AN ABSTRACT OF THE THESIS OF

Cristiano Rodrigues Garibotti for the degree of Master of Science in Mathematics presented on March 8, 2007.

Title: Upscaling Non-Darcy Flow Using Mixed Finite Element Method .

Abstract approved: ______________________________

Malgorzata Peszyńska

In this paper we develop an upscaling technique for non-Darcy flow in porous media. Non-Darcy model of flow applies to flow in porous media when large velocities occur. The well-posedness results for theory of quasilinear elliptic partial differential equations. To discretize the model we used lowest order Raviart-Thomas mixed finite element spaces. The resulting non-linear system is solved using fixed point iteration; we provide sufficient conditions for this iteration to converge. Then we formulate an upscaling method for non-Darcy flow extending a method by Durlofsky originally given for Darcy flow. The method computes effective coefficients which can be used to simulate the Darcy flow on a coarse grid. We compare numerical results for Darcy and non-Darcy flow and upscaling results for two scenarios; in all cases results apply to a problem with rate-specified wells.

Upscaling Non-Darcy Flow Using Mixed Finite Element Method

by

Cristiano Rodrigues Garibotti

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented March 8, 2007
Commencement June 2007

Master of Science thesis of Cristiano Rodrigues Garibotti presented on March 8, 2007

APPROVED:

Major Professor, representing Mathematics

Chair of the Department of Mathematics

Dean of the Graduate School

Cristiano Rodrigues Garibotti, Author

ACKNOWLEDGEMENTS

Academic

I am indebted to Dr. Malgorzata Peszynska for the time and patience she provided me and for her valuable advice and guidance.

Research in this paper was partially supported by National Science Foundation, grant DMS-0511190 “Model Adaptivity in Porous Media”, under the direction of Malgorzata Peszynska (Principal Investigator).

Personal

I wish to thank to my parents, Heron and Maria do Carmo, for all of the support they have given me. Special thanks to my brothers, Alessandro and Daniel, and to my sister, Viviane, for the incentive and emotional support they have provided me.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

UPSCALING NON-DARCY FLOW USING MIXED FINITE ELEMENT METHOD

1. INTRODUCTION

"A porous medium is a heterogeneous material consisting of a solid matrix and a pore space contained therein"[15]. Porous media occur in many areas of applied science and engineering: mechanics (acoustics, geomechanics, soil mechanics, rock mechanics), engineering (petroleum engineering, construction engineering), geosciences (hydrogeology, petroleum geology, geophysics), biology and biophysics, material science, etc. Fluid flow through porous media is a subject of most common interest and has emerged a separate field of study.

In this paper we focus on applications in the geosciences, more specifically in modeling and simulation of fluid flows in petroleum reservoirs. The goals of this paper are to present a unified technique for solving the pressure equation that arises from the Darcy's [1] (linear) and non-Darcy's (non linear) law for single phase flow, and to use it in *upscaling*. Darcy's equations applies to a fluid flowing at low velocity, which linearly correlates pressure drop and velocity. Non-Darcy's flow is governed by the Forchheimer[2] equations

[1]Henry Philibert Gaspard Darcy (1803 to 1858) was a French engineer. He invented the modern style Pitot tube, was the first researcher to suspect the existence of the boundary layer in fluid flow, contributed in the development of the Darcy-Weisbach equation for pipe flow resistance, made major contributions to open channel flow research and of course developed Darcy's Law for flow in porous media. His Law is a foundation stone for several fields of study including ground-water hydrology, soil physics, and petroleum engineering. `http://biosystems.okstate.edu/darcy/`

[2]Philipp Forchheimer (1852 to 1933): Austrian hydraulic engineer who made significant studies of groundwater hydrology. Early in his academic career, he worked on problems of soil mechanics. Later, he turned to hydraulic problems, establishing the scientific basis of the discipline by applying standard techniques of mathematical physics - in particular Laplace's equation - to problems of groundwater movement. Laplace's equation had already been well developed for heat

for which the gradient of pressure and its velocity are nonlinearly related.

This paper is composed of seven chapters as follows. In the second chapter of this paper we present the physical model for both Darcy's and non-Darcy's flows and give results on well-posedness of the models. We also present the mixed variational form of Darcy's flow that will be used later to show the equivalence between cell-centered finite difference and mixed finite element formulations on rectangular grid.

In Chapter 3 we apply the cell-center finite difference method to solve the pressure equation that arrives from Darcy's formulation. We also describe the mixed finite element formulation and its equivalence to the cell-centered finite difference method when we use certain quadrature rules on a rectangular grid. The boundary conditions are treated separately for three different types: no-flow(Newman), Dirichlet and periodic. We also discuss the implementation of production and injection wells. The chapter ends with a brief explanation on the solution of the linear system.

Chapter 4 is used to extend the cell-center finite difference method to the non-Darcy's model and to present the nonlinear solver chosen to deal with the nonlinear difference equation generated for the Forchheimer equation. We make use of the fixed point method to deal with the non-linearities of the discrete non-Darcy model. The fixed point theory is briefly described and sufficient conditions for the convergence of the fixed point method are derived.

In Chapter 5, we describe a method of calculating *effective grid block permeability.* This application arrives from the fact that we need to adapt highly detailed geological models to computational grids. Due the difference between these two scales we need to scale up some of the microscale rock properties (permeability in our case) to be used in a coarse grid simulation. This process is called *upscaling.* The theory for *upscaling* has been

flow and fluid flow. Forchheimer extended the preexisting mathematical theory to calculations of groundwater flow. He was also the first to both mathematically and experimentally examine the features of dambreak waves in a rectangular channel (with his PhD student Armin Schoklitsch). `http://www.todayinsci.com/cgi-bin/indexpage.pl?http://www.todayinsci.com/8/8_07.html` developed in [8] for Darcy flow. Here we extend these ideas to the non-Darcy flow and discuss the consequences of the nonlinear behavior of the equation to the calculation of the effective permeability. We end the chapter by showing how to calculate the *effective Forchheimer parameter* β.

Chapter 6 is dedicated to numerical experiments. We simulate first a 2-injection/2-production well model in order to compare the solutions for Darcy's and non-Darcy's problems. Next, we calculate the effective permeability on a square domain for two distinct scenarios. Then the upscaled values for permeabilities are used to simulate a 1-injection/1-production well model for the two distinct scenarios. A method of comparison of the upscaled solution on coarse grid to the one on fine grid is developed.

Chapter 7 contains conclusions from this work and lists future work. The code used in the thesis is attached in the Appendix.

2. PHYSICAL MODEL AND ANALYSIS

Porous media appear in nature and manufactured materials. Soils and aquifers are examples in geosciences; porous catalysts, concrete, ceramics, moisture absorbabsorbentsants are important in chemical engineering. Even the human skin and the placenta can be considered porous media. We are interested in modeling and simulation of fluid flows in both petroleum and groundwater reservoirs. In this chapter we present the basic equations that describe the flow of a fluid in porous media using terminology referring to natural soil as porous medium. We start with a linear model created by H. Darcy in 1856 [6] and extend it to a nonlinear model, known as generalized Forchheimer model [11]. We also present the variational form of Darcy's flow.

2.1. Darcy Flow

Consider the system of partial differential equations representing the two physical principles

$$\mathbf{u} = -\mathbf{K}(\mathbf{x})(\nabla p - \rho \mathbf{g}), \quad \mathbf{x} \in \Omega \quad (\text{Darcy}'\text{s law}), \tag{2.1}$$

$$\phi \frac{\partial \rho}{\partial t} + \operatorname{div}(\rho \mathbf{u}) = \overline{q}, \quad \mathbf{x} \in \Omega \quad (\text{Conservation of mass}), \tag{2.2}$$

where Ω is a open bounded domain in two- or three-space, p is the pressure, ρ is the fluid density, $\mathbf{g}$ is the gravitational vector, $\mathbf{u}$ is the volumetric flow rate (or velocity) of the fluid, ϕ is the porosity of the medium, and $\overline{q}$ is an external mass flow rate, and $\mathbf{K}$ is the permeability tensor. In two dimensions, in the $x - y$ coordinate system, $\mathbf{K}$ is represented as

$$\mathbf{K} = \begin{pmatrix} K_{xx} & K_{xy} \\ K_{yx} & K_{yy} \end{pmatrix}. \tag{2.3}$$

$\mathbf{K}(\mathbf{x})$ is possibly discontinuous but bounded below and above by positive constants and is symmetric and uniformly positive definite.

In this paper we assume that the fluid is incompressible, i.e., ρ is constant, and so equation (2.2) becomes

$$\mathrm{div}(\mathbf{u}) = q, \quad \mathbf{x} \in \Omega \tag{2.4}$$

where from now on we use $q = \overline{q}/\rho$ Combining equations (2.1) and (2.4) we get

$$-\nabla \cdot (\mathbf{K}(\mathbf{x})(\nabla p - \rho \mathbf{g})) \;\equiv\; \nabla \cdot \mathbf{u} = q \qquad \mathbf{x} \in \Omega, \tag{2.5}$$

We usually assume no-flow boundary conditions in reservoir modeling

$$(\mathbf{K}(\mathbf{x})\nabla p) \cdot \nu \;\equiv\; \mathbf{u} \cdot \nu = 0 \qquad \mathbf{x} \in \partial\Omega. \tag{2.6}$$

But, in order to deal with some upscaling problems in this work, we also consider Dirichlet boundary conditions of the form

$$p = g(\mathbf{x}) \qquad \mathbf{x} \in \partial\Omega,$$

or periodic boundary conditions. If the no-flow, or Neumann boundary conditions, are prescribed, the pressure p is only determined up to an additive constant. One can fix it by requiring

$$\int_\Omega p\, dx = 0, \tag{2.7}$$

or by prescribing the pressure at some point in the domain. Also the compatibility condition

$$\int_\Omega q\, dx = \int_\Omega \nabla \cdot \mathbf{u}\, dx = \int_{\partial\Omega} \mathbf{u} \cdot \nu\, ds = 0$$

must be satisfied for no-flow and periodic boundary conditions. The above condition simply says that the total fluid input to an incompressible system with no-flow boundary must be zero.

2.1.1 Well-posedness of the Model

For simplicity we consider equation (2.5) without the gravity term, i.e.,

$$-\nabla \cdot (\mathbf{K}(\mathbf{x})(\nabla p)) = q \qquad \mathbf{x} \in \Omega, \tag{2.8}$$

subject to Neuman boundary conditions

$$(\mathbf{K}(\mathbf{x})\nabla p) \cdot \nu \;\; \equiv \;\; \mathbf{u} \cdot \nu = 0 \qquad \mathbf{x} \in \partial\Omega. \tag{2.9}$$

Here ν is the outward pointing normal defined almost everywhere on Ω.

In order to establish the well-posedness of the above problem we need to define the following linear spaces of functions over Ω:

- $L^2(\Omega)$: space of square-integrable (equivalence classes of) functions over Ω;
- $H^m(\Omega)$: Sobolev space of $L^2(\Omega)$ functions with square-integrable weak derivatives up to order m;
- $C^k(\Omega)$: set of functions with continuous derivatives up to order k.

