NTNU – Trondheim
Norwegian University of
Science and Technology

Numerical Aspects of Flow Based Local Upscaling

Lars Vingli Odsæter

Master of Science in Physics and Mathematics
Submission date: Januar 2013
Supervisor: Helge Holden, MATH
Co-supervisor: Alf Birger Rustad, Statoil

Norwegian University of Science and Technology
Department of Mathematical Sciences

Problem Description

Reservoir simulation is a key technology for increasing hydrocarbon recovery from offshore oil and gas fields. Current full field simulation models are typically limited to a grid resolution around hundred meters laterally and five meters vertically. This implies that important reservoir features will not be resolved in a full field model, and hence needs to be accounted for in models with finer resolution. The most promising route to handle this problem is by homogenization. Homogenization is a large field of applied mathematics in its own right, and presents one of the deepest touching points between mathematics and petroleum technology.

The elliptic diffusion equation is the most fundamental differential equation for flow in porous media as it governs single phase flow. Flow based local upscaling techniques[1] has been a natural choice for handling fine scale heterogeneities. However, models representing fine layering of geology has proven to be numerically challenging. Currently there are three different approaches for handling boundary conditions for local methods in industry usage, namely fixed, linear and periodic. Their properties with respect to convergence or correctness is not well understood. The main task here is to test numerically the three leading choices for boundary conditions, both in terms of convergence and in terms of correctness.

[1] Durlofsky, L. J. Upscaling and Gridding of Fine Scale Geological Models for Flow Simulation. In *Proceedings of the 8th International Forum on Reservoir Simulation, Stresa, Italy* (June 20–24, 2005), Stanford University.

Abstract

We start this thesis by giving an introduction to reservoir simulation and upscaling in particular. The most common upscaling techniques, including power averaging methods and flow based local and global methods [10], are introduced. Hybrid methods and multiscale methods are also included, and we consider both single- and two-phase systems. Upscaling is viewed in the context of a representative elementary volume (REV), and we argue why flow based local methods can be preferable for this purpose. The elliptic diffusion equation is central in flow based upscaling methods as it governs single-phase flow. We present numerical methods to solve it numerically on corner-point grids, which are the industry standard grids. The newly developed mimetic finite difference method (MFDM) [8] has shown to work nicely on such grids [3], and the MFDM is explained in some details in this thesis. For flow based local upscaling methods, three sets of boundary conditions (BCs) are used by the industry, namely fixed, linear and periodic. We give a detailed analysis of the numerical implications of the three sets of BCs, and we discuss correctness and numerical convergence of these. Results from numerical computations on realistic reservoir models show that linear BCs are considerably faster to solve for. Based on this and that periodic BCs seems intuitively most correct, we propose a new representation and implementation of periodic BCs. This approach is based on mortar methods. Numerical calculations show, however, that this method fails both in terms of correctness and in terms of numerical convergence.

Sammendrag

Vi starter denne oppgaven med å gi en introduksjon til reservoarsimularing og oppskalering spesielt. De mest utbredte oppskaleringsteknikkene, inkludert gjennomsnittsmetoder og flytbaserte lokale og globale metoder [10], blir introdusert. Hybride metoder og flerskala metoder er også inkludert, og vi studerer både en- og to-fase systemer. Oppskalering er betraktet i sammenheng med et representativt elementært volum (REV), og vi argumenterer for at flytbaserte lokale metoder kan være å foretrekke i denne sammenhengen. Den elliptiske diffusjonsligningen er sentral i flytbaserte metoder ettersom den beskriver en-fase flyt. Vi presenterer numeriske metoder for å løse den numerisk på hjørnepunktsgrid, som er industriens standard grid. Den nylig utviklede mimetiske endelige differansemetoden (MFDM) [8] har vist seg å fungere godt på slike grid [3], og MFDM er grundig forklart i denne oppgaven. For flytbasert lokal oppskalering er tre sett med randbetingelser brukt av industrien, nemlig faste, lineære og periodiske. Vi gir en detaljert analyse av de numeriske konsekvensene for de tre settene med randbetingelser, og vi diskuterer korrekthet og numerisk konvergens for disse. Resultater fra numeriske beregninger på realistiske reservoarmodeller viser at lineære randbetingelser er betydelig raskere å løse for. Basert på dette og at periodiske randbetingelser intuitivt virker mest korrekt, foreslår vi en ny representasjon og implementasjon av periodiske randbetingelser. Denne tilnærmingen er basert på mortar metoder. Numeriske beregninger viser imidlertid at denne metoden feiler både med tanke på korrekthet og med tanke på numerisk konvergens.

Contents

Preface

This thesis is my completion of a Master's degree in the field of Industrial Mathematics at NTNU, Department of Mathematical Sciences. The thesis is done in cooperation with Statoil.

The Open Porous Media (OPM) project has been used extensively in this thesis. It has been a pleasure using it, as it gives direct access to many years of development. This has given me the opportunity to work with realistic reservoir problems that the industry needs, rather than starting from scratch. I am grateful that both OPM and DUNE are open source, so that I freely can use it and share my contributions to academia. Further, Statoil has provided me realistic reservoir models, but with no strings attached, and hence free for anyone to use. Thus, it should be possible to reproduce the results of this thesis.

I would like to thank my supervisors, Prof. Helge Holden, Department of Mathematical Sciences, NTNU and Dr. Alf Birger Rustad, Statoil. Both of you have shown great engagement to my work and also been available on a short notice, although I know you are busy. A special thanks to Alf for introducing me to the problem and for good discussions on the contents of this thesis.

The list of contributors to this thesis is not limited to my supervisors. Thanks to Dr. Arne Morten Kvarving, SINTEF ICT, for helping me understand and implement the mortar method. Further, I would like to thank Prof. Knut-Andreas Lie, Dr. Atgeirr Flø Rasmussen and Dr. Bård Skaflestad at SINTEF ICT in Oslo, for their welcoming during my visit in September 2012. The help from Atgeirr and Bård has been important for my understanding of OPM. Also thanks to the Department of Mathematical Sciences at NTNU for financing the visit to SINTEF ICT in Oslo and the attendance at the DUNE course in Heidelberg in March 2012. I would also thank Dr. Markus Blatt, HPC-Simulation-Software & Services, for kindly answering my e-mails with questions related to the numerical linear solvers provided by DUNE. Further, thanks Dr. Carl Fredrik Berg and the rest of the PLP department at Statoil for all help. Lastly, I would like to thank Espen Birger Raknes for proofreading this thesis.

Lars Vingli Odsæter Trondheim, January 31, 2013.

Chapter 1

Introduction

The aim of reservoir simulation is to use numerical methods to describe the fluid flow in porous media. Knowledge of the flow pattern is important for optimizing the oil or gas recovery. Permeability is an important parameter in this context, as it describes how well the fluids flow through the material. Reservoirs can be very big, so the discretization cells in a simulation model can be large, typically 100 meters in the horizontal directions and 5 meters in the vertical direction. These cells are too big to capture small scale heterogeneities, which may be important for the global flow pattern. Small scale heterogeneities are captured in models with finer resolution. It is therefore of great value to find the effective permeability in each simulation cell as if the cells were homogeneous pieces. This issue is referred to as permeability upscaling and is a large field of study.

Many different upscaling techniques has been derived. Some are based on different averaging methods, while others are flow based. In this thesis we give an overview of the most common upscaling methods. The notion of a representative elementary volume (REV) will be central in our presentation. A REV denotes a volume of the property field that is large enough to capture a representative amount of heterogeneity [6]. We will explain the notion of a REV more thoroughly in the next chapter.

Mathematically, flow based upscaling methods result in solving the elliptic diffusion equation, which governs single-phase flow. Reservoir properties often vary on a large scale and the underlying fine grid models may include complex geometry. Thus, solving the elliptic diffusion equation numerically is non-trivial. The multi-point flux approximation (MPFA), see for instance [4], is a discretization method that is commonly used in this context. Another promising numerical method is the newly developed mimetic finite difference method (MFDM) [8].

There are two main types of flow based methods. *Local* upscaling methods solve flow problems over the domain corresponding to the simulation cell that is to be upscaled for, whereas *global* upscaling methods try to take into account global

flow effects by solving the flow problem globally [10]. In between these, several hybrid methods have been developed. In this thesis we argue why local methods can be preferable in a REV framework, and thereafter we choose to consider local methods only.

For local upscaling methods, the elliptic diffusion equation must be accompanied with some boundary conditions (BCs). Traditionally three different sets of BCs have been used by the industry, namely fixed, linear and periodic. It is well known (see for instance [10], [14] and references therein) that these BCs may produce quite different upscaling results. One aim of this thesis is to compare the three sets of BCs in terms of correctness, i.e., which BCs that give the most correct upscaled property. As we will see this depends on the geology of the model.

Another aim of this thesis is to consider numerical implications in terms of convergence for the three sets of BCs. This is not well understood and this issue is barely referred to in the literature. It is observed that the problem with linear BCs are faster to solve [14], but no reasoning is found. In this thesis we give a comprehensive comparison and report both number of iterations for the linear iterative solver, computation time, condition numbers and memory usage. We set these properties in context and discuss and give reasons for the results. Realistic reservoir models are considered.

We will see that periodic BCs are most correct in many cases, but that the numerical convergence is slow, at least compared to linear BCs. Thus, the representation of periodic BCs has been investigated to better understand this. A new representation based on mortar methods is derived and tested to see if it can improve the numerical convergence and reduce the memory consumption. A recent study[1] used this approach on a similar problem related to upscaling of elastic parameters. This was applied to a finite element method. In this thesis we apply the approach to the elliptic diffusion equation and with the MFDM as the underlying discretization method.

This thesis is outlined as follows. In Chapter 2 we give a basic introduction to reservoir simulation. Important reservoir properties will be presented, we introduce the central notion of a REV and the governing flow equations in a porous media reservoir are described. Next, corner-point grids, which are the industry standard grids, and traditional discretization methods will be introduced.

Chapter 3 gives a review of the most common methods for upscaling of permeability and relative permeability. Local upscaling methods will be given an extra attention. The presentation is mostly based on [10]. The chapter ends in a discussion of the advantages and disadvantages of the different upscaling methods.

In Chapter 4 we explain in some details the MFDM. This method is used

[1]This study is not yet published. Thanks to Arne Morten Kvarving, Trond Kvamsdal and Runar Holdahl, SINTEF ICT, for letting their work be at our disposal.

to solve the elliptic diffusion equation and has shown to work nicely on corner-point grids [3]. We give no proofs in this thesis, but refer to the literature. The presentation given is primarily based on [16] and [9].

The problem of representing and implementing periodic BCs is discussed in Chapter 5. We present three different approaches. The latter is based on the mortar method, and it is described how it can be implemented and applied to the MFDM.

In Chapter 6 we present numerical results from applying the local upscaling methods on both simple synthetic models and realistic reservoir models. Both single- and two-phase systems are considered. The main goal is to compare the different BCs with respect to correctness and numerical convergence. Such a detailed documentation on numerical implications of the three sets of BCs is to our knowledge not found elsewhere in the literature. Lastly, the implementation of the mortar method is tested.

This thesis ends with some concluding remarks in Chapter 7, and some suggestions for further work are listed.

For all numerical computations we use an open source simulation toolbox provided by the Open Porous Media (OPM) project. OPM is built on DUNE (the Distrubuted and Unified Numerics Environment), which is a software toolbox for solving partial differential equations (PDEs) numerically. A short description of OPM and DUNE and instructions on how to use them are given in Appendix A. All implementation and models used in this thesis are open for anyone to use, and in Appendix A it is explained where this can be found and how to use it. Thus, it should possible for the reader to reproduce the results of this thesis.

Chapter 2

Reservoir Simulation

A reservoir is built up of different types of porous media with hydrocarbons, e.g. oil and gas, and water situated in the pores. The fraction of the void pore volume and the total volume is called the porosity and determines how much fluid that is present in the reservoir. Further, the saturation of a phase (e.g., oil, water or gas) is the fraction of the pore volume that contains the current phase. Another important reservoir parameter is permeability, which is a measure of the ability of a material to allow fluid to pass through it. Often, a reservoir is built up of many different layers as we see an example of in Figure 2.1. The layers are typically alternating and parallel to each other. The rock properties may vary on a large scale between the different layers. Materials with low permeability, e.g. mud, are often referred to as low-permeability materials, while materials with high permeability are referred to as high-permeability materials. It is not an objective of this thesis to give a detailed background on porous media theory. For an introduction, the reader could consult for instance [1] or references therein.

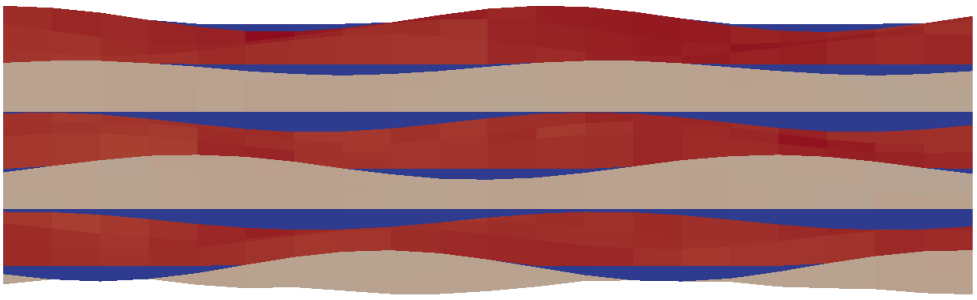


Figure 2.1: Example of a section of a reservoir with three different materials.

In reservoir simulation it is common to refer to many different length scales. The smallest scale is the pore scale ($\sim 10^{-3}$ m), and this is where the actual flow

takes place. Pressure differences and gravitational forces drive the fluid flow in a pore, and the fluid velocity, $\vec{v}$, can be assumed do obey Darcy's law,

$$\vec{v} = -\frac{k}{\mu}\nabla(p - \rho\vec{g}), \tag{2.1}$$

where k is the permeability, μ is the fluid viscosity, p is the pressure, ρ is the fluid density and $\vec{g}$ is the gravitational constant.

The next scale is the core scale ($\sim 10^{-2}$ m). Often, rock samples on core scale from real reservoirs are used to measure the properties of different rocks in a laboratory. The information from both pore and core scales are used to build geological models. A geological model is a three-dimensional representation of a reservoir, and is built up of cells with constant porosity and permeability. The permeability is now represented by a tensor, where the diagonal terms represent flow in the direction caused by a pressure drop, and the off-diagonal terms represent flow perpendicular to the pressure drop. We say that a model is isotropic if the permeability is equal in all directions. In this case, the permeability can be expressed as a scalar. Notice that on pore scale — see equation (2.1) — permeability is treated as a scalar. This is because the direction is given by the orientation of the pore. The size of a cell in a geological model is typically 10–50 m in the horizontal directions and 0.1–1 m in the vertical direction. A collection of geological cells forms a simulation cell, whose size is typically 100 m in the horizontal directions and 5 m in the vertical direction.

2.1 Representative Elementary Volume

The notion of a representative elementary volume (REV) is important in reservoir simulation, especially in an upscaling point of view. A REV denotes a volume of the property field that is large enough to capture a representative amount of the heterogeneity [6]. In Figure 2.2, we see how a measured property can vary with the sample volume. When the volumes are small they capture little of the heterogeneity, so that a measurement is unstable with respect to the volume size and location. This means that a small change of volume can result in a large change in the measured property. As the volume gets larger it captures more heterogeneity and small scale variations become less dominant. Thus, the measured property stabilizes, and at some point a REV is identified. An *appropriate volume* is a volume where the measured property is relatively insensitive to small changes in volume and location [19].

In a reservoir it is common to identify several different REVs. Figure 2.3 shows how three different REVs are identified at different approximate length scales. At the smallest volumes the measured property is dependent on whether a pore or

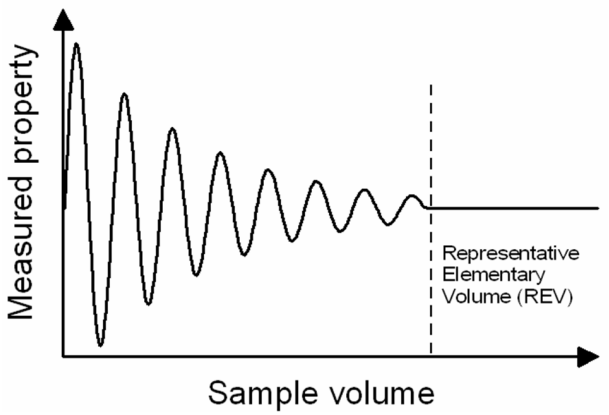


Figure 2.2: Schematic graph of how a measured property varies with sample volume. The domain of the REV is also shown. The figure is taken with permission from [19].

a grain is measured. As the volume increase, the fraction of pores stabilizes and eventually we reach the first REV, the lamina REV. With a further increase in the measurement volume, the measured property starts to oscillate again. This is because different types of laminae are now captured by the volume. When a sufficient amount of the different laminae are captured by the volume, we encounter a new REV, the lithofacies REV. The same pattern repeats again — we get new oscillations and we finally reach a new REV, the facies association REV. Figure 2.4 gives a nice overview of different geological heterogeneity types that can be identified in a reservoir. In general the number and types of REVs are not restricted to the three mentioned here.

Identifying REVs is important because we know that the measurements are representative for the whole material structure under consideration. Measurements that do not coincide with a REV should be rescaled to a REV for consistent flow simulation [19]. However, identifying REVs is a difficult task, as they are different from reservoir to reservoir and also dependent on which property that is measured.

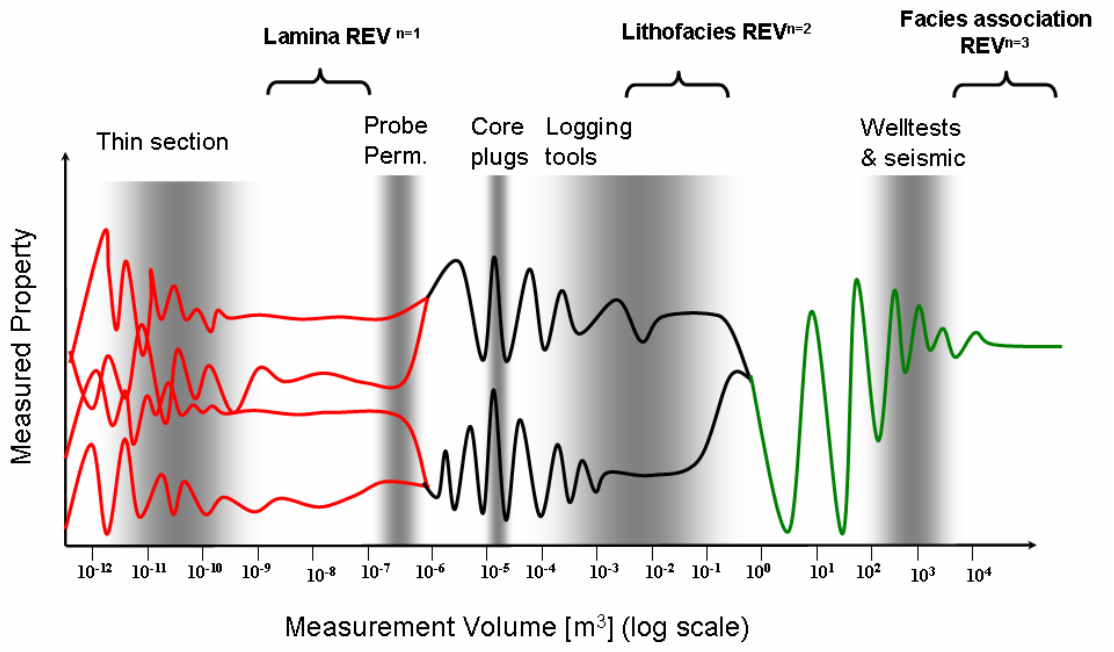


Figure 2.3: A conceptual sketch of different REVs that may occur in a reservoir. In this illustration there are four different lamina types that combine into two lithofacies which again combine in one facies association. It is important to notice that this figure illustrates an example of one particular reservoir type and that the REVs may be different for other reservoirs. The figure is taken with permission from [19].

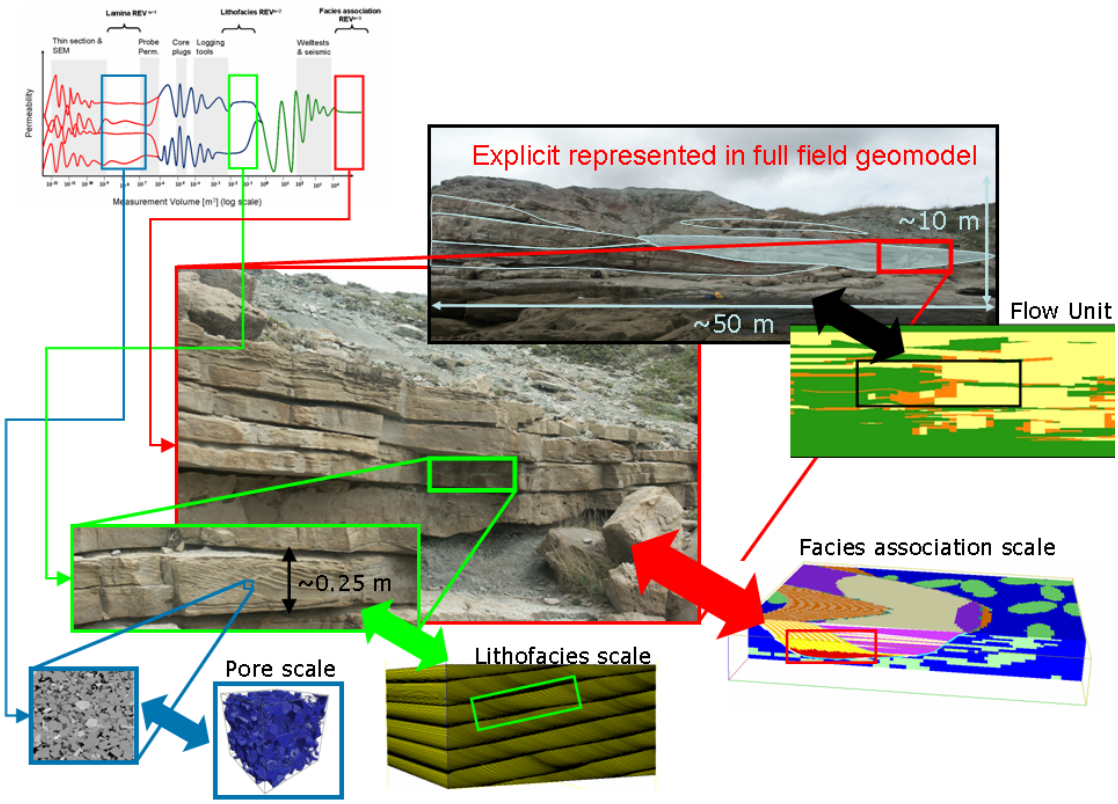


Figure 2.4: Overview of different geological heterogeneity types and the relation to the REV framework for a fluvial reservoir. The illustration is credited Kjetil Nordahl, Statoil.

2.2 Flow Equations

On larger scales Darcy's law (2.1) extends to

$$\vec{v} = -\frac{\mathbf{K}}{\mu}\nabla(p - \rho\vec{g}), \tag{2.2}$$

where $\mathbf{K}$ is the 3×3 permeability tensor. This is valid if only one phase is present. If we have a multi-phase system with n phases, we need to take into account the interactions between the phases. The phase permeability tensor, $\mathbf{K}_i$, of a phase i, can be expressed as

$$\mathbf{K}_i = \mathbf{K}_{ri}\mathbf{K}, \qquad i = 1, 2, \ldots, n, \tag{2.3}$$

where $\mathbf{K}_{ri}$ is the relative permeability of phase i and depends on the saturation of the other phases. When considering multi-phase systems, the permeability, $\mathbf{K}$, is often referred to as the absolute permeability. In the literature, relative permeability is often described by a scalar. This is the case in for instance [1], [2], [10] and [11]. However, as for absolute permeability, relative permeability is often spatially dependent, and assuming otherwise may be wrong is some cases. In cases where we assume that relative permeability is isotropic, we write k_{ri} for relative permeability. In Figure 2.5 we see typical curves for relative permeability of oil and water as functions of water saturation. Darcy's law for a multi-phase system can be expressed as

$$\vec{v}_i = -\frac{\mathbf{K}_{ri}\mathbf{K}}{\mu_i}\nabla(p_i - \rho_i\vec{g}), \tag{2.4}$$

where subscript i denotes the property for phase i.

The second fundamental equation is conservation of mass,

$$\frac{\partial}{\partial t}(\phi\rho_i S_i) + \nabla \cdot (\rho_i \vec{v}_i) = q_i, \tag{2.5}$$

where t is time, ϕ is the porosity, S_i is the saturation of phase i and q_i is a source term. If we assume that the fluids and the rocks are incompressible and insert Darcy's law (2.4) into (2.5), we get the full flow equation,

$$\phi\frac{\partial S_i}{\partial t} - \nabla \cdot \left[\frac{\mathbf{K}_{ri}\mathbf{K}}{\mu_i}(\nabla p_i - \rho_i\vec{g})\right] = \frac{q_i}{\rho_i}. \tag{2.6}$$

I addition, the sum of the saturations must be equal to one, $\sum_{i=1}^{n} S_i = 1$. Observe, that for a single-phase system, (2.6) reduces to

$$-\nabla \cdot \left[\frac{\mathbf{K}}{\mu}(\nabla p - \rho\vec{g})\right] = \frac{q}{\rho}. \tag{2.7}$$

Notice that fluid flow in a single-phase system is not time dependent when we assume incompressibility. In the theory of partial differential equations, (2.7) is classified as elliptic when $\mathbf{K}$ is symmetric positive definite (SPD) [12, p. 312–313].

We will now investigate equation (2.6) further for a two-phase system consisting of oil(o) and water(w). This will enlighten which forces that act on the fluids in a reservoir. We assume isotropic relative permeability. First, we need to introduce capillary pressure,

$$p_c = p_o - p_w,$$

i.e., the pressure difference between the two phases. Further, denote by $\lambda_i = \frac{k_{ri}}{\mu_i}$ the mobility of phase i $(i = o, w)$, and by $\lambda_t = \lambda_w + \lambda_o$ the total mobility. Following [2], one can introduce a global pressure p, and a total velocity $\vec{v}_t = \vec{v}_w + \vec{v}_o$. Oil saturation, S_o, can be eliminated from the two transport equations (2.6) via the relationship $S_w + S_o = 1$, and we are left with the system

$$\nabla \cdot \vec{v}_t = q_t, \quad \text{where } \vec{v}_t = -\mathbf{K}\left[\lambda_t \nabla p - (\lambda_w \rho_w + \lambda_o \rho_o)\vec{g}\right], \tag{2.8}$$

$$\phi \frac{\partial S_w}{\partial t} + \nabla \cdot \left[f_w(\vec{v}_t + \mathbf{K}\lambda_o \nabla p_c + \mathbf{K}\lambda_o(\rho_w - \rho_o)\vec{g})\right] = \frac{q_w}{\rho_w}, \tag{2.9}$$

where $f_w = \frac{\lambda_w}{\lambda_t}$ is the fractional flow of water, and $q_t = \frac{q_w}{\rho_w} + \frac{q_o}{\rho_o}$ is the total source scaled by density. Equation (2.9) includes three terms that represent the three types of forces that act on the fluids [2]:

1. *Viscous forces*, represented by the term $f_w \vec{v}_t$.
2. *Capillary forces*, represented by the term $f_w \mathbf{K}\lambda_o \nabla p_c$.
3. *Gravitational forces*, represented by the term $f_w \mathbf{K}\lambda_o(\rho_w - \rho_o)\vec{g}$.

