called multisegment well models have been introduced to model multiphase effects, fluid storage, and better account for pressure drops caused by chokes and valves, downhole pumps and separators, and frictional forces in long horizontal wells [170, 82].

# Chapter 3

# Discretization Schemes

With a geological model equipped with petrophysical properties, a fluid model with governing equations and constitutive relations, and a model for wells and production facilities, we can now translate the problem of fluid flow in a subsurface reservoir into computer code by means of numerical methods. We start by writing conservation of mass for an immiscible phase $\alpha$ (2.12) on residual form,

$$\mathcal{R}_\alpha := \partial_t \mathcal{M}_\alpha + \nabla \cdot \vec{\mathcal{F}}_\alpha - \mathcal{Q}_\alpha = 0. \tag{3.1}$$

In this chapter, we will consider discretization of the different terms in this equation. The discretization of compositional and black-oil equations is analogous.

## 3.1 Temporal discretization

The most common temporal discretizations used in reservoir simulation are special cases of the general $\theta$ scheme. By denoting the unknowns (in our example, saturations $S_\alpha$ and pressures $p_\alpha$) as $u$, we write $\mathcal{M}_\alpha = \mathcal{M}_\alpha(u)$ to signify that $\mathcal{M}$ depends on the unknowns $u$. Moreover, we write $u^n$ to signify that the unknowns are evaluated at a discrete time $t_n$, and introduce the short-hand notation $\mathcal{M}_\alpha^n := \mathcal{M}_\alpha(u^n)$. With this notation, the $\theta$ scheme written in residual form reads

$$\begin{aligned} \mathcal{R}_\alpha^{n+1} =& \frac{1}{\Delta t}(\mathcal{M}_\alpha^{n+1} - \mathcal{M}_\alpha^n) \\ &+ \theta\big(\nabla \cdot \vec{\mathcal{F}}_\alpha^{n+1} - \mathcal{Q}_\alpha^{n+1}\big) + (1-\theta)\big(\nabla \cdot \vec{\mathcal{F}}_\alpha^{n} - \mathcal{Q}_\alpha^{n}\big) = 0, \end{aligned} \tag{3.2}$$

where $\Delta t = t_n - t_{n-1}$ denotes the time step. The value of $\theta$ determines the method: $\theta = 0$ gives the explicit forward Euler method and $\theta = 1$ gives

the implicit backward Euler method, which are both first-order accurate, whereas $\theta = 1/2$ yields the Crank-Nicholson method, which is formally second-order accurate. Description of these methods can be found in any standard textbook on numerical mathematics, see e.g., [174].

An explicit scheme ($\theta = 0$) yields a linear residual equation (3.2), meaning that we can solve a linear system of residual equations to advance the solution one timestep $\Delta t$ from time $t^n$ to $t^{n+1}$. The main drawback is that this formulation is only conditionally stable and generally puts limitations on the choice of time steps. For reservoir simulation, these restrictions are rather severe, as we will come back to in more detail in Section 3.4, and explicit methods are thus rarely used in engineering computations.

The implicit scheme ($\theta = 1$), on the other hand, is unconditionally stable, and is therefore the standard approach to reservoir simulation of black-oil type models. The implicit scheme results in a system of nonlinear residual equations, and we must typically use a *nonlinear* solver like Newton's method to find its solution. This can be computationally demanding, and much research has thus been invested in formulating solution strategies that decouple the flow equations, either at the continuous or (semi-)discrete level, so that they can be solved more efficiently using some kind of divide-an-conquer approach.

It is also possible to combine the two methods by using different values for $\theta$ in different parts of the reservoir, which is commonly referred to as the adaptive implicit method (AIM) [177].

### Sequential splitting

The physical quantities involved in (3.1) exhibit very different physical behavior: Given a reservoir in equilibrium, a well that suddenly starts injecting fluids will create a pressure transient that results in a rapidly moving pressure pulse through the reservoir. This pulse will drive the fluid transport. The transport is significantly slower, and will mostly take place locally, seen as a slowly propagating fluid front. At the temporal scales of interest in reservoir simulation, we can say (somewhat simplified) that pressure is a parabolic variable with a strong elliptic character, whereas fluid transport is parabolic with a strong hyperbolic character [24, 111]. This mixed mathematical character can be exploited when discretizing (3.1) by splitting the equations into subproblems for flow and transport [193, 180] that then can be solved sequentially.

*Flow*

We can construct a flow equation for the evolving pressure at the semi-discrete level as the weighted sum of the conservation equations

$$\mathcal{R}_F^{n+1} = \sum_{\alpha=a,\ell,v} \omega_\alpha \mathcal{R}_\alpha^{n+1} = 0, \tag{3.3}$$

where the decoupling weights $\omega_\alpha$ satisfy

$$\sum_{\alpha=a,\ell,v} \partial_u(\omega_\alpha \mathcal{M}_\alpha^{n+1}) = 0 \quad \text{for all non-pressure variables } u.$$

In a sequential solution strategy, the flow equation (3.3) is first solved for one of the phase pressures $p_\alpha$ and for the total Darcy velocity

$$\vec{v} = \vec{v}_a + \vec{v}_\ell + \vec{v}_v,$$

while keeping the transport quantities (saturations and mass fractions) fixed.

Neglecting compressibility and gravity, the weights for (3.1) are simply 1, which gives the flow equation

$$\mathcal{R}_F^{n+1} = \frac{\phi}{\Delta t}\left(\sum\nolimits_\alpha S_\alpha^{n+1} - \sum\nolimits_\alpha S_\alpha^n\right) + \nabla \cdot \left(\sum\nolimits_\alpha \vec{v}_\alpha^{n+1}\right) + \sum\nolimits_\alpha q_\alpha^{n+1} = 0.$$

Because of the closure relation, the terms $\mathcal{M}_F = \sum_\alpha S_\alpha = 1$ at time $n$ and $n+1$ cancel. The source/sink term is simply $\mathcal{Q}_F = q_T = \sum_\alpha q_\alpha$, and with no capillary effects ($p_\alpha = p$), we can use Darcy's law to write the flux term as

$$\vec{\mathcal{F}}_F = -\sum_{\alpha=a,\ell,v} \lambda_\alpha \mathbf{K} \nabla p = -\lambda_T \mathbf{K} \nabla p.$$

This gives the flow equation

$$\mathcal{R}_F^{n+1} = -\nabla \cdot \lambda_T \mathbf{K} \nabla p^{n+1} + q_T^{n+1} = 0,$$

where we have introduced the total mobility $\lambda_T = \sum_\alpha \lambda_\alpha$. Note that in this simple case, summation of the continuous residual equations (3.1) would result in an analogous continuous pressure equation. This is generally true for immiscible, incompressible multiphase flow, where the weights are $\omega_\alpha = 1/\rho_\alpha$. For more complex scenarios such as compositional simulation, analytic expressions for the decoupling weights are cumbersome to compute, and the decoupling is usually done once the equations are fully discretized in time and space, see e.g., [44, 133].

*Explicit transport*

After solving the flow equation with fixed transport quantities, the transport equations $\mathcal{R}_\alpha^{n+1} = 0$ must be solved to advance the solution in time. One possible approach is to do this explicitly, e.g., by setting $\theta = 0$ in (3.2). For immiscible multiphase flow, this gives a trivial transport equation on the form

$$S_\alpha^{n+1} = \frac{1}{(\phi\rho_\alpha)^{n+1}} \left( (\phi\rho_\alpha S_\alpha)^n + \Delta t \left( \nabla \cdot \vec{\mathcal{F}}_\alpha^n - \mathcal{Q}_\alpha^n \right) \right).$$

Explicit transport is usually combined with an implicit treatment of the flow equation, and if both equations are advanced with the same time step[1], the resulting method is commonly referred to as the IMplicit Pressure, Explicit Saturation (IMPES) method [44, 43]. As with any explicit method, this requires that the timestep satisfies the Courant–Friedrichs–Lewy (CFL) condition, given that the spatial discretization of the transport equation involves a computational grid. We will come back to this later.

*Implicit transport*

If we instead set $\theta = 1$ and solve the transport equations implicitly, we need expressions for the phase velocities $\vec{v}_\alpha^{n+1}$ at timestep $n+1$. We can obtain this by using the total velocity $\vec{v}_T$ to reformulate the expression for the phase velocities

$$\vec{v}_\alpha = f_\alpha \left( \vec{v}_T + \mathbf{K} \sum_{\beta=a,\ell,v} \lambda_\beta [\vec{G}_\alpha - \vec{G}_\beta] \right), \tag{3.4}$$

where we have introduced the terms

$$f_\alpha = \frac{\lambda_\alpha}{\lambda_a + \lambda_\ell + \lambda_v}, \quad \text{and} \quad \vec{G}_\alpha = \rho_\alpha g \nabla z - \nabla p_c^\alpha. \tag{3.5}$$

We refer to $f_\alpha$ as the *fractional flow function*, which for two-phase flow usually has a characteristic S shape. Figure 3.1 shows fractional flow functions for two-phase flow with simple Brooks-Corey relative permeabilities $k_{r,\alpha} = S_\alpha^2$ and varying viscosity ratios between the wetting $(w)$ and non-wetting $(n)$ phases. The figure also shows corresponding horizontal 1D displacement profiles for incompressible flow with constant injection

[1] Many authors also mistakenly refer to methods that subdivide the implicit pressure step into multiple explicit substeps when solving the transport equation as IMPES, but strictly speaking, such schemes should be called sequentially explicit.

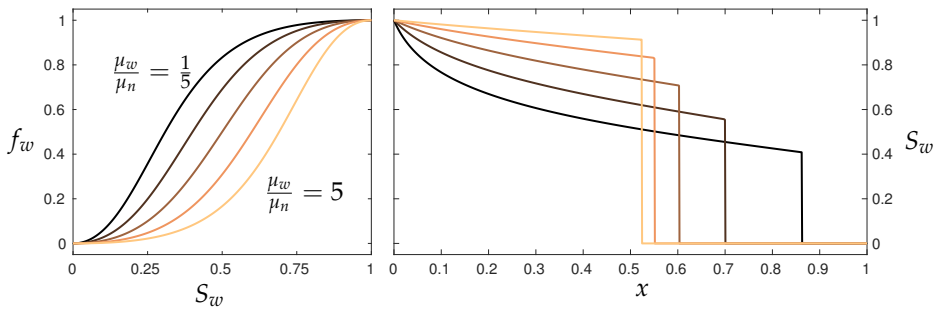



**Figure 3.1:** Different fractional flow functions and corresponding Buckley–Leverett displacement profiles. The 1D reservoir is initially filled with the non-wetting phase, and is displaced by the wetting phase. The relative permeability exponent equals 2 for all profiles, but the the wetting- to non-wetting viscosity ratio differs. The lowest ratio $1/5$ (black) results in a weak and fast propagating front, whereas a ratio of 5 gives a strong, piston-like displacement.

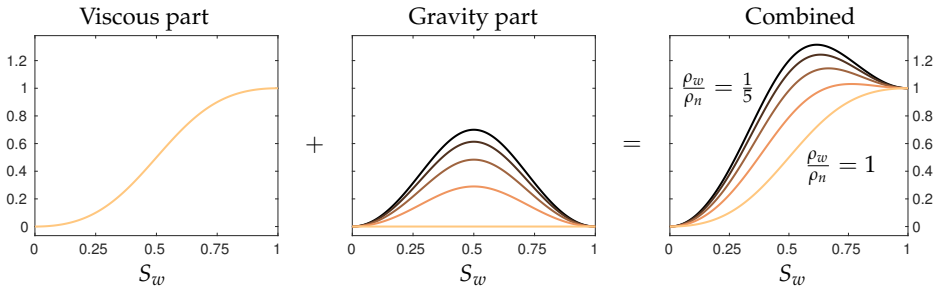



**Figure 3.2:** Different wetting- to non-wetting density ratios changes the fractional flow function. Notice that at the maximum, the sum of the viscous and gravity part is *greater than* one, reflecting that when we have counter-current flow due to gravity, the phase velocity $\vec{v}_\alpha$ can be greater than the total velocity $\vec{v}$.

of the wetting phase through the left boundary, which is an example of a Buckley–Leverett displacement. Figure 3.2 shows the fractional flow function for varying density ratios $\rho_w/\rho_n$, and its two components: the viscous flux $f_w\vec{v}_T$, and the gravity flux $f_w\mathbf{K}\lambda_n(\vec{G}_w - \vec{G}_n)$. After solving the pressure equation, the implicit transport equations $\mathcal{R}_\alpha^{n+1} = 0$ are reformulated by inserting the fractional-flow expression (3.4) for the phase velocities, and then solved to advance the transport solution in time.

The sequential splitting methodology is appealing not only because it divides a large and coupled system of equations into two smaller subproblems that are less expensive to solve, but also because it enables us to use

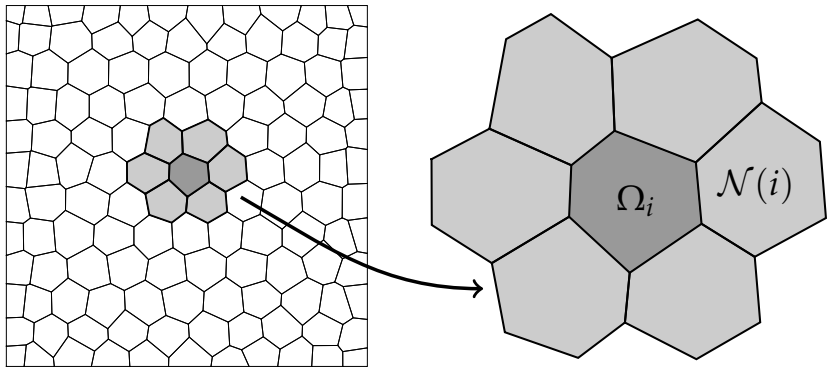



**Figure 3.3:** Computational grid with polygonal cells in 2D. Cell $\Omega_i$ is colored dark gray, whereas its topological neighbors $\mathcal{N}(i)$ are colored light gray.

specialized solvers that exploit the distinctly different mathematical characters of each subproblem. Therefore, it is an integral component in much of the work in this thesis. On the other hand, there is no guarantee that the method converges to the fully coupled solution when the coupling between flow and transport is strong. We will come back to this in Chapter 4, where we also discuss specialized solvers for the different subproblems.

## 3.2 Computational grids

The starting point for almost any spatial discretization method is to subdivide the computational domain $\Omega$ into $n_c$ non-overlapping cells $\{\Omega_i\}_{i=1}^{n_c}$, which we refer to as our computational grid, or simply grid. Figure 3.3 shows an example of a computational grid in 2D with polygonal cells. The figure shows an enlarged view of cell $\Omega_i$, and its topological neighbors $\mathcal{N}(i)$. In the following, we will refer to a cell either by $\Omega_i$, or its index $i$.

Since a geological model consists of non-overlapping REVs, it can be used directly as a computational grid. However, as mentioned in Section 2.3, real geological models typically consist of so many cells that it is necessary to use upscaling to construct a computational grid of reasonable size. This grid should preferably adapt to important geological structures and wells in the reservoir, since these have a direct impact on the the flow pattern.

The de facto grid format in reservoir simulation is *stratigraphic grids*, which were first introduced by Ponting [148]. Figure 3.4 illustrates the construction of such grids: The starting point is a (structured) mesh of co-

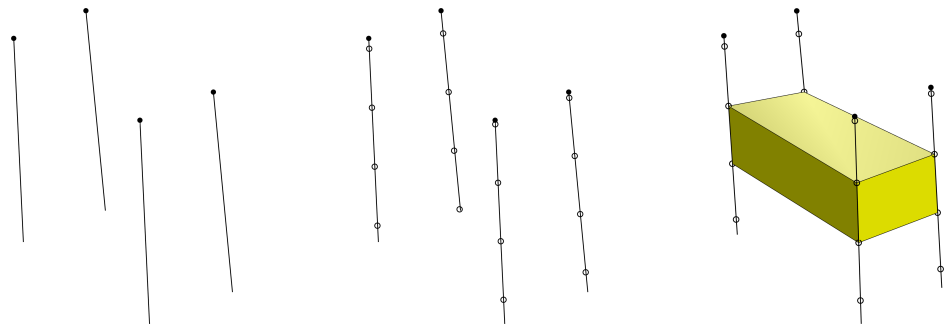


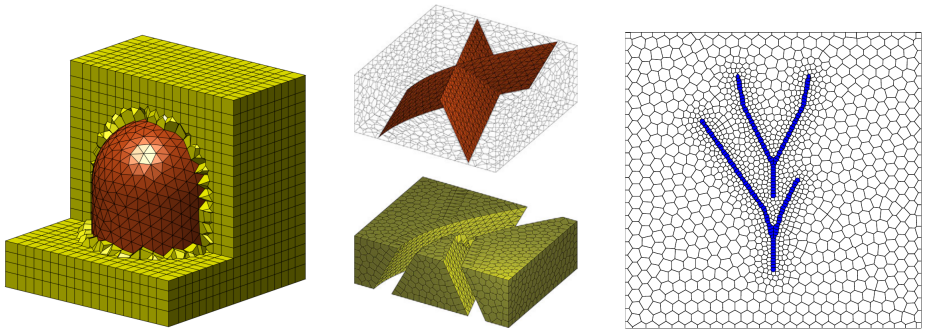

**Figure 3.4:** Construction of a corner-point grid. Pillars are extended downwards from a structured mesh, shown here as four black dots. Next, depth coordinates are defined along each pillar, indicated by circles. Finally, two depth coordinates on each of the four pillar defines a hexahedral grid cell.

ordinates covering the reservoir lateral area, with a pillar extending downwards into the formation from each point. A grid cell is defined by eight corners given as two depth coordinates along the four pillars that delimit the cell laterally, which is why the descriptive term *corner-point grid* is often used. This construction induces a logical $ijk$-ordering of the grid cells, and the resulting grid format can adapt to geological features such as faults, fractures, and erosion. As such, stratigraphic grids are appealing from a geological point of view. On the other hand, such grids are not able to capture more complex geological features such as Y-shaped faults or easily adapt to deviated and undulating well trajectories. The Norne field model introduced in Figure 2.9 is an example of a stratigraphic grid.

One way to accurately conform to complex features in the reservoir is to use an unstructured grid format. The grid in Figure 3.3 is an example of an unstructured format called perpendicular bisector (PEBI) grid (see, e.g. [79]). Also referred to as Voronoi diagrams, this type of grid is constructed as the dual of a Delaunay triangulation with vertices $\{\boldsymbol{x}_i\}_{i=1}^{n_c}$: a vertex $\boldsymbol{x}_i$ in the triangulation defines a grid cell that consists of all points that are at least as close to $\boldsymbol{x}_i$ as any other vertex:

$$\Omega_i = \{\boldsymbol{x} \in \Omega : |\boldsymbol{x} - \boldsymbol{x}_i| < |\boldsymbol{x} - \boldsymbol{x}_j| \text{ for all } j \neq i\}.$$

PEBI grids can be constructed so that they conform to surfaces representing features like faults and fractures, and curves representing well paths, see e.g., [124, 66]. In Paper I and Paper II, we introduce a new method for generating PEBI grids that conform to surfaces and piecewise linear curves



Salt dome | Inclined, intersecting faults | Y-shaped well paths

**Figure 3.5:** PEBI grids can be constructed to conform to complex features such as geological surfaces, and honor intersecting faults and complex well paths. Figures from Paper I and [111].

and enable spatially varying resolution; Figure 3.5 shows a few illustrative examples.

## 3.3 Spatial discretization

Equipped with a computational grid, we are ready to derive a fully discrete version of the conservation equations. For notational convenience, we henceforth simply set $\theta = 1$ in (3.2), which gives us the implicit backward Euler method, and tactically postpone the rationale behind this choice until after we have introduced spatial discretizations. Multiplying $\mathcal{R}_\alpha^{n+1}$ with a test function $\psi$ in a function space $V$, and integrating in space over a cell $\Omega_i$ gives the weak formulation:

$$\begin{aligned}
&\text{Find } u^{n+1} \in U \text{ such that for all } \psi \in V, \quad i = 1, \dots n_c, \\
&\mathcal{R}_{\alpha,i}(u^{n+1}, u^n, \psi) = \frac{1}{\Delta t} \int_{\Omega_i} \left( \mathcal{M}_\alpha(u^{n+1}) - \mathcal{M}_\alpha(u^n) \right) \psi \, \mathrm{d}V \\
&\qquad + \int_{\Omega_i} \left( \nabla \cdot \left[ \vec{\mathcal{F}}_\alpha(u^{n+1}) \psi \right] - \vec{\mathcal{F}}_\alpha(u^{n+1}) \cdot \nabla \psi \right) \mathrm{d}V \\
&\qquad - \int_{\Omega_i} \mathcal{Q}_\alpha(u^{n+1}) \psi \, \mathrm{d}V = 0.
\end{aligned} \tag{3.6}$$

We get a fully discrete formulation by replacing the function spaces[2] $U$ and $V$ with suitable finite-dimensional subspaces $U_h$ and $V_h$ and replacing the unknowns $u$ and the test function $\psi$ with approximations $u_h \in U_h$ and $\psi_h \in V_h$.

By imposing continuity of the functions at the interface between different cells, we obtain a continuous Galerkin (cG) method, whereas allowing for discontinuity gives a discontinuous Galerkin (dG) method. A description of cG and dG is outside the scope of this introduction, and we refer the interested reader to e.g., [152] for an introduction to dG methods applicable to flow in porous media. Discontinuous Galerkin methods for black-oil and compositional models are discussed in Paper VII and Paper VIII.

In the rest of this chapter, as well as in most of the papers that make up the second part of this thesis, we focus on finite-volume methods (FV), which can be seen as a special case of dG methods in which $V_h$ is taken to be the space of functions that are constant on each cell. This gives a robust method that makes very few assumptions on cell geometry and topology, and is therefore currently the industry-standard in reservoir simulation. With this choice of $V_h$, the weak formulation (3.6) simplifies to

$$\mathcal{R}_{\alpha,i}^{n+1} = \frac{1}{\Delta t}\int_{\Omega_i}\left(\mathcal{M}_\alpha^{n+1} - \mathcal{M}_\alpha^n\right)\mathrm{d}V + \int_{\Omega_i}\nabla\cdot\vec{\mathcal{F}}_\alpha^{n+1}\,\mathrm{d}V - \int_{\Omega_i}\mathcal{Q}_\alpha^{n+1}\,\mathrm{d}V = 0. \tag{3.7}$$

In the following, we explain how to discretize the different terms in this equation. First, we write the mass term as

$$\int_{\Omega_i}\mathcal{M}_\alpha^{n+1}\,\mathrm{d}V = |\Omega_i|\mathcal{M}_{\alpha,i}^{n+1}, \tag{3.8}$$

where $|\Omega_i|$ is the volume of cell $i$ and subscript $i$ indicates a volume average over $\Omega_i$. For immiscible, multiphase flow, the mass term reads $\mathcal{M}_{\alpha,i}^{n+1} = (\phi\rho_\alpha S_\alpha)_i^{n+1}$, which we usually take to be the product of the individual cell averages $\rho_{\alpha,i}$ and $S_{\alpha,i}$, whereas $\phi_i$ is assumed constant over the cell. Discretizing the flux and source terms needs a more elaborate description.

### Discretization of flux terms

In order to discretize the flux term, we first move the integral to the surface $\partial\Omega_i$ of $\Omega_i$,

$$\int_{\Omega_i}\nabla\cdot\vec{\mathcal{F}}_\alpha\,\mathrm{d}V = \int_{\partial\Omega_i}\vec{\mathcal{F}}_\alpha\cdot\vec{n}_i\,\mathrm{d}S,$$

[2]On purpose, we do not specify these spaces in any detail, since different unknowns and the test functions can potentially belong to different function spaces.

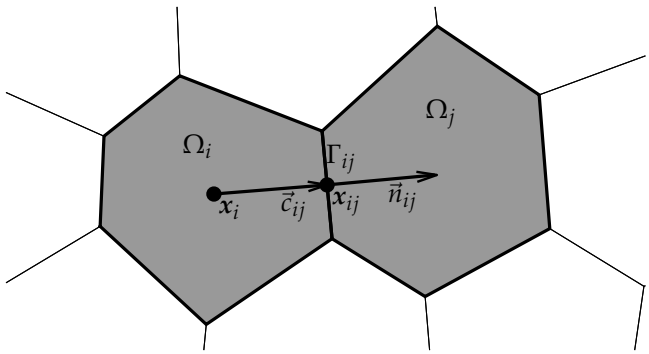



**Figure 3.6:** Cells $\Omega_i$ and $\Omega_j$ share a common interface $\Gamma_{ij}$, with the interface normal $\vec{n}_{ij}$ oriented form cell $\Omega_i$ to cell $\Omega_j$. The vector $\vec{c}_{ij}$ points from the cell centroid $\boldsymbol{x}_i$ of $\Omega_i$ to the face centroid $\boldsymbol{x}_{ij}$ of $\Gamma_{ij}$.

where $\vec{n}_i$ is the outward surface normal of $\partial\Omega_i$. Recalling that $\mathcal{N}(i)$ denotes the set of cells that share a common interface with cell $i$ (see Figure 3.3), this integral can be written as a sum

$$\int_{\partial\Omega_i} \vec{\mathcal{F}}_\alpha \cdot \vec{n}\,\mathrm{d}S = \sum_{j\in\mathcal{N}(i)} \int_{\Gamma_{ij}} \vec{\mathcal{F}}_\alpha \cdot \vec{n}_{ij}\,\mathrm{d}S = \sum_{j\in\mathcal{N}(i)} |\Gamma_{ij}| \vec{\mathcal{F}}_{\alpha,ij} \cdot \vec{n}_{ij}. \tag{3.9}$$

Here, $\Gamma_{ij}$ is the common interface between $\Omega_i$ and $\Omega_j$, with normal $\vec{n}_{ij}$ pointing from cell $i$ to cell $j$, whereas $|\Gamma_{ij}|$ is the interface area, as depicted in Figure 3.6. Analogous to the volume integrals, subscript $ij$ indicates that quantities inside the parenthesis are the areal average over $\Gamma_{ij}$. Using again immiscible multiphase flow as an example, the flux term (3.9) reads

$$\int_{\Omega_i} \nabla \cdot \vec{\mathcal{F}}_\alpha\,\mathrm{d}V = \sum_{j\in\mathcal{N}(i)} |\Gamma_{ij}| (\phi\rho_\alpha \vec{v}_\alpha)_{ij} \cdot \vec{n}_{ij}.$$

Since we have assumed that our primary unknowns take on different constant values in $\Omega_i$ and $\Omega_j$, we must be careful when we define this areal average.

### *Two-point flux approximation*

In a finite-volume-setting, it is much more convenient to work with interface fluxes rather than velocities. To this end, we define the flux across interface $\Gamma_{ij}$ as

$$v_{ij} = \int_{\Gamma_{ij}} \vec{v} \cdot \vec{n}\,\mathrm{d}S = -\int_{\Gamma_{ij}} \lambda \mathbf{K}(\nabla p - \rho g \nabla z) \cdot \vec{n}\,\mathrm{d}S, \tag{3.10}$$

where we have inserted Darcy's law from (2.14), omitting the phase subscript $\alpha$ for readability. With this definition, $v_{ij}$ is positive if the flow across $\Gamma_{ij}$ is from $\Omega_i$ to $\Omega_j$, and negative if it is from $\Omega_j$ to $\Omega_i$. For the method to be locally conservative, we must have that $v_{ij} = -v_{ji}$. First, consider the term $-\mathbf{K}\nabla p$. If we let $p_i$ and $p_{ij}$ denote the pressure at the centroid of cell $\Omega_i$ and interface $\Gamma_{ij}$, respectively, we can use finite differences to approximate the pressure gradient

$$-\int_{\Gamma_{ij}} \mathbf{K}\nabla p \cdot \vec{n}\, \mathrm{d}S \approx |\Gamma_{ij}| \frac{(\mathbf{K}_i \vec{n}_{ij}) \cdot \vec{c}_{ij}}{|\vec{c}_{ij}|^2}(p_i - p_{ij}) = T_{i,j}(p_i - p_{ij}). \tag{3.11}$$

Here, $\vec{c}_{ij}$ is the vector from the cell centroid $\boldsymbol{x}_i$ to the interface centroid $\boldsymbol{x}_{ij}$, as illustrated in Figure 3.6, and $\mathbf{K}_i$ is the constant permeability in $\Omega_i$. The quantity $T_{i,j}$ is referred to as the one-sided transmissibility. An analogous expression holds for the term $\mathbf{K}\nabla z$, with the vertical coordinates $z_i$ of the cell centroid $\boldsymbol{x}_i$ and $z_{ij}$ of the interface centroid $\boldsymbol{x}_{ij}$ in place of $p_i$ and $p_{ij}$.