We also, are going to make use of the following definitions extracted from [4]:

Definition 1 *Let H be a Hilbert space. A bilinear form $a : H \times H \to \mathbb{R}$ is called* ***continuous*** *provided there exists $C > 0$ such that*

$$|a(u, v)| \leq C\|u\|\|v\| \quad \text{for all } v \in H.$$

A symmetric continuous bilinear form a is called ***H-ellipic****, or for short* ***elliptic*** *or* ***coercive****, provided for some $\alpha > 0$,*

$$a(v, v) \geq \alpha\|v\|^2 \quad \text{for all } v \in H.$$

We clearly can see that every H-elliptic bilinear form a induces a norm via

$$\|v\|_a = \sqrt{a(v, v)}. \tag{2.10}$$

This norm is called **energy norm** and it is equivalent to the norm of the Hilbert space H.

Definition 2 *A function $f : \mathbb{R}^n \supset D \to \mathbb{R}^m$ is called **Lipschitz continuous** provided that for some number c, $\|f(x) - f(y)\| \leq c\|x - y\|$, for all $x,\ y \in D$. A domain Ω is called a **Lipschitz domain** provided that for every $x \in \partial\Omega$, there is a neighborhood of $\partial\Omega$ which can be represent as a graph of a Lipschitz continuous function.*

Clearly the pressure equation (2.8) with Neumann boundary conditions (2.9) determine a function up to a additive constant. This suggests that in formulating the weak version of this problem we restrict ourselves to the subspace

$$V = \{v \in H^1(\Omega) : \int_\Omega v\, d\mathbf{x} = 0\}.$$

Then, let v be a smooth function on Ω, more precisely $v \in V$. Multiply both sides of (2.8) by v and integrate over Ω and use the Stokes theorem to get

$$-\int_\Omega \nabla \cdot (\mathbf{K}(\mathbf{x})\nabla p) v d\mathbf{x} = \int_\Omega \mathbf{K}(\mathbf{x})\nabla p \cdot \nabla v d\mathbf{x} - \int_{\partial\Omega} (\mathbf{K}(\mathbf{x})\nabla p) \cdot \nu v = \int_\Omega q v d\mathbf{x} \tag{2.11}$$

for $p \in H^2\Omega$. Then by (2.9) we get

$$\int_\Omega \mathbf{K}(\mathbf{x})\nabla p \cdot \nabla v d\mathbf{x} = \int_\Omega q v d\mathbf{x} \tag{2.12}$$

for all $v \in V$.

The bilinear form

$$a(p, v) = \int_\Omega \mathbf{K}(\mathbf{x})\nabla p \cdot \nabla v d\mathbf{x} \tag{2.13}$$

is not $H^1(\Omega)$-elliptic, but thanks to the following result it is V-elliptic:

***A variant of Friedrichs' inequality** [4]: Let Ω be a Lipschitz domain, and suppose* *that it satisfies the cone condition*[3]*. Then there is a constant $c = c(\Omega)$ such that*

$$\begin{aligned} \|v\|_{L^2} &\le c(|\overline{v}| + |v|_1) \quad \text{for all } v \in H^1(\Omega) \\ \text{with} \quad \overline{v} &= \frac{1}{\mu(\Omega)} \int_\Omega v(x)dx, \end{aligned} \tag{2.14}$$

where is the $\| \cdot \|_1$ represents the Sobolev semi-norm of order 1, and $\mu(\Omega) = \int_\Omega 1dx$.

Now, introducing $\mathbf{w} = \mathbf{K}(\mathbf{x})\nabla p$ equations (2.8) and (2.9) become

$$-\text{div}\mathbf{w} = q \text{ in } \Omega, \quad \nu \cdot \mathbf{w} = 0 \text{ on } \partial\Omega.$$

By Gauss integral theorem,

$$\int_\Omega \text{div}\mathbf{w}\, d\mathbf{x} = \int_{\partial\Omega} \mathbf{w} \cdot \nu,$$

and thus

$$\int_\Omega q d\mathbf{x} = 0. \tag{2.15}$$

By Lax-Milgram Theorem (see Theorem 2.5, pg. 38 in [4]), there exists a unique solution $p \in V$ to

$$a(p, v) = (q, v)_{L^2(\Omega)} \quad \text{for all } v \in V, \tag{2.16}$$

where $(\cdot, \cdot)_{L^2(\Omega)}$ represents the standard scalar product in $L^2(\Omega)$. Because of (2.15), (2.16) also holds for $v = \text{const}$, and thus for all $v \in H^1(\Omega)$. The next theorem allows us to deduce that every classical solution of the variational problem satisfies (2.8)-(2.9).

Theorem 1 *Let Ω be bounded, and suppose Ω has piecewise smooth boundary. In addition Ω satisfies the cone condition. Then the variational problem*

$$\begin{aligned} &\text{Find } p \in H^1(\Omega) \text{ such that} \\ &a(p, v) = (q, v)_{L^2(\Omega)}\ \forall v \in V \end{aligned} \tag{2.17}$$

[3]A domain Ω satisfies a cone condition if there is a fixed cone Ksuch that at any point $y \in \partial\Omega$ one can place the vertex at y with $K - y$ lying within Ω.

has exactly one solution $p \in H^1(\Omega)$. The solution of the variational problem lies in $C^2(\Omega) \cap C^1(\Omega)$ if and only if there exists a classical solution of the boundary-value problem

$$\begin{aligned} Lp = q \quad \text{in } \Omega, \\ \sum_{i,k} \nu_i a_{ik} \partial_k p = 0 \quad \text{on } \partial\Omega, \end{aligned} \tag{2.18}$$

in which case the two solutions are identical.

Here L is a second order elliptic partial differential operator with divergence structure

$$Lp = -\sum_{i,k=1}^{n} \partial_i (a_{i,k} \partial_k p) + a_0 p,$$

where $a_0(x) \geq 0$ for $x \in \Omega$.

More general boundary value problems are treated in [20, 21].

2.2. Non-Darcy Flow

Darcy's law can be extended to a model of momentum conservation which is more accurate than Darcy's model (2.1) in order to better describe the flow for larger Reynolds number (Re), for example, when the velocities are large. Typically, Darcy's model is valid when $Re \leq 1$ and non-Darcy's model is valid for $1 \leq Re \leq 10^2$. We consider the system of equations that describes a flow of a single-phase fluid in a porous medium subject to non-Darcy flow, also called generalized Forchheimer's law, for which the gradient of pressure and its velocity are nonlinearly related. In particular, assume that the flow of an incompressible fluid is described, as in [7, 16], by the system of equations

$$G(\mathbf{u}) + \nabla p = 0, \qquad \mathbf{x} \in \Omega,\ t \geq 0, \tag{2.19}$$

$$\operatorname{div}(\mathbf{u}) = q, \qquad \mathbf{x} \in \Omega,\ t \geq 0, \tag{2.20}$$

where Ω is a bounded domain in two- or three-space, p is the pressure, $\mathbf{u}$ is the volumetric flow rate (or velocity) of the fluid, ϕ is the porosity of the medium, and q is an external mass flow rate. The function G can be assumed to be smooth function of its arguments and to generate a monotone operator with respect to its velocity argument [7]. The classical form of Forchheimer's law is given by

$$G(\mathbf{u}) = \mathbf{K}^{-1}\mathbf{u} + \beta|\mathbf{u}|\mathbf{u} \tag{2.21}$$

where β is a parameter with units of [length^{-1}], which is called Forchheimer's coefficient [9] and is a porous medium property that needs to be measured experimentally. Combining equations (2.19), (2.20) we have

$$-\nabla\left(A(\mathbf{K}, \beta; \mathbf{u})\nabla p\right) = q, \tag{2.22}$$

where we define

$$A(\mathbf{K}, \beta; \mathbf{u}) = \left(\mu\mathbf{K}^{-1} + \beta|\mathbf{u}|\right)^{-1}. \tag{2.23}$$

Here the velocity $\mathbf{u}$ is given by

$$\mathbf{u} = \left(\mathbf{K}^{-1} + \beta|\mathbf{u}|\right)^{-1}(-\nabla p). \tag{2.24}$$

Thus rewriting the velocity in terms of $A(\mathbf{K}, \beta; \mathbf{u})$, we have

$$\mathbf{u} = -A(\mathbf{K}, \beta; \mathbf{u})\nabla p. \tag{2.25}$$

2.2.1 Well-posedness of the model

We have assumed $G(\rho, \mathbf{u})$ is a smooth function and generate a monotone operator with respect to its velocity. The equation (2.22) is a *quasilinear* elliptic equation. Suppose that (2.22) is also V-coercive, and that Ω is bounded domain in $\mathbb{R}^n$. Then the existence and uniqueness of (2.22) subject to Neuman or Dirichlet boundary conditions is guaranteed by the Browder-Vishik Theorem [21].

2.3. Mixed Variational Form of Darcy's Flow

Let $H(div;\Omega)$, with Ω being a bounded, open subset of $\mathbb{R}^2$, be the set of vector functions $\mathbf{v} \in (L^2(\Omega))^2$ such that $\nabla \cdot \mathbf{v} \in L^2(\Omega)$, where $\nabla\cdot$ is taken in the sense of weak derivatives. Let

$$V = \{\mathbf{v} \in H(div;\Omega) : \mathbf{v} \cdot \nu = 0 \text{ on } \partial\Omega\}. \tag{2.26}$$

Let $W = L^2(\Omega)$. To obtain a variational form of (2.1),(2.4) and (2.6), we multiply (2.1) by $\mathbf{K}^{-1}$, and by $\mathbf{v} \in V$, integrate over Ω, integrate by parts, and apply the divergence theorem to see that

$$(\mathbf{K}^{-1}\mathbf{u}, \mathbf{v}) - (p, \nabla \cdot \mathbf{v}) = 0, \qquad \mathbf{v} \in V. \tag{2.27}$$

Next multiply (2.4) by $w \in W$ and integrate to obtain

$$(\nabla \cdot \mathbf{u}, w) = (q, w), \qquad w \in W. \tag{2.28}$$

The system (2.27)-(2.28) is the mixed variational form of (2.5), (2.6). If $\mathbf{u}$ and p satisfy (2.1) and (2.4) , they also satisfy (2.27)-(2.28). The converse also holds if p is sufficiently smooth (eg., if $p \in H^2(\Omega)$) [5]). We will use this form (2.27), (2.28) in Chapter 3.