Viscous forces stems from pressure gradients and are due to the inertia and viscosity of the fluids. Capillary forces are due to the capillary pressure. If you put the bottom of a sugar cube in a cup of coffee, it is capillary forces that make the coffee penetrate the cube. The balance of these forces vary a lot and depends particularly on model size, flow-rate, heterogeneities and which fluids that are present [2]. On small scales or in regions with low flow rate, capillary forces tends to dominate the viscous forces and vica verca. Because of larger difference in density, gravitational forces are more dominant in oil-gas or water-gas systems than in oil-water systems. For vertically thin reservoirs, gravity is less dominant.

Capillary pressure is dependent on the saturation, but also on the process due to hysteresis[1]. Capillary pressure as a function of saturation is strictly monotone.

[1]Hysteresis means that a quantity is not only dependent on the current state, but also on its past. For a oil-water system, the capillary pressure have different curves for imbibition (water displaces oil) and drainage (oil displaces water).

A typical curve for capillary pressure can be seen in Figure 2.5. In the oil industry it is common to use a dimensionless scaling of the capillary pressure called the Leverett J-function (hereafter only denoted J-function),

$$J(S_w) = \frac{p_c(S_w)\sqrt{\frac{k}{\phi}}}{\sigma\cos(\theta)}, \tag{2.10}$$

where σ is the surface tension and θ is the contact angle between the phases. It is only possible to use the J-function when the model is isotropic, since permeability, k, is represented by a scalar.

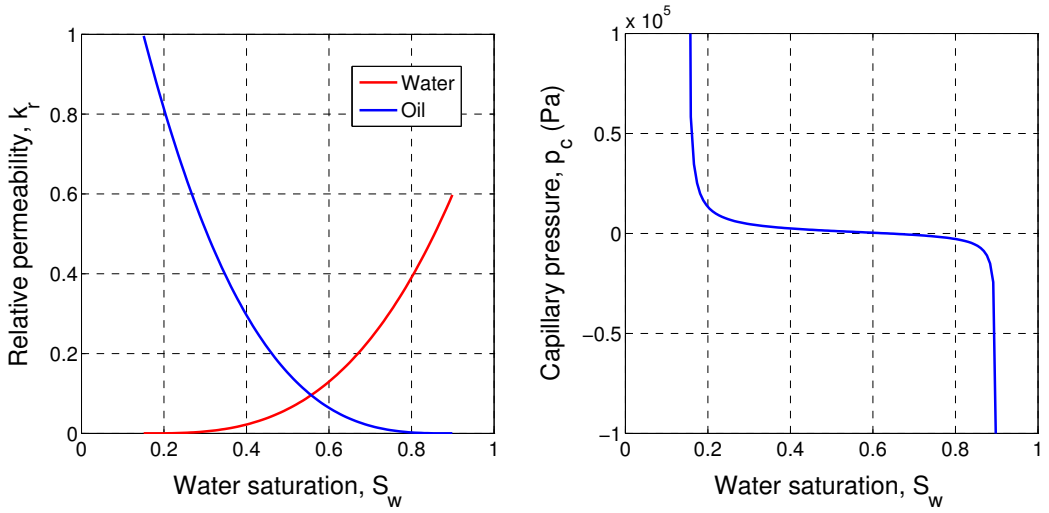


Figure 2.5: Typical curves for relative permeability, k_r, and capillary pressure, p_c, as functions of water saturation, S_w, in an oil-water system.

2.3 Corner-Point Grids

Corner-point grids [20] are commonly used by the oil industry. A corner-point grid is built up of several vertical or inclined pillars, which are straight lines defined by two points each. A constant number of points are defined on each pillar, and the hexahedron defined by four neighbouring pillars and two neighbouring points on each pillar, makes up a grid cell [23], see Figure 2.6. The points on the pillars are called corner points and are allowed to coincide, which result in degenerated cells with less than eight corners. Some cells may also disappear completely, and this introduce connections between cells that are not seen as neighbours. Another feature of corner-point grids is that they easily allow for discontinuities across faces. This can be used to include fractures and faults in the model. A fracture is

a crack or a surface of breakage, across which the rocks have not been displaced, while a fault is a planar surface in the rocks, across which the rocks have been displaced [2]. Introducing fractures and faults results in non-conforming grids. Conforming grids are grids where the intersection of two arbitrary cells is either empty, a face, an edge or a vertex. With these features it is possible to make complex geological models.

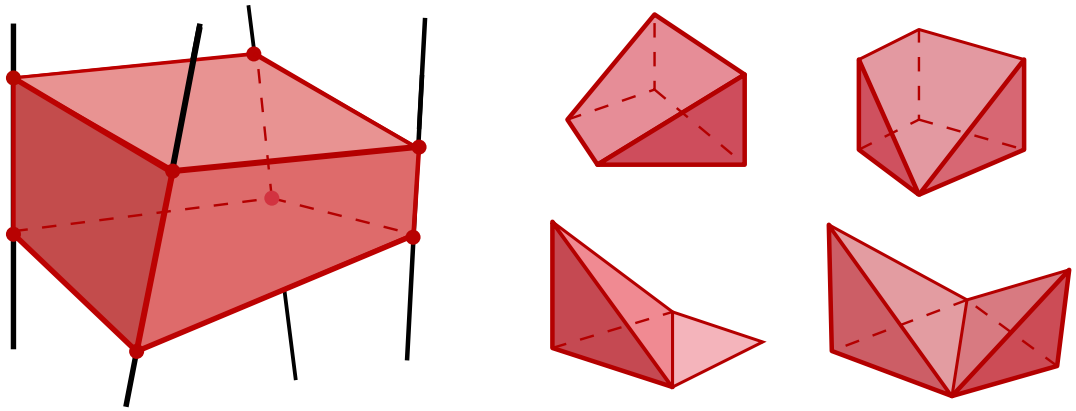


Figure 2.6: To the left, a general corner-point cell is shown. The black pillars and the red points define the corner-point cell in light red. To the right we see four examples of degenerated cells.

The flexible grid geometry introduces some additional difficulties [3]. First, since a grid face is defined by four points, the faces will generally be bilinear and possibly strongly curved. We also notice that corner-point grids are not uniquely defined as a surface defined by four points is not unique. Next, one needs to handle cells with zero volume, and the resulting non-neighbour connections, which may result in discretization matrices with a complex sparsity pattern. Further, the presence of degenerated cells calls for a very flexible discretization that is not sensitive to the geometry of each cell or to the number of faces and corner points. The discretization method should also be able to handle non-conforming grids, where the intersection of two connected cells not necessarily coincides with the two touching cell faces, as it does for conforming grids. In addition, the variation in the reservoir properties are typically much higher in the vertical direction. This results in corner-point cells that are thin compared to their horizontal propagation. Thus the aspect ratio will be high.

To this end, we will distinguish between faces and interfaces. An interface will be referred to as an intersection between two neighboring cells, whereas a face is related to one specific cell. In a conforming grid, they will coincide, but the total number of faces in a grid will be twice that of interfaces plus boundary faces.

In the literature, it is often referred to K-orthogonal grids. Assume that a cell has s faces, and let $\vec{n}_k$ and $\vec{c}_k$, for $k = 1, \ldots, s$, be the unit normal vector of face k and the vector from the cell centroid to the kth face centroid respectively. Then, the grid is said to be K-orthogonal [17] if for all cells,

$$\vec{c}_k^{\top} \mathbf{K} \vec{n}_k = 0, \qquad \text{for } k = 1, \ldots, s.$$

2.4 Discretization Methods

We now return to the elliptic pressure equation (2.7) and present some commonly used discretization methods. Notice that the steady-state equation of the multiphase problem (2.6) is on the exact same form. We state this problem as a system of two equations,

$$\vec{v} = -\frac{\mathbf{K}}{\mu} \nabla p, \qquad \text{on } \Omega, \tag{2.11a}$$

$$\nabla \cdot \vec{v} = b, \qquad \text{on } \Omega. \tag{2.11b}$$

Here $b = b(\vec{x})$ is the source term and Ω is the computational domain with boundary $\partial\Omega$. For simplicity, we have neglected gravity and we assume homogeneous Dirichlet boundary conditions, i.e., $p|_{\partial\Omega} = 0$, where $p|_{\partial\Omega}$ means p evaluated on $\partial\Omega$. Let Ω be discretized into a corner-point grid with N cells, N_i interfaces and N_f faces. Generally we require that our methods are mass conservative in each cell, E, and to be exact for linear pressure.

Let p_E be the pressure at the cell centroid. Further, let s_E be the number of faces of cell E, and denote by π_E^k and v_E^k, for $k = 1, \ldots, s_E$, the pressure and the flux respectively at face k of cell E. Flux is defined as the normal component of the velocity, $\vec{v} \cdot \vec{n}$. General finite volume methods can now be written in the form [17]

$$\mathbf{v}_E = \mathbf{T}_E(\mathbf{e}_E p_E - \boldsymbol{\pi}_E), \quad \mathbf{e}_E = [1, \ldots, 1]^{\top}, \tag{2.12}$$

where $\mathbf{T}_E$ is a $s_E \times s_E$ matrix, $\mathbf{v}_E = [v_E^1, \ldots, v_E^{s_E}]^{\top}$ and $\boldsymbol{\pi}_E = [\pi_E^1, \ldots, \pi_E^{s_E}]^{\top}$. This is the discrete analogue to Darcy's law (2.11a). Different choices of $\mathbf{T}_E$ leads to different methods.

Mass conservation (2.11b), can be stated as $\sum_{k=1}^{s_E} v_E^k = b_E$. We also require the flux to be continuous across cell interfaces, i.e., $v_{E_1}^{k_1} = -v_{E_2}^{k_2}$ if cell E_1 shares is k_1th face with face k_2 of cell E_2. Together with (2.12), we get the following system of equations [17],

$$\begin{bmatrix} \mathbf{M} & \mathbf{C}^\top & \mathbf{D}^\top \\ \mathbf{C} & \mathbf{0} & \mathbf{0} \\ \mathbf{D} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ -\mathbf{p} \\ \boldsymbol{\pi} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{b} \\ \mathbf{0} \end{bmatrix}, \tag{2.13}$$

where $\mathbf{v}$, $\mathbf{p}$ and $\boldsymbol{\pi}$ are the global vectors of unknown fluxes, cell pressures and interface pressures respectively. We have one flux unknown for each face, and one interface pressure unknown for each interface. Further, $\mathbf{M}$ is a block diagonal matrix with blocks $\mathbf{T}_E^{-1}$, $\mathbf{C}$ is a $N \times N_f$ matrix, where each row corresponds to a cell having ones at the positions corresponding to the cells faces, and $\mathbf{D}$ is a $N_i \times N_f$ matrix, where each row corresponds to an interface having ones at the positions corresponding to the two cell faces. Finally, $\mathbf{b}$ is a vector of cell sources.

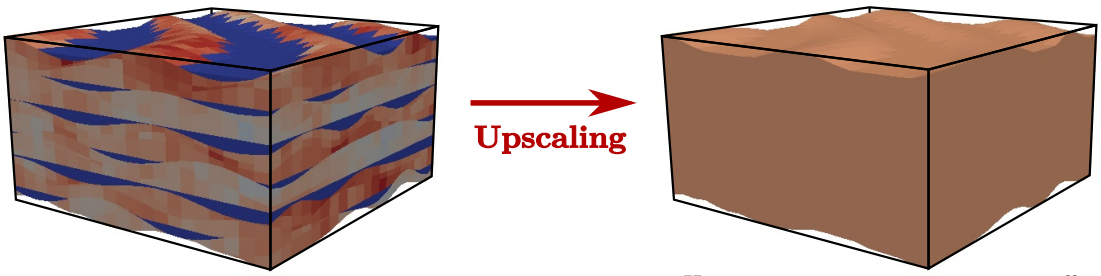
The two-point flux approximation (TPFA), see for instance [1], have two flux unknowns per interface and $\mathbf{T}_E$ is diagonal [17]. TPFA requires the grid to be K-orthogonal, which in general is not satisfied for corner-point grids. The multi-point flux approximation (MPFA), see for instance [4] or [1], have more unknowns at the interfaces and can be applied to more general grids. However, it is hard to implement on corner-point grids, especially if the grid is non-conforming [3]. The newly developed mimetic finite difference method (MFDM) [8] is quite flexible and easy to use with corner-point grids as it handles degenerated cells and non-conforming grids [3]. We will go further in details of the MFDM in Chapter 4.

For completeness we also mention that mixed finite element methods (MFEM) can be used to solve (2.11). This will also result in a system equal to (2.13), but for degenerated cells, the MFEM is not trivial to use [3].

Chapter 3

Upscaling of Permeability and Relative Permeability

Upscaling of a reservoir property refers to the technique of determining the effective property value of a heterogeneous bulk as if the bulk was homogeneous, see Figure 3.1. For porosity and saturation, a simple volume weighted average is commonly used as the upscaling technique. For permeability it is often necessary to take fluid flow into account in order to capture most of the heterogeneity in a simulation cell. We will assume that we have a fine scale model with small scale heterogeneity and a coarse scale model (typically the simulation model). Further, let $\mathbf{K}(\vec{x})$ be the permeability field and $\phi(\vec{x})$ the porosity in the fine scale model.



Fine grid model, possibly isotropic cells.

Homogeneous, anisotropic coarse cell.

Figure 3.1: Illustration of the upscaling process.

The goal of permeability upscaling is to calculate an upscaled permeability tensor, $\tilde{\mathbf{K}}(\vec{x})$, which is constant in each coarse cell and which captures as much of the fine scale variations as possible. The reason for doing so is that the number of geological cells in a full reservoir typically is so large that running full field simulations are very demanding and in some cases impossible with today's numerical methods and computer power. Anyway, running on a coarser grid will reduce

the computational costs considerably. It is also a goal that the cell that is to be upscaled for is a representative elementary volume (REV), so that it can be used several places in the simulation model.

A large class of different upscaling techniques is presented in [10], and these will be introduced and discussed in this chapter. We will initially consider only single-phase upscaling, but we return to two-phase upscaling in Section 3.4. We assume that no wells are present in our models and that the coarse grid cells are non-overlapping unions of underlying fine grid cells. See [10] for a treatment of wells and irregular coarse grid cells.

In [10] flow based gridding techniques are also presented. These are not upscaling methods, but try to construct grids that have high level of grid refinement in regions with high flow rates and coarser descriptions in regions with lower flow rates. The motivation behind this is to reduce the number of cells so that global simulation can be performed sufficiently efficient and still capture important geological features. However, this technique does not fit into the framework of a REV. Flow-based gridding techniques will therefore not be further discussed in this thesis.

Upscaling methods can be divided into two main types. First, we have different averaging methods. These are analytical and easy to calculate. Secondly, we have flow based methods. These calculate upscaled permeability by solving a flow problem. Analytical solutions exists only for very simple geometries, so numerical calculations are required. There are two main types of flow based methods. Local methods only take into account the permeability field of the target cell, while global methods try to include global effects into the upscaling. In between these two, different hybrid methods have been developed. At the end of this chapter, we will make a discussion on which methods are preferable in different scenarios.

Some of the upscaling techniques that will be presented upscales transmissibility rather than permeability. Thus, a short introduction to transmissibility is needed. Transmissibility is a numerical interface quantity which relates the flow from one cell to a neighboring cell. Let i and $i+1$ be two neighboring cells with cell pressures p_i and p_{i+1} respectively, and with a flow rate $q_{i+1/2}$ across the common interface. Then, we write

$$q_{i+1/2} = T_{x,i+1/2}(p_i - p_{i+1}), \tag{3.1}$$

where $T_{x,i+1/2}$ is the transmissibility between the cells in the x-direction. In a two-point flux approximation scheme, $T_{x,i+1/2}$ can be expressed as [10]

$$T_{x,i+1/2} = \frac{2k_{x,i+1/2}A_{i+1/2}}{\Delta x_{i+1} + \Delta x_i},$$

where $A_{i+1/2}$ is the area of the interface, Δx_i is the average cell size in the x-

direction, and $k_{x,i+1/2}$ is the volume weighted harmonic average of the x-component of the permeability in the two cells, i.e.,

$$k_{x,i+1/2} = \frac{(\Delta x_i + \Delta x_{i+1})k_{x,i}k_{x,i+1}}{\Delta x_{i+1}k_{x,i} + \Delta x_i k_{x,i+1}}.$$

Here, $k_{x,i}$ is the x-component of the permeability tensor in cell i. Transmissibilities in the other directions are defined similarly. We see that transmissibility and permeability are related.

3.1 Power Averaging Methods

Power averaging methods are analytical, so they do not require numerical calculations. This makes them computationally very efficient. They can be viewed as local methods, as they only depend on the fine scale permeability field in the target cell. These methods assume that the fine scale permeability tensor is diagonal, and also calculates a diagonal upscaled permeability tensor. Hence, off-diagonal terms are neglected.

Let Ω denote the target cell and $|\Omega|$ its volume. Further, let $k_\xi(\vec{x})$ denote the fine scale permeability at position $\vec{x} \in \Omega$ in the ξ-direction and $\tilde{k}_\xi$ the (constant) upscaled permeability in Ω in the ξ-direction. Then each upscaling component can be calculated via

$$\tilde{k}_\xi = \left(\frac{1}{|\Omega|} \int_\Omega (k_\xi)^{w_\xi} \, \mathrm{d}\Omega \right)^{\frac{1}{w_\xi}}, \qquad \text{for } \xi = x, y, z. \tag{3.2}$$

The upscaled permeability tensor is now given as

$$\tilde{\mathbf{K}} = \begin{bmatrix} \tilde{k}_x & 0 & 0 \\ 0 & \tilde{k}_y & 0 \\ 0 & 0 & \tilde{k}_z \end{bmatrix}.$$

The power average exponent, w_ξ, can vary with the direction, so even if we have an isotropic permeability field, the upscaled tensor can be anisotropic. We require that $w_\xi \in [-1, 1]$ and observe that the extremes, $w_\xi = -1$ and $w_\xi = 1$, corresponds to the harmonic and arithmetic averages respectively. In the limit $w_\xi \to 0$, (3.2) becomes

$$\tilde{k}_\xi = \exp\left(\frac{1}{|\Omega|} \int_\Omega \log(k_\xi) \, \mathrm{d}\Omega \right).$$

A great advantage of the power average methods is that they are applicable to coarse cells of any shape. However, in many cases, it is a too simple approach and

neglecting off-diagonal terms can be a poor assumption. For a motivation of these methods, see for instance [2].

3.2 Local Methods

Local upscaling methods consider the elliptic pressure equation (2.7) locally on the target cell,

$$\nabla \cdot [-\mathbf{K}(\vec{x})\nabla(p - \rho\vec{g})] = 0, \qquad \text{on } \Omega. \tag{3.3}$$

The source term is omitted as we assume no wells. Local methods assume that the target cell, Ω, is a hexahedron. Let $\partial\Omega^{\xi,i}$, for $\xi = x, y, z$ and $i = 1, 2$, denote the six boundary faces of Ω as illustrated in Figure 3.2. We first solve (3.3) numerically for p_η, and then calculate $\vec{v}_\eta$ from Darcy's law (2.2). The subscript η denotes the solution with a unit pressure drop imposed in the η-direction. Now the upscaled permeability tensor can be calculated as

$$\tilde{\mathbf{K}} = \begin{bmatrix} \tilde{k}_{xx} & \tilde{k}_{xy} & \tilde{k}_{xz} \\ \tilde{k}_{yx} & \tilde{k}_{yy} & \tilde{k}_{yz} \\ \tilde{k}_{zx} & \tilde{k}_{zy} & \tilde{k}_{zz} \end{bmatrix}, \qquad \text{where} \quad \tilde{k}_{\xi\eta} = Q_\xi^\eta L_\eta, \quad \xi, \eta = x, y, z. \tag{3.4}$$

L_η is the average distance between opposite faces in the η-direction and Q_ξ^η is the net flow in the ξ-direction when a pressure drop is imposed in the η-direction, i.e.,

$$Q_\xi^\eta = \frac{1}{2|\partial\Omega^{\xi,2}|}\int_{\partial\Omega^{\xi,2}} \vec{v}_\eta \cdot \vec{n}\ \mathrm{d}S - \frac{1}{2|\partial\Omega^{\xi,1}|}\int_{\partial\Omega^{\xi,1}} \vec{v}_\eta \cdot \vec{n}\ \mathrm{d}S, \tag{3.5}$$

where $\vec{n}$ is the outward pointing unit normal vector on $\partial\Omega$. We see that (3.3) has to be solved for $\eta = x, y, z$ to get all the entries of $\tilde{K}$.

The local upscaling methods depend on which boundary conditions (BCs) we choose. Three sets of BCs are often used in the upscaling procedure; *fixed*, *linear* and *periodic*. All three BCs have in common that a unit pressure drop is imposed in one direction. We will denote by η the direction for the pressure drop, and by ξ the other directions.

Fixed BCs

Fixed BCs, also known as no-flow or sealed BCs, enforces no flow on the boundaries parallel to the pressure drop, i.e.,

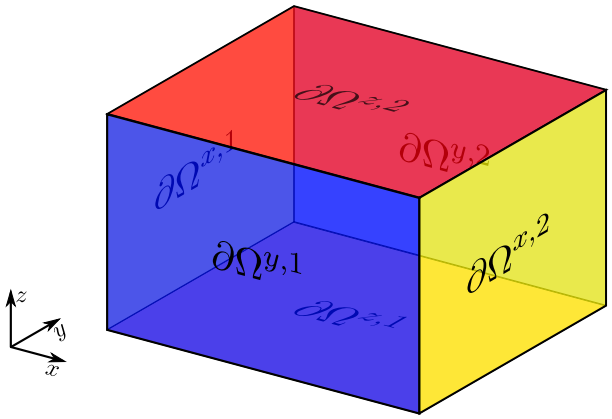


Figure 3.2: The target cell, Ω, and naming conventions for the six boundary faces.

$$\begin{aligned} p_\eta|_{\partial\Omega^{\eta,1}} &= 1, \\ p_\eta|_{\partial\Omega^{\eta,2}} &= 0, \\ (\vec{v}_\eta \cdot \vec{n})|_{\partial\Omega^{\xi,i}} &= 0, \qquad i = 1, 2. \end{aligned}$$

Since no flow is allowed on the boundaries parallel to the pressure drop, we see from (3.5) that the off-diagonal terms in (3.4) will vanish. Thus, cross terms of $\tilde{\mathbf{K}}$ is neglected, and we get a diagonal upscaled tensor. In some cases, for instance when the grid is not K-orthogonal, these cross terms can be significant [10].

One approach for generating a full upscaled tensor from fixed BCs is presented in [10]. Instead of calculating the net flow over the boundaries, as it is done in (3.5), volume weighted averages of the fine scale solutions over the hole domain, Ω, is calculated,

$$\langle \vec{v}_\eta \rangle = \frac{1}{|\Omega|} \int_\Omega \vec{v}_\eta \, \mathrm{d}\Omega, \qquad \eta = x, y, z, \tag{3.6}$$

$$\langle (\nabla p)_\eta \rangle = \frac{1}{|\Omega|} \int_\Omega (\nabla p)_\eta \, \mathrm{d}\Omega, \qquad \eta = x, y, z. \tag{3.7}$$

Each of these two equations have three components, and we denote by $\langle \vec{v}_\eta \rangle_\xi$ the ξ-component of $\langle \vec{v}_\eta \rangle$, and similarly for $\langle (\nabla p)_\eta \rangle$. We can now set up a linear system of nine equations, whose solution gives us the nine components in the upscaled permeability tensor (3.4),

$$\begin{aligned}
\langle \vec{u}_\eta \rangle_x &= -\left(\tilde{k}_{xx} \langle (\nabla p)_\eta \rangle_x + \tilde{k}_{xy} \langle (\nabla p)_\eta \rangle_y + \tilde{k}_{xz} \langle (\nabla p)_\eta \rangle_z \right), \\
\langle \vec{u}_\eta \rangle_y &= -\left(\tilde{k}_{yx} \langle (\nabla p)_\eta \rangle_x + \tilde{k}_{yy} \langle (\nabla p)_\eta \rangle_y + \tilde{k}_{yz} \langle (\nabla p)_\eta \rangle_z \right), \\
\langle \vec{u}_\eta \rangle_z &= -\left(\tilde{k}_{zx} \langle (\nabla p)_\eta \rangle_x + \tilde{k}_{zy} \langle (\nabla p)_\eta \rangle_y + \tilde{k}_{zz} \langle (\nabla p)_\eta \rangle_z \right),
\end{aligned} \tag{3.8}$$

for $\eta = x, y, z$. The upscaled tensor calculated from this linear system will in general be full, but not symmetric. To get a symmetric tensor, one can simply use the average for the cross terms, that is, use $\frac{1}{2}(\tilde{k}_{\eta\xi} + \tilde{k}_{\xi\eta})$ for $\eta \neq \xi$. Alternatively, one can add the three equations

$$\tilde{k}_{\eta\xi} - \tilde{k}_{\xi\eta} = 0, \qquad \eta = x, y, \quad \xi = y, z, \quad \eta \neq \xi, \tag{3.9}$$

to the linear system (3.8) and use the least squares solution as the upscaled tensor.

Linear BCs

Linear BCs specify linearly decreasing pressure on the boundaries parallel to the pressure drop. If the coarse cell is a rectangular hexahedron with side lengths L_x, L_y and L_z, this can be expressed as

$$\begin{aligned}
p_\eta|_{\partial\Omega^{\eta,1}} &= 1, \\
p_\eta|_{\partial\Omega^{\eta,2}} &= 0, \\
p_\eta|_{\partial\Omega^{\xi,i}} &= 1 - \frac{\eta}{L_\eta}, \qquad i = 1, 2,
\end{aligned}$$

We see that flow is allowed out of all boundaries, and hence we get a full tensor from the calculations (3.4) and (3.5). However, the upscaled tensor is in general not symmetric. To obtain a symmetric tensor, the technique given by the equations (3.6) to (3.9) can be used.

Periodic BCs

Periodic BCs connects opposite faces, so that what flows out at one boundary flows in at the opposite boundary, or more precisely,

$$\begin{aligned}
(\vec{v}_\eta \cdot \vec{n})|_{\partial\Omega^{\xi,1}} &= -(\vec{v}_\eta \cdot \vec{n})|_{\partial\Omega^{\xi,2}}, \\
p_\eta|_{\partial\Omega^{\xi,1}} &= p_\eta|_{\partial\Omega^{\xi,2}} + \delta_{\xi,\eta},
\end{aligned} \tag{3.10}$$

where $\delta_{\xi,\eta}$ is the Kronecker delta, i.e., $\delta_{\xi,\eta} = 1$ if $\xi = \eta$ and 0 otherwise. For periodic BCs we must require Ω to be a regular hexahedron, so that opposite boundary

faces can be connected. In addition, the problem given by (3.3) and (3.10), must be closed by specifying the pressure at a given point. This is necessary to ensure a unique solution. The upscaled permeability tensor obtained from (3.4) and (3.5) is symmetric and positive definite [10]. The technique given by the equations (3.6) to (3.9) will result in the same tensor.

Analytical Solutions

For the special case where the model is built up of parallel layers, we can easily derive analytical solutions. We consider a model with different layers alternating in the z-direction, that is all layers are parallel to the xy-plane. Since this model is periodic, which means that the materials matches on opposite boundaries, the upscaled permeability tensor for the fixed and periodic cases should be equal. The upscaled permeability tensor, $\tilde{\mathbf{K}}_{fp}$, for these two cases, can be calculated exact using arithmetic and harmonic averages (3.2),

$$\tilde{\mathbf{K}}_{fp} = \begin{bmatrix} \tilde{k}_{a,x} & 0 & 0 \\ 0 & \tilde{k}_{a,y} & 0 \\ 0 & 0 & \tilde{k}_{h,z} \end{bmatrix}, \tag{3.11}$$

where $\tilde{k}_{a,\xi}$ and $\tilde{k}_{h,\xi}$, for $\xi = x, y, z$, are the arithmetic and harmonic averages in the ξ-direction respectively. The analytical solution can be used to verify that the upscaling procedures are implemented correctly. This result holds for the fixed and periodic BCs only. For linear BCs, an analytical solution is not so easy derived.