Imposing continuity of the face pressures, mobilities, and densities ($p_{ij} = p_{ji}$, etc.), the approximations in cells $\Omega_i$ and $\Omega_j$ read

$$\begin{aligned} v_{ij} &= \lambda_{ij} T_{i,j} \left( [p_i - p_{ij}] - \rho_{ij} g [z_i - z_{ij}] \right), \\ v_{ji} &= \lambda_{ij} T_{j,i} \left( [p_j - p_{ij}] - \rho_{ij} g [z_j - z_{ij}] \right). \end{aligned}$$

Since a conservative method requires $v_{ij} = -v_{ji}$, this can be simplified to

$$v_{ij} = \lambda_{ij} T_{ij} \big( [p_i - p_j] - \rho_{ij} g [z_i - z_j] \big), \tag{3.12}$$

where we have introduced the transmissibility

$$T_{ij} = \left( T_{i,j}^{-1} + T_{j,i}^{-1} \right)^{-1}, \tag{3.13}$$

associating the permeabilities of cells $i$ and $j$ with their common interface.

This method is commonly referred to as the two-point flux approximation (TPFA) scheme: an easy-to-implement method that makes very few assumptions on the grid geometry and topology. In fact, apart from the vertical coordinate of the cell centroids, all the information we need from the grid and petrophysical properties are contained in the graph $(\{\Omega_i\}, \{T_{ij}\})$ defined using cells as nodes and transmissibilities as edges. For this reason, TPFA is widely used in the reservoir simulation community, and the standard method in most commercial simulators [111].

Since the method only uses two points to approximate the intercell fluxes, it is only consistent for so-called K-orthogonal grids [7, 195], in

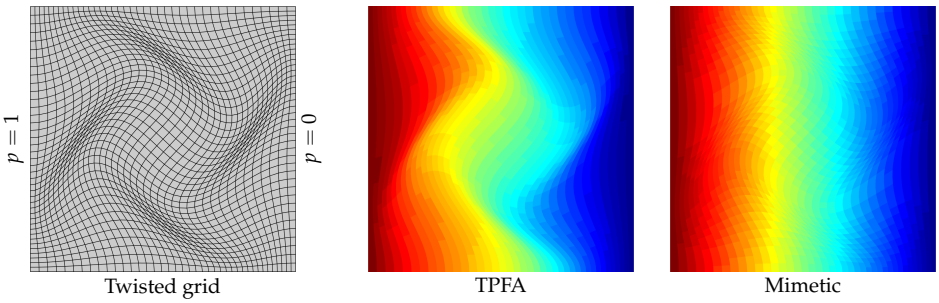



**Figure 3.7:** A twisted Cartesian grid that is not K-orthogonal with respect to an isotropic permeability tensor. Pressure boundary conditions are imposed on the left and right sides, with no-flow conditions on the top and bottom. Corresponding solutions using TPFA and the consistent mimetic finite difference method are shown to the right.

which the vector joining the two points used in the discretization aligns with the flow direction,

$$(\mathbf{K}_i \vec{n}_{ij}) \times \vec{c}_{ij} = 0, \quad \text{for all } i, j.$$

When this is not the case, TPFA will not converge towards the correct solution when we increase the grid resolution. A large research effort has therefore focused on developing methods that are consistent for general combinations of grids and permeability tensors: the multipoint flux approximation [6, 58], mimetic finite differences [34], and the *nonlinear* two-point flux approximation [108, 142, 163], in addition to the discontinuous Galerkin method [15], to name a few. Paper I and Paper III introduce some of these methods in more detail and compare and contrast how they perform on general unstructured grids.

Figure 3.7 illustrates how severe grid orientation effects can be for the TPFA method when the grid is not K-orthogonal. Twisting the inner nodes of a regular Cartesian according to the transformation

$$\begin{aligned}(x, y) &\mapsto (x + f(x, y), y - f(x, y)), \\ f(x, y) &= 0.1 \sin(\pi x) \sin(3\pi(y - 0.5)),\end{aligned}$$

gives a grid that is not K-orthogonal even for isotropic permeability. Imposing unequal constant pressure boundary conditions on the left and right sides and no-flow boundary conditions on the top and bottom sides should result in a linear pressure drop. However, the the pressure contours in

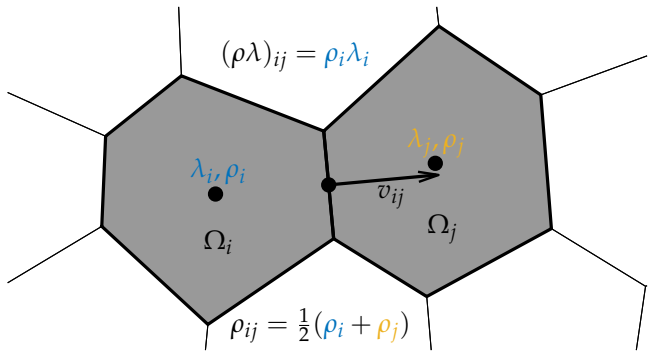



**Figure 3.8:** Upstream and average evaluations involved in the numerical flux calculation (3.16) when the flow is from $\Omega_i$ to $\Omega_j$.

the TPFA solution follows the grid lines and not the axial directions. Using a consistent method, here exemplified with a mimetic finite difference method, correctly reproduces a linear pressure drop.

Note that we have not given all details on how to discretize the Darcy flux: We still have to estimate the interface values $\lambda_{ij}$ and $\rho_{ij}$. The interface mobility is commonly evaluated differently for each phase depending on the direction of the phase flux,

$$\lambda_{\alpha,ij} = \begin{cases} \lambda_{\alpha,i} & \text{if } v_{\alpha,ij} \geq 0, \\ \lambda_{\alpha,j} & \text{if } v_{\alpha,ij} < 0. \end{cases} \tag{3.14}$$

This is called the single-point upstream (SPU) scheme. The interface density is defined as the average between the two cell averages,

$$\rho_{\alpha,ij} = \frac{1}{2}(\rho_{\alpha,i} + \rho_{\alpha,j}). \tag{3.15}$$

To sum up, the finite-volume method with two-point flux approximation and single-point upstream mobility weighting gives the discretized flux term

$$|\Gamma_{ij}|\vec{\mathcal{F}}_{\alpha,ij} \cdot \vec{n}_{ij} \approx (\rho_\alpha\lambda_\alpha)_{ij} T_{ij} \left( [p_{\alpha,i} - p_{\alpha,j}] - \rho_{\alpha,ij} g [z_i - z_j] \right). \tag{3.16}$$

Note that SPU (3.14) is used for both terms in $(\rho_\alpha\lambda_\alpha)_{ij}$, whereas $\rho_{\alpha,ij}$ is calculated as the arithmetic average. Figure 3.8 shows the upstream and average evaluations for multicomponent flow for a case where $v_{ij}$ is positive.

The definition of the upstream mobility weighting (3.14) is generally implicit, since it depends on the direction of the phase flux, which itself depends on the mobility. This may cause convergence problems in sequential simulations. Equivalent explicit expressions for the upstream direction have been derived by Brenier and Jaffré [32] and are used in the sequential simulations in this thesis.

*Higher-order flux discretizations*

We have now described the three main ingredients of perhaps the most common discretization in reservoir simulation: finite volumes with two-point flux approximation and single-point upstream mobility weighting. While robust and stable, this method is only first-order accurate. Higher accuracy can be achieved by employing a more sophisticated reconstruction of interface quantities based on a wider stencil, such as TVD methods with slope limiters, see e.g., [81, Chapter 3]. Interface values can also be reconstructed by interpolating between the values in neighboring cells, which is the idea behind the essentially non-oscillatory (ENO) [76] and weighted essentially non-oscillatory (WENO) [39] method. In Chapter 6, we give a short description of WENO methods for unstructured and stratigraphic grids.

### Discretization of source terms

Finite-volume discretizations assume that the volumetric average pressure inside a cell is representative for the entire cell. Even in realistic reservoir models, where grid cells may span tens to hundreds of meters in the lateral direction, this is a reasonable assumption in large parts of the domain. The diameter of a typical well in a subsurface reservoirs is approximately $1/100$ to $1/1000$ times the lateral dimensions of a grid cell. Near these wells, the spatial pressure variations are so large that using the pressure at the wellbore as an approximation of pressure in the entire cell is not realistic. It is therefore common to associate the pressure in a well cell to the bottom-hole pressure $p_{\text{bh}}$, defined as the pressure at a given point in the wellbore.

A well is usually defined by a set consecutive cells in the simulation grid, and the well trajectory is approximated by line segments connecting the cell centroids. We refer to the cells intersected by the well trajectory as perforation cells. In a cell $\Omega_i$ intersected by the well trajectory, the inflow of phase $\alpha$ is modelled as a source term $q_{\alpha,i}$ related to the bottom-hole pressure

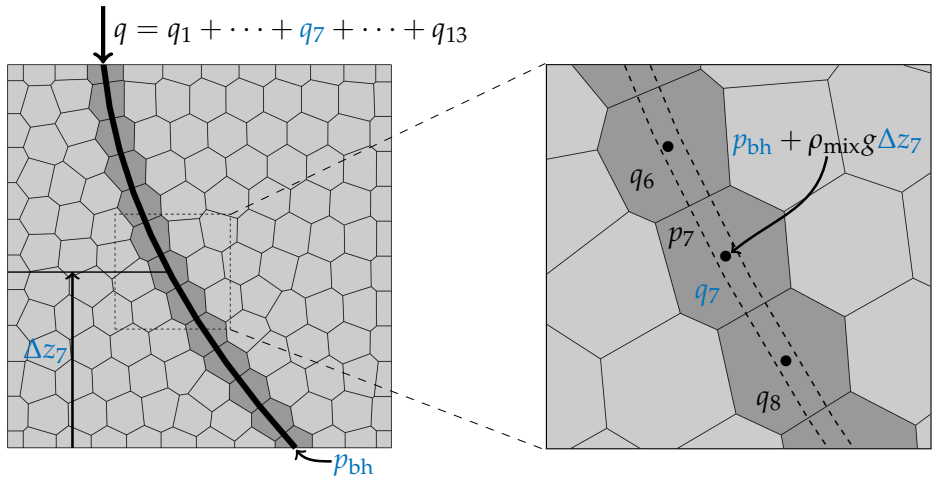



**Figure 3.9:** A sloping vertical well trajectory perforating 13 cells. The inset shows some of the quantities that are used in (3.17) to relate the wellbore pressure in perforation 7 to the average cell pressure $p_7$. Notice that the centroid of perforation 7 is above the reference depth of the bottom-hole pressure, so that $\Delta z_7 < 0$.

by an expression similar to (3.14):

$$q_{\alpha,i} = \begin{cases} W \gamma_\alpha \lambda_{T,i} (p_{\alpha,i} - p_{\text{bh}} + \rho_{\text{mix}} g \Delta z_i) & \text{if } q_{\alpha,i} \geq 0, \\ W \lambda_{\alpha,i} (p_{\alpha,i} - p_{\text{bh}} + \rho_{\text{mix}} g \Delta z_i) & \text{if } q_{\alpha,i} < 0, \end{cases} \tag{3.17}$$

where $\gamma_\alpha$ is the volumetric fraction of phase $\alpha$ in the injected fluid, i.e., if the well injects water only, $\gamma_\alpha = 1$ for $\alpha = a$ and zero otherwise. The mixture density $\rho_{\text{mix}}$ is the density of the fluid mixture in the well, and $\Delta z_i$ is the depth of the centroid in $\Omega_i$, defined relative to the depth at which the bottom-hole pressure is defined. This is used to compute the pressure inside the wellbore in perforation $i$, assuming hydrostatic equilibrium, as $p_{\text{bh}} + \rho_{\text{mix}} g \Delta z_i$. The proportionality constant $W$ is called the well injectivity/productivity index for injection/production wells, and accounts for the difference between the average cell pressure $p_i$ and the wellbore pressure in perforation $i$. In Peaceman-type well models [145, 144], this is commonly computed using an analytical expression, as discussed in Section 2.4. Figure 3.9 shows some of the quantities in (3.17) for a sloping vertical well that perforates 13 cells.

In summary, the source/sink terms are discretized as

$$\int_{\Omega_i} \mathcal{Q}_\alpha^{n+1} \, \mathrm{d}V = |\Omega_i| \rho_{\alpha,i}^{n+1} q_{\alpha,i}^{n+1},$$

where $q_{\alpha,i}$ is given from (3.17), and $\rho_{\alpha,i}$ is simply taken to be the average density in $\Omega_i$.

Wells are commonly either controlled by the bottom-hole pressure $p_{\text{bh}}$, or the injection rate $q_\alpha$. This means that (3.17) introduces one more unknown variable for each well in the system (either $p_{\text{bh}}$ or $q_\alpha$) so that we need an additional equation. This is easily implemented as

$$\mathcal{R}_W = p_{\text{bh}} - \overline{p}_{\text{bh}} \quad \text{or} \quad \mathcal{R}_W = \sum_{i=1}^{n} q_{\alpha,i} - \overline{q}_\alpha,$$

where the first equation corresponds to a well with a target bottom-hole pressure $\overline{p}_{\text{bh}}$, and the second to a well with a target injection rate $\overline{q}_\alpha$. Here, $n$ is the number of perforation cells of the well.

This completes the finite-volume discretization of the conservation equation (3.1), which we can write as

$$\mathcal{R}_{\alpha,i}^{n+1} = \left(\mathcal{M}_{\alpha,i}^{n+1} - \mathcal{M}_{\alpha,i}^{n}\right) + \frac{\Delta t}{|\Omega_i|} \sum_{j \in \mathcal{N}(i)} |\Gamma_{ij}| \vec{\mathcal{F}}_{\alpha,ij}^{n+1} \cdot \vec{n}_{ij} - \mathcal{Q}_{\alpha,i}^{n+1} = 0. \quad (3.18)$$

## 3.4 Challenges in realistic reservoir simulations

Now that we have outlined a finite-volume discretization of the pertinent model equations, we take a moment to look at a few related challenges that arise when we consider realistic reservoir simulation models.

### Why we need implicit discretizations

First, let us justify the choice of implicit temporal discretization that we made in Section 3.3. Consider single-phase flow in a horizontal 1D porous channel aligned with the $x$-axis, with constant velocity $v = 1$. For simplicity, we also assume unit viscosity, $\mu = 1$. Conservation of mass for this problem reads

$$\phi \rho_t + \rho_x = 0. \quad (3.19)$$

Through a transformation to time-of-flight coordinates, $(x, t) \to (\tau, t) = (\phi x, t)$, we can use that $\frac{\partial}{\partial x} = \frac{\partial}{\partial \tau}\frac{\partial \tau}{\partial x} = \phi \frac{\partial}{\partial \tau}$ to write this as

$$\rho_t + \rho_\tau = 0. \quad (3.20)$$

Using the theta rule (3.2) and the spatial discretization outlined above, the discretized residual equation for this problem reads

$$\mathcal{R}_i^{n+1} = \frac{\Delta \tau}{\Delta t}\left(\rho_i^{n+1} - \rho_i^n\right) + \left(\rho_i^{n+\theta} - \rho_{i-1}^{n+\theta}\right) = 0.$$

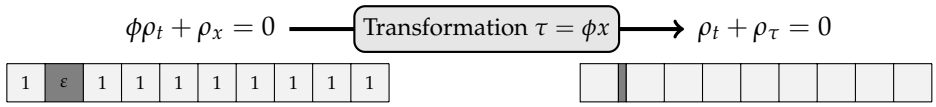



**Figure 3.10:** One-dimensional reservoir in normal (left) and time-of-flight (right) coordinates. The porosity is equal to one in all cells except the second, where it equals $\varepsilon$.

By subtracting the Taylor expansion of the terms in this formula from the Taylor expansion of $\rho_i^{n+1}$ around $\rho_i^n$, we find that the local truncation error of our method is

$$\frac{1}{2}\left(\Delta\tau + [2\theta - 1]\Delta t\right)\rho_{\tau\tau}.$$

Recognizing this as the heat equation, we can argue that our method will smear a discontinuity across a region of width proportional to the root mean square of the Gaussian heat kernel, $\mathcal{O}\sqrt{t(\Delta t \pm \Delta\tau)}$. Here, a minus sign corresponds to an explicit method, whereas a plus sign corresponds to an implicit method.

To understand the practical implications of this, we look an example examined earlier by Mykkeltvedt et al. [137]. Consider the 1D reservoir in Figure 3.10, with unit porosity in all cells except the second, where it equals $\varepsilon$. The reservoir is also shown in time-of-flight coordinates. The total smearing of a discontinuity traveling through this reservoir will be

$$\frac{9}{10}(\Delta x \pm \Delta t) + \frac{1}{10}\varepsilon(\varepsilon\Delta x \pm \Delta t) = \frac{9}{10}\Delta x(1 \pm \varepsilon\nu) + \frac{1}{10}\Delta x\varepsilon^2(1 \pm \nu),$$

where we have written it in terms of the Courant (CFL) number $\nu = \frac{\Delta t}{\varepsilon\Delta x}$. For explicit schemes, $\nu$ should not exceed 1 to ensure stability. Since the computational grid is usually fixed during a simulation, this dictates the timestep. In our example, the cell with porosity $\varepsilon$ governs the choice of timestep. On the other hand, we see from the expression above that the total smearing is dominated by the smearing that accumulates as the discontinuity propagates through the high-porosity region. In other words, even if we choose a timestep that gives a CFL number $\nu \ll 1$, we still have significant smearing of discontinuities. Using an implicit method will not reduce the smearing, but does not introduce any theoretical timestep limitations. In fact, looking at the dominant term, $\frac{9}{10}\Delta x(1 \pm \varepsilon\nu)$, one can argue that $\nu$ should preferably be chosen somewhere between 1 and $1/\varepsilon$ to optimize accuracy versus computational cost.

While this may seem a contrived example, the situation may be even worse in real reservoir models – recall the very large local variations in

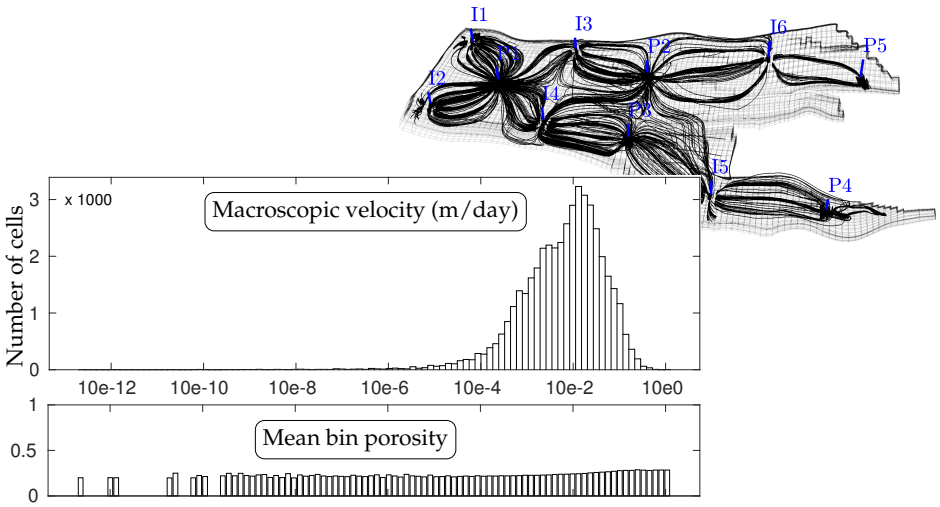



**Figure 3.11:** Magnitude of the macroscopic velocity in a typical injection scenario for the Norne field model. The bar at the bottom shows that the velocity is distributed equally over the entire range of porosities. The well pattern is completely artificial, with five injectors and five producers, and the inset shows well positions and streamlines.

permeability and porosity of the Norne field model in Figure 2.9 and Figure 2.10. The magnitude of the macroscopic velocity in a typical injection scenario, shown in Figure 3.11, varies with almost 13 orders, from practically zero in stagnant regions, to 1 m/day near wells. However, the majority of the cells have a macroscopic velocity that is 1–4 orders of magnitude less than the maximum velocity. Moreover, we see that the velocity is distributed equally across the entire range of porosities. From the analysis above, we can conclude that choosing a small timestep that honors the severe CFL restriction in the high-flow region introduces significant smearing in the majority of the cells. This is the reason why we need implicit temporal discretizations in reservoir simulations.

## Challenging computational grids

As mentioned in Section 3.2, the computational grid should represent the geology and well trajectories of the real reservoir as accurately as possible. As a result, simulation grids tend have cells with very large aspect ratios and contorted geometries. This is particularly true for stratigraphic grids, and requires that the discretizations that we use are sufficiently robust. As

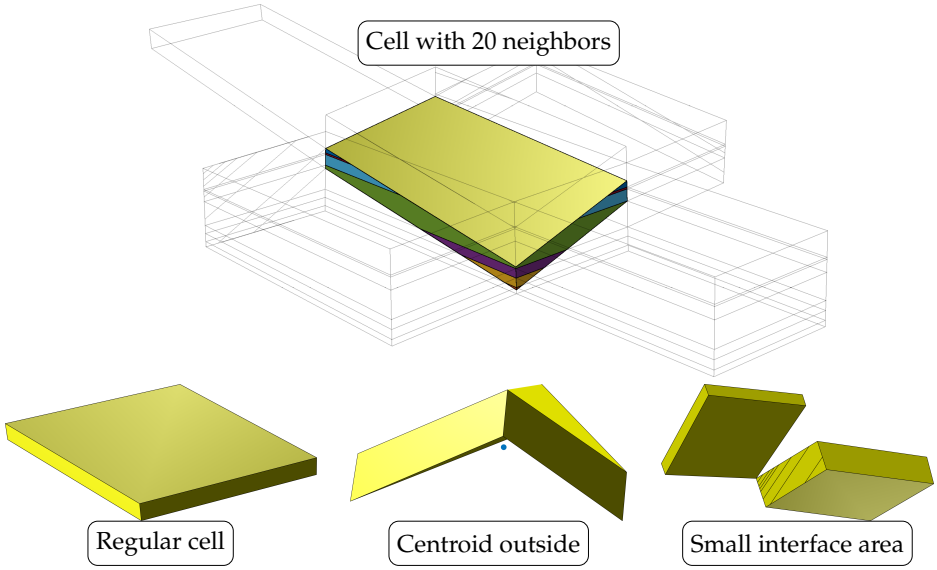



**Figure 3.12:** Different examples of grid cells in the Norne field model: a cell with 20 neighbors ; a regular cell, a cell with its centroid outside the cell volume, and two neighboring cells with a very small interface area. In the latter case, the largest interface area of the right cell is nine orders of magnitude larger than its smallest interface area.

an example, consider the computational grid of the Norne field model. Figure 3.12 shows a few examples of cells. In heavily faulted regions, corner-point grids typically have a complex topology, as can be seen at the top in the figure: this cell has 20 neighbors, each outlined and indicated with a different color at its interface with the center cell. This may also result in very small interface areas, exemplified in the lower-right of the figure. The two cells have a very small common interface, and the right-most cell has nine orders-of-magnitude variation in interface areas. Moreover, since the depth coordinates along the vertical pillars are chosen with geology rather than geometry in mind, the hexahedral cells tend to be skewed, with non-planar faces. An example of this is shown in the bottom center, where the cell centroid lies *outside* the cell. The figure also shows a typical cell in the grid, of approximately $80 \times 80 \times 7$ m$^3$. The standard TPFA and SPU discretizations are very robust, and apart from possible grid orientation effects discussed in Section 3.3, the performance of the resulting scheme seems hardly to be adversely affected by such rough grids. More sophisticated schemes are usually constructed and tested on grids that are much

more regular, both with respect to topology and cell geometries, and tend to have tacit (regularity) assumptions that only appear as limiting factors when you seek to apply the methods on highly complex grid models [116].

This short discussion emphasizes some of the many challenging aspect of reservoir simulations. In addition to the moments outlined thus far, one could also add the need to resolve delicate balances among capillary, gravity, and viscous forces (pressure gradients) that may vary significantly throughout the reservoir and over time; large variations in time constants, in particular for near-incompressible systems; strongly nonlinear and hysteretic functional relationships that may contain kinks or discontinuities; and abrupt and large changes in source terms (or boundary conditions) as wells are turned on or shut in. Altogether, this indicates that even though the basic governing equations are fairly simple, it is not an easy task to construct robust discretizations that can handle all the complexity of industry-grade simulation models. In the next two chapters, we first take a look at methods used to solve the resulting discretized system of equations (3.18) and then briefly introduce the open-source toolbox we have use as a research platform to implement, test, and validate all the methods developed as part of this thesis.

# Chapter 4

# Components of a Reservoir Simulator

In this chapter, we look at different components of a reservoir simulator that complement the discretizations outlined in Chapter 3. This includes discrete operators, nonlinear solvers, linearization schemes, and linear solvers. First, however, we discuss some of the details of three different solution strategies.

## 4.1 Three solution strategies: FI, SI, and SFI

As explained in Section 3.1, the governing equations for flow and transport in porous media can be split into a flow and transport subproblem, where the flow variables tend to be elliptic or parabolic with small time constants, whereas the transport has a hyperbolic character. This is referred to as the sequential implicit (SI) method, in contrast to the fully-implicit (FI) method, where we resolve all variables simultaneously. In addition to a significant reduction in the problem size, a major advantage of the SI method is that we can use specialized solvers to target each subproblem. Because of this, the SI method is a popular approach in academia, but is only used to a limited extent in commercial finite-volume simulators, which typically use FI, IMPES, AIM, and IMPSAT [38, 78] (implicit pressure and saturations, explicit mass fractions). For black-oil type models, the FI method remains particularly popular due to its robustness and its ability to fully resolve couplings between flow and transport.

To ensure convergence of the SI solution towards the FI solution, it is possible to add an outer loop that revisits the flow problem after the trans-

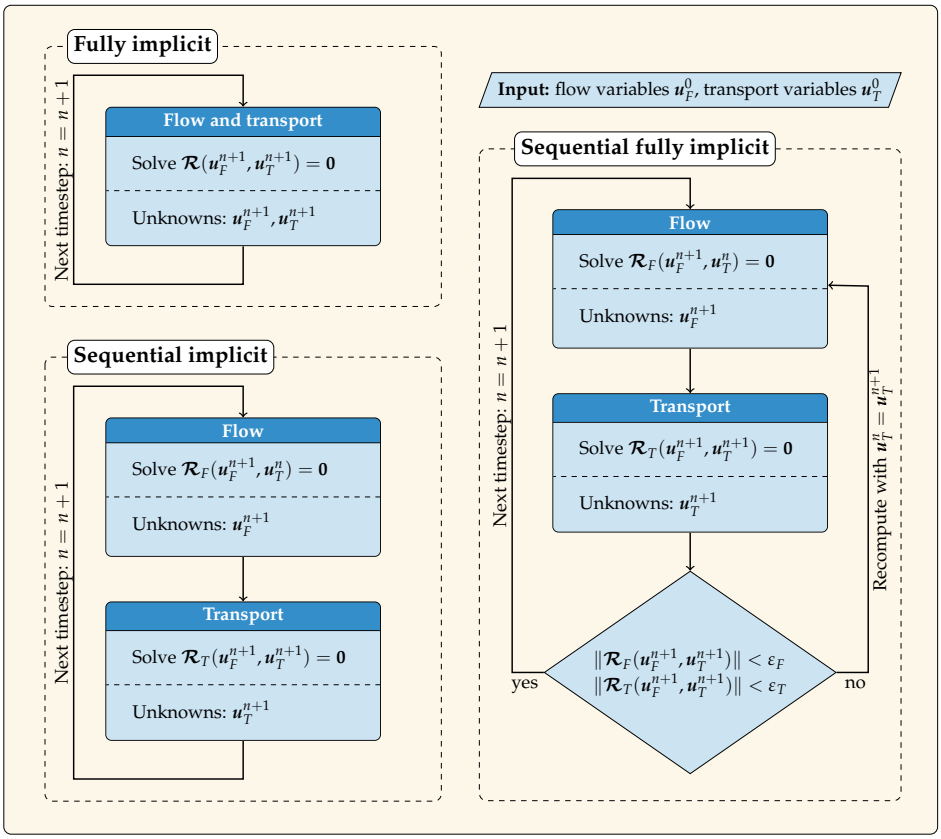



**Figure 4.1:** Schematic overview of three common solution strategies: fully implicit (FI), sequential implicit (SI) and sequential fully implicit (SFI). In each of the process boxes, a nonlinear residual equation is solved for all or a subset of the unknowns. As usual, a superscript $n$ indicates the timestep. The initial state $\boldsymbol{u}^0$ is assumed to be known, and we have used subscript $F$ and $T$ to indicate the flow and transport residual equations and varibles, respectively.

port problem is solved in an iterative fashion, thereby resolving the coupling between the two problems in a sequential fully implicit (SFI) framework [90, 109, 73]. This is, for instance, the default approach in the forthcoming multiscale option of the INTERSECT simulator [99, 115].

Figure 4.1 gives a schematic comparison of the FI, SI, and SFI solution strategies. In this figure, we have written the nonlinear residual equations in vector form, i.e.,

$$\boldsymbol{\mathcal{R}}(\boldsymbol{u}) = \boldsymbol{0}. \tag{4.1}$$

The residual equations $\boldsymbol{\mathcal{R}}$ and the unknowns $\boldsymbol{u}$ we solve for in a given

step of the solution procedure depend on the method we are using: In FI simulations, we solve for all equations and unknowns $\boldsymbol{\mathcal{R}}(\boldsymbol{u}) = \mathbf{0}$, whereas in SI and SFI simulations, we solve for either the flow ($\boldsymbol{\mathcal{R}}_F(\boldsymbol{u}_F) = \mathbf{0}$) or transport ($\boldsymbol{\mathcal{R}}_T(\boldsymbol{u}_T) = \mathbf{0}$). Note that in Figure 4.1, the flow and transport subproblems are written with both flow and transport variables to indicate which variables we solve for and which we keep fixed at each step. We assume for the moment that we know the necessary residual equations on vector form, and will come back to how these are derived later.