3. DISCRETIZATION FOR DARCY FLOW

In this chapter we apply the cell-centered finite difference method to discretize Darcy flow problems. For simplicity we consider a rectangular domain Ω. We also present the relation between this method and the mixed finite element formulation on rectangular grids using Raviart-Thomas elements using certain quadrature rules [19]. We approximate the velocity $\mathbf{u}$ by U, the pressure p by P. Here we assume $\mathbf{K}$ is a diagonal tensor and we write

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

Consider the system (2.1),(2.4) and (2.6) which we rewrite here for convenience

$$\begin{aligned} -\nabla \cdot (\mathbf{K}(\mathbf{x})\nabla p) \quad &\equiv \quad \nabla \cdot \mathbf{u} = q \qquad \mathbf{x} \in \Omega \\ (\mathbf{K}(\mathbf{x})\nabla p) \cdot \nu \quad &\equiv \quad \mathbf{u} \cdot \nu = 0 \qquad \mathbf{x} \in \partial\Omega \end{aligned} \tag{3.1}$$

where Ω is a rectangular bounded domain in $\mathbb{R}^2$, i.e., $\Omega = (a,b) \times (c,d)$ with a, b, c, d $\in \mathbb{R}$; $a < b$, $c < d$. We assume K is smooth enough to theory of Chapter 2. applies and the problem (3.1) is well-posed

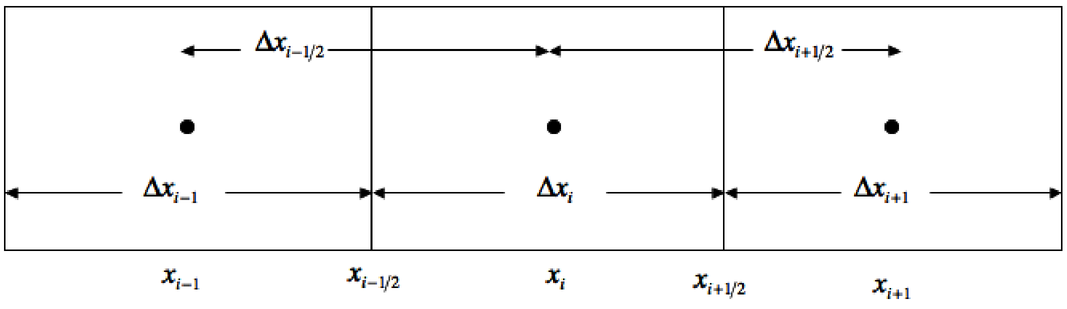

FIGURE 3.1: Cell-centered finite difference coordinates

Partition $[a,b]$ into m subintervals of length $\triangle x$ and $[c,d]$ into l subintervals of length $\triangle y$ and set (see Figure 3.1)

$$\begin{aligned}
x_{k+\frac{1}{2}} &= a + k\triangle x, \\
y_{j+\frac{1}{2}} &= c + j\triangle y, \\
x_k &= \frac{\left(x_{k-\frac{1}{2}} + x_{k+\frac{1}{2}}\right)}{2}, \\
y_j &= \frac{\left(y_{j-\frac{1}{2}} + y_{j+\frac{1}{2}}\right)}{2}, \\
\triangle x_k &= x_{k+\frac{1}{2}} - x_{k-\frac{1}{2}}, \\
\triangle y_j &= y_{j+\frac{1}{2}} + y_{j-\frac{1}{2}}, \\
\triangle x_{k+\frac{1}{2}} &= x_{k+1} - x_k, \\
\triangle y_{j+\frac{1}{2}} &= y_{j+1} - y_j.
\end{aligned} \tag{3.2}$$

3.1. Cell-Centered Finite Difference Method

We discretize (3.1) by replacing derivatives with difference quotients and approximations $P_{k,j} \approx p(x_k, y_j)$, $K_{k,j} \approx \mathbf{K}(x_k, y_j)$, and $Q_{k,j} \approx q(x_k, y_j)$. So, we approximate (3.1) at (x_k, y_j) by

$$\begin{aligned}
&-\frac{1}{x_{k+\frac{1}{2}} - x_{k-\frac{1}{2}}}\left[(K_{xx})_{k+\frac{1}{2},j}\frac{P_{k+1,j} - P_{k,j}}{x_{k+1} - x_k} - (K_{xx})_{k-\frac{1}{2},j}\frac{P_{k,j} - P_{k-1,j}}{x_k - x_{k-1}}\right] \\
&-\frac{1}{y_{j+\frac{1}{2}} - y_{j-\frac{1}{2}}}\left[(K_{yy})_{k,j+\frac{1}{2}}\frac{P_{k,j+1} - P_{k,j}}{y_{j+1} - y_j} - (K_{yy})_{k,j-\frac{1}{2}}\frac{P_{k,j} - P_{k,j-1}}{y_j - y_{j-1}}\right] = Q_{k,j}, \\
&\qquad 0 \le k \le m,\ 0 \le j \le l.
\end{aligned} \tag{3.3}$$

Multiplying (3.3) by $\triangle x_k \triangle y_j$ (the area of a cell) and assuming an uniform grid we get

$$\begin{aligned}
&\triangle y_j\left[-(T_x)_{k+\frac{1}{2},j}\left(P_{k+1,j} - P_{k,j}\right) + (T_x)_{k-\frac{1}{2},j}\left(P_{k,j} - P_{k-1,j}\right)\right] + \\
&\triangle x_k\left[-(T_y)_{k,j+\frac{1}{2}}\left(P_{k,j+1} - P_{k,j}\right) + (T_y)_{k,j-\frac{1}{2}}\left(P_{k,j} - P_{k,j-1}\right)\right] \\
&\qquad = \triangle x_k \triangle y_j Q_{k,j}, \\
&\qquad 0 \le k \le m,\ 0 \le j \le l.
\end{aligned} \tag{3.4}$$

Here the transmissibilities T_x and T_y defined in [17] are given by

$$(T_x)_{k+\frac{1}{2},j} = \frac{(K_{xx})_{k+\frac{1}{2},j}}{\triangle x_{k+\frac{1}{2}}}; \qquad (T_y)_{k,j+\frac{1}{2}} = \frac{(K_{yy})_{k,j+\frac{1}{2}}}{\triangle y_{j+\frac{1}{2}}}. \tag{3.5}$$

See Figure 3.2 for illustration.

Since the values for K are known at the center of each cell, the value of K on the boundary between cells is given by the harmonic average of the value for K between the two adjacent cells (cf. Figure 3.2). Thus the transmissibilities become

$$(T_x)_{k+\frac{1}{2},j} = \frac{2\left((K_{xx})_{k,j}\,(K_{xx})_{k+1,j}\right)}{\left((K_{xx})_{k,j} + (K_{xx})_{k+1,j}\right)\triangle x_{k+\frac{1}{2}}},\ (T_y)_{k,j+\frac{1}{2}} = \frac{2\left((K_{yy})_{k,j}\ (K_{yy})_{k,j+1}\right)}{\left((K_{yy})_{k,j} + (K_{yy})_{k,j+1}\right)\triangle y_{j+\frac{1}{2}}}. \tag{3.6}$$

No-flow boundary conditions are easily incorporated by setting

$$(K_{**})_{\frac{1}{2},j} = (K_{**})_{m+\frac{1}{2},j} = (K_{**})_{k,\frac{1}{2}} = (K_{**})_{k,l+\frac{1}{2}} = 0,$$

where $** = xx$ or yy.

This method is first order convergent which follows by equivalence to a certain mixed finite element method sown in the next section.

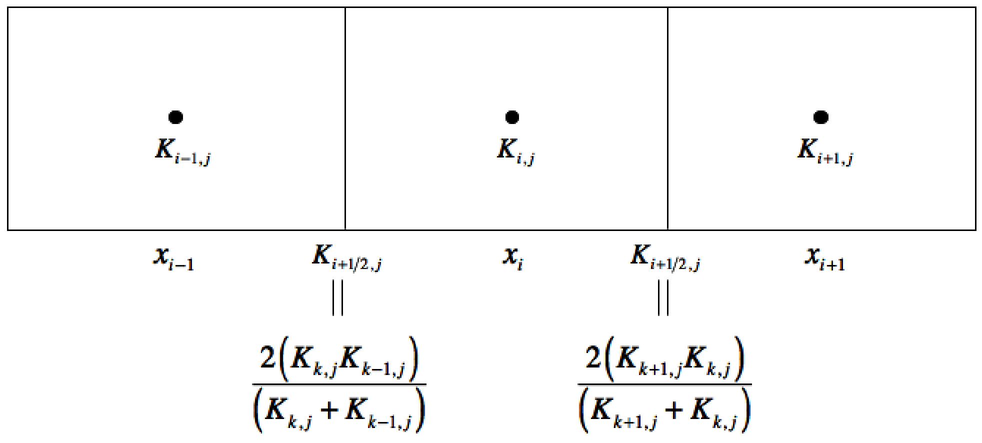
$K_{i-1,j}$
$K_{i,j}$
$K_{i+1,j}$
x_{i-1}
$K_{i+1/2,j}$
x_i
$K_{i+1/2,j}$
x_{i+1}
$\frac{2(K_{k,j}K_{k-1,j})}{(K_{k,j}+K_{k-1,j})}$
$\frac{2(K_{k+1,j}K_{k,j})}{(K_{k+1,j}+K_{k,j})}$

FIGURE 3.2: Permeabilities on the boundary between cells.

3.2. Mixed Finite Element Method

The idea of mixed methods for the pressure equation is to approximate the pressure and velocity simultaneously in a variational method. In this section, following the arguments in [19], we use the mixed finite element method to approximate system (2.27)-(2.28) for finite-dimensional subspaces $V_h \subset V$ and $W_h \subset W$, where V and W are the spaces described in section 2.3. To do that we define the piecewise-polynomial space $\mathcal{M}$ on a rectangular mesh $\triangle$ such that the horizontal and vertical edges of rectangles are parallel to the x- and y-coordinate axes, respectively, and adjacent elements completely share their common edge by

$$\mathcal{M}_q^r(\triangle) = \{\mathbf{v} \in C^q([a,b]) : v \text{ is a polynomial of degree } \leq \text{r on each subinterval of } \triangle\}.$$

On a rectangular mesh over the rectangle $a \leq x \leq b,\ c \leq y \leq d$, described by $\triangle_x$ and $\triangle_y$ as before, let the spaces of index r, $r = 0, 1, 2, \dots$ be

$$\begin{aligned}
W_h^r &= \mathcal{M}_{-1}^r(\triangle_x) \otimes \mathcal{M}_{-1}^r(\triangle_y),\\
\widetilde{V}_h^r &= [\mathcal{M}_0^{r+1}(\triangle_x) \otimes \mathcal{M}_{-1}^r(\triangle_y)] \times [\mathcal{M}_{-1}^r(\triangle_x) \otimes \mathcal{M}_0^{r+1}(\triangle_y)],\\
V_h^r &= \{\mathbf{v} = (v_x, v_y) \in \widetilde{V}_h^r : v_x(a,y) = v_x(b,y) = 0,\ v_y(y,a) = v_y(x,b) = 0\},\\
&= \{\mathbf{v} \in \widetilde{V}_h^r : \mathbf{v} \cdot \nu = 0 \text{ on } \partial\Omega\},
\end{aligned} \tag{3.7}$$

where h measures the largest linear dimension in the mesh, and $\otimes$ represents tensor-product.