A special feature of the upscaled permeability tensor for the periodic case, is that it is SPD, and hence it can be diagonalized. By symmetry arguments, it is possible to calculate the analytical permeability tensor for a periodic model with alternating parallel layers even if the layers are not perpendicular to any of the coordinarte directions. The analytical solution is given by

$$\tilde{\mathbf{K}}_p = \mathbf{A}^\top \tilde{\mathbf{K}}_{\text{diag}} \mathbf{A}, \tag{3.12}$$

where $\mathbf{A}$ is the linear transformation matrix taking the coordinate system $\vec{x}$ onto a coordinate system, $\vec{x}'$, which has two coordinate directions parallel to the layers and the third direction perpendicular to the layers. $\tilde{\mathbf{K}}_{\text{diag}}$ is the analytical solution in the $\vec{x}'$ system, i.e., given by (3.11). We must require $\mathbf{A}$ to be unitary, $\mathbf{A}^\top \mathbf{A} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

Extended Local Methods

Extended local upscaling methods try to capture some of the effect of the surrounding permeability field by including a *border region*. The border region consists of

a ring of neighboring coarse cells with the fine scale permeability field, see Figure 3.3. The problem (3.3) is now solved over this extended domain, accompanied with any of the BCs presented above. Next, one uses (3.6) and (3.7) to calculate $\langle \vec{v}_\eta \rangle$ and $\langle (\nabla p)_\eta \rangle$ over the target cell (the shaded cell in Figure 3.3). Then (3.8) is used to calculate the upscaled tensor.

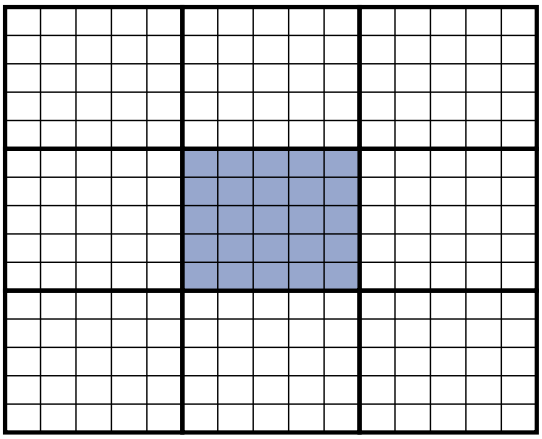


Figure 3.3: The border region and the target cell (shaded). Both fine grid (thin lines) and coarse grid (bold lines) are shown. These grids are two-dimensional and structured for simplicity of the illustration, but extended local methods are valid for more general grids. The figure is modified from [10].

Notice that this method in general will result in a full tensor for all BCs. Also notice that the tensor is no longer symmetric in general for the periodic case as this was lost when introducing the border region. To ensure symmetry, the least squares technique introduced by equation (3.9) can be applied. In [10] it is pointed out that introducing the border region may reduce the effect of using different BCs. But on the other hand, the problem to be solved is much bigger than in the pure local method. For a two-dimensional grid, the total number of fine cells will be nine times bigger, and for a three-dimensional grid, it will be 27 times bigger if we assume that all coarse cells contains the same number of fine cells. This will increase the computational costs significantly. It is also possible to extend the domain even more by adding more rings of coarse cells to the border region. However, this has shown little improvement [10] and the computational costs will increase even more.

3.3 Global Methods

Global methods primarily upscale transmissibility rather than permeability, but as we have seen these quantities are related. The idea of global methods is to solve a global flow problem on the fine grid model and use the solution to compute upscaled transmissibilities (or permeabilities) on the coarse grid. At a first glance this seems to violate the purpose of upscaling, as the computational costs of solving the global flow problem is huge. However, when this is done once, one can solve several different flow scenarios on the coarse grid by using the upscaled quantities calculated from the fine scale solution.

Let i and $i+1$ be the index of two neighboring coarse cells, and denote by $\tilde{T}_{i+1/2}$ the upscaled transmissibility between the cells. Let $q_{i+1/2}$ be the flow rate across the common interface and $\langle p \rangle_i$ the average over cell i of the fine scale pressure solution. From (3.1), upscaled transmissibilities can now be calculated as

$$\tilde{T}_{i+1/2} = \frac{q_{i+1/2}}{\langle p \rangle_i - \langle p \rangle_{i+1}}.$$

This gives a first estimate for the upscaled transmissibilities. In highly heterogeneous models, this procedure may lead to negative transmissibilities, and iterations are performed until all are positive and a sufficient level of agreement between the fine and coarse solution is achieved [10]. The level of agreement can be measured in many ways, and this gives rise to different versions of global methods.

Quasi Global Methods

Quasi global methods, also called local-global methods, try to take into account effects from the global flow without actually solving the global fine scale problem. Instead they try to utilize coarse scale simulations to estimate which BCs to be used in the extended local calculations. Transmissibility is primarily considered when doing quasi global upscaling, but the method is analogous for permeability, which we will consider here.

First, upscaled permeability tensors for all coarse cells are calculated from the extended local method with any of the BCs described above. Next, a global simulation on the coarse grid is performed using the upscaled permeability tensors. Further, the extended local method is performed again, now with the pressure solution from the previous global simulation as BCs. Notice that the pressure solution on the boundary needs to be interpolated onto the fine scale grid in order to be used as BCs. These steps are iterated until the upscaled permeabilities no longer change with iteration. Usually, a thresholding procedure is included, so that only coarse cells with flow rates bigger than a preset magnitude is upscaled in

each iteration. This handles unfeasible results (such as negative transmissibilities) and decreases the computational costs.

Multiscale Methods

Multiscale methods are not in themselves upscaling methods, but may be viewed as alternatives to the classical upscaling methods described above. They are just briefly mentioned in [10], so we base this presentation on [3]. The main idea of the multiscale method presented therein is to solve the global flow problem on a coarse grid, where the local discrete approximation spaces are constructed in such a way that fine-scale heterogeneity is accounted for. In a finite element setting, this means that the basis functions are computed numerically by solving local flow problems [3]. An algorithm for calculating these basis functions is given in [3]. The method can be applied to very general coarse grid geometries, but to ensure accurate solutions, some guidelines should be followed.

The main advantage of the multiscale method is that it can be as efficient as the classical upscaling methods, and at the same time produce a detailed and conservative velocity solution on the fine grid [3]. It should be mentioned that this is based on a comparison with quasi-global methods. Another advantage is that it handles more general coarse cell geometry, as most of the classical upscaling methods require the coarse cells to be a hexahedrons.

Many different variants of the multiscale methods have been developed, not just the one mentioned here. However, the principles are the same.

3.4 Two-Phase Upscaling

When considering a multi-phase system, it is often necessary to upscale relative permeability in addition to absolute permeability. Relative permeability is dependent on saturation and the governing equation (2.6) is also time dependent. Thus, upscaling of a multi-phase system is much more involved than for a single-phase system. In this presentation we will restrict ourselves to a two-phase system of oil(o) and water(w), but the methods described here can also be applied to other two-phase systems. We also assume that the permeability on the underlying fine grid is isotropic, and that we for each rock type present in the model, are given relative permeability for both phases and the J-function as functions of water saturation. We denote these functions by $k_{ri}(S_w)$ and $J(S_w)$ respectively. In this thesis, only local steady-state upscaling will be considered. Other techniques are introduced for instance in [2].

Steady-state upscaling methods assume that the time dependency in equation (2.6) can be neglected [2]. Without any wells, the governing equation is

$$\nabla \cdot [\mathbf{K}_i(\vec{x})(\nabla p_i - \rho_i \vec{g})] = 0, \qquad \text{on } \Omega, \quad i = o, w. \tag{3.13}$$

This is on the same form as (3.3), so we can use the local methods from Section 3.2. This will result in an upscaled phase permeabilty tensor, $\tilde{\mathbf{K}}_i$, from which we can use relation (2.3) to calculate the upscaled relative permeability,

$$\tilde{\mathbf{K}}_{ri} = \tilde{\mathbf{K}}_i \cdot \tilde{\mathbf{K}}^{-1}. \tag{3.14}$$

This deviates slightly from most literature, as we assume relative permeability to be a tensor, not just a scalar. The phase permeability on the underlying fine grid, can be computed from the relative permeability curves, i.e.,

$$\mathbf{K}_i(\vec{x}) = k_{ri}(S_w)\mathbf{K}(\vec{x}). \tag{3.15}$$

Notice that one has to use the relative permeability curve associated with the rock type at position $\vec{x}$. Since S_w is a dependent parameter, the upscaled relative permeability calculated from (3.14) is valid for this particular saturation distribution. It is common to upscale S_w, and say that one particular $\tilde{\mathbf{K}}_{ri}$ is valid for this upscaled water saturation. To get the full picture, the upscaling procedure is repeated for a given number of saturation points. Water saturation is upscaled by volume averaging, i.e.,

$$\tilde{S}_w = \frac{\int_\Omega \phi(\vec{x}) S_w(\vec{x}) \, \mathrm{d}\Omega}{\int_\Omega \phi(\vec{x}) \, \mathrm{d}\Omega}. \tag{3.16}$$

The only problem we are left with now is to find the saturation distribution, $S_w(\vec{x})$, in the model at steady-state. This is not a trivial problem, and generally this forces us to run the full flow problem (2.6) until steady-state is reached. This problem is further discussed later, but first we look at two methods to approximate the steady-state distribution.

Two-Phase Upscaling at Capillary Equilibrium

In the first approach we assume that capillary pressure is constant in time and space and that capillary forces dominates, so that we can neglect viscous and gravitational forces. This is typically a good approximation on small scales or in regions with low flow rate. Recall that the J-function (2.10) is dependent on the water saturation and can be expressed as

$$J(S_w) = \frac{p_c \sqrt{\frac{k}{\phi}}}{\sigma \cos(\theta)}.$$

For each rock type, $J(S_w)$ is a known function, and it is strictly monotone. Hence, we can find the saturation by taking its inverse [2],

$$S_w(\vec{x}) = J^{-1}\left(\frac{p_c\sqrt{\frac{k}{\phi}}}{\sigma\cos(\theta)}\right). \tag{3.17}$$

This means that given the capillary pressure, we can calculate the water saturation in each cell from (3.17). Notice that we must use the correct J-function, that is, the one that corresponds to the rock type at position $\vec{x}$.

It is relatively easy to include gravitational effects into this method [2]. Now the capillary pressure is not constant in space, but assumed to be in gravitational equilibrium, which means that

$$p_c(\vec{x}) = \hat{p}_c + (\rho_w - \rho_o)gz_0,$$

where $\hat{p}_c$ is the capillary pressure at a fixed point, for instance at the model center, and z_0 is the vertical distance between $\vec{x}$ and the model center. Since gravitational effects are not taken into account in the J-function, $J(S_w)$, capillary forces should still be the dominant force [2].

The upscaling process is repeated for several capillary pressures (or center capillary pressures). In practice, one initially select a uniform distribution of upscaled saturation points that one wants to upscale for. Then the corresponding capillary pressure points, are calculated from the J-function curves. This process is done to ensure that the upscaled relative permeabilities correspond to upscaled saturations which are nearly uniformly distributed.

In practice, the J-function and the relative permeabilities as functions of S_w are given as discrete values. Interpolation is needed in order to evaluate these functions between the discrete points. One can assume that these curves originally are continuously differentiable, and that the interpolation therefore should be at least cubic. Also, both the J-function and the relative permeability functions, are monotonic, and we have seen that this is a necessary property for the upscaling procedure to work. Hence, the interpolation should also be monotonicity preserving.

Two-Phase Upscaling in the Viscous Limit

The second approach assumes that viscous forces are dominant, and that capillary and gravitational forces can be neglected. This can be a good approximation in regions with high flow rate. Further, the fractional flow is assumed constant throughout the model. In [11] it is argued why this is a reasonable assumption for stable displacements. Since capillary pressure is neglected, the pressure of the phases are equal. Thus, we write $p \equiv p_w = p_o$. We still assume that the

permeability is isotropic. The fractional flow rate in any direction is now given by Darcy's law (2.4),

$$\frac{v_w}{v_o} = \frac{\frac{-kk_{rw}}{\mu_w}\nabla p}{\frac{-kk_{ro}}{\mu_o}\nabla p} = \frac{k_{rw}\mu_o}{k_{ro}\mu_w}. \tag{3.18}$$

Since we are given the relative permabilities as functions of water saturation, we can find the two unknowns $S_w(\vec{x})$ and $S_o(\vec{x})$ from the system of equations [2],

$$\begin{aligned} k_{rw}(S_w) &= \frac{v_w}{v_o}\frac{\mu_w}{\mu_o}k_{ro}(S_w), \\ S_w + S_o &= 1. \end{aligned} \tag{3.19}$$

This system has a unique solution since $k_{ri}(S_w)$ is strictly monotone and thus invertible. Notice that the relative permeability curves associated with the rock type at $\vec{x}$ must be used. The flow rate, $\frac{v_w}{v_o}$, is constant by assumption, and the upscaling procedure can be repeated for different flow rates to get the full picture. For a given flow rate, the upscaled water saturation can be calculated from (3.16). Observe that the saturation is equal for fine grid cells with the same rock type. Thus, if we only have one relative permeability curve, the saturation will be equal in all fine grid cells.

General Steady-State Upscaling

General steady-state upscaling involves solving the full transport equation (2.6) until steady-state is reached [2]. This is very costly compared to the two approximations above, as we will see in Section 6.3. Together with one of the BCs described in Section 3.2, one needs to specify the fluid injection rates at the boundaries and an initial saturation distribution. To ensure computational efficiency, it is essential to make a good initial "guess". One approach is to use the saturation distribution obtained at the capillary equilibrium (3.17). Another appraoch is to use the distribution obtained in the viscous limit (3.19).

The main drawback of the general steady-state upscaling is that it is computational very costly. The method also assumes that the transport problem has a unique steady-state solution, and this property is hard to prove in most cases. But, on the other hand it is quite general, as it takes into account both viscous, capillary and gravitational forces.

3.5 Discussion

We have now seen that permeability can be upscaled in a variety of ways. Which of the methods to choose depends on many factors. It is clearly important that the upscaled properties capture as much of the fine scale heterogeneities as possible. This means that there should be a high level of agreement between the full field simulation on the fine grid and full field simulation on the coarse grid. This is primarily a theoretical objective as full field simulation on the fine grid is seldom performed for all flow scenarios. It is also important to specify which characteristics that are most important to capture from the fine scale model. Further, computational costs should be taken into account.

Global methods are expected to give a higher level of agreement between the simulations than local methods. This is because global methods take into account global effects, while local methods only consider the target cell. But global methods require to solve the global flow problem on the fine grid. This can be computationally very costly and in some cases also impossible with today's computer resources. The upscaled quantities are also dependent on the specific choice of global flow problem. Thus, global methods are expected to give good results for flow scenarios that are similar to the fine grid simulation, but we have no guarantee that this holds for more different scenarios.

Since global methods require to solve a global flow problem, they put restrictions on the underlying fine grid. This grid can not be a too fine refinement of the simulation grid if it should be computationally possible to solve the global flow problem. This means that global methods are best suited for upscaling from geological models to simulation models, as geological models typically are just small refinements of the simulation model. Local methods are more general since they can be used to upscale also from finer grid models. They are also applicable to other scales. Further, local methods fit better into a REV framework than global methods do. Our main objective is to upscale REV that can be used several places in the full field simulation model. Thus, upscaling of every simulation cell is often not necessary or even meaningless if all parts of the reservoir do not have unique measurements. Local methods are therefore considered better for prediction, which is what we primary aim for, while global methods more or less reproduce a particular historic flow.

Another advantage of local methods is that they are computationally less costly compared to global methods. They are also independent of the rest of the reservoir. Thus, the problem of upscaling reservoir cells can easily be split into smaller tasks. This is for instance is very convenient for parallelization.

When using local methods, it can be difficult to determine which BCs to use in the different cases. However, as we have seen, this issue is not completely removed for global methods either. A claim stated by King [14] can be very useful in this

context[1]:

Claim 1 *The upscaled permeability resulting from the local method with fixed BCs is a lower bound on the permeability, while the upscaled permeability resulting from the local method with linear BCs is an upper bound.*

Thus, if we get similar results for both fixed, linear and periodic BCs, we can be more certain that the upscaled permeabilities are good. If not, one has to do a more detailed analysis of the model to figure out which BCs that are most feasible.

Based on the discussion above, we have chosen to consider only local upscaling methods in the following. Mathematically this corresponds to solving the elliptic partial differential equation (3.3). For complex grids this has to be done numerically, and any of the methods mentioned in 2.4 can be used. In OPM, and also in this thesis, the mimetic finite difference method (MFDM) is chosen.

[1]From private correspondence it was pointed out that this claim was built on a variational argument based on dissipation and the number of degrees of freedom in the velocity field. However, the proof was not found in the literature.

Chapter 4

Mimetic Finite Difference Method

The mimetic finite difference method (MFDM) as solver for the elliptic diffusion equation (3.3) has shown to work nicely on corner-point grids [3]. The main reason for this is that the MFDM is very flexible with respect to cell geometry. Hence, degenerated cells with less than six faces and extended cells with more than six faces can easily be handled. Cells with more than six faces can be used to make non-conforming grids conforming by introducing additional cell faces along non-conforming surfaces in the grids. Based on this flexibility, and with the implications of the other discretization methods discussed in Section 2.4 in mind, we have chosen to use the MFDM in this thesis.

The MFDM can be presented in a variety of ways. The presentation given here is based on [16] (but without the reaction term, i.e., $c \equiv 0$), since this gives an element-based derivation which we find the most intuitive. For other presentations, see for instance [8], [9] or [3]. The latter shows how the MFDM can be viewed as a counterpart to a mixed hybrid finite element method. The main idea of the MFDM is to mimic the underlying properties of the original continuum differential operators [8].

We are considering the three dimensional elliptic diffusion problem (3.3), which may be written as a system of two equations,

$$\begin{aligned} \vec{v} &= -\mathbf{K}\nabla p, \\ \nabla \cdot \vec{v} &= b, \end{aligned} \qquad \text{in } \Omega \subset \mathbb{R}^3. \tag{4.1}$$

For simplicity, we have neglected gravity. The source term, $b = b(\vec{x})$, is zero in the upscaling problem, but it is included here for completeness. The permeability tensor, $\mathbf{K} = \mathbf{K}(\vec{x})$, is assumed to be full in general and strongly elliptic, i.e., there exists constants α_1 and α_2 such that

$$\alpha_1 \|\vec{u}\|^2 \leq \vec{u}^\top \mathbf{K}(\vec{x})\vec{u} \leq \alpha_2 \|\vec{u}\|^2, \qquad \forall \vec{u} \in \mathbb{R}^3, \quad \forall \vec{x} \in \Omega. \tag{4.2}$$

Equation (4.1) is accompanied with some appropriate boundary conditions (BCs), which are described later.

Let Ω^h be a partition of Ω into N closed simply-connected polyhedrons E_i, that is,

$$\Omega^h = \bigcup_{i=1}^{N} E_i.$$

Notice that in a corner-point grid, E_i corresponds to a cell. However, the number of faces will not be restricted to six. Hereafter, we denote by E an arbitrary polyhedra in $\{E_i, i = 1, \ldots, N\}$, and by ∂E its boundary. Further, if E has s_E faces, we denote by $\mathcal{F}_E^k$, for $k = 1, \ldots, s_E$, the faces of E. If there is no ambiguity which polyhedron we are on, we simplify and just write $\mathcal{F}^k$, for $k = 1, \ldots, s$.

Following [16], we integrate the second equation of (4.1) over E, and apply the divergence theorem. This yields

$$\sum_{k=1}^{s} \int_{\mathcal{F}^k} \vec{v} \cdot \vec{n} \, \mathrm{d}S = \int_E b \, \mathrm{d}\Omega,$$

where $\vec{n}$ is the unit normal vector pointing out of E. This equation motivates why the normal components of the velocity averaged over the faces are used as discrete unknowns in the MFDM. We will refer to these quantities as flux.

Next, we introduce the flux operator, $\mathcal{G}$, and the generalized divergence operator, $\mathcal{D}$,

$$\mathcal{G}p = -\mathbf{K}\nabla p, \tag{4.3}$$

$$\mathcal{D}\vec{v} = \begin{cases} \nabla \cdot \vec{v} & \text{on } E, \\ -\vec{v} \cdot \vec{n} & \text{on } \partial E. \end{cases} \tag{4.4}$$

Further, let X and Q denote the velocity and pressure spaces respectively, and define the following scalar products on them:

$$[\vec{v}, \vec{u}]_X = \int_E \vec{v}\mathbf{K}^{-1}\vec{u} \, \mathrm{d}\Omega, \tag{4.5}$$

$$[p, q]_Q = \int_E pq \, \mathrm{d}\Omega + \int_{\partial E} pq \, \mathrm{d}S. \tag{4.6}$$

Now recall Green's formula,

$$-\int_E \vec{v}\nabla p \, \mathrm{d}\Omega = \int_E p\nabla \cdot \vec{v} \, \mathrm{d}\Omega - \int_{\partial E} p\vec{v} \cdot \vec{n}\mathrm{d}S, \tag{4.7}$$

which we can rewrite as

$$[\vec{v}, \mathcal{G}p]_X = [p, \mathcal{D}\vec{v}]_Q, \tag{4.8}$$

and hence we see that $\mathcal{G}$ and $\mathcal{D}$ are adjoint, $\mathcal{G} = \mathcal{D}^*$, with respect to the scalar products (4.5) and (4.6) [16]. This property is preserved in the MFDM when discrete counterparts to $\mathcal{G}$ and $\mathcal{D}$ are defined.

4.1 Discretization on Element

The MFDM is usually presented in four steps, and so it will be here. This section is purely based on [16], but with some additional intermediate calculations. The *first* step is to specify the degrees of freedom (DOF) and their location. For the pressure unknown we set the DOF to $s+1$, one at the center of mass of E and the rest at the centroids of the s faces. Hence, let Q_E^h be the $s+1$ dimensional vector space of discrete pressure functions, $\mathbf{p}_E = (p_E^0, p_E^1, \ldots, p_E^s)$, where p_E^0 is the pressure unknown at the center of mass, and p_E^k, for $k = 1, \ldots, s$, are the pressure unknowns at the face centroids. We use subscript E to denote that these unknowns are associated with element E.

The unknowns for the discrete velocity are chosen to be the normal components of the velocity, $v_E^1, \ldots, v_E^s$, located at the face centroids. That is, v_E^k approximate the scalar product $\vec{v}\cdot\vec{n}$ on face $\mathcal{F}_E^k$. Hence, the discrete velocity space X_E^h is defined as the s dimensional vector space of discrete flux functions $\mathbf{v}_E = (v_E^1, \ldots, v_E^s)$.

The *second* step is to define scalar products on the discrete spaces:

$$[\mathbf{p}_E, \mathbf{q}_E]_{Q_E^h} = p_E^0 q_E^0 |E| + \sum_{k=1}^{s} p_E^k q_E^k |\mathcal{F}^k|, \qquad \forall \mathbf{p}_E, \mathbf{q}_E \in Q_E^h, \tag{4.9}$$

$$[\mathbf{v}_E, \mathbf{u}_E]_{X_E^h} = \mathbf{v}_E^\top \mathbf{M}_E \mathbf{u}_E = \sum_{r,c=1}^{s} M_{E,r,c} v_E^r v_E^c, \qquad \forall \mathbf{v}_E, \mathbf{u}_E \in X_E^h, \tag{4.10}$$

where $|E|$ is the volume of E, $|\mathcal{F}^k|$ is the area of $\mathcal{F}^k$ and $\mathbf{M}_E$ is a $s \times s$ symmetric positive definite (SPD) matrix with entries $M_{E,r,c}$. The choice of $\mathbf{M}_E$ is one of the most crucial part of the MFDM as it must somehow approximate the continuous one with sufficient accuracy. We will return to how this matrix can be constructed later.

The *third* step is to define the discrete counterpart to $\mathcal{D}$, $\mathcal{D}^h$. First, we define the discrete divergence operator,

$$\mathcal{DIV}^h \mathbf{v}_E = \frac{1}{|E|} \sum_{k=1}^{s} v_E^k |\mathcal{F}^k|, \tag{4.11}$$

which is motivated by the divergence theorem,

$$\int_E \nabla \cdot \vec{v} \, \mathrm{d}\Omega = \int_{\partial E} \vec{v} \cdot \vec{n} \, \mathrm{d}S = \sum_{k=1}^{s} \int_{\mathcal{F}^k} \vec{v} \cdot \vec{n} \, \mathrm{d}S.$$

We can now define $\mathcal{D}^h$ as

$$\mathcal{D}^h \mathbf{v}_E = (\mathcal{DIV}^h \mathbf{v}_E, -v_E^1, \ldots, -v_E^s). \tag{4.12}$$

Finally, the *fourth* and last step is to define the discrete flux operator, $\mathcal{G}^h$, as the adjoint to $\mathcal{D}^h$ with respect to the scalar products (4.9) and (4.10), i.e.,

$$[\mathbf{v}_E, \mathcal{G}^h \mathbf{p}_E]_{X_E^h} = [\mathbf{p}_E, \mathcal{D}^h \mathbf{v}_E]_{Q_E^h}, \qquad \forall \mathbf{p}_E \in Q_E^h, \quad \forall \mathbf{v}_E \in X_E^h. \tag{4.13}$$

If we look closer on the left and right hand sides of this equation and use the definitions, we see that

$$\begin{aligned}
\text{LHS} &= \mathbf{v}_E^\top \mathbf{M}_E \mathcal{G}^h \mathbf{p}_E \\
\text{RHS} &= \left[\left(p_E^0, p_E^1, \ldots, p_E^s \right), \left(\mathcal{DIV}^h \mathbf{v}_E, -v_E^1, \ldots, -v_E^s \right) \right]_{Q_E^h} \\
&= p_E^0 \frac{1}{|E|} \sum_{k=1}^{s} \left(v_E^k |\mathcal{F}^k| \right) |E| + \sum_{k=1}^{s} p_E^k (-v_E^k) |\mathcal{F}^k| \\
&= \sum_{k=1}^{s} (p_E^0 - p_E^k) v_E^k |\mathcal{F}^k| = \mathbf{v}_E^\top \mathbf{D}_E \mathbf{d}_E,
\end{aligned}$$

where

$$\mathbf{D}_E = \begin{bmatrix} |\mathcal{F}^1| & & 0 \\ & \ddots & \\ 0 & & |\mathcal{F}^s| \end{bmatrix} \quad \text{and} \quad \mathbf{d}_E = \begin{bmatrix} p_E^0 - p_E^1 \\ \vdots \\ p_E^0 - p_E^s \end{bmatrix}. \tag{4.14}$$

Since (4.13) should hold for all $\mathbf{v}_E \in X_E^h$, we conclude that $\mathbf{M}_E \mathcal{G}^h \mathbf{p}_E = \mathbf{D}_E \mathbf{d}_E$ or

$$\mathcal{G}^h \mathbf{p}_E = \mathbf{M}_E^{-1} \mathbf{D}_E \mathbf{d}_E. \tag{4.15}$$

We are now ready to present the local discretization equations, which follows from substituting the continuous operators with the discrete ones into (4.1),

$$\begin{aligned}
\mathbf{v}_E &= \mathcal{G}^h \mathbf{p}_E, \\
\mathcal{DIV}^h \mathbf{v}_E &= \frac{1}{|E|} b_E,
\end{aligned} \tag{4.16}$$

where

$$b_E = \int_E b(\vec{x}) \mathrm{d}\Omega.$$

4.2 Interface and Boundary Conditions

To close the system, we impose continuity conditions on the interfaces. This section is also built on [16]. Hereafter, let v_i^k and p_i^k denote the flux and pressure respectively on $\mathcal{F}_{E_i}^k$, which we for simplicity will denote $\mathcal{F}_i^k$ from now. Thus, the continuity conditions can be expressed as

$$v_{i_1}^{k_1} = -v_{i_2}^{k_2} \tag{4.17}$$
$$p_{i_1}^{k_1} = p_{i_2}^{k_2} \tag{4.18}$$

if polydron E_{i_1} shares its face k_1 with the k_2th face of E_{i_2}.