Using Taylor expansion around $\boldsymbol{u}$ and assuming that $\boldsymbol{u} + \Delta\boldsymbol{u}$ is a solution to the residual equation, we get that

$$\mathbf{0} = \boldsymbol{\mathcal{R}}(\boldsymbol{u} + \Delta\boldsymbol{u}) = \boldsymbol{\mathcal{R}}(\boldsymbol{u}) + \boldsymbol{J}\Delta\boldsymbol{u} + \mathcal{O}(\Delta\boldsymbol{u}^2),$$

where $\boldsymbol{J} = \partial\boldsymbol{\mathcal{R}}/\partial\boldsymbol{u}$ is the Jacobian of the residual equation. Neglecting higher-order terms gives us the well-known Newton's method[1]:

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \Delta\boldsymbol{u}, \quad -\boldsymbol{J}\Delta\boldsymbol{u} = \boldsymbol{\mathcal{R}}(\boldsymbol{u}^k). \tag{4.2}$$

This equation outlines the iteration scheme of a nonlinear solver. As such, it concisely sums up the main computational steps of a reservoir simulator based on one of the solution strategies just outlined:

1. Computing the discrete residual $\boldsymbol{\mathcal{R}}(\boldsymbol{u})$. This requires a computational grid and temporal and spatial discretizations, which we discussed in Chapter 3. In Section 4.2, we will describe how the single-point upstream mobility-weighted finite-volume discretization with two-point flux approximation can be translated into computer code through discrete operators.

2. Linearizing the residual equations. This involves computing the Jacobian $\boldsymbol{J}$, which is addressed in Section 4.3.

3. Solving the linear system $-\boldsymbol{J}\Delta\boldsymbol{u} = \boldsymbol{\mathcal{R}}(\boldsymbol{u})$. This step often accounts for the majority of the simulation time in reservoir simulation, and an effective linear solver is therefore crucial for efficiency. We look at linear solvers in Section 4.4.

After we have discussed these three integral components, we look at aspects of the nonlinear solver itself in Section 4.5.

[1] Newton's method was devised by Isaac Newton in 1669, first formulated iteratively by Joseph Raphson in 1690, and for systems of nonlinear equations by Thomas Simpson in 1740, so in all fairness, the correct term would be the *Newton-Raphson-Simpson* method. For the sake of brevity, we will, however, simply write *Newton's* method.

## 4.2 Discrete operators

The first step in the nonlinear iteration scheme (4.2) is to express the residual equation (3.18) in vector form

$$\boldsymbol{\mathcal{R}}_\alpha^{n+1} = \frac{1}{\Delta t}(\boldsymbol{\mathcal{M}}_\alpha^{n+1} - \boldsymbol{\mathcal{M}}_\alpha^{n}) + \mathtt{div}(\boldsymbol{\mathcal{F}}_\alpha^{n+1}) - \boldsymbol{\mathcal{Q}}_\alpha^{n+1} = 0. \tag{4.3}$$

As in Chapter 3, we denote the unknown solution variables by $u$, and with some slight abuse of notation, we use $u$ to refer to all unknowns (e.g., pressure, saturation, compositions), or just a subset of them, and we will emphasize the distinction whenever necessary. Since we consider a finite-volume discretization, each solution variable is represented in terms of its cell averages, i.e., by one constant value per cell. The vector form of a variable $u$ is then

$$\boldsymbol{u} = (u_1, \ldots, u_{n_c})^T,$$

and we use $u_i$ to denote the constant value of $\boldsymbol{u}$ in cell $i$. We also introduce the vector of pore-volumes,

$$\boldsymbol{\Phi} = (|\Omega_1|\phi_1, \ldots, |\Omega_{n_c}|\phi_{n_c})^T.$$

For simplicity, we consider immiscible multiphase flow, and write the mass and source terms compactly as

$$\boldsymbol{\mathcal{M}}_\alpha = \boldsymbol{\Phi}\boldsymbol{\rho}_\alpha \boldsymbol{S}_\alpha \quad \text{and} \quad \boldsymbol{\mathcal{Q}}_\alpha = \boldsymbol{\rho}_\alpha \boldsymbol{q}_\alpha.$$

Here, $\boldsymbol{uv}$ should be understood as element-wise vector multiplication so that $(\boldsymbol{uv})_i = u_i v_i$, whereas $\boldsymbol{S}_\alpha$ is the vector form of the cell-wise saturations, $\boldsymbol{\rho}_\alpha$ denotes densities, and $\boldsymbol{q}_\alpha$ volumetric source terms.

To write the flux $\boldsymbol{\mathcal{F}}_\alpha$ in terms of discrete operators, we need a few more definitions. Equivalent to the notation for volumetric flux over interface $ij$ in (3.10), we adopt the notation $\boldsymbol{v}$ for a vector field expressed as a set of interface fluxes, where $v_{ij}$ is the flux from cell $i$ to cell $j$. In this notation, we can write the vector form of the upstream operator in (3.14):

$$\mathtt{upw}(\boldsymbol{u};\boldsymbol{v})_{ij} = \begin{cases} u_i & \text{if } v_{ij} > 0, \\ u_j & \text{otherwise.} \end{cases} \tag{4.4}$$

In addition, we define the discrete divergence operator,

$$\mathtt{div}(\boldsymbol{v})_i = \sum_{j \in \mathcal{N}(i)} v_{ij}. \tag{4.5}$$

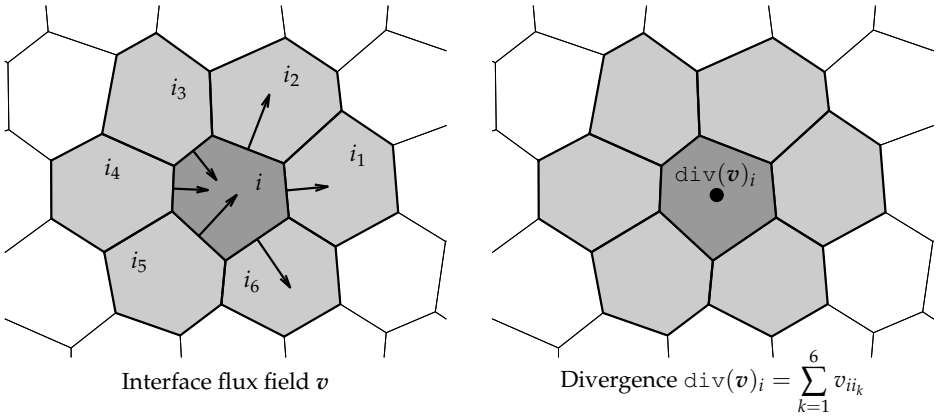



**Figure 4.2:** Illustration of the div operator for a vector field $v$ given as a set of interface fluxes. The operation $\mathtt{div}(v)$ maps $v$ from interfaces to cells, and its value in cell $i$ is the sum of $v$ over all interfaces between cell $i$ and its neighbors $\mathcal{N}(i) = \{i_1, \ldots, i_6\}$.

Figure 4.2 gives an illustration of the div operator for a vector field $v$. These two operators are all we need to write the flux term:

$$\mathtt{div}(\mathcal{F}_\alpha) = \mathtt{div}\left(\mathtt{upw}(\rho_\alpha; v_\alpha) v_\alpha\right). \tag{4.6}$$

Finally, we need two more operators to express the Darcy flux discretization in vector form, namely the gradient and face average operators:

$$\mathtt{grad}(u)_{ij} = u_j - u_i, \quad \mathtt{avg}(u)_{ij} = \frac{1}{2}(u_i + u_j).$$

Figure 4.3 gives an illustration of the gradient operator for a scalar field $u$. The two-point discretization (3.12) of the Darcy flux now reads

$$v_\alpha = -T\mathtt{upw}(\rho_\alpha; -\Pi_\alpha)\Pi_\alpha, \quad \Pi_\alpha = \mathtt{grad}(p - p_\alpha^c) - \mathtt{avg}(\rho_\alpha)\, g\, \mathtt{grad}(z).$$

As with the intercell fluxes, we have used $T$ to denote the vector of interface transmissibilities $T_{ij}$ defined in (3.13).

### The discrete maximum principle

With this notation, it is easy to verify an important property for discretizations of elliptic equations. The solution to the elliptic Poisson equation on the form (2.9) satisfies an elliptic maximum principle, which says that for any subdomain $\Omega$ with no sources or sinks, the pressure $p$ should attain

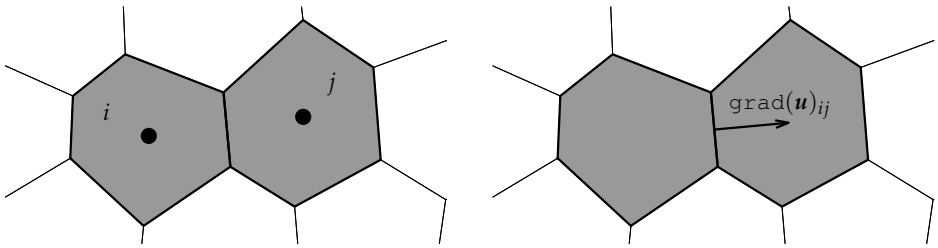



**Figure 4.3:** Illustrations of the `grad` operator for a scalar field $\boldsymbol{u}$. The operator $\texttt{grad}(\boldsymbol{u})$ maps $\boldsymbol{u}$ from cells to interfaces, and its value on interface $ij$ is the difference $u_j - u_i$.

its maximum and minimum on the boundary $\partial\Omega$ (see e.g., [62, Chapter 6]). Without loss of generality, we neglect capillary and gravity effects, and write out the vectorized discrete flux term (4.6) with two-point flux approximation for cell $i$ in Figure 4.2:

$$-\texttt{div}\big(\boldsymbol{T}\texttt{grad}(\boldsymbol{p})\big)_i = T_{ii_1}(p_i - p_{i_1}) + \cdots + T_{ii_6}(p_i - p_{i_6}) = 0.$$

Since all $T_{ii_k}$ are strictly positive by construction, there must be at least one $k \in [1,6]$ so that $p_i < p_{i_k}$, and at least one $\ell \in [1,6]$ so that $p_i > p_{i_\ell}$. If not, the sum would either be positive or negative, which is a contradiction. By repeating the argument for all neighbors $\mathcal{N}(i)$, until we reach the boundary, we can show that TPFA satisfies the *discrete maximum principle*. This fundamental mathematical property is particularly important in multiphase simulations, where violation of the maximum principle may cause convergence issues. We discuss this in further detail in Paper I and Paper III.

To sum up, we can write the single-point, upstream mobility-weighted, finite-volume discretization with two-point flux approximation using four easy-to-implement discrete operators: `div`, `grad`, `upw`, and `avg`. We discretize the residual equations $\boldsymbol{\mathcal{R}}_W$ representing our well models (see Section 3.9) in a similar fashion.

## 4.3 Linearization

Once we have the residual equations expressed in vector form, the next step is linearization. To this end, we write the Jacobian matrix on block-

matrix form as

$$\boldsymbol{J} = \frac{\partial \boldsymbol{\mathcal{R}}}{\partial \boldsymbol{u}} = \begin{bmatrix} \frac{\partial \boldsymbol{\mathcal{R}}_a}{\partial \boldsymbol{u}_a} & \frac{\partial \boldsymbol{\mathcal{R}}_a}{\partial \boldsymbol{u}_\ell} & \frac{\partial \boldsymbol{\mathcal{R}}_a}{\partial \boldsymbol{u}_v} & \frac{\partial \boldsymbol{\mathcal{R}}_a}{\partial \boldsymbol{u}_W} \\ \frac{\partial \boldsymbol{\mathcal{R}}_\ell}{\partial \boldsymbol{u}_a} & \frac{\partial \boldsymbol{\mathcal{R}}_\ell}{\partial \boldsymbol{u}_\ell} & \frac{\partial \boldsymbol{\mathcal{R}}_\ell}{\partial \boldsymbol{u}_v} & \frac{\partial \boldsymbol{\mathcal{R}}_\ell}{\partial \boldsymbol{u}_W} \\ \frac{\partial \boldsymbol{\mathcal{R}}_v}{\partial \boldsymbol{u}_a} & \frac{\partial \boldsymbol{\mathcal{R}}_v}{\partial \boldsymbol{u}_\ell} & \frac{\partial \boldsymbol{\mathcal{R}}_v}{\partial \boldsymbol{u}_v} & \frac{\partial \boldsymbol{\mathcal{R}}_v}{\partial \boldsymbol{u}_W} \\ \frac{\partial \boldsymbol{\mathcal{R}}_W}{\partial \boldsymbol{u}_a} & \frac{\partial \boldsymbol{\mathcal{R}}_W}{\partial \boldsymbol{u}_\ell} & \frac{\partial \boldsymbol{\mathcal{R}}_W}{\partial \boldsymbol{u}_v} & \frac{\partial \boldsymbol{\mathcal{R}}_W}{\partial \boldsymbol{u}_W} \end{bmatrix}, \tag{4.7}$$

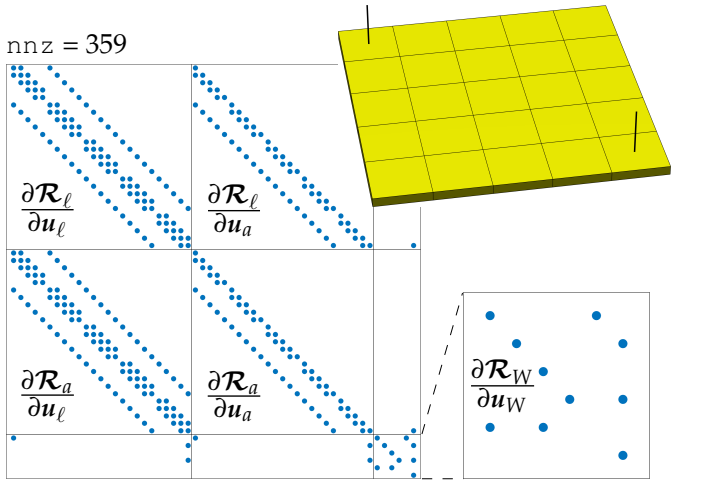
where we have grouped the variables representing reservoir quantities as $\boldsymbol{u} = (\boldsymbol{u}_a, \boldsymbol{u}_\ell, \boldsymbol{u}_v)^T$, and $\boldsymbol{u}_W$ are the well variables (e.g., bottom-hole pressure, water injection rate, etc.). This does not mean that $\boldsymbol{u}_\alpha$ must be a variable that depends directly on phase $\alpha$; a common choice is

$$(\boldsymbol{u}_a, \boldsymbol{u}_\ell, \boldsymbol{u}_v)^T = (\boldsymbol{p}_\ell, \boldsymbol{S}_a, \boldsymbol{S}_v)^T. \tag{4.8}$$

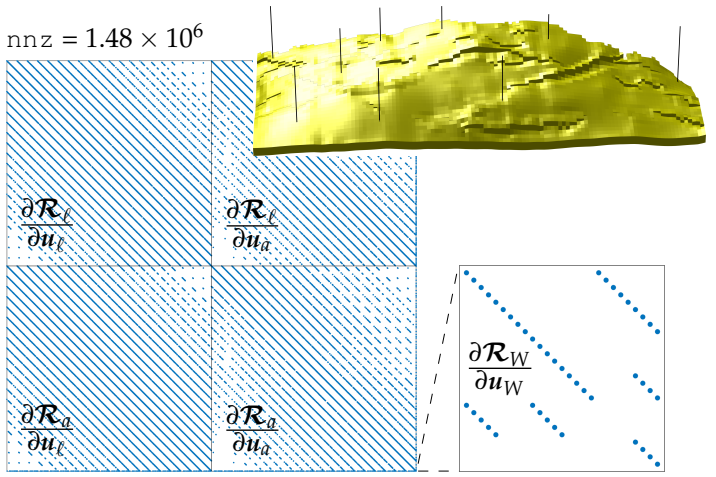
The liquid phase saturation can be found from the closure relation $\boldsymbol{S}_\ell = 1 - (\boldsymbol{S}_a + \boldsymbol{S}_v)$, whereas the aqueous and vapor phase pressures can be found from the capillary function relations $\boldsymbol{p}_c^\alpha = \boldsymbol{p}_\ell - \boldsymbol{p}_\alpha$. We refer to the set of variables $\boldsymbol{u}$ as *primary variables*. Other choices of primary variables are also possible, see, e.g., [111, Chapter 8]. In our finite-volume formulation, element $(\partial \boldsymbol{\mathcal{R}}_\alpha / \partial \boldsymbol{u}_\beta)_{i,j}$ of the Jacobian $\boldsymbol{J}$ is the derivative of residual equation $\boldsymbol{\mathcal{R}}_\alpha$ in cell $i$ with respect to variable $\boldsymbol{u}_\beta$ in cell $j$. Figure 4.4 shows the linearized system from an arbitrary nonlinear step during simulation of water injection in an oil-filled reservoir for two different cases: a $5 \times 5$ Cartesian grid with two wells, and a realization of the SAIGUP model [122]. The latter is a realistic model of a shallow marine oil reservoir with multiple faults, resulting in a complex grid topology.

The Jacobian can generally be calculated in three different ways [197]:

**Manual differentiation.** This involves deriving and hard-coding analytic expressions for the Jacobians, and has been the standard approach in commercial simulators [159]. The obvious advantage of this is that it results in computer code that is both highly efficient and exact. However, calculating Jacobians by hand is generally a time-consuming and error-prone process, in particular when the constitutive relationships involve complex tabulated data and hysteretic behavior. The calculations are even more challenging if we consider higher-order and high-resolution schemes that give wider stencils involving more unknowns. In addition, the method is inflexible: any change to the model equations or discretization requires derivation of new expressions.




$5 \times 5$ Cartesian grid with two wells




SAIGUP model with nine wells

**Figure 4.4:** Sparsity patterns of the linearized system at an arbitrary nonlinear step during the solution of two different cases: a $5 \times 5$ Cartesian grid with two wells, and a realization of the SAIGUP model with nine wells. The first case has 56 unknowns and 359 nonzero Jacobian elements. The second case has 157 467 unknowns and a Jacobian with 1.48 million nonzero elements. Different Jacobian blocks are outlined and labeled. For simplicity, both cases are presented for a simple two-phase liquid-aqueous system. Notice the size of the well blocks relative to the full system.

**Numerical differentiation.** This approach uses finite differences to approximate the derivatives, and is thus much more flexible with respect to changes in the model equations/discretizations. A drawback of this approach is that it will generally not work well with conditional expressions found in upstream operators, slope limiters, etc. In addition, the method is by construction not exact, and the computational complexity tends to limit the efficiency. Moreover, numerical differentiation requires a discretization of the function we want to differentiate. This is not always straightforward, especially since it can have kinks or even discontinuities.

**Automatic differentiation (AD) [141].** The premise of AD is that any computation of numerical quantities can be broken down to a sequence of binary operators like addition, subtraction, multiplication, and division, or evaluation of unary mathematical functions like sin, cos, exp, and log, for which elementary differentiation rules exist. By applying the chain rule repeatedly in combination with known differentiation rules, we can easily compute the derivative of the residual equations with respect to all primary variables. This results in highly flexible and exact Jacobian evaluation. In terms of performance, AD will generally introduce a certain overhead compared to manual differentiation, since the latter allows for simplifications and cancellations not captured by a computer. In this thesis, AD has been extensively used as a tool for Jacobian calculation, and we therefore discuss it in some detail below.

Automatic differentiation can be categorized into two groups: forward and backward AD. Forward AD is the most intuitive, and also the approach commonly used in reservoir simulation. AD has seen much development the last ten years, mainly due to the need for gradient computations in machine learning with neural networks, see e.g., [20]. In reservoir simulation, AD was first introduced in the INTERSECT simulator [51], and later tailored for reservoir simulation through the AD-GPRS simulator [37, 188, 187] using the ADETL library [197, 196, 198], and in the MATLAB Reservoir Simulation Toolbox (MRST) [100, 17, 111, 129].

In practice, AD can be conveniently implement through operator overloading and object-oriented programming with an AD variable class. An instance of this class will have both the function value and the value of the Jacobian of the function. In addition, the class should include operator overloading for fundamental algebraic operators such as plus, minus and multiplication, and of unary mathematical functions like sin, cos, exp, and log, such that each time we perform computations with an AD vari-

able, we also compute its derivatives with respect to all primary variables. Chapter 5 gives a brief introduction to AD in MRST.

AD is a significant contribution to computational mathematics in general, and reservoir simulation in particular: It is what enables us to easily include complex fluid physics with tabulated property functions and employ higher-order discretizations in implicit simulations. However, in simulation of real asset models, which may consist of several million grid cells and fluids with multiple hydrocarbon species, the efficiency of AD may represent a bottleneck. Therefore, recent research has focused on speeding up Jacobian calculations, including exploiting the local nature of the finite-volume, single-point upstream stencil in AD [106], and operator-based linearization [185, 94, 95].

## 4.4 Linear solvers

The blocks $\partial \boldsymbol{\mathcal{R}}_W / \partial \boldsymbol{u}_\alpha$ of the Jacobian in (4.7), corresponding to derivatives of the residual well equations, are typically very small relative to the full system: If $n_W$ is the number of wells, the diagonal block is $n_W \times n_W$, and the coupling blocks are $n_c \times n_W$ or $n_W \times n_c$. Whereas $n_W$ rarely exceeds a few thousand, $n_c$ can be several million. In comparison, all other blocks in the system will be of size $n_c \times n_c$ – much larger than the well blocks. The well equations are therefore usually eliminated after linearization through a Schur complement reduction. This nonetheless leaves us with a linear system that typically involves so many unknowns that using a direct solver is not practical, and iterative solvers are therefore necessary.

Most modern iterative linear solvers are based on representing the solution in a Krylov subspace, which we refer to as Krylov subspace methods. Perhaps the most famous Krylov subspace method is the conjugate gradient method (CG) for symmetric, positive definite linear systems [80]. Linearized systems arising in reservoir simulations are typically not symmetric, and popular methods for such problems include the biconjugate gradient method (BiCG) [104] and its stabilized version Bi-CGSTAB [184], and the generalized minimal residual method (GMRES) [156]. A brief introduction to Krylov subspace methods can be found in [72]. Due to its robustness, GMRES is usually the method of choice in reservoir simulation.

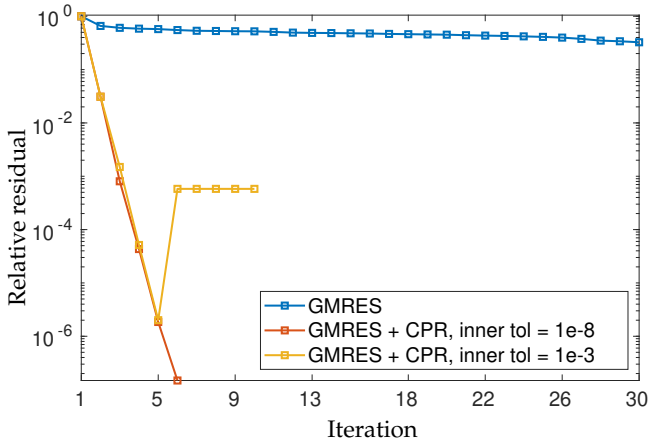



**Figure 4.5:** Convergence history for the solution of a linearized problem from simulation of water injection in the SAIGUP model. GMRES is used with and without CPR preconditioning. Algebraic multigrid (AMG) is used as the elliptic solver in the CPR preconditioner, with a strict ($10^{-8}$) and non-strict ($10^{-3}$) tolerance. We clearly see that preconditioning is necessary for rapid convergence of GMRES. Moreover, in this example, a non-strict convergence of the pressure subproblem in CPR causes the convergence of the full problem to halt.

### Preconditioners

For any iterative linear solver, fast convergence of the method depends strongly on the spectrum of the matrix. Unfortunately, large aspect ratios and orders-of-magnitude variations in petrophysical properties tend to result in linear systems with high condition numbers. Effective preconditioners are therefore crucial [103].

For FI simulations, the mixed elliptic/hyperbolic structure of the underlying PDE carries over to the linearized system, and physics-based preconditioners that exploit this are therefore popular. In particular, the two-stage constrained pressure-residual (CPR) method [189, 190] is based on the reasoning of IMPES methods described in Section 3.1: Given a solution to the flow equation for fluid pressure, discretized with an implicit method, an explicit update of the transport variables yields a meaningful guess for the solution to the full system [103]. Figure 4.5 shows an illustrative example of the convergence history for the solution of a linearized problem from simulation of water injection in the SAIGUP model (Figure 4.4). The problem is solved using GMRES without preconditioning, and with a two-stage CPR preconditioner with a strict and a non-strict tolerance. We clearly see that GMRES converges slowly without preconditioning.

### Multigrid and multiscale methods

If we use a sequential solution procedure, the mixed mathematical character is decoupled at the nonlinear level, and we can use specialized linear solvers for the flow and transport problems. In particular, the elliptic character of the linearized flow system indicates that we should use an efficient elliptic solver, standalone or as a preconditioner for e.g., GMRES.

Algebraic multigrid methods (AMG) [181] systematically reduce the residual by iteratively solving the linear system at different coarse scales. This is done by means of a restriction operator $\mathcal{R}$ that restricts the fine-scale problem to a coarser scale, and a prolongation operator $\mathcal{P}$ that prolongates the problem back to the fine scale. Algebraic multigrid methods use a hierarchy of coarse scales with decreasing number of degrees of freedom. The rationale behind this is that for elliptic problems, high-frequency error components can be effectively reduced by one or more iterations of a so-called smoother. Once the high-frequency errors are reduced on a given grid resolution, the remaining intermediate and low-frequency error components are projected onto a coarser grid, where some of these components become high-frequency component and can be reduced by smoothing, and so on. Popular smoothers include incomplete LU-factorization with zero fill-in, and Gauss-Seidel and Jacobi variants. Multigrid methods have been successfully applied to reservoir simulation [103, 172, 69], and can also be used in FI simulations [103, 70].

Multiscale methods were initially developed as an alternative to upscaling (see Section 2.3) that more accurately captures local features in the fine-scale model [85]. During the last two decades, multiscale methods for Poisson-type equations with strongly varying coefficients have seen a massive development. Among the most notable contributions relevant to reservoir simulation are the multiscale finite-volume (MsFV) [88, 89, 90], the multiscale mixed finite-element (MsMFE) [1, 2, 3], and the multiscale restriction-smoothed basis (MsRSB) [131, 132, 133] methods. The first multiscale method to be implemented and fully integrated in a commercial simulator was MsMFE, which was implemented in the FrontSim streamline simulator [140]. MsFV was implemented in the prototyping branch of INTERSECT, but has later been superseded by MsRSB, which is planned to be commercially available within a year. Lie et al. [115] gives a more comprehensive review of multiscale methods applicable to real reservoir simulation.

In their original form, multiscale methods were developed as approximate pressure solvers that capture the essential features of the fine-scale pressure using significantly fewer degrees of freedom. Consider a fine grid

with $n_c$ cells $\{\Omega_i\}_{i=1}^{n_c}$ that incorporates all the fine-scale details of the underlying geological model and a corresponding linearized pressure equation

$$\boldsymbol{A}\boldsymbol{p} = \boldsymbol{q}. \tag{4.9}$$

A finite-volume-type, algebraic multiscale method for this problem can be formulated by defining a coarse partition $\{\Omega_i^c\}_{i=1}^m$ where each fine cell $\Omega_j$ belongs to a single coarse block $\Omega_i^c$. The restriction operator $\mathcal{R} : \{\Omega_j\} \rightarrow \{\Omega_i^c\}$ is linear and can be written as a matrix $\boldsymbol{R} \in \mathbb{R}^{m \times n_c}$, where row $i$ of $\boldsymbol{R}$ is the $i$th basis function of $\mathcal{R}$, which takes on the constant value $R_{i,j}$ in cell $j$ of the fine-scale grid. Analogously, the prolongation operator $\mathcal{P} : \{\Omega_i^c\} \rightarrow \{\Omega_j\}$ can be written in matrix form $\boldsymbol{P}$, with basis function $i$ in column $i$ of $\boldsymbol{P}$. The basis functions of $\boldsymbol{P}$ are computed numerically by solving a localized flow problem (MsFV) or by localized algebraic smoothing (MsRSB).

The operators $\mathcal{R}$ and $\mathcal{P}$ can now be used to reduce the linear system: Instead of searching for a fine-scale solution in $\mathbb{R}^{n_c}$, we seek an approximate solution $\boldsymbol{p}$ that can be written as a prolongation of a solution defined on $\{\Omega_i^c\}$; that is, $\boldsymbol{p} = \boldsymbol{P}\boldsymbol{p}_c$. Inserting this into (4.9) and multiplying both sides by the restriction operator gives

$$(\boldsymbol{RAP})\boldsymbol{p}_c = \boldsymbol{Rq}, \quad \text{or} \quad \boldsymbol{A}_c\boldsymbol{p}_c = \boldsymbol{q}_c. \tag{4.10}$$

Once this reduced system is solved for the coarse-scale pressure $\boldsymbol{p}_c$, we prolongate the solution onto the fine grid to obtain an approximate fine-scale solution $\boldsymbol{p} = \boldsymbol{P}\boldsymbol{p}_c$. Figure 4.6 illustrates the algebraic multiscale method.