The approximation of system (2.27)-(2.28) is $\{U, P\} \in V_h \times W_h$ satisfying

$$\begin{aligned}
(K^{-1}U, \mathbf{v}) - (P, \nabla \cdot \mathbf{v}) = 0, \qquad \mathbf{v} \in V_h,\\
(\nabla \cdot U, w) = (q, w), \qquad w \in W_h.
\end{aligned} \tag{3.8}$$

We wish to show that the block-centered finite difference method (3.3) is equivalent to the lowest-order mixed method (3.8 with $V_h = V_h^0, W_h = W_h^0$) with special numerical quadrature rules. In the case $r = 0$, with partitions $\triangle_x$ and $\triangle_y$, let the bases for the subspaces be

$$\begin{aligned} &\mathcal{M}_0^1(\triangle_x) : \{v_k^x : 1 \le k \le m-1\}, && v_k^x(x_n) = \delta_{kn}, \\ &\mathcal{M}_{-1}^0(\triangle_x) : \{w_k^x : 1 \le k \le m\} && w_k^x = 1 \quad \text{if } x_{k-1} < x < x_k, \\ &\mathcal{M}_1^0(\triangle_y) : \{v_j^y : 1 \le j \le l-1\}, && v_j^y(x_n) = \delta_{jn}, \\ &\mathcal{M}_{-1}^0(\triangle_y) : \{w_j^y : 1 \le j \le l\} && w_j^y = 1 \quad \text{if } y_{j-1} < y < y_j. \end{aligned} \tag{3.9}$$

Then bases for $(V_h)_x$, $(V_h)_y$ and W_h are, respectively, $\{v_k^x w_j^y\}$, $\{w_k^x v_j^y\}$, and $\{w_k^x w_j^y\}$. The dimension of V_h is four. The degrees of freedom for V_h are the values of normal components of functions at the midpoint on each edge in $\triangle_h$ (cf. Fig. 3.3)

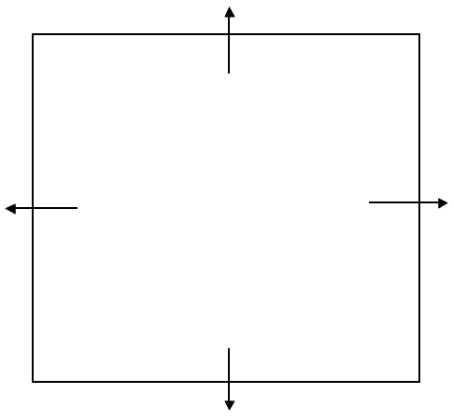
FIGURE 3.3: The rectangular element in $\mathrm{RT}_{[0]}$

We start by writing (3.4) in terms of the Darcy-velocity

$$\triangle y_j \left((U_x)_{k+\frac{1}{2},j} - (U_x)_{k-\frac{1}{2},j} \right) + \triangle x_k \left((U_y)_{k,j+\frac{1}{2}} - (U_y)_{k,j-\frac{1}{2}} \right) = \triangle x_k \triangle y_j q_{k,j} \tag{3.10}$$

where U_x and U_y are velocity components defined by

$$(U_x)_{k+\frac{1}{2},j} = -(T_x)_{k+\frac{1}{2},j} \left(P_{k+1,j} - P_{k,j} \right) \tag{3.11}$$

$$(U_y)_{k,j+\frac{1}{2}} = -(T_y)_{k,j+\frac{1}{2}} \left(P_{k,j+1} - P_{k,j} \right). \tag{3.12}$$

Let U the unique function in V_h^0 satisfying (3.11) and (3.12), and let P be the piecewise-constant function with cell values $P_{k,j}$. Note that

$$(U_x)_{k+\frac{1}{2},j} - (U_x)_{k-\frac{1}{2},j} = \triangle x_k \frac{\partial}{\partial x}(U_x), \qquad (U_y)_{k,j+\frac{1}{2}} - (U_y)_{k,j-\frac{1}{2}} = \triangle y_j \frac{\partial}{\partial y}(U_y) \tag{3.13}$$

and so, the right hand side of (3.10) is equal to

$$\triangle y_j \triangle x_k \frac{\partial}{\partial x}(U_x) + \triangle x_k \triangle y_j \frac{\partial}{\partial y}(U_y) = \int_{x_{k-1/2}}^{x_{k+1/2}} \int_{y_{j-1/2}}^{y_{j+1/2}} \nabla \cdot U. \tag{3.14}$$

The right hand side of (3.10) is the midpoint-rule (reference) for integral of q over the cell. Thus U satisfies

$$(\nabla \cdot U, w) = (q, w)_{M_x M_y}, \qquad w \in W_h^0. \tag{3.15}$$

where $M_x M_y$ denotes midpoint-rule quadrature in both directions. Note that $\triangle x_{k+1/2} = \frac{1}{2}(\triangle x_k + \triangle x_{k+1})$. Now we multiply both sides of (3.11) by $\triangle y_j$ and rearrange the terms to get

$$\frac{1}{2}(\triangle x_k + \triangle x_{k+1}) \triangle y_j \frac{1}{K_{k+1/2,j}} (U_x)_{k+1/2,j} - \triangle y_j (P_{k,j} - P_{k+1,j}) = 0. \tag{3.16}$$

Now we use the trapezoidal-rule quadrature in the x-direction (T_x) and midpoint-rule quadrature (M_y) in the y-direction to write the above expression as

$$\left(K^{-1} U_x, v_{k+1/2}^x w_j^y\right)_{T_x M_y} - \left(P, \frac{\partial}{\partial x}(v_{k+1/2}^x w_j^y)\right) = 0 \tag{3.17}$$

where $v_{k+1/2}^x$ and w_j^y correspond to v_k^x (linear basis function) and w_j^k(constant basis function) on (3.9). Similarly, for (3.12), we have

$$\left(K^{-1} U_y, w_k^x v_{j+1/2}^y\right)_{M_x T_y} - \left(P, \frac{\partial}{\partial y}(w_k^x v_{j+1/2}^y)\right) = 0 \tag{3.18}$$

Combining (3.17) and (3.18), we obtain

$$(K^{-1} U_x, v_x)_{T_x M_y} + (K^{-1} U_y, v_y)_{M_x T_y} - (P, \nabla \cdot \mathbf{v}) = 0 \quad \mathbf{v} \in V_h^0. \tag{3.19}$$

Equations (3.15) and (3.19) satisfy the mixed-method formulation (3.8), since we assumed no-flow boundary conditions and we have set $(K_{**})_{\frac{1}{2},j} = (K_{**})_{m+\frac{1}{2},j} = (K_{**})_{k,\frac{1}{2}} =$

$(K_{**})_{k,l+\frac{1}{2}} = 0$, where $** = xx$ or yy. In the block-centered finite difference, (3.11) and (3.12) yield $U \cdot \nu = 0$, which is the condition imposed by the space V_h^0 in the mixed method. Thus the block-center finite difference method is equivalent to the mixed method provided that the quadrature rules are used as indicated and no-flow boundary conditions are assumed.

3.3. Boundary Conditions

Different types of boundary conditions are treated in distinct ways in the discrete problem. Here we introduce difference equations to approximate three different types of boundary conditions of relevant importance on the applications we are interested in.

3.3.1 Neumann Boundary Conditions

No-flow boundary conditions are incorporated in the discrete model (cell-centered finite difference) as previously mentioned i.e.

$$(K_{**})_{\frac{1}{2},j} = (K_{**})_{m+\frac{1}{2},j} = (K_{**})_{k,\frac{1}{2}} = (K_{**})_{k,l+\frac{1}{2}} = 0,$$

where $** = xx$ or yy. These conditions applied to (3.3) will result in a singular system. The uniqueness of the system is achieved with the use of (2.7), which results in a extra equation.

Thus the linear system we have to solve here has $(ml+1)$ equations (ml) unknowns.

3.3.2 Dirichlet Boundary Condition

Here we denote the value of the pressure on the boundary faces by the superscript *(see Figure 3.4).

Following the mixed formulation we rewrite equation (3.16) for the cells on the left boundary

$$\frac{1}{2}\triangle x_k \triangle y_j \frac{1}{(K_{xx})_{k+1/2,j}}(U_x)_{k+1/2,j} = \triangle y_j (P_{k,j} - P^*_{k+1,j}). \tag{3.20}$$

The above form, as explained in [18], suggests using a transmissibility on the left boundary as

$$(T_x)_{\frac{1}{2},j} = \triangle y_j \frac{2(K_{xx})_{\frac{1}{2},j}}{\triangle x_{\frac{1}{2}}}.$$

Similar idea is applied on the right boundary.

Assuming rectangular domain Ω with rectangular grid we have set for all blocks on the left boundary

$$(T_x)_{\frac{1}{2},j}(P_{1,j} - P^*_{0,j}) = U_{\frac{1}{2},j}, \quad 1 \le j \le l. \tag{3.21}$$

The linear system to be solved for $P_{k,j}$ in this case has ml equations and ml unknowns.

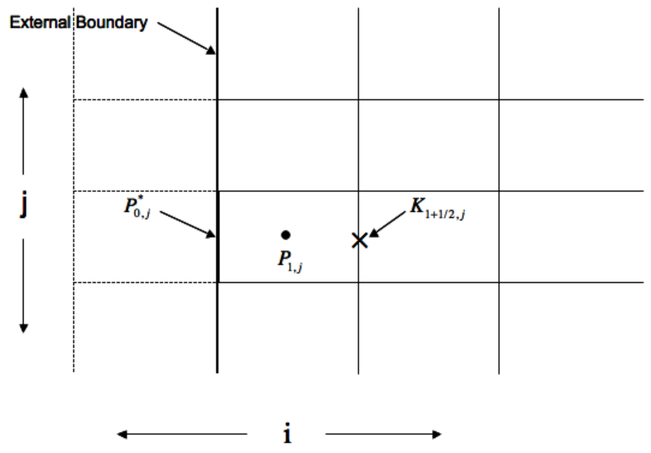

FIGURE 3.4: Computational domain with interior and boundary cells

3.3.3 Periodic Boundary Conditions

Here, again, we are going to denote the value of pressure on the boundary faces by the superscript *. In this case the values of pressure on the boundary are considered unknowns for the resulting linear system. This is going to increase the dimension of the linear system to be solved.

To impose periodic boundary conditions we need equations representing periodicity conditions of the system, i.e. correspondences between flux and pressure on the opposite boundaries. The correspondence between pressure is given by the equation

$$P^*_{m+1,j} = P^*_{0,j} + G, \qquad 1 \leq j \leq l, \tag{3.22}$$

where G is the pressure gradient (in this case in the x-direction) (see Figure 3.5). The above equation relates the pressures on the two boundaries to the imposed pressure gradient G, as explained in [8].

