NTNU
Norwegian University of
Science and Technology

A General Nonlinear Reservoir Simulator with the Full Approximation Scheme

Raymond Toft

Master of Science in Physics and Mathematics
Submission date: June 2018
Supervisor: Knut Andreas Lie, IMF
Co-supervisor: Olav Møyner, IMF

Norwegian University of Science and Technology
Department of Mathematical Sciences

Abstract

Simulation of multiphase flow and transport in porous rock formations gives rise to large systems of strongly coupled nonlinear equations. Solving these equations is computationally challenging because of orders of magnitude local variations in parameters, mixed hyperbolic-elliptic character, grids with high aspect ratios, and strong coupling between local and global flow effects.

The state-of-the-art solution approach is to use a Newton-type solver with an algebraic multigrid preconditioner for the elliptic part of the linearized system. Herein, we discuss the use and implementation of a full approximation scheme (FAS), in which algebraic multigrid is applied on a nonlinear level. By use of this method, global and semi-global nonlinearities can be resolved on the appropriate coarse scale.

Improved nonlinear convergence is demonstrated on standard benchmark cases from the petroleum literature. The method is implemented in the solver framework of the open-source Matlab Reservoir Simulation Toolbox (MRST). With this framework, the implemented FAS method can be applied on a broad range of classes of discrete reservoir and fluid models.

Sammendrag

Simulering av flerfaseflyt og transport i porøse steinformasjoner gir grunnlag for store systemer med sterkt koblede ikkelineære ligninger. Løsningen av disse ligningene er beregningsmessig utfordrende som følge av størrelsen på de lokale variasjonene i parametre, blandet hyperbolsk-elliptiske karakter, grid med høye størrelsesforhold, og sterke koblinger mellom lokale og globale flyteffekter.

En standard løsningsmetode for industrien er å anvende en Newton type løser med en algebraisk flergrid prekondisjonering for den elliptiske delen av det lineariserte systemet. Innunder dette diskuterer vi bruken og implementasjonen av full approximation scheme (FAS), hvor algebraisk flergrid blir anvendt på et ikkelineært nivå. Ved bruk av denne metoden, kan globale og semi-globale ikkelineariteter bli løst på en tilfredsstillende grov skala.

Det er demonstrert forbedret ikkelineær konvergering ved standard referansemålinger hentet fra litteraturen innen petroleum. Metoden er implementert i løsningsrammeverket fra den åpen-kilde Matlab Reservoir Simulation Toolbox (MRST). Med dette rammeverket, kan den implementerte FAS metoden annvendes på et bredt spekter av klasser med diskrete reservoir og fluidmodeller.

Preface

This thesis is my final work in the Master's program Physics and Mathematics at the Norwegian University of Science and Technology (NTNU). My specialization has been within numerical mathematics. This thesis is written in cooperation with SINTEF Digital, Mathematics and Cybernetics department in Oslo.

First, I would like to thank my two supervisors at SINTEF, Knut-Andreas Lie and Olav Moyner, who provided excellent support during this thesis. This thesis have taken form through our numerous and helpful discussions. Especially the sparring with Olav have provided deeper insight into the world of MRST and reservoir simulation in general. They also contributed by proofreading the thesis and providing plenty of useful feedback.

The HPC-lab and the director, Anne C. Elster at NTNU also deserves credit. The access to this lab have helped tremendously towards writing the code and executing the numerous tests. It have been a great experience to be a part of the HPC-lab.

Finally I would like to thank all my dear friends and family who have helped and supported me trough many a frustrating period. Among these I would like to thank you who have helped me proofread my thesis: Magnus, Stian, Sara and Mikkel, your aid have raised my thesis to a much higher level.

With this, an era has finally come to an end. A new dawn is about to arise from the valley of years of hard work and dedication towards finalizing my masters degree.

Trondheim, June 26, 2018

Raymond Toft

Abbreviations and frequently used symbols

Abbreviations

NTNU Norwegian University of Science and Technology

MRST Matlab Reservoir Simulation Toolbox

FAS Full Approximation Scheme

AMG Algebraic Multigrid

CPR Constrained Pressure Residual

ILU(0) Incomplete LU factorization

GMRES Generalized Minimal Residual

FV Finite Volume

PDE Partial Differential Equation

QFS Quarter-five-spot

Frequently used symbols

ρ Density in kg/m^3

ϕ Porosity, fraction of material allowing for fluid flow

ψ Flux, flow rate through a surface in kg/sm^2

α Fluid phase

K Permeability, resistance to flow in a porous media in milli darcy (md)

k_r Relative permeability, tension between phases and relation to the porous media

S Saturation, fraction of volume occupied by each phase

λ Mobility, ratio of relative permeability to phase viscosity

μ Viscosity in kg/ms

Contents

Chapter 1

Introduction

Enormous amounts of oil and gas are pumped up every day offshore on the Norwegian plateau. The world is still in need of stable supply of petroleum resources until the technology of green sources can deliver enough energy to completely phase them out (Zou et al. (2006), Demirbas (2009)). Every day, it becomes harder to exhaust the petroleum reservoirs, and there is a continuous search for new and better techniques to perform this in a better, safer and more efficient manner. One part of these improvements is to develop ever better simulations that can give fast and continuous prediction of the flow that occurs during petroleum recovery. With the ability to run more and larger simulations to predict outcomes of retrieving the resources, companies can more accurately take safety precautions and make decisions that are more economical. This may change which oil field to pursue, avoid blowouts and choose optimal extraction strategies. Mathematical models of oil reservoirs are strongly nonlinear with large variations in the coefficients and constitute a highly interesting problem to study. The same equations and methods are also adaptable

to other fields of study besides petroleum. They can be applied to simulations of storage of CO_2, movement of ground water, geothermal energy, flow in fuel cells and much more (Seternes (2015), Bastian et al. (2013), Jain et al. (2008)).

Modern numerical analysis and scientific computation are credited to start with the paper by Von Neumann and Goldstine (1947). By the introduction of electronic computers in the mid 20th century, the ability to simulate fluid flow increased drastically. The increase of processing power allowed for simulations on larger scales and with more complexity. For decades the growth of processing power has closely followed Moore's law of regularly doubling the transistor density (Moore (1965), Mollick (2006)). This has not diminished the importance of improving numerical techniques. By continuously improving the numerical methods, there has been a tremendous gain in speed and accuracy in addition to the improvement of pure processing power.

Testing a new method or algorithm usually takes considerably long time and effort. Often the researcher needs extensive knowledge about the details about the tools needed for implementing and testing new ideas. Within reservoir simulation, SINTEF has taken steps towards a more fluent process from the forming of an idea to actually be able to produce test results of the concept. With the Matlab Reservoir Simulation Toolbox, MRST, this process has been reduced drastically (Lie (2016)). MRST provides a wide range of resources towards research of new and existing methods within reservoir simulation. We are thus relieved of much of what can be considered as the gritty details of programming, and can instead focus on how to get the concept to work when developing new solution methods.

This thesis is a continuation of my project thesis on the same subject (Toft

(2017)). In the project, I investigated the Full Approximation Scheme, FAS, for a dead-oil immiscible and compressible two-phase subsea model. A simple implementation was conducted with the MRST framework, and tested against an industry standard Newton's method.

Parts of this work was presented at the Norwegian Informatics Conference, NIK (Toft et al. (2018)).

1.1 Motivation

Physical phenomena are often modeled by equations that relate several partial derivatives of physical quantities. The typical way to solve such equations is to discretize and approximate them by equations involving a finite number of unknowns N. This gives rise to large and sparse matrix problems. There are two strategies to solve such systems. A direct method such as Gauss elimination, requires $O(N^3)$ arithmetic operations and solves the system exactly (Gauss (1809),Süli and Mayers (2003)). Much research has been invested to reduce the cost of solving linear systems. Iterative methods such as Jacobi or Gauss-Seidel drastically reduce the computation cost to $O(N^2)$ (Saad (2003)). These methods find an approximation of the solution of partial differentiation equations, PDE's, by using an initial guess to generate a sequence of improving solution approximations. The development of the Krylov subspace methods gave rise to many of the most important iterative methods used today (Trefethen and Bau (1997)). These methods apply projections for solving linear systems. Among these are the Generalized Minimal Residual, GMRES (Saad and Schultz (1986)) and Orthogonal Minimization, ORTHOMIN (Vinsome (1976)). For sparse sys-

tems these methods can have a computation cost of $O(N)$. Krylov methods use only the information given directly from the linear system $Ax = b$. GMRES is also frequently used as preconditioner in solvers for nonsymmetrical sparse systems to enhance the solution efficiency for the main solver. By preconditioning, we transform the original linear problem into a similar system with the same solution. This transformed system is likely to be easier to solve with another iterative solver. A preconditioner well suited for systems derived from a reservoir simulation has been shown to be the constrained-pressure-residual preconditioning method, or CPR method (Wallis (1983), Wallis et al. (1985)).

One class of numerical methods that has become quite popular is the multigrid methods (Trottenberg et al. (2001), Saad (2003)). These methods take another approach than the classic iterative methods and Krylov methods by exploiting the information given by the original problem, e.g the PDE, directly instead. With this approach it is possible to take advantage of properties of the problem not directly available from A and b. Multigrid methods were initially developed to specifically solve elliptic PDE's where there is a strong relationship between eigenfunctions of the iterative matrix and the grid. This relationship is exploited by applying multiple grid sizes. Multigrid methods have the ability to, at least theoretically, maintain the convergence rate independently of the grid size (Trottenberg et al. (2001)). Elliptic problems tend to be linear and strongly connected systems. These systems are difficult to solve for many iterative solvers. Multigrid methods are regarded as one of the fastest methods available, and much research has been devoted to this class of numerical methods. A generalization of the multigrid methods is the algebraic multigrid, AMG, developed to solve matrix equations using the principles of standard multi-

grid methods (Ruge and Stüben (1987)). AMG provides a very robust solution method and has an practical advantage since it can be applied directly to both structured and unstructured grids. An enhancement of the AMG was made by utilizing aggregation of the variables making it more robust than the classical AMG. This method, termed Aggregation-based Algebraic Multigrid Method, AGMG, has become an promising and robust black-box solver (Notay (2010)).

A property that has given multigrid methods much attention is that they often have excellent parallel properties. This allows the usage of modern massively parallel computer architectures, to accelerate the computations and thus speed up the simulation. Especially the usage of general-purpose graphical processing units, generally referred as GPGPUs, have received much attention. An example of this is AGMG with GPU acceleration, which has proven to be a robust, efficient and scalable solver (Gandham et al. (2014)).

Another approach to the algebraic multigrid are the variants of geometric multigrid. While algerbraic multigrid perform a linearization from the finest grid to each grid level, the geometric methods conserves the nonlinearity of the system on each grid. Herein, FAS is a nonlinear multigrid method which considers the nonlinearity of the system on each grid level (Molenaar (1995), Henson (2003), Trottenberg et al. (2001)). Within the research of multigrid methods, FAS has been reintroduced to applications of reservoir simulation (Christensen et al. (2016), Christensen (2016)). FAS is a part of the family of nonlinear multigrid aimed to perform the linearization locally instead of globally. This is well suited for solving the mixed hyperbolic-elliptic equations arising from reservoir simulations. In reservoir simulations AMG is primarily used as a preconditioner, see e.g. Bui et al. (2016). For the linear system in AMG, a preconditioned

GMRES it is frequently used as the solver. The state-of-the-art approach is to combine the CPR with AMG as a preconditioner for the elliptic part of the linearized system. In this study we want to investigate the efficiency of FAS for subsurface porous media flow with CPR-AMG to accelerate the solution process.

1.2 Problem Formulation

As stated in the previous section, there is a continuous need for improving the numerical methods for reservoir simulations. This thesis further investigates applications of the FAS method for the black-oil equations, modeling immiscible three-phase flow in a subsurface porous media. The investigation will be done by implementing the FAS method and comparing the performance with the conventional method that applies a global linearization to the system of equations. The reasoning for using methods such as FAS, is that the conservation laws used for modeling flow in porous media are nonlinear. Usually, a global linearization method such as Newton's method is used to get an approximate solution. The FAS method, on the other hand, is applied directly to the nonlinear equations and has the ability to become more efficient than traditional methods (Christensen et al. (2016)). The goal is to implement the FAS method with the object-oriented automatic derivation, AD-OO, framework of MRST for a fully implicit and compressible three-phase flow model in a porous media.

1.3 Objectives

The main objectives of this thesis are:

1. Investigate the mathematical black-oil model for an immiscible and compressible three-phase flow in a subsurface porous media.

2. Investigate the properties of the multigrid FAS method as a nonlinear solver.

3. Become familiar with the object-oriented functionality of MRST.

4. Implement the FAS method in the AD-OO framework of MRST for the chosen mathematical model.

5. Compare the performance of FAS with standard Newton's method.

1.4 Limitations

This study further investigates the FAS method. It consists of a few simplifications of the three-phase model. Besides the assumptions presented for the model, the effects of temperature and capillary pressure are neglected. These limitations are common for modeling of reservoirs in the Nordic sea. The main focus will be on the conceptual correctness and relative efficiency compared to the standard Newton's method.

1.5 Approach

FAS is implemented in MATLAB. This is a scientific programming language that is particular beneficial in mathematical applications. MRST is used as a foundation for the implementation. The accompanying user guide (Lie (2016)) provides a detailed description of the basics of MRST and the implementation of reservoir simulators. By applying MRST, we are given the freedom of applying already existing code that provides many of the basic tools and functions needed in any numerical method. Functions from MRST will be used both as a framework around, and in the core of the FAS method. In particular, a standard Newton's method solver will be used both as a core solver and as a source of benchmark comparison.

1.6 Structure of the Thesis

The rest of the thesis is structured as follows: Chapter 2 provides the background of the mathematical model, a presentation of the basics of numerical methods, and finally a presentation of the theory of the FAS method. Chapter 3 presents the implementation details and an overview of MRST. Chapter 4 presents numerical experiments and comparisons with the standard Newton's method. Chapter 5 finalizes the thesis with a summary and conclusion of the results, and brief discussion of further work.

Chapter 2

Theoretical Background

An petroleum reservoir consists of multiple components, representing individual species, e.g. methane, propane, and decane. These components may form up to three different phases within the pores of the rock formation: aqueous, gaseous and oleic. It is common to bundle heavy hydrocarbons occurring as a liquid phase on the surface of an oil component, while the other components occurring as gaseous is bundled as a gas component. In a reservoir both of these components may occur as both a liquid and gas phase (Peaceman (1977)).

To simulate flow of the three-phase system when injecting water into a reservoir, we have to build a mathematical model. For each of the physical properties of the reservoir we will present and include a mathematical description. Conservation of mass and Darcy's law describe the behaviour of the flow and the interaction between the three phases. In addition, one needs to account for the rock's ability to store and transmit fluids as well as how the rocks and fluids compress and change with pressure.

After completing our mathematical model, we will explain the numerical

method. This will be done in several steps, starting with the basics of the finite-volume discretization. We then go through the building blocks forming a multigrid method. This includes restriction and interpolation, smoothing of the residual, and how this affects and benefits the simulation. Finally, we present the procedure of the FAS method.

2.1 Mathematical Model

2.1.1 Law of Mass Conservation

One of the most central principles of the physical world is the conservation of mass, momentum and energy. A proper starting point to develop our mathematical model is therefore the law of mass conservation. We will here follow along the lines of Aziz and Settari (1979) and Trangenstein and Bell (1989), who also provide a more detailed discussion of this topic. Observing that the reservoir is a closed volume, with only the wells as sources and sinks, we can describe the conservation of mass for a single fluid phase within this volume. We start by denoting the reservoir domain by $\Omega \in \mathbb{R}^3$, and the boundary as $\partial\Omega$. Conservation of mass can then be written as

$$\frac{\partial m}{\partial t} = \int_\Omega \frac{\partial \rho}{\partial t} dV = -\oint_{\partial\Omega} \rho \boldsymbol{\psi} \cdot \mathbf{n} dA + \int_\Omega q dV. \tag{2.1}$$

Here, m represents the mass of the fluid in the control volume, $\boldsymbol{\psi}$ is the velocity of the fluid or flux, giving the flow rate through a infinitesimal volume, ρ its density, ϕ the rock porosity and $\mathbf{n}$ is a unit normal vector pointing outward from the surface $\partial\Omega$. The term q represents sources or sinks, which in our case will

be injection and production wells, respectively. Here the fluid velocity is not a intrinsic velocity of the fluid, but rather the flux density through a infinitesimal volume.

Porosity

A porous medium is a material that contains small pores or voids which allows fluids to flow through. By introducing a porosity function ϕ, we take into account that only a fraction of the volume is open for fluid flow. Porosity may be either homogeneous or vary throughout the domain. In addition, the porous media is not a rigid material. It is thereby affected by pressure, causing it to expand and contract. The porosity can be written as a function of the pressure p in the rock, $\phi = \phi\left(\mathbf{x}, p\right)$, where $\mathbf{x}$ is the spatial location.