For those faces that lie on the boundary $\partial\Omega$, we must impose some boundary conditions. Here, Dirichlet and Neumann conditions will be considered. Periodic BCs will be treated in Chapter 5. We assume that $\partial\Omega^h$ can be partitioned into two domains, Γ_D and Γ_N, such that $\partial\Omega = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$. The BCs can now be written as

$$\begin{aligned} p &= g_D \quad \text{on } \Gamma_D, \\ \vec{v}\cdot\vec{n} &= g_N \quad \text{on } \Gamma_N, \end{aligned}$$

where $g_D = g_D(\vec{x})$ and $g_N = g_N(\vec{x})$ are known functions. If $\mathcal{F}_i^k$ is on Γ_D, then

$$p_i^k = \frac{1}{|\mathcal{F}_i^k|}\int_{\mathcal{F}_i^k} g_D(\vec{x})\ \mathrm{d}S, \tag{4.19}$$

or if $\mathcal{F}_i^k$ is on Γ_N, then

$$v_i^k = -\frac{1}{|\mathcal{F}_i^k|}\int_{\mathcal{F}_i^k} g_N(\vec{x})\ \mathrm{d}S. \tag{4.20}$$

4.3 Assembly Process

We now split the pressure unknowns on a polyhedron E into cell pressure, $p_E = p_E^0$, and interface pressures, $\boldsymbol{\pi}_E = (\pi_E^1, \ldots, \pi_E^s) = (p_E^1, \ldots, p_E^s)$. If we use the relation (4.15), the first equation in (4.16) can be written as

$$\mathbf{M}_E\mathbf{v}_E - \mathbf{C}_E^\top p_E + \mathbf{D}_E\,\boldsymbol{\pi}_E = 0,$$

where $\mathbf{D}_E$ is as defined in (4.14) and

$$\mathbf{C}_E = \begin{bmatrix} |\mathcal{F}^1| & \cdots & |\mathcal{F}^s| \end{bmatrix}. \tag{4.21}$$

Further, the second equation can be written as

$$\mathbf{C}_E \mathbf{v}_E = b_E.$$

Finally, the element contribution from the continuity condition (4.17) multiplied with the face areas, can be written as

$$\mathbf{D}_E \mathbf{v}_E = 0.$$

If we first assume that we are on an internal polyhedron E, we can write the local system of equations on matrix form,

$$\begin{bmatrix} \mathbf{M}_E & \mathbf{C}_E^\top & \mathbf{D}_E^\top \\ \mathbf{C}_E & \mathbf{0} & \mathbf{0} \\ \mathbf{D}_E & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v}_E \\ -p_E \\ \boldsymbol{\pi}_E \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ b_E \\ \mathbf{0} \end{bmatrix}. \tag{4.22}$$

If we *are* on the boundary, that is, if some of the faces of E are on $\partial\Omega$, we use the boundary conditions (4.19) and (4.20). This will remove some entries on the left hand side and introduce new terms on the right hand side that are related to the boundary conditions.

Let N_f be the total number of faces, that is $N_f = \sum_{i=1}^{N} s_{E_i}$. Notice that internal faces in Ω^h are counted twice. Let N_i denote the number of unique interfaces in Ω^h. We now collect the flux unknowns into a global vector $\mathbf{v} = (\mathbf{v}_{E_1}, \ldots, \mathbf{v}_{E_N})$ of dimension N_f, and the cell pressure unknowns into a global vector $\mathbf{p} = (p_1, \ldots, p_N)$ of dimension N. Further, we use the continuity condition on the pressure (4.18) to reduce the number of unknown interface pressures from N_f to N_i, and denote by $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_{N_i})$ the global vector of unknowns for the interface pressures. By using local to global mappings, we can assemble the local contributions into a global system [16],

$$\begin{bmatrix} \mathbf{M} & \mathbf{C}^\top & \mathbf{D}^\top \\ \mathbf{C} & \mathbf{0} & \mathbf{0} \\ \mathbf{D} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ -\mathbf{p} \\ \boldsymbol{\pi} \end{bmatrix} = \begin{bmatrix} \mathbf{g}_D \\ \mathbf{b} \\ \mathbf{g}_N \end{bmatrix}. \tag{4.23}$$

Here, $\mathbf{M}$ is a $N_f \times N_f$ matrix, $\mathbf{C}$ is a $N \times N_f$ matrix, and $\mathbf{D}$ is a $N_i \times N_f$ matrix. The N_f dimensional vector $\mathbf{g}_D$ stems from Dirichlet boundary contributions, and its ith component is given by (4.19) if the ith flux unknown is on Γ_D, and 0 otherwise. Similarly, the N_i dimensional vector $\mathbf{g}_N$ stems from Neumann boundary contributions, and its ith component is given by (4.20) if the ith interface pressure unknown is on Γ_N, and 0 otherwise.

The first block system of (4.23) is the discrete analogue to Darcy's law, $\vec{v} = -\mathbf{K}\nabla p$, while the second block system is the discrete analogue to mass conservation, $\nabla \cdot \vec{v} = b$. The third block system corresponds to the continuity condition

(4.17). Notice that for $\mathbf{v}$ and $\mathbf{p}$, the unknowns are only connected within a single polyhedron, and hence we have that

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}_{E_1} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{M}_{E_N} \end{bmatrix} \quad \text{and} \quad \mathbf{C} = \begin{bmatrix} \mathbf{C}_{E_1} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{C}_{E_N} \end{bmatrix}. \tag{4.24}$$

The matrix $\mathbf{D}$ will not have this structure as one unknown interface pressure is shared by two polyhedrons, unless we are at the boundary. Now, take a look at the third block system of (4.23), $\mathbf{Dv} = \mathbf{g}_N$. Here, each row corresponds to one interface, and for an internal interface the row vectors of $\mathbf{D}$ has two non-zero entries, both equal to the area of the corresponding interface. These non-zero entries are located at the positions corresponding to the two flux unknowns associated with this interface, and hence the continuity condition (4.17) is satiesfied.

4.4 Schur-Complement Reduction

We will now show that the system (4.23) can be reduced to a symmetric positive-definite (SPD) system [16]. We start by solving the first block system for $\mathbf{v}$,

$$\mathbf{v} = \mathbf{M}^{-1}(\mathbf{C}^\top \mathbf{p} - \mathbf{D}^\top \boldsymbol{\pi} + \mathbf{g}_D). \tag{4.25}$$

Insertion into the two other blocks of (4.23) gives the reduced system

$$\begin{bmatrix} \mathbf{E} & -\mathbf{F}^\top \\ \mathbf{F} & -\mathbf{D}\mathbf{M}^{-1}\mathbf{D}^\top \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ \boldsymbol{\pi} \end{bmatrix} = \begin{bmatrix} \mathbf{b} - \mathbf{C}\mathbf{M}^{-1}\mathbf{g}_D \\ \mathbf{g}_N - \mathbf{D}\mathbf{M}^{-1}\mathbf{g}_D \end{bmatrix}, \tag{4.26}$$

where $\mathbf{E} = \mathbf{C}\mathbf{M}^{-1}\mathbf{C}^\top$ and $\mathbf{F} = \mathbf{D}\mathbf{M}^{-1}\mathbf{C}^\top$. Since $\mathbf{M}$ is block diagonal with SPD blocks, we have that

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{M}_{E_1}^{-1} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{M}_{E_N}^{-1} \end{bmatrix},$$

so inverting $\mathbf{M}$ is fairly cost effective. In practice, $\mathbf{M}_E^{-1}$ can be calculated directly, not $\mathbf{M}_E$, as we will see in the next section. Because of the block structure of $\mathbf{C}$ and $\mathbf{M}^{-1}$, $\mathbf{E}$ is diagonal, and hence $\mathbf{E}$ is trivially invertible.

Next, solve the first equation in (4.26) for $\mathbf{p}$,

$$\mathbf{p} = \mathbf{E}^{-1}(\mathbf{F}^\top \boldsymbol{\pi} + \mathbf{b} - \mathbf{C}\mathbf{M}^{-1}\mathbf{g}_D), \tag{4.27}$$

and insert this into the other block system. This results in the SPD system

$$\mathbf{S}\,\boldsymbol{\pi} = \mathbf{r}, \quad \text{where } \begin{cases} \mathbf{S} = \mathbf{D}\mathbf{M}^{-1}\mathbf{D}^{\top} - \mathbf{F}\mathbf{E}^{-1}\mathbf{F}^{\top}, \\ \mathbf{r} = (\mathbf{D} - \mathbf{F}\mathbf{E}^{-1}\mathbf{C})\mathbf{M}^{-1}\mathbf{g}_D - \mathbf{g}_N + \mathbf{F}\mathbf{E}^{-1}\mathbf{b}. \end{cases} \tag{4.28}$$

The system is now reduced considerably, and once $\boldsymbol{\pi}$ is calculated, $\mathbf{p}$ and $\mathbf{v}$ can be found from (4.27) and (4.25) respectively.

It is also possible to do Schur-complement reduction directly in the local matrix (4.22). This results in the system

$$\mathbf{S}_E\,\boldsymbol{\pi}_E = \mathbf{r}_E. \tag{4.29}$$

The linear system (4.28) can now be obtained by assembling (4.29). In OPM, and also in this thesis, (4.28) is solved iteratively by preconditioned conjugate gradient methods. These numerical linear solvers are provided by DUNE.

4.5 Scalar Product in the Discrete Flux Space

We now return to how the element matrix $\mathbf{M}_E$ can be constructed. In the following, denote by $[\mathbf{v}, \mathbf{u}]_E$ the local scalar product on the discrete flux space, i.e., $[\mathbf{v}, \mathbf{u}]_E = [\mathbf{v}_E, \mathbf{u}_E]_{X_E^h}$, see (4.10). In [9] it is shown how this scalar product can be calculated. This derivation is reviewed here, but we have chosen to include some additional intermediate calculations.

Let X^h be the global function space of discrete flux. Further, let $L^s(\Omega)$ be the set of Lebesgue measurable real-valued functions w on Ω for which [12, p. 702]

$$\|w\|_{L^s(\Omega)} \equiv \left(\int_\Omega |w|^s \, \mathrm{d}\Omega\right)^{\frac{1}{s}} < \infty, \qquad 1 \le s < \infty.$$

We now define an interpolation operator. For every $\vec{u} \in (L^s(\Omega))^3$, $s > 2$, with $\nabla \cdot \vec{u} \in L^2(\Omega)$, we define $\mathbf{u}^I \in X^h$ by

$$(\mathbf{u}^I)_E^k = \frac{1}{|\mathcal{F}_E^k|} \int_{\mathcal{F}_E^k} \vec{u} \cdot \vec{n}_E^k \, \mathrm{d}S, \qquad \forall E \in \Omega^h, \quad k = 1, 2, \ldots, s_E. \tag{4.30}$$

By $\vec{u} \in (L^s(\Omega))^3$ we mean that each component of $\vec{u}$ is in $L^s(\Omega)$, i.e., $u_i \in L^s(\Omega)$ for $i = 1, 2, 3$.

According to [8], two conditions are sufficient to prove convergence and stability in pressure and velocity of the MFDM for very general polyhedrons:

(S1) There exists two positive constants s_* and S^* such that for every element $E \in \Omega^h$ we have

$$s_* \sum_{k=1}^{s_E} (u_E^k)^2 |E| \leq [\mathbf{u}, \mathbf{u}]_E \leq S^* \sum_{k=1}^{s_E} (u_E^k)^2 |E|, \qquad \forall \mathbf{u} \in X^h. \tag{4.31}$$

(S2) For every element E, every linear function q^1 on E and every $\mathbf{u} \in X^h$, we have

$$[(\mathbf{K}\nabla q^1)^I, \mathbf{u}]_E + \int_E q^1 (\mathcal{DIV}^h \mathbf{u}_E) \, \mathrm{d}\Omega = \sum_{k=1}^{s_E} u_E^k \int_{\mathcal{F}_E^k} q^1 \, \mathrm{d}S. \tag{4.32}$$

The first condition states that there should exist a global bound on the eigenvalues of all $\mathbf{M}_E$, and the second condition states that the inner product $[\cdot, \cdot]_E$ should obey the discrete version of Green's formula (4.7) for linear pressure. A direct consequence is that the method will be exact for linear pressure [8].

We will rewrite **(S2)** by, for each E, setting the origin equal to the center of the mass of E, and use that for any q^1 linear we can write $q^1 = a_0 + a_1 x_1 + a_2 x_2 + a_3 x_3$, where (x_1, x_2, x_3) are the Cartesian coordinates. Observe that

$$\int_E x_i (\mathcal{DIV}^h \mathbf{u}_E) \, \mathrm{d}V = (\mathcal{DIV}^h \mathbf{u}_E) \int_E x_i \, \mathrm{d}V = 0 \quad \text{for } i = 1, 2, 3,$$

since $(0, 0, 0)$ is the center of mass of E. Further, $\{1, x_1, x_1, x_2\}$ is a basis for q^1, and hence we can replace **(S2)** by the following one[1].

(S2') For every element E with center of mass at the origin and every $\mathbf{u} \in X^h$, we have

$$[(\mathbf{K}\nabla x_i)^I, \mathbf{u}]_E = \sum_{k=1}^{s_E} u_E^k \int_{\mathcal{F}_E^k} x_i \, \mathrm{d}S, \qquad i = 1, 2, 3. \tag{4.33}$$

If we apply the divergence theorem on the vector field $\mathbf{f} = x_j \mathbf{K}\nabla x_i$ and observe that $\nabla(\mathbf{K}\nabla x_i) = 0$ since $\mathbf{K}$ is assumed constant on E, we get the following identity,

$$\int_{\partial E} (\mathbf{K}\nabla x_i) \cdot \vec{n} x_j \, \mathrm{d}S = \int_E (\mathbf{K}\nabla x_i) \cdot \nabla x_j \, \mathrm{d}V = |E| K_{j,i}, \tag{4.34}$$

where $K_{i,j}$ is the entry (i, j) of $\mathbf{K}$. Next, define the $s_E \times 3$ matrices $\mathbf{R}$ and $\mathbf{N}$ as

$$R_{k,i} = \int_{\mathcal{F}^k} x_i \, \mathrm{d}S \quad \text{and} \quad N_{k,i} = (\mathbf{K}\nabla x_i) \cdot \vec{n}_E^k, \tag{4.35}$$

for $k = 1, 2, \ldots, s_E$ and $i = 1, 2, 3$. If we look at the matrix product $\mathbf{R}^{\mathrm{T}}\mathbf{N}$ and use (4.34), we see that

[1]The identity basis (1) results in the definition of the discrete divergence operator (4.11) when inserted into (4.32).

$$
\begin{aligned}
(\mathbf{R}^{\mathrm{T}}\mathbf{N})_{i,j} &= \sum_{k=1}^{s_E} \left\{ \int_{\mathcal{F}^k} x_i \, \mathrm{d}S \; (\mathbf{K}\nabla x_j) \cdot \vec{n}_E^k \right\} \\
&= \sum_{k=1}^{s_E} \left\{ \int_{\mathcal{F}^k} x_i (\mathbf{K}\nabla x_j) \cdot \vec{n}_E^k \, \mathrm{d}S \right\} \\
&= \int_{\partial E} x_i (\mathbf{K}\nabla x_j) \cdot \vec{n} \, \mathrm{d}S = |E| K_{i,j}.
\end{aligned}
$$

Thus, we see that

$$
\mathbf{R}^{\mathrm{T}}\mathbf{N} = |E|\mathbf{K}. \tag{4.36}
$$

We now rewrite (4.33) by using (4.30), (4.10) and (4.35). We begin with the left hand side (LHS):

$$
\begin{aligned}
\mathrm{LHS} &= \sum_{k,l=1}^{s_E} M_{E,k,l} \left((\mathbf{K}\nabla x_i)^I \right)_E^k u_E^l \\
&= \sum_{k,l=1}^{s_E} M_{E,k,l} \left(\frac{1}{|\mathcal{F}^k|} \int_{\mathcal{F}^k} (\mathbf{K}\nabla x_i) \cdot \vec{n}_E^k \, \mathrm{d}S \right) u_E^l \\
&= \sum_{k,l=1}^{s_E} M_{E,k,l} \frac{1}{|\mathcal{F}^k|} \int_{\mathcal{F}^k} N_{k,i} \, \mathrm{d}S \; u_E^l \\
&= \sum_{k,l=1}^{s_E} M_{E,k,l} N_{k,i} u_E^l.
\end{aligned}
$$

Next, consider the right hand side (RHS) of (4.33),

$$
\mathrm{RHS} = \sum_{k=1}^{s_E} R_{k,i} u_E^k.
$$

Since (4.33) is to hold for all $\mathbf{u} \in X^h$ and for $i = 1, 2, 3$, we conclude that

$$
\mathbf{M}_E\mathbf{N} = \mathbf{R}, \tag{4.37}
$$

and that this is equivalent to **(S2)**. Next, define

$$
\mathbf{M}_0 = \frac{1}{|E|}\mathbf{R}\mathbf{K}^{-1}\mathbf{R}^{\top}. \tag{4.38}
$$

We easily see from (4.36) that $\mathbf{M}_0$ satisfies (4.37). Also, $\mathbf{M}_0$ is symmetric, but only positive semidefinite [9]. The following theorem, stated in [9], shows that we can add an extra term to $\mathbf{M}_0$ to make it SPD.

Theorem 1 *Let* $\mathbf{G}$ *be a* $s_E \times (s_E - 3)$ *matrix whose columns span the null space of the full rank matrix* $\mathbf{N}^\top$*, so that* $\mathbf{N}^\top \mathbf{G} = 0$*. Then, for every* $(s_E - 3) \times (s_E - 3)$ *SPD matrix* $\mathbf{U}$*, the following symmetric matrix*

$$\mathbf{M}_E = \mathbf{M}_0 + \mathbf{G}\mathbf{U}\mathbf{G}^\top \tag{4.39}$$

satisfies (4.37) *and is SPD.*

By construction, $\mathbf{M}_E\mathbf{N} = \mathbf{M}_0\mathbf{N}$, so $\mathbf{M}_E$ satisfies (4.37). $\mathbf{M}_E$ is also by construction symmetric and positive semidefinite. Hence, it remains to show that $\mathbf{M}_E$ is non-singular. For a proof of this, see [9].

Analogously, the next theorem, also stated and proved in [9], shows how to construct $\mathbf{M}_E^{-1}$ directly.

Theorem 2 *Let* $\mathbf{H}$ *be a* $s_E \times (s_E - 3)$ *matrix whose image span the null space of* $\mathbf{R}^\top$*, that is* $\mathrm{im}(\mathbf{H}) = \ker(\mathbf{R}^\top)$*. Then, for every* $(s_E - 3) \times (s_E - 3)$ *SPD matrix* $\tilde{\mathbf{U}}$*, the following symmetric matrix*

$$\mathbf{M}_E^{-1} = \frac{1}{|E|}\mathbf{N}\mathbf{K}^{-1}\mathbf{N}^\top + \mathbf{H}\tilde{\mathbf{U}}\mathbf{H}^\top \tag{4.40}$$

satisfies (4.37) *and is SPD.*

The matrix $\mathbf{M}_E$ calculated from (4.39) is certainly not unique. The SPD matrix $\mathbf{U}$ has at least $\frac{1}{2}(s_E - d + 1)(s_E - d)$ free parameters [9], and the family of matrices defined by (4.39) also includes the TPFA [17].

We have now explained how to calculate SPD matrices $\mathbf{M}_E$ and $\mathbf{M}_E^{-1}$, so that **(S2)** is satiesfied. To satisfy **(S1)**, some additional mild restriction on $\mathbf{U}$ (or $\tilde{\mathbf{U}}$) are necessary. This corresponds to making some restrictions on the grid cells. These restrictions are mild, and a wide variety of shapes are allowed, for instance degenerated and non-convex cells. These restrictions are handled in [8], and not further discussed here.

4.6 Convergence Results

Stability and convergence of the MFDM is proved in [8]. The convergence results are stated here without proofs. Let us first define norms on the spaces X^h and Q^h,

$$\|\mathbf{p}\|_{Q^h}^2 = \sum_{i=1}^{N} p_E^2 |E|,$$
$$\|\mathbf{v}\|_{X^h}^2 = \sum_{i=1}^{N} [\mathbf{v}, \mathbf{v}]_E.$$

Further, let h_E be the diameter of E, and $h = \sup_E h_E$. The diameter of a polyhedron is defined as the longest distance between two vertices. We denote by $\mathbf{v}^h$ and $\mathbf{p}^h$ the numerical solutions obtained from solving (4.23), and by $\mathbf{v}^I$ and $\mathbf{p}^I$ the interpolants of the exact solutions, $\vec{v}$ and p, respectively. An error estimate for the velocity is now given by [8]

$$\|\mathbf{v}^I - \mathbf{v}^h\|_{X^h} \leq Ch\|p\|_{H^2(\Omega)},$$

where C is a constant only dependent on the geometry and on the constants given in **(S1)**, **(S1)** and (4.2). Similarly, an error estimate for the pressure is given by [8]

$$\|\mathbf{p}^I - \mathbf{p}^h\|_{Q^h} \leq C^* h^2 \left(\|p\|_{H^2(\Omega)} + \|b\|_{H^1(\Omega)}\right),$$

where C^* is a constant with the same dependencies as C. $H^1(\Omega)$ and $H^2(\Omega)$ are the standard Sobolev spaces, defined for instance in [12, p. 258–259].

We end this chapter by concluding that the MFDM is of first order in velocity and of second order in pressure [8].

Chapter 5

Implementing Periodic Boundary Conditions

Periodic boundary conditions (BCs), see (3.10), have some properties that make them very interesting. First of all they intuitively seem to be more correct (from an upscaling point of view) than fixed or linear BCs, especially if the reservoir is close to periodic in nature. In Chapter 6, we will support these statements by numerical results. Further, Claim 1 (p. 33) states that fixed and linear BCs result in lower and upper bounds respectively, and that the results from periodic BCs will be somewhere in between the two extremes. Lastly, the permeability tensor resulting from periodic BCs is SPD, which is a nice property in many contexts of mathematics.

On the other hand, as we will see in Chapter 6, periodic BCs has some numerical implications which makes them harder to solve, especially compared to linear BCs. Further, periodic BCs connect opposite boundary faces, and this is not always trivial. In this chapter we therefore look into how periodic BCs can be implemented into the MFDM. The aim is to improve the current representation in OPM. Three different approaches will be described, where the first is the one currently implemented in OPM. The second is not implemented or tested in this thesis, but included for completeness. The latter is based on mortar methods, and is derived, implemented and tested in this thesis.

We assume that our domain Ω is formed like a regular hexahedron, see Figure 3.2. When we refer to the grid on a boundary face, we mean the two-dimensional grid obtained by taking the trace of the three-dimensional corner-point grid onto the boundary face. Further, we say that two such grids are matching if they are equal to each other, see Figure 5.1 for an example. We will assume that all pillars are vertical. This means that the grids on $\partial\Omega^{z,1}$ and $\partial\Omega^{z,2}$ are matching. This is in general not the case for the other boundary faces, so enforcing periodicity is not straightforward.

5.1 Single-Point Constraints

Single-point constraints (SPC) require the grids on opposite boundaries to be matching. Let $\mathcal{L}$ and $\mathcal{R}$ denote two opposite boundaries. If the grid on $\mathcal{L}$ and $\mathcal{R}$ matches, then every face on $\mathcal{L}$ has what we will refer to as a periodic partner on $\mathcal{R}$. Let i be the global index of a cell face on either $\mathcal{L}$ or $\mathcal{R}$. Then we denote by i' the index of its periodic partner, see Figure 5.1.

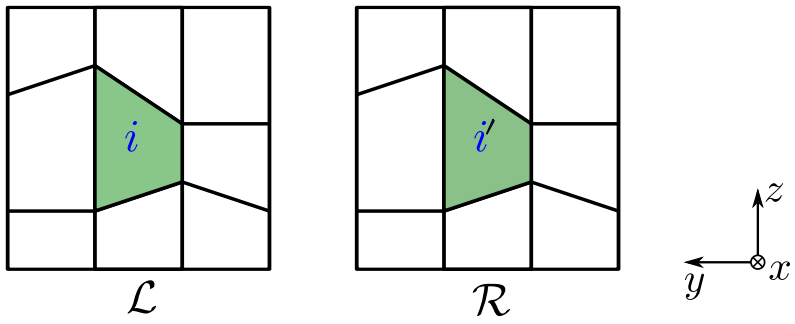


Figure 5.1: Example of a matching grid, here in the x-direction. Periodic partners, i and i', are also shown.

The periodicity of the interface pressure is now enforced by simply connecting periodic partners,

$$\pi_i = \pi_{i'} \quad \text{or} \quad \pi_i = \pi_{i'} + \Delta p \tag{5.1}$$

if there is a pressure drop, Δp, in the current direction. Next, we define which of our degrees of freedom (DOFs) on the boundary that should be treated as masters and which should be treated as slaves. Master DOFs are the real unknowns in our system, while slaves are calculated from (5.1). In the assembly process, all contributions to slave DOFs should be redirected to their corresponding master DOFs. Master and slave DOFs could be defined in any way such that either i or i' is the master, and the other one is the slave. In the OPM framework, it is chosen to define the DOFs with the lowest index as the master. That is, if $i < i'$ then i is the master and vica versa. So, in the assembly of the system (4.28), whenever you are on a boundary face, you have to check if the periodic partner has a lower index. If so, the contribution from the element matrix should be directed to the index of the periodic partner. If there is a pressure drop, this should be included on the right hand side of equation (4.28).

Periodicity of the flux unknowns are enforced similarly, i.e.,

$$v_i = -v_{i'}. \tag{5.2}$$

The minus sign is due to the opposite directions of the normal vectors on the two faces. This constraint states continuity in the flux across boundary faces. Recall that flux continuity across interfaces are weakly enforced through the equation $\mathbf{Dv} = \mathbf{g}_N$ in (4.23). Thus the constraint (5.2), can be included into this system.

For realistic reservoir models, it is very seldom that the grids on opposite boundaries match in the horizontal directions. So, to be able to use SPC, one needs to make the boundary grids matching. This can be done, as illustrated in Figure 5.2, by introducing extra faces on the two opposite boundaries so that the two grids are the intersection of the original grids. This process can be costly in itself, and the number of unknowns increases, which makes our linear system larger and possibly harder to solve. Also notice that the corresponding cells now can have more than six faces. This is not a problem in the MFDM, as the derivation in Chapter 4 was done for arbitrary polyhedrons. Once the process of adding extra faces is done, the SPC method can be applied as described above. This technique is used in OPM.

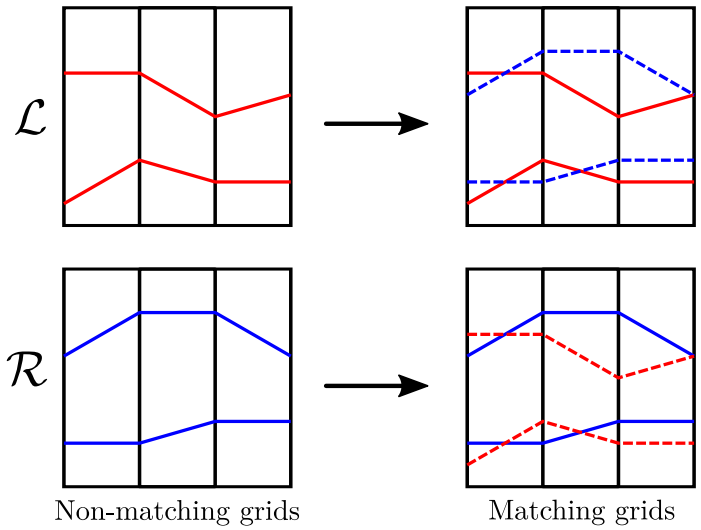


Figure 5.2: The process of making two non-matching grids matching. The new grids are simply the intersection of the two original grids. As we see, new faces are introduced.