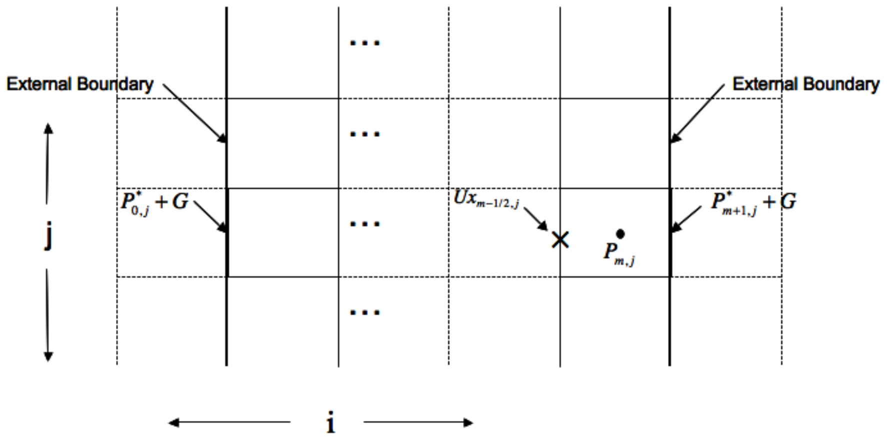

FIGURE 3.5: Computational domain with interior and boundary cells representing periodic boundary conditions

Also, we have to consider the relations between flux on the left and right boundaries of our domain. As given in [8] these relationships have the form

$$(T_x)_{\frac{1}{2},j}(P_{1,j} - P^*_{0,j}) - (T_x)_{m+1,j}(P^*_{m+\frac{1}{2},j} - P_{m,j}) = 0, \quad 1 \leq j \leq l. \tag{3.23}$$

The above relationships specify that the flux into Ω at the left boundary is equal and opposite to the flux through the right boundary. Here the transmissibilities on the boundaries are given as in the Dirichlet case.

As before, to ensure uniqueness, we impose equation (2.7), i.e., the pressures must add up to zero, which give us one more equation to represent this

$$\sum_{k=1}^{m}\sum_{j=1}^{l} P_{k,j} = 0. \tag{3.24}$$

The linear system (3.4)-(3.22) to be solved here has $ml + 2l$ unknowns and $ml + l$ equations.

3.4. Implementation of Wells

Numerical simulation of fluid flow in a petroleum reservoir has to account for the presence of wells. In order to simulate the existence of wells in our domain we have to incorporate them in the right hand side of the pressure equation (3.1), i.e., we treat them as source terms. Since the compatibility condition must be satisfied, we have to assume that the net injection and production is zero. In order to introduce these effects in our discretized model we we have to assign to q the values α_i for the cells where the injection wells are located and β_j for the cells where the production wells are located such that

$$\sum_{i=1}^{iw} \alpha_i + \sum_{j=1}^{pw} \beta_j = 0, \tag{3.25}$$

where $\alpha_i \in \mathbb{R}_+$, $\beta_j \in \mathbb{R}_-$, iw is the number of injection wells, and pw is the number of production wells. We assume that the wells are located at the center of the corresponding grid cell.

In the discrete form, for a injection well located in the cell m, n we have that

$$Q_{m,n} = \alpha_i.$$

3.5. Obtaining the Numerical Solution

After choosing an ordering of the $P_{k,j}$ for $k = 0\ldots,m,\ j = 0\ldots,l$ the difference equation (3.3) generates a linear system of the form:

$$M_{k,j}P_{k,j} = Q_{k,j}, \tag{3.26}$$

with $M \in \mathbb{R}^{M_1\times M_1}$ and $P,\ Q \in \mathbb{R}^{M_1}$, where $M_1 = ml$. Here the matrix M incorporates the coefficients (T_x) and (T_y).

We are interested in finding the values for pressure and velocity. Since the permeabilities $K_{k,j}$ are given for each grid cell we can summarize the process as folow:

- Compute the transmissibilities $(T_*)_{k,j}$, where $* = x$ or y;
- Construct the matrix M;
- Solve the linear system to get $P_{k,j}$;
- Post-process to get $(U_*)_{k,j}$, i.e., use equations (3.11) and (3.12) to compute $(U_*)_{k,j}$.

4. DISCRETIZATION AND SOLVER FOR NON-DARCY FLOW

In this chapter we use cell-centered finite difference method to discretize equation (2.22) and apply appropriated boundary conditions. We follow the same ideas used in Chapter 3 to discretize equation (3.1). The equivalence of difference equations to a certain mixed method for quasilinear elliptic problems will be considered elsewhere.

The discretization of 2.22 yields a nonlinear discrete equation. In order to solve the nonlinear equation we use the fixed point iteration. We present a brief description of the fixed point method and sufficient conditions for the convergence of the method are derived for our problem.

4.1. Cell-Centered Finite Difference Method for Non-Darcy Flow

Here, due to nonlinearity of the coefficient $A(\mathbf{K}, \beta; \mathbf{u})$ in (2.22), the transmissibilities, unlike in Chapter 3 and equation (3.5), depend on U, i.e., $T_x = T_x(U_x)$ and $T_y = T_y(U_y)$. We denote this by T_x^u and T_y^u, respectively.

With the definitions (3.2), the finite difference approximation of equation (2.22) is given by

$$\begin{aligned} &\triangle y_j \left[-(T_x^u)_{k+\frac{1}{2},j}(P_{k+1,j} - P_{k,j}) - (T_x^u)_{k-\frac{1}{2},j}(P_{k,j} - P_{k-1,j})\right] \\ &\triangle x_j \left[-(T_y^u)_{k,j+\frac{1}{2}}(P_{k,j+1} - P_{k,j}) - (T_y^u)_{k,j-\frac{1}{2}}(P_{k,j} - P_{k,j-1})\right] = \triangle x_k \triangle y_j Q_{k,j}, \qquad (4.1) \\ &\qquad\qquad 0 \le k \le m,\ 0 \le j \le l, \end{aligned}$$

where the transmissibility (T_x^u) is given by

$$(T_x^u)_{k+\frac{1}{2},j} = \frac{\left(\left(\frac{2\left((K_{xx})_{k,j}\cdot(K_{xx})_{k+1,j}\right)}{\left((K_{xx})_{k,j}+(K_{xx})_{k+1,j}\right)\triangle x_{k+\frac{1}{2}}}\right)^{-1} + \beta(U_x)_{k+\frac{1}{2},j}\right)^{-1}}{\triangle x_{k+\frac{1}{2}}}. \qquad (4.2)$$

Similarly, in y-direction, (T_y^u) is given by

$$(T_y^u)_{k,j+\frac{1}{2}} = \frac{\left(\left(\frac{2\left((K_{yy})_{k,j}\cdot(K_{yy})_{k,j+1}\right)}{\left((K_{yy})_{k,j}+(K_{yy})_{k,j+1}\right)\triangle y_{k,j+\frac{1}{2}}}\right)^{-1} + \beta(U_y)_{k,j+\frac{1}{2}}\right)^{-1}}{\triangle y_{j+\frac{1}{2}}}. \tag{4.3}$$

We approximate the equation (2.24) at (x_k, y_j) in the x- and y-direction, respectively, by

$$(U_x)_{k+\frac{1}{2},j} = -(T_x^u)_{k+\frac{1}{2},j}(P_{k+1,j} - P_k, j),$$

$$(U_y)_{k,j+\frac{1}{2}} = -(T_y^u)_{k,j+\frac{1}{2}}(P_{k,j+1} - P_k, j).$$

This definitions (4.1), (4.2) and (4.3) are similar to the Darcy case (3.4), (3.11), (3.12). However we pointed out that the system (4.1), unlike in Darcy's case, is nonlinear. The solution method is described in the next section.

4.2. Fixed Point Formulation

Equation (2.22) and its discrete approximation (4.1) with definitions (4.2), (4.2) are nonlinear equations. We are interested in solving equation (4.1) using the fixed point formulation for which the basic theory is presented next. Here we are going to use definitions and theorems from [15, 2].

In general, the problem may be formulated as follows:

Let $S \subset \mathbb{R}^n$ be open and $f : S \to \mathbb{R}^n$ be a mapping.

$$\text{Find} \quad x \in S \;\text{ with }\; f(x) = x. \tag{4.4}$$

Then x is called a *fixed point.*

In most cases, a fixed point cannot be calculated (with exact arithmetic) in a finite number of operations, but only by an iterative method, i.e., by a mapping

$$\Phi : S \to S,$$

so that for the sequence

$$x^{(k+1)} := \Phi(x^{(k)}) \tag{4.5}$$

with a given initial guess $x^{(0)}$ we get

$$x^{(k)} \to x \quad \text{for} \quad k \to \infty. \tag{4.6}$$

Here x is the solution of (4.4). The iterative method (4.5) is also called *nonlinear Richardson iteration*, *Picard iteration*, or *the method of successive substitutions.*

In the case of a continuous map Φ it follows from (4.5) and (4.6) that the limit x satisfies

$$x = \Phi(x). \tag{4.7}$$

This means that equation (4.7) should imply that x is a solution of equation (4.4).

For the fixed point formulation (4.4) we choose $\Phi := f$, in other words, the *fixed point iteration* reads

$$x^{(k+1)} := f(x^{(k)}). \tag{4.8}$$

To ensure that

$$\left\|\Phi\left(x^{(k+1)}\right) - \Phi\left(x^{(k)}\right)\right\| = \|x^{(k+2)} - x^{(k+1)}\| < \|x^{(k+1)} - x^{(k)}\|$$

it is sufficient that the iteration function (here $\Phi = f$) be a contraction. Here $\|\cdot\|$ can be any norm in $\mathbb{R}^n$. The following definition explain the notion of contractivity

Definition 3 *Let $S \subset \mathbb{R}^n$. A function $\Phi : S \to \mathbb{R}^n$ satisfies the **Lipschitz condition** on S (with respect to the norm $\|\cdot\|$) if there exists a constant $L > 0$ such that, for any two points $x,\ y \in S$,*

$$\|\Phi(x) - \Phi(y)\| \leq L\|x - y\|.$$

*The greatest lower bound for for such constants is the **Lipschitz constant** for Φ on S. If Φ has Lipschitz constant $L < 1$ on S, then Φ is a **contraction** on S.*

Sufficient conditions for a contraction are given by the following lemma:

Lemma 1 *Let $S \in \mathbb{R}^n$ be open and convex, and $g : S \to \mathbb{R}^n$ continuously differentiable. If*

$$\sup_{x \in S} \|Dg(x)\| = L < 1$$

holds, where $\|\cdot\|$ in $\mathbb{R}^{n,n}$ is compatible with $\|\cdot\|$ in $\mathbb{R}^n$, then g is contracting in S.

Therefore, if $S \subset \mathbb{R}^n$ is open, $f : S \subset \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable, and if there exists some $\widetilde{x} \in S$ with $\|Df(\widetilde{x})\| < 1$, then there exists a closed convex neighborhood $\widetilde{S}$ of $\widetilde{x}$ with

$$\|Df(x)\| \leq L < 1 \quad \text{for } x \in \widetilde{S}$$

and, for example, $L = \|Df(\widetilde{x})\| + \frac{1}{2}(1 - \|Df(\widetilde{x})\|$, guaranteeing the contractivity of f in S.