Fluid compressibility

The above equation for mass conservation is currently in its basic form, and we will now investigate the terms representing the compressibility of fluids. In a subsurface reservoir we experience a high pressure and will thereby observe an effect on how the fluids flow due to their compressibility. The compressibility model for fluids is defined by

$$c_f = -\frac{1}{V}\frac{\partial V}{\partial p} = \frac{1}{\rho}\frac{\partial \rho}{\partial p}. \tag{2.2}$$

By integration we find the expression for the fluids density

$$\rho = \rho_0 e^{\left[c_f (p - p_0)\right]}, \tag{2.3}$$

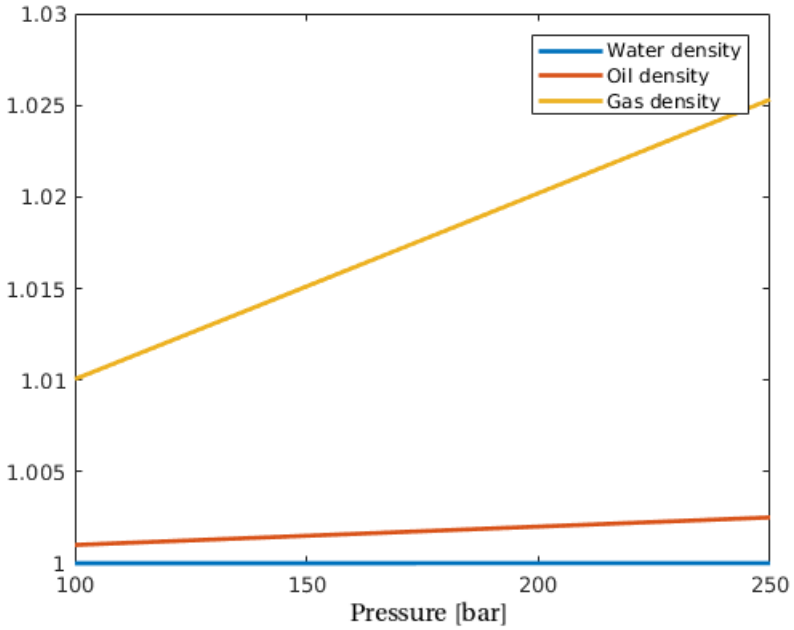


Figure 2.1: Illustration of the compressibility of the three phases water, oil and gas as. As we see, the changes in density as a function of the pressure, is different for each phase.

where ρ_0 is the density at the reference pressure p_0. A slightly compressible fluid has a small, often constant compressibility in the range 10^{-6} to 10^{-5} psi^{-1}. In a reservoir, it is normal that water, dead oil and under saturated oil behave as slightly compressible fluids. More compressible fluids usually have a compressibility in the range 10^{-4} to 10^{-3} psi^{-1}. In Figure 2.1 we see the compressibility that is used for the three phases used for this thesis, water oil and gas.

The high compressibility of gas, especially compared to the small compressibility of oil and water, leads to considerable volume changes at pressures normal in reservoirs. The black-oil model incorporates these volume changes by

relating the reservoir volumes of each phase to the reference volume at standard surface conditions (Trangenstein and Bell (1989)). By assuming the fluids are under constant temperature and thermodynamic equilibrium throughout the reservoir, the formation volume factors may be expressed as:

$$B_\alpha = \frac{[V_\alpha]_R}{[V_\alpha]_S}, \qquad \alpha \in \{w, o, g\}. \tag{2.4}$$

Here, $[V_\alpha]_R$ and $[V_\alpha]_S$ are the phase volume under reservoir and standard conditions, respectively. In the literature some authors prefer to work with reciprocal factors, i.e. $b = 1/B_\alpha$. The densities of the three phases at reservoir conditions are then related to densities at standard conditions:

$$\rho_{\alpha R} = \frac{1}{B_\alpha}\rho_{\alpha S}. \tag{2.5}$$

Rock compressibility

For non-rigid rock, the rock compressibility can be defined as

$$c_r = \frac{1}{\phi}\frac{\partial \phi}{\partial p} = \frac{\partial \ln(\phi)}{\partial p}, \tag{2.6}$$

where p is the pressure in the reservoir. We will apply rock with a constant compressibility and integrate (2.6) to find the pressure dependent porosity

$$\phi(p) = \phi_0 e^{[c_r(p-p_0)]}, \tag{2.7}$$

where ϕ_0 is the porosity at the reference pressure p_0.

To relate flow through a surface of the volume to the flow within the volume

we need to apply the divergence theorem, also known as Gauss' theorem (Gauss (1813))

$$\int_V (\nabla F)\, dV = \oint_{\partial V} F \cdot \mathbf{n} dS. \tag{2.8}$$

By applying this theorem to (2.1) and assuming sufficient smoothness, we can write mass conservation in differential form

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

where the $\mathbf{f} = \rho \boldsymbol{\psi}$ is a vector-flux function.

2.1.2 Darcy's law

As a result of the fluid flowing through a porous medium, there will be some resistance to the flow. This is caused by physical forces such as shear stress when some parts of the fluid are affected by friction causing different speeds within the fluid. This resistance to the flow in a porous medium is described by Darcy's Law

$$\boldsymbol{\psi} = -\frac{\mathbf{K}}{\mu}(\nabla p - \rho g \nabla z) \tag{2.10}$$

where $\boldsymbol{\psi}$ is flux, $\mathbf{K}$ the permeability tensor, μ the viscosity, p the pressure and g the gravitational acceleration (Darcy (1856), Whitaker (1986)).

Permeability

Permeability is the porous medium's ability to let fluids pass through it. This capacity if often referred to as absolute permeability. The permeability generally depends on location, and can vary with flow direction. It is represented as

a positive definite symmetric tensor

$$\mathbf{K} = \begin{pmatrix} K_{xx} & k_{xy} & K_{xz} \\ K_{yx} & K_{yy} & K_{yz} \\ K_{zx} & K_{zy} & k_{zz} \end{pmatrix}, \tag{2.11}$$

but is commonly used as a diagonal tensor for modelling purposes:

$$\mathbf{K} = \begin{pmatrix} K_{xx} & 0 & 0 \\ 0 & K_{yy} & 0 \\ 0 & 0 & K_{zz} \end{pmatrix}. \tag{2.12}$$

Here, the diagonal terms represent how the flow rate in one axial direction depends on the pressure drop in the same direction. A full tensor is needed to model local flow in directions at an angle to the coordinate axes (Lie (2016)).

Viscosity

The viscosity of a fluid is a measure of its resistance to deformation by shearing forces. Later we will look into how the difference between the viscosity of several phases affects the behaviour of the displacement front.

With Equations (2.1) and (2.10), we have a complete mathematical model for a single-phase flow in a reservoir. The next step is to account for the interaction between multiple fluid phases.

2.1.3 Three-phase Flow Model

When more than one fluid is present in a reservoir, we need to consider the interaction between the phases. In three-phase systems, one of the fluids is more strongly attracted by the solid than the others. This phase is known as the wetting phase and are indicated with the subscript w. In general, water is the wetting fluid relative to oil and gas. There exists exceptions based on the composition of the hydrocarbons. The nonwetting phases are indicated with o and g for oil and gas, respectively. For our simulations we assume that the fluids do not dissolve into each other.

In essence, we are trying to find the pressure and saturation for the three phases for each time step. The saturation S_α with $\alpha \in \{w, o, g\}$, is defined as the fraction of the pore volume that is filled with each fluid. The saturation constraint is given as

$$S_w + S_o + S_g = 1. \tag{2.13}$$

We can then rewrite the law of mass conservation (2.1):

$$\frac{\partial \left(S_\alpha \rho_\alpha(p)\phi(p)\right)}{\partial t} + \nabla \cdot \mathbf{f}_\alpha \left(S_\alpha, p\right) = q_\alpha. \tag{2.14}$$

To account for the interaction between the three phases, we need to include a relative permeability function $k_{r\alpha}(S_\alpha)$ into Darcy's law. The function accounts for the friction between the phases and is thereby a part of the description of the mobility of the fluids. The presence of another fluid causes more friction, and the corresponding shear forces lower the mobility of the fluids. The mobility therefore depends on saturation. Relative permeability measures the tension between the phases and how they relate to the pores of the

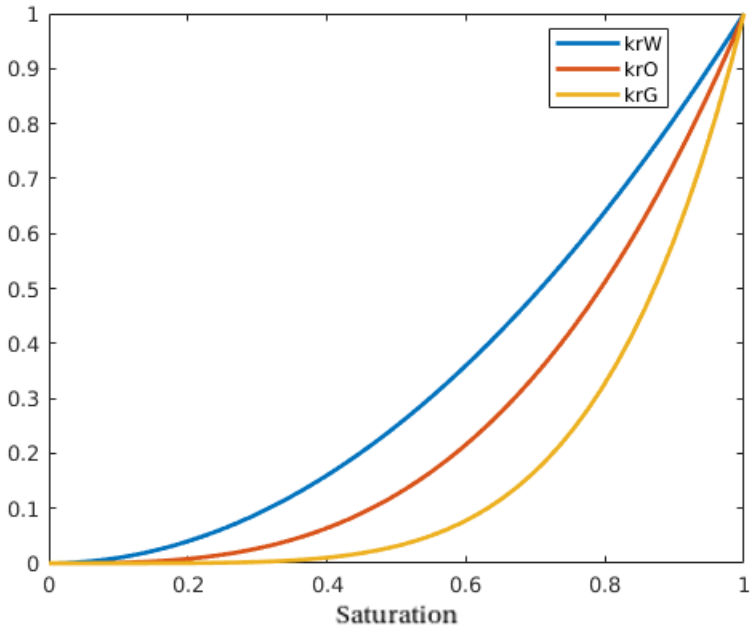


Figure 2.2: Relative permeability curves for the three phases water, oil and gas.

rock (Muskat and Wyckoff (1937)). It is common to represent relative permeability with a tabulated or simplified analytic relationships. In this thesis, we use a simple power-law relative permeability function also called a Corey model

$$k_{r\alpha}(S_\alpha) = S_\alpha^{n_\alpha}, \tag{2.15}$$

with the exponents $n_\alpha \geq 1$. This is a simplification of the model presented by Brookes and Corey (1964) which also used capillary pressure. In (2.15) we neglect the residual saturation that is often used to normalize the saturation. The residual saturation represent the volume fraction of phases trapped in the pores. This can be water enclosed in the pores during the formation of the rock. An example of relative permeability curves is shown in Figure 2.2.

Equation (2.10) then becomes

$$\boldsymbol{\psi_\alpha} = -\lambda_\alpha \mathbf{K}\left(\nabla p_\alpha - \rho_\alpha g \nabla z\right), \tag{2.16}$$

where the mobility function λ_α is the ratio of relative permeability $k_{r\alpha}$ and viscosity μ_α,

$$\lambda_\alpha = \frac{k_{r\alpha}(S_\alpha)}{\mu_\alpha}. \tag{2.17}$$

The densities $\rho_\alpha = \rho_\alpha\left(p\right)$ of the fluids are the same as in (2.3).

By inserting Equation (2.16) and (2.17) into (2.14), we can rewrite the law of mass conservation to account for a three-phase flow in a subsurface porous medium:

$$\frac{\partial\left(S_\alpha \rho_\alpha \phi\right)}{\partial t} = q_\alpha - \nabla \cdot \left(-\mathbf{K}\frac{k_{r\alpha}(S_\alpha)}{\mu_\alpha}\left(\nabla p_\alpha - \rho_\alpha g \nabla z\right)\right). \tag{2.18}$$

2.1.4 Initial and Boundary Conditions

To close the coupled conservation equations for the three phases, we require proper boundary and initial conditions. For a reservoir where the petroleum has been trapped for millions of years, it is natural to use a no-flow boundary conditions. We let the boundary $\partial\Omega = \partial\Omega_+ \cup \partial\Omega_- \cup \partial\Omega_0$ be the boundary of the domain, where $\partial\Omega_+, \partial\Omega_-, \partial\Omega_0$ is inflow, outflow, and no flow boundary, respectively. No flow is modeled by providing a homogeneous Neumann condition for the external boundary,

$$\boldsymbol{\varphi}_\alpha \times \mathbf{n} = 0 \quad \forall\, \mathbf{x} \in \partial\Omega_0. \tag{2.19}$$

With the presents of wells, we also need to pose condition for the fluxes of injection and production wells.For the injection well we set a inhomogeneous Neumann condition

$$\boldsymbol{\psi}_w \times \mathbf{n} = q_w \quad \forall\, \mathbf{x} \in \Omega_-, \tag{2.20}$$

where q_w is injection rate and $\partial\Omega_-$ is influx boundary. For the production wells, it is common to account for the boundary condition with a static pressure. In this thesis this is modeled in terms of a Dirichlet condition in the form

$$p(\mathbf{x}, t) = p_0 \quad \forall\, \mathbf{x} \in \Omega_+, \tag{2.21}$$

where p_0 is the constant external pressure and $\partial\Omega_+$ is the boundary for the production well.

Since we have compressible fluids, we need to specify an initial pressure distribution. This pressure is set to be hydrostatic, and can then be given by the ordinary differential equation

$$\frac{dp_\alpha}{dz} = \rho_\alpha g, \quad p_\alpha(z_0) = p_0. \tag{2.22}$$

Finally we need to specify a initial saturation for the different phases such that

$$S_{w0} + S_{o0} + S_{g0} = 1. \tag{2.23}$$

Combined, Equation (2.13) and (2.18), the boundary conditions (2.19 -2.21) and initial conditions (2.22 - 2.23) form a complete model for a three-phase compressible flow in a sub-surface reservoir simulation.

2.2 Numerical Methods

We now have established the mathematical model of the system. For our numerical approach we will first look into the spatial and then the temporal discretization. There are presently three primary approaches used to discretize the flow problem in space:

- Finite-difference methods (FDM),
- Finite-volume methods (FVM),
- Finite-element methods (FEM).

See e.g., Schäfer (2006) for more details of the discretization schemes. FDM is the simplest of the three, whereas FEM and FVM are most used in practice. FEM is mainly used in structural computations such as stress and elasticity. The FVM is very well suited for flow simulation as it is geared towards solution of conservation laws, and is thereby the discretization method we will apply as a foundation for our numerical method.

There are two main classes of temporal discretization, implicit and explicit methods. The simplest and most intuitive method is the forward Euler method (Euler (1768)), in which one takes the current phase velocities and compute directly where the phases are moving during the next time step. A simple illustration of this can be drawn from (2.14):

$$S_\alpha^{n+1} = S_\alpha^n + \frac{\Delta t}{\rho_\alpha \phi}\left(q_\alpha^n + \nabla \cdot \mathbf{f}_\alpha^n\right), \tag{2.24}$$

where n is the time step. Here, approximations to $\Delta t q_\alpha^n$ and $\Delta t \mathbf{f}_\alpha^n$ can be computed directly and added to the saturation from the previous time step. This

leads to a simple and cost effective scheme for each time step. The downside is the requirement of small time steps to obtain stability. Another approach is to apply implicit methods. These methods are much more costly as they compute sequences of improving approximations. There are several implicit methods, where one of the simplest and most used is the backward Euler method. This can similarly be illustrated by

$$S_\alpha^{n+1} = S_\alpha^n + \frac{\Delta t}{\rho_\alpha \phi}\left(q_\alpha^{n+1} + \nabla \cdot \mathbf{f}_\alpha^{n+1}\right). \tag{2.25}$$

In this case, q_α^{n+1} and $\mathbf{f}_\alpha^{n+1}$ are unknown and we need to apply an iterative solver such as Newton's method to get an approximation. In other words, the implicit method produces a system of nonlinear equations that we have to solve for each time step. We will apply the backward Euler due to its simplicity and stability.

2.2.1 Finite Volume Method

The FVM uses control volumes, or cells, in which the values are averaged. This means that the domain Ω is divided into several smaller sub-domains Ω_i. A example of this subdivision is shown in Figure 2.3. These sub-domains are what FVM denotes as control volumes or cells, as illustrated in Figure 2.4.

The geometrical shape of the control volume can be specified for each application, and can vary from simple cubes to more complex shapes such as a polyhedral as in Figure 2.4. The cell average for a 2D rectangular cell $F_i(t)$ is defined as

$$F_i(t) = \frac{1}{|\Omega_i|}\int_{\Omega_i} F(\mathbf{x}, t)\, d\mathbf{x} \tag{2.26}$$

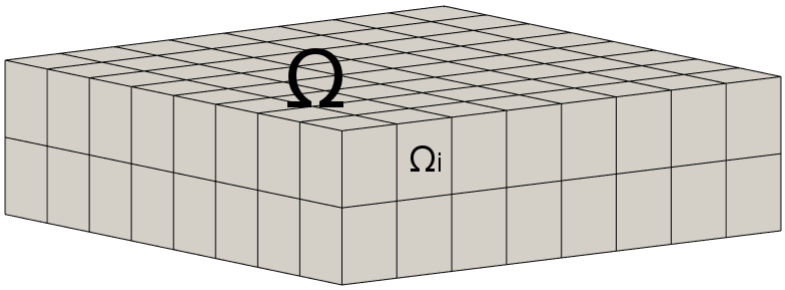


Figure 2.3: A FV division of Ω into sub-domains Ω_i for a uniform grid.

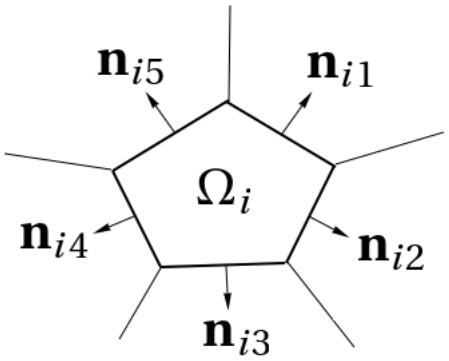


Figure 2.4: A FV control volume

With the cell average as a basis for the approximation of the values within each cell, we now turn to the discretization of the governing equations.

2.2.2 Discretization

By applying the discretization method as described above, we can discretize the governing equations in two steps. First, we perform the spatial discretization, where we utilize the finite-volume method with cell centering. We use N dis-

tinct grid cells and denote the ith cell by $\Omega_i, i \in C = \{1,...,N\}$ with the surface $\partial\Omega_i$ of each cell.