This framework is very similar to AMG, and there is nothing preventing us from combining it with a smoother to obtain an iterative framework equivalent to a two-level AMG solver. This approach has been taken by a number of authors, (see e.g., [192, 132]), and has also been extended to use multiple pairs of restriction and prolongation operators that are not necessarily organized hierarchically as in a multigrid method, but instead each targets specific features in the geological model [114]. This iterative multibasis multiscale method is conceptually equivalent to multigrid methods, but philosophically different due to the purely physical/geometrical interpretation. Recently, efforts have also been made to employ MsFV operators in AMG [59].

Paper IV describes how linear convergence for the pressure subproblem can be accelerated by using multiple multiscale operators that each targets specific features in the geological model and/or adapts to changes in saturation and pressure. This is used as the pressure solver in a CPR preconditioner.

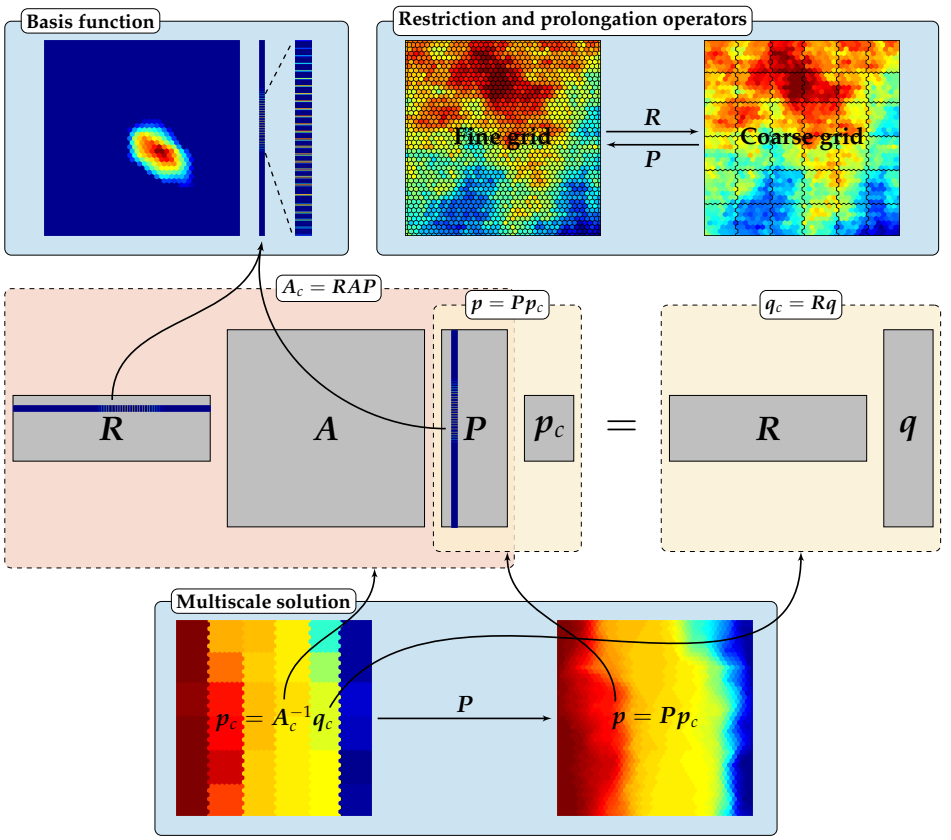



**Figure 4.6:** Algebraic multiscale solvers reduce the fine-scale linearized pressure equation to a much smaller system through the restriction and prolongation operators $\boldsymbol{R}$ and $\boldsymbol{P}$, as seen in the middle section. The upper left box shows an example of an MsRSB basis function, both in physical space and as a row or column vector in $\boldsymbol{R}$ and $\boldsymbol{P}$. The solution to the fine-scale system is approximated by solving the reduced system and prolongating the solution $\boldsymbol{p}_c$ back to the fine scale. The figure is a modified version of one in Paper IV.

## 4.5 Nonlinear solvers

Newton's method outlined in Section 4.3 is known to offer quadratic convergence when the iterates are close to the solution. However, this property seldom carries over to real cases, where large timesteps combined with physical processes taking place on different temporal scales imply that the initial guess is generally *not* close to the solution. This means that Equation (4.2) is somewhat simplified compared to Newton methods commonly used in reservoir simulators, where *global* Newton methods are usually

needed [53]. In this section, we look at different nonlinear solution strategies for (4.1).

### Damping strategies

It is well-known that Newton's method (4.2) will experience convergence issues when the update $\Delta \boldsymbol{u}$ passes inflection points, kinks, and discontinuities in the residual function. This has motivated a family of methods that aim to find damping factors $\{\theta_i\}_{i=1}^N$ to ensure that the Newton updates are not too large,

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \boldsymbol{\Theta}\Delta \boldsymbol{u}, \quad \boldsymbol{\Theta} = \text{diag}(\theta_1, \dots, \theta_N). \tag{4.11}$$

Here, $\boldsymbol{\Theta} = \boldsymbol{I}$ gives the standard Newton's method. Criteria for what is a too large update can either be physically/mathematically motivated, or be based on simple heuristics. A popular method that falls into the category of physically-motivated methods is called Appleyard chopping [159], in which updates are dampened in cells where the update results in a phase going from mobile to immobile and vice versa, or when a phase appears/disappears. In either case, saturation updates are dampened so that they are close to the corresponding critical point. This method is typically combined with heuristics such as limiting the absolute and/or relative saturation update to, e.g., be less than 0.2. The latter relative update limit is commonly referred to as *geometric penalty*. These heuristics can also be used for the temperature variable in simulations involving thermal effects. Likewise, in cases with strong capillary pressure effects, one may wish to limit changes in capillary pressure.

As pointed out by Jenny et al. [91], convergence issues in the nonlinear transport solver often arise from the S-shaped form of the flux functions $f_\alpha$ defined in (3.5). In particular, inflection points and kinks caused by upstream changes may send the Newton iterates to different contraction regions, which leads to oscillations or even divergence. This has spawned a family of so-called *trust-region methods* in which the damping factors $\theta_i$ are chosen based on these problematic points [91, 191, 110, 186, 130]. Trust region methods offer unconditional convergence for any timestep length, but may require a significantly higher number of expensive residual evaluations than the standard Newton's method. In addition, the damping factors tend to be overly restrictive. In Paper V and Paper VI, we outline a trust-region framework that improves performance by introducing adaptivity in the trust-region algorithm. Figure 4.7 shows different damping strategies applied to the same single-cell problem, including line search with backtracking. The example is taken from [130].

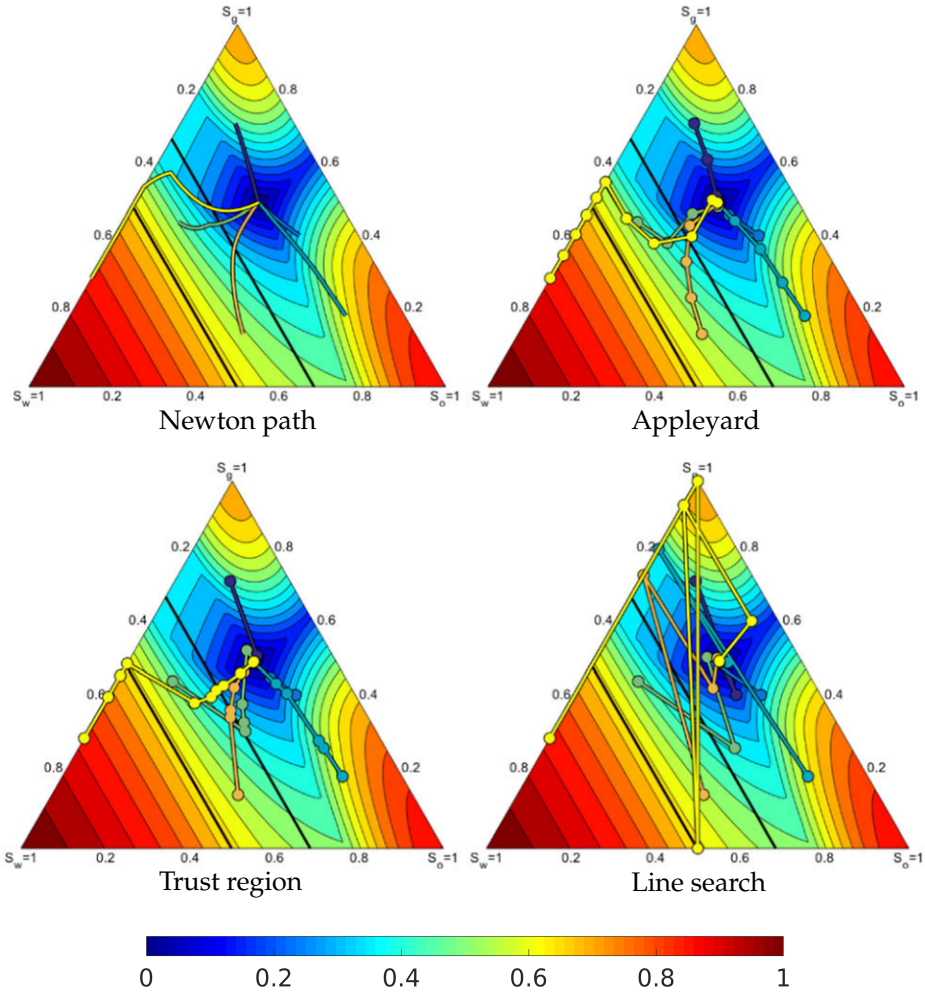



**Figure 4.7:** Different damping strategies applied to a single-cell transport problem with different initial guesses. The contour plot shows the normalized residual surface, with thick black lines indicating changes in the upstream direction due to gravity. The continuous Newton path is approximated by limiting the absolute value of the Newton update $\Delta \boldsymbol{u}$ by 0.001. Appleyard chopping with $\boldsymbol{\Theta} = 0.1$ converges, but dampens many of the iterates more than necessary. Using trust regions gives fast and accurate convergence, whereas line search converges, but passes inflection points and several upstream changes for some of the initial guesses, resulting in oscillations. Figure from [130].

The hyperbolic nature of the transport subproblem means that transport quantities will have finite speed of propagation. A consequence of this is that the number of cells a propagating displacement front passes through during a timestep will be limited by the timestep length $\Delta t$. Large $\Delta t$ therefore tends to be challenging for the nonlinear solver. It is, however, generally hard to predict the largest possible timestep we can take without compromising nonlinear convergence. A common practice is therefore to use a pragmatic "whatever-works-approach" to this problem: If the nonlinear solver does not converge for a given timestep after a prescribed maximum number of iterations, the timestep is reduced, and the solver starts over. Whereas this works well for most simulation scenarios, it may result in a large amount of wasted computational effort, since we simply discard any information we get from the iterations of timesteps that do not converge. Heuristic timestep selection methods aim to estimate the maximum timestep possible based on a target Newton update size, see, e.g., [37].

### Localized nonlinear solvers

The local nature of hyperbolic conservation laws has an important implication for the nonlinear solver: During a timestep, updates will be small, and even zero, in large parts of the domain, except from near propagating fluid fronts, and in regions with large pressure updates, e.g., in the vicinity of wells. This means that if we iterate the Newton method the same number of steps in all cells, we are wasting substantial computational efforts by solving for a large number of zeros. It also means that different parts of the domain may experience very different nonlinear convergence. This is particularly true in real reservoir models, where fluid transport mainly takes place in the drainage regions of the wells, whereas potentially large parts of the domain essentially remain unchanged throughout the simulation. A number of methods exploit this locality: one family of methods is based on a priori identifying a support region inside which updates are nonzero [167, 166]. Another family of methods are based on reordering the grid cells so that the transport subproblems can be solved cell-by-cell. This approach has been the focus of much of the research in this thesis, and we therefore describe it in some detail here.

#### *Reordering*

Due to the upstream discretization (4.4), the neighbors $\mathcal{N}(i)$ of a grid cell $\Omega_i$, and consequently also the discrete divergence operator (4.5), can be

split into upstream $\mathcal{U}(i)$ and downstream $\mathcal{D}(i)$ cells:

$$\mathtt{div}(\boldsymbol{v})_i = \sum_{j \in \mathcal{N}(i)} v_{ij} = \sum_{j \in \mathcal{U}(i)} v_{ij} + \sum_{j \in \mathcal{D}(i)} v_{ij}. \tag{4.12}$$

Moreover, it follows directly from the definition of the upstream operator that

$$\mathtt{upw}(\boldsymbol{u}; \boldsymbol{v})_{i,j} = \begin{cases} u_j, & \text{if } j \in \mathcal{U}(i), \\ u_i, & \text{if } j \in \mathcal{D}(i), \end{cases} \tag{4.13}$$

and we can therefore write

$$\mathtt{div}(\boldsymbol{\mathcal{F}}_\alpha)_i = \mathtt{div}(\mathtt{upw}(\boldsymbol{\rho}_\alpha; \boldsymbol{v}_\alpha)\boldsymbol{v}_\alpha)_i = \sum_{j \in \mathcal{U}(i)} \rho_{\alpha,j} v_{\alpha,ij} + \sum_{j \in \mathcal{D}(i)} \rho_{\alpha,i} v_{\alpha,ij}.$$

Since $\mathtt{div}(\boldsymbol{\mathcal{F}}_\alpha)_i$ is the only term in the residual $\boldsymbol{\mathcal{R}}_{\alpha,i}$ that depends on values in neighboring cells, this means that if the values in all upstream cells $\mathcal{U}(i)$ are known, all unknowns in $\boldsymbol{\mathcal{R}}_{\alpha,i}$ are local to cell $i$. In this case, $\boldsymbol{\mathcal{R}}_{\alpha,i} = \mathbf{0}$ can be solved as a single-cell problem.

This was first observed by Appleyard and Cheshire [12], who proposed to use it as a means of accelerating Newton's method for fully implicit problems. Termed the *Cascade method*, one nonlinear iteration of the method consists of solving the fully-implicit linearized system to obtain updated pressures and intercell fluxes. The method then sweeps through the grid cells in the order of decreasing pressure potential, updating the transport variables cell-by-cell with fixed pressure. As the sweep progresses, the number of cells with known upstream values increases rapidly, hence the name.

In the absence of capillary and gravity forces, the flow will be cocurrent, meaning that all phases will flow in the same direction. This gives the same pressure potential for all phases. After ordering the nonlinear transport equations from 1 to $n_c$ based on this potential, the nonlinear system of discrete transport equations takes a lower-triangular form:

$$\begin{bmatrix} \boldsymbol{\mathcal{R}}_1(S_1) & & \\ \boldsymbol{\mathcal{R}}_2(S_1, S_2) & & \\ \vdots & \ddots & \\ \boldsymbol{\mathcal{R}}_{n_c}(S_1, S_2, \ldots, S_{n_c}) & & \end{bmatrix} = \mathbf{0}.$$

When gravity/capillary forces are present, fluids may exhibit countercurrent flow, which means that the potential ordering will be different for different phases. In this case, it is not possible to permute the transport

equations into lower-triangular form. The authors instead suggested to perform multiple sweeps of the Cascade method, using it as a nonlinear Gauss–Seidel solver. Note, however, that this method will not work if the pressure contains local minima: Such a cell would have only inflow fluxes, so that solving transport for all phases in this cell simultaneously is generally not possible. Even if we use a two-point flux approximation, which we saw in Section 4.2 satisfies the discrete maximum principle, there is no guarantee that the pressure is monotone before convergence.

Although promising results were reported, the Cascade method remained a rarity in the reservoir simulation community for almost 35 years. A possible reason for this is that most available simulators lack the required flexibility to implement the method. Kwok and Tchelepi [102] addressed the shortcomings of the Cascade method, devising a reordering strategy for black-oil models with capillary and gravity effects. The method uses one reordering per fluid phase. In black-oil models, relative permeability of the water only depends on the water saturation $S_w$, whereas the oil relative permeability depends on $S_w$ and $S_o$ (see Section 2.2). Therefore, the cells are first swept according to descending water potential, solving the single-cell transport problems for the water phase. Next, the cells are swept in order of descending oil potential, solving the transport subproblems for the oil phase. The gas equations are eliminated using the closure relation $S_w + S_o + S_g = 1$. This corresponds to permuting the nonlinear transport equations into the following lower-triangular form:

$$\begin{bmatrix} \boldsymbol{\mathcal{R}}_{a,1}(S_{a,1}) & & & & & & \\ \boldsymbol{\mathcal{R}}_{a,2}(S_{a,1}, S_{a,2}) & & & & & & \\ \vdots & \ddots & & & & & \\ \boldsymbol{\mathcal{R}}_{a,n_c}(S_{a,1}, S_{a,2}, \ldots, S_{a,n_c}) & & & & & & \\ \boldsymbol{\mathcal{R}}_{\ell,1}(S_{a,1}, S_{a,2}, \ldots, S_{a,n_c}, S_{\ell,1}) & & & & & & \\ \boldsymbol{\mathcal{R}}_{\ell,2}(S_{a,1}, S_{a,2}, \ldots, S_{a,n_c}, S_{\ell,1}, S_{\ell,2}) & & & & & & \\ \vdots & \ddots & & & \vdots & \ddots & \\ \boldsymbol{\mathcal{R}}_{\ell,n_c}(S_{a,1}, S_{a,2}, \ldots, S_{a,n_c}, S_{\ell,1}, S_{\ell,2}, \ldots, S_{\ell,n_c}) & & & & & & \end{bmatrix} = \mathbf{0}.$$

With capillarity, the water pressure must be chosen as the pressure unknown, as this is the only choice that preserves the upstream dependence of the transport residual equations for water. This approach is tailored for black-oil models and will generally not work for compositional models, since conservation of mass for a component generally depends on all phase saturations.

Natvig and Lie [138] suggested a different and more general approach that sorts grid cells based on the intercell fluxes. In particular, intercell

fluxes obtained by solving the flow subproblem form a directed graph $(\{\Omega_i\}, \{v_{\alpha,ij}\})$, where each grid cell $\Omega_i$ represents a node $i$, and the inter-cell flux $v_{\alpha,ij}$ represent directed edges; that is, $v_{\alpha,ij}$ connects node $i$ to node $j$ if $v_{\alpha,ij} > 0$, and connects node $j$ to node $i$ if $v_{\alpha,ij} < 0$. In cases with only cocurrent flow (i.e., if all phases flow in the same direction), this will be a directed and acyclic graph (DAG). As for potential-based ordering, we can then solve single-cell transport problems by traversing this graph in topological order. With capillary or gravity forces, the flow across an interface may go in opposite directions for different fluid phases (countercurrent flow). This introduces connected components in the graph, corresponding to cycles of mutually dependent cells that must be resolved simultaneously. Cycles can also arise if pressure is computed by a consistent but nonmonotone method.

A topological ordering is still possible if we first group all cells belonging to the same connected component into a super-node, which results in a new DAG with fewer nodes. By traversing the nodes in topological order, the transport subproblem can be solved cell-by-cell for nodes or cycle-by-cycle for supernodes. Assuming that node $i$ in the topological ordering consists of three cells $i_1, i_2, i_3$, the system of nonlinear transport equations takes the following form

$$\begin{bmatrix} \boldsymbol{\mathcal{R}}_1(S_1) & & & & & & & \\ \boldsymbol{\mathcal{R}}_2(S_1, S_2) & & & & & & & \\ \vdots & \ddots & & & & & & \\ \boldsymbol{\mathcal{R}}_{i-1}(S_1, S_2, \ldots, S_{i-1}) & & & & & & & \\ \boldsymbol{\mathcal{R}}_i(S_1, S_2, \ldots, S_{i-1}, S_{i_1}, S_{i_2}, S_{i_3}) & & & & & & & \\ \boldsymbol{\mathcal{R}}_{i+1}(S_1, S_2, \ldots, S_{i-1}, S_{i_1}, S_{i_2}, S_{i_3}, S_{i+1}) & & & & & & & \\ \vdots & \ddots & \vdots & & & & \ddots & \\ \boldsymbol{\mathcal{R}}_{n_n}(S_1, S_2, \ldots, S_{i-1}, S_{i_1}, S_{i_2}, S_{i_3}, S_{i+1}, \ldots, S_{n_c}) & & & & & & & \end{bmatrix} = \mathbf{0}.$$

When solving this problem, residual equation $i$ contains three unknowns $S_{i_1}, S_{i_2}, S_{i_3}$ that must be solved for simultaneously. Note that $n_n < n_c$ denotes the number of nodes in the graph, and not the number of grid cells. Figure 4.8 illustrates the method for a small PEBI grid with gravity effects, where a light vapor phase is injected into a heavy liquid phase.

Computing the topological ordering is called a topological sort, and can be performed in $\mathcal{O}(n_c)$ operations in the case of only cocurrent flow using, e.g., a depth-first search, (see [47, Chapter 22]). With counter-current flow, a topological order can be obtained in $\mathcal{O}(n_c + n_f)$ operations using Tarjan's algorithm [176], where $n_c$ is the number of cells and $n_f$ the number of interfaces in the grid. This approach has been successfully applied to the

transport subproblem in sequentially implicit solution schemes for tow-phase flow [138], two-phase, three-component systems describing polymer flooding [113, 150], and for linear transport equations for time-of-flight and steady-state tracer distributions [139, 60, 149]. Figure 4.8 shows an illustrating 2D example of reordering based on the topological flux graph, where fluid flows from an injector in the lower left corner to a producer in the upper right corner.

Use of optimal ordering may speed up the simulation of transport equations significantly, since it offers local control of the nonlinear solution process. In particular, computational resources are spent where updates are nonzero, whereas large parts of the domain requires zero or a single nonlinear iteration to converge. All discretization schemes for which the stencil is restricted to the upstream neighbors in addition to the cell itself fits into the reordering method. As such, the increased computational efficiency obtained by reordering also facilitates the use of otherwise prohibitively expensive higher-order dG methods, se e.g., [139, 138, 149] and Papers VII and VIII.

Papers VII to IX discuss localized nonlinear solvers based on reordering for black-oil (Paper VII) and compositional (Paper VIII) problems and how this approach can be combined with dynamic spatial discretizations (Paper IX).

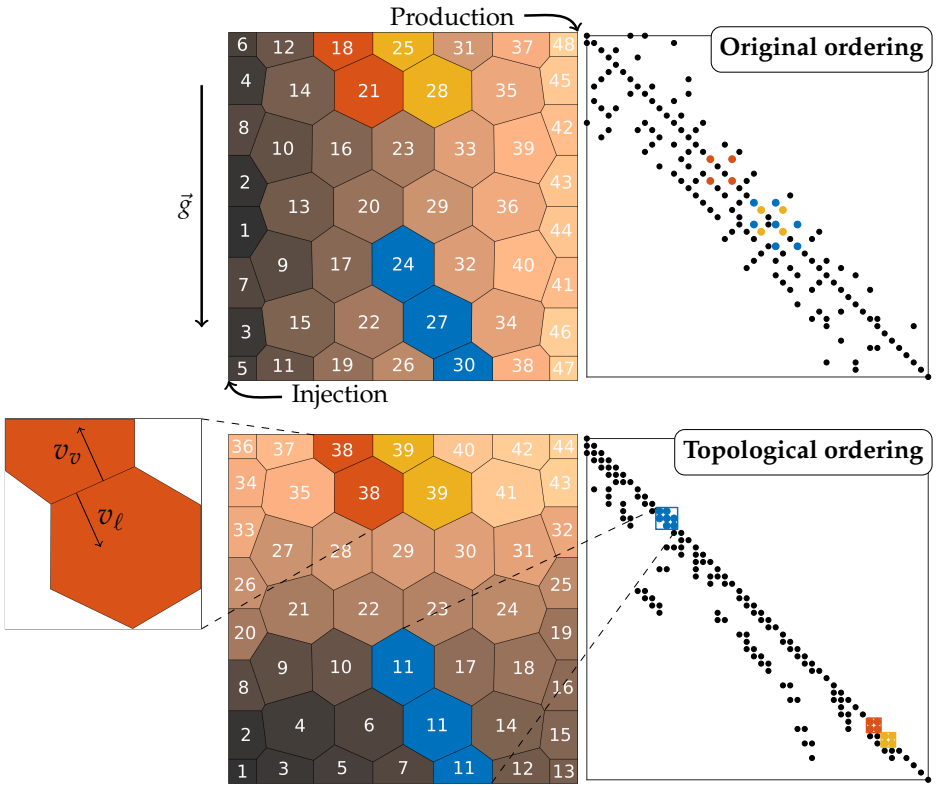



**Figure 4.8:** Topological ordering of the grid cells for a case with gravity, posed in a PEBI grid filled with a heavy liquid phase. A light vapor phase is injected in the lower-left corner, and flows towards a producer in the upper-right corner. The original ordering runs from left to right, whereas the topological ordering starts at the injector, and increases towards the producer. The figure also shows the structure of the system of nonlinear transport equations as sparsity plots, both in the original and topological ordering. Buoyant forces introduce three connected components of mutually dependent cells, indicated by a unique color in the grids and corresponding sparsity plots. Reordering, permutes the nonlinear transport system into lower-triangular form, with three small diagonal blocks corresponding to cycles. The inset shows one of the cycles, with the direction of the vapor and liquid fluxes imposed. Figure from Paper IX.

# Chapter 5

# The MATLAB Reservoir Simulation Toolbox

A continuous drive towards more complex reservoirs means that reservoir simulation becomes an increasingly important tool for reservoir engineers. There exist a number of commercial reservoir simulators: ECLIPSE 100/300 [160] and INTERSECT [161] from Schlumberger, the GPU-based ECHELON simulator from Stoneridge Technology [171], tNavigator from Rock Flow Dynamics [154], Nexus from Landmark [105], and IMEX, GEM and STARS from Computer Modelling Group [42], to name a few. In addition, large companies like ExxonMobil and Saudi-Aramco have commercial in-house simulators.

Commercial reservoir simulators are typically implemented in an optimized manner to meet the robustness and efficiency commonly required for industrial applications. This means that they often lack the flexibility that makes it simple to implement and test new gridding and discretization methods, new solution strategies, etc. More important, source code is hardly ever available to users. Therefore, some research groups develop open-source reservoir simulation software to facilitate rapid development of new ideas. This also contributes to bridge the gap between industry and academia, and reduce the time from conception to implementation of new ideas in an industrial simulator environment. There exists several excellent open-source software resources for reservoir simulation, and review of these is clearly outside the scope of this thesis. Instead, we refer to e.g., [27], and mention here the two open-source software packages used in this thesis: the Open Porous Media Initiative (OPM) [143], collaboratively developed by NORCE, SINTEF, Equinor, and others; and the MATLAB Reservoir Simulation Toolbox (MRST) [169], developed by SINTEF. The latter

has been extensively used in this work, and we therefore describe it here in some detail.

## 5.1 Geological model

MRST is organized in modules and offers a wide variety of tools. Here, we go through an example to show the basic steps necessary to set up a simulation model. The first step is to construct a grid and populate it with petrophysical properties.

We want to construct a 2.5D PEBI grid, and start by loading the `upr` module originating from the master thesis of Berge [26], which is also described in detail in Paper I and later improved in Paper II.

```
mrstModule add upr; % Load upr module for PEBI gridding
```

We first construct an areal $1000 \times 500$ m$^2$ grid with an injector in $(50, 50)$ m and a producer in $(950, 450)$ m and increased resolution around the wells.

```
lx = 1000*meter; % Length in x direction
ly = 500*meter;  % Length in y-direction
n  = 20;         % Approximate number of cells in x-direction

inj  = [ 50, 50]*meter; % Injector coordinates
prod = [950,450]*meter; % Producer coordinates

G2D = pebiGrid(lx/n, [lx, ly]    , ...
    'wellLines'     , {inj, prod}, ... % Wells
    'wellGridFactor', 0.25       , ... % Relative well cell size
    'wellRefinement', true       );    % Refinement around wells
```

MRST offers a variety of plotting functionality, so we plot the resulting `G2D` grid. All cells have a Boolean tag, which `pebiGrid` sets to true for all well cells. Figure 5.1 shows the result.