5.2 Multi-Point Constraints

Multi-point constraints (MPC) are not implemented or tested in this thesis. However, the method is included for completeness. MPC is a generalization of SPC,

as it builds on the same principles with master and slave DOFs, but it does not require matching grids. The basic idea is to define the DOFs at one of the two opposite boundaries as the master DOFs, and define the DOFs at the other boundary as the slave DOFs. Then, the contributions to the slave DOFs are redirected to one or several master DOFs.

Assume that we are at a DOF on the slave boundary with index i and with a corresponding unknown π_i. If the face which the DOF is on has a matching face on the opposite boundary, then we are lucky and can use the SPC (5.1). If not, the DOF will be connected to n_i DOFs, with indexes $i_1, \ldots, i_{n_i}$, at the opposite boundary,

$$\pi_i = \sum_{k=1}^{n_i} w_k \pi_{i_k} + \Delta p, \tag{5.3}$$

see Figure 5.3 for an illustration. The weights, w_k, reflects the distance from the slave DOF to the connected master DOF, and we must require that $\sum_{k=1}^{n_i} w_k = 1$. Now the contributions to π_i from the assembly should be redirected, with fractions w_k, to π_{i_k}, for $k = 1, \ldots, n_i$. Any pressure drop will be added to the right hand side as in the SPC method. The number of connected DOF, n_i, will vary depending on the spesific connection, but one way to do this is to connect a slave DOF to all master DOFs which are on faces that intersect with the face associated with the slave DOF. This is done in the example in Figure 5.3.

Periodicity of flux can be enforced by the constraint

$$A_i v_i = -\sum_{k=1}^{n_i} A_{i,i_k} v_{i_k}. \tag{5.4}$$

The minus sign is due to the opposite directions of the normal vectors on the faces. In this context, n_i is the number of faces on the master grid which intersect with face i (which was assumed to be on the slave boundary). Further, A_i is the area of face i, and A_{i,i_k} is the area of the intersection between face i and face i_k. Clearly, $A_i = \sum_{k=1}^{n_i} A_{i,i_k}$. The constraint (5.4) can be included into the equation $\mathbf{D}\mathbf{v} = \mathbf{g}_N$ in the linear system (4.23).

5.3 A Mortar Method

Mortar methods are usually used to connect non-matching grid blocks, e.g., in domain decomposition methods, see for instance [5] or [7]. Here, we will use a variant of them to impose periodic BCs on non-matching boundary grids. This variant has been studied (not yet published) on a similar problem related to elasticity with a finite element method as the underlying discretization. However, for the elliptic

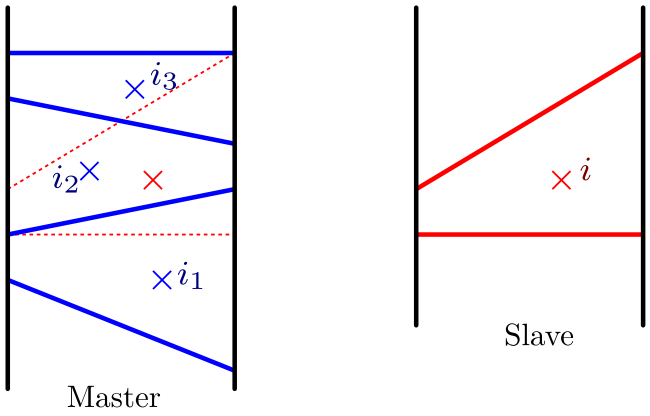


Figure 5.3: Example of two matching pillars on the master and slave boundary and how DOF i can be connected to DOF i_1, i_2 and i_3. DOF i can also be connected to DOF on other pillars, but this is not taken into account here.

diffusion equation (4.1) and in connection to the MFDM, this technique has not been studied before.

Since the grids are matching in the z-direction, we can simply use SPC here, so in this section we focus on the x- and y-directions. We will also only consider continuity of the interface pressures. Continuity in the flux can be enforced similarly. We assume that the pillars on opposite boundary faces are matching. Let $\partial\Omega^{\xi}$ denote the boundary that we want to impose periodicity on. The mortar method enforces periodicity weakly by adding the constraint

$$\int_{\partial\Omega^{\xi}} \lambda \left(p|_{\partial\Omega^{\xi,1}} - p|_{\partial\Omega^{\xi,2}}\right) \, \mathrm{d}S = 0, \qquad \forall \lambda \in \Lambda \tag{5.5}$$

to our original system of equations (4.1). Here, λ is a Lagrangian multiplier and Λ its function space. The mortar method introduced here is not a classical mortar method due to our choice of function space Λ. However, we will still refer to it as a mortar method. Classical mortar methods takes the intersection of the two boundary grids, and do integration over each resulting element, see Figure 5.4. But finding this intersection is non-trivial and it may result in very degenerated elements. We also see from Figure 5.4 that the number of faces may increase, which results in increased computational costs. Instead, we take advantage of our grid geometry, and define Λ to be the set of bilinear functions on elements defined by two neighboring pillars. Thus the dimension of Λ, N_l, is two times the number of pillars on the boundary grid. The grid consisting of elements defined by neighboring pillars, will be referred to as the mortar grid, see Figure 5.4. This may seems like a coarse discretization, but it has shown to work quite nice on the

elasticity problem.

If there is defined a pressure drop, Δp, in the current direction, the mortar constraint (5.5) will be

$$\int_{\partial\Omega^\xi} \lambda \left(p|_{\partial\Omega^{\xi,1}} - p|_{\partial\Omega^{\xi,2}}\right) \, \mathrm{d}S = \int_{\partial\Omega^\xi} \Delta p \lambda \, \mathrm{d}S, \qquad \forall \lambda \in \Lambda. \tag{5.6}$$

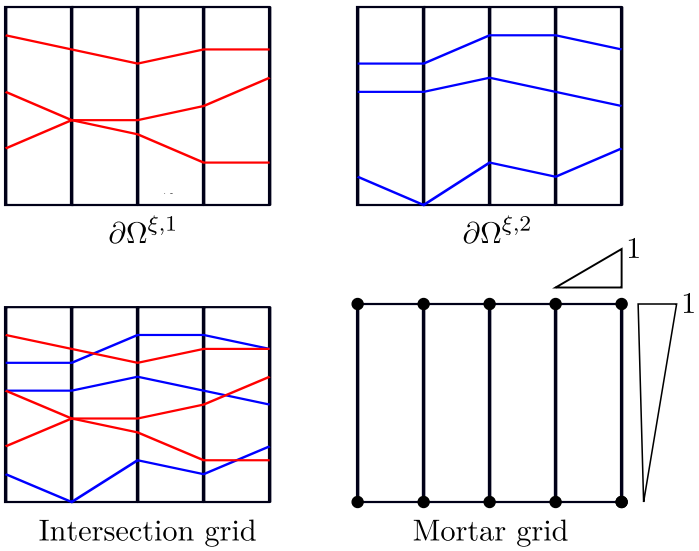


Figure 5.4: Example of how the intersected grid will look like given two arbitrary boundary grids. The mortar grid consisting only of the pillars is also shown along with the DOF for the Lagrangian multiplier, λ. The basis function for the upper right DOF is also illustrated.

System of Equations

We now show how the mortar constraint (5.6) can be transformed into a linear system of equations that can be added to the linear system (4.28). The derivations of the linear systems are the same for the x- and y-directions. Hence, let ξ be either x or y in the following. For easier notation, denote by $\mathcal{L}$ the left side of the boundary, i.e., $\mathcal{L} \equiv \partial\Omega^{\xi,1}$, and by $\mathcal{R}$ the right side of the boundary, i.e., $\mathcal{R} \equiv \partial\Omega^{\xi,2}$. Let n_L and n_R be the number of faces on $\mathcal{L}$ and $\mathcal{R}$ respectively. Further, denote by $\mathcal{Q}_j^L$, for $j = 1, \ldots, n_L$, the faces on $\mathcal{L}$, and by $\mathcal{Q}_j^R$, for $j = 1, \ldots, n_R$, the faces on $\mathcal{R}$. Next, define the mappings

$$\beta_L : \{1, \dots, n_L\} \to \{1, \dots, N_i\},$$
$$\beta_R : \{1, \dots, n_R\} \to \{1, \dots, N_i\},$$

which maps the boundary interface index to the global interface index. We have already discretized the pressure at the interfaces in the previous chapter, denoting those unknowns $\boldsymbol{\pi} = (\pi_1, \dots, \pi_{N_i})$. Hereafter, let p_j^L denote the interface pressure on $\mathcal{Q}_j^L$, so that $p_j^L = \pi_{\beta_L(j)}$. Similarly, let p_j^R denote the interface pressure on $\mathcal{Q}_j^R$, so that $p_j^R = \pi_{\beta_R(j)}$. Hence, we can write

$$p|_{\partial\Omega^{\xi,1}}(\vec{x}) = \sum_{j=1}^{n_L} p_j^L \chi_j^L(\vec{x}), \qquad \text{where} \quad \chi_j^L(\vec{x}) = \begin{cases} 1 & \text{if } \vec{x} \in \mathcal{Q}_j^L \\ 0 & \text{otherwise.} \end{cases}$$
$$p|_{\partial\Omega^{\xi,2}}(\vec{x}) = \sum_{j=1}^{n_R} p_j^R \chi_j^R(\vec{x}), \qquad \text{where} \quad \chi_j^R(\vec{x}) = \begin{cases} 1 & \text{if } \vec{x} \in \mathcal{Q}_j^R \\ 0 & \text{otherwise.} \end{cases}$$

If we further let $\{\varphi_1, \dots, \varphi_{N_L}\}$ be a basis for Λ, we can rewrite (5.6) as

$$\begin{aligned}
\int_{\partial\Omega^\xi} \varphi_i \left(\sum_{j=1}^{n_L} p_j^L \chi_j^L - \sum_{j=1}^{n_R} p_j^R \chi_j^R \right) \mathrm{d}S &= \Delta p \int_{\partial\Omega^\xi} \varphi_i \, \mathrm{d}S, \qquad i = 1, \dots, N_l \\
\sum_{j=1}^{n_L} p_j^L \int_{\partial\Omega^\xi} \varphi_i \chi_j^L \, \mathrm{d}S - \sum_{j=1}^{n_R} p_j^R \int_{\partial\Omega^\xi} \varphi_i \chi_j^R \, \mathrm{d}S &= \Delta p \int_{\partial\Omega^\xi} \varphi_i \, \mathrm{d}S, \qquad i = 1, \dots, N_l \\
\sum_{j=1}^{n_L} p_j^L \int_{\mathcal{Q}_j^L} \varphi_i \, \mathrm{d}S - \sum_{j=1}^{n_R} p_j^R \int_{\mathcal{Q}_j^R} \varphi_i \, \mathrm{d}S &= \Delta p \int_{\partial\Omega^\xi} \varphi_i \, \mathrm{d}S, \qquad i = 1, \dots, N_l.
\end{aligned} \tag{5.7}$$

Next, let $\mathbf{L}$ be a $N_f \times N_l$ matrix whose entries are defined as

$$L_{i,j} = \begin{cases} \displaystyle\int_{\mathcal{Q}_k^L} \varphi_j \, \mathrm{d}S & \text{if } i = \beta_L(k) \text{ for some } k \in \{1, \dots, n_L\}, \\ \displaystyle -\int_{\mathcal{Q}_k^R} \varphi_j \, \mathrm{d}S & \text{if } i = \beta_R(k) \text{ for some } k \in \{1, \dots, n_R\}, \\ 0 & \text{otherwise.} \end{cases} \tag{5.8}$$

Further, let $\mathbf{h}$ be a N_f dimensional vector with entries

$$h_i = \begin{cases} \Delta p & \text{if } i = \beta_L(k) \text{ for some } k \in \{1, \dots, n_L\}, \\ 0 & \text{otherwise.} \end{cases}$$

Now, (5.7) can be written in matrix form,

$$\mathbf{L}^\top \boldsymbol{\pi} = \mathbf{L}^\top \mathbf{h}. \tag{5.9}$$

Notice that if we do not have a pressure drop in the current direction, then $\mathbf{h} = \mathbf{0}$, and the right hand side of (5.9) is zero.

The matrix $\mathbf{L}$ is referred to as the mortar matrix, and (5.9) is the mortar constraints that are imposed on the original linear system (4.28) in order to enforce periodicity. The system matrix $\mathbf{S}$ is SPD, so from the theory of optimization [18], (4.28) can be formulated as a optimization problem,

$$\min \quad q(\boldsymbol{\pi}) \equiv \frac{1}{2}\boldsymbol{\pi}^\top \mathbf{S}\,\boldsymbol{\pi} - \boldsymbol{\pi}^\top \mathbf{r}.$$

Next, we add the mortar constraints (5.9) and end up with a constrained optimization problem,

$$\begin{aligned} \min \quad & q(\boldsymbol{\pi}) \equiv \frac{1}{2}\boldsymbol{\pi}^\top \mathbf{S}\,\boldsymbol{\pi} - \boldsymbol{\pi}^\top \mathbf{r}. \\ \text{subject to} \quad & \mathbf{L}^\top \boldsymbol{\pi} = \mathbf{L}^\top \mathbf{h}. \end{aligned}$$

The Karush-Kuhn-Tucker (KKT) conditions [18, p. 451] states that there exists a N_l dimensional vector of Lagrange multipliers, $\boldsymbol{\ell}$, s.t.

$$\begin{bmatrix} \mathbf{S} & \mathbf{L} \\ \mathbf{L}^\top & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\pi} \\ \boldsymbol{\ell} \end{bmatrix} = \begin{bmatrix} \mathbf{r} \\ \mathbf{L}^\top \mathbf{h} \end{bmatrix}.$$

So far we have only enforced periodicity in one of the horizontal directions. In the following, denote by $\mathbf{L}_1$ and $\mathbf{L}_2$ the mortar matrices in the x- and y-directions respectively, by $\boldsymbol{\ell}_1$ and $\boldsymbol{\ell}_2$ the vectors of Lagrangian multipliers, and by $\mathbf{L}_1^\top \mathbf{h}_1$ and $\mathbf{L}_2^\top \mathbf{h}_2$ the corresponding right hand sides of (5.9). As mentioned, we use SPC in the z-direction. By the argument based on optimization theory presented above, we get the augmented system

$$\begin{bmatrix} \mathbf{S} & \mathbf{L}_1 & \mathbf{L}_2 \\ \mathbf{L}_1^\top & \mathbf{0} & \mathbf{0} \\ \mathbf{L}_2^\top & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\pi} \\ \boldsymbol{\ell}_1 \\ \boldsymbol{\ell}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{r} \\ \mathbf{L}_1^\top \mathbf{h}_1 \\ \mathbf{L}_2^\top \mathbf{h}_2 \end{bmatrix}. \tag{5.10}$$

This system is clearly symmetric, but not proven to be positive definite. Hence, we use a bi-conjugate gradient stabilized method (Bi-CGSTAB) [22] with an incomplete LU zero fill-in (ILU0) preconditioner [21] as the linear solver. This solver is provided by DUNE.

The magnitude of the elements in the system matrix, $\mathbf{S}$, and the mortar matrices, $\mathbf{L}_1$ and $\mathbf{L}_2$, can differ on a large scale. To avoid floating point issues, the

mortar matrices should be scaled by a factor, α, approximating the ratio between the typical magnitude of the elements in the matrices. If we look at the augmented system (5.10), such a scaling can be done without changing the solution. Motivated by the the built-in routines provided by DUNE, we have chosen to define α as the ratio between the infinity norms of the matrices, i.e.,

$$\alpha = \frac{\|\mathbf{S}\|_\infty}{\|\mathbf{L}_i\|_\infty}, \qquad i = 1, 2.$$

The infinity norm is defined as the maximum absolute row sum of the matrix.

Calculating the Mortar Matrices

We now show how the entries in $\mathbf{L}$ can be calculated. We focus merely on the left side, and look at a particular face $\mathcal{Q}_k^L$. Because of our choice of function space for the Lagrangian multipliers, Λ, only four basis functions has support on $\mathcal{Q}_k^L$. Let $\varphi_{i_1}, \ldots, \varphi_{i_4}$ be those basis functions, see Figure 5.5. This means that each row of $\mathbf{L}$ that corresponds to a boundary face, has exactly four non-zero entries. All other rows are simply zero.

Let $\hat{\mathcal{Q}}$ be the unit square, i.e., $\hat{\mathcal{Q}} = \{(\hat{x}, \hat{y}) \in \mathbb{R}^2 : 0 \leq \hat{x}, \hat{y} \leq 1\}$, also referred to as the reference element. Then there exists a one-to-one transformation θ taking $\hat{\mathcal{Q}}$ onto $\mathcal{Q}_k^L$, i.e., $(x, y) = \theta(\hat{x}, \hat{y})$. Similarly, there exists a one-to-one transformation ϑ taking $\hat{\mathcal{Q}}$ onto the pillar element on which $\mathcal{Q}_k^L$ is, see Figure 5.5. Further, let $\mathbf{J}(\hat{x}, \hat{y})$ be the Jacobian of the transformation θ,

$$\mathbf{J}(\hat{x}, \hat{y}) = \begin{bmatrix} \frac{\partial \theta_1}{\partial \hat{x}} & \frac{\partial \theta_1}{\partial \hat{y}} \\ \frac{\partial \theta_2}{\partial \hat{x}} & \frac{\partial \theta_2}{\partial \hat{y}} \end{bmatrix},$$

and denote by $|\mathbf{J}|(\hat{x}, \hat{y})$ its determinant. The integral in equation (5.8) can now be written as

$$\begin{aligned} \int_{\mathcal{Q}_k^L} \varphi_j \, \mathrm{d}S &= \int_{\mathcal{Q}_k^L} \varphi_j(x, y) \, \mathrm{d}x\mathrm{d}y \\ &= \int_{\hat{\mathcal{Q}}} \varphi_j \left(\theta(\hat{x}, \hat{y})\right) |J|(\hat{x}, \hat{y}) \, \mathrm{d}\hat{x}\mathrm{d}\hat{y} \\ &= \int_{\hat{\mathcal{Q}}} \hat{\varphi}_{\phi(j)} \left(\vartheta^{-1}\theta(\hat{x}, \hat{y})\right) |J|(\hat{x}, \hat{y}) \, \mathrm{d}\hat{x}\mathrm{d}\hat{y}, \end{aligned} \tag{5.11}$$

where $\phi : \{1, \ldots, N_l\} \to \{1, \ldots, 4\}$ is a global to local mapping of the Lagrangian multipliers, and $\hat{\varphi}_{\phi(j)}$ is the shape function associated with corner number $\phi(j)$ on the reference element $\hat{\mathcal{Q}}$. The four shape functions are bilinear and defined as

$$\begin{aligned}\hat{\varphi}_1(\hat{x},\hat{y}) &= 1-\hat{x}-\hat{y}+\hat{x}\hat{y},\\ \hat{\varphi}_2(\hat{x},\hat{y}) &= \hat{x}-\hat{x}\hat{y},\\ \hat{\varphi}_3(\hat{x},\hat{y}) &= \hat{x}\hat{y},\\ \hat{\varphi}_4(\hat{x},\hat{y}) &= \hat{y}-\hat{x}\hat{y}.\end{aligned}$$

Notice that these shape functions are Lagrangian, that is, they are 0 in all corners of $\hat{\mathcal{Q}}$ except in their associated corner where they equals 1.

We use Gaussian quadrature to evaluate the integral (5.11). Since all shape functions are bilinear, it suffices to use an order 1 method, which means that only one quadrature point is needed — the geometric center $(\hat{x},\hat{y}) = \left(\frac{1}{2},\frac{1}{2}\right)$ with weight 1. Hence the integral can be evaluated as

$$\int_{\mathcal{Q}_k^L} \varphi_j(x,y)\,\mathrm{d}x\mathrm{d}y = \hat{\varphi}_{\phi(j)}\left(\vartheta^{-1}\theta\left(\tfrac{1}{2},\tfrac{1}{2}\right)\right)|J|\left(\tfrac{1}{2},\tfrac{1}{2}\right).$$

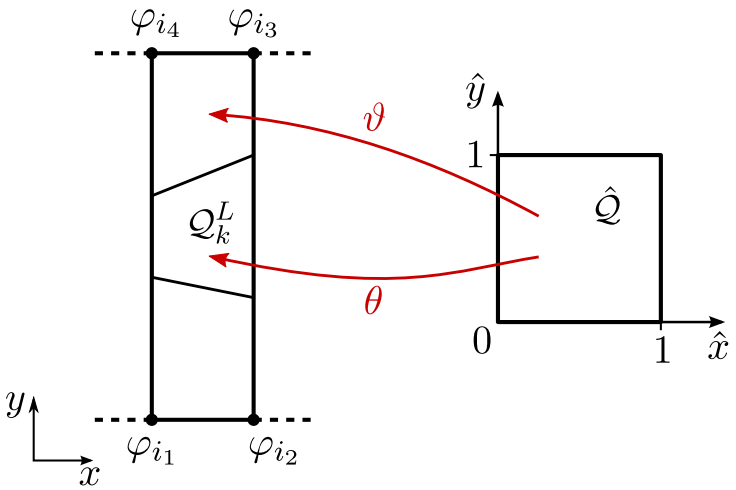


Figure 5.5: Shows the current face $\mathcal{Q}_k^L$ and the pillar element it is on, together with the two transformations θ and ϑ from the reference element $\hat{\mathcal{Q}}$ onto the face and the pillar element respectively. The basis functions with support on $\mathcal{Q}_k^L$ are also displayed.

Chapter 6

Numerical Results

In this chapter numerical results for different upscaling scenarios are presented. Based on the discussion in Section 3.5, we only consider flow based local upscaling methods. The aim is to compare different boundary conditions (BCs). One aspect is to look at which BCs that gives the most correct upscaling results. This may depend on the particular model under consideration. Another aspect is to enlighten the numerical implications associated with the upscaling methods.

We start by looking at single-phase systems. First, simple synthetic models are considered, before we increase complexity by looking at larger realistic reservoir models. Next, we expand our analysis to two-phase systems, where both the capillary equilibrium approach, the viscous limit approach and the general steady-state upscaling from Section 3.4 will be used and compared. The latter is close to full simulation, but without any wells. Thus, we have a natural increase in complexity throughout the chapter. The last section is devoted to the implemented mortar method for enforcing periodic BCs. We compare it with the current implementation in OPM to see if the numerical convergence is improved.

6.1 Verification of Single-Phase Upscaling

We start with three simple synthetic test cases to verify the calculated upscaled permeability tensors. This will also give an intuition of permeability upscaling that will be useful when analyzing realistic reservoir models. We need to understand the simple problems before moving to larger and more complex problems.

Simple Uniform Layered Model

Consider the model displayed in Figure 6.1a. This is a uniform $3 \times 3 \times 3$ grid, where the top and bottom layers is a high permeability material, while the mid

layer is a low permeability material. The permeability is isotropic in all layers and equals 100 mD and 0.1 mD respectively. The porosity is set to 0.1 throughout the model.

The calculated upscaled permeability tensors for fixed (f), linear (l) and periodic (p) BCs are shown in (6.1). First observe that all off-diagonal entries are 0. This means that there is no flow out of the boundaries parallel to the pressure drop. This is as expected due to the simplicity of our model. The x- and y-components of the upscaled permeability tensor should obviously be equal because of the symmetry in our model. For this simple model, they should equal the volume weighted arithmetic average, i.e., $\tilde{k} = \frac{1}{3}(2 \cdot 100 \text{ mD} + 0.1 \text{ mD}) = 66.7$ mD. We see that all BCs give the correct x and y components of the upscaled permeability.

In the fixed and periodic case, the z-component should equals the volume weighted harmonic average, i.e., $\tilde{k} = 3\left(\frac{2}{100 \text{ mD}} + \frac{1}{0.1 \text{ mD}}\right)^{-1} = 0.299$ mD. We see that this is satisfied. For linear BCs the z-component is much larger. This is because linear BCs allow fluid to flow out of the boundary, while for fixed BCs the fluid must flow through the low permeability layer. The result is the same for periodic BCs as for fixed since the grid layers are periodic in the the x- and y-directions, so the flow is also here forced through the mid layer. For this simple model, we conclude that linear BCs overestimate the z-component significantly.

$$\begin{aligned} \tilde{\mathbf{K}}_f &= \begin{bmatrix} 66.7 & 0 & 0 \\ 0 & 66.7 & 0 \\ 0 & 0 & 0.299 \end{bmatrix}, \quad \tilde{\mathbf{K}}_l = \begin{bmatrix} 66.7 & 0 & 0 \\ 0 & 66.7 & 0 \\ 0 & 0 & 78.5 \end{bmatrix}, \\ \tilde{\mathbf{K}}_p &= \begin{bmatrix} 66.7 & 0 & 0 \\ 0 & 66.7 & 0 \\ 0 & 0 & 0.299 \end{bmatrix}. \end{aligned} \tag{6.1}$$

Perodic Tilted Model

Next, we consider the model displayed in Figure 6.1c. This model is built up of alternating parallel layers of two isotropic materials, one high permeability material and one low permeability material. The permeability is set to 100 mD and 0.1 mD respectively, and porosity is set 0.1 throughout the model. The layers are tilted such that the model is periodic with respect to the material. By this, we mean that the grids and the materials on opposite boundary faces are matching.

The calculated upscaled permeability tensors are shown in (6.2). We start by looking at the case where the pressure drop is imposed in the x-direction. We see that one of the low permeability layers covers the yz-plane, so with fixed BCs the fluid has to pass through this layer. This results in a low upscaled permeability. For the periodic case, the fluid can flow out of one boundary and in to the opposite

boundary. Thus, the flow can avoid the barrier, and the upscaled permeability will be much higher than for fixed BCs. This is observed in (6.2). Most of the flow will follow the direction of the layers, causing flow out of the bottom and into the top. This corresponds to the negative z-direction and explains why $\tilde{k}_{zx}$ is negative and significantly different from 0 in the periodic case. There are no such effects in the y-direction, and hence $\tilde{k}_{yx} = 0$. Linear BCs also allow flow out of the top and bottom boundaries, and the upscaled permeability should be much higher than for fixed BCs. It is even higher than for periodic BCs, because the flow in and out of the boundaries are not restricted to the layers.

Next, consider a pressure drop in the y-direction. As noticed earlier, there are now flow barriers in this direction, and for fixed x and z, the permeability is constant for all y. Thus, the upscaled permeability should equal the volume weighted arithmetic average, i.e., $\tilde{k} = \frac{1}{2}(100 \text{ mD} + 0.1 \text{ mD}) = 50.05$ mD. We see that this is the case for all BCs.

For a pressure drop in the z-direction we should expect similar results as for a pressure drop in the x-direction, since these two cases are almost equal. However, observe that the domain is not a cube, but that the distance between the boundary faces in the x-direction is twice that in the z-direction. Hence, the angle between the pressure drop direction and the layers is bigger for a pressure drop in the z-direction than in the x-direction. This results in more resistance in the z-direction and explains why the z-component of the upscaled permeability is less than the x-component. This is confirmed by running on a similar model, but stretched by a factor 2 in the z-direction. In this case the angle is 45 degrees in both the x- and z-directions, and $\tilde{k}_{xx} = \tilde{k}_{zz}$ for all BCs.

$$\begin{aligned}
\tilde{\mathbf{K}}_f &= \begin{bmatrix} 3.053 & 0 & 0 \\ 0 & 50.05 & 0 \\ 0 & 0 & 0.9051 \end{bmatrix}, \quad \tilde{\mathbf{K}}_l = \begin{bmatrix} 39.69 & 0 & -19.65 \\ 0 & 50.05 & 0 \\ -19.28 & 0 & 10.41 \end{bmatrix}, \\
\tilde{\mathbf{K}}_p &= \begin{bmatrix} 38.80 & 0 & -19.30 \\ 0 & 50.05 & 0 \\ -19.30 & 0 & 9.848 \end{bmatrix}.
\end{aligned} \tag{6.2}$$

The exact solution for the periodic case can be calculated by using equation (3.12). The transformation matrix $\mathbf{A}$ is here given as

$$\mathbf{A} = \begin{bmatrix} \cos(\alpha) & 0 & -\sin(\alpha) \\ 0 & 1 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) \end{bmatrix} = \frac{1}{\sqrt{5}} \begin{bmatrix} 2 & 0 & -1 \\ 0 & \sqrt{5} & 0 \\ 1 & 0 & 2 \end{bmatrix},$$

where α is the angle between the layers and the xy-plane. Thus, from (3.12), the exact solution is

$$\tilde{\mathbf{K}}_{p,\text{exact}} = \mathbf{A}^\top \begin{bmatrix} 50.05 & 0 & 0 \\ 0 & 50.05 & 0 \\ 0 & 0 & 0.1998 \end{bmatrix} \mathbf{A} = \begin{bmatrix} 40.44 & 0 & -20.12 \\ 0 & 50.05 & 0 \\ -20.12 & 0 & 10.26 \end{bmatrix}.$$

We see that these values depart approximately 4% from the calculated ones. Observe that the upscaled tensor with linear BCs is also very close to the exact solution. Thus, in this case, linear BCs are quite accurate. Fixed BCs are however far away from the exact solution as flow in the x- and z-directions are considerably underestimated.