For the spatial discretization we need to approximate the flux between each cell. We start by considering Equation 2.1 for an arbitrary control volume $\Omega_i \in \Omega$

$$\int_{\Omega_i} \frac{\partial \rho_\alpha}{\partial t} dV = -\oint_{\partial\Omega_i} \rho_\alpha \boldsymbol{\psi_\alpha} \cdot \mathbf{n} dA + \int_{\Omega_i} q_\alpha dV. \tag{2.27}$$

This relation is the basis of the finite volume discretization. To simplify the equation, the volume of the right-hand side integral is approximated with the porosity and saturation. In addition as long as the entire non-zero source function is completely embedded within one grid block, the value of the total source strength

$$Q_{\alpha,i} = \int_{\Omega_i} q_{\alpha,i} dV \tag{2.28}$$

is independent of the actual distribution of the source within the block. With this we write 2.27 as

$$\frac{\partial \left(\phi \rho_\alpha S_\alpha\right)}{\partial t} dV = -\oint_{\partial\Omega_i} \rho_\alpha \boldsymbol{\psi_\alpha} \cdot \mathbf{n} dA + Q_{\alpha,i}. \tag{2.29}$$

Here, where we write the contour integral as a sum of the integrals over all cell faces j

$$\frac{\partial \left(\phi \rho_\alpha S_\alpha\right)}{\partial t} dV = -\sum_{j \in \mathcal{N}_{(i)}} \oint_{\partial\Omega_{ij}} \rho_\alpha \boldsymbol{\psi_\alpha} \cdot \mathbf{n} dA + Q_{\alpha,i}, \tag{2.30}$$

where $\mathcal{N}_{(i)}$ is the set of neighboring cells of cell i. For the next step we use 2.26 to let the value of $\boldsymbol{\psi_\alpha}$ be approximated by an cell average $\bar{\boldsymbol{\psi}}_{\boldsymbol{\alpha}}$. In addition we

denote A_{ij} the surface between two neighbouring cells i and j.

$$\frac{\partial\left(\phi\rho_\alpha S_\alpha\right)}{\partial t}dV = -\sum_{j\in\mathcal{N}_{(i)}} \rho_\alpha \boldsymbol{\bar{\psi}_{\alpha ij}} A_{ij} + Q_{\alpha,i}, \tag{2.31}$$

By rewriting the terms of 2.31 the spatial discretization of the conservation law then becomes

$$\frac{\partial\left(V_p\rho_\alpha S_{\alpha,i}\right)}{\partial t} = Q_{\alpha,i} - \sum_{j\in\mathcal{N}(i)} \rho_\alpha T_{ij}\lambda_{\alpha,ij}\left(\Delta p - \rho_\alpha g\Delta z\right)_{ij}, \quad i\in C, j\in\mathcal{N}_{(i)}, \tag{2.32}$$

with the pressure difference

$$\Delta p_{ij} = p_i - p_j \tag{2.33}$$

and vertical spatial coordinate difference

$$\Delta z_{ij} = z_i - z_j. \tag{2.34}$$

Instead of explicitly working with the porosity, the equation has been multiplied with the pore volume, $V_p = \phi V$. The movement of the flow is represented by the mobility $\lambda_{\alpha,ij}$ and transmissibility T_{ij}. The latter measures the fluid's efficiency in flowing and are a combination of geometric quantities. For a uniform Cartesian grid we express the transmissibility as:

$$T_{ij} = \frac{A_{ij}k_{ij}}{\Delta h}, \quad i\in C, j\in\mathcal{N}_{(i)}, \tag{2.35}$$

where Δh denotes the mesh width, and the permeability between the cells is

approximated by the harmonic mean

$$k_{ij} = \frac{2k_i k_j}{k_i + k_j}. \tag{2.36}$$

Other discretizatons of the transmissibuility is used in the literature, although this is the most commonly used (Toronyi and Ali (1974)).

The solution of fluid flow problems can be strongly influenced by the orientation of the underlying grid (Brand et al. (1991)). The mobility may mathematically be midpoint weighted between two cells, but physically this is incorrect. This may lead to different solutions depending on whether the line drawn from the source to the sink are parallel or diagonal to the underlying grid. This is called the grid orientation effect and is a problem in numerical simulations. To ensure stability, the flux over the interface between cells are approximated with a upstream mobility weighting. In reservoir simulation, a single-point upstream is a commonly used scheme (Aziz and Settari (1979),Chen (2007)) where the flow direction determines which side the approximation made:

$$\lambda_{ij} = \begin{cases} \lambda_j & \text{if } \left(\Delta p - \rho_\alpha g \Delta z\right)_{ij} < 0 \\ \lambda_i & \text{otherwise.} \end{cases} \tag{2.37}$$

As discussed above, for the temporal discretization we apply the backward Euler method and Equation (2.32) becomes

$$S_{\alpha,i}^{n+1} = S_{\alpha,i}^{n} + \frac{\Delta t}{V_p \rho_\alpha}\left(Q_{\alpha,i}^{n+1} + f_{\alpha,i}^{n+1}\right), \tag{2.38}$$

with the residuals

$$r_{\alpha,i}(S_\alpha) = \frac{\Delta t}{V_p \rho_\alpha}\left(s_{\alpha,i}^{n+1} + f_{\alpha,i}^{n+1}\right) + S_{\alpha,i}^{n} - S_{\alpha,i}^{n+1}, \tag{2.39}$$

$$r_{v,i} = \left(\sum_\alpha S_\alpha\right)^{n+1} - 1. \tag{2.40}$$

When we solve the system, we rewrite (2.23) and use

$$S_o = 1 - S_w - S_g \tag{2.41}$$

to eliminate a variable.

Finally, we have completed the discretization of the mass conservation equations. Next, we need to simultaneously solve for the primary unknowns S_w, S_g and p. With the discretization and boundary conditions (2.19 - 2.21) we have a well posed problem that admits a unique solution.

2.3 Solvers for Reservoir Simulation

Now that we have established the mathematical model of the three-phase flow and discretized it, we turn our attention towards examining the standard methods used to solve our system of nonlinear equations. These methods will also form the subroutines of our main method as we will discuss later. Because of the implicit nature of the discretization we have chosen, we need an iterative solver. Our discussion will thereby start with Newton's method.

2.3.1 Newton's method

As with most any multigrid methods, at the core of FAS we need a fast iterative solver to solve the system of nonlinear equations. Newton's method and its variations have in many ways become the gold standard for many applications (Newton (1687), Süli and Mayers (2003)). It has been proven to be a robust method and have a quadratic convergence if the initial guess is sufficiently close to the solution. We will use the Newton-Raphson method, which for multiple dimensions can be written as

$$\mathbf{u_{k+1}} = \mathbf{u_k} + \alpha \mathbf{p_k} = \mathbf{u_k} - J\,(\nabla \mathbf{f_k})^{-1}\,\mathbf{f_k}. \tag{2.42}$$

Here, the step length α by default is set to one and the search direction $\mathbf{p_k}$ is set to the function evaluated at the previous step, multiplied with the inverse of the Jacobian. The Jacobian matrix is defines as

$$J_{ij} = \frac{\partial f_i}{\partial u_j}. \tag{2.43}$$

The gradient is found by automatic differentiation which we will look at in the discussion of the implementation.

2.3.2 Constrained pressure Residual (CPR)

The linear systems of equations arising from the linearization by use of Newton's method can be denoted by

$$A\mathbf{u} = \mathbf{f}, \quad A \in \mathbb{R}^{n\times n}. \tag{2.44}$$

The matrix corresponding to a simultaneous fully implicit discretization is often highly non-symmetric and not necessarily diagonally dominant. These two features are central for the efficiency of many solution methods. In addition there is multiple unknowns per grid cell. The computational effort required to solve the linear systems is often substantially greater than any other part of a simulation (Wallis (1985)). A common approach to accelerate the convergence rate is to precondition to system (Saad (2003)). As mentioned in the introduction, this is performed by replacing the original problem (2.44) with an easier problem with the same solution,

$$MA\mathbf{u} = M\mathbf{f}. \tag{2.45}$$

Here, M is called the preconditioning matrix or preconditioner. This is a left preconditioning, while there are other variants such as right preconditioning reading as $AM\mathbf{u} = \mathbf{f}M$. In the literature 2.45 is often written as $P^{-1}A = P^{-1}\mathbf{f}$. Since the matrix P is rearly formed, we denote $M = P^{-1}$ and name M the preconditioner. The ideal is to find an M such that it is close to the inverse of A or at least converting the system matrix such that the product MA is diagonally dominant. A linear problem with a strongly diagonal dominant matrix is easier to solve as the eigenvalues is close to the diagonal elements.

The global linearization arising from the discretization of 2.44 and use of Newton-Raphson can be written as

$$J\mathbf{u} = \begin{bmatrix} J_{pp} & J_{ps} \\ J_{sp} & J_{ss} \end{bmatrix} \begin{bmatrix} \mathbf{u_p} \\ \mathbf{u_s} \end{bmatrix} = \mathbf{r} = \begin{bmatrix} \mathbf{r_p} \\ \mathbf{r_s} \end{bmatrix} \tag{2.46}$$

where J_{pp} is the pressure block coefficients, J_{ss} is the saturation block coefficients, and J_{ps} and J_{sp} is the respective coupling coefficients. $\mathbf{u_p}$ and $\mathbf{u_s}$ represent the increments in respectively the pressure and saturation variables. The CPR preconditioner recognize that (2.46) is a mixed hyperbolic-elliptic system type (Wallis (1983)). CPR targets the elliptic part of the system in a separate stage by an predictor-corrector strategy on the pressure (Cusini et al. (2014)). The precondition matrix for the pressure written with Schur complement (Haynsworth (1968), Zhang (2005)) of the matrix block J_{ss}, is on the form

$$M = \begin{bmatrix} I & -Q \\ 0 & I \end{bmatrix}. \tag{2.47}$$

By multiplying the original system (2.46) with the preconditioner M we obtain

$$\begin{bmatrix} J_{pp} - QJ_{sp} & J_{ps} - QJ_{ss} \\ J_{sp} & J_{ss} \end{bmatrix} \begin{bmatrix} \mathbf{u_p} \\ \mathbf{u_s} \end{bmatrix} = \mathbf{r} = \begin{bmatrix} \mathbf{r_p} - Q\mathbf{r_s} \\ \mathbf{r_s} \end{bmatrix} \tag{2.48}$$

It is assumed that

$$Q = J_{ps}J_{ss}^{-1} \approx QI \tag{2.49}$$

where the matrices J_{ps} and J_{ss} are replaced with vectors where the elements are the sums of each column. This simplifies the Schur complement procedure and implies that J_{ps} is close to zero such that

$$J_{pp}^{*}\mathbf{u_p} = \mathbf{r_p^*} \tag{2.50}$$

where $J_{pp}^{*} = J_{pp} - QJ_{sp}$ and $\mathbf{r_p^*} = \mathbf{r_p} - Q\mathbf{r_s}$. Alternatively the pressure matrix can

be derived by

$$J^*_{pp} = C^T M J C, \quad C^T = [I \quad 0] \tag{2.51}$$

The effectiveness of CPR methods relies heavily on the quality of the pressure matrix J^*_{pp} (Cusini et al. (2014)). The pressure Equation (2.50) forms an approximation of the elliptic part of the discrete system and is considered separately from the remaining hyperbolic part. With the preconditioning of the elliptic part, the CPR method improves the convergence rate of the full linear system (Wallis et al. (1985)). CPR is commonly a part of a two-step preconditioning, where AMG is used to approximate the pressure solution of 2.50. In the second stage, another preconditioner, e.g. the ILU(0) is applied to the full system 2.46. The resulting preconditioned linear system is then solved by GEMRES, which will be further explained later in the thesis

2.3.3 Incomplete LU factorization (ILU(0))

One of the simplest methods to define a preconditioner is to perform an incomplete factorization of the system matrix A (Saad (2003)). This admits a decomposition of the form

$$A = LU - R, \tag{2.52}$$

where L and U are lower and upper triangular matrices, respectively. R is here the residual error of the factorization. The LU is formed with a variant of the Gaussian elimination were some elements are dropped. The dropped elements are nonpositive entries outside the main diagonal, and is transformed into zeros. This gives an incomplete LU factorization. The dropped elements follows

a specified pattern determined in advance. A zero pattern is written such that

$$P \subset \{(i,j) \mid i \neq j;\ 1 \leq i,j \leq n\}, \tag{2.53}$$

where i, j is the matrix indexes of $A^{n\times n}$. The elements dropped in the Gaussian elimination construct the R matrix in 2.52. The simplest variant of the ILU factorization takes the zero pattern to be the exact zero pattern of A, and uses no fill-in. This variant is noted as ILU(0).

2.3.4 General Minimal Residual (GMRES)

GEMRES is a projection method based on Krylov subspaces

$$\mathscr{K}_n = \langle \mathbf{f}, A\mathbf{f}, ..., A^{n-1}\mathbf{f} \rangle \tag{2.54}$$

(Saad and Schultz (1986),Trefethen and Bau (1997)). The method approximates the solution of the system $\mathbf{u}$ by the vector $\mathbf{u_n} \in \mathscr{K}_n$ that minimizes the Euclidean norm of the residual

$$\mathbf{r_n} = \mathbf{f} - A\mathbf{u_n}. \tag{2.55}$$

Since the vectors forming the basis of the Krylov space might be close to linear dependent, Arnoldi iterations is used to find orthogonal vectors $\mathbf{q_1}, \mathbf{q_2}, ...\mathbf{q_n}$ forming a basis for $\mathscr{K}_n$ (Arnoldi (1951)). By writing the basis as an Krylov matrix $Q_n^{m\times n}$we write 2.55 as an least square problem

$$J(y) = \| AQ_n\mathbf{y} - \mathbf{f} \|, \tag{2.56}$$

where $\mathbf{u_n} = Q_n\mathbf{y_n}$ with $\in \mathcal{K}_n$, $\mathbf{y_n} \in \mathbb{R}$ and $J(y)$ the minimum value. The Arnoldi process also produces an upper Hessenberg matrix $\tilde{H}_n^{(n+1)\times n}$, that is, a matrix with zeros below the first subdiagonal (Hessenberg (1942)):

$$AQ_n = Q_{n+1}\tilde{H}_n. \tag{2.57}$$

Since the columns of Q_n is orthogonal, we have

$$\|A\mathbf{u_n} - \mathbf{f}\| = \|\tilde{H}_n\mathbf{y_n} - Q_{n+1}^T\mathbf{f}\| = \|\tilde{H}_n\mathbf{y_n} - \beta\mathbf{e_1}\|, \tag{2.58}$$

where $\mathbf{e_1} = (1,0,0,...,0)^T$ is the first vector in the standard basis of $\mathbb{R}^{n+1}$ and $\beta = \|\mathbf{f} - A\mathbf{u_0}\|$. With this we find x_n by minimizing the Euclidean norm of the residual

$$\mathbf{r_n} = \tilde{H}_n\mathbf{y_n} - \beta\mathbf{e_1}. \tag{2.59}$$

The Eucledean norm, also known as the 2-norm is defined for vectors as

$$\|\mathbf{x}\|_2 = \left(\sum_{i=1}^{n} |\, x_i \,|^2\right)^{\frac{1}{2}}. \tag{2.60}$$

This yields the final step of the GMRES method. When $\mathbf{y_n}$ minimizing $\|\mathbf{r_n}\|$ is found, we set

$$\mathbf{u_n} = Q_n\mathbf{y_n}, \tag{2.61}$$

and update the solution.

2.4 The Full Approximation Scheme (FAS)

Now that we have discussed the standard solvers for reservoir simulations, we will first investigate the multigrid methods in general, and finally the full approximation scheme.

2.4.1 The Fundamentals of Multigrid Methods

Multigrid methods have become a common approach for solving many linear problems of the form

$$A\mathbf{u} = \mathbf{f}. \tag{2.62}$$

The methods have gained popularity as a result of the characteristics of having converging speed independently of the discretization mesh size h, and the number of arithmetic operations are proportional with the number of grid points (Briggs et al. (2000),Trottenberg et al. (2001)). The multigrid idea is based on the two principles of error smoothing and coarse grid correction. The initial steps consists of an iterative relaxation to remove highly oscillatory error modes. This step is often referred as smoothing or application of a smoother. The error is defined as $\mathbf{e} = \mathbf{u} - \mathbf{v}$, where $\mathbf{v}$ is the approximation of the exact solution $\mathbf{u}$. The residual is defined to be

$$\mathbf{r} = \mathbf{f} - A\mathbf{v}. \tag{2.63}$$

By inserting (2.63) into (2.62), we can write the residual equation as

$$A\mathbf{e} = \mathbf{r}. \tag{2.64}$$

The method can be described as a recursive two-layered grid (h, H) process. The error is smoothed on the fine grid and restricted to a coarse grid where the residual equation is solved. The classic iterative methods have a strong smoothing effect on the error of nonlinear systems and are effective in removing the local error modes. Jacobi's method, variants of Gauss-Seidel, and incomplete LU factorization are all iterative solvers for which convergence rates are highly problem dependent. With the errors sufficiently smooth, we can accurately represent them on a coarser grid. Since the coarse problem is much smaller, solving it becomes far less expensive than on the fine grid. The residual (2.63) on the fine grid Ω_h is restricted to the coarse grid Ω_H

$$\mathbf{r_h} = I_h^H \mathbf{r_h}. \tag{2.65}$$

It is then used as the right-hand side for the coarse grid residual equation

$$A_H \mathbf{e_H} = \mathbf{r_H}. \tag{2.66}$$

The solution of this equation then gives the error and is interpolated back to the fine grid. Here it is used to correct the fine-grid approximation

$$\mathbf{v_h} \leftarrow \mathbf{v_h} + I_H^h \mathbf{e_h}. \tag{2.67}$$

By recursively solving the coarse grid problem with this two-grid process, we have established the multigrid algorithm.

2.4.2 Error smoothing

The error occurring in a system can be classified as high and low frequency errors (Briggs et al. (2000),Trottenberg et al. (2001)). The high frequency error is relatively easy to remove with a few iterations, and we say that the system is relaxed or smoothed. A smooth error is one of the governing requirements allowing for an approximation of a system on a coarser grid without any essential loss of information. The low frequency error is not visible on the fine grid, but with the high frequency error removed, the low frequency error becomes visible on the coarser grid. This is illustrated in Figure 2.5.