The unique existence of a fixed point and the convergence of (4.8) is guaranteed if the set S where f is a contraction is mapped into itself, what is explicit in the next theorem, for which the proof can be found in [15]:

Theorem 2 (Banach's fixed-point theorem) *Let $S \subset \mathbb{R}^n$, $S \neq \emptyset$, and S be closed. Let $f : S \to \mathbb{R}^n$ be a contraction with Lipschitz constant $L < 1$ and $f[S] \subset S$. Then we have:*

(1) *There exists one and only one fixed point $x \in S$ of f.*

(2) *For arbitrary $x^{(0)} \in S$ the fixed point iteration 4.8 converges to x, and we have*

$$\begin{aligned} \left\| x^{(k)} - x \right\| &\leq \frac{L}{1-L} \left\| x^{(k)} - x^{(k-1)} \right\| \\ &\leq \frac{L^k}{1-L} \left\| x^{(1)} - x^{(0)} \right\|. \end{aligned}$$

Remark 1 *The theorem can be generalized from $\mathbb{R}^n$ to any Banach space X, with $S \in X$.*

This enable us to define iterative schemes directly in the function space for nonlinear boundary value problems, which means that the resulting (linear) problems in the iteration step are to be discretized. We can often construct a closed S such that f is a contraction on S and satisfies the hypothesis of lemma 1. To verify that $f[S] \subset S$, the following lemma is helpful:

Lemma 2 *Let $S \subset \mathbb{R}^n$, $f : S \to \mathbb{R}^n$. If there exists a $y \in S$ and a $r > 0$ with*

$$\overline{B}_r(y) \subset S,$$

with f contraction on $\overline{B}_r(y)$ with Lipschitz constant $L < 1$, so that

$$\|y - f(y)\| \leq r(1 - L),$$

then f has one and only one fixed point in $\overline{B}_r(y)$, and 4.8 converges.

In the setting of Theorem 2 the fixed-point iteration is thus globally convergent in S. In the setting of Lemma 2 it is locally convergent in S (globally in $\overline{B}_r(y)$). We see that in the situation of Theorem 2 the sequence $(x^{(k)})$, because of

$$\left\| x^{(k+1)} - x \right\| = \left\| f(x^{(k)}) - f(x) \right\| \leq L \left\| x^{(k)} - x \right\|,$$

converges linearly to x [13] (and in general not faster than linear).

4.3. Fixed-Point Approximation to Non-Darcy Flow

Now we want to apply the fixed point iteration to solve the system (4.1). A good initial guess is provided by the solution to the Darcy flow that is, using $\mathbf{u}^{(0)} = -A(\mathbf{K}, \mathbf{0}; \mathbf{0})\nabla p$. With this initial guess we iterate the whole system (4.1) until convergence, that is we execute the fixed point step

$$U^{(k+1)} = \Phi(U^{(k)}). \tag{4.9}$$

$\Phi(U^k)$ is computed by the following algorithm

- Given U^k compute the transmissibilities T_x^u, T_y^u with (4.2) and (4.3);
- Construct the matrix M as in (3.26);
- Solve the linear system (3.26) to get $P_{k,j}$;
- Post-process to get the velocities U^{k+1}, i.e., use equations (3.11) and (3.12) with definitions (4.2) and (4.3).

Now we want to derive sufficient conditions for the iteration (4.9) to converge. It is easier to consider the scalar case of (2.25) first. Thus we rewrite (2.25) as

$$u = -A(K, \beta; u)(p'), \tag{4.10}$$

where p' represents the derivative of pressure with respect to the spatial variable, and K is a positive real number.

Setting it up as a fixed point iteration we have

$$u^{(k+1)} = -A(K, \beta; u^{(k)})p'. \tag{4.11}$$

As required in Theorem 2 , to guarantee the convergence of the fixed point iteration we need to assure that the term $A(K, \beta, u)p'$ in equation (4.11) is a contraction. By Lemma 1 we need a condition to guarantee that

$$|D(A(K, \beta, u)p')| < 1, \tag{4.12}$$

where the operator D represents the derivative with respect to u. Here we consider that p' is data for this problem.

We estimate

$$|D(-\left(K^{-1} + \beta|u|\right)^{-1} p')| = \left|\frac{\beta p'}{(K^{-1} + \beta|u|)^2}\right| < \left|\frac{\beta p'}{(K^{-1})^2}\right|.$$

Therefore, to ensure (4.12) it is sufficient to require

$$\beta < \frac{1}{K^2|p'|}. \tag{4.13}$$

The latter is not the best estimate, but is a good sufficient condition.

Example 1 *Assuming the values* $K = 1$, $p' = -20$ *and* $\beta = \frac{1}{K^2 10 p'}$, initial guess $= 0$, *we find the fixed point* $u = 18.3216$ *of the scalar equation* (4.10) *in 8 iterations.*

When solving the whole nonlinear system (4.1) with a fixed point iteration (4.9) we have convergence only for small enough β as suggested by equation (4.13). The precise form of a condition on β applicable to the system (4.1) will not be derived here.

Remark 2 *If we consider that* β *is correlated to* K *[12, 10, 1] by the relation* $\beta \approx C_1 K^{-C_2}$, *where* C_2 is 0.5, or 1, or 1.5. *For* $C_2 = 0.5$ *the estimate* (4.13) *becomes*

$$\beta < \frac{1}{K^{\frac{3}{2}} |\nabla p|}. \tag{4.14}$$

4.3.1 Boundary Conditions for Non-Darcy Flow

The difference equations used to approximate the three different types of boundary conditions referred in Sec. 3.3. are the same to those in the Darcy flow with the necessary modifications in the transmissibilities, which depend on U.

5. UPSCALING

Upscaling is an important part of computational modeling of heterogeneous porous media. It is commonly used to handle fine scale heterogeneities in subsurface formations where rock properties such as porosity and permeability vary. Due to computational limitations on simulation of flows on a fine scale for real problems, we want to incorporate these fine scale data into a coarse scale flow simulation.

Upscaling has been developed to bridge the gap between these two scales, which may differ by a factor of 100.

To do this, an upscaling algorithm is designated to obtain suitable values for the porosity, permeability and other property data for use in a coarse grid simulation. In this work we follow the algorithm developed by Louis J. Durlofsky [8]. We focus on scale up, by averaging techniques, the fine scale permeabilities to the larger scale permeabilities and other parameters. The "averaged" permeability is referred to as an *effective* or *equivalent* permeability [8, 14].

5.1. Use of Upscaling for Calculating Effective Grid Block Permeability for the Darcy Flow

Following [8] we calculate numerically the effective permeability of heterogeneous porous medium. We consider first a single-phase, incompressible flow described by Darcy's law and continuity in the form of equations (2.1) and (2.4), respectively. Our intent here is to extend these results to the non-Darcy case.

Now we temporarily change the notation of the coordinate system in the flow region Ω. We want to account for a coarse scale $\mathbf{x} = (x_1, x_2)$ and a fine scale $\mathbf{y} = (y_1, y_2)$. To accommodate this change the coefficient $\mathbf{K}$ is now given by

$$\mathbf{K} = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix}. \tag{5.1}$$

It is assumed that the permeability $\mathbf{K}$ varies on two distinct scales, a fine scale $\mathbf{y}$ with fast variation and a coarse scale $\mathbf{x}$ with slow variation.

Now, combining equations (2.1) and (2.4) we get the one-phase pressure equation

$$-\nabla \cdot (\mathbf{K}(\mathbf{x}, \mathbf{y}) \nabla p) = q \tag{5.2}$$

The notation $\mathbf{K}(\mathbf{x}, \mathbf{y})$ is used to emphasize that $\mathbf{K}$ varies on both scales.

We want to find an effective permeability $\mathbf{K}^*$ that varies only on the coarse scale $\mathbf{x}$ and solve the new pressure equation

$$-\nabla_x \cdot (\mathbf{K}^*(\mathbf{x}) \nabla_x p) = q, \tag{5.3}$$

where the subscript x indicates that the gradient operator operates on the $\mathbf{x}$ scale. This simplification is possible as shown in [3] in the context of porous media.

Now we follow ideas from [8]. The average flow over the $\mathbf{y}$ scale, denoted by $\langle \mathbf{u} \rangle$, is related to the $\mathbf{x}$ scale pressure gradient $\mathbf{G}$ by

$$\langle \mathbf{u} \rangle = -\mathbf{K}^* \cdot \mathbf{G}. \tag{5.4}$$

So, to determine $\mathbf{K}^*$ we have to solve the pressure equation on the $\mathbf{y}$ scale, i.e.

$$-\nabla_y \cdot (\mathbf{K}^*(\mathbf{y}) \nabla_y p) = 0 \tag{5.5}$$

subject to the conditions that both the pressure field and the local velocity field must themselves be periodic since the system is periodic. If we decompose the pressure gradient $\mathbf{G}$ into its two components in the $\mathbf{y}$ coordinate system; $\mathbf{G} = G_1 \mathbf{i_1} + G_2 \mathbf{i_2}$, where $\mathbf{i_1}$ and $\mathbf{i_2}$ are the unit coordinate directions, then we have to solve two problems in order to compute the full effective permeability tensor $\mathbf{K}^*$.

5.1.1 Upscaling with periodic boundary conditions

First, assume $G_2 = 0$ and specify the boundary conditions as follows (see Figure 5.1):

$$
\begin{aligned}
p(y_1, y_2 = 0) &= p(y_1, y_2 = 1) && (5.6)\\
&\quad \text{on } \partial D_1 \text{ and } \partial D_2, \\
\mathbf{u}(y_1, y_2 = 0)\cdot \mathbf{n}_1 &= -\mathbf{u}(y_1, y_2 = 1)\cdot \mathbf{n}_2 && (5.7)\\
&\quad \text{on } \partial D_1 \text{ and } \partial D_2, \\
p(y_1 = 0, y_2) &= p(y_1 = 1, y_2) - G_1 && (5.8)\\
&\quad \text{on } \partial D_3 \text{ and } \partial D_4, \\
\mathbf{u}(y_1 = 0, y_2)\cdot \mathbf{n}_3 &= -\mathbf{u}(y_1 = 0, y_2)\cdot \mathbf{n}_4 && (5.9)\\
&\quad \text{on } \partial D_3 \text{ and } \partial D_4,
\end{aligned}
$$

where $\mathbf{n}_i$ $(i = 1, 2, 3, 4)$ is the outward pointing normal at either of the boundaries. Therefore, solving (5.5) subject to (5.6)-(5.9), we are able to determine the average velocity through the $\mathbf{y}$ scale as follows:

$$\langle u_1 \rangle = -\int_{\partial D_3} \mathbf{u}\cdot \mathbf{n}_3 \, dy_2, \tag{5.10}$$

$$\langle u_2 \rangle = -\int_{\partial D_1} \mathbf{u}\cdot \mathbf{n}_1 \, dy_1. \tag{5.11}$$

Thus, by (5.4) we have explicit expression for $\mathbf{K}^*$ in terms of $\langle u_1 \rangle$, $\langle u_2 \rangle$, as follows:

$$\langle u_1 \rangle = -(K^*_{11} G_1 + K^*_{12} G_2), \tag{5.12}$$

$$\langle u_2 \rangle = -(K^*_{21} G_1 + K^*_{22} G_2). \tag{5.13}$$

Since we have assumed that $G_2 = 0$ and G_1, and $\langle \mathbf{u} \rangle$ are known, then K^*_{11} and K^*_{21} are easily determined.

By solving a second problem with $G_1 = 0$ and $G_2 \neq 0$ and similar boundary conditions we can determine K^*_{22} and K^*_{21}.