Non-Periodic Tilted Model

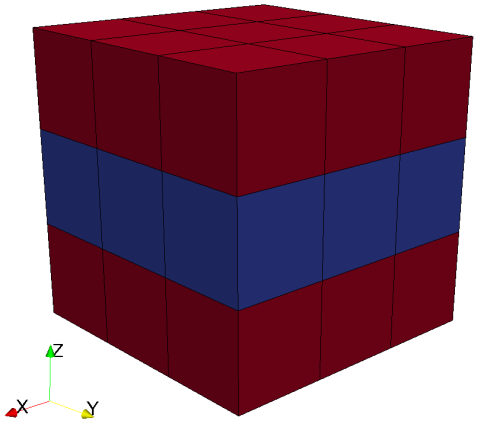
Finally, we include one more model, see Figure 6.1b. This model is similar to the previous one, but less tilted. Observe that the model is not periodic in the y-direction. The upscaled permeability tensors are seen in (6.3). We observe again that the z-component is overestimated in the linear case.

Next, consider $\tilde{k}_{yy}$ in the periodic case. This is much lower than for the fixed and linear cases. By a first glance, this seems to be an computational error, as we expect the fluid to flow relatively easy through the high permeability layers. However, with periodic BCs, what flows out at $\partial\Omega^{y,2}$ must flow in at $\partial\Omega^{y,1}$. And since the model is non-periodic, the flow through a high permeability layer has to flow into a low permeability layer at the boundary. This explains the low permeability in the y-direction for periodic BCs. It is not realistic that the true permeability is so low, and hence this is a weakness of the periodic BCs. We also observe that the off-diagonal terms are very small in the periodic case. This is due to the same effect. By intuition, one would expect these cross terms to be larger because of the layered structure. These cross terms are better captured by linear BCs, and if it were not for the overestimated z-component, linear BCs might be the best choice here.

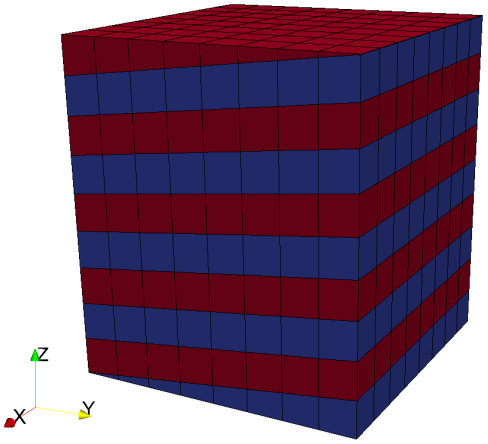
$$\begin{aligned} \tilde{\mathbf{K}}_f &= \begin{bmatrix} 49.97 & 0 & 0 \\ 0 & 44.67 & 0 \\ 0 & 0 & 0.2132 \end{bmatrix}, \quad \tilde{\mathbf{K}}_l = \begin{bmatrix} 49.97 & 0 & 0 \\ 0 & 48.82 & 9.253 \\ 0 & 5.195 & 8.336 \end{bmatrix}, \\ \tilde{\mathbf{K}}_p &= \begin{bmatrix} 49.97 & 0 & 0 \\ 0 & 1.989 & 0.2041 \\ 0 & 0.2041 & 0.2363 \end{bmatrix}. \end{aligned} \tag{6.3}$$

Lastly, we observe that $k_{p,yy} < k_{f,yy} < k_{l,yy}$. This is a contradiction of Claim 1 (p. 33), and we conclude that this claim is not valid under all circumstances. It

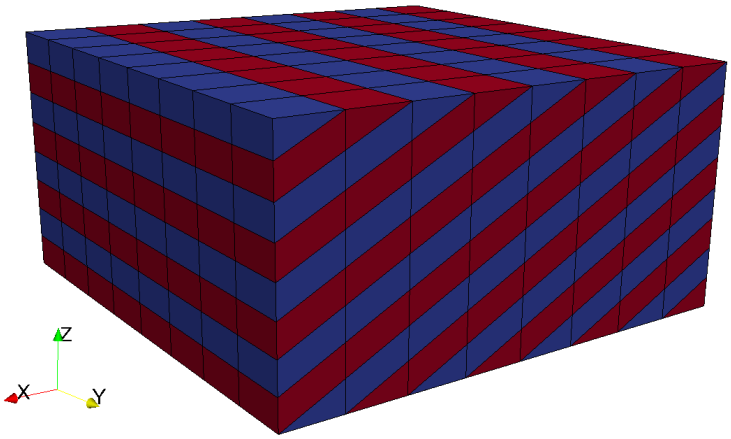
might be that the claim only is valid for periodic models, but without any proof at hand it is hard to say. Anyway, the claim should be treated with some care.



(a) Simple uniform layered model.



(b) Non-periodic tilted model.



(c) Periodic tilted model.

Figure 6.1: Synthetic test models used for verification of single-phase upscaling. All models are isotropic with permeability equal to 100 mD and 0.1 mD in the red and blue layers respectively. The porosity is set to 0.1 throughout all models.

6.2 Single-Phase Upscaling on Realistic Models

In this section we will look at realistic models, one small, which we call test model 1 (TM1), and one bigger, which we call test model 2 (TM2). The size and the degrees of freedom (DOF) for the two models can be seen in Table 6.1. Observe that the DOF for the periodic grids are larger than for the non-periodic. This is because of the process of making opposite boundary grids matching as explained in Section 5.1. The geometry of the models and the x-component of the permeability field can be seen in Figure 6.3. We see that both models contain complex patterns, degenerated cells and great variance in the permeability field. The x- and y-components of the permeability fields are equal in both models, but the z-component is different. Hence, both models are anisotropic.

Table 6.1: Number of active cells and degrees of freedom (DOF) for both the non-periodic and the periodic grid for the two models TM1 and TM2.

Model	Active cells	DOF non-periodic	DOF periodic
TM1	40350	64593	71996
TM2	93531	175724	186696

The single-phase upscaling results for TM1 and TM2 can be seen in (6.4) and (6.5) respectively. Observe that $\tilde{\mathbf{K}}_p$ is close to symmetric in both cases. Due to numerical errors in the MFDM and the linear solver, they are not exactly symmetric. The upscaled tensors seems reasonable by a first glance. However, there are great differences between the tensors, and this enlighten that the upscaling results from flow based local methods can be highly dependent on the BCs. A more detailed analysis of the models must be performed in order to understand the results better. Observe that Claim 1 (p. 33) does not hold here either, as $\tilde{k}_{p,xx} < \tilde{k}_{f,xx} < \tilde{k}_{l,xx}$ for both models.

$$\begin{aligned}
\tilde{\mathbf{K}}_f &= \begin{bmatrix} 63.42 & 0 & 0 \\ 0 & 18.86 & 0 \\ 0 & 0 & 0.8773 \end{bmatrix}, \quad \tilde{\mathbf{K}}_l = \begin{bmatrix} 95.70 & -5.293 & -2.587 \\ -1.621 & 56.43 & 6.892 \\ -0.2360 & 0.1864 & 1.623 \end{bmatrix}, \\
\tilde{\mathbf{K}}_p &= \begin{bmatrix} 46.82 & -2.153 & 0.08838 \\ -2.153 & 18.90 & -0.1765 \\ 0.09117 & -0.1788 & 0.8559 \end{bmatrix}.
\end{aligned} \tag{6.4}$$

$$\tilde{\mathbf{K}}_f = \begin{bmatrix} 62.36 & 0 & 0 \\ 0 & 14.03 & 0 \\ 0 & 0 & 0.1082 \end{bmatrix}, \quad \tilde{\mathbf{K}}_l = \begin{bmatrix} 67.46 & -2.854 & 0.2095 \\ -8.750 & 42.07 & 6.429 \\ 0.1134 & 1.027 & 0.6289 \end{bmatrix},$$
$$\tilde{\mathbf{K}}_p = \begin{bmatrix} 48.49 & -0.5352 & -0.002006 \\ -0.5352 & 17.97 & 0.06417 \\ 0.005045 & 0.06870 & 0.1100 \end{bmatrix}. \tag{6.5}$$

We now turn to study the numerical implications of single-phase upscaling with respect to BCs and linear solver. Recall from Section 3.2 that equation (3.3) must be solved for three different pressure drop directions (PDD) to get the full upscaled tensor. The computation time and the memory usage for single-phase computations for the different BCs and PDD were measured. Two different linear solvers where used — the conjugate gradient (CG) method with an incomplete LU zero fill-in (ILU0) preconditioner and CG with an algebraic multigrid (AMG) preconditioner, see [21] for both. The linear solvers are provided by DUNE.

The results are shown in Table 6.2. The condition number for the system matrix in (4.28) is also calculated numerically by using the `condest` routine provided by Matlab. This routine is based on Higham's modification of Hager's method [13], and gives a lower estimate of the condition number. The convergence of CG is dependent on the condition number [21]. In Figure 6.2, we see the sparsity pattern for the system matrices for different BCs for test model 1. The sparsity pattern is equal for all PDD.

We first consider the condition numbers. Observe that for the linear and fixed cases, the condition numbers are of the same order of magnitude. For the periodic case the condition number is much bigger, approximately a factor 10^6 larger than for fixed and linear BCs. This can be explained by the sparsity patterns seen in Figure 6.2. In the periodic case, the system matrix is much more dense than for the fixed and linear cases. The periodic system matrix also contains more entries away from the diagonal. The additional entries are due to connection of periodic partners on the boundary.

If we look at the computation times, we observe that the problem with linear BCs is by far the fastest to solve. This is also reflected by the number of iterations (IT) required. These observations were also documented in [15]. The reasons for this is probably compounded. One argument is that the flow is less restricted with linear BCs, as fluids may flow freely out of the boundary. We also see that the sparsity pattern (Figure 6.2) has lower bandwidth than with fixed and periodic BCs. However, having in mind that the degrees of freedom for test model 2 is almost three times as big as for test model 1, it is quite remarkable that the computation time only increase from 1.24 s to 2.75 s for ILU0/CG and

from 1.57 s to 4.11 s for AMG/CG when going from test model 1 to test model 2. In the fixed and periodic case, we observe an increase by a factor close to 10 in total computation time. This is closer to what one could expect.

It is also interesting to observe that with fixed and periodic BCs, the problem with a pressure drop in the z-direction is faster to solve. This is especially seen for test model 2, where the total iteration time (TIT) and the number of iterations (IT) for PDD z is up to 10 times smaller than PDD x and y. A feasible explanation is that this has to do with the restrictions that are enforced on the boundaries perpendicular to the pressure drop, and that these restrictions are easier satisfied on $\partial\Omega^{z,1}$ and $\partial\Omega^{z,2}$ (the bottom and top boundaries). The fact that this effect is not seen in the linear case, where fluids freely can flow out of the boundaries, supports this argument. Maybe more intuitively, one would expect the opposite result — that the problem with a pressure drop in the z-direction should be harder to solve — since there is more variation in the permeability field in the z-direction and also more flow barriers (i.e., layers with low permeability). One could expect this to result in numerical complications, but as reported this is not the case. To fully understand this result a more detailed analysis of the models and the numerical methods (both the MFDM and the linear solvers) would be required.

We now turn to comment on the differences between the linear solvers. For linear and fixed BCs, ILU0/CG is faster than AMG/CG for both test models. We see that AMG/CG uses considerably less iterations than ILU0/CG, but that the total iteration time is higher. This means that one iteration with AMG/CG is more expensive. This is confirmed by the memory usage, which is higher for AMG/CG than for ILU0/CG. Further, it takes more time to build the AMG preconditioner. For large systems it is expected that this effort will be payed back by reduced computation time, but for the systems considered here, this is not the case. However, if we look closer to the fixed case on test model 2, we see that the total iteration time in fact is lower for AMG/CG than for ILU0/CG. The relative differance in total computation time for the two linear solvers is also smaller for test model 2 than for test model 1. There are therefore reasons to believe that AMG/CG is faster than ILU0/CG for bigger systems, at least for fixed BCs. It should be mentioned that the AMG preconditioner is only dependent on the grid, so it is the same for all PDD. Thus it only needs to be built once. This means that in cases where we solve many flow problems on the same grid, the savings of AMG/CG would be greater. This is for instance the case in steady-state two-phase upscaling, where the problem is solved for different saturations.

Next, consider periodic BCs. In contrast to fixed and linear BCs, the total iteration time and the total computation time is less for AMG/CG than for ILU0/CG. In fact, with AMG/CG the total computation time is smaller for periodic BCs than for fixed BCs even though the original system matrix is much worse conditioned in

the periodic case. This shows that the AMG preconditioner is especially powerful on the periodic problem, which is worst conditioned.

As mentioned earlier the memory usage (MEM) for AMG/CG is higher than for ILU0/CG. This shows that the iterations for AMG/CG are more costly. But, as observed, AMG/CG requires considerably less iterations. If we consider ILU0/CG, we see that the memory usage is almost equal for fixed and linear BCs. For periodic BCs the memory usage is approximately 10% higher. This might be due to the increased number of unknowns for the periodic problem as reported in Table 6.1.

The tesselation times for the non-periodic and the periodic grids were also measured. By tesselation time we mean the time to build the corner-point grid. For test model 1 the tesselation times were 0.15 s and 0.21 s for the non-periodic and the periodic grids respectively, and for test model 2 the tesselation times were 0.34 s and 0.52 s respectively. The differences are quite small and close to negligible compared to the total computation time, at least for fixed and periodic BCs. Hence, we conclude that the process of making opposite boundary grids matching is not so costly.

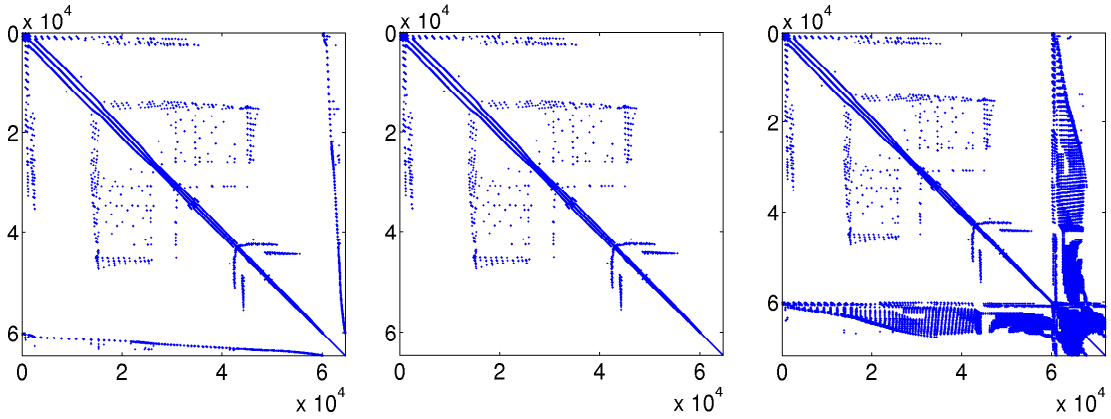
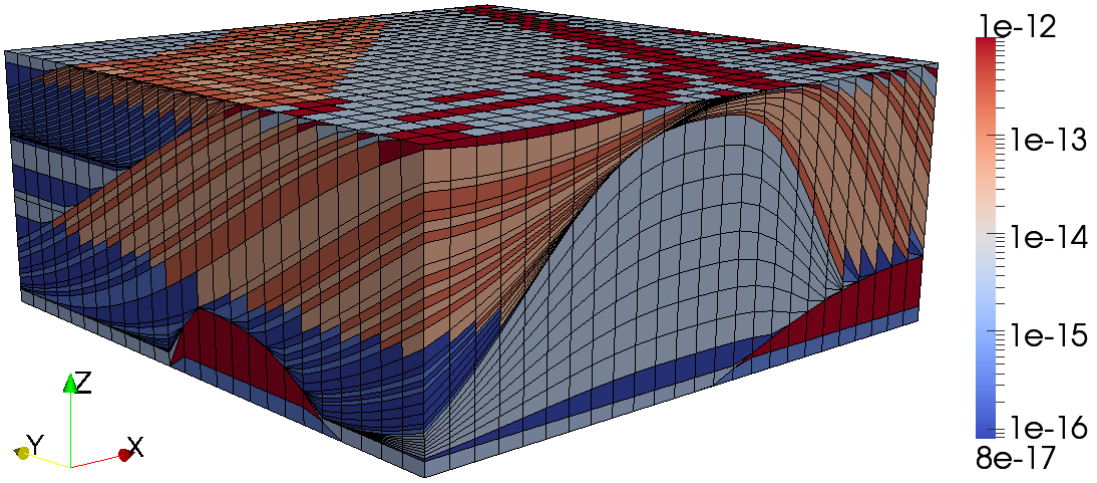
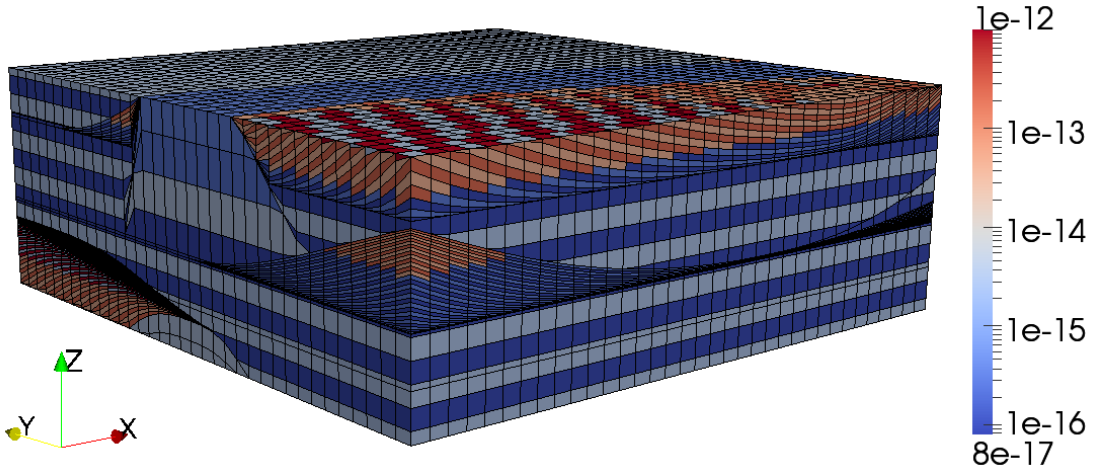


Figure 6.2: Sparsity pattern for the system matrices for fixed, linear and periodic BCs respectively for test model 1. The patterns are the same for all PDD.



(a) Test model 1.



(b) Test model 2.

Figure 6.3: Test model 1 and 2, here scaled by a factor 10 in the z-direction. The x-component of the permeability fields are shown on logarithmic scales with unit m^2. Model sizes are seen in Table 6.1.

Table 6.2: Computational data for test model 1 (a) and test model 2 (b). Single-phase upscaling were run for different BCs and with two different linear solvers, ILU0/CG and AMG/CG. The tables show estimated condition number (COND), number of iterations (IT), total iteration time (TIT) and memory usage (MEM) for the three pressure drop directions (PDD). Also total computation time (TOT) is shown. All timing results are given in seconds, whereas memory usage is in megabyte.

(a) Test model 1.

			ILU0/CG				**AMG/CG**			
BCs	**PDD**	**COND**	IT	TIT	MEM	TOT	IT	TIT	MEM	TOT
	x	$2.7 \cdot 10^{16}$	178	0.70	70		39	0.55	81	
f	y	$2.7 \cdot 10^{16}$	196	0.76	70	2.29	40	0.59	96	3.22
	z	$1.3 \cdot 10^{16}$	103	0.44	70		31	0.53	96	
	x	$1.3 \cdot 10^{16}$	51	0.21	70		13	0.20	81	
l	y	$1.3 \cdot 10^{16}$	49	0.19	70	1.24	12	0.18	81	1.57
	z	$1.3 \cdot 10^{16}$	88	0.33	70		24	0.35	81	
	x	$3.5 \cdot 10^{22}$	231	1.31	79		34	0.71	98	
p	y	$3.5 \cdot 10^{22}$	231	1.28	79	3.68	29	0.60	98	2.65
	z	$3.5 \cdot 10^{22}$	62	0.34	79		8	0.18	98	

(b) Test model 2.

			ILU0/CG				**AMG/CG**			
BCs	**PDD**	**COND**	IT	TIT	MEM	TOT	IT	TIT	MEM	TOT
	x	$2.9 \cdot 10^{17}$	879	9.35	177		279	11.15	213	
f	y	$2.9 \cdot 10^{17}$	938	10.05	177	21.58	221	8.15	256	25.23
	z	$2.3 \cdot 10^{17}$	93	0.99	177		21	0.87	257	
	x	$2.3 \cdot 10^{17}$	42	0.44	178		10	0.42	212	
l	y	$2.3 \cdot 10^{17}$	40	0.44	178	2.75	16	0.66	212	4.11
	z	$2.3 \cdot 10^{17}$	51	0.59	178		19	0.78	212	
	x	$9.7 \cdot 10^{22}$	1281	17.48	193		167	8.31	239	
p	y	$9.7 \cdot 10^{22}$	1228	16.68	193	37.34	198	9.64	239	22.20
	z	$9.7 \cdot 10^{22}$	111	1.51	193		25	1.23	239	

6.3 Two-Phase Upscaling

We now turn to two-phase upscaling and consider upscaling of relative permeability based on the three methods described in Section 3.4, namely the capillary equilibrium approach, the viscous limit approach and the general steady-state approach. Two-phase upscaling is considered to be computationally more demanding than single-phase upscaling, and we continue our focus on numerical implications. We compare the different BCs to see if the variations observed in the single-phase computations also are observed here. Further, we compare the three different upscaling approaches to test their validity. We also discuss whether relative permeability is directional dependent and thus should be treated as a tensor, or if it is satisfactory to treat it as a scalar.

Throughout this section, test model 1 (Figure 6.3a) is considered. Realistic relative permeability curves for each rock type are used as input. These are assumed isotropic for each rock type, but the (absolute) permeability is anisotropic. The model consists of 12 rock types, and the input curves for each of them are displayed in Figure 6.4.

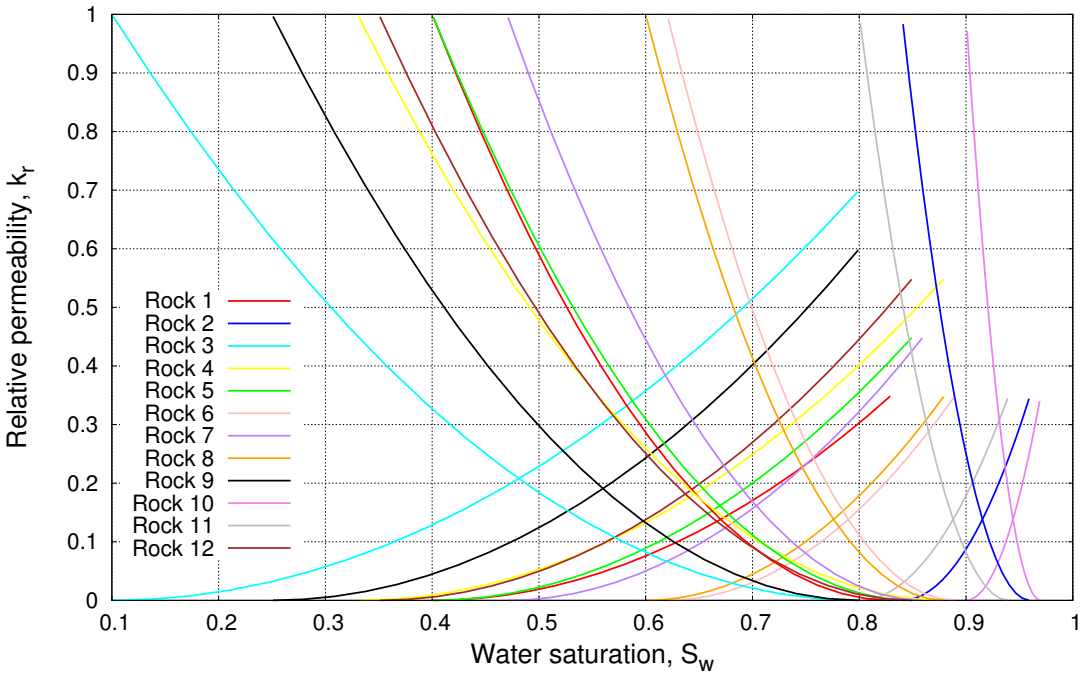


Figure 6.4: Input relative permeability curves for the 12 rock types present in test model 1. Decreasing curves are relative permeability of oil, while increasing curves are relative permeability of water.

We start with the capillary equilibrium approach. We use fixed BCs and upscale for a uniform distribution of 10 saturation points. The results are shown in Figure 6.5, where we also see the input curves for reference. Our first observation is that the shape of the upscaled curves are different from the input curves. This shows that flow based upscaling is meaningful, and that some kind of averaging of the input curves are not sufficient in this case. We also observe that there are great differences among the directional components. This supports that relative permeability should be treated as a tensor, and that assuming otherwise may be wrong in some cases.

In the previous section we documented that there are great differences in numerical convergence between the three types of BCs. We will now see if there are any differences between the different saturation points that are upscaled for. Table 6.3 shows the number of iterations (IT), total iteration time (TIT) and the condition number (COND) of the system matrix for the different saturation points and for different pressure drop directions (PDD). Numerical data for both oil and water phase calculations are shown. ILU0/CG was used as the numerical solver.

If we consider oil first, we see that the data are quite similar in the two horizontal directions (PDD x and y). This is as expected since these two case are quite similar. For the z-direction, the data are smaller, meaning that the numerical convergence is faster. This was also observed in the previous section. Apart from this the variation with respect to water saturation, S_w, follows the same pattern for all three directions. The main finding is that there is a sudden drop in the condition number for $S_w = 0.80$, followed by a huge jump for $S_w = 0.85$. The consequences on the number of iterations and the total iteration time is small for $S_w = 0.80$. However, at $S_w = 0.85$, the computation time blows up by a factor ~ 6 for the horizontal directions and a factor ~ 2 for the vertical direction.

The same observations are made for the calculation of upscaled relative permeability of water, but turned up side down. Here the condition number and the calculation time is significantly larger for $S_w = 0.40$. For water, the jump is largest for the vertical direction, in contrast to the observation for oil computations. A feasible argument that explains these observations is that at the end points the relative permeabilities, and thereby the phase permeabilities, are close to zero. From (4.40), we see that the system matrix is directly dependent on the permeability tensor (or the phase permeability tensor in a two-phase system, cf. (3.13)). So if the (phase) permeability tensor is close to zero, the resulting system matrix may be ill-conditioned. This explains the high condition numbers and the high computational costs.