As we can see from the illustration, the high frequency errors are effectively removed, leaving the low frequency. Further iterations on the fine grid is less effective since the low frequency error is not visible for the smoother on the fine grid. On the coarse grid, the low frequencies are made visible for the smoother, and are effectively removed. The fact that the different frequencies are visible

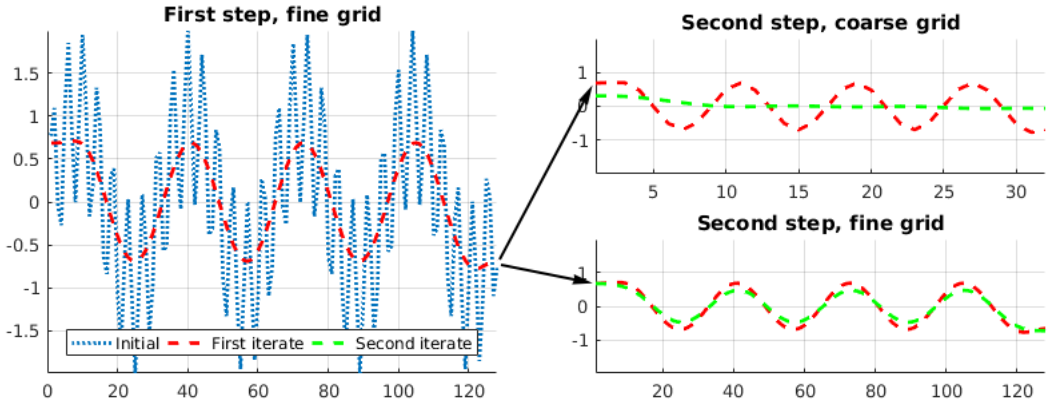


Figure 2.5: The smoothing of error frequencies performs different on fine and coarse grids. The high frequencies are first removed on the fine grid. The low frequency is then smoothed on both coarse and fine grids with different result.

on their respective grids is called aliasing of the frequencies. High frequencies that are not smoothed away first, may lead to insufficient convergence and construction of artifacts when the coarse system is solved and interpolated back to the fine grid.

The residual found on the fine grid is corrected by restricting the domain into a coarser grid in which the relaxation is continued. On the coarse grid, the residual equation is solved to acquire a correction term. The grid solution is then interpolated back to the fine grid, where the approximate solution is corrected. This basic recursive method works as a result of the linearity of the residual equation. For nonlinear problems, on the other hand, a slightly different approach is needed. The conventional technique is to apply a global linearization such as Newton's method and utilize multigrid for the solution of the Jacobian system in each iteration. We will look into FAS which applies the multigrid procedure directly to the nonlinear problem.

2.4.3 Geometric Multigrid

The multigrid methods can be divided into two classes, geometric and algebraic multigrid. This classification is based on how the method apply the smoothers and coarsening steps as illustrated in Figure 2.6. For geometric algebra, the coarsening steps are fixed. A hierarchy of grids and discretization are defined prior to the simulation and conserves the nonlinearity of the system on each grid level. The coarsening process is fixed and kept as simple as possible. This fixation of the hierarchy leads to special requirements on the properties of the smoothers. The error components that is unaffected on the coarse grid problem needs to be effectively reduced during the smoothing process (Trotten-

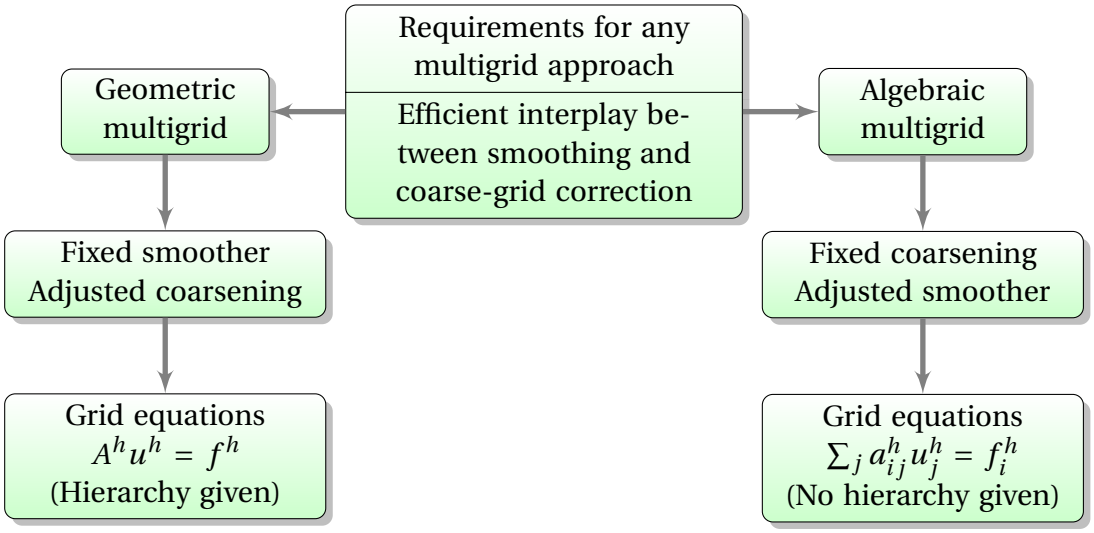


Figure 2.6: Geometric versus algebraic multigrid

berg et al. (2001)). The simple classical iterations may by this have issues with smoothing the nonlinearity properly and more sophisticated methods are required. We will look further into this when we discuss nonlinear multigrid in the next section.

2.4.4 Algebraic Multigrid (AMG)

The second approach to the application multiple grids are the algebraic multigrid (Ruge and Stüben (1987)). This approach is contrasting from geometric multigrid as it attempts to maintain the smoothers simple and operate without any given hierarchy of predetermined grids. The smoother is usually a simple iterative method. In AMG methods the coarse grid correction is fully defined when the prolongation is known. This leads to a flexible method where the coarsening process is automatically adjusted to coarsen only in the direction where the smoother really smoothens the error. This ability to adapt to specific

problem requirements is the main reason for AMG's robustness for solving large classes of problems despite using very simple pointwise smoothers (Trottenberg et al. (2001)). The downside of this flexibility is the extra cost of the setup phase where the problem is analyzed, the coarse levels are constructed and operators are assembled. This leads to extra overhead, which is why AMG in general is less effective than geometric multigrid approaches. This is of course dependent on the problem where geometric multigrid can be applied effectively.

2.4.5 Nonlinear Multigrid Method: FAS

Similar to the linear case, the nonlinear FAS multigrid method computes a coarse-grid correction term by use of the residual from the finer grids (Brandt (1982),Molenaar (1995),Henson (2003)). Even though the mechanics are very similar to many other multigrid methods, the FAS is actually computing a full approximate solution on the coarsest grid, rather than only acting on the residual as in linear multigrid. The FAS method is performed in cycles and uses multiple layers of grids that are determined in advance. The number of levels is noted as l and the method starts at the finest level $k = 1$. Here, we note the sequence of grids as Ω_k.

In the case of a nonlinear problem, we can write (2.62) on Ω_h as

$$N(\mathbf{u}) = \mathbf{f}, \tag{2.68}$$

where N is the nonlinear operator on $\mathbf{u}$. Frequently, the nonlinearity of a system is written as $A(\mathbf{u})$, but in this thesis we choose to use N for simplicity and clarity. Here we define the error $\mathbf{e} = \mathbf{u} - \mathbf{v}$ and the residual $\mathbf{r} = \mathbf{f} - N(\mathbf{v})$ in the same

manner as in the linear case. Since N is nonlinear, $N(\mathbf{e}) \neq \mathbf{r}$ in general. This means we can not determine the error by solving a simple linear equation on the coarse grid as with linear multigrid. The residual equation must thereby be written as

$$N(\mathbf{u}) - N(\mathbf{v}) = \mathbf{r}. \tag{2.69}$$

By using this error relation we can rewrite (2.69) to

$$N(\mathbf{v} + \mathbf{e}) - N(\mathbf{v}) = \mathbf{r}. \tag{2.70}$$

Suppose we have an appropriate discretization N_h and found an approximation, $\mathbf{v_h}$, then (2.70) on Ω_h becomes

$$N_h(\mathbf{v_h} + \mathbf{e_h}) - N_h(\mathbf{v_h}) = \mathbf{r_h}. \tag{2.71}$$

The coarse grid correction on Ω_H is then

$$N_H(\mathbf{v_H} + \mathbf{e_H}) - N_H(\mathbf{v_H}) = \mathbf{r_H} \tag{2.72}$$

where the coarse grid residual is the restriction of the residual on the fine grid

$$\mathbf{r_H} = I_h^H \mathbf{r_h} = I_h^H(\mathbf{f_h} - N_h \mathbf{v_h}). \tag{2.73}$$

Here, I_h^H is the restriction operator. In the same way, the fine-grid approximation is restricted to the coarse grid with $\mathbf{v_H} = I_h^H \mathbf{v_h}$. This restriction of the approximation is what makes this method different from linear multigrid, where only the residual is restricted. By substituting the restriction into the coarse-

grid residual equation (2.72), we can write it as

$$N_H(\underbrace{I_h^H \mathbf{v_h} + \mathbf{e_H}}_{\mathbf{u_H}}) = \underbrace{N_H(I_h^H \mathbf{v_h}) + I_h^H(\mathbf{f_h} - N_h(\mathbf{v_h}))}_{\mathbf{f_H}}. \tag{2.74}$$

Here, the right-hand side of the equation is known and on the same form as (2.68). If we assume that we find a solution $\mathbf{u_H}$ to the system, we can then compute the coarse grid correction term

$$\mathbf{e_H} = \mathbf{u_H} - I_h^H \mathbf{v_h}. \tag{2.75}$$

This can then be interpolated back to the fine grid and used to correct the fine-grid approximation $\mathbf{v}_h$:

$$\mathbf{v_h} \leftarrow \mathbf{v_h} + I_H^h \mathbf{e_H}. \tag{2.76}$$

If N_h and N_H are linear operators, it is easy to see that the FAS two-grid method is equivalent to the linear multigrid correction scheme introduced in section 2.4.1. FAS is by this regarded as a generalization of coarse-grid correction for nonlinear problems. A variation of FAS is written to view the method as an enhancement of the coarse-grid equations (Henson (2003)). The coarse grid equations (2.74) then takes the form

$$N_H(\mathbf{u_H}) = \mathbf{f}_H + \tau_{\mathbf{h}}^{\mathbf{H}}, \tag{2.77}$$

where

$$\tau_{\mathbf{h}}^{\mathbf{H}} = N_H(I_h^H \mathbf{v_h}) - I_h^H(N_h(\mathbf{v_h})) \tag{2.78}$$

defines the tau correction $\tau_{\mathbf{h}}^{\mathbf{H}}$. In the literature, $\tau_{\mathbf{h}}^{\mathbf{H}}$ is also called the (h,H)-

relative truncation error since it is closely related to the role of the truncation error $\tau_{\mathbf{h}}$, which is the local discretization error between the continuous solution on Ω and the discretized approximation on Ω_h. By this analogy we see that since $\tau_{\mathbf{h}}^{\mathbf{H}} \neq 0$ in general, the solution $\mathbf{u_H}$ of the coarse grid,is not the same as the solution of the original equation. The solution $\mathbf{u_H}$ is actually converging towards an accuracy that matches the solution of the fine grid, but with the resolution of the the coarse grid.

As described earlier, the method received its name since the coarse-grid problem is solved for the full approximation rather than only the error $\mathbf{e}_H$. A complete summary of the FAS multigrid cycle is described in pseudocode in Algorithm 2.1.

Algorithm 2.1 : Pseudocode of the FAS multigrid cycle

procedure FASCYCLE($\mathbf{u_k^m}, N_k, \mathbf{f_k}, \nu_1, \nu_2, k, l$)
  $\bar{\mathbf{u}}_\mathbf{k}^\mathbf{m} = \text{SMOOTHRESIDUALS}(\mathbf{u_k^m}, N_k, \mathbf{f_k}, \nu_1)$
  $\bar{\mathbf{r}}_\mathbf{k+1}^\mathbf{m} = \hat{I}_k^{k+1}\left(\mathbf{f_k} - N_k \bar{\mathbf{u}}_\mathbf{k}^\mathbf{m}\right)$
  $\bar{\mathbf{u}}_\mathbf{k+1}^\mathbf{m} = I_k^{k+1} \bar{\mathbf{u}}_\mathbf{k}^\mathbf{m}$
  $\mathbf{f_{k+1}} = \bar{\mathbf{r}}_\mathbf{k+1}^\mathbf{m} + N_{k+1}\left(\bar{\mathbf{u}}_\mathbf{k+1}^\mathbf{m}\right)$
  if k < l **then**
    $\hat{\mathbf{v}}_\mathbf{k+1}^\mathbf{m} = \text{FASCYCLE}(\bar{\mathbf{u}}_\mathbf{k+1}^\mathbf{m}, N_{k+1}, \mathbf{f_{k+1}}, \nu_1, \nu_2, k+1, l))$
  else
    $\hat{\mathbf{v}}_\mathbf{k+1}^\mathbf{m} = \text{SOLVE}(N_{k+1}, \mathbf{v_{k+1}^m}, \mathbf{f_{k+1}})$
  end if
  $\mathbf{u_k^m} = \bar{\mathbf{u}}_\mathbf{k}^\mathbf{m} + I_{k+1}^k\left(\bar{\mathbf{u}}_\mathbf{k+1}^\mathbf{m} - \hat{\mathbf{v}}_\mathbf{k+1}^\mathbf{m}\right)$
  $\mathbf{u_k^{m+1}} = \text{SMOOTHRESIDUALS}(\mathbf{u_k^m}, N_k, \mathbf{f_k}, \nu_2)$
  return $\mathbf{u_k^{m+1}}$
end procedure

2.4.6 The Phases and Components of FAS

The FAS algorithm can be divided into three main phases. Two smoothing phases and a main phase in which the coarse grid correction is performed. For FAS to be effective as an multigrid method, we need a relaxation scheme that is effective for nonlinear systems. The conventional smoothers mentioned for the linear case may have difficulties with smoothing nonlinear modes. Instead, one frequently uses the nonlinear Gauss-Seidel method (Ortega and Rheinboldt (2000)). In our case we apply Newton with CPR-AMG as a smoother to achieve a semi-global linearization.

We will now look into the details of each phase of a single step in the FAS-cycle:

(1) Presmoothing: The first step of the cycle is to run $\nu_1 (\geq 0)$ smoothing steps on the system where $\bar{\mathbf{u}}_k^n$ is the residual smoothed approximation of $\mathbf{u}_k^n$. Here, we apply a single or at most a few iterations of a smoother to ensure that the system is properly smooth. It is not necessary for the errors to become small in this step.

(2) Coarse-grid correction: This is executed in several steps, where each variable is restricted from the fine grid Ω_k to Ω_{k-1}. First the current residual $\bar{\mathbf{r}}_k^n$ of the smoothed approximation is computed

$$\bar{\mathbf{r}}_k^n = \mathbf{f}_k - N_k\left(\bar{\mathbf{u}}_k^n\right), \tag{2.79}$$

where N_k is an appropriate discrete operator on Ω_k and $\mathbf{f}_k$. Then, we restrict the residual by applying a restriction function I

$$\bar{\mathbf{r}}_{k-1}^n = I_k^{k-1} \bar{\mathbf{r}}_k^n. \tag{2.80}$$

In addition, we need to restrict the smoothed approximation

$$\bar{\mathbf{u}}_{k-1}^n = \hat{I}_k^{k-1} \bar{\mathbf{u}}_k^n, \tag{2.81}$$

where $\hat{I}$ is the restriction function for the solution approximation. These two restriction functions may in general be identical.The next step is to compute the right-hand side

$$\mathbf{f}_{k-1} = \bar{\mathbf{r}}_{k-1}^n + N_{k-1}\left(\bar{\mathbf{u}}_{k-1}^n\right). \tag{2.82}$$

Now that $\mathbf{f}_{k-1}$ is established, we compute the coarse-grid approximate solution $\hat{\mathbf{v}}_{k-1}^n$ in one of two ways depending on the current value of k. If $k > 1$, the cycle is restarted with $\bar{\mathbf{u}}_{k-1}^n$ as the initial approximation. Otherwise, if $k = 1$, a fast solver is applied to compute the approximate solution $\hat{\mathbf{v}}_{k-1}^n$ of

$$N_{k-1}^n\left(\mathbf{v}_{k-1}^n\right) = \mathbf{f}_{k-1}. \tag{2.83}$$

Finally, the correction is computed and interpolated back into the finer grid by

$$\mathbf{e}_{k-1}^n = \bar{\mathbf{u}}_{k-1}^n - \hat{\mathbf{v}}_{k-1}^n, \tag{2.84}$$

$$\mathbf{e}_k^n = I_{k-1}^n \mathbf{e}_{k-1}^n. \tag{2.85}$$

When the correction term is brought back to Ω_k, it is applied to the previous approximation

$$\mathbf{u}_k = \bar{\mathbf{u}}_k^n + \mathbf{e}_k^n. \tag{2.86}$$

(3) Post-smoothing: When the coarse-grid correction is completed, the solution for the next time step is computed by applying $\nu_2 (\geq 0)$ smoothing steps to $\mathbf{u}_k$.

Restriction and prolongation operators

The most basic and often used choice for operators between different grids is the standard coarsening in which the mesh size h is doubled in each direction (Trottenberg et al. (2001)). In some cases, there is also of interest to utilize semi-coarsening in which the mesh size in one dimension is not doubled. This is especially useful in reservoir simulation, where the domain is very thin in the z-direction compared to the other axes. We will thereby use $H = 2h$ for each coarsening step in the x- and y-direction. For the restriction operator $I_h^{H=2h}$, there is a broad spectrum of functions to choose from. From MRST, we will use a simple summing function for structured grids in the same way as in Christensen et al. (2016). This is called full-weighting of the restriction operator. In the two-dimensional case, this can be written in stencil notation as

$$I_h^H = \frac{1}{4} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}_h^H. \tag{2.87}$$

If we want higher resolution of the restriction operator, we can write the full weighting operator as

$$I_h^H = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}_h^H . \tag{2.88}$$

For simplicity, we choose the same restriction functions for both the approximation and the residual. For the interpolation, we utilize a simple injection function that mirrors the restriction function.

$$I_H^h = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}_H^h . \tag{2.89}$$

The brackets are reversed to indicate that it is a distribution process. Here, the values from a single cell is distributed into four cells on the finer grid. Since there is no overlap between the distributions from any coarse cell to the cells on the finer grid, it is not necessary to weight the cells. Here as well, a higher resolution of the interpolation operator gives the full weighting as

$$I_h^H = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}_H^h . \tag{2.90}$$

2.4.7 FAS with Multiple Grids

Now that we have described a single cycle, it is time to investigate how multiple cycles can be performed. There are two main cycles that form the basis of more sophisticated cycles. V- and W-cycle are the simplest and most common cycles (Saad (2003)). The V-cycle can be described as a simple recursive restriction of the grid until the coarsest level is reached. The approximated solution is then interpolated back up to the finest level. The W-cycle starts off in the same manner as the V-cycle and restricts the grid down to the coarsest level. Then it interpolates up a level, before it restricts the grid again. This is repeated such that it forms a 'W' as illustrated in Figure 2.7. To indicate which cycle type is applied, a cycle index γ is used. Usually, the cases are $\gamma = 1$ for V-cycle and $\gamma = 2$ for W-cycle.