Next, we make a 2.5D grid by stacking five layers of the areal grid on top of each other using `makeLayeredGrid`:

```
nl = 5;                                    % Number of layers
dz = 5*meter;                              % Layer thickness
G  = makeLayeredGrid(G2D, dz*ones(nl,1));  % Make layered grid
G.cells.tag = repmat(G2D.cells.tag, nl,1); % Repeat well cell tag
G  = computeGeometry(G);                   % Compute geometry
```

We then populate the grid with petrophysical properties. To mimic stratigraphic layers, we use `logNormLayers` to construct a 3D array of scalar

```
plotGrid(G2D, 'facecolor', 'none')              % Plot grid cell outline
plotGrid(G2D, G2D.cells.tag, 'facecolor', 'r') % Plot only well cells
axis equal tight
```

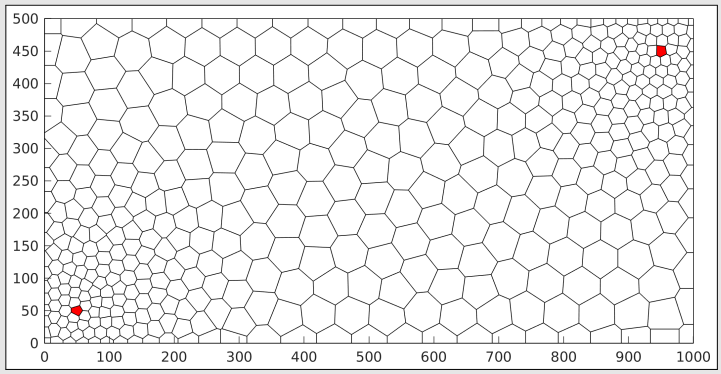



**Figure 5.1:** 2D PEBI grid with refinement around well cells (shown in red).

permeability values, lognormally distributed with a different mean in each layer. We then sample values from this array onto the 2.5D grid.

```
K    = logNormLayers([2*n, 2*n/2, nl], nl);             % Generate
perm = sampleFromBox(G, reshape(K, [2*n, 2*n/2, nl])); % Sample
```

This is used to construct a `rock` structure, where we for simplicity assume that the porosity is directly correlated to the permeability:

```
perm = perm*milli*darcy;         % Assign units
poro = perm/max(perm)*0.5;       % Correlated porosity
rock = makeRock(G, perm, poro); % Make rock structure
```

Finally, we use the built-in function `peaks` to shift the vertical coordinates.

```
x = G.nodes.coords(:,1:2);                          % Map xy-coords ...
x = bsxfun(@rdivide, x,[lx,ly])*4 - 4/2;            % to peaks' domain
z = peaks(x(:,1), x(:,2))*5 + 2000*meter;           % Compute z coords
G.nodes.coords(:,3) = G.nodes.coords(:,3) + z; % Assign z coords
G = computeGeometry(G);                             % Compute geometry
```

This alters the grid geometry, and we thus called `computeGeometry(G)` once more. Figure 5.2 shows the resulting geological model.

```
plotCellData(G, log10(rock.perm(:,1))); % Plot log10 of x-permeabilty
axis equal tight, view(3);
```

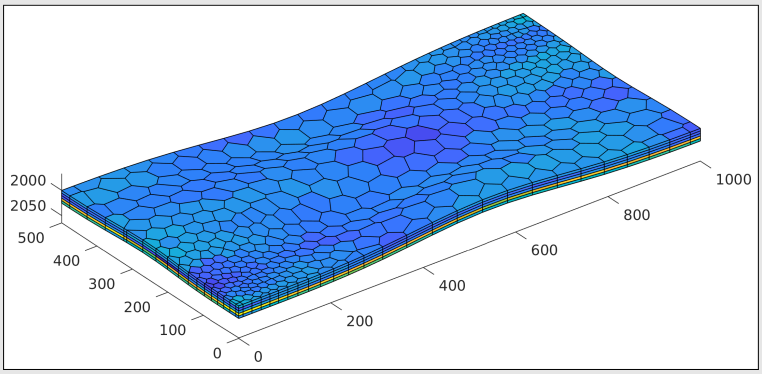



**Figure 5.2:** Geological model constructed by extruding a 2D PEBI grid in the vertical direction, and then shifting the vertical coordinate. The permeability field has five distinct layers, where each layer is lognormal with a different mean.

## 5.2 Flow model

Many of the components that constitute the simulation model in MRST are built using automatic differentiation (see Section 4.3) and object-orientation (AD-OO). We will not describe this in detail here, and refer instead to [111, 17, 129]. First, we load the core functionality (`ad-core`), as well as functionality for fluid properties and for black-oil-type models.

```
mrstModule add ad-core ad-props ad-blackoil
```

We the construct a fluid object representing a simple immiscible three-phase fluid model in which each phase has simple quadratic Brooks–Corey relative permeabilities and distinct viscosity and density.

```
fluid = initSimpleADIFluid( ...
   'phases', 'WOG'                              , ... % Fluid phases
   'n'     , [2,2,2]                            , ... % Relperm expn
   'mu'    , [0.5, 1, 0.1]*centi*poise          , ... % Viscosity
   'rho'   , [1000, 800, 500]*kilogram/meter^3);    % Density
```

Finally, we construct the fluid model. This contains all the information we need about the geological model and fluid properties.

```
modelFI = ThreePhaseBlackOilModel(G, rock, fluid);
```

The `ThreePhaseBlackOilModel` class implements the fully-implicit finite-volume discretization of the black-oil equations (2.15) with upstream mobility-weighting and two-point flux approximation as discussed earlier in Chapter 2 and 3. The implementation is based on the discrete operators outlined in Chapter 4.

MRST describes the reservoir state (e.g., pressure, saturations, etc.) at any time as a `state` object. Given the `state` at timestep $n+1$ and `state0` at timestep $n$, a timestep size `dt`, and a `drivingForces` struct that holds information about wells and boundary conditions, the residual equations can be evaluated as

```
>> problem = modelFI.getEquations(state, state0, dt, drivingForces)

problem =

  LinearizedProblem with properties:

         equations: {[1×1 ADI]  [1×1 ADI]  [1×1 ADI]  [1×1 ADI]  ...}
             types: {'cell'  'cell'  'cell'  'perf'  ...}
     equationNames: {'water'  'oil'  'gas'  'waterWells'  ...}
   primaryVariables: {'pressure'  'sW'  'sG'  'qWs'  ...}
                ...
```

This gives a `LinearizedProblem` with the residual equations on linearized form: the residual equation for the water, oil and gas phases, in addition to the well equations (we only show the well equation for the water phase here). The equations are linearized using automatic differentiation, which we described in Section 4.3:

```
>> eq = problem.equations{1}

eq =

  ADI with properties:

    val: [1195×1 double]
    jac: {[1195×1195 double]  [1195×1195 double]  [1195×1195 double] ...}
```

AD variables are implemented in the `ADI` class, where the `value` field contains the value of the residual equation in each grid cell. The blocks of the Jacobians are stored in the `jac` field, where block `jac{i}` holds the derivatives of the equation with respect to primary variable number `i` (remember Figure 4.4).

## 5.3 Well model

We want to simulate ten years of water injection. To this end, we set one of the wells to inject a total of 1.5 pore volumes over a period of 10 years, and the producer to operate at a fixed bottom-hole pressure of 100 bar.

```
time = 10*year;                              % Total prodution time
rate = 1.5*sum(poreVolume(G, rock))/time; % Injection rate
bhp  = 100*barsa;                            % Bottom-hole pressure
```

The well objects are constructed using `addWell`.

```
W = [];
% Injection well
wc1 = G.cells.tag & G.cells.centroids(:,1) < 500; % Well cells
W   = addWell(W, G, rock, wc1                , ...
  'type'    , 'rate'                         , ... % Rate-ctrld well
  'val'     , rate                           , ... % Injection rate
  'comp_i'  , [1,0,0]                        , ... % Injected comp
  'refDepth', min(G.cells.centroids(wc1,3)), ... % Ref depth bhp
  'name'    , 'INJ'                          );    % Well name

% Production well
wc2 = G.cells.tag & G.cells.centroids(:,1) > 500; % Well cells
W   = addWell(W, G, rock, wc2                , ...
  'type'    , 'bhp'                          , ... % BHP-ctrld well
  'val'     , bhp                            , ... % BHP
  'comp_i'  , [1,0,0]                        , ... % Mandatory
  'refDepth', min(G.cells.centroids(wc2,3)), ... % Ref depth bhp
  'name'    , 'PROD'                         );    % Well name
```

Figure 5.3 shows the wells. The discrete equations relating flow between the wellbore and the reservoir are contained in a well model represented by model class `SimpleWell`, which implements a standard Peaceman-type well model; see Section 3.3. This model is constructed by the flow model upon starting the simulation.

## 5.4 Simulation

We must provide the initial state, which consists of gas in the upper parts ($z < 2000$ depth) and oil in the rest of the reservoir. Pristine reservoirs are usually in hydrostatic equilibrium, but for simplicity, we assign a constant pressure in the entire model.

```
plotGrid(G, 'facecolor', 'none', 'edgealpha', 0.2); % Plot grid outline
plotWell(G, W, 'color', 'k');                       % Plot wells
```

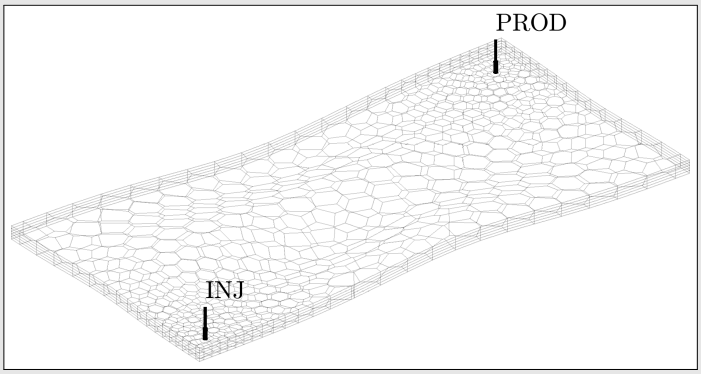



**Figure 5.3:** The injector and producer in our setup.

```
sat0       = repmat([0,1,0], G.cells.num,1); % Filled with oil
ix         = G.cells.centroids(:,3) < 2000;  % Gas in upper parts
sat0(ix,:) = repmat([0,0,1], nnz(ix), 1);
state0     = initResSol(G, bhp, sat0);       % Initial state
```

To simulate water injection, we construct a `schedule` that holds the timesteps $\Delta t$ we want to simulate, and the control (e.g., wells, boundary conditions, etc.) used for each timestep.

```
dt       = rampupTimesteps(time, 30*day); % Timesteps
schedule = simpleSchedule(dt, 'W', W);    % Construct schedule
```

Note that we have not specified boundary conditions in this example. Since we calculate flow over each cell interface, this is equivalent to assigning no-flow boundary conditions.

We also need a nonlinear and a linear solver. The nonlinear solvers in MRST are implemented in the `NonLinearSolver` class, which is part of the `ad-core` module. In this small example, MATLAB's direct solver, `mldivide`, is very efficient. Iterative solvers are found in a separate module, `linear-solvers`, and for the sake of the example we instead use the iterative GMRES solver with CPR preconditioning (see Section 4.4 and Paper IV).

```
mrstModule add linearsolvers
lsol  = CPRSolverAD();      % GMRES solver with CPR preconditioner
nlsol = NonLinearSolver(); % Nonlinear solver
```

We can now simulate the schedule for the given flow model and initial state. MRST usually measures convergence as in commercial simulators, where both the maximum volume error and the total mass-balance error are checked [159]. To be consistent with the discussion in preceding chapters, we will instead only check the discrete residual equations for convergence.

```
modelFI.useCNVConvergence = false;
[wellSolFI, statesFI, reportsFI] ...
    = simulateScheduleAD(state0, modelFI, schedule, ...
                         'linearSolver'   , lsol , ...
                         'nonlinearSolver', nlsol);
```

The following text is the first five columns of output to the command window resulting from timestep 1:

```
Solving timestep 001/130:  -> 2 Hours, 2925 Seconds
=================================================================
| It # | water (cell) | oil (cell) | gas (cell) | waterWells (perf) | ...
=================================================================
|    1 | 3.88e-03     |*0.00e+00   |*0.00e+00   | 1.35e-03          |
|    2 | 8.91e-05     | 2.82e-04   | 3.81e-06   | 2.93e-03          |
|    3 |*6.58e-09     | 4.67e-06   | 2.45e-06   |*7.18e-09          |
|    4 |*3.21e-16     |*9.58e-12   |*1.50e-09   |*3.00e-16          |
=================================================================
Solved timestep with 1 accepted ministep (0 rejected, 3 total iterations)
Completed 3 iterations in 0.77 seconds (0.26s per iteration)
```

The first column shows the nonlinear iteration, whereas the `water`, `oil` and `gas` columns report corresponding norms of the discrete residuals $\mathcal{R}_\alpha$ (3.18), multiplied by the timestep and divided by the pore volume. Moreover, there is one column for each phase for the well equations, in addition to a column for the control equation for each well.

The object `wellSolFI` holds the simulated well states (bottom-hole pressures, water/oil/gas rates, etc.) for each timestep, `statesFI` holds the reservoir states, and `reportsFI` a solver report for each timestep, with information about nonlinear and linear iterations, timestep cuts, etc. Figure 5.4 shows the resulting reservoir saturation at three selected timesteps. Since most of the work in the scientific papers in this thesis is focused on sequential splitting, we simulate the same setup using sequential splitting. This can be done by loading the `blackoil-sequential` module.

```
plotCellData(G, statesFI{step}.s(:,1), 'edgealpha', 0.2);
plotWell(G, W, 'height', 50, 'color', 'k');
caxis([0,1]), axis equal off, view(3)
```

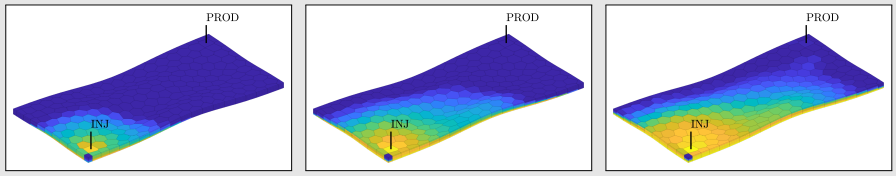



**Figure 5.4:** Saturations at three selected timesteps during the simulation.

```
mrstModule add blackoil-sequential
modelSI = getSequentialModelFromFI(modelFI); % FI model->SI model
[wellSolSI, statesSI, reportsSI] ...          % Simulate
    = simulateScheduleAD(state0, modelSI, schedule);
```

In this case, the command window output reads

```
Solving timestep 001/130:  -> 2 Hours, 2925 Seconds
================================================================================
| It # | Delta P  | waterWells (perf) | oilWells (perf) | gasWells  ...
================================================================================
|    1 | Inf      | 1.35e-03          |*0.00e+00        |*0.00e+00
|    2 | 5.51e+06 | 2.96e-03          | 2.96e-03        |*0.00e+00
|    3 | 2.17e-02 |*8.67e-18          |*8.67e-19        |*0.00e+00
|    4 | 1.43e-02 |*3.04e-18          |*8.67e-19        |*0.00e+00
|    5 | 1.62e-03 |*3.04e-18          |*2.17e-18        |*0.00e+00
|    6 |*1.06e-05 |*8.67e-19          |*2.60e-18        |*0.00e+00
================================================================================
PRESSURE: Solved timestep with 1 accepted ministep (0 rejected, ...
===================================
| It # | oil (cell) | gas (cell) |
===================================
|    1 | 5.46e-02   | 2.93e-03   |
|    2 |*4.11e-05   |*1.01e-07   |
===================================
TRANSPORT: Solved timestep with 1 accepted ministep (0 rejected, ...
Solved timestep with 1 accepted ministep (0 rejected, 0 total iterations)
Completed 0 iterations in 1.03 seconds (Infs per iteration)
```

Here, the flow subproblem (`PRESSURE`) is first solved until the magnitude of the pressure update $\Delta p$ is below a given threshold in all cells, before we solve the transport subproblems (`TRANSPORT`). Note that since the well equations determine the bottom-hole pressure and flow rates, they are solved during the flow step only.

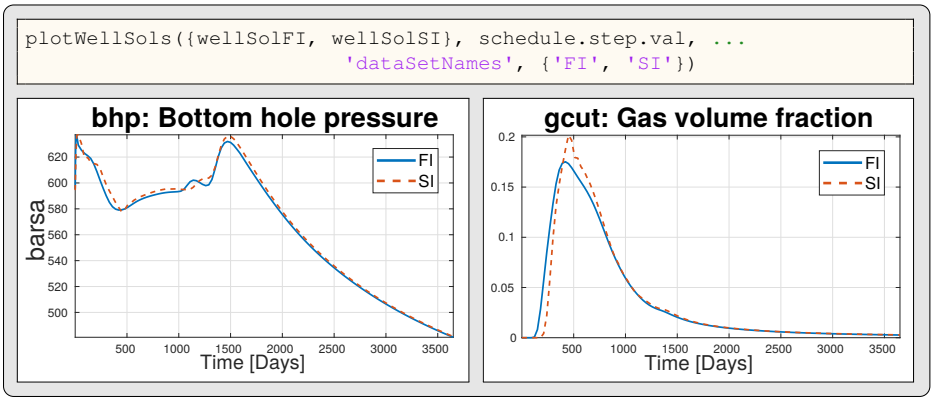



**Figure 5.5:** Bottom-hole pressure in the injector (left) and gas cut in the producer computed using FI and SI.

To compare the results of the SI and FI simulations, we plot the injector bottom-hole pressure and producer gas cut for the two solutions in Figure 5.5. In this case, SI is quite close to FI, even though there is no outer loop to ensure that we resolve couplings between flow and transport.

This short introduction to MRST is by no means exhaustive, but shows some of the core functionality and exemplifies construction of the main components of a simulation model: the geological model, the flow model and the well model. A much more detailed introduction can be found in [111].

# Chapter 6

# Summary of Papers

In total, thirteen scientific papers have been written over the course of this PhD. Four of these are not included, either because they were later worked into journal publications that are included in this thesis, or because they consider topics that are outside the core focus of the thesis. The remaining nine papers make up the second part of this thesis and can be grouped into four broader thematic categories:

- Papers I – III cover unstructured gridding and consistent discretizations. Paper I and Paper II outline a methodology for constructing Voronoi grids that conform to one- and two-dimensional constraints. In particular, the methodology enables cell interfaces of the grid to conform to prescribed surfaces, which could for instance model faults or fractures, and cell centroids to conform to prescribed lines, which e.g., could represent well trajectories. The resulting grids are not necessarily K-orthogonal like many other types of grids generated to accurately represent complex reservoir geometries. The standard two-point flux-approximation scheme is not consistent if the grid is not K-orthogonal; see the discussion in Chapter 2. Papers I and III therefore consider discretizations for elliptic Poisson-type equations that are guaranteed to be consistent on general stratigraphic and unstructured grids.

- Paper IV discusses a new multibasis formulation of the multiscale restriction-smoothed basis (MsRSB) method, in which extra sets of multiscale basis functions are introduced to represent static features in the geomodel more accurately, and to adapt to local changes in the pressure field. The new method is used as a CPR preconditioner in fully implicit reservoir simulation. As such, the paper differs from

the rest of the work in this thesis, since it is the only one that considers fully implicit simulation, and the only paper concerned with linear solvers.

- Papers V and VI address damping strategies for Newton's method applied to transport problems. In Paper V, a purely numerical trust-region method is extended to a two-phase, three-component system modelling polymer flooding, and its efficiency is improved by only invoking trust regions reactively as a response to convergence problems. Paper VI outlines a further improved, robust adaptive trust-region strategy in which trust-region checks are only performed when deemed necessary. The adaptive method performs similar to a standard Newton method when this works well, and approaches the standard trust-region method for particularly challenging cases.

- Papers VII – IX focus on efficient and localized nonlinear solvers for transport problems. Previous research has shown that it is possible to develop highly efficient nonlinear solvers that solve the nonlinear system cell by cell by first performing a topological sort of the graph induced by the total intercell fluxes. This permutes the nonlinear transport problem in to a block-triangular form that can be solved using a nonlinear back-substitution method. Paper VII extends the methodology to black-oil problems, and uses a dG formulation to implement higher-order spatial discretization for the transport equations. The same dG formulation is used in Paper VIII, where the methodology is further extended to compositional problems. This paper also shows how coarsening can be used as a means of dynamic grid adaptivity, and combines it with the dG discretization. Finally, in Paper IX, a robust adaptive coarsening strategy is combined with the reordering-based localized nonlinear solver strategy.

# Paper I – III

**Unstructured Gridding and Consistent Discretizations for Reservoirs With Faults and Complex Wells**

Øystein S. Klemetsdal, Runar Lie Berge, Knut-Andreas Lie, Halvor Møll Nilsen, Olav Møyner

*In proceedings of the 2017 SPE Reservoir Simulation Conference, Montgomery, Texas, USA*

*DOI: 10.2118/182666-MS*

**Unstructured Voronoi Grids Conforming to Lower-dimensional Objects**

Runar Lie Berge, Øystein S. Klemetsdal, Knut-Andreas Lie

*Computational Geosciences, volume 23, issue 1, pp. 169–188, 2019*

*DOI: 10.1007/s10596-018-9790-0*

**A Comparison of Consistent Discretizations for Elliptic Poisson-Type Problems on Unstructured Polyhedral Grids**

Øystein S. Klemetsdal, Olav Møyner, Xavier Raynaud, Knut-Andreas Lie

*Manuscript in preparation, 2019*

The topics of the first three papers in this thesis is summarized in the title of Paper I: Unstructured gridding and consistent discretizations for reservoirs with faults and complex wells. This is evidently two different topics, but they are closely related: The computational grid has a direct impact on the quality of the numerical solution, but what is the best computational grid will depend on the specific discretization. This is particularly true in reservoir simulation, where the grid is usually generated to represent the spatial variation in the coefficients of the PDE accurately and with as few volumetric cells as possible. Paper I is concerned with gridding and discretizations, Paper II gives a more in-depth and updated discussion of the novel gridding method first presented in Paper I, whereas Paper III presents an updated discussion of consistent discretizations for unstructured polyhedral grids.

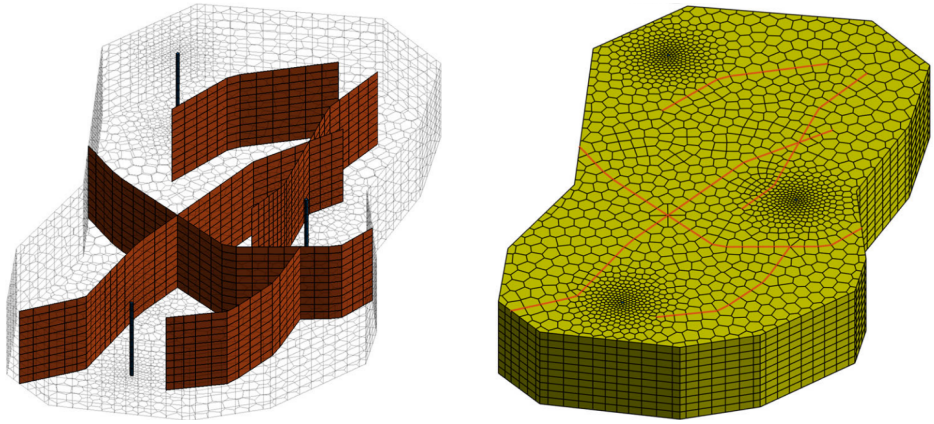
## Unstructured Voronoi grids

Papers I and II discuss application of Voronoi grids, commonly referred to as Perpendicular BIsector (PEBI) grids in reservoir simulation, to represent

complex reservoir geometries. This is not a new idea – PEBI grids were introduced in reservoir simulation in the early 1990s [79, 67, 165]. While quite flexible, the main challenge with PEBI grids has been to accurately capture complex geometries like multiple intersecting fault surfaces. Methods for this, as well as strategies for handling faults intersecting at sharp angles were outlined by Branets et al. [31] and later Manzoor et al. [123], Toor et al. [179]. These methods are based on constrained Delaunay triangulation, in which a protection layer is placed around faults. This ensures that the grid conforms to intersecting faults and well trajectories, but tends to create regions with small and skewed cells near intersections. These undesirable effects can be mitigated by using a conflict-point removal scheme [54], or by optimizing the PEBI grid so that cells cut in two by fractures or fault surfaces have a minimal volume [126, 127]. The latter approach is based on the fact that a PEBI grid minimizes an energy function on the form $\sum_{i=1}^{n_c} \int_{\Omega_i} |\boldsymbol{x}_i - \boldsymbol{x}|^2 \, \mathrm{d}V$. The PEBI grid can also be optimized by treating the edges and vertices in the underlying Delaunay triangulation as springs and connecting joints. The optimum is then found as the equilibrium to this system [146].

In Papers I and II, we present a method for generating grids that satisfy two types of conformity requirements: interface-conformity, where given surfaces are traced by cell interfaces, and cell-conformity, where a given line or surface is traced by cell centroids. This is achieved by building the grid dimension-by-dimension: 0D features (points) represent line endpoints and intersections; 1D features (lines) represent intersections between surfaces, or line objects such as well trajectories; whereas 2D features represent surfaces such as faults and stratigraphic horizons. This construction gives a nice representation of all constraints in the computational grid, and ensures that all objects of dimension $d-1$ are represented as interfaces of objects in dimension $d$.

The gridding methodology is flexible and robust, and can accurately conform to complex compositions of surface and line constraints. This is demonstrated through a number of examples. On the other hand, the 3D method has certain inherent limitations, since constraints are only communicated from lower-dimensional grids to higher-dimensional grids. As a result, sites (or control points used to constrain the grid to surfaces) from different surface constraints may interfere with each other so that the resulting grid is not guaranteed to conform to all these constraints. In contrast, the 2D methodology allows for communication between constraints of the same dimension, so that the grid conforms to all interface constraints.



Fractures 2.5D PEBI grid

**Figure 6.1:** A 2.5D PEBI grid conforming to fracture surfaces and having increased resolution near three vertical wells. Figure from Paper II.

## Consistent discretizations

As pointed out by e.g., Samier and Masson [157], Schneider et al. [164], elliptic problems for flow in porous media should be discretized with a method that:

1. is locally mass conservative, meaning that mass conservation carries over from the continuous conservation law to the discretized version;
2. is consistent and coercive, which together gives a convergent method;
3. is monotone, meaning that the discrete solution satisfies similar minimum/maximum principles as the true solution;
4. ensures that the resulting discretization matrix is reasonably sparse.

In Papers I and III, we compare different consistent discretizations for elliptic Poisson-type problems with these properties in mind. To limit the discussion, we consider a representable set of discretizations including multipoint flux approximation (MPFA) methods [8, 58, 6], mimetic finite difference (MFD) methods [101, 34, 23], and nonlinear TPFA methods [108, 118, 142, 163]. We also include a fairly recent development in finite-element methods called *virtual element methods* (VEM) [21, 9, 22] in the comparison. VEM enables finite-element discretizations on general polyhedral grids by introducing moments over the cells as degrees of freedom, thereby removing the requirement of explicit knowledge of basis functions.

Paper I presents the mentioned methods and compares them on various test cases. In particular, consistency, robustness, and computational complexity is considered. Paper III reports a more in-depth study, which also focuses on how the different methods impact the transport solver when used as the pressure solver in a sequential setting.

As many studies before, the work in these two papers emphasizes the importance of using a consistent method. The novelty of the work lies in a unified comparison of several methods within the same open-source framework. Results are, however, inconclusive with respect to what discretization is the best, but suggest that further studies should also take into account how the different methods affect subsequent solution of a transport equation. In terms of comparisons, the two studies are somewhat limited in that they do not include mixed/hybrid finite-element methods [33], nor the promising vertex approximation gradient method [63, 64, 157].

# Paper IV

**Accelerating Multiscale Simulation of Complex Geomodels by Use of Dynamically Adapted Basis Functions**

Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie

*Computational Geosciences, published ahead of print, 2019*
*DOI: 10.1007/s10596-019-9827-z*

A natural extension of using algebraic multiscale methods as iterative linear solvers [74, 192, 121, 132] is to include multiple sets of restriction/prolongation operators, which gives a truly *multi*scale method [50, 114, 120]. Lie et al. [114] proposed to use different multiscale operators to increase the resolution of specific model features such as fractures and wells, demonstrating a significantly improved convergence rate over a simple two-stage multiscale method. This was facilitated by using the multiscale restriction-smoothed basis (MsRSB) method [133, 132, 133], which imposes very few restrictions on the underlying coarse grid. In Paper IV, we follow the idea of Lie et al. [114] and introduce additional dynamically adapted basis functions to capture local changes in the pressure field associated with propagating fluid fronts to improve the convergence rate.

Algebraic multiscale methods are natural candidates as the elliptic solver in CPR preconditioners [189, 190], and results on this was first published by Cusini et al. [49]. In our work, we use the iterative multiscale-multibasis framework to accelerate linear convergence in fully implicit simulations. The result is a convergence speedup in the linear solver of 10-60% compared to using a multiscale solver with only one set of basis functions. Figure 6.2 gives a schematic illustration of the linear solution process. The paper only discusses adding basis functions that adapt to changes in the pressure and saturation of specified phases, but the framework is general and also enables adaption to other dynamic cell quantities like mass fractions, residuals, or error indicators (see Paper IX).