The viscous limit approach was also applied to the same problem. The results are seen in Figure 6.6. If we compare with the results from the capillary equilibrium approach (Figure 6.5), we observe that the y- and z-components are almost

equal, but that the x-component is quite different. We also see that the spatial dependency is less in the viscous limit approach, especially for oil. Further, the upscaled curves for the viscous limit approach has the same shape as the input curves, and the results looks more like some kind of an average of the input curves.

The effect of using different BCs is also investigated. In Figure 6.7 and 6.8, we see the upscaling results from the capillary equilibrium approach with periodic and linear BCs respectively. By comparing these with Figure 6.5, we clearly see that the upscaled relative permeabilities are dependent on the BCs. It is $k_{ro,xx}$, $k_{ro,yy}$ and $k_{rw,yy}$ that deviate the most.

At last, general steady-state upscaling were performed. Recall that in order to get the saturation distribution, full simulation must be performed. This is computationally very costly compared to the two previously used approaches. The general steady-state upscaling is however expected to be more correct as is takes both capillary, viscous and gravitational forces into account.

For the full simulation problem, a pressure drop of 1 bar (10^5 Pa), corresponding to a water flow rate of ~ 0.01 feet/day, were enforced in the direction that was upscaled for. This is a quite small flow rate and representative for regions far away from wells. Fixed BCs were used[1] and the initial saturation distribution was set to the capillary equilibrium. The results are shown in Figure 6.9. We see that the spatial differences are small, especially for oil, and that the shape of the curves are similar to the input curves. In Figure 6.10 we compare with the results from the viscous limit approach. Observe that the differences are small, and it seems like the viscous limit approach is a good approximation in this case. This result is surprising, as we expected the capillary equilibrium approach to be closer to general steady-state upscaling due to the low flow rate[2].

It should be mentioned that the results presented here are dependent on the particular model considered, and that with other models, the results may look different. We know, for instance, that test model 1 is not a REV. However, we have observed for many other models that the upscaling problem at the saturation end points are harder to solve.

Figure 6.11 displays the water saturation distribution for test model 1. In (a) we see the distribution at capillary equilibrium for an upscaled water saturation of 0.65, while in (b) we see the distribution at steady-state when a pressure drop is imposed in the x-direction. The upscaled water saturation is 0.714 at steady-state. The distributions in the two cases are almost equal, and we conclude that the capillary equilibrium is a good initial "guess".

[1] It was also intended to use periodic BCs, but due to some errors in OPM, this was not done.

[2] Based on feedback from the OPM developers, there are reasons to questioning the correctness of the current implementation of the general steady-state upscaling routine. This result should therefore not be fully trusted and verification of the code is necessary.

Table 6.3: Numerical data for the capillary equilibrium approach on test model 1. The number of iterations (IT), total iteration time (TIT) and the condition number (COND) for the different upscaled water saturations for the three different pressure drop directions (PDD) are shown. Both data from oil (a) and water (b) computations are included. All timing results are given in seconds, whereas memory usage is in megabyte.

(a) Oil.

	PDD x			PDD y			PDD z		
S_w	IT	TIT	COND	IT	TIT	COND	IT	TIT	COND
0.40	244	1.67	$2.7 \cdot 10^{16}$	267	1.81	$2.7 \cdot 10^{16}$	138	0.96	$1.4 \cdot 10^{16}$
0.45	262	1.82	$4.5 \cdot 10^{16}$	283	1.90	$4.5 \cdot 10^{16}$	146	0.93	$2.3 \cdot 10^{16}$
0.50	276	1.82	$7.1 \cdot 10^{16}$	296	1.95	$7.1 \cdot 10^{16}$	156	1.03	$3.6 \cdot 10^{16}$
0.55	281	1.88	$1.2 \cdot 10^{17}$	303	2.07	$1.2 \cdot 10^{17}$	160	1.07	$5.8 \cdot 10^{16}$
0.60	281	1.94	$1.9 \cdot 10^{17}$	305	2.08	$1.9 \cdot 10^{17}$	165	1.14	$9.6 \cdot 10^{16}$
0.65	273	1.83	$3.0 \cdot 10^{17}$	293	2.12	$3.0 \cdot 10^{17}$	167	1.10	$1.5 \cdot 10^{17}$
0.70	268	1.92	$4.3 \cdot 10^{17}$	289	1.98	$4.3 \cdot 10^{17}$	169	1.14	$2.2 \cdot 10^{17}$
0.75	260	1.76	$4.7 \cdot 10^{17}$	281	1.83	$4.7 \cdot 10^{17}$	163	1.11	$2.3 \cdot 10^{17}$
0.80	239	1.68	$8.8 \cdot 10^{15}$	260	1.84	$8.8 \cdot 10^{15}$	130	0.94	$4.4 \cdot 10^{15}$
0.85	1577	10.47	$2.8 \cdot 10^{19}$	1654	11.76	$2.8 \cdot 10^{19}$	320	2.19	$2.8 \cdot 10^{19}$

(b) Water.

	PDD X			PDD Y			PDD Z		
S_w	IT	TIT	COND	IT	TIT	COND	IT	TIT	COND
0.40	333	2.38	$2.7 \cdot 10^{19}$	295	2.10	$2.7 \cdot 10^{19}$	1207	8.51	$2.7 \cdot 10^{19}$
0.45	207	1.53	$1.6 \cdot 10^{15}$	216	1.58	$1.6 \cdot 10^{15}$	84	0.59	$1.6 \cdot 10^{15}$
0.50	211	1.37	$2.3 \cdot 10^{15}$	220	1.43	$2.3 \cdot 10^{15}$	86	0.56	$2.3 \cdot 10^{15}$
0.55	215	1.39	$2.9 \cdot 10^{15}$	227	1.49	$2.9 \cdot 10^{15}$	88	0.58	$2.9 \cdot 10^{15}$
0.60	222	1.61	$3.7 \cdot 10^{15}$	232	1.50	$3.7 \cdot 10^{15}$	91	0.63	$3.7 \cdot 10^{15}$
0.65	224	1.44	$4.5 \cdot 10^{15}$	242	1.70	$4.5 \cdot 10^{15}$	93	0.67	$4.5 \cdot 10^{15}$
0.70	232	1.66	$5.2 \cdot 10^{15}$	248	1.76	$5.2 \cdot 10^{15}$	96	0.64	$5.2 \cdot 10^{15}$
0.75	238	1.68	$5.5 \cdot 10^{15}$	257	1.86	$5.5 \cdot 10^{15}$	109	0.76	$5.5 \cdot 10^{15}$
0.80	244	1.60	$6.3 \cdot 10^{16}$	270	1.86	$6.3 \cdot 10^{16}$	148	1.00	$3.1 \cdot 10^{16}$
0.85	240	1.65	$5.5 \cdot 10^{16}$	270	1.87	$5.5 \cdot 10^{16}$	148	1.03	$2.7 \cdot 10^{16}$

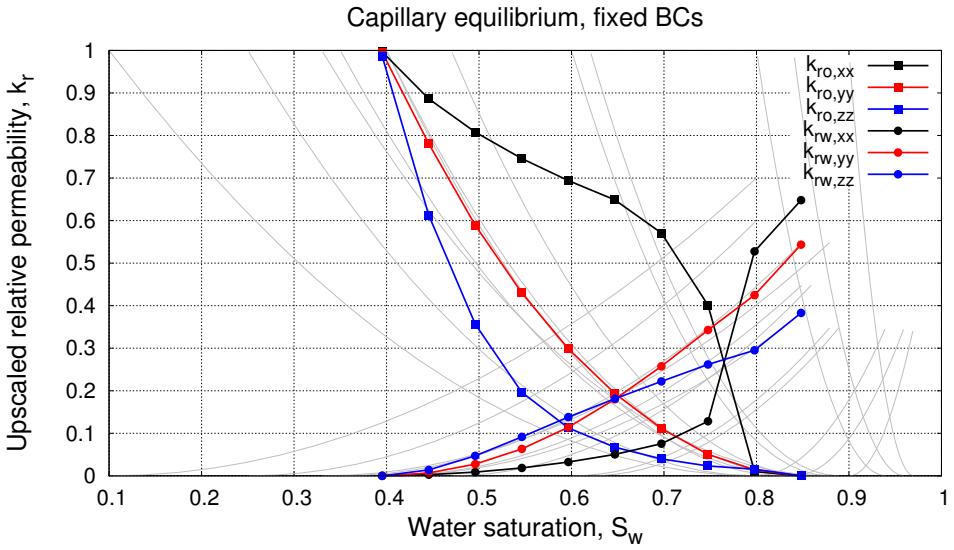


Figure 6.5: Upscaled relative permeability as function of water saturation for TM1 obtained by the capillary equilibrium approach with fixed BCs. Both oil and water curves are shown, and the input relative permeability curves are shown in light grey for reference.

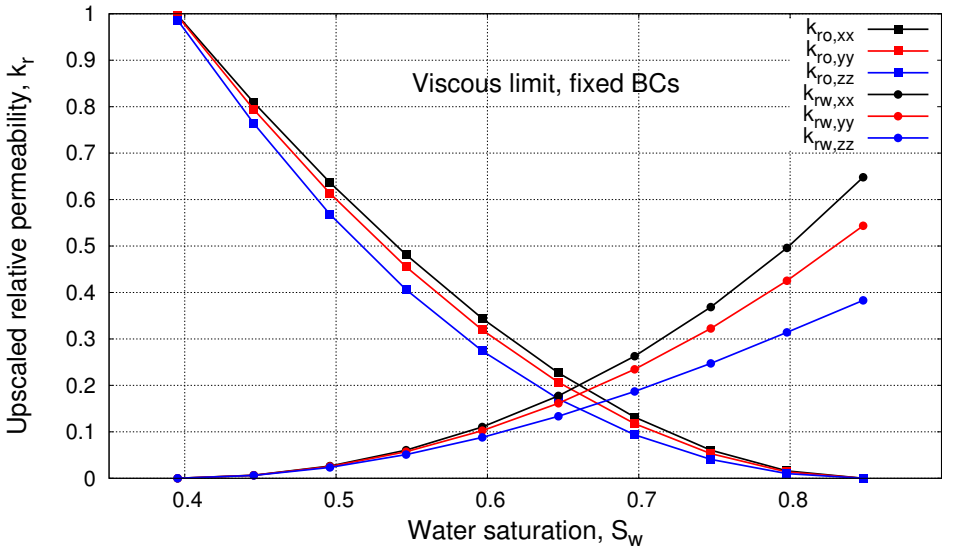


Figure 6.6: Upscaled relative permeability as function of water saturation for TM1 obtained by the viscous limit approach with fixed BCs. Both oil and water curves are shown.

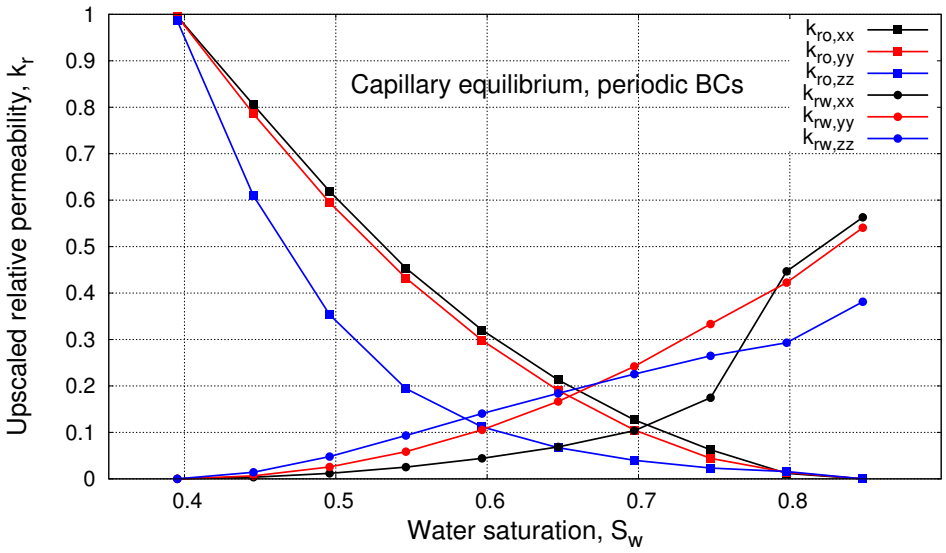


Figure 6.7: Upscaled relative permeability as function of water saturation for TM1 obtained by the capillary equilibrium approach with periodic BCs. Both oil and water curves are shown.

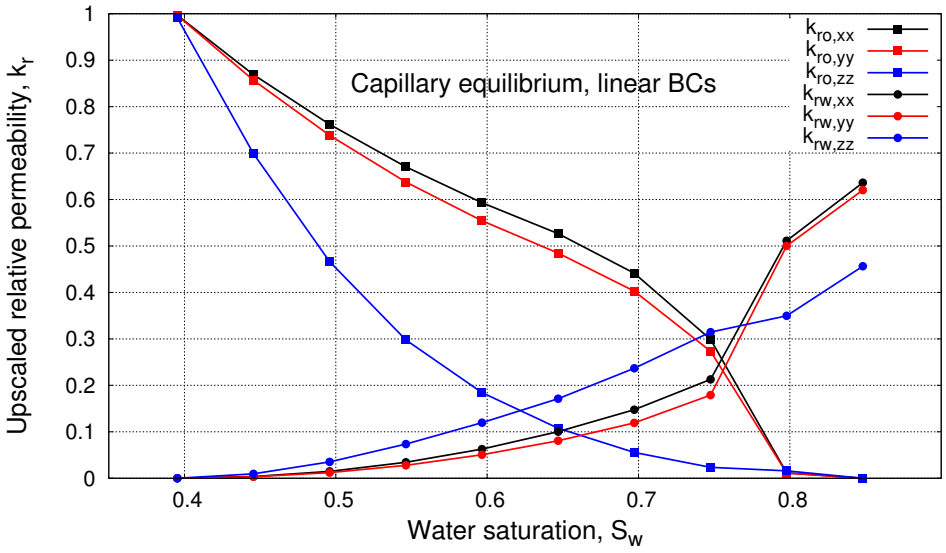


Figure 6.8: Upscaled relative permeability as function of water saturation for TM1 obtained by the capillary equilibrium approach with linear BCs. Both oil and water curves are shown.

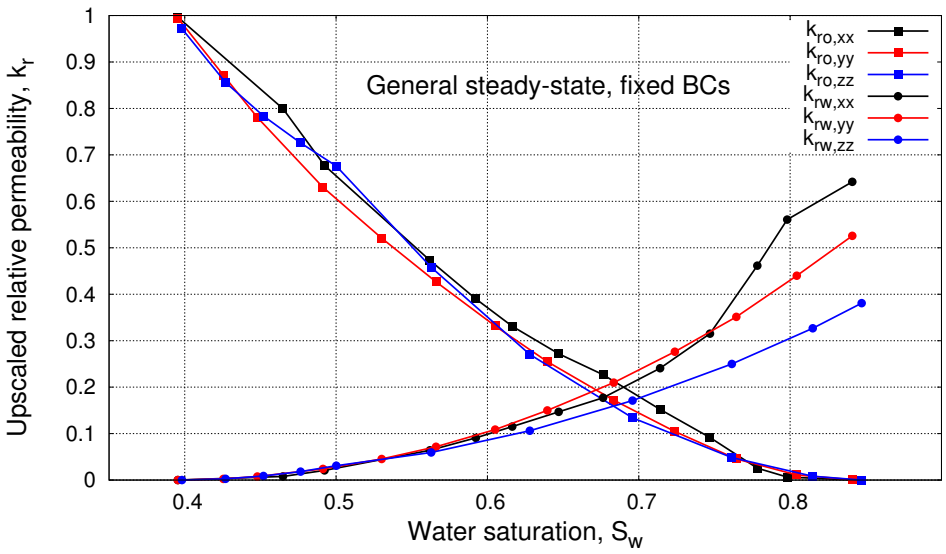


Figure 6.9: Upscaled relative permeability as function of water saturation for TM1 obtained by the general steady-state approach with fixed BCs. Both oil and water curves are shown. The steady-state saturation distribution is found by imposing a pressure drop of 1 bar in the flow direction and starting at capillary equilibrium.

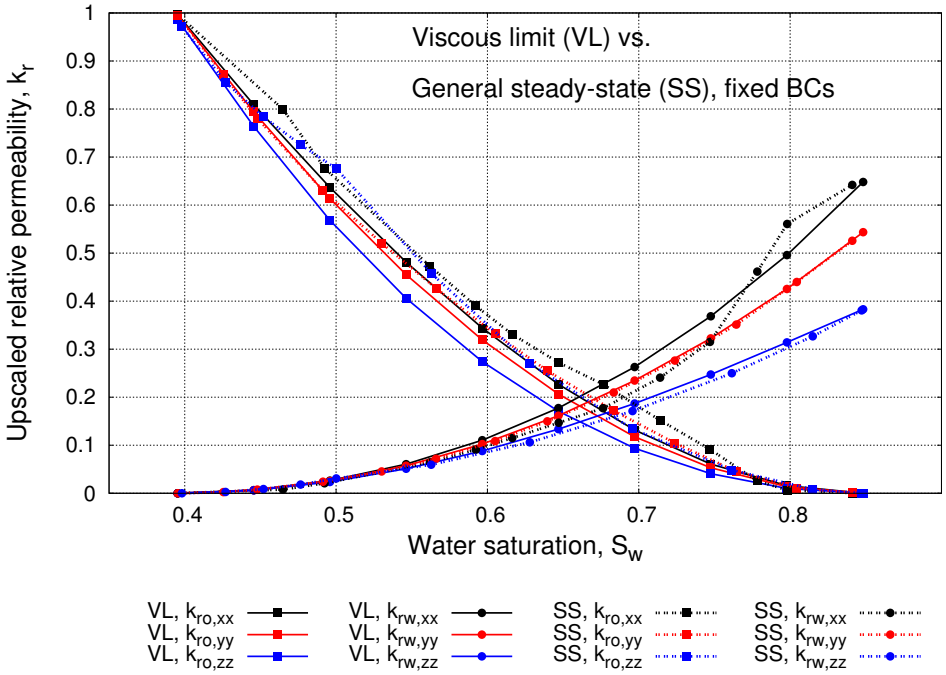
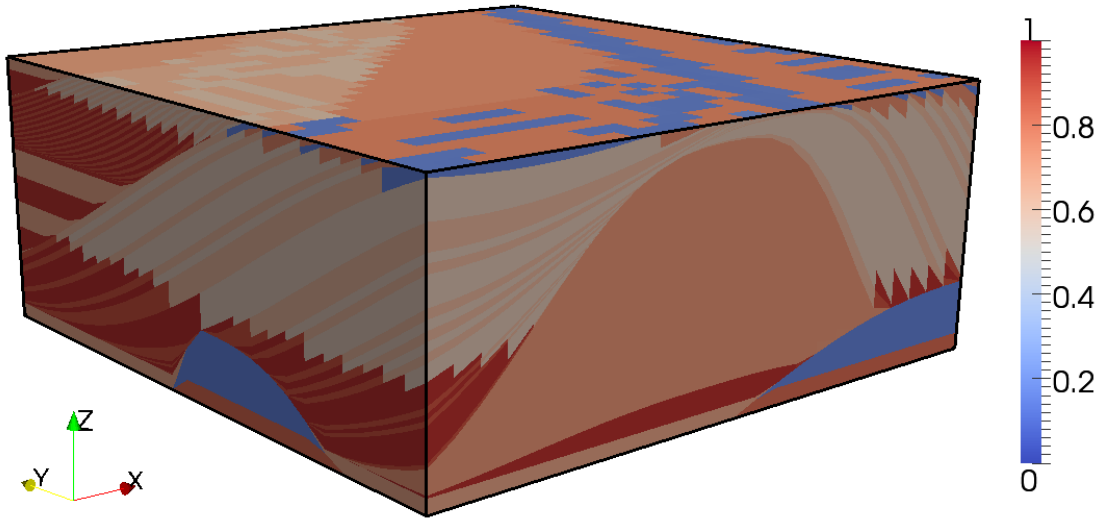
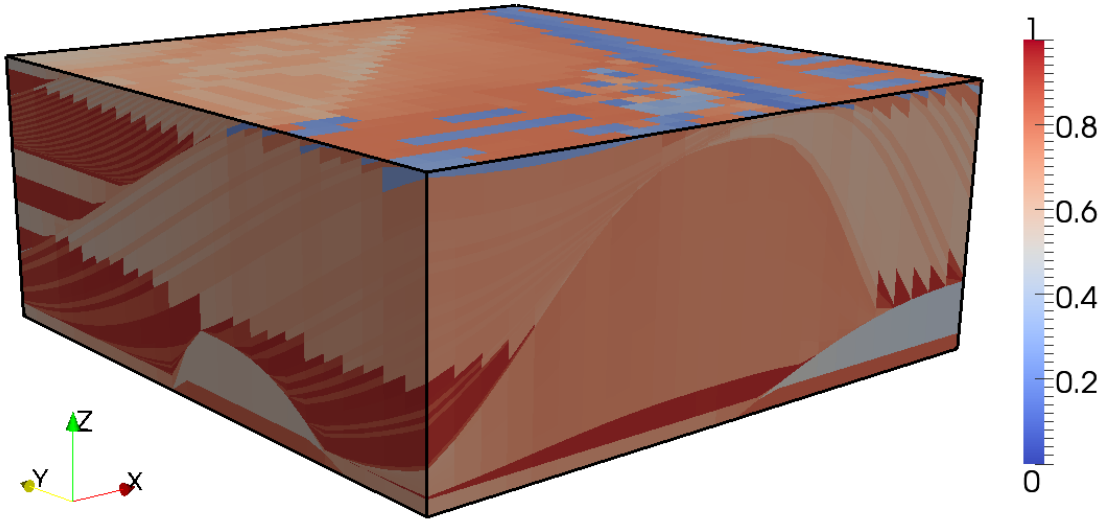


Figure 6.10: Comparison of the results from the viscous limit (VL) approach and the general steady-state (SS) approach with fixed BCs.



(a) Initial distribution (capillary equilibrium).



(b) Steady-state distribution.

Figure 6.11: Water saturation in TM1. In (a) we see the distribution at capillary equilibrium for an upscaled water saturation of 0.65. This is used as the initial guess for the general steady-state upscaling. In (b) we see the steady-state distribution when a pressure drop of 1 bar is imposed in the x-direction. The upscaled water saturation is 0.714 at steady-state. The deviations are largest in regions with high permeability (cf. Figure 6.3a), as the water saturation is lower at capillary equilibrium.

6.4 Testing the Mortar Method

In this section, we test the implementation of the mortar method derived in Section 5.3. We have seen that periodic BCs may be preferable in some cases, but at the same time we have encountered some numerical implications. This motivates a new implementation of the periodic BCs, and the aim of this section is to test whether the mortar method is better in terms of numerical convergence and memory consumption than the current implementation based on SPC.

We use SPC in the vertical direction, as the boundary grids are matching here, and the mortar method in the horizontal directions. However, we have only enforced periodicity on the interface pressures. As we will explain later, using the mortar method to enforce periodicity on the flux is not easily implemented in the OPM framework. However, we wanted to test whether the implementations are correct and if the results are correct in some cases.

Simple Uniform Homogeneous Model

We start with the simplest problem of this type and consider a constant permeability field equal to 1 on the unit cube. We use periodic BCs with a pressure drop in the x-direction. This problem can be formulated as

$$\begin{aligned} \vec{v} &= -\nabla p && \text{in } \Omega = (0,1)^3, \\ \nabla \cdot \vec{v} &= 0 && \text{in } \Omega, \\ (\vec{v}\cdot\vec{n})|_{\partial\Omega^{\xi,1}} &= -(\vec{v}\cdot\vec{n})|_{\partial\Omega^{\xi,2}} && \text{for } \xi = x,y,z, \\ p|_{\partial\Omega^{\xi,1}} &= p|_{\partial\Omega^{\xi,2}} && \text{for } \xi = y,z, \\ p|_{\partial\Omega^{x,1}} &= p|_{\partial\Omega^{x,2}} + 1. \end{aligned} \tag{6.6}$$

To complete the system, we set the pressure equal to 1 on $\partial\Omega^{x,1}$. It is easy to verify that a solution to (6.6) is

$$p(x,y,z) = 1 - x \quad \text{and} \quad \vec{v}(x,y,z) = [1,0,0]^\top.$$

Problem (6.6) was solved on a uniform $8\times8\times8$ grid, and the numerical solution is displayed in Figure 6.12. The numerical solution is verified to be equal to the analytical solution within computer precision. This is as expected since the MFDM is exact for linear pressure. We conclude that the implementations of the mortar method work for a constant permeability field on a uniform grid.

Simple Uniform Layered Model

We now introduce a model with a high permeability layer in the mid layers, i.e.,

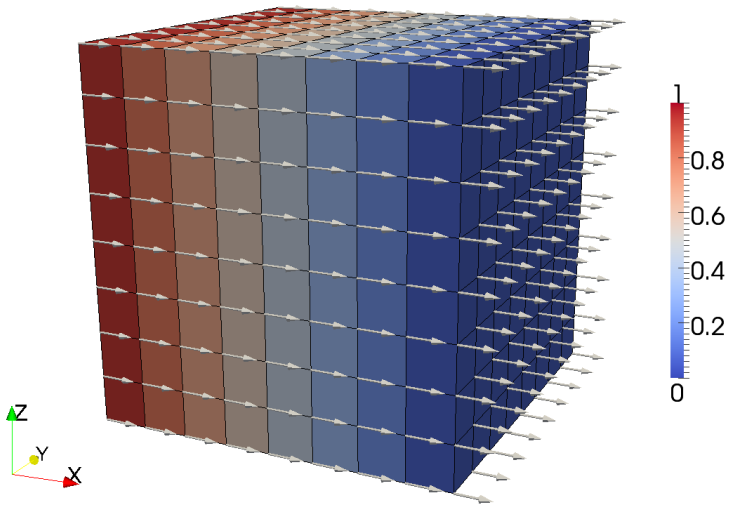


Figure 6.12: The numerical solution of problem (6.6). The cell pressures are colored, while the arrows denote the velocity field, here given at each vertex. The velocity at the vertices are calculated as the mean of the surrounding face values.

$$\mathbf{K}(x,y,z) = \begin{cases} 10 & \text{for } |z-0.5| \leq 0.25, \\ 1 & \text{elsewhere.} \end{cases}$$

The permeability field is shown in Figure 6.13. The numerical solutions from the mortar method, will be compared to the solutions from the SPC method, which is the original solver used in OPM and assumed to be correct. Recall that the problem considered are given by (3.3) and (3.10), but without gravity.

We start with imposing a unit pressure drop in the x-direction. The solutions are displayed in Figure 6.14a and 6.14b. We see that the SPC solution is exactly equal to the solution of the simple uniform homogeneous model, see Figure 6.12. This is as expected since the pressure drop is parallel to the layer, and hence the permeability at a point (x, y, z) does not change for fixed y, z. The mortar solution is significantly different and wrong – we clearly see that $p|_{\partial\Omega^{x,1}} \neq p|_{\partial\Omega^{x,2}} + 1$. However, if we integrate the pressure over the boundary faces, the mortar constraint (5.6) is satisfied. This also holds for the flux, that is, if we integrate the flux over the boundary faces, we observe that the total flow into one boundary equals the total flow out of the opposite boundary. A pressure drop in the y-direction, yields the same results, only rotated. This is due to symmetry of the model.