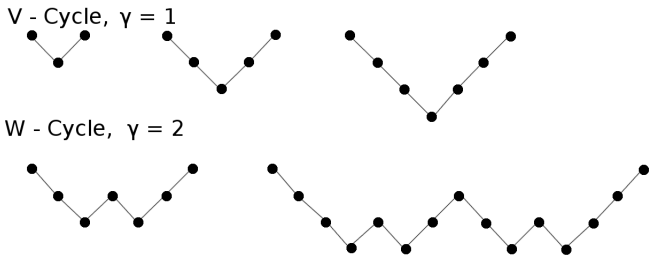


Figure 2.7: Illustration of the basic multigrid cycles. The three on top represent V-cycles with two, three, and four grids. The two lower represent W-cycle for three and four grids.

It is convenient that the number of smoothing steps ν_1, ν_2 and the cycle index γ are fixed numbers. Another approach is to let $\gamma = \gamma_k$ depend on the grid level k. Here, we mention only the F-cycle which is a combination of the V- and W-cycle and is illustrated in Figure 2.8. In this thesis, we only apply the

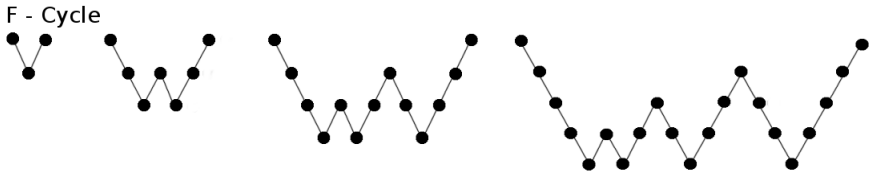


Figure 2.8: Illustration of the structure of an F-cycle with two to five grids.

V-cycle in the numerical experiments.

With this, we have finally established the full method. We have described the three-phase flow model based on mass conservation and Darcy's motion equation, discretized the equations with the finite-volume method, and described the FAS-cycle with Newton's method at the core.

Chapter 3

Implementation

As stated in the introduction, we implement FAS by using the scientific programming language MATLAB. We also apply the framework provided by MRST (Lie (2016)). The standard Newton method that is used as a reference in the testing of FAS is also provided from the MRST library. We start with presenting the automatic differentiation algorithm, which is one of the core tools in MRST. We then present MRST and how we utilized it in the implementation of FAS.

3.1 Automatic Differentiation

Automatic differentiation, AD, is a technique developed to simplify the evaluation of differential functions (Bücker et al. (2006), Nocedal (2006), Lie (2016)). The foundation of AD is built upon the observation that any function regardless of how complicated it is, can be broken down to an evaluation of a limited set of elementary and arithmetic operations such as addition, subtraction, multiplication, and division. In MATLAB we also have fundamental binary evaluation

of unary functions, e.g. exp, sin, cos, log, etc. These all have simple and well-known differentiation rules. Combined with the chain rule and by taking into consideration that any computer code consists of a sequence of instructions, we see that automatic differentiation of functions is a powerful tool. This can be illustrated with the evaluation of the function

$$f(x,y) = \left(3x^2 + xy - y\right)\left(\cos\left(\mathrm{y}\right) + 4\right) \tag{3.1}$$

as a code list:

$$\begin{aligned}
s_1 &= x, & s_6 &= s_4 + s_5, \\
s_2 &= y, & s_7 &= s_6 - s_2, \\
s_3 &= \text{power}\,(s_1, 2)\,, & s_8 &= \cos\left(s_2\right), \\
s_4 &= 3 \times s_3, & s_9 &= s_7 + 4 \\
s_5 &= s_1 \times s_2, & s_{10} &= s_7 \times s_9.
\end{aligned} \tag{3.2}$$

Here the function $\text{power}\,(a,b) = a^b$ has been included as it is one of the basic operations in MATLAB.

The key idea of automatic differentiation is to keep track of the function quantities and derivatives simultaneously. Every time an operation is performed on one of the functions, the corresponding differential operation is applied to its derivative. There are two main approaches to how AD is executed. The simplest of the two is known as the *forward mode*. Here, a directional derivative of each variable x_i in a given direction $p \in \mathbb{R}$, is evaluated and carried forward simultaneously with the evaluation of x_i itself. This gives a repeatedly substi-

tuting of the derivative of the inner functions in the chain rule:

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial w_1}\frac{\partial w_1}{\partial x} = \frac{\partial y}{\partial w_1}\left(\frac{\partial w_1}{\partial w_2}\frac{\partial w_2}{\partial x}\right) = \frac{\partial y}{\partial w_1}\left(\frac{\partial w_1}{\partial w_2}\left(\frac{\partial w_2}{\partial w_3}\frac{\partial w_3}{\partial x}\right)\right) = \ldots \tag{3.3}$$

The second method is the reverse mode in which the function f is evaluated first, and then the partial derivatives are recovered with respect to each variable x_i. In the end of the process, the gradient vector is combined from the partial derivatives with respect to the independent variables. There are several software tools developed for the execution of automatic differentiation. MRST provides a variant of the forward mode.

3.2 The Matlab Reservoir Simulation Toolbox (MRST)

MRST is a open-source project developed at SINTEF Digital (Lie et al. (2012), Krogstad et al. (2015), Lie (2016), Sintef (2018)). It consists of a large number of modules, and functionality from several third-party developers. In addition, there is functionality that supports external third party code in compiling languages. This is primarily aimed against highly effective implementation of solvers that outperforms similar implementation in high-level scripting languages such as Matlab. An example of this is the implementation of the aggregation-based algebraic multigrid, AGMG (Notay (2010)), which is applied in our FAS-framework.

The main goal of MRST is to provide a toolbox for rapid prototyping of new methods (Krogstad et al. (2015), Lie (2016)). It is a research tool that supports modeling and simulation of flow in porous media and provides a large set of mathematical models, computational methods, visualization tools and utility

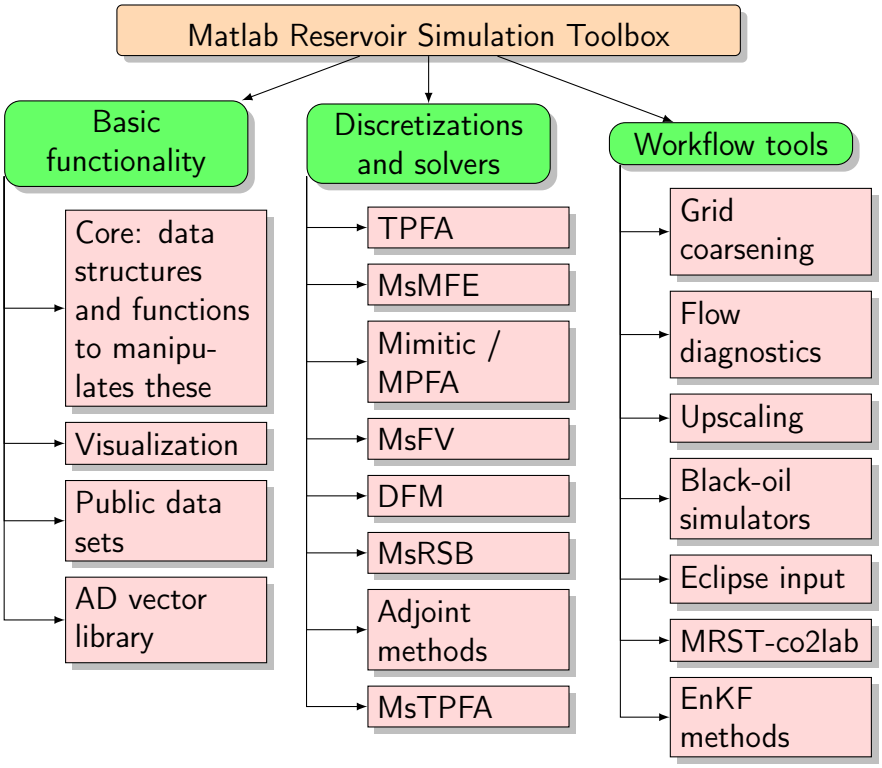


Figure 3.1: The MATLAB Reservoir Simulation Toolbox, MRST, is primarily developed at SINTEF. It is module-based and the functionality can be divided into three sections.

routines. It utilizes Matlab's highly vecorized syntax and support for a wide range of mathematical functions that enable compact and efficient programs (Alberty et al. (1999), Aarnes et al. (2007)).

MRST can be divided into three sections as illustrated in Figure 3.1. The first section makes the foundation of MRST and consists of the basic functionality. It provides the main data structures, generic functionality, a vectorized AD library, visualization tools and access to public data sets. The second section is a collection of discretizations and solvers. Much of the functionality we use is located in the third section, the workflow tools. Here we have functions providing

grid coarsening, flow diagnostics, upscaling tools and black-oil simulators.

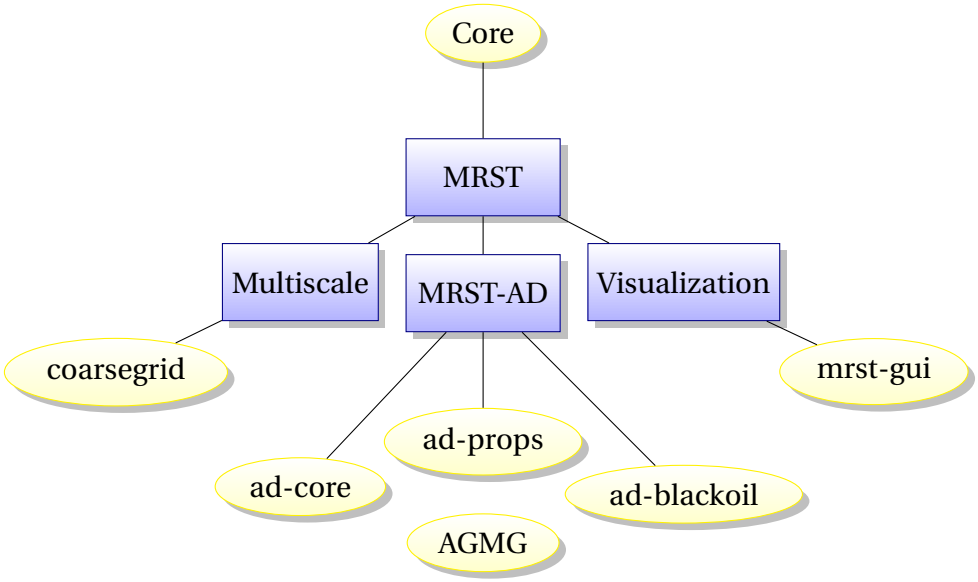


Figure 3.2: MRST modules, in ellipses, applied for the implementation of FAS. The isolated module is a third-party module supported by MRST. All modules are add-ons to the core module that provides the basic data structures and utility functions.

To ensure the independence of the functionality of MRST it is divided into modules. Each module consists of a set of functions and scripts that extend or modify the existing capabilities of MRST. These modules can easily be used as add-ons for the simulations. The modules applied in the implementation and execution of FAS are shown in Figure 3.2. As the figure illustrate, we have used a module named *ad-core*. This is an example of an add-on module that alternate the basic functionality of the core module as it implements the base classes for a object oriented framework for implicit and sequential simulators. In our implementation we have applied the object oriented automatic differentiation, (AD-OO), based solvers. These solvers are written as a object oriented

framework that makes it easy to implement new methods. The main part of our implementation consists of two classes in which one inherits from an existing class in MRST. The class `FullApproximationModel` inherits from the main model class named `PhysicalModel`. Our second class `ModelHierarchy` is an independent class that connects FAS to the models generated by a subclass of `PhysicalModel`. Figure 3.3 shows how these classes are connected and how they are integrated in the workflow of MRST. As we see from the figure, the workflow of an simulation starts with the set up of the models and other parameters. These are then passed to the Simulator that calls the Nonlinear Solver for each timestep. The solver further calls FAS which performs the approximation of the timestep.

To solve the system, we use a state-of-the-art CPR-AMG preconditioner to accelerate the solution process (Notay (2010)). This solution strategy is applied for both FAS and standard Newton. In FAS, CPR-AMG is applied in the subroutines of the smoothing processes and main solver at the coarsest grid.

By applying the modularity of MRST, we are able to construct a general nonlinear solver that can be applied on a broad range of classes of discrete reservoir and fluid models. This use of generic components makes it easy to switch subroutines such as the smoother and core solver. This provides the freedom and flexibility of testing the effect on the performance by each individual component. The approach also supports the extension of future testing for more advanced flow systems.

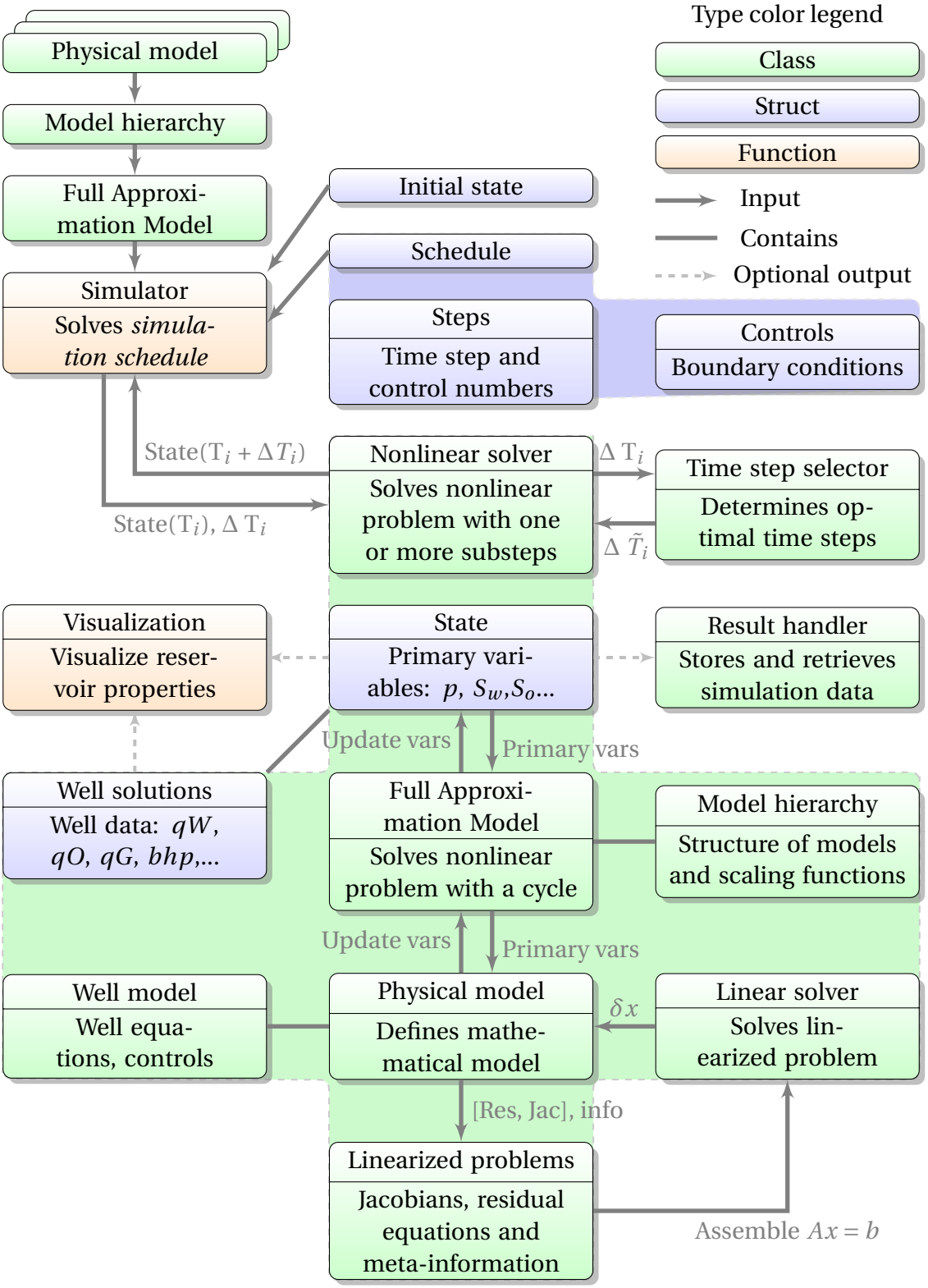


Figure 3.3: Simulation workflow for MRST with the expansion of FAS.

Chapter 4

Numerical Experiments

This chapter presents the experimental setting and the numerical results from Simulations. For all tests, two different solvers are compared. The industry-standard solution method is referred as the standard Newton, (SN), or Newton, and the FAS method is conventionally referred as FAS(l), where l indicates the number of used grid levels.

4.1 Experimental Setting

Three different reservoir models are used for the numerical experiments. The initial tests are with the classical quarter-five-spot well-pattern, QFS, with a single injector and producer (Willhite (1986)). We also consider simulations with the top layers of the SPE 10, Model 2 (Christie and Blunt (2010)). This is a standard benchmark model with significant variance in the permeability and porosity fields. The model is corresponding to the Tarbert formation. The significant heterogeneity of the permeability field makes the simulation problem

more challenging than the QFS problem. Our final tests are performed with a realization of the OLYMPUS reservoir model (Fonseca et al. (2017)). This optimization benchmark model uses a synthetic 3D reservoir, inspired by an oil field in the North Sea. The model is an unstructured grid with a permeability field consisting of multiple layers, several faults, and two zones separated by an impermeable shale layer. This provides a challenging test case and is the largest model in our tests with a few hundred thousand cells.