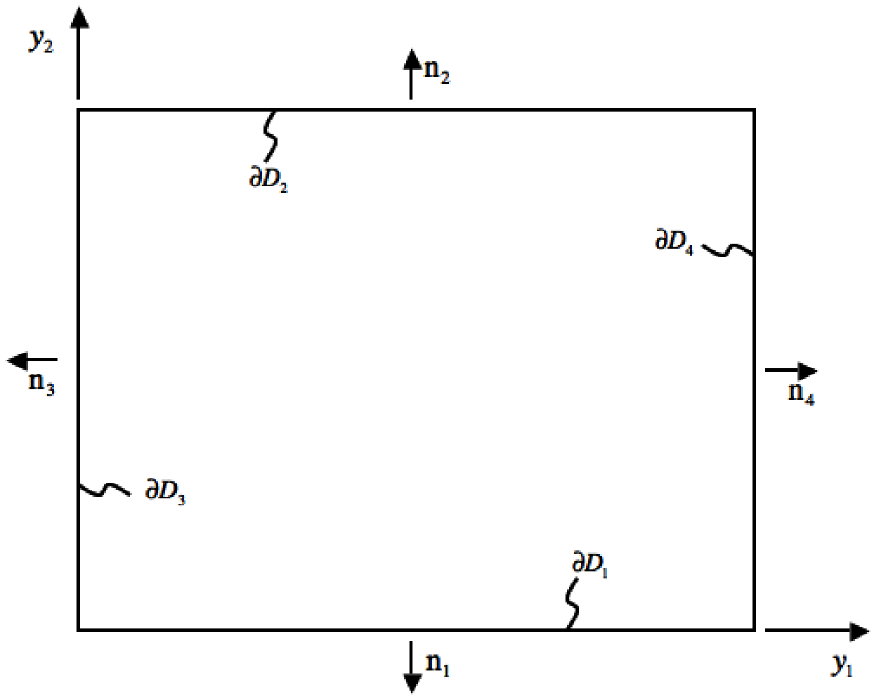

FIGURE 5.1: Boundary specification for effective permeability calculation.

5.1.2 Upscaling with Dirichlet Boundary Conditions

Sometimes it is difficult to use periodic boundary conditions. In addition, the off-diagonal terms K^*_{12} and K^*_{21} cannot be used in simple numerical approximation schemes such as cell centered finite difference.

In such cases instead of periodic boundary conditions (5.6)-(5.9) one may use, to determine K^*_{11}, the following boundary conditions.

On ∂D_1 and ∂D_2 we use no-flow boundary conditions.

On ∂D_3 and ∂D_4 we impose Dirichlet boundary conditions.

$$p(y_1 = 0, y_2) = 0 \tag{5.14}$$

$$p(y_1 = 1, y_2) = G_1. \tag{5.15}$$

Once pressure and velocity are computed, we can then compute the average velocities $\langle u_1 \rangle$

and $\langle u_1 \rangle$ from (5.10) and (5.11). Then to get K^*_{11} we apply a modification of (5.12)

$$\langle u_1 \rangle = -(K^*_{11} G_1) \tag{5.16}$$

Note that off-diagonal terms are note computed and are considered equal to zero. Analogously we can compute K^*_{22}.

5.2. Extension to the Non-Darcy Flow

Now we extend the algorithm presented above for the Darcy case to the non-Darcy case. The main difference is that instead of solving the linear pressure equation (2.1) representing the Darcy flow we solve the non-linear pressure equation (2.22) that describe the non-Darcy flow. This causes some consequences.

We have in (5.4) that $\langle \mathbf{u} \rangle = -\mathbf{K}^* \, \mathbf{G}$, that is $\langle \mathbf{u} \rangle$ depends linearly on $\mathbf{G}$. However in non-Darcy case $\mathbf{u}$ depends nonlinearly on pressure gradients. Therefore $\langle \mathbf{u} \rangle$ depends nonlinearly on $\mathbf{G}$.

For the non-Darcy flow we postulate

$$\langle \mathbf{u} \rangle = -\mathbf{K}^*(\mathbf{u}) \, \mathbf{G} \tag{5.17}$$

This is not a linear relation and so if $\mathbf{G}$ varies then $\mathbf{K}^*$ varies nonlinearly with respect to $\mathbf{G}$. While $\mathbf{K}^*$ remains constant when we vary $\mathbf{G}$ in the Darcy flow, in the non-Darcy flow $\mathbf{K}^*$ varies when we change $\mathbf{G}$.

We can see this difference in Figure 5.2. The parameters used for this example are

$$\mathbf{K}(\mathbf{x}, \mathbf{y}) \equiv \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \tag{5.18}$$

and $\beta = 4 \times 10^{-4}$. The region $\Omega = [0,1] \times [0,1]$ is discretized with a uniform grid 30×30. We use Dirichlet boundary conditions ($p = G_1$ on the left boundary and $p = 0$ on the right boundary) and no-flow boundary conditions on the top and bottom of the domain Ω. We want to find an effective $\mathbf{K}^*$ and β^* but we only show the results for K^*_{11} and β^*_1.

Clearly for Darcy case the effective $\mathbf{K}^*$ should be equal to the original, constant $\mathbf{K}$. This is confirmed in Figure 5.2.

However for non-Darcy case K^*_{11} will depend on $\mathbf{G}$ as shown in Figure 5.2.

In this whole example we use Dirichlet boundary conditions as described in Section 5.1.2.

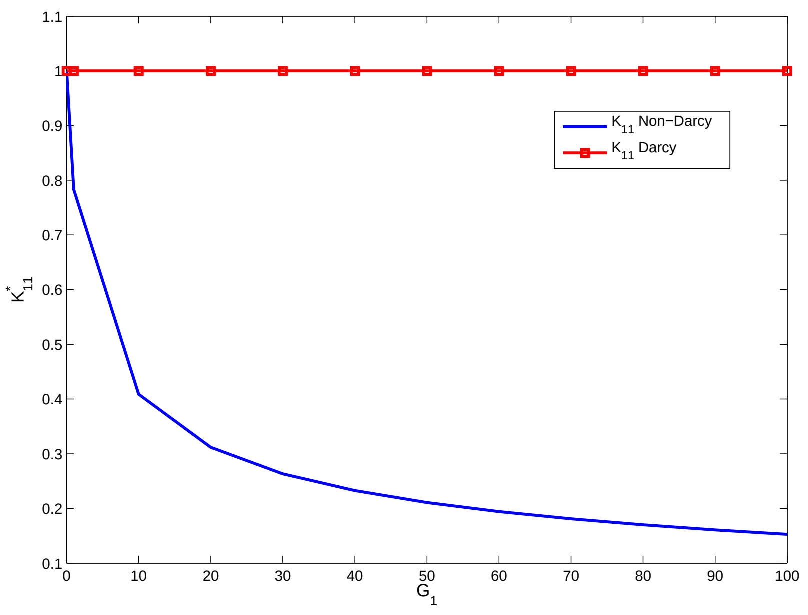

FIGURE 5.2: Variation of the effective permeability K^*_{11} for different values of the pressure gradient G_1.

5.2.1 The Effective β^*

Now we want to find an effective β^*. Solving (5.17) for $K^*_{11}(u_1)$ we get

$$K^*_{11}(\langle u_1 \rangle) = -\langle u_1 \rangle \, G_1^{-1}, \tag{5.19}$$

substituting the Forchheimer's law for the velocity into (5.19) we have

$$((K_{11}^*)_D + \beta|\langle u_1\rangle|)^{-1} = -\frac{\langle u_1\rangle}{G_1} \tag{5.20}$$

where the subscript $_D$ represents the Darcy case.

Now, we solve for β to get

$$\beta^* = |\langle u_1\rangle|^{-1}\left(K_{11}^*(\langle u_1\rangle)^{-1} - ((K_{11}^*)_D)^{-1}\right). \tag{5.21}$$

We refer to the above parameter as the effective Forchheimer parameter that will be denote as $\beta^*(\langle u_1\rangle)$. Now we use the same data as before to find effective β^* by using expression (5.21) and we plot $\beta^*(\langle u_1\rangle)$ against the velocity $\langle u_1\rangle$ in Figure 5.3. We observe that the variation in β^* is small if compared with the variation in $\langle u_1\rangle$.

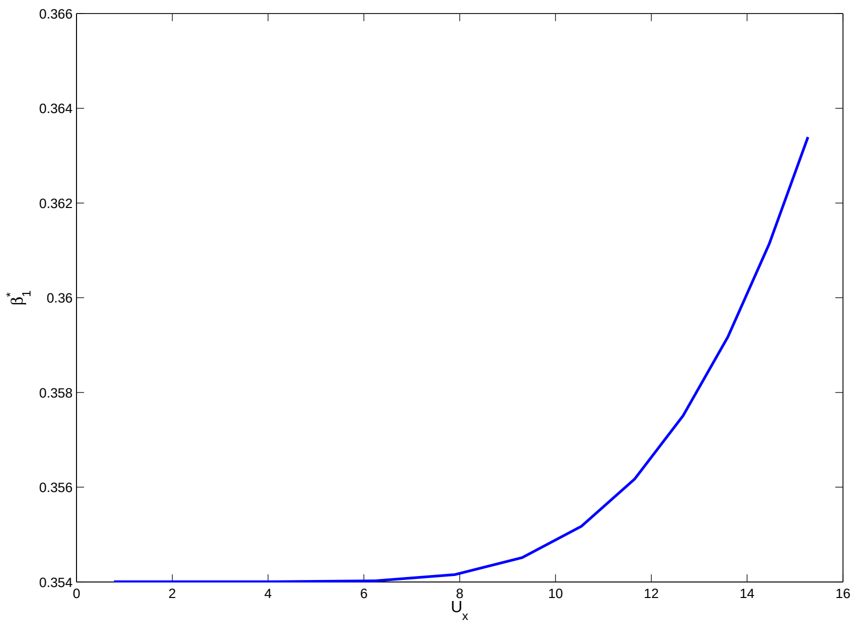

FIGURE 5.3: Variation of the effective Forchheimer parameter $\beta^*(\mathbf{u})$ for different values of velocity.

6. NUMERICAL EXPERIMENTS

In this chapter we present numerical experiments which show applications of cell-center finite difference method to non-Darcy flow and upscaling .

We start with a simple example of a well problem in order to compare the solutions for the Darcy and non-Darcy flows. After that we use the ideas of Chapter 5 to compute the effective permeability of an idealized heterogeneous porous medium. We solve for both Darcy's and non-Darcy's flow. An algorithm used to compare the computed effective permeability with original permeability of the system is described. The code used in these experiments is written in Matlab. The linear solver is the "backslash" operator of Matlab, which utilizes Gaussian elimination to solve the linear system.

6.1. Problem 1 - Well Problem for Darcy and Non-Darcy flow

Here we solve numerically equations (3.4) for Darcy and (4.1) for non-Darcy for pressure. In each case we have two production and two injection wells. We use no-flow boundary conditions for the unit square domain. The permeability $\mathbf{K}$ is chosen to be homogeneous and isotropic flow, i.e., $\mathbf{K} = I$, and $x \in \Omega$. The Forchheimer's coefficient β in the non-Darcy equation is a chosen to be a constant value equal to 0.04. The cartesian coordinates of the injection wells are $(0.3, 0.3)$ and $(0.6, 0.2)$; and the coordinates of the production wells are $(0.7, 0.7)$ and $(0.3, 0.6)$. The grid is (20×20). The results are shown in Figure 6.1 and Figure 6.2.