The methodology is extensively tested through several numerical examples, including EOR scenarios and realistic reservoir models, but the paper lacks a comparison with a CPR solver that uses a more standard elliptic solver such as AMG. In addition, only one cycle of the iterative multiscale solver is used, while in fact a number of cycles or convergence to a non-strict tolerance may be a better choice. The reason for using only one cycle is to keep the comparison between different multiscale solvers as clean-cut as possible. Moreover, the proof-of-concept implementation in MRST has a certain computational overhead compared with compiled

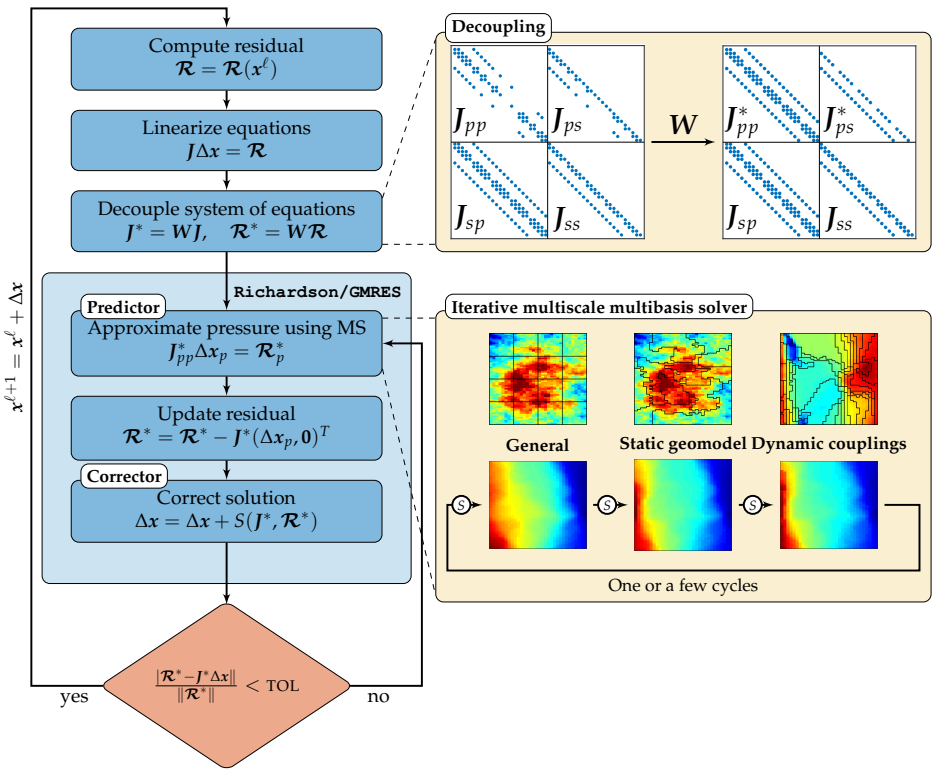



**Figure 6.2:** Schematic illustration of how the iterative multiscale solver with multiple basis functions can be used as the elliptic solver in a CPR preconditioner. After decoupling of the linearized fully-implicit system, the block $J^*_{pp}$ associated with the pressure variables resembles the discretization matrix of an elliptic operator, and the solution to $J^*_{pp}\Delta x_p = \mathcal{R}^*_p$ is a reasonable approximation of the true pressure update.

languages, and performance is thus reported in term of the number of linear iterations per nonlinear iteration. So, even though the paper includes a complexity analysis supporting the results, a more thorough numerical investigation with CPU times is necessary to demonstrate the efficiency of the solver on realistic field-scale models.

# Paper V – VI

**Non-linear Newton Solver for a Polymer Two-phase System Using Interface-localized Trust Regions**

Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie

*In proceedings of the 19th European Symposium on Improved Oil Recovery, 2017, Stavanger, Norway*

*DOI: 10.3997/2214-4609.201700356*

**Robust Nonlinear Newton Solver with Adaptive Interface-Localized Trust Regions**

Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie

*SPE Journal, volume 24, issue 4, pp. 1576–1594, 2019*

*DOI: 10.2118/195682-PA*

In sequential splitting, the flux function tends to have a characteristic S-shape, which introduces different contraction regions for an associated nonlinear Newton solver. Therefore, even though the implicit backward Euler temporal discretization is unconditionally stable, the Newton solver often exhibits convergence issues when applied to the transport subproblem. Jenny et al. [91] suggested to dampen Newton updates based on *trust regions*, which is a well-known method from mathematical optimization that delineates different contraction regions (see e.g., [53, Chapter 3]). The result is an unconditionally convergent method for any timestep length.

Identification of the trust regions was originally based on closed-form expressions, and has been successfully applied to two-phase flow with buoyancy [191] and capillarity [110]. Voskov and Tchelepi [186] extended the methodology to compositional flow by considering trust regions along key tie-lines. These early versions of the method applied the same damping factor to all cells, since this preserves the direction of the Newton update. However, the hyperbolic nature of the transport equations means that Newton's method will converge much faster in some parts of the domain than others. Møyner [130] extended the methodology to multiple phases in a fully numerical framework that considers Newton updates across cell interfaces. He also suggested a localized version of the method, in which the damping factor of a cell is chosen as the minimum of the required damping factors in the cell itself and in all its upstream neighbors. This was an improved generalization of the original trust-region method, but requires a significant number of residual evaluations to approximate

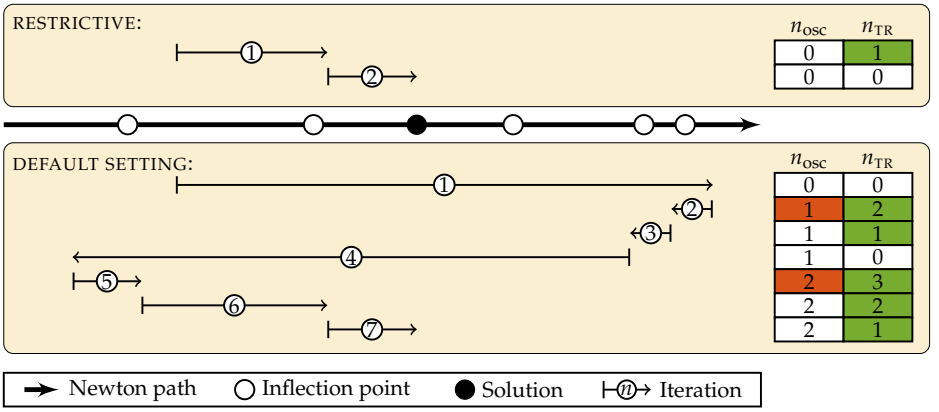



**Figure 6.3:** The adaptive trust-region framework. Trust regions are invoked over an interface when we observe an oscillation, and kept on for $n_{\text{TR}}$ iterations, which depends on the number of observed oscillations $n_{\text{osc}}$ over the interface during the timestep. Figure from Paper VI.

the trust regions, and is therefore expensive. In addition, the damping factors tend to be overly restrictive.

Paper V extends the numerical trust-region framework of Møyner [130] to a two-phase three-component system modelling polymer flooding [178] and improves efficiency by introducing adaptivity in the trust-region algorithm. In particular, the paper suggests a heuristic approach that invokes trust-region damping as a response to convergence problems rather than as a default restriction. The methodology is further improved in Paper VI, where we propose using oscillation detection to activate trust regions only when needed. Criteria for turning trust regions on and off come in the form of a number of parameters that can either be set dynamically or be prescribed by the user to achieve a nonlinear solver tailored to the problem at hand. When properly configured, the method performs similar to a Newton solver when this performs at its best, and approaches the static trust-region solver for particularly challenging cases. This is demonstrated through a number of test cases, from contrived scenarios constructed to challenge the nonlinear solver, to reservoir models with realistic grids and fluid physics. The proposed solver is compared to Newton's method with industry-standard damping strategies with and without line search, and to the trust-region solver of Møyner [130]. Figure 6.3 illustrates the method.

To ensure a clean-cut comparison between the different nonlinear solvers, the standard Newton solver includes industry-standard chopping methods based on solution increments, number of iterations, etc., but does not

include modified Appleyard-chopping with treatment of transitions to/from mobile/immobile saturations [159], which may improve the convergence of this method. The work demonstrates promising results for the adaptive trust-region solver, but once again, results are only reported in terms of nonlinear iterations, since the trust-region method is not possible to vectorize, and consequently not possible to implement fully CPU efficient in MATLAB. As a compromise, the number of residual evaluations per nonlinear iteration are also reported, and should give a clear indication of expected performance in a compiled language.

# Paper VII – IX

**Efficient Reordered Nonlinear Gauss-Seidel Solvers With Higher Order For Black-Oil Models**

Øystein S. Klemetsdal, Atgeirr Flø Rasmussen, Olav Møyner, Knut-Andreas Lie

*Computational Geosciences, published ahead of print, 2019*

*DOI: 10.1007/s10596-019-09844-5*

**Implicit High-resolution Compositional Simulation with Optimal Ordering of Unknowns and Adaptive Spatial Refinement**

Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie

*In proceedings of the 2019 SPE Reservoir Simulation Conference, Galveston, Texas, USA*

*DOI: 10.2118/193934-MS*

**Dynamic Coarsening and Local Reordered Nonlinear Solvers for Simulating Transport in Porous Media**

Øystein S. Klemetsdal, Knut-Andreas Lie

*Manuscript in preparation, 2019*

## Localized nonlinear solvers

The key ideas behind localized nonlinear solvers for the transport subproblem based on topological sorting have been thoroughly introduced in Section 4.5, and amounts to using the intercell flux field obtained from the solution to the pressure subproblem to sort the grid cells so that the discrete transport equations are permuted to a (block) triangular form. The methodology was first introduced by Natvig and Lie [138], and has later been applied to a wide range of problems including linear (tracer) transport [60, 149], two-phase flow [4, 138], and polymer flooding [113, 150]. Lie et al. [112] also discuss how the method can be combined with operator splitting techniques for cases with strong buoyancy/capillary effects.

Much of the groundwork for implementing the localized nonlinear solver framework in MRST together with a discontinuous Galerkin discretization of transport equations was done as a preparation for the ECMOR XVI conference [97]. This paper was later developed into a journal article, included in this thesis as Paper VII, which focuses on simulation of black-oil models using localized nonlinear solvers based on optimal cell ordering.

This first version implemented in MRST did not support capillary/gravity forces, but the paper also presents results from a solver implemented in OPM Flow, which was capable of simulating the full Norne field model with real reservoir geology and full black-oil fluid physics, including gravity and capillary pressure. Paper VIII extends the methodology to compositional simulations with capillary/buoyancy forces, whereas Paper IX demonstrates how it can be combined with dynamic coarsening.

Whereas solving the transport subproblems cell-by-cell is optimal in theory, modern hardware typically comprises multicore CPUs with SIMD-type registers. Feeding sufficient computations to each core is therefore imperative for fast execution time. In Paper VII, we therefore devise a block-wise reordering solver that instead of solving for one node (with one or a cycle of mutually dependent cells) in the intercell flux graph, solves for a block of the next $n_b$ cells. This should be included in any optimized implementation of the reordering methods, and possibly include parameter tuning to utilize specific memory architectures as efficiently as possible.

### Higher-order methods

The last few decades have seen a large effort into research of higher-order methods for transport in porous media; see, e.g., [25, 29, 57, 13, 153, 119, 14]. These methods are nevertheless not yet commonplace in daily engineering simulations. A possible reasons for this is that they tend to be cumbersome to implement in an implicit setting, and once implemented, the added complexity significantly impedes nonlinear convergence [137, 116]. Likewise, it is not fully clear how to handle the skewed and general cell geometries and unstructured grid topology encountered in real field models in an efficient an robust manner (remember the cells of the Norne field model in Figure 3.12).

The unidirectional flow property exploited in localized nonlinear solvers carries over to all discretizations based on variational formulations on the form (3.6) as long as support of each basis function is restricted to a single cell, and we use upstream mobility evaluation. In addition to finite-volume methods, this is for instance the case in discontinuous Galerkin (dG) methods. These methods were first introduced for hyperbolic conservation laws by Cockburn and Shu [45, 46], and their utility has been demonstrated for different types of fluid models, including, e.g., single-phase, multicomponent flow [175, 84]; two-phase flow [153, 61, 138]; and black-oil and compositional flow [128, 10]. However, all these papers assume that the grid is structured (or relatively regular).

In Paper VII and Paper VIII, an effort has been made to tailor dG methods to the type of grids seen in engineering simulations. This includes use of tensor products of Legendre polynomials defined on the *bounding box* of the cell as basis functions, and construction of efficient cubature rules based on moment-fitting [135]. Referring back to Section 4.3, we see that having as few integration points as possible is particularly important in implicit simulations with automatic differentiation, where evaluation of the residual function in an integration point also involves potentially costly operations on the Jacobian.

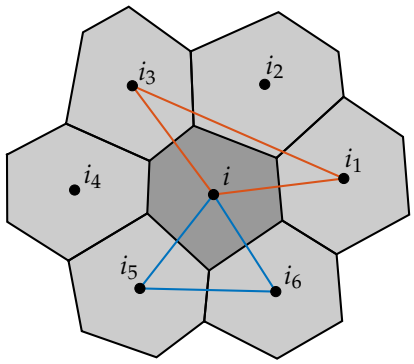
Another known problem with dG methods is that discontinuities in the solution tend to create spurious oscillations. This is usually handled by imposing a slope limiter that effectively introduces enough numerical diffusion to dampen oscillations. Such limiters are somewhat involved to implement in implicit discretizations, and Natvig et al. [139] therefore suggested an alternative "slope kill" approach: Whenever the jump across an interface is larger than a given threshold, the polynomial degree is reduced to 0 in related cells, which are also subdivided into smaller cells (i.e., h-refinement, p-reduction). In Papers VII and VIII, we use a simplified approach in which the degree is reduced to 0, but with no compensating spatial refinement. This will not preserve the formal order and the local accuracy of the method, but has proven to be quite robust.

Numerical examples indicate that the proposed modifications give a robust and fairly efficient dG method. The method has, however, so far only been applied to immiscible flow. Although work is underway, a fully robust implementation of dG for compositional flow remains to be worked out.

### *Comparison of implicit discontinuous Galerkin and WENO schemes on stratigraphic and unstructured grids*

In Paper VIII, we also compare our dG implementation with a weighted essentially non-oscillatory (WENO) discretization, and a more in-depth comparison of dG and WENO schemes for stratigraphic and unstructured grids was presented at the 2019 SIAM Geosciences conference [98]. These results are not yet published, and we therefore give a short introduction to WENO schemes and report some of the results from this work here.

High-resolution finite-volume discretizations can be constructed by introducing higher-order interpolation for interface quantities. As an example, consider a cell $i$ in Figure 6.4, with neighbors $\mathcal{N}(i) = \{i_1, \ldots, i_6\}$. A




| Primary triplets | Secondary triplets |
|---|---|
| $(i, i_1, i_2)$ | $(i, i_1, i_3)$ |
| $(i, i_2, i_3)$ | $(i, i_1, i_4)$ |
| $(i, i_3, i_4)$ | $(i, i_1, i_5)$ |
| $(i, i_4, i_5)$ | $(i, i_1, i_6)$ |
| $(i, i_5, i_6)$ | $(i, i_2, i_4)$ |
| $(i, i_6, i_1)$ | $(i, i_2, i_5)$ |
| | $\vdots$ |

**Figure 6.4:** Stencils used to for linear reconstruction of interface quantities in a cell $i$ with six neighbors $\mathcal{N}(i) = \{i_1, \ldots, i_6\}$.

linear reconstruction of the mobility $\lambda$ in cell $i$ can be written as

$$\lambda(\boldsymbol{x}) = \lambda(\boldsymbol{x}_i) + (\boldsymbol{x} - \boldsymbol{x}_i) \cdot \nabla\lambda + \mathcal{O}(|\boldsymbol{x} - \boldsymbol{x}_i|^2). \tag{6.1}$$

We can approximate the gradient $\boldsymbol{\sigma} \approx \nabla\lambda$ using a triplet of cell $i$ and two of its neighbors. Cell $i$ has 15 such triplets: 6 primary triplets (i.e., triplets where all cells are neighbors), and 9 secondary triplets. Generally, with $N$ neighbors, we get a total of $N_i = \binom{N}{2}$ gradient approximations $\{\boldsymbol{\sigma}_i^k\}_{k=1}^{N_i}$ and corresponding linear reconstructions,

$$\hat{\lambda}_i^k(\boldsymbol{x}) = \lambda_i + (\boldsymbol{x} - \boldsymbol{x}_i) \cdot \boldsymbol{\sigma}_i^k, \tag{6.2}$$

where we have used that $\lambda(\boldsymbol{x}_i) = \lambda_i$. By using these to define linear reconstructions $\hat{\lambda}_i$ in each cell and combining these with the upwind scheme in equation (3.14),

$$\lambda_{ij} = \begin{cases} \hat{\lambda}_i(\boldsymbol{x}_{ij}), & v_{ij} > 0, \\ \hat{\lambda}_j(\boldsymbol{x}_{ij}), & v_{ij} < 0, \end{cases}$$

we get a second-order accurate method. Higher order approximations are possible if we accept to use wider stencils (i.e., sextets for quadratic reconstructions, etc.). By choosing the smoothest linear reconstruction in each cell, determined based on a suitable smoothness indicator, we get an essentially non-oscillatory (ENO) method [76]. WENO [39] is a natural alternative to the ENO method, in which we define the reconstructions $\hat{\lambda}_i$ as a linear combination of all available reconstructions $\{\hat{\lambda}_i^k\}_{k=1}^{N_i}$: given linear weights, $\sum_k \gamma_i^k = 1$, we compute *nonlinear* weights $w_i^k$,

$$w_i^k = \frac{\beta_i^k}{\sum_{k=1}^{N_i} \beta_i^k}, \qquad \text{where} \quad \beta_i^k = \frac{\gamma_i^k}{(\varepsilon + \Lambda_i^k)^2}, \quad \Lambda_i^k = |\Omega_i|\,|\boldsymbol{\sigma}_i^k|^2.$$

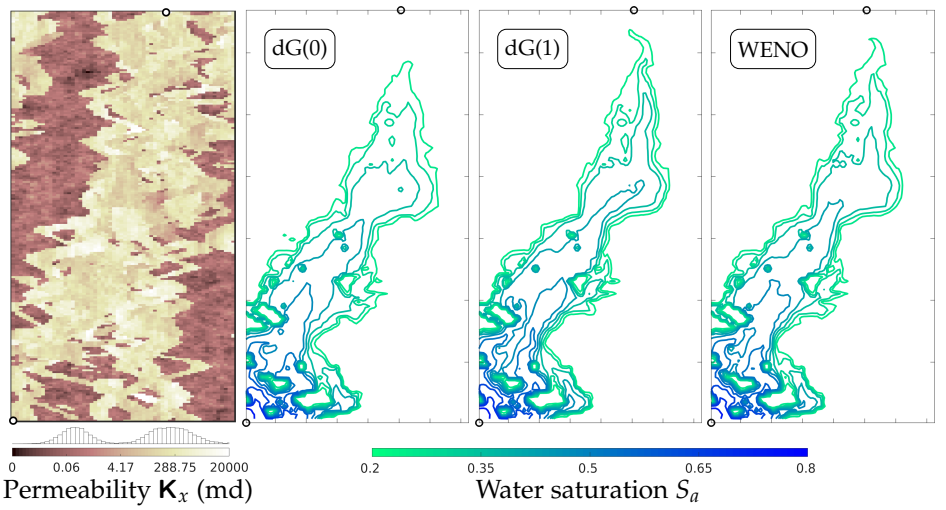



**Figure 6.5:** Water injection in Layer 72 from the Upper Ness formation in Model 2 of the SPE10 benchmark simulated using the dG(0), dG(1) and WENO.

Here, $\Lambda_i^k$ is the smoothness indicator, and $\varepsilon$ is a small number to avoid division by zero. The reconstruction is then defined as

$$\hat{\lambda}_i(\boldsymbol{x}) = \sum_{k=1}^{N_i} w_i^k \hat{\lambda}_i^k(\boldsymbol{x}).$$

Mykkeltvedt et al. [137] recently demonstrated how to implement fully implicit higher-order WENO methods for polymer flooding in 1D and 2D. These ideas have been further pursued to tailor the second-order WENO method for real reservoir models [116] by, e.g., considering an SVD-based transformation of the cell coordinates for gradient approximations, and choosing only a subset of the cell neighbors in the linear combination.

Figure 6.5 shows results from simulation of water injection in Layer 72 of SPE10 Model 2, which we introduced in Section 2.3. This layer is part of the Upper Ness formation and consists of an intertwined pattern of fluvial sand channels on a background of low-permeability mudrock. The results are obtained using dG(0), dG(1), and WENO, where $k$ in dG($k$) indicates the polynomial degree of the basis functions in the dG method, so that dG($k$) is of formal order $k+1$. We see that the two higher-order methods result in visually sharper and less diffuse displacement profiles, in particular for dG(1). Note that we have used the sequential implicit method with dG for the transport subproblem only, whereas WENO results are computed

using the fully implicit method. The results are therefore not directly comparable.

Figures 6.6–6.8 report a comparison on a subset of the SAIGUP model, which we briefly introduced in Section 4.3. This is a synthetic model with petrophysical properties based on data from real shallow-marine formations, represented in stratigraphic grid format [122]. We extract the three top layers, and simulate waterflooding by imposing a pressure drop from the western to the eastern boundary, and no-flow conditions elsewhere. Again, dG(0) introduces significantly more diffusion than dG(1) and WENO, which in this case are visually indistinguishable.

These two simple examples highlight how numerical diffusion affects the quality of the solution, and may lead to inaccurate predictions. In terms of coding complexity, WENO is appealing because it fits naturally into the finite-volume framework widely used by the reservoir simulation community. Discontinuous Galerkin methods, on the other hand, require heavier machinery, including basis functions and cubature rules. Moreover, the number of unknowns increases significantly with the order of the method. WENO, on the other hand, retains the same low number of unknowns as a standard FV method, albeit with a much denser discretization matrix. On the other hand, the WENO stencil introduces both upstream and downstream dependencies, as can be seen from Figure 6.4, and it is therefore not possible to combine with the reordering method, at least not without major modifications of the weighting scheme.

### Dynamic coarsening

Dynamic grid adaptivity can be used either as an alternative or as a complement to higher-order spatial discretization to improve the solution accuracy. A particularly popular approach applicable to reservoir simulation is local grid refinement (LGR), in which a static or temporary refinement is introduced by subdividing the grid cells into finer cells in parts of the domain. This has been successfully applied to a number of scenarios, including two-phase [134, 107], black-oil [75, 162] and compositional EOR [19, 173, 83] simulations, albeit only for grids with a structured topology.

A challenge with LGR is the lack of readily available, robust refinement strategies for the contorted cell geometries usually encountered in real simulation models. However, such models are usually upscaled from a much finer geocellular model (see Section 2.3), which motivates the use of dynamic coarsening instead of dynamic refinement. This has been used by a number of authors to construct static and dynamic grids with locally in-

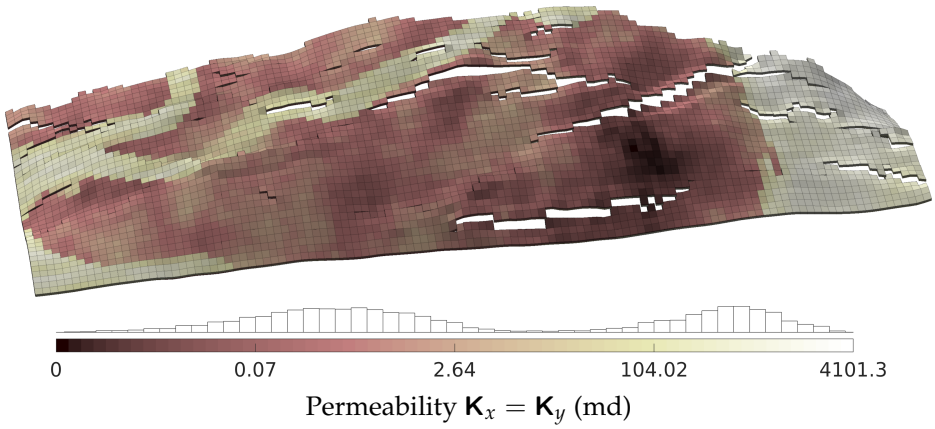



**Figure 6.6:** Permeability in the top three layers of the SAIGUP model. The reservoir is modelled after data from real, shallow-marine formations. Water flooding is simulated by imposing unequal pressure boundary conditions on the western and eastern boundaries, and no-flow conditions elsewhere.

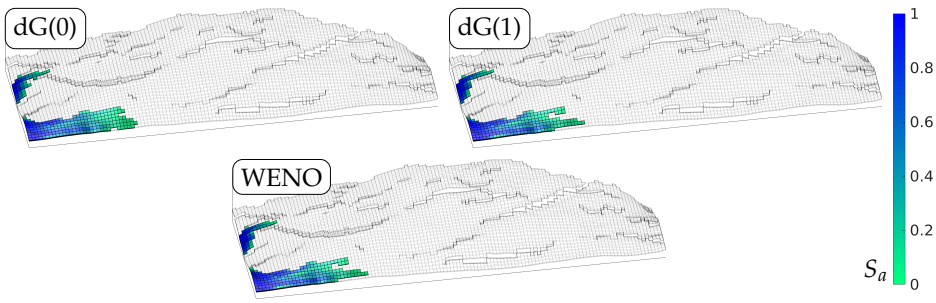



**Figure 6.7:** Water saturation after 1869 days.

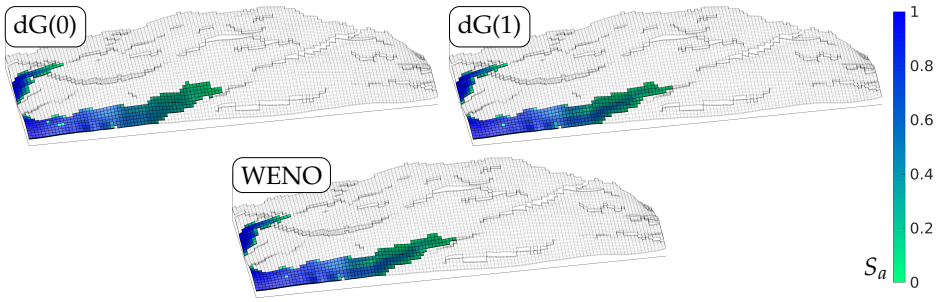



**Figure 6.8:** Water saturation after 2760 days.

creased spatial resolution [56, 5, 77, 92, 93, 71, 50, 48], and in Paper IV in the context of dynamic multiscale methods for pressure equations.

In Paper VIII, we presented an early two-level version of adaptive coarsening, and combined it with a dG(1) discretization. Paper IX presents a multilevel generalization of the same method. The method has few restrictions and assumptions on the grid levels, except that they must form a succession of increasingly coarser resolutions. However, in the special case that the coarse partitions are nested, one can avoid a lot of the bookkeeping that would otherwise be necessary, and further reduce the computational overhead of the dynamic grid adaption by precomputing several quantities needed for the coarse-scale discretization [83]. The paper therefore focuses on validating the method in this case, and also combines it with the localized nonlinear transport solver. Figure 6.9 illustrates the framework for a small PEBI grid.

For simplicity, Paper IX only includes examples with immiscible two-phase flow. Robust dynamic coarsening is significantly more challenging for compositional flow, especially mapping of variables between different dynamic grids. Moreover, mapping intercell fluxes from the fine grid to a coarsened grid tends to introduce connected components in the intercell flux graph, even though the flow is unidirectional on the fine grid. This means that applying the local nonlinear solver based on reordering will not give as good speedup factors as for the underlying fine-scale model. To fully utilize the combination of these two methods, improved and more robust mapping of variables between different grid levels therefore needs to be investigated further.

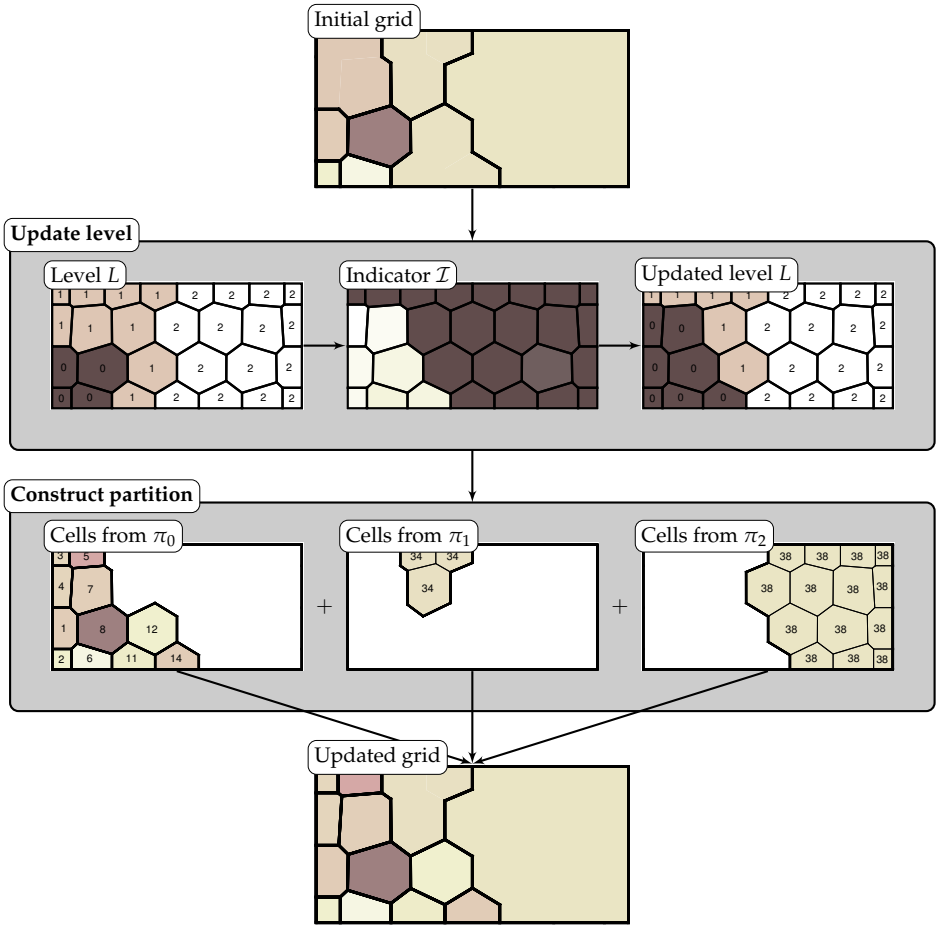



**Figure 6.9:** Illustration of dynamic coarsening for a small PEBI grid. A dynamically adapted grid can be constructed by combining blocks from different coarse versions of the fine grid. Figure from Paper IX.