4.2 Simulation Set-Up

This setup is inspired from the litterature and represent common values for the fluids and reservoir (Lie (2016), Christensen et al. (2016)). For all the simulations, the stopping criteria and tolerances for the nonlinear solver are the same for both standard Newton and FAS. The final stopping criteria for the residual is set to an absolute tolerance of $\text{tol}_{\text{abs}} = 10^{-5}$ and the relative tolerance is set to $\text{tol}_{\text{rel}} = 10^{-4}$. The residual is computed with the 2-norm. The initial pressure of the reservoir is set to 250 bar. In addition, the pressure at the production well is held at a constant level of 200 bar. The injection rate is a constant function of $100\text{m}^3/\text{day}$. The properties of the fluids and rock are presented in Table 4.1. For all the tests, FAS is executed with V-cycles where the initial number of presmoothings and postsmoothings is set to a maximum of 1. The number of smoothing steps is doubled on each coarsening step. Experimentation show that this is a satisfactory approach to the number of smoothing steps. In addition a second smoothing step is performed after each cycle. A small convergence check is conducted after each presmoothing phase to avoid applying

more grid levels than necessary. The maximum number of outer iterations is set to the MRST default of 25.

Table 4.1: Properties of the three phases and the rock formation.

Medium	Property	Symbol	Value	unit
Water	Viscosity	μ_w	1	kg/sm
	Compressibility	c_w	10^{-8}	1/bar
	Density	ρ_w	1022	kg/m^3
	Initial Saturation	S_w	0.2	-
	Corey-exponent	-	2	-
Oil	Viscosity	μ_o	5	kg/sm
	Compressibility	c_o	10^{-5}	1/bar
	Density	ρ_o	[800 - 940]	kg/m^3
	Initial Saturation	S_o	0.7	-
	Corey-exponent	-	3	-
Gas	Viscosity	μ_o	0.1	kg/sm
	Compressibility	c_o	10^{-4}	1/bar
	Density	ρ_o	100	kg/m^3
	Initial Saturation	S_g	0.1	-
	Corey-exponent	-	5	-
Rock	Permeability	$\mathbf{K}$	[0.1 - 3000]	md
	Porosity	ϕ	0.3	-

When we are testing for the algorithmic efficiency, we primarily look at the number of outer iterations required to achieve convergence. For FAS, a single V-cycle corresponds to one outer iteration. The timings are measured for only the simulation in it self, and excludes the time used for simulation setup. For FAS this includes the restriction and interpolations as a part of the solver. It is also worth noting that the timings are the total wall-time during the simulation, including many subroutines of the simulation algorithm not affected by FAS. For this project we look at the performance sensitivity against changes in grid size, mobility ratio and homogeneous as well as heterogeneous permeability. The

same generated heterogeneous permeability fields is used for both methods.

4.3 Quarter-Five-Spot

The QFS test is a standard benchmark in reservoir simulation due to its symmetry and well known reference solution. The model is also used to show difference in scalability properties between Newton and FAS. An illustration of the model is shown in Figure 4.1. The first step in our tests is to validate the correctness of our implementation of FAS. This is done by computing and comparing the oil production for both FAS and the reference industry-standard Newton method. To provide a solid verification, we let the MRST simulation run for 30 years with constant injection rate. This assures that most of the cells are subject to phase displacement. Figure 4.2 shows the validation of FAS with the oil production. We observe that the two methods have produced the same amount of oil for each timestep, and thus verifies the correctness of FAS.

4.3.1 Homogeneous permeability field

For the initial tests we use a homogeneous permeability field. With this simulation set up we test how the methods are affected by change in grid size and for different permeability values.

Scalability

For the first test we compare the number of outer iterations for the two methods with different grid sizes. The results are presented in Table 4.2. As we can see the algorithmic efficiency for FAS is higher than for the standard Newton since

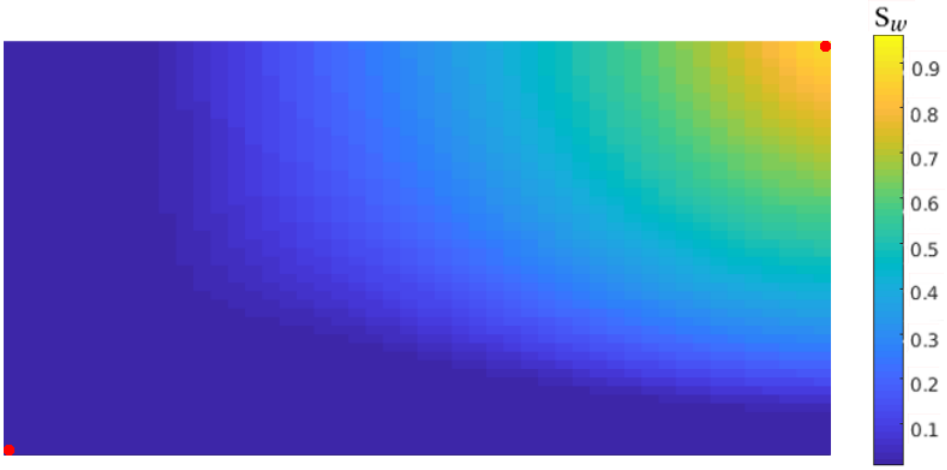


Figure 4.1: Water saturation after injection of 40% pore volume for the quarter-five-spot model with homogeneous permeability field. A well in the top-right corner injects water and a production well is in the bottom-left corner. The domain size is $1000 \times 500 \times 20$m .

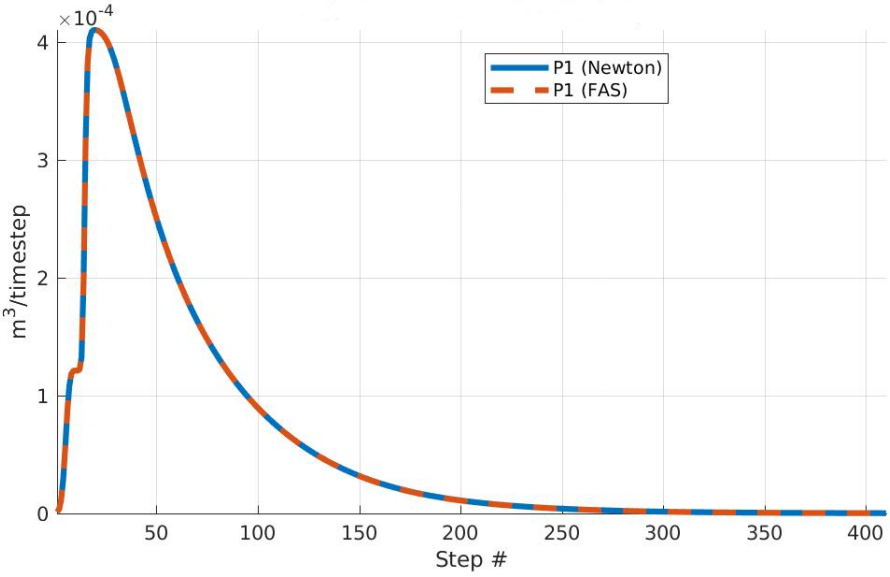


Figure 4.2: Validation of FAS against an industry standard Newton solver for the QFS model.

it requires fewer outer iterations. This difference grows in favor for FAS as the grid size is increased. This implies that the scalability of FAS is better than with Newton. For FAS(2) the change in outer iterations is close to negligible, and with three grid levels a second cycle is never performed. This is as expected with regard to the fact that multigrid methods are theoretically less affected by changes in grid sizes on the coarser grids.

Table 4.2: Test of algorithmic efficiency with the QFS-model. Displays the average number of outer iterations for both Newton and FAS. Simulation duration is 5 years with 200 steps.

	Problem Size			
Method	16×16×10	32×32×10	64×64×10	128×128×10
Newton	2.05	2.07	2.09	2.16
FAS(2)	1.00	1.00	1.01	1.03
FAS(3)	1.00	1.00	1.00	1.00

In Figure 4.3 we see that the runtime of FAS is lower than for Newton for growing grid size. For the homogeneous permeability field the difference between applying two and three grids are close to negligible. FAS(3) is only slightly faster than FAS(2).

Permeability

In addition to scalability, we are also interested in the sensitivity of increasing permeability. As described in the theory chapter, higher permeability means the flow is less hindered. This causes the flow to go faster, and the increased flux may make the computations more challenging. From Figure 4.4 we observe that the two methods have similar, if not identical reactions to the change in permeability. The results show that FAS outperforms Newton in runtime ef-

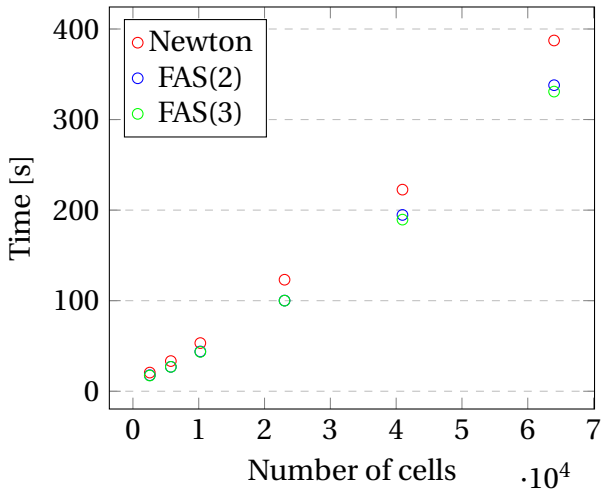


Figure 4.3: Runtime comparison between Newton and FAS for different grid sizes. The Permeability field was set to 100md with domain dimension of 1000 × 500 × 20m. Simulation duration was 5 years with 100 timesteps.

ficiency for simple permeability fields.

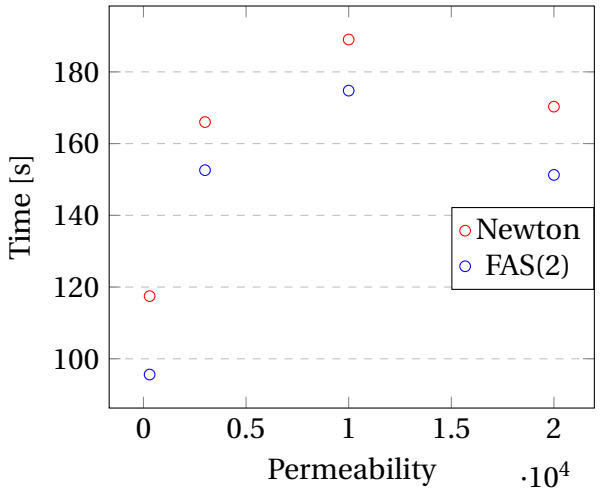
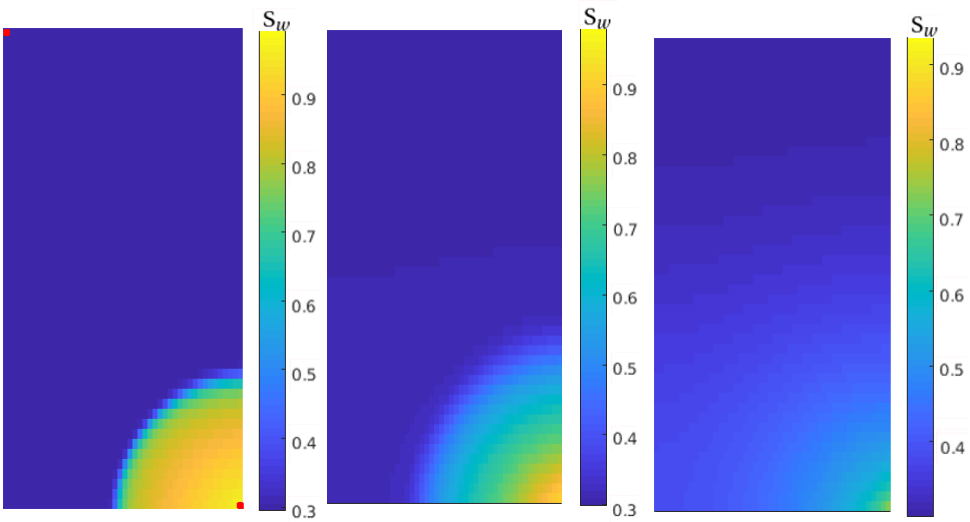


Figure 4.4: Runtime comparison between Newton and FAS(2) for different permeability. The grid size is set to 48 × 48 × 10 with domain dimension of 1000 × 500 × 20m. The duration of the simulation is 150 days with 100 time steps.

Mobility ratio

The mobility ratio has a significant impact on the fluid flow. Figure 4.5 illustrate the effect of different mobility ratios between water and oil. As we see, the front interface between the two phases become less sharp with decreasing mobility ratio. When comparing the sensitivity for the mobility ratio between water and oil, we observe from Figure 4.6, that there is a slight reaction difference between Newton and FAS. Here we see that both react relatively similar for the higher mobility ratios. The largest difference is observed for the lower mobility ratio, where the standard Newton have much higher increase in run time compared to FAS. When the mobility ratio becomes smaller we expect to get a shock within the reservoir between the two fluids and the system becomes more stiff. The fact that FAS have less increase in run time for the lowest mobility ratio, suggests that Newton is relatively more sensitive to stiff systems. FAS on the other hand, seems to have a slower degeneration of computation efficiency when the system become increasingly stiff.

When we perform the same tests for a longer time period, we see a change in the results between the two methods. With a simulation duration of 5 years we see in Figure 4.7 that Newton have a slightly shorter simulation runtime than FAS(2). At the same time, we observe the same tendency in algorithmic efficiency as in the scaling tests in Table 4.3. FAS is on average converging after only a single iteration, while Newton requires up to three iterations. Upon closer inspection of the runtime report from each timestep, we observe that FAS have a tendency to require less computation time for the first part of the simulation. About a hundred days into the simulation, Newton begins to have a shorter computation time than FAS. This coinside closely with the timesteps



(a) Mobility ratio 10:1 (b) Mobility ratio 1:1 (c) Mobility ratio 1:15

Figure 4.5: Illustration of the effect on fluid flow with different mobility ratio between water and oil. Water saturation after injection of 20% pore volume for the quarter-five-spot model with homogeneous permeability field. A well in the bottom-right corner injects water and a production well is in the top-left corner. The domain size is $1000 \times 500 \times 20$m. The duration of the simulation is 150 days with 100 time steps.

where Newton begins to require only two iterations. During this part of the simulation Newton alternate mostly between using three and four iterations for each timestep. After the initial simulation time, Newton stabilizes on two iterations for each timestep and is slightly more time efficient than FAS. In the final year of the simulation time, Newton drops down to requiring a single iteration. In this time period we observe that FAS again is slightly more time efficient than Newton. These observations may be a result of the degree of nonlinearity in the system. Typically, we expect the nonlinearity to be stonger in the beginning of the simulation when water is injected into a stable system. As expected the semi-linearization with FAS perform better than the global linearization with

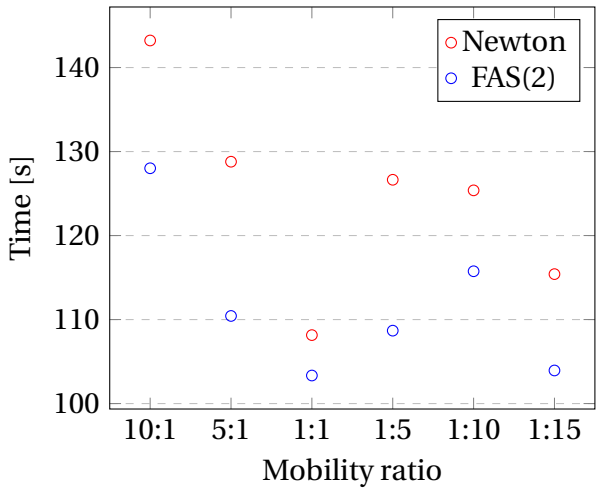


Figure 4.6: Runtime comparison between Newton and FAS for different mobility ratios between water and oil. The grid size is $48 \times 48 \times 10$ with domain dimension of $1000 \times 500 \times 20$m. The duration of the simulation is 150 days with 100 time steps.

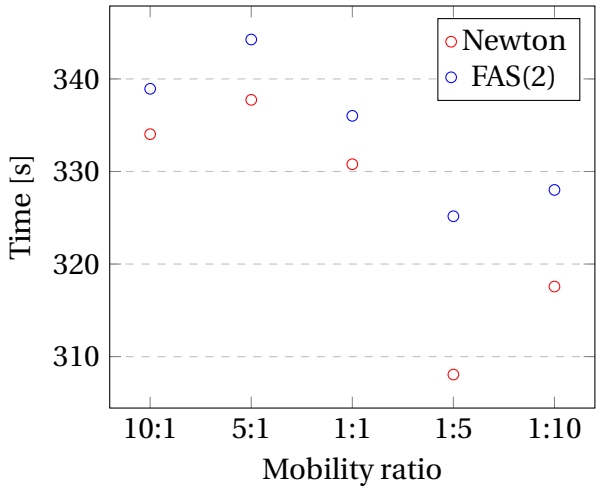


Figure 4.7: Runtime comparison between Newton and FAS for different mobility ratios between water and oil. The grid size is $48 \times 48 \times 10$ with domain dimension of $1000 \times 500 \times 20$m. The duration of the simulation is 5 years with 100 time steps.

Newton in handling the system when the nonlinearity is stongest.

These results suggests that the simulation may be optimized by first applying FAS for the first time period of the simulation when the nonlinearity is strongest, and then switch to standard Newton. When Newton starts to converge after a single iterations, our observations suggests to switch back to FAS for acceleration of the computation for the final simulation duration.

Table 4.3: Test of algorithmic efficiency for different mobility ratios between water and oil. Displays the average number of outer iterations and runtime for both Newton and FAS. The duration of the simulation is 5 years with 100 timesteps.

	Iterations		Runtime	
Mobility ratio	Newton	FAS(2)	Newton	FAS(2)
10:1	2.85	1.00	334.12	338.93
1:1	2.90	1.00	337.74	344.26
1:5	3.00	1.00	330.78	336.01
1:10	2.73	1.00	308.06	325.16
1:15	2.77	1.00	317.57	328.01

4.3.2 Heterogeneous permeability fields

With MRST we create a randomized heterogeneous permeability field within a given range, as displayed in Figure 4.8. This causes the phases to flow more irregularly through the domain.