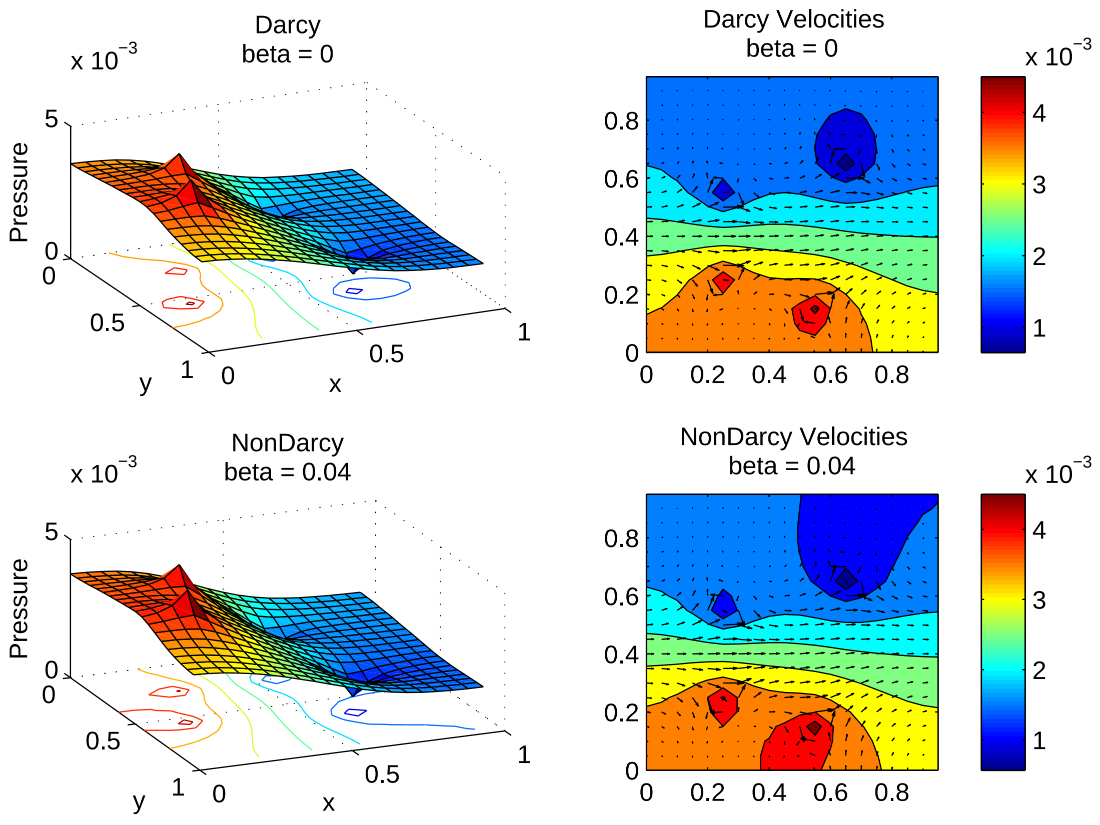

FIGURE 6.1: Pressure distribution and velocity field for two-production/ two-injection well problem simulation

6.2. Problem 2 - Darcy's Flow Upscaling

Here we assume that Ω is the unit square which is divided in a 30×30 grid and $\mathbf{K}$ represents a heterogeneous field with values ranging from 1 to 610 (see Figure 6.8 (a)). We assume $K_{11}(x) = K_{22}(x)$.

Case 1

We coarsen the original grid in to a 3×3 grid in order to find the effective permeability for these nine new grid blocks. Therefore we solve the pressure equation 9 times

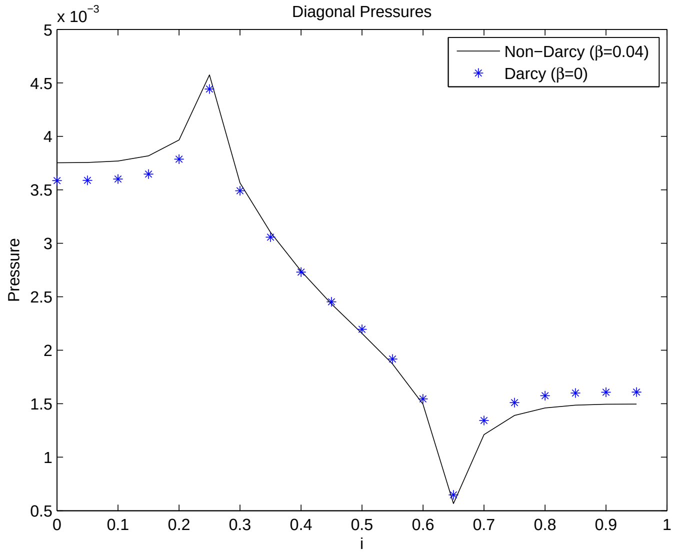

FIGURE 6.2: Values of Darcy and non-Darcy pressure on diagonal grid cells for two-production/ two-injection well problem simulation.

in a grid 10×10. The effective permeability is then calculated as in (5.16). The following table presents the results for K^*_{11}. It has to be viewed as the new grid over the unit square with the values of the effective permeabilities found for each subregion.

$K^*_{11} =$ (6.1)

23.231	25.291	25.954
24.127	22.887	20.785
21.938	23.684	22.651

Now, we find the K^*_{22} analogously, values are given in the next table

$K^*_{22} =$ (6.2)

23.404	26.087	24.131
25.651	22.143	20.995
25.103	21.607	24.387

Case 2

Now we divide the original 30×30 grid in a 6×6 grid. Using the same boundary conditions a the same data we find the the following values for effective permeabilities:

$K^*_{11} =$ (6.3)

23.429	22.355	36.936	20.226	25.251	25.895
18.119	33.64	32.128	24.093	21.251	34.061
32.969	28.948	25.576	26.681	23.885	20.437
16.187	29.297	24.242	17.301	14.831	26.299
34.124	20.749	21.081	28.734	20.838	15.999
12.41	30.877	23.234	22.361	33.798	26.456

$K^*_{22} =$ (6.4)

20.131	22.462	36.138	23.183	24.718	23.215
20.371	30.576	24.969	25.922	22.475	29.548
25.564	28.312	21.462	28.861	25.192	20.621
18.134	33.931	25.439	16.571	16.271	28.266
43.692	21.885	19.843	20.963	26.805	17.308
14.997	28.622	27.557	21.268	28.204	27.981

Application for a Darcy well model

With these results in hands we now solve the pressure equation for Darcy's flow over the unit square with a production well in the position $(0.3, 0.3)$ and a injection well in the position $(0.7, 0.7)$. We assume no-flow boundary conditions.

We first solve the problem using the original grid and plot the pressure in Figure 6.3.

The velocity field is shown in Figure 6.4 and the values of the pressure in the diagonal grid-cells is shown in Figure 6.5.

We use the effective permeability found before to solve the same well problem. Here we present the results for the case where we divided the original grid in a (6×6) grid.

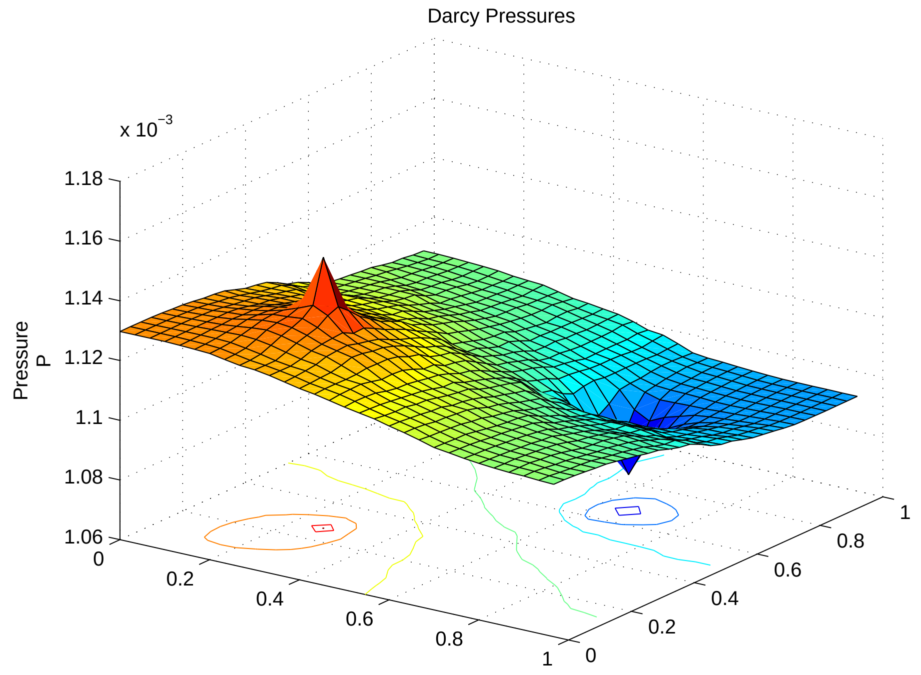

FIGURE 6.3: Pressure distribution on the unit square for problem using the original permeability

Comparison between Figure 6.4 and Figure 6.6 show that the behavior of the solution on the fine grid is qualitative similar to the one in coarse grid when we use the effective permeability.

6.2.1 Comparison of Pressures on Fine and Coarse Scale

Now we ask the question *"How do we know if the effective permeability is a good approximation?"*. We want to compare the results but they are in different scales. The strategy we used is the following

- Take the effective permeability calculated for certain subregion of our domain;
- Use it in each cell of the original scale in such region, that is, we replace the heterogeneous permeability in that region for a effective permeability $\mathbf{K}^*$, we are making

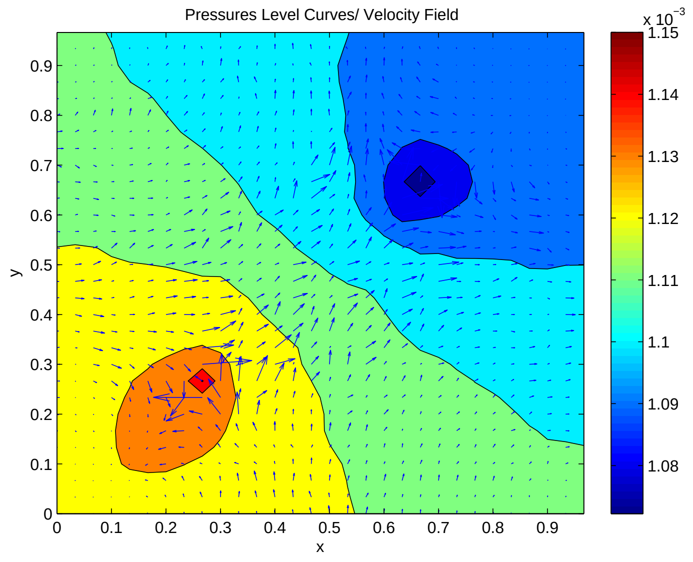

FIGURE 6.4: Velocity field using the original permeability tensor