The numerical solutions for the problem with a pressure drop in the z-direction are shown in Figure 6.14c and 6.14d. First, observe that the pressure gradient is much smaller in the mid layer than for the solution in Figure 6.14a. This is due

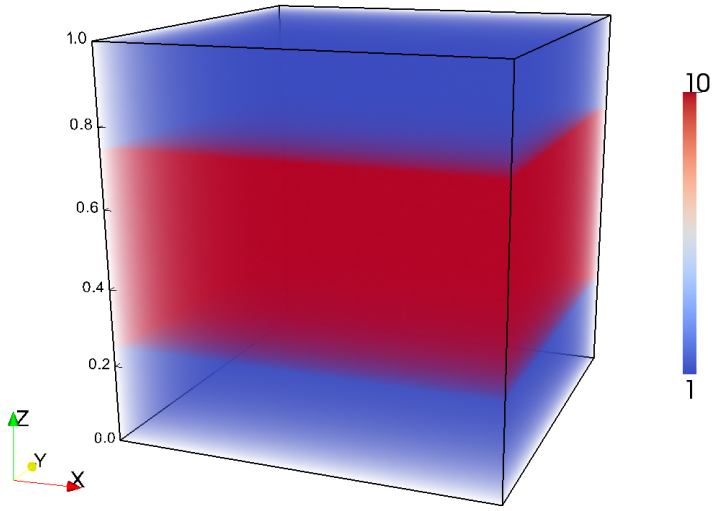


Figure 6.13: The permeability field for the simple layered model.

to the high permeability layer. Further, the two solutions are equal, and hence we conclude that the mortar method works fine with a pressure drop in the z-direction.

From these results, we conclude that the mortar method implemented in this thesis only works on very limited models. As long as the permeability field in every section perpendicular to the pressure drop is constant, it works fine, but for more general permeability fields, it fails. The mortar method only imposes that the integrals of the pressure over opposite boundary faces are equal, and does not strongly impose periodicity. This explains why the solutions differ. Hence, we can conclude that pressure must be imposed strongly.

An approach that might work, is to impose the mortar constraints (5.6) directly on the flux unknowns, i.e.,

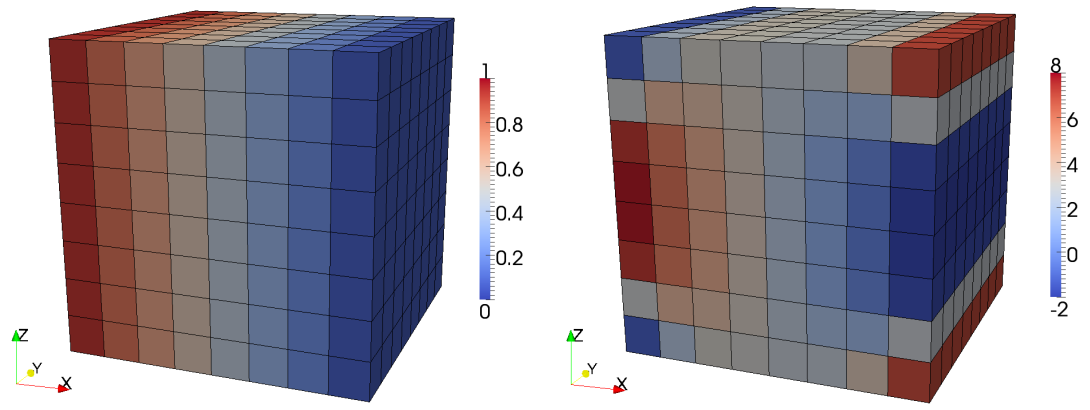
$$\int_{\partial\Omega^\xi} \lambda \left((\vec{v}\cdot\vec{n})|_{\partial\Omega^{\xi,1}} - (\vec{v}\cdot\vec{n})|_{\partial\Omega^{\xi,2}} \right) \mathrm{d}S = 0, \qquad \forall \lambda \in \Lambda, \quad \xi = x, y, z. \tag{6.7}$$

Added to the system (4.23), this would result in a linear system,

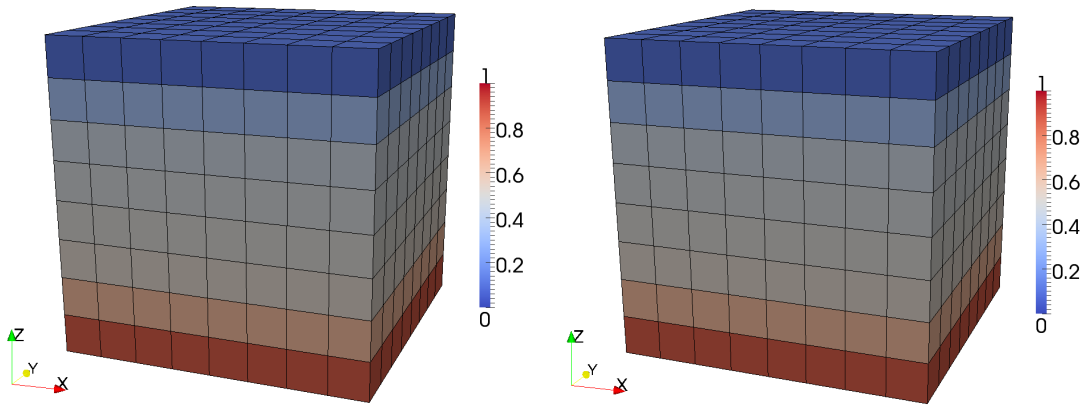
$$\begin{bmatrix} \mathbf{M} & \mathbf{C}^\top & \mathbf{D}^\top & \mathbf{L} \\ \mathbf{C} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{D} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{L}^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ -\mathbf{p} \\ \boldsymbol{\pi} \\ \boldsymbol{\ell} \end{bmatrix} = \begin{bmatrix} \mathbf{g}_D \\ \mathbf{b} \\ \mathbf{g}_N \\ \mathbf{0} \end{bmatrix}.$$

However, in the framework of OPM, this is not straightforward as the matrix in (4.23) is not assembled directly. Instead the local matrix (4.22) is calculated and reduced to (4.29) by Schur-complement reduction, before the total system (4.28) is

assembled and solved numerically. In order for the mortar method (6.7) to work, we need to add the mortar constraints to the global system (4.23). So to test if it is possible to impose mortar constraints on the flux, it is necessary to build a new flow solver almost from scratch. This would have been way beyond the scope of this thesis. Another approach is to use the MPC method, but the same problem is encountered.



(a) SPC solution, PDD x. **(b)** Mortar solution, PDD x.



(c) SPC solution, PDD z. **(d)** Mortar solution, PDD z.

Figure 6.14: Numerical pressure solution on a uniform $8 \times 8 \times 8$ grid of the simple layered model displayed in Figure 6.13. Solutions for pressure drop direction (PDD) x and z with both SPC and Mortar methods are shown. The solutions for PDD y is similar (only rotated) to PDD x.

Realistic Test Model

Finally, we have tested the mortar implementations on a more realistic model, see Figure 6.15. The aim is to document the numerical implications. We have run the single-phase upscaling procedure with periodic BCs on this model with both the SPC and the mortar methods. The results are shown in (6.8).

$$\begin{aligned} \tilde{\mathbf{K}}_{\text{mortar}} &= \begin{bmatrix} 5705 & -30.06 & 0.2287 \\ -86.26 & 4899 & 0.5921 \\ -181.9 & -48.31 & 2.895 \end{bmatrix}, \\ \tilde{\mathbf{K}}_{\text{SPC}} &= \begin{bmatrix} 106.4 & -0.1435 & -0.006907 \\ -0.1434 & 110.2 & -0.01295 \\ -0.006907 & -0.01296 & 2.280 \end{bmatrix}. \end{aligned} \tag{6.8}$$

We understand from these results that the mortar method is not working properly. However, it is observed that both the pressure and the flux integrals over opposite boundary faces are equal. Hence, we conclude that the implementations are correct.

To test the numerical convergence of the two methods, the number of iterations (IT), total iteration time (TIT), memory usage (MEM) and an estimate on the condition number (COND) where measured. ILU0/CG were chosen as the linear solver for the SPC method, while ILU0/Bi-CGSTAB were used for the mortar method (cf. discussion in Section 5.3). The results for different PDD are shown in Table 6.4. Further, the sparsity pattern of the system matrices are displayed in Figure 6.16. The sparsity patterns are almost equal, expect for the entries associated with unknowns on the boundary. The mortar matrix seems to be a little less dense and have fewer unknowns in total. Apart from this, we see from Table 6.4 that the mortar method is significantly worse conditioned than the SPC method, and that the SPC method is faster and uses less memory.

The results from this section shows that the mortar method considered in this thesis is correct implemented, but fails both in terms of correctness and in terms of numerical convergence.

Table 6.4: Numerical data for the realistic test model (Figure 6.15).

	Mortar				**SPC**			
PDD	IT	TIT	MEM	COND	IT	TIT	MEM	COND
X	270	0.61	38	Inf	121	0.16	22	$1.2 \cdot 10^{19}$
Y	302	0.65	38	Inf	123	0.16	22	$1.2 \cdot 10^{19}$
Z	255	0.56	38	Inf	115	0.15	22	$1.2 \cdot 10^{19}$

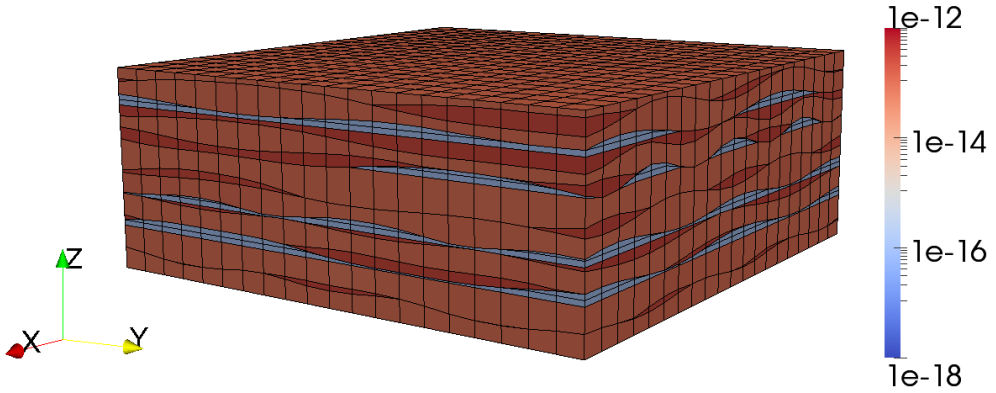


Figure 6.15: The model used for testing the mortar method. Three different isotropic materials are present, two high permeability materials with average permeabilities 300 mD and 100 mD respectively, and a low permeability material with average permeability 0.1 mD. The average porosities are 0.3, 0.2 and 0.01 respectively. The permeability field in m^2 is shown on a logarithmic scale.

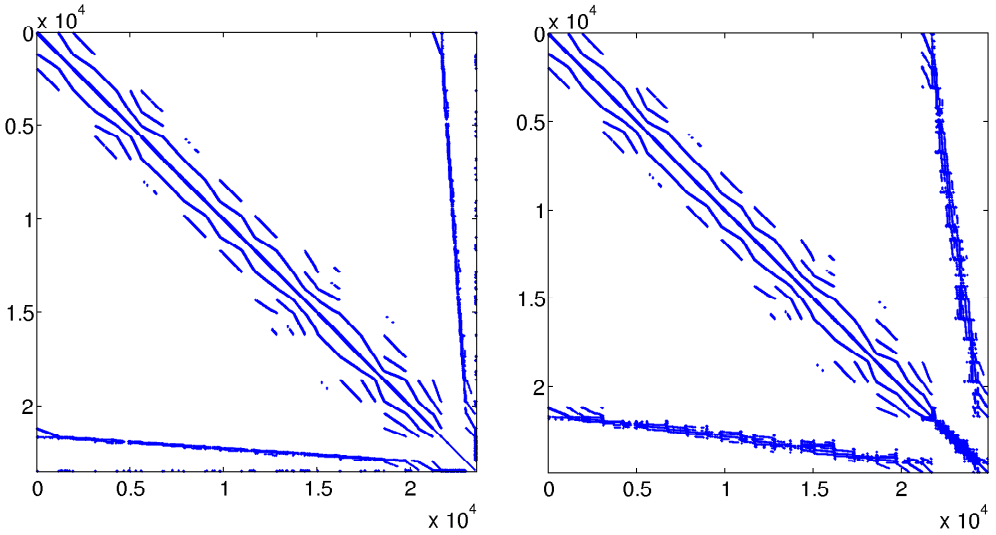


Figure 6.16: Sparsity pattern for the system matrices associated with the mortar method (left) and the SPC method (right), see equations (5.10) and (4.28) respectively.

Chapter 7

Concluding Remarks

We now try to summarize the results of this thesis. We have given a basic introduction to reservoir simulation and upscaling in particular. Upscaling was seen in relation to the notion of a representative elementary volume (REV), and we argued why flow based local upscaling methods can be preferable in this context.

Different traditional discretization methods for solving the elliptic diffusion equation has been introduced, in particular the MFDM. The main advantage of the MFDM is that it is easy to implement on corner-point grids that are non-conforming and contain degenerated cells. The MFDM was explained in some details in order to fully understand how periodic BCs are implemented and to be able to derive and implement a new representation of periodic BCs.

We have documented that flow based local upscaling is highly dependent on the BCs. This was the case for both (absolute) permeability and relative permeability. We have tried to analyze the differences both in terms of correctness and in terms of numerical convergence. Linear BCs were shown to have faster convergence, that is, requiring less iterations of the numerical solver and hence reducing the computation time. But linear BCs tend to avoid flow barriers, and thus overestimate the permeability in directions with flow barriers. This is often encountered in the vertical direction due the the geology of reservoirs.

Fixed BCs enforce no flow out of boundary faces parallel to the pressure drop. This seems rather artificial and results in diagonal upscaled permeability tensors. Thus, off-diagonal terms are neglected, and this may be wrong in many cases, especially in models where there exists flow channels not parallel to the pressure drop. Fixed BCs are also very sensitive to flow barriers, and may underestimate flow in such cases.

Periodic BCs seems intuitively most correct in the REV framework. But as observed, permeability may be underestimated in situations where the model is not periodic. It might seem like periodic BCs are best in cases where the layers are aligned with the xy-plane. This choice is often done in reservoir modeling.

Claim 1 (p. 33) states that fixed and linear BCs represent lower and upper bounds on the upscaled permeability respectively. This is a nice result, because it lets us create an interval on which the true upscaled permeability is contained. However, we have created an counter-example, on which the claim fails. This shows that the claim is not valid under all circumstances.

In terms of numerical convergence, we have seen that the problems with fixed and periodic BCs are hard to solve compared to using linear BCs. The implementation of periodic BCs was tried improved by using a variant of mortar methods, but without success.

7.1 Further Work

This thesis gives an overview of the status of upscaling as of today, and it enlightens many of the problems that upscaling face. A natural extension of this thesis would be to dig deeper into why linear BCs are much faster to solve for. As suggested here, this may have a physical argument based on more freedom for the fluid flow. Mathematically, we have shown that linear BCs result in a system matrix with a nicer sparsity pattern that for the other BCs. This has to do with the current implementation, and another representation might improve the numerical convergence for fixed and periodic BCs.

The attempt to improve the implementation of periodic BCs that was tested in this thesis was not successful. We believe, however, that the work done here can be built upon later, and that a further study may reveal a variant that is better. But, as pointed out, this may require to change the way the assembly process is structured and implemented in OPM. This is probably beyond the scope of a single Master's thesis.

A third aspect of this thesis that would be interesting to examine further, is the relation between the three two-phase upscaling approaches described in Section 3.4. The capillary equilibrium approach is expected to be a good approximation in regions with low flow rates, while the viscous limit approach is expected to be a good approximation in regions with high flow rate. But how fast does general steady-state upscaling converge to these limits as the flow rate either decrease or increase? As mentioned, general steady-state upscaling is computationally very demanding, so if either the capillary equilibrium approach or the viscous limit approach are close enough to general steady-state upscaling, this would reduce the computation time considerably. Before this work is started, a verification of the general steady-state upscaling routine provided by OPM must be performed.

Appendix A

Source Code and Model Data

The Open Porous Media (OPM) project is an important part of this thesis, as it is used for all numerical computations. In this appendix we give a short introduction to OPM and provide information on how the source code can be obtained and instructions for building it. Further, we list the OPM routines used in this thesis and explain were the implementations done in this thesis are located. Lastly, we describe how the models used in this thesis can be obtained. The main purpose of this appendix is to make it easier for the reader to reproduce the results of this thesis.

A.1 What is OPM?

OPM[1] is an open source software suite for porous media flow under the terms of the GNU General Public License (GPL)[2] (version 3). All source code is available as git repositories on GitHub, visit `https://github.com/OPM`. The source code can be viewed directly on this link or downloaded (see intructions in Section A.2). OPM has several contributors from both universities, industry and research organizations. It aims for being flexible and it should be relatively easy to use and contribute to it. This, and that it is open source, is of great importance in trying to lower the gap between researchers and the industry.

OPM is built on DUNE[3] (the Distributed and Unified Numerics Environment), which is a C++ based software toolbox for solving PDE's numerically. DUNE is also open source and licensed under the GNU GPL (version 2) with a so called "run-time exception". It is grid-based and supports both finite elements, finite volumes and finite differences. DUNE is built up of modules and contains the fundamental

[1] `http://opm-project.org`
[2] `http://www.gnu.org/licenses/gpl.html`
[3] `http://www.dune-project.org`

routines needed to build a PDE solver, e.g., grid implementations, reference elements, shape functions, sparse vectors and matrices, assemblers, iterative linear solvers etc.

OPM is also modular. The four modules used in this thesis are described below:

- `opm-core` is the base module and provides generic infrastructures.
- `dune-cornerpoint` provides routines for building and using corner-point grids.
- `opm-porsol` contains different flow solvers, e.g., the MFDM.
- `opm-upscaling` contains local upscaling methods, both averaging methods and flow based methods.

A.2 Building DUNE and OPM

This section provides instructions for downloading and building DUNE and OPM. The instructions are based on a Linux operating system, and we use DUNE's own building system, *dunecontrol*[4]. The following packages and libraries are prerequisites that must be installed in advance; svn, git, g++, gcc, gfortran, automake, autoconf, libtool, pkg-config, blas, lapack, boost-system, boost-filesystem, boost-date-time, boost-test and superlu.

DUNE is given as svn repositories. To obtain the DUNE modules (we use the stable 2.2 release in this thesis), issue the following commands in a terminal:

```
# svn co https://svn.dune-project.org/svn/dune-common/branches/release-2.2
  dune-common
# svn co https://svn.dune-project.org/svn/dune-geometry/branches/release-2.2
  dune-geometry
# svn co https://svn.dune-project.org/svn/dune-grid/branches/release-2.2
  dune-grid
# svn co https://svn.dune-project.org/svn/dune-istl/branches/release-2.2
  dune-istl
```

The OPM modules can be obtained by the following commands:

```
# git clone https://github.com/OPM/opm-core.git
# git clone https://github.com/OPM/opm-porsol.git
# git clone https://github.com/OPM/opm-upscaling.git
```

[4]Recent updates make it possible to also use *cmake* for building. Debian packages are also under construction.

```
# git clone https://github.com/OPM/dune-cornerpoint.git
```

OPM is in constant progress. In this thesis the following revisions are used:

```
opm-core          @ 8741c93a1a
dune-cornerpoint  @ 7de40f2a9a
opm-porsol        @ 99f42af258
opm-upscaling     @ bf5f8c56d2
```

To checkout a specific revision, e.g., for `opm-core`, issue

```
# cd opm-core
# git checkout 8741c93a1a
```

If all DUNE and OPM modules are downloaded, the command `ls` should produce

```
dune-common       dune-grid  opm-porsol
dune-cornerpoint  dune-istl  opm-upscaling
dune-geometry     opm-core
```

You are now ready to build[5]:

```
# ./dune-common/bin/dunecontrol all
```

A.3 Upscaling Routines

If you have successfully built all modules, you will find the upscaling routines in `opm-upscaling`. The routines used in this thesis are:

- `opm-upscaling/examples/upscale_perm`.
  Single-phase upscaling.
- `opm-upscaling/examples/upscale_relperm`.
  Two-phase upscaling at capillary equilibrium.
- `opm-upscaling/examples/upscale_relpermvisc`.
  Two-phase upscaling in the viscous limit.
- `opm-upscaling/dune/upscaling/test/steadystate_test_implicit`.
  General steady-state upscaling.

Explanation of use are printed if calling a routine without arguments, e.g.,

```
# ./opm-upscaling/examples/upscale_perm
```

[5]See `www.dune-project.org/doc/buildsystem/buildsystem.pdf` for a complete tutorial on the DUNE build system.

Visit for instance `https://public.ict.sintef.no/openrs/wiki/Main_Page` or `www.opm-project.org` for more documentation.

A.4 Implementation of the Mortar Method

The implementation of the mortar method is available as forks of the modules `opm-porsol` and `opm-upscaling`. The source code can be viewed directly at `https://github.com/laods`, where the repositories are found. The mortar method is implemented in the branch *mortar*. The code can also be checked out (requires that the steps in A.2 are fulfilled):

```
# cd opm-porsol
# git pull https://github.com/laods/opm-porsol.git mortar
# git checkout bc25a6330a
# cd ../opm-upscaling
# git pull https://github.com/laods/opm-upscaling.git mortar
# git checkout aade33bfe6
```

The implementations done in this thesis are primarily found in the directory `opm-porsol/dune/porsol/mortar/`, and especially the file `mortar.hpp` therein. The implementations are compatible with revision `9571b30556` of `opm-core` and revision `7de40f2a9a` of `dune-cornerpoint`. Thus, these need to be checked out, see instructions above. After changing the revision, you will have to run `dunecontrol` again:

```
# dune-common/bin/dunecontrol --only=opm-core,dune-cornerpoint,opm-porsol,
  opm-upscaling make
```

The routines implemented or used in this thesis are:

- `opm-porsol/examples/mortar_mimetic_periodic_test`.
  Solve elliptic diffusion equation with periodic BCs using the MFDM and mortar couplings.
- `opm-porsol/examples/mortar_mimetic_uniform_test`.
  Solve elliptic diffusion equation on a uniform grid using the MFDM and mortar couplings.
- `opm-porsol/examples/mimetic_periodic_test`.
  Solve elliptic diffusion equation with periodic BCs using the MFDM and SPC couplings.

- `opm-upscaling/examples/upscale_perm_mortar_test`.
  Single-phase upscaling using the MFDM and mortar couplings if periodic BCs.

A.5 Model Data

All models used in this thesis are free for anyone to use. A zip-file (`models.zip`) containing all models is attached to this thesis. If the zip-file is not received, it can be downloaded from this link: `http://ubuntuone.com/3sFtyARJ5p0v7jXEau2TxG`. The contents of the zip-file is shown in Table A.1.

Table A.1: Contents of `models.zip` sorted for each section.

Section	Model name	File name
6.1	Simple uniform layered model	simpleLayeredUniform.grdecl
	Non-periodic tilted model	non-periodicTilted.grdecl
	Periodic tilted model	periodicTilted.grdecl
6.2	Test model 1 (TM1)	testModel1.grdecl
	Test model 2 (TM2)	testModel2.grdecl
6.3	List of rock files for TM1	file_list.txt
	Rock files containing input curves for TM1	rock_files/
6.4	Realistic test model	mortarTestModel.grdecl

Remark 1: The rock files for TM1 are also valid for TM2.

Remark 2: The uniform cartesian models referred to as *simple uniform homogeneous model* and *simple uniform layered model* in Section 6.4 were constructed by routines provided by `dune-cornerpoint`, see for instance `opm-porsol/examples/mortar_mimetic_uniform_test.cpp`.

Bibliography

[1] Aarnes, J. E., Gimse, T., and Lie, K.-A. An Introduction to the Numerics of Flow in Porous Media using Matlab. In *Geometric Modelling, Numerical Simulation, and Optimization*, G. Hasle, K.-A. Lie, and E. Quak, Eds. Springer Berlin Heidelberg, 2007, pp. 265–306.

[2] Aarnes, J. E., Kippe, V., Lie, K.-A., and Rustad, A. B. Modelling of Multiscale Structures in Flow Simulations for Petroleum Reservoirs. In *Geometric Modelling, Numerical Simulation, and Optimization*, G. Hasle, K.-A. Lie, and E. Quak, Eds. Springer Berlin Heidelberg, 2007, pp. 307–360.

[3] Aarnes, J. E., Krogstad, S., and Lie, K.-A. Multiscale Mixed/Mimetic Methods on Corner-Point Grids. *Comp. Geosciences 12*, 3 (2008), 297–315.

[4] Aavatsmark, I. An Introduction to Multipoint Flux Approximations for Quadrilateral Grids. *Comp. Geosciences 6*, 3–4 (2002), 405–432.

[5] Arbogast, T., Cowsar, L. C., Wheeler, M. F., and Yotov, I. Mixed Finite Element Methods on Nonmatching Multiblock Grids. *SIAM J. Num. Anal. 37*, 4 (2000), 1295–1315.

[6] Bear, J. *Dynamics of fluids in porous media.* Dover, New York, 1988, p. 19.

[7] Berndt, M., Lipnikov, K., Shashkov, M., Wheeler, M. F., and Yotov, I. A Mortar Mimetic Finite Difference Method on Non-Matching Grids. *Numerische Mathematik 102*, 2 (2005), 203–230.

[8] Brezzi, F., Lipnikov, K., and Shashkov, M. Convergence of Mimetic Finite Difference Method for Diffusion Problems on Polyhedral Meshes. *SIAM J. Num. Anal. 43* (2005), 1872–1895.

[9] Brezzi, F., Lipnikov, K., and Simoncini, V. A family of mimetic finite difference methods on polygonal and polyhedral meshes. *Math. Models Methods Appl. Sci. 15* (2005), 1533–1553.

[10] Durlofsky, L. J. Upscaling and Gridding of Fine Scale Geological Models for Flow Simulation. In *Proceedings of the 8th International Forum on Reservoir Simulation, Stresa, Italy* (June 20–24, 2005), Stanford University.

[11] Ekrann, S., and Aasen, J. O. Steady-State Upscaling. *Transport in Porous Media 41*, 3 (2000), 245–262.

[12] Evans, L. C. *Partial Differential Equations*, second ed. American Mathematical Society, Providence, Rhode Island, 2010, pp. 255–263, 311–315, 702.

[13] Higham, N. J. Fortran Codes for Estimating the One-Norm of a Real or Complex Matrix, with Applications to Condition Estimation. *ACM Trans. Math. Soft. 14* (1988), 381–396.

[14] King, M. J., MacDonald, D. G., Todd, S. P., and Leung, H. Application of Novel Upscaling Approaches to the Magnus and Andrew Reservoirs. In *European Petroleum Conference, The Hague, Netherlands* (October 20–22, 1998), pp. 133–148.

[15] King, M. J., and Mansfield, M. Flow Simulation of Geologic Models. *SPE Reservoir Evaluation & Engineering 2* (1999), 351–367.

[16] Lipnikov, K., Shashkov, M., and Svyatskiy, D. The mimetic finite difference discretization of diffusion problem on unstructured polyhedral meshes. *J. Comput. Phys. 211* (2005), 473–491.

[17] Nilsen, H. M., Lie, K.-A., and Natvig, J. R. Accurate Modeling of Faults by Multipoint, Mimetic, and Mixed Methods. *SPE Journal 17*, 2 (2012), 568–579.

[18] Nocedal, J., and Wright, S. J. *Numerical Optimization*, second ed. Springer, 2006, pp. 10–18, 448–454.

[19] Nordahl, K., and Ringrose, P. S. Identifying the Representative Elementary Volume for Permeability in Heterolithic Deposits Using Numerical Rock Models. *Mathematical Geosciences 40*, 7 (2008), 753–771.

[20] Ponting, D. K. Corner Point Geometry in Reservoir Simulation. In *Proceedings of the 1st European Conferance on Mathematics of Oil Recovery* (1989), P. King, Ed., Oxford, pp. 45–65.

[21] Saad, Y. *Iterative Methods for Sparse Linear Systems*, 2nd ed. SIAM Philadelphia, 2003, pp. 203–205, 287–296, 437–445.

[22] VAN DER VORST, H. A. Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems. *SIAM J. Sci. and Stat. Comput. 13*, 2 (1992), 631–644.

[23] WIKIPEDIA. Corner-point grid.
`http://en.wikipedia.org/wiki/Corner-point_grid` (02.01.2013).