Scalability

We also test scalability for heterogeneous permeability fields. Figure 4.9 displays the runtime for FAS and Newton, for different grid sizes for a fixed permeability field with range $\mathbf{K} \in [10 - 1000]$md. Here as well, we observe that FAS

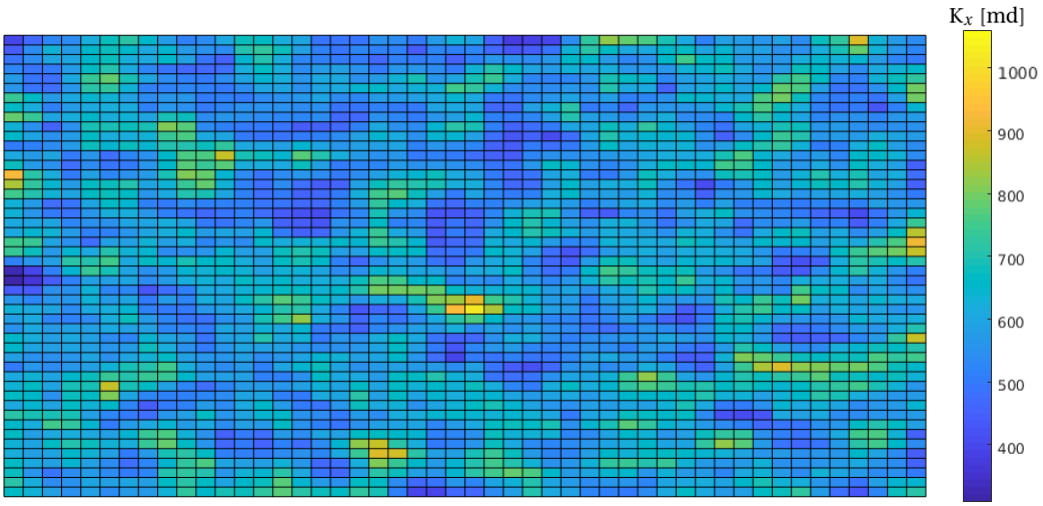


Figure 4.8: Heterogeneous permeability field for a QFS model with domain size $1000 \times 500 \times 20$m.

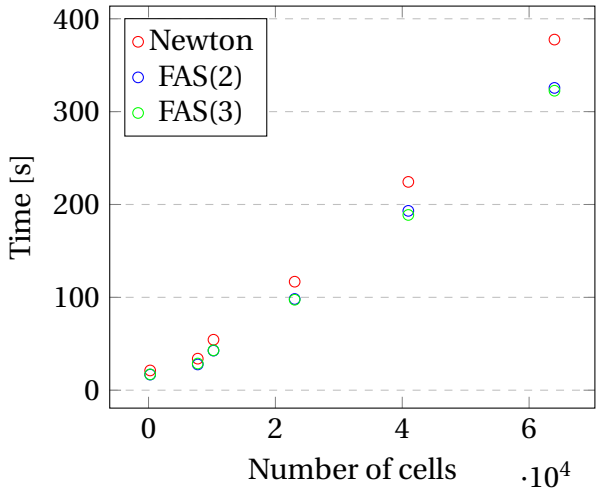


Figure 4.9: Runtime comparison between Newton and FAS for different grid sizes. The Permeability range was set to $\mathbf{K} \in [10-1000]$md with domain dimension of $1000 \times 500 \times 20$m. he duration of the simulation is 5 years with 100 time steps.

is less affected by an increase in grid size. When comparing the results for the scaling tests with homogeneous and heterogeneous permeability fields, we see that the two methods have similar increase in runtime. This indicates that neither method has an advantage when computing on heterogeneous permeability fields. We also observe that the runtime for heterogeneous permeability field is slightly higher than for the homogeneous case. This confirms that the heterogeneous case is more challenging, but grid size is more prominent in affecting the performance. When looking into the algorithmic efficiency, we observe the same tendency as for the homogeneous case in Table 4.2. While Newton requires between 2.00 and 2.14 with a steady increase according to the grid size, FAS(2) requires only 1.00 to 1.03 outer iterations for the same grid sizes.

Permeability ranges

To further investigate the sensitivity of heterogeneous permeability fields, we performed tests with different ranges of permeability. The results are displayed in Figure 4.10. Here we see as before that FAS outperforms Newton in runtime efficiency. We also observe that the difference in runtime between the two varies relatively little with the difference in permeability ranges. This indicates that the two methods react relatively equally to the change in the permeability field. This confirms our previous observations that none of the methods have an advantage concerning the permeability field itself. We also observe, as expected, that the permeability field with the largest span proves to be the most challenging case for both FAS and Newton.

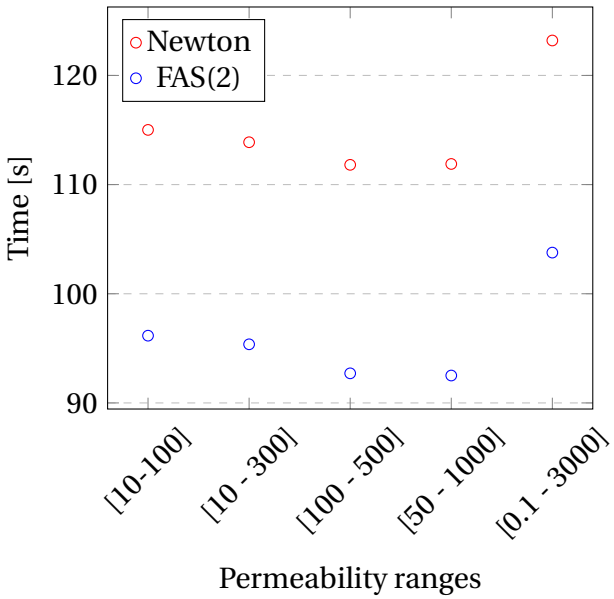


Figure 4.10: Runtime comparison between Newton and FAS for different permeability ranges. The grid size is set to $48 \times 48 \times 10$ with domain dimension of $1000 \times 500 \times 20$m. The duration of the simulation is 150 days with 100 time steps.

4.4 SPE10

The case derived from Model 2 of SPE10 provides an appropriate benchmark test with its highly variable permeability field. Originally intended as an upscaling benchmark, the significant variation and freely available data set has made it a de-facto benchmark for any novel solver. Figure 4.11 shows the permeability field of the top layer of SPE10, whereas Figure 4.12 illustrates the water saturation after injection of 40% of the pore volume. The impact of the permeability field is clearly seen on the displacement front. As in the QFS example, FAS and standard Newton were tested for algorithmic efficiency.

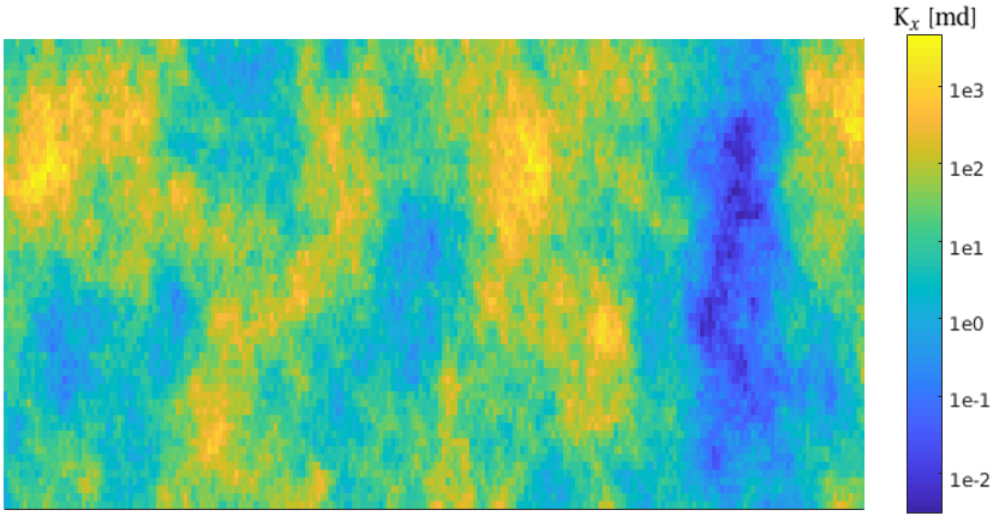


Figure 4.11: Permeability field of the top layer of the SPE10 model. The high variance in permeability makes it a challenging model to simulate on. The permeability distribution is plotted in the unit millidarcy.

Time step size

Compared to the previous case, we observe from Table 4.4 that FAS(2) requires more iterations than standard Newton for small timesteps. It is interesting to observe that only Newton needs additional iterations when longer time-steps are used. This is in contrast with FAS, which converges at approximately the same rate when the length of the time-steps increases, indicating a better treatment of the nonlinearity of the problem. It is worth noting that the linear systems do not generally become more difficult to solve for longer time-steps. Longer time-steps are typically a source of reduced convergence, or even convergence failure in reservoir simulation. As before we see that FAS with three grids converges after only a single cycle. It is also worth noting that by investigation of runtime, FAS(3) uses approximately half the time FAS(2) requires to

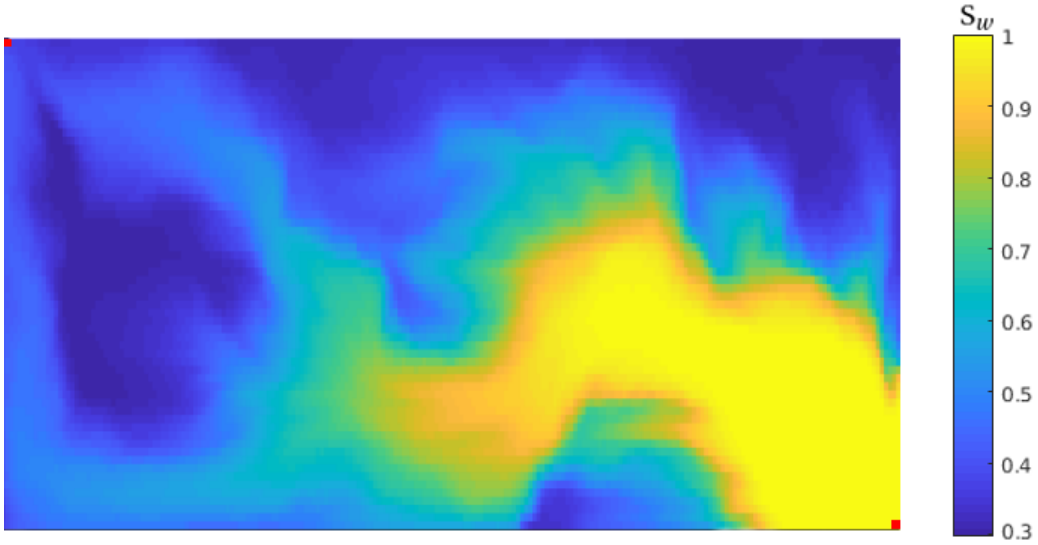


Figure 4.12: Illustration of the water saturation in the top SPE10 layer after injection of 40% pore volume. An injection well is placed in the bottom right corner and a production well is in the top left corner. A highly heterogeneous permeability field significantly impacts the flow direction.

complete the simulation. This leads towards to the conclusion that FAS(3) is much more efficient than FAS(2) and Newton in terms of numerical efficiency.

Table 4.4: Test of algorithmic efficiency for the top SPE10 layer for different length of timesteps. Displays the average number of outer iterations for both Newton and FAS. The duration of the simulation is 5 years.

	Step size in days					
Method	$\frac{1}{2}$	1	2	4	8	16
Newton	2.04	2.19	2.67	2.40	2.81	3.15
FAS(2)	2.99	2.99	2.98	2.97	2.95	2.93
FAS(3)	1.00	1.00	1.00	1.00	1.00	1.00

Initial saturation

In the following tests we want to investigate how the placement of different oils affect the simulation efficiency. We apply a light oil with density of $\rho_{0,l} = 800$ and a heavy oil with density $\rho_{0,h} = 940$. In the first saturation test, we place a light oil in the top layer, and a heavy oil in the lower layer. In each oil section the initial saturation of the respective oil is set to 70% and water to 30%. As before, we place an injection and production well in opposite corners of the reservoir model. In Figure 4.13 we see the saturation flow of water after injection of 40% pore volume in the first saturation test. From Table 4.5 we observe that both methods require more iterations to solve the three phase system with two different oil phases. As before we see that FAS have an advantage over Newton and on average require fewer outer iterations for each timestep. Compared with the results in Table 4.4, we see that Newton have a larger increase in outer iterations than FAS. This may indicate that Newton is more affected by the presence of a second oil phase than by a gas phase in three phase systems. We also see that FAS(3) converges faster than FAS(2) and converges after a single cycle for the cases where the oils are placed in separate ends of the reservoir.

Table 4.5: Test of algorithmic efficiency for different initial saturation for the top two SPE10 layers. Displays the average number of outer iterations for both Newton and FAS. The simulation duration is 150 days.

	Method		
Oil placement	Newton	FAS(2)	FAS(3)
Light oil on top, heavy oil below	3.91	2.91	2.52
Light oil to the right, heavy oil to the left	3.3	2.91	1.00
Light oil to the left, heavy oil to the right	3.73	2.85	1.00

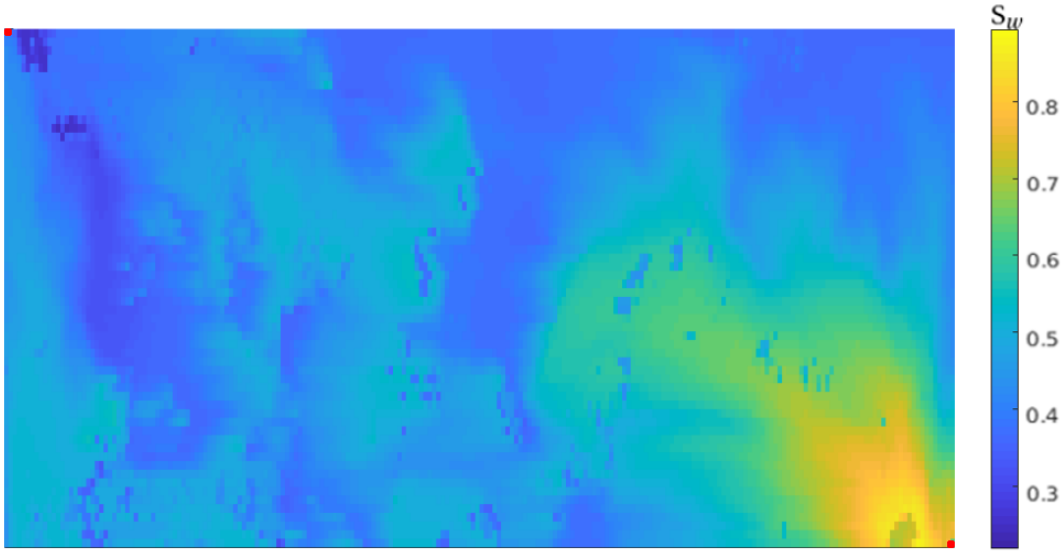


Figure 4.13: Water saturation in the top two SPE10 layers after injection of 40% pore volume. A light oil is placed in the top layer, and a heavy oil is placed in the lower layer. An injection well is placed in the bottom right corner and a production well is in the top left corner.

Wells

In the recovery process of hydrocarbons, multiple wells are used in the same reservoir. For the tests with multiple wells, the two top layers of SPE10 was used with simulation duration of 5 years and 200 timesteps. The constant injection rate was chosen such that during the simulation the volume of the injected water equals the total pore volume of the reservoir. The production wells were placed in corners of the reservoir, and furthest away from the corner of the injection well. With four production wells, the injection well was placed in the center of the reservoir, see Figure 4.14. In Table 4.6 we see the effect on the performance of the methods for growing number of production wells. As before we see that FAS requires fewer iterations than Newton. For a single and

two production wells, we see that FAS(2) only uses slightly fewer iterations than Newton. The difference become more significant for three and four production wells. We also observe that FAS converges after only one iteration with three grid levels. In addition FAS(3) completes the simulation in approximately half the time of FAS(2). This is the case for all well configurations. This suggests that FAS with three grids has a significant benefit for cases where two grids only have a slight algorithmic improvement compared with Newton.

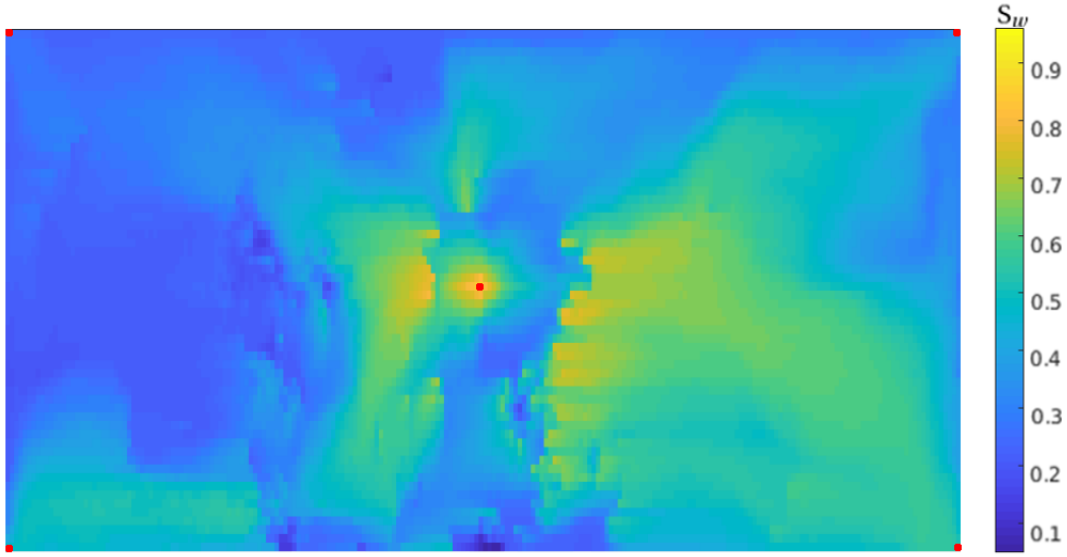


Figure 4.14: Water saturation in the top two SPE10 layers after injection of 40% pore volume. An injection well is placed in the center, and four production wells are in the corners of the reservoir.