# Chapter 7

# Concluding Remarks and Future Work

## 7.1 Concluding remarks

This thesis comprises work on various topics in reservoir simulation, with an overarching goal of developing accurate and efficient discretization and solution strategies by exploiting the physical and mathematical structure of the underlying problem. In particular, the thesis discusses gridding, discretizations, and linear and nonlinear solvers, both for fully implicit and sequential implicit solution strategies. From a mathematical point of view, we study two different types of equations that have distinctly different character in the incompressible limit: Elliptic equations governing fluid pressure are linear and have infinite speed of propagation, whereas the nonlinear hyperbolic equations governing transport of components or fluid phases have finite speed of propagation and typically exhibit discontinuous solutions. When solved separately in a sequential solution procedure, one benefits from using specialized solvers that target and utilize features of each equation.

Papers I – III of the thesis focus on problems that are fundamental to reservoir simulation – accurate description of reservoir geology and well paths in the simulation grid, and consistent discretizations for the pressure part of the flow problem. Papers IV – IX focus on efficient, specialized solvers. The primary target of the thesis was to develop robust and efficient nonlinear solvers for water-based EOR, a goal that in the author's opinion has been achieved by the adaptive trust-region solver introduced in Paper V and Paper VI, and the localized nonlinear solver based on optimal reordering of grid cells, discussed in Papers VII – IX.

### Reproducible and transparent research

Particular effort has been made to ensure that the research in this thesis is reproducible and transparent. To this end, all the proposed solution strategies are implemented in the open-source MRST framework (see Chapter 5). The implementation of VEM and the new method for constrained Voronoi mesh generation have already been published online as the add-on modules `vem` and `upr` to MRST. Likewise, we plan to release the implementation of discontinuous Galerkin discretization, nonlinear solvers based on reordering, and the dynamic coarsening framework. However, these codes need to be cleaned up and documented better to be accessible and useful to others. Work on this is in progress at the time of writing, but not yet finished. Moreover, to investigate the efficacy of the proposed solution strategies, a large number of numerical experiments have been conducted. It has always been a priority to come up with examples that challenge the new methods, and make sure to also report cases when they *do not* work well. With this in mind, we have made an effort to include examples with a high degree of realism, both in terms of fluid physics, geological models, and wells/injection strategies. On these grounds, it is my sincere opinion that the research presented herein to a large degree is both reproducible and transparent.

## 7.2 Future work

In the author's experience, working with one new research idea tends to spawn at least two new ones. The fact that this thesis spans a wide range of different topics therefore means that there are many ideas that could not be pursued within the timeframe of a 3-year scholarship. Below is therefore a non-exhaustive presentation of research ideas that the author hopes to follow up on in later work.

### Localized nonlinear solvers

Scalability of the localized nonlinear solver used in this thesis is an open question, in particular because of the causality requirement that makes this solution strategy strictly serial. Vikøren [183] investigated a strategy that treats the entire problem in parallel in his master's thesis, but observed no speedup. Another possible approach is to use domain decomposition, and initiate the local solver on each subdomain simultaneously. When a domain is finished, updated boundary conditions can be communicated

to downstream subdomain neighbors, before restarting a new sweep. Domain decomposition strategies for the transport solver are used in commercial simulators [99], and results of using the block-wise reordering solver, reported in Papers VII – IX, indicate that using a localized nonlinear solver in each domain can significantly speed up the simulation.

The local nature of the transport equations means that different subdomains will converge at a very different rate, especially in the beginning of a simulation. Idle processor time can then be reduced by employing a parallelization approach called work-stealing, in which an idle process "steals" work from the stack (i.e., list of cells to be solved for) of a busy process [28].

As pointed out by Wong [194], the sequentially implicit (SI) method can be used as an efficient nonlinear block Gauss–Seidel preconditioner for Newton's method applied to the outer iteration in a sequentially fully implicit (SFI) formulation. This results in a nonlinear multiplicative Schwarz preconditioning method [36]. The localized nonlinear transport solver fits very well into this framework, and using it in such a setting is a natural extension of the work presented here.

Multirate methods exploit the local nature of the transport subproblem: The entire domain is first solved with the same timestep, after which a subset of the grid cells are identified based on an error indicator, and solved using a local, smaller timestep [158, 52]. Linga et al. [117] recently presented a truly local timestepping approach that uses the local nonlinear solver with reordering to solve each (super)node, or block of (super)nodes, in the total-flux graph using a locally determined subdivision of a global timestep. We plan to develop this method further, and research how it can be combined with higher-order spatial discretizations and dynamic coarsening.

### Spatial discretizations

Elliptic discretizations in subsurface flow are typically used to discretize the pressure equation in multiphase flow simulations. However, consistent discretizations for elliptic problems have primarily been studied for the pressure equation alone, possibly with examples of how they affect the solution accuracy of the transport problem. In Paper III, we pointed out that different pressure discretizations will also affect the *performance* of the nonlinear solver used to compute the transport solution. We therefore intend to study how different elliptic discretizations affect the qualitative transport properties of the induced flux fields, measured using, e.g., tracer transport, residence-time distributions, interwell connections, and other

types of flow diagnostics [111, Chap. 13]. Likewise, we will study how discretizations affect the number, distribution, and size of the connected components in the flux graph, which not only determine the efficiency of localized nonlinear solvers, but may also affect the convergence of global Newton-type solvers.

Papers VII and VIII discuss some aspects on how to tailor dG methods to realistic reservoir simulations. High spatial accuracy is particularly important to reduce numerical smearing and capture linear or weakly nonlinear waves, e.g., to correctly resolve their effect on increased recovery in EOR models, and to accurately track the propagation of different species in compositional simulations. A particular challenge with dG for compositional flow is how to robustly handle the flash calculations associated with the equation of state. Continued research on dG methods, and in particular their application to compositional flow and complex grid geometries, as well as robust handling of spurious oscillations, is therefore natural.

As mentioned in Chapter 6, it is not evident how to combine higher-order WENO methods with the local nonlinear solver because WENO, and similar schemes like TVD methods with slope/flux limiters introduce downstream dependencies in the discretization stencil. Modifying WENO for reordering is therefore a possible future research topic. First of all, Lie et al. [116] demonstrated that using only a subset of the neighbors of a cell to reconstruct interface quantities still gives high-quality results. Likewise, Mykkeltvedt et al. [137] suggested to lag the evaluation of the nonlinear WENO weights (and TVD limiters) in the higher-order spatial reconstruction over the time step as a means to alleviate some of the added nonlinearity. This will also eliminate downstream dependency in the nonlinear weights in the discretization stencil. Unfortunately, as pointed out later by Lie et al. [116], this approach seems to only work well when the displacement profile between injector and producer consists of a concatenation of waves with increasing wave speeds, and fails to work, e.g., for water-alternating-gas injections and other displacement scenarios involving significant wave interactions. Instead, Lie et al. [116] claim that: *"Lagging the evaluation in the nonlinear iteration process does not seem to cause similar breakdown, but has little effect on the computational efficiency."* This conclusion refers to the use of a *global* nonlinear solver, and an interesting question is whether significant speedup can be achieved by introducing a *localized* solver. Lie et al. [116] think so, and continue: *"We believe a better approach would be to localize reconstruction to regions with significant fluid movement (...), and try to reuse previous nonlinear weights for stencils where changes in cell averages are below a prescribed value."* We believe that it may be possible

to use cell indicators discussed in Paper IX to mark cells that need implicit WENO weights and group these into supernodes in the flux graph. Likewise, one could use causality to systematically pick stencils that predominately involve upstream cells only. However, whether this will work well and provide significant speedup remains to be seen.

### Adaptivity

The dynamic coarsening framework presented in Paper IX is an interesting candidate for high-resolution reservoir simulation. However, further research on robust, mass-conservative prolongation and restriction of variables between grid blocks of different resolution (i.e., blocks originating from different partition levels) remains to be worked out, in particular for compositional flow. This includes developing strategies to mitigate the creation of (spurious) cycles in the interblock flux graph.

As mentioned in the conclusions of Paper IX, another possible topic of future research is to use adaptive coarsening as a nonlinear preconditioner for the fine-scale solution to the transport problem. It also seems possible to use arguments and indicators from [117] to ensure that dynamic coarse blocks have reasonable CFL numbers.

The multiscale method with dynamically adapted partitions introduced in Paper IV was only applied as the elliptic solver in a CPR preconditioner. Only lack of time has prevented us from investigating use of the same ideas within a sequentially fully implicit (SFI) framework. Moreover, combining the method with the localized nonlinear solver and dynamic coarsening seems like a promising candidate for highly efficient SI and SFI simulations. The dynamic coarsening framework can also be used here to construct high-quality multiscale partitions.

Altogether, I believe that further research on the topics presented in this thesis will result in a powerful suite of technologies that can be used in highly efficient next-generation reservoir simulators.

The research output of this thesis may also have applications in areas outside petroleum recovery – we mentioned simulation of geothermal energy systems, $CO_2$ storage, and freshwater aquifer management in Chapter 1. Moreover, whereas tailored for flow and transport in porous media, the local nonlinear solver strategy discussed in this work is generally applicable to implicit discretizations of transport problems with a strong (unidirectional) hyperbolic nature, whereas dynamic and feature-enriched multiscale methods may have applications to other elliptic Poisson-type problems in which the coefficients have abrupt spatial variations. Finally,

other researchers can leverage from the work in this thesis by use and further development of the published open-source software.

# Bibliography


[1] J. E. Aarnes. On the use of a mixed multiscale finite element method for greater flexibility and increased speed or improved accuracy in reservoir simulation. *Multiscale Model. Simul.*, 2(3):421–439, 2004. doi: 10.1137/030600655.

[2] J. E. Aarnes, V. Kippe, and K.-A. Lie. Mixed multiscale finite elements and streamline methods for reservoir simulation of large geomodels. *Adv. Water Resour.*, 28(3):257–271, 2005. doi: 10.1016/j.advwatres.2004.10.007.

[3] J. E. Aarnes, S. Krogstad, and K.-A. Lie. A hierarchical multiscale method for two-phase flow based upon mixed finite elements and nonuniform coarse grids. *Multiscale Model. Simul.*, 5(2):337–363, 2006. doi: 10.1137/050634566.

[4] J. E. Aarnes, S. Krogstad, K.-A. Lie, and J. R. Natvig. Fast sequential implicit porous media flow simulations using multiscale finite elements and reordering of cells for solution of nonlinear transport equation. In *ECMOR X - 10th European Conference on the Mathematics of Oil Recovery*, 2006. doi: 10.3997/2214-4609.201402519.

[5] J. E. Aarnes, V. L. Hauge, and Y. Efendiev. Coarsening of three-dimensional structured and unstructured grids for subsurface flow. *Adv. Water Resour.*, 30(11):2177–2193, 2007. doi: 10.1016/j.advwatres.2007.04.007.

[6] I. Aavatsmark. An introduction to multipoint flux approximations for quadrilateral grids. *Comput. Geosci.*, 6(3-4):405–432, 2002. doi: 10.1023/A:1021291114475.

[7] I. Aavatsmark. Interpretation of a two-point flux stencil for skew parallelogram grids. *Comput. Geosci.*, 11(3):199–206, 2007. doi: 10.1007/s10596-007-9042-1.

[8] I. Aavatsmark, T. Barkve, Ø. Bøe, and T. Mannseth. Discretization on non-orthogonal, curvilinear grids for multi-phase flow. In *ECMOR IV - 4th European Conference on the Mathematics of Oil Recovery*, 1994. doi: 10.3997/2214-4609.201411179.

[9] B. Ahmad, A. Alsaedi, F. Brezzi, L. D. Marini, and A. Russo. Equivalent projectors for virtual element methods. *Comput. Math. Appl.*, 66(3):376–391, 2013. doi: 10.1016/j.camwa.2013.05.015.

[10] M. A. Amooie and J. Moortgat. Higher-order black-oil and compositional modeling of multiphase compressible flow in porous media. *Int. J. Multiph. Flow*, 105(December):45–59, 2018. doi: 10.1016/j.ijmultiphaseflow.2018.03.016.

[11] O. Andersson. Aquifer thermal energy storage. In *Thermal Energy Storage for Sustainable Energy Consumption*, chapter 8, pages 155–176. Springer, 2007. ISBN 978-1-4020-5288-0. doi: 10.1007/978-1-4020-5290-3.

[12] J. R. Appleyard and I. M. Cheshire. The Cascade method for accelerated convergence in implicit simulators. In *European Petroleum Conference*, 1982. doi: 10.2118/12804-MS.

[13] T. Arbogast, L. C. Cowsar, M. F. Wheeler, and I. Yotov. Mixed finite element methods on nonmatching multiblock grids. *SIAM J. Numer. Anal.*, 37(4):1295–1315, 2000. doi: 10.1137/S0036142996308447.

[14] T. Arbogast, C.-s. Huang, and X. Zhao. Accuracy of WENO and adaptive order WENO reconstructions for solving conservation laws. *SIAM J. Numer. Anal.*, 56(3):181–1847, 2018. doi: 10.1137/17M1154758.

[15] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM J. Numer. Anal.*, 39(5):1749–1779, 2002. doi: 10.1137/S0036142901384162.

[16] L. Baker. Three-phase relative permeability correlations. In *SPE Enhanced Oil Recovery Symposium*, 1988. doi: 10.2118/17369-MS.

[17] K. Bao, K.-A. Lie, O. Møyner, and M. Liu. Fully implicit simulation of polymer flooding with MRST. *Comput. Geosci.*, 42(3):545–563, 2017. doi: 10.3997/2214-4609.201601880.

[18] J. W. Barker and S. Thibeau. A critical review of the use of pseudorelative permeabilities for upscaling. *SPE Reservoir Eng.*, 12(02):138–143, 1997. doi: 10.2118/35491-pa.

[19] D. W. V. Batenburg, M. Bosch, P. M. Boerrigter, A. H. De Zwart, and J. C. Vink. Application of dynamic gridding techniques to ior/eor-processes. In *SPE Reservoir Simulation Symposium*, 2011. doi: 10.2118/141711-MS.

[20] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind. Automatic differentiation in machine learning: a survey. *J. Mach. Learn. Res.*, 18(14):1–43, 2018. URL http://jmlr.org/papers/v18/16-107.html.

[21] L. Beirão da Veiga, F. Brezzi, A. Cangiani, G. Manzini, D. L. Marini, and A. Russo. Basic principles of virtual element methods. *Math. Model Meth. Appl. Sci.*, 23(01):199–214, 2013. doi: 10.1142/S0218202512500492.

[22] L. Beirão da Veiga, F. Brezzi, L. D. Marini, and A. Russo. The hitchhiker's guide to the virtual element method. *Math. Model Meth. Appl. Sci.*, 24(08): 1541–1573, 2014. doi: 10.1142/S021820251440003X.

[23] L. Beirão da Veiga, K. Lipnikov, and G. Manzini. *The Mimetic Finite Difference Method for Elliptic Problems*, volume 11. Springer, New York, 2014. doi: 10.1007/978-3-319-02663-3.

[24] J. B. Bell, J. Trangenstein, and G. Shubin. Conservation laws of mixed type describing three-phase flow in porous media. *SIAM J. Appl. Math.*, 46(6): 1000–1017, 1986. doi: 10.1137/0146059.

[25] J. B. Bell, P. Colella, and J. A. Trangenstein. Higher order Godunov methods for general systems of hyperbolic conservation laws. *J. Comput. Phys.*, 82(2): 362–397, 1989. doi: 10.1016/0021-9991(89)90054-5.

[26] R. L. Berge. Unstructured PEBI-grids adapting to geological features in subsurface reservoirs. Master's thesis, Norwegian University of Science and Technology, 2016. URL http://hdl.handle.net/11250/2411565.

[27] L. Bilke, B. Flemisch, T. Kalbacher, O. Kolditz, R. Helmig, and T. Nagel. Development of open-source porous media simulators: Principles and experiences. *Transport Porous Media*, 2019. doi: 10.1007/s11242-019-01310-1.

[28] R. D. Blumofe and C. E. Leiserson. Scheduling multithreaded computations by work stealing. *Journal of the ACM*, 46(5):720–748, 1999. doi: 10.1145/324133.324234.

[29] M. Blunt and B. Rubin. Implicit flux limiting schemes for petroleum reservoir simulation. *J. Comput. Phys.*, 102(1):194–210, 1992. doi: 10.1016/S0021-9991(05)80015-4.

[30] M. J. Blunt. *Multiphase Flow in Permeable Media: A Pore-Scale Perspective*. Cambridge University Press, 2017. ISBN 9781316145098. doi: 10.1017/9781316145098.

[31] L. Branets, S. S. Ghai, S. L. Lyons, and X.-H. Wu. Efficient and accurate reservoir modeling using adaptive gridding with global scale up. In *SPE Reservoir Simulation Symposium*, 2009. doi: 10.2118/118946-ms.

[32] Y. Brenier and J. Jaffré. Upstream differencing for multiphase flow in reservoir simulation. *SIAM J. Numer. Anal.*, 28(3):685–696, 1991. doi: 10.1137/0728036.

[33] F. Brezzi and M. Fortin. *Mixed and Hybrid Finite Element Methods*. Springer, New York, 1991. doi: 10.1007/978-1-4612-3172-1.

[34] F. Brezzi, K. Lipnikov, and V. Simoncini. A family of mimetic finite difference methods on polygonal and polyhedral meshes. *Math. Model Meth. Appl. Sci.*, 15(10):1533–1551, 2005. doi: 10.1142/S0218202505000832.

[35] R. Brooks and T. Corey. Hydraulic properties of porous media and their relation to drainage design. *Transactions of the ASAE*, 7(1):26–28, 1964.

[36] X.-C. Cai and D. E. Keyes. Nonlinearly preconditioned inexact newton algorithms. *SIAM J. Sci. Comput.*, 24(1):183–200, 2003. doi: 10.1137/s106482750037620x.

[37] H. Cao. *Development of Techniques for General Purpose Simulators*. PhD thesis, Stanford University, 2002.

[38] H. Cao and K. Aziz. Performance of IMPSAT and IMPSAT-AIM models in compositional simulation. In *SPE Annual Technical Conference and Exhibition*, 2002. doi: 10.2118/77720-MS.

[39] X.-D. L. Chan, S. Osher, and T. Chan. Weighted essentially non-oscillatory schemes. *J. Comput. Phys.*, 115(1):200–212, 1994. doi: 10.1006/jcph.1994.1187.

[40] M. Christie. Upscaling for reservoir simulation. *J. Petrol. Tech.*, 48(11), 1996. doi: 10.2118/37324-jpt.

[41] M. A. Christie and M. J. Blunt. Tenth SPE comparative solution project: A comparison of upscaling techniques. *SPE Reservoir Eval. Eng.*, 4(4):308–316, 2001. doi: 10.2118/72469-PA.

[42] CMG Computer Modelling Group Ltd. Computer Modelling Group, 2019. URL https://www.cmgl.ca/.

[43] K. Coats. IMPES stability: The CFL limit. In *SPE Reservoir Simulation Symposium*, 2001. doi: 10.2118/66345-MS.

[44] K. H. Coats. A note on impes and some impes-based simulation models. *SPE J.*, 2000. doi: 10.2118/65092-PA.

[45] B. Cockburn and C.-W. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws ii: General framework. *Mathematics of Computation*, 52(186):411, 1989. doi: 10.2307/2008474.

[46] B. Cockburn and C.-W. Shu. The Runge-Kutta local projection P1-Discontinuous-Galerkin finite element method for scalar conservation laws. *ESAIM Math. Model. Numer. Anal.*, 25(3):337–361, 1991.

[47] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2009. ISBN 780262533058.

[48] M. Cusini and H. Hajibeygi. Algebraic dynamic multilevel (ADM) method for simulations of multiphase flow with an adaptive saturation interpolator. In *ECMOR XVI - 16th European Conference on the Mathematics of Oil Recovery*, 2018. doi: 10.3997/2214-4609.201802254.

[49] M. Cusini, A. A. Lukyanov, J. R. Natvig, and H. Hajibeygi. Constrained pressure residual multiscale (CPR-MS) method for fully implicit simulation of multiphase flow in porous media. *J. Comput. Phys.*, 299:472–486, 2015. doi: 10.1016/j.jcp.2015.07.019.

[50] M. Cusini, C. van Kruijsdijk, and H. Hajibeygi. Algebraic dynamic multilevel (ADM) method for fully implicit simulations of multiphase flow in porous media. *J. Comput. Phys.*, 314:60–79, 2016. doi: 10.1016/j.jcp.2016.03.007.

[51] D. Debaun, T. Byer, P. Childs, J. Chen, F. Saaf, M. Wells, J. Liu, H. Cao, L. Pianelo, V. Tilakraj, P. Crumpton, D. Walsh, H. Yardumian, R. Zorzynski, K. Lim, M. Schrader, V. Zapata, C. Texaco, J. Nolen, and H. Tchelepi. An extensible architecture for next generation scalable parallel reservoir simulation. In *SPE Reservoir Simulation Symposium*, 2005. doi: 10.2118/93274-MS.

[52] L. Delpopolo Carciopolo, L. Bonaventura, A. Scotti, and L. Formaggia. A conservative implicit multirate method for hyperbolic problems. *Comput. Geosci.*, pages 647–664, 2018. doi: 10.1007/s10596-018-9764-2.

[53] P. Deuflhard. *Newton Methods for Nonlinear Problems*, volume 35. Springer, 2011. ISBN 978-3-642-23898-7. doi: 10.1007/978-3-642-23899-4.

[54] X. Y. Ding and L. S. Fung. An unstructured gridding method for simulating faulted reservoirs populated with complex wells. In *SPE Reservoir Simulation Symposium*, 2015. doi: 10.2118/173243-ms.

[55] H. Dong and M. J. Blunt. Pore-network extraction from micro-computerized-tomography images. *Phys. Rev. E*, 80(3):1–11, 2009. doi: 10.1103/PhysRevE.80.036307.

[56] L. J. Durlofsky, R. C. Jones, and W. J. Milliken. A nonuniform coarsening approach for the scale-up of displacement processes in heterogeneous porous media. *Adv. Water Resour.*, 20(5):335–347, 1997. doi: 10.1016/S0309-1708(96)00053-X.

[57] M. G. Edwards. A higher-order godunov scheme coupled with dynamic local grid refinement for flow in a porous medium. *Comput. Methods Appl. Mech. Engrg.*, 131(3-4):287–308, 1996. doi: 10.1016/0045-7825(95)00935-3.

[58] M. G. Edwards and C. F. Rogers. A flux continuous scheme for the full tensor pressure equation. In *ECMOR IV - 4th European Conference on the Mathematics of Oil Recovery*, 1994. doi: 10.3997/2214-4609.201411178.

[59] S. Ehrmann, S. Gries, and M. A. Schweitzer. Generalization of algebraic multiscale to algebraic multigrid. *Comput. Geosci.*, 2019. doi: 10.1007/s10596-019-9826-0.

[60] B. Eikemo, K.-A. Lie, G. T. Eigestad, and H. K. Dahle. Discontinuous Galerkin methods for advective transport in single-continuum models of fractured media. *Adv. Water Resour.*, 32(4):493–506, 2009. doi: 10.1016/j.advwatres.2008.12.010.

[61] Y. Epshteyn and B. Rivière. Fully implicit discontinuous finite element methods for two-phase flow. *Appl. Numer. Math.*, 57(4):383–401, 2007. doi: 10.1016/j.apnum.2006.04.004.

[62] L. C. Evans. *Partial Differential Equations*. American Mathematical Society, 1998. ISBN 9780821807729.

[63] R. Eymard, C. Guichard, and R. Herbin. Small-stencil 3D schemes for diffusive flows in porous media. *ESAIM-Math. Model. Num.*, 46(2):265–290, 2012. doi: 10.1051/m2an/2011040.

[64] R. Eymard, C. Guichard, R. Herbin, and R. Masson. Vertex-centred discretization of multiphase compositional Darcy flows on general meshes. *Comput. Geosci.*, 16(4):987–1005, 2012. doi: 10.1007/s10596-012-9299-x.

[65] C. L. Farmer. Upscaling: A review. *Int. J. Numer. Meth. Fluid.*, 40(1-2):63–78, 2002. doi: 10.1002/fld.267.

[66] D. Filippov, I. Kudryashov, D. Maksimov, D. Mitrushkin, A. Roshchektaev, and B. Vasekin. Complex-structured reservoir modeling on dynamically adaptive PEBI-grids. In *ECMOR XVI - 16th European Conference on the Mathematics of Oil Recovery*, 2018. doi: 10.3997/2214-4609.201802152.

[67] L.-K. Fung, A. Hiebert, and L. Nghiem. Reservoir simulation with a control-volume finite-element method. *SPE Reservoir Eng.*, 7(03):349–357, 1992. doi: 10.2118/21224-pa.

[68] M. G. Gerritsen and L. J. Durlofsky. Modeling fluid flow in oil reservoirs. *Annu. Rev. Fluid Mech.*, 37(1):211–238, 2005. doi: 10.1146/annurev.fluid.37.061903.175748.

[69] S. Gries, K. Stüben, G. L. Brown, D. Chen, and D. a. Collins. Preconditioning for efficiently applying algebraic multigrid in fully implicit reservoir simulations. *SPE J.*, 19(04), 2014. doi: 10.2118/163608-MS.

[70] S. Gries, B. Metsch, K. M. Terekhov, and P. Tomin. System-AMG for fully coupled reservoir simulation with geomechanics. In *SPE Reservoir Simulation Conference*, 2019. doi: 10.2118/193887-MS.

[71] A. N. Guion, B. Skaflestad, and K.-a. Lie. Validation of a non-uniform coarsening and upscaling framework. In *SPE Reservoir Simulation Conference*, 2019. doi: 10.2118/193891-MS.

[72] M. H. Gutknecht. A brief introduction to Krylov space methods for solving linear systems. In *Frontiers of Computational Science*, pages 53–62, 2007. doi: 10.1007/978-3-540-46375-7{_}5.

[73] H. Hajibeygi and H. A. Tchelepi. Compositional multiscale finite-volume formulation. *SPE J.*, 19(02):316–326, 2013. doi: 10.2118/163664-pa.

[74] H. Hajibeygi, G. Bonfigli, M. a. Hesse, and P. Jenny. Iterative multiscale finite-volume method. *J. Comput. Phys.*, 227:8604–8621, 2008. doi: 10.1016/j.jcp.2008.06.013.

[75] D. Han, D. Han, C. Yan, and L. Peng. A more flexible approach of dynamic local grid refinement for reservoir modeling. In *SPE Reservoir Simulation Symposium*, 1987. doi: 10.2523/16014-ms.

[76] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes, III. *J. Comput. Phys.*, 71 (1):231–303, 1987. doi: 10.1016/0021-9991(87)90031-3.

[77] V. L. Hauge, K.-A. Lie, and J. R. Natvig. Flow-based coarsening for multiscale simulation of transport in porous media. *Comput. Geosci.*, 16(2):391–408, 2012. doi: 10.1007/s10596-011-9230-x.

[78] J. Haukas. *Compositional Reservoir Simulation With Emphasis on the IMPSAT Formulation*. PhD thesis, University of Bergen, 2006.

[79] Z. Heinemann, C. Brand, M. Munka, and Y. Chen. Modeling reservoir geometry with irregular grids. *SPE Reservoir Eng.*, 6(02):225–232, 1991. doi: 10.2118/18412-pa.

[80] M. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Natl. Bur. Stand.*, 49(6):409, 1952. doi: 10.6028/jres.049.044.

[81] H. Holden and N. H. Risebro. *Front Tracking for Hyperbolic Conservation Laws*. Springer, 2015. ISBN 9783662475065. doi: 10.1007/978-3-662-47507-2.

[82] J. Holmes, T. Barkve, and O. Lund. Application of a multisegment well model to simulate flow in advanced wells. In *SPE European Petroleum Conference*, 1998. doi: 10.2523/50646-ms.

[83] H. Hoteit and A. Chawathe. Making field-scale chemical enhanced-oil-recovery simulations a practical reality with dynamic gridding. *SPE J.*, 21 (06), 2016. doi: 10.2118/169688-PA.

[84] H. Hoteit and A. Firoozabadi. Multicomponent fluid flow by discontinuous Galerkin and mixed methods in unfractured and fractured media. *Water Resour. Res.*, 41(11):1–15, 2005. doi: 10.1029/2005WR004339.

[85] T. Y. Hou and X.-H. Wu. A multiscale finite element method for elliptic problems in composite materials and porous media. *J. Comput. Phys.*, 134 (1):169–189, 1997. doi: 10.1006/jcph.1997.5682.

[86] International Energy Agency. Energy technology perspectives 2016. Technical report, International Energy Agency, 2016. URL https://www.oecd-ilibrary.org/energy/energy-technology-perspectives-2015_energy_tech-2015-en.

[87] N. IO Center. The Norne Benchmark Case, 2019. URL http://www.ipt.ntnu.no/~norne/.

[88] P. Jenny, S. H. Lee, and H. A. Tchelepi. Multi-scale finite-volume method for elliptic problems in subsurface flow simulation. *J. Comput. Phys.*, 187(1): 47–67, 2003. doi: 10.1016/S0021-9991(03)00075-5.

[89] P. Jenny, S. H. Lee, and H. Tchelepi. Adaptive multiscale finite-volume method for multiphase flow and transport in porous media. *Multiscale Model. Simul.*, 3(1):50–64, 2004. doi: 10.1137/030600795.

[90] P. Jenny, S. H. Lee, and H. A. Tchelepi. Adaptive fully implicit multi-scale finite-volume method for multi-phase flow and transport in heterogeneous porous media. *J. Comput. Phys.*, 217(2):627–641, 2006. doi: 10.1016/j.jcp.2006.01.028.

[91] P. Jenny, H. A. Tchelepi, and S. H. Lee. Unconditionally convergent nonlinear solver for hyperbolic conservation laws with S-shaped flux functions. *J. Comput. Phys.*, 228(20):7497–7512, 2009. doi: 10.1016/j.jcp.2009.06.032.

[92] M. Karimi-Fard. Unstructured adaptive mesh refinement for flow in heterogeneous porous media. In *ECMOR XIV - 14th European Conference on the Mathematics of Oil Recovery*, 2014. doi: 10.3997/2214-4609.20141856.

[93] M. Karimi-Fard and L. J. Durlofsky. An unstructured dual-grid model for flow in fractured and heterogeneous porous media. In *ECMOR XVI - 16th European Conference on the Mathematics of Oil Recovery*, 2018. doi: 10.3997/2214-4609.201802264.

[94] M. Khait and D. V. Voskov. GPU-offloaded general purpose simulator for multiphase flow in porous media. In *SPE Reservoir Simulation Conference*, 2017. doi: 10.2118/182663-ms.

[95] M. Khait and D. V. Voskov. Operator-based linearization for general purpose reservoir simulation. *J. Petrol. Sci. Eng.*, 157(November 2016):990–998, 2017. doi: 10.1016/j.petrol.2017.08.009.

[96] J. Killough. Ninth SPE comparative solution project: A reexamination of black-oil simulation. In *SPE Reservoir Simulation Symposium*, 1995. doi: 10.2118/29110-MS.

[97] Ø. S. Klemetsdal, O. Møyner, and K.-A. Lie. Nonlinear gauss-seidel solvers with higher order for black-oil models. In *ECMOR XVI – 16th European Conference on the Mathematics of Oil Recovery*, 2018. doi: 10.3997/2214-4609.201802130.

[98] Ø. S. Klemetsdal, O. Møyner, K.-A. Lie, and T. Mykkeltvedt. Comparison of implicit discontinuous Galerkin and WENO schemes on stratigraphic and unstructured grids (conference presentation). SIAM Conference on Mathematical & Computational Issues in the Geosciences (GS19), 2019.

[99] A. Kozlova, Z. Li, J. R. Natvig, S. Watanabe, Y. Zhou, K. Bratvedt, and S. H. Lee. A real-field multiscale black-oil reservoir simulator. *SPE J.*, 21(06):2049–2061, 2016. doi: 10.2118/173226-PA.

[100] S. Krogstad, K.-A. Lie, O. Møyner, H. M. Nilsen, X. Raynaud, and B. Skaflestad. MRST-AD – an open-source framework for rapid prototyping and evaluation of reservoir simulation problems. *SPE Reservoir Simulation Symposium*, 2015. doi: 10.2118/173317-MS.

[101] Y. Kuznetsov, K. Lipnikov, and M. Shashkov. The mimetic finite difference method on polygonal meshes for diffusion-type problems. *Comput. Geosci.*, 8(4):301–324, 2004. doi: 10.1007/s10596-004-3771-1.

[102] F. Kwok and H. Tchelepi. Potential-based reduced Newton algorithm for nonlinear multiphase flow in porous media. *J. Comput. Phys.*, 227(1):706–727, 2007. doi: 10.1016/j.jcp.2007.08.012.

[103] S. Lacroix, Y. Vassilevski, J. Wheeler, and M. F. Wheeler. Iterative solution methods for modeling multiphase flow in porous media fully implicitly. *SIAM J. Sci. Comput.*, 25(3):905–926, 2003. doi: 10.1137/S106482750240443X.

[104] C. Lanczos. Solution of systems of linear equations by minimized iterations. *J. Res. Natl. Bur. Stand.*, 49(1):33, 1952. doi: 10.6028/jres.049.006.

[105] Landmark. Nexus Reservoir Simulator, 2019. URL https://www.landmark.solutions/Nexus-Reservoir-Simulation.

[106] A. Lauser, A. F. Rasmussen, T. Sandve, and H. Nilsen. Local forward-mode automatic differentiation for high performance parallel pilot-level reservoir simulation. In *ECMOR XVI - 16th European Conference on the Mathematics of Oil Recovery*, 2018. doi: 10.3997/2214-4609.201802153.

[107] G. Lavie and A. M. Kamp. An evaluation of criteria for dynamic gridding during reservoir simulation of water flooding and solvent injection. In *SPE Reservoir Simulation Symposium*, The Woodlands, Texas, 2011. doi: 10.2118/141796-ms.

[108] C. Le Potier. A nonlinear finite volume scheme satisfying maximum and minimum principles for diffusion operators. *Int. J. Finite Vol.*, 6(2):1–20, 2009.

[109] S. H. Lee, C. Wolfsteiner, and H. A. Tchelepi. Multiscale finite-volume formulation for multiphase flow in porous media: Black oil formulation of compressible, three-phase flow with gravity. *Comput. Geosci.*, 12(3):351–366, 2008. doi: 10.1007/s10596-007-9069-3.

[110] B. Li and H. A. Tchelepi. Unconditionally convergent nonlinear solver for multiphase flow in porous media under viscous force, buoyancy, and capillarity. *Energy Procedia*, 59:404–411, 2014. doi: 10.1016/j.egypro.2014.10.395.

[111] K.-A. Lie. *An Introduction to Reservoir Simulation Using MATLAB/GNU Octave – User Guide for the MATLAB Reservoir Simulation Toolbox (MRST)*. Cambridge University Press, 2019. ISBN 9781108591416. doi: 10.1017/9781108591416.

[112] K.-A. Lie, J. R. Natvig, and H. M. Nilsen. Discussion of dynamics and operator splitting techniques for two-phase flow with gravity. *Int. J. Numer. Anal. Mod.*, 9(3):684–700, 2012.

[113] K.-A. Lie, H. M. Nilsen, A. F. Rasmussen, and X. Raynaud. Fast simulation of polymer injection in heavy-oil reservoirs on the basis of topological sorting and sequential splitting. *SPE J.*, 19(06):0991–1004, 2014. doi: 10.2118/163599-PA.

[114] K.-A. Lie, O. Møyner, and J. R. Natvig. Use of multiple multiscale operators to accelerate simulation of complex geomodels. *SPE J.*, 22(06):1–929, 2017. doi: 10.2118/182701-PA.

[115] K.-A. Lie, O. Møyner, J. R. Natvig, A. Kozlova, K. Bratvedt, S. Watanabe, and Z. Li. Successful application of multiscale methods in a real reservoir simulator environment. *Comput. Geosci.*, 21(5-6):981–998, 2017. doi: 10.1007/s10596-017-9627-2.

[116] K. A. Lie, T. S. Mykkeltvedt, and O. Møyner. A fully implicit weno scheme on stratigraphic and unstructured polyhedral grids. *Comput. Geosci.*, 2019. doi: 10.1007/s10596-019-9829-x.

[117] G. Linga, O. Møyner, H. Møll, A. Moncorgé, and K.-a. Lie. An implicit local time-stepping method based on cell reordering for multiphase flow in porous media. *J. Comput. Phys.*, 2019. Submitted for publication.

[118] K. Lipnikov, M. Shashkov, D. Svyatskiy, and Y. Vassilevski. Monotone finite volume schemes for diffusion equations on unstructured triangular and shape-regular polygonal meshes. *J. Comput. Phys.*, 227(1):492–512, 2007. doi: 10.1016/j.jcp.2007.08.008.

[119] B. T. Mallison, M. G. Gerritsen, K. Jessen, and F. M. Orr. High order upwind schemes for two-phase, multicomponent flow. *SPE J.*, 10(03):297–311, 2005. doi: 10.2118/79691-pa.

[120] A. M. Manea and T. Almani. A multi-level algebraic multiscale solver (ML-AMS) for reservoir simulation. In *ECMOR XVI - 16th European Conference on the Mathematics of Oil Recovery*, 2018. doi: 10.3997/2214-4609.201802253.

[121] A. M. Manea, J. Sewall, and H. A. Tchelepi. Parallel multiscale linear solver for highly detailed reservoir models. *SPE J.*, 21(06):2062–2078, 2016. doi: 10.2118/173259-pa.

[122] T. Manzocchi, J. N. Carter, A. Skorstad, B. Fjellvoll, K. D. Stephen, J. A. Howell, J. D. Matthews, J. J. Walsh, M. Nepveu, C. Bos, J. Cole, P. Egberts, S. Flint, C. Hern, L. Holden, H. Hovland, H. Jackson, O. Kolbjørnsen, A. MacDonald, P. A. R. Nell, K. Onyeagoro, J. Strand, A. R. Syversveen, A. Tchistiakov, C. Yang, G. Yielding, and R. W. Zimmerman. Sensitivity of the impact of geological uncertainty on production from faulted and unfaulted shallow-marine oil reservoirs: objectives and methods. *Petrol. Geosci.*, 14(1):3–15, 2008. doi: 10.1144/1354-079307-790.

[123] S. Manzoor, M. G. Edwards, A. H. Dogru, and T. M. Al-Shaalan. Interior boundary-aligned unstructured grid generation and cell-centered versus vertex-centered CVD-MPFA performance. *Comput. Geosci.*, 22(1):195–230, 2018. doi: 10.1007/s10596-017-9686-4.

[124] S. Manzoor, M. G. Edwards, and A. H. Dogru. Three-dimensional geological boundary aligned unstructured grid generation, and CVD-MPFA flow computation. In *SPE Reservoir Simulation Conference*, 2019. doi: 10.2118/193874-MS.

[125] Y. Mehmani and H. A. Tchelepi. Multiscale computation of pore-scale fluid dynamics: Single-phase flow. *J. Comput. Phys.*, 375:1469–1487, 2018. doi: 10.1016/j.jcp.2018.08.045.

[126] R. Merland, B. Lévy, and G. Caumon. Building pebi grids conforming to 3D geological features using centroidal voronoi tessellations. In *Proceedings of IAMG*, Salzburg, 2011. doi: 10.5242/iamg.2011.0064.

[127] R. Merland, G. Caumon, B. Lévy, and P. Collon-Drouaillet. Voronoi grids conforming to 3D structural features. *Comput. Geosci.*, 18(3-4):373–383, 2014. doi: 10.1007/s10596-014-9408-0.

[128] J. Moortgat. Adaptive implicit finite element methods for multicomponent compressible flow in heterogeneous and fractured porous media. *Water Resour. Res.*, 53:73–92, 2017. doi: 10.1002/2016WR019644.

[129] O. Møyner. *Next Generation Multiscale Methods for Reservoir Simulation*. PhD thesis, Norwegian University of Science and Technology, 2016.

[130] O. Møyner. Nonlinear solver for three-phase transport problems based on approximate trust regions. *Comput. Geosci.*, 21(5-6):999–1021, 7 2017. doi: 10.1007/s10596-017-9660-1.

[131] O. Møyner and K.-A. Lie. A multiscale restriction-smoothed basis method for high contrast porous media represented on unstructured grids. *J. Comput. Phys.*, 304:46–71, 2016. doi: 10.1016/j.jcp.2015.10.010.

[132] O. Møyner and K.-A. Lie. A multiscale restriction-smoothed basis method for compressible black-oil models. *SPE J.*, 21(06):2079–2096, 2016. doi: 10.2118/173265-PA.

[133] O. Møyner and H. A. Tchelepi. A mass-conservative sequential implicit multiscale method for isothermal equation-of-state compositional problems. *SPE J.*, 23(06):2376–2393, 2018. doi: 10.2118/182679-pa.

[134] W. A. Mulder and R. H. J. G. Meyling. Numerical simulation of two-phase flow using locally refined grids in three space dimensions. *SPE Advanced Technology Series*, 1(1):36–41, 1993. doi: 10.2118/21230-PA.

[135] B. Müller, F. Kummer, and M. Oberlack. Highly accurate surface and volume integration on implicit domains by means of moment-fitting. *Int. J. Numer. Methods Eng.*, 96(512):512–528, 2013. doi: 10.1002/nme.

[136] M. Muskat and R. D. Wyckoff. *The Flow of Homogeneous Fluids Through Porous Media*. McGraw-Hill, New York, 1937.

[137] T. S. Mykkeltvedt, X. Raynaud, and K.-A. Lie. Fully implicit higher-order schemes applied to polymer flooding. *Comput. Geosci.*, 21(5):1245–1266, 2017. doi: 10.1007/s10596-017-9676-6.

[138] 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.*, 227(24):10108–10124, 2008. doi: 10.1016/j.jcp.2008.08.024.

[139] J. R. 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. doi: 10.1016/j.advwatres.2007.05.015.

[140] J. R. Natvig, B. Skaflestad, F. Bratvedt, K. Bratvedt, K. A. Lie, V. Laptev, and S. K. Khataniar. Multiscale mimetic solvers for efficient streamline simulation of fractured reservoirs. *SPE J.*, 16(4):880–888, 2011. doi: 10.2118/119132-PA.

[141] R. D. Neidinger. Introduction to automatic differentiation and MATLAB object-oriented programming. *SIAM Rev.*, 52(3):545–563, 2010. doi: 10.1137/080743627.

[142] K. Nikitin, K. Terekhov, and Y. Vassilevski. A monotone nonlinear finite volume method for diffusion equations and multiphase flows. *Comput. Geosci.*, 18(3):311–324, 2013. doi: 10.1007/s10596-013-9387-6.

[143] OPM. The Open Porous Media (OPM) Initative, 2019. URL https://opm-project.org/.

[144] D. W. Peaceman. Interpretation of well-block pressures in numerical reservoir simulation with nonsquare grid blocks and anisotropic permeability. *SPE J.*, 18(3):183 – 194, 1978. doi: 10.2118/6893-PA.

[145] D. W. Peaceman. Interpretation of well-block pressures in numerical reservoir simulation. *SPE J.*, 18(3):183 – 194, 1978. doi: 10.2118/6893-PA.

[146] P.-o. Persson and G. Strang. A simple mesh generator in MATLAB. *SIAM Rev.*, 46(2):329–345, 2004. doi: 10.1137/S0036144503429121.

[147] D. Picchi and I. Battiato. The impact of pore-scale flow regimes on upscaling of immiscible two-phase flow in porous media. *Water Resour. Res.*, 54(9):6683–6707, 2018. doi: 10.1029/2018WR023172.

[148] D. K. Ponting. Corner point geometry in reservoir simulation. In *ECMOR I - 1st European Conference on the Mathematics of Oil Recovery*, 1989. doi: 10.3997/2214-4609.201411305.

[149] A. F. Rasmussen and K.-A. Lie. Discretization of flow diagnostics on stratigraphic and unstructured grids. In *ECMOR XIV - 14th European Conference on the Mathematics of Oil Recovery*, 2014. doi: 10.3997/2214-4609.20141844.

[150] X. Raynaud, K.-A. Lie, H. M. Nilsen, and A. F. Rasmussen. The single-cell transport problem for two-phase flow with polymer. *Comput. Geosci.*, 20(3):495–507, 2016. doi: 10.1007/s10596-015-9502-y.

[151] P. Renard and G. de Marsily. Calculating equivalent permeability: a review. *Adv. Water Resour.*, 20(5-6):253–278, 1997. doi: 10.1016/S0309-1708(96)00050-4.

[152] B. Rivière. *Discontinuous Galerkin Methods for Solving Elliptic and Parabolic Equations*. Society for Industrial and Applied Mathematics, Houston, Texas, 2008. ISBN 9780898716566. doi: 10.1137/1.9780898717440.

[153] B. Rivière and M. F. Wheeler. Discontinuous Galerkin methods for flow and transport problems in porous media. *Comm. Numer. Meth. En.*, 18(1):63–68, 2002. doi: 10.1002/cnm.464.

[154] Rock Flow Dynamics. tNavigator, 2019. URL https://rfdyn.com/tnavigator/.

[155] D. Russell and M. Prats. Performance of layered reservoirs with crossflow – single-compressible-fluid case. *SPE J.*, 2(01):53–67, 1962. doi: 10.2118/99-pa.

[156] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, 1986. doi: 10.1137/0907058.

[157] P. Samier and R. Masson. Implementation of a vertex-centered method inside an industrial reservoir simulator: Practical issues and comprehensive comparison with corner-point grids and perpendicular-bisector-grid models on a field case. *SPE J.*, 22(02):660–678, 2017. doi: 10.2118/173309-pa.

[158] V. Savcenco, W. Hundsdorfer, and J. G. Verwer. A multirate time stepping strategy for stiff ordinary differential equations. *BIT Numer. Math.*, 47(1): 137–155, 2007. doi: 10.1007/s10543-006-0095-7.

[159] Schlumberger. *ECLIPSE Technical Description*. ECLIPSE, 2013.

[160] Schlumberger. ECLIPSE Industry-Reference Reservoir Simulator, 2019. URL https://www.software.slb.com/products/eclipse.

[161] Schlumberger. INTERSECT High-Resolution Reservoir Simulator, 2019. URL https://www.software.slb.com/products/intersect.

[162] G. H. Schmidt and F. J. Jacobs. Adaptive local grid refinement and multigrid in numerical reservoir simulation. *J. Comput. Phys.*, 77(1):140–165, 1988. doi: 10.1016/0021-9991(88)90160-X.

[163] M. Schneider, B. Flemisch, R. Helmig, K. Terekhov, and H. Tchelepi. Monotone nonlinear finite-volume method for challenging grids. *Comput. Geosci.*, 22(2):565–586, 2018. doi: 10.1007/s10596-017-9710-8.

[164] M. Schneider, D. Gläser, B. Flemisch, and R. Helmig. Comparison of finite-volume schemes for diffusion problems. *Oil Gas Sci. Technol.*, 82(73):1–17, 2018. doi: 10.2516/ogst/2018064.

[165] A. Settari and K. Aziz. Use of Voronoi grid in reservoir simulation. *SPE Advanced Technology Series*, 2(2):69–77, 1994. doi: 10.2118/22889-PA.

[166] S. Sheth and R. Younis. Localized linear systems for sequential implicit simulation of flow and transport. *SPE J.*, 22(05), 2017. doi: 10.2118/173320-PA.

[167] S. M. Sheth and R. M. Younis. Localized solvers for general full-resolution implicit reservoir simulation. In *SPE Reservoir Simulation Conference*, 2017. doi: 10.2118/182691-MS.

[168] I. A. Shiklomanov. World water resources. Technical report, International Hydrological Programme, Paris, 1998.

[169] SINTEF Computational Geosciences. The MATLAB Reservoir Simulation Toolbox (MRST), 2019. URL https://www.mrst.no.

[170] T. Stone, N. R. Ednumds, and B. J. Kristoff. A comprehensive wellbore/reservoir simulator. In *SPE Reservoir Simulation Symposium*, 1989. doi: 10.2118/18419-MS.

[171] Stoneridge technology. ECHELON – The fastest, most scalable simulator in the world, 2019. URL https://stoneridgetechnology.com/echelon/.

[172] K. Stüben, T. Clees, H. Klie, B. Lu, and M. F. Wheeler. Algebraic multigrid methods (AMG) for the efficient solution of fully implicit formulations in reservoir simulation. In *SPE Reservoir Simulation Symposium*, 2007. doi: 10.2118/105832-MS.

[173] V. S. Suicmez, D. W. V. Batenburg, T. Matsuura, M. Bosch, and D. M. Boersma. Dynamic local grid refinement for multiple contact miscible gas injection. In *International Petroleum Technology Conference*, 2012. doi: 10.2523/IPTC-15017-MS.

[174] E. Süli and D. F. Meyers. *An introduction to numerical analysis*. Cambridge University Press, 2003. ISBN 9780511801181. doi: 10.1017/CBO9780511801181.

[175] S. Sun, B. Rivière, and M. F. Wheeler. A combined mixed finite element and discontinuous Galerkin method for miscible displacement problem in porous media. In *Recent Progress in Computational and Applied PDES*, pages 323–351. Springer, 2002.

[176] R. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972. doi: 10.1137/0201010.

[177] G. Thomas and D. Thurnau. Reservoir simulation using an adaptive implicit method. *SPE J.*, 23(05), 1983. doi: 10.2118/10120-PA.

[178] M. Todd and W. Longstaff. The development, testing and application of a numerical simulator for predicting miscible flood performance. *J. Petrol. Tech.*, 24(7):874–882, 1972. doi: 10.2118/3484-PA.

[179] S. M. Toor, M. G. Edwards, A. Dogru, and T. Shaalan. Boundary aligned grid generation in three dimensions and CVD-MPFA discretization. In *SPE Reservoir Simulation Symposium*, 2015. doi: 10.2118/173313-ms.

[180] J. A. Trangenstein and J. B. Bell. Mathematical structure of the black-oil model for petroleum reservoir simulation. *SIAM J. Appl. Math.*, 49(3):749–783, 1989. doi: 10.1137/0149044.

[181] U. Trottenberg, C. Oosterlee, and A. Schuller. *Multigrid*. Elsevier, 2000. ISBN 9780080479569.

[182] Utvinningsutvalget, Olje- og Energidepartementet. Økt utvinning på norsk kontinentalsokkel – en rapport fra utvinningsutvalget. Technical report, Olje- og energidepartementet, 2010. In Norwegian.

[183] H. Vikøren. Towards a parallel multiphase solver based on potential ordering. Master's thesis, Norwegian University of Technology and Science, 2015. URL http://hdl.handle.net/11250/2352705.

[184] H. A. v. d. Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 13(2):631–644, 1992.

[185] D. V. Voskov. Operator-based linearization approach for modeling of multiphase multi-component flow in porous media. *J. Comput. Phys.*, 337:275–288, 2017. doi: 10.1016/j.jcp.2017.02.041.

[186] D. V. Voskov and H. A. Tchelepi. Compositional nonlinear solver based on trust regions of the flux function along key tie-lines. *SPE Reservoir Simulation Symposium*, 2011. doi: 10.2118/141743-MS.

[187] D. V. Voskov and H. A. Tchelepi. Comparison of nonlinear formulations for two-phase multi-component eos based simulation. *J. Petrol. Sci. Eng.*, 82-83: 101–111, 2012. doi: 10.1016/j.petrol.2011.10.012.

[188] D. V. Voskov, H. A. Tchelepi, and R. Younis. General nonlinear solution strategies for multiphase multicomponent EoS based simulation. In *SPE Reservoir Simulation Symposium*, 2009. doi: 10.2118/118996-ms.

[189] J. Wallis. Incomplete Gaussian elimination as a preconditioning for generalized conjugate gradient acceleration. In *SPE Reservoir Simulation Symposium*, 1983. doi: 10.2118/12265-MS.

[190] J. Wallis, R. Kendall, and T. Little. Constrained residual acceleration of conjugate residual methods. In *SPE Reservoir Simulation Symposium*, 1985. doi: 10.2118/13536-MS.

[191] X. Wang and H. A. Tchelepi. Trust-region based solver for nonlinear transport in heterogeneous porous media. *J. Comput. Phys.*, 253:114–137, 2013. doi: 10.1016/j.jcp.2013.06.041.

[192] Y. Wang, H. Hajibeygi, and H. A. Tchelepi. Algebraic multiscale solver for flow in heterogeneous porous media. *J. Comput. Phys.*, 259:284–303, 2014. doi: 10.1016/j.jcp.2013.11.024.

[193] J. Watts. A compositional formulation of the pressure and saturation equations. *SPE Reservoir Eng.*, 1(03):243–252, 1986. doi: 10.2118/12244-PA.

[194] Z. Y. Wong. *Sequential-Implicit Newton's Method for Geothermal Reservoir Simulation*. PhD thesis, Stanford University, 2018.

[195] X.-H. Wu and R. Parashkevov. Effect of grid deviation on flow solutions. *SPE J.*, 14(01):67–77, 2009. doi: 10.2118/92868-pa.

[196] R. Younis. *Modern advances in software and solution algorithms for reservoir simulation*. PhD thesis, Stanford University, 2011.

[197] R. Younis and K. Aziz. Parallel automatically differentiable data-types for next-generation simulator development. In *SPE Reservoir Simulation Symposium*, 2007. doi: 10.2118/106493-ms.

[198] Y. Zhou, H. A. Tchelepi, and B. T. Mallison. Automatic differentiation framework for compositional simulation on unstructured grids with multi-point discretization schemes. In *SPE Reservoir Simulation Symposium*, 2011. doi: 10.2118/141592-ms.

# Part II

# Scientific Papers

# Paper I

## Unstructured Gridding and Consistent Discretizations for Reservoirs With Faults and Complex Wells

Øystein S. Klemetsdal, Runar Lie Berge, Knut-Andreas Lie, Halvor Møll Nilsen, Olav Møyner

*In proceedings of the 2017 SPE Reservoir Simulation Conference, Montgomery, Texas, USA*
*DOI: 10.2118/182666-MS*

# Paper II

## Unstructured Voronoi Grids Conforming to Lower-dimensional Objects

Runar Lie Berge, Øystein S. Klemetsdal, Knut-Andreas Lie

*Computational Geosciences, volume 23, issue 1, pp. 169–188, 2019*
*DOI: 10.1007/s10596-018-9790-0*

# Paper III

## A Comparison of Consistent Discretizations for Elliptic Poisson-Type Problems on Unstructured Polyhedral Grids

Øystein S. Klemetsdal, Olav Møyner, Xavier Raynaud, Knut-Andreas Lie

*Manuscript in preparation, 2019*

# Paper IV

## Accelerating Multiscale Simulation of Complex Geomodels by Use of Dynamically Adapted Basis Functions

Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie

*Computational Geosciences, published ahead of print, 2019*
*DOI: 10.1007/s10596-019-9827-z*

# Paper V

## Non-linear Newton Solver for a Polymer Two-phase System Using Interface-localized Trust Regions

Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie

*In proceedings of the 19th European Symposium on Improved Oil Recovery, 2017, Stavanger, Norway*
*DOI: 10.3997/2214-4609.201700356*

# Paper VI

## Robust Nonlinear Newton Solver with Adaptive Interface-Localized Trust Regions

Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie

*SPE Journal, volume 24, issue 4, pp. 1576–1594, 2019*
*DOI: 10.2118/195682-PA*

# Paper VII

## Efficient Reordered Nonlinear Gauss-Seidel Solvers With Higher Order For Black-Oil Models

Øystein S. Klemetsdal, Atgeirr Flø Rasmussen, Olav Møyner, Knut-Andreas Lie

*Computational Geosciences, published ahead of print, 2019*
*DOI: 10.1007/s10596-019-09844-5*

# Paper VIII

## Implicit High-resolution Compositional Simulation with Optimal Ordering of Unknowns and Adaptive Spatial Refinement

Øystein S. Klemetsdal, Olav Møyner, Knut-Andreas Lie

*In proceedings of the 2019 SPE Reservoir Simulation Conference, Galveston, Texas, USA*
*DOI: 10.2118/193934-MS*

# Paper IX

## Dynamic Coarsening and Local Reordered Nonlinear Solvers for Simulating Transport in Porous Media

Øystein S. Klemetsdal, Knut-Andreas Lie

*Manuscript in preparation, 2019*

ISBN 978-82-326-4346-2 (printed version)
ISBN 978-82-326-4347-9 (electronic version)
ISSN 1503-8181

NTNU
Norwegian University of
Science and Technology

Doctoral Thesis

Øystein Strengehagen Klemetsdal

Doctoral theses at NTNU, 2019:374

Øystein Strengehagen Klemetsdal

# Efficient Solvers for Field-Scale Simulation of Flow and Transport in Porous Media

**NTNU**
Norwegian University of
Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Mathematical Sciences

Doctoral theses at NTNU, 2019:374

NTNU

NTNU
Norwegian University of
Science and Technology