4.5 Olympus

The Olympus reservoir simulation model is an artificial model based on existing fields in the Nordic Sea (Fonseca et al. (2017)). The field is 9km by 3km

Table 4.6: Test of algorithmic efficiency for growing number of wells. Displays the average number of outer iterations for both Newton and FAS. The production wells are placed in corners furthest away from the injection well. The simulation duration is 5 years with 100 timesteps on the two top SPE10 layers.

	Production wells			
Method	1	2	3	4
Newton	3.18	3.08	2.55	2.82
FAS(2)	2.94	2.91	1.15	2.18
FAS(3)	1.00	1.00	1.00	1.00

and is 50m thick. It consists of 16 layers and 6 faults. A fault is a displacement of the layers. This gives a discontinuity in the permeability fields and is frequently occurring in oil reservoirs. In reservoirs, one can often observe large differences in permeability between different rock types. The permeability field seen in Figure 4.15 is a typical example of high permeable channel deposits interbedded on a low-permeable background flood-plaine deposit. Here, the main flow will follow the high-permeable river channels, while at the same time the background flood-plain deposits have sufficient permeability to transmit fluids. This gives an gives a complex model that are challenging to simulate and is an excellent benchmark test for the robustness of numerical algorithms.

Time step size

In table 4.7 we see the algorithmic efficiency for the two methods with different length of the time step. The duration of the simulations is 100 days with an injection rate of 100m^3 per day. As we have seen before, FAS requires few cycles for each simulation. It is interesting to see that FAS(2) on average uses only a single cycle to compute each timestep. When we investigated the behaviour of the cycles closer, we observed that FAS often did not enter the second grid level.

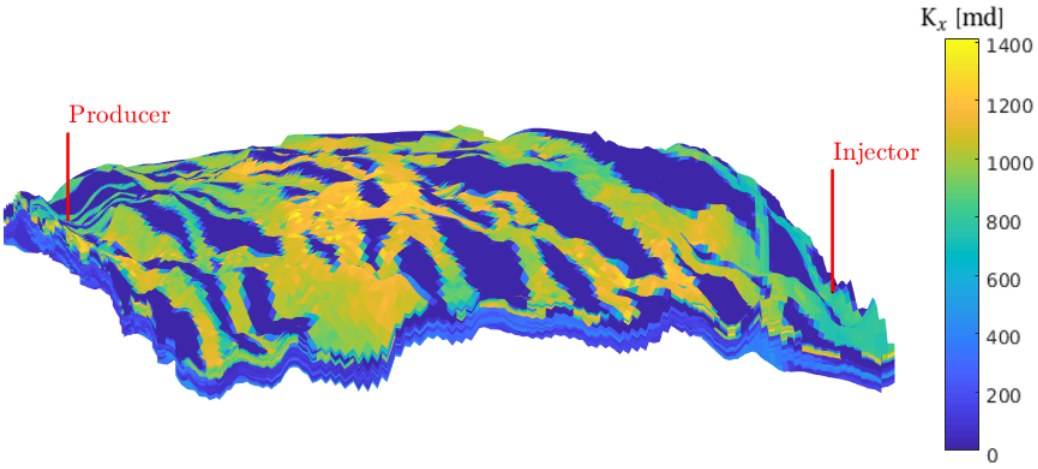


Figure 4.15: Illustration of the permeability field of the Olympus model. The grid consists of 118 × 181 × 16 cells approximately 50m × 50m × 3m each. The injection and production wells are paced in each end of the reservoir.

This was caused by an 'premature' convergence during the smooting step. This coincided with the reference Newton solver converging in a single step at the same timesteps. This behaviour is exactly as intended since it is undesirable to continue with the cycle if convergence is already met during the presmoothing phase. This enhances the earlier observations that in periods of the simulation it is sufficient to apply standard Newton.

Table 4.7: Test of algorithmic efficiency for an Olympus model with different step lengths. Displays the average number of outer iterations for both Newton and FAS. The simulation duration is 100 days.

	Timesteps			
Method	50	100	150	200
Newton	2.76	2.18	2.07	2.01
FAS(2)	1.00	1.00	1.00	1.00

Injection rate

The injection rate indicates the speed and stress on the phases in the reservoir. In Table 4.8 we see the results from 150 days of simulation with 100 timesteps. As we observe, Newton have an increasing amount of outer timesteps for the corresponding increase of injection rate. The only exception seems to be with an injection rate of $80 m^3/day$, where it is a drop in the average number of iterations. For the same injection rate, we see that FAS(2) also have the lowest amount of iterations. The tendency we see with Newton is not present for FAS(2). There seems to be no particular pattern or correspondence between increasing injection rate and the algorithmic efficiency of FAS(2). The results show that FAS(2) on average needs only slightly more than a single cycle to reach convergence for the different cases. As with most of the other results, we see that FAS have a significant advantage in algorithmic efficiency in reservoir simulation compared with the standard Newton.

Table 4.8: Test of algorithmic efficiency for an Olympus model with different injection rates. Displays the average number of outer iterations for both Newton and FAS. The simulation duration is 150 days with 100 timesteps.

	Injection rate (m^3/day)					
Method	40	60	80	100	120	140
Newton	2.32	2.34	2.31	2.35	2.37	2.38
FAS(2)	1.01	1.03	1.00	1.05	1.01	1.03

Chapter 5

Concluding Remarks

The main purpose of this work was to investigate and implement the multigrid method FAS for a immiscible and compressible three-phase black-oil model. For comparison we have used standard Newton from the MRST framework as a benchmark for the efficiency of FAS. For the numerical experiments we applied the quarter-five-spot, SPE10, and a realization of the Olympus model.

5.1 Summary and Conclusions

In this work, we have investigated the applicability of the nonlinear multigrid method FAS for three-phase subsurface reservoir simulations. To investigate the method, we have studied three conceptual test cases with immiscible and compressible black-oil fluids. With the standard Newton solvers for three-phase flow as a starting point, we have implemented the method in a generic manner. This will later allow us to incorporate more advanced fluid behavior, reusing the already proven components of MRST. Our approach, favoring reuse and

generic components, resulted in a semi-global linearization that was benchmarked against a standard Newton method with global linearization. For both solvers, we used a state-of-the-art CPR-AMG preconditioner to accelerate the solution process. The numerical experiments where performed with the quarter-five-spot, SPE10 and the Olympus model.

We have demonstrated that for the given model equations and problems considered, FAS outperforms standard Newton in terms of algorithmic efficiency. In the experiments we have also observed that for some cases there is an alternation between which of Newton and FAS that is the most efficient for different parts of the simulation. For the tests, both homogeneous and heterogeneous permeability fields have been applied. We have also demonstrated that the FAS method can easily be implemented with already existing solution methods in MRST as building blocks.

5.2 Further Work

Following this preliminary work for the application of FAS for reservoir simulation, there are several possible avenues for further research. One possible and natural direction is to extend the method to more advanced flow systems from MRST. E.g. black-oil flow with dissolution and/or fully compositional problems containing strong nonlinearities, for which the FAS methodology may prove beneficial. Further investigation of an hybrid between FAS and standard Newton would also be of interest, where one alternate between the two depending on the state and strength of the nonlinearity of the simulation. Another consideration is systematic benchmarking of different choices for interpolation, re-

striction, smoothers, and other solver components.

A challenge in working with multigrid methods is finding a proper number of smoothing steps. This is often not directly addressed in articles on multigrid methods other than stating what where applied. Since it is critical that the error is sufficiently smooth, it would be interesting to look further into optimizing the number of smoothing steps. In addition one could expand the investigation into other cycle schemes, such as W- and F-cycles. This may further improve the efficiency of the method.

One could also consider implementing parts of the algorithm in a compiled language for a more accurate runtime assessment in a parallel environment. Another consideration could also be to take advantage of the support of parallel computations on GPU's with Matlab.

Bibliography

Aarnes, J. E., Gimse, T., and Lie, K.-A. (2007). An introduction to the numerics of flow in porous media using matlab. In Hasle, G., Lie, K.-A., and Quak, E., editors, *Geometric Modelling, Numerical Simulation, and Optimization: Applied Mathematics at SINTEF*, pages 265–306. Springer, Berlin, Germany.

Alberty, J., Carstensen, C., and Funken, S. (1999). Remarks around 50 lines of matlab: short finite element implementation. *Numerical Algorithms*, 20:117–137.

Arnoldi, W. E. (1951). The principle of minimized iteration in the solution of the matrix eigenvalue problem. *Quareterly of Applied Amthematics*, 9:17–29.

Aziz, K. and Settari, A. (1979). *Petroleum Reservoir Simulation*. Applied Sciences Publishers Ltd, Essex, UK.

Bastian, P., Kraus, J., Scheichl, R., and Wheeler, M. (2013). *Simulation of Flow in Porous Media - Applications in Energy and Environment*. De Gruyter, Berlin, Germany.

Brand, C., Heinemann, J., Leoben, M., and Aziz, K. (1991). The grid orienta-

tion effect in reservoir simulation. In *SPE Reservoir Simulation Symposium*, volume 11, pages 275–286. SPE.

Brandt, A. (1982). Guide to multigrid development. In Hackbusch, W. and Trottenberg, U., editors, *Multigrid Methods*, pages 220–312, Berlin, Heidelberg. Springer Berlin Heidelberg.

Briggs, W., Henson, V., and McCormick, S. (2000). *A Multigrid Tutorial*. SIAM, Philadelphia, USA.

Brookes, R. H. and Corey, A. T. (1964). Hydraulic properties of porous media. *Hydrology Papers*.

Bui, Q., Elman, H., and J.D., M. (2016). Algebraic multigrid preconditioners for multiphase flow in porous media. *SIAM Journal on Scientific and Statistical Computing*, 39(5):662–680.

Bücker, M., Corliss, G., Hovland, P., Naumann, U., and Norris, B. (2006). Automatic differentiation: Applications, theory, and implementations. In *Lecture Notes in Computational Science and Engineering*, volume 50, Berlin, Germany. Springer.

Chen, Z. (2007). *Reservoir Simulation: Mathematical Techniques in Oil Recovery*. SIAM, University of Calgary, Alberta, Canada.

Christensen, M. l. C. (2016). *Multilevel techniques for Reservoir Simulation*. PhD thesis, Technical University of Denmark, Kongens Lyngby, Denmark.

Christensen, M. l. C., Eskildsen, K. L., Engsig-Karup, A. P., and Wakefield, M.

(2016). Nonlinear multigrid for reservoir simulation. *SPE Journal*, 21(3):888–898. Doi: 10.2118/178428-PA.

Christie, M. A. and Blunt, M. (2010). Tenth SPE comparative solution project: A comparison of upscaling techniques. *SPE Reservoir Evaluation & Engineering*, 4:308–217.

Cusini, M., Lukyanov, A., Natvig, J., and Hajibeygi, H. (2014). A constrained pressure residual multiscale (CPR-MS) compositional solver. In *14th European Conference on the Mathematics of Oil Recovery 2014, ECMOR 2014*, Sicily, Italy. SPE.

Darcy, H. (1856). *Les fontaines publiques de la ville de Dijon*. Dalmont, Paris, France.

Demirbas, A. (2009). Global renewable energy projections. *Energy Sources Part B*, 4:212–224.

Euler, L. (1768). *Institutionum calculi integralis*. Iterative Solution of Nonlinear Equations in Several Variables, Paris, France.

Fonseca, R., Geel, C., and Leeuwenburgh, O. (2017). Description of olympus reservoir model for optimization challenge. http://www.isapp2.com/optimization-challenge/reservoir-model-description.html.

Gandham, R., Esler, K., and Zhang, Y. (2014). A GPU accelerated aggregation algebraic multigrid method. *Computers & Mathematics with Applications*, 68 (10):1151–1160.

Gauss, C. (1809). *Theoria motus corporum coelestium in sectionibus conicis solem ambientium*. F. Perthes und I.H. Besser,.

Gauss, C. (1813). Theoria attractionis corporum sphaeroidicorum ellipticorum homogeneorum methodo nova tractata. *Commentationes societatis regiae scientiarium Gottingensis recentiores*, 2:355–378.

Haynsworth, E. (1968). On the schur complement. *Basel Mathematical Notes*, BMN 20.

Henson, V. E. (2003). Multigrid methods for nonlinear problems: An overview. In *Proceedings of SPIE - The International Society for Optical Engineering*, volume 5016, pages 36–48. Doi: 10.1117/12.499473.

Hessenberg, K. (1942). *Die Berechnung der Eigenwerte und Eigenlösungen linearer Gleichungssysteme*. PhD thesis, Technische Hochscule, Darmstadt, Germany.

Jain, K., Cole, J., Kumar, S., Gidwani, A., and Vaidya, N. (2008). A multiphase, two-fluid model for water transport in a pem fuel cell. In *ECS Transactions, Issue 2 PART 1*, volume 16, pages 45–56. Doi: 10.1149/1.2981842.

Krogstad, S., Lie, K.-A., Møyner, O., Nilsen, H. M., Raynaud, X., and Skaflestad, B. (2015). MRST-AD - an open-source framework for rapid prototyping and evaluation of reservoir simulation problems. *SPE Reservoir Simulation Symposium*.

Lie, K.-A. (2016). *An Introduction to Reservoir Simulation Using MATLAB - User Guide for the Matlab Reservoir Simulation Toolbox (MRST)*. SINTEF ICT, Department of Applied Mathematics Oslo, Norway.

Lie, K.-A., Krogstad, S., Ligaarden, I. S., Natvig, J. R., Nilsen, H. M., and Skaflestad, B. (2012). Open-source MATLAB implementation of consistent discretisations on complex grids. *Computational Geosciences*, 16 (2):297–322.

Molenaar, J. (1995). Multigrid methods for fully implicit oil reservoir simulation. In *Proceedings Copper Mountain Conference on Multigrid Methods*, volume Report 95-40, pages 581–590. Delft University of Technology.

Mollick, E. (2006). Establishing moore's law. *IEEE Annals of the History of Computing*, 28(3):62–75.

Moore, G. (1965). Cramming more components onto integrated circuits. *Electronics,*, 38(8):114–117.

Muskat, M. and Wyckoff, R. D. (1937). *The flow of homogeneous fluids through porous media*. McGraw-Hill Book Company, New York, USA.

Newton, I. (1687). *Philosophiae Naturalis Principia Mathematica*. Imprimatur, London, England.

Nocedal, J. (2006). *Numerical Optimization*. Springer, New York, USA.

Notay, Y. (2010). An aggregation-based algebraic multigrid method. *Electronic Transactions On Numerical Analysis*, 37:123–146.

Ortega, J. and Rheinboldt, W. (2000). *Institutionum calculi integralis*, chapter 7. SIAM, Pennsylvania, USA.

Peaceman, D. (1977). *Fundamentals of Numerical Reservoir Simulation*. Elsevier Scientific Pub. Co., Texas, USA.

Ruge, J. and Stüben, K. (1987). Algebraic multigrid. In *Multigrid Methods*, chapter 4, pages 73–130. SIAM, Pennsylvania, USA.

Saad, Y. (2003). *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, USA.

Saad, Y. and Schultz, M. (1986). A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):494–526.

Schäfer, M. (2006). *Computational Engineering - Introduction to numerical Methods*. Springer, Berlin, Germany.

Seternes, G. (2015). Simulations of CO2 migration with a fully-integrated VE model on the GPU. Master's thesis, Norwegian University of Science and Technology (NTNU).

Sintef (2018). MRST - MATLAB reservoir simulation toolbox. Retrieved May, 2018, from https://www.sintef.no/mrst.

Süli, E. and Mayers, D. (2003). *An introduction to Numerical Analysis*. Cambridge University Press, New York, USA.

Toft, R. (2017). Two-phase reservoir simulation with the full approximation scheme. Technical report, Norwegian University of Science and Technology (NTNU).

Toft, R., Lie, K.-A., and Moyner, O. (2018). Full approximation scheme for reservoir simulation. In *Norsk Informatikkonferanse (NIK) 2018*.

Toronyi, R. and Ali, S. (1974). Determining interblock transmissibility in reservoir simulators. *Journal of Petroleum Technology*, 26:77–78.

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

Trefethen, L. N. and Bau, D. (1997). *Numerical Linear Algebra*. SIAM, Philadelphia, USA.

Trottenberg, U., Oosterlee, C., and Schuller, A. (2001). *Multigrid*. Academic Press, San Diego, California, USA.

Vinsome, P. (1976). Orthomin, an iterative method for solving sparse sets of simultaneous linear equations. In *Fourth Symposim of Numerical Simulation*, California, USA. SPE.

Von Neumann, J. and Goldstine, H. (1947). Numerical inverting of matrices of high order. *Bulletin of the American Mathematical Society*, 53(11):1021–1099.

Wallis, J. (1983). Incomplete gaussian elimination as a preconditioning for generalized conjugate gradient acceleration. In *SPE Reservoir Simulation Symposium*, volume 27, pages 325 – 334, California, USA. SPE.

Wallis, J. (1985). Incomplete gaussian elimination as a preconditioning for generalized conjugate gradient acceleration. *SPE Reservoir Simulation Symposium*, 20:117–137.

Wallis, J., Kendall, R., and Little, T. (1985). Constrained residual acceleration of

conjugate residual methods. In *SPE Reservoir Simulation Symposium*, volume 37, pages 415–428, Texas, USA. SPE.

Whitaker, S. (1986). Flow in porous media i: A theoretical derivation of darcy's law. *Transport in Porous Media*, 1(1):3–25.

Willhite, G. (1986). *Waterflooding*. SPE, Texas, USA.

Zhang, F. (2005). *The Schur Complement and Its Applications*. Springer, Boston, USA.

Zou, C., Zhao, Q. Zhang, G., and Xiong, B. (2006). Energy revolution: From a fossil energy era to a new energy era. *Natural Gas Industry B*, 3:1–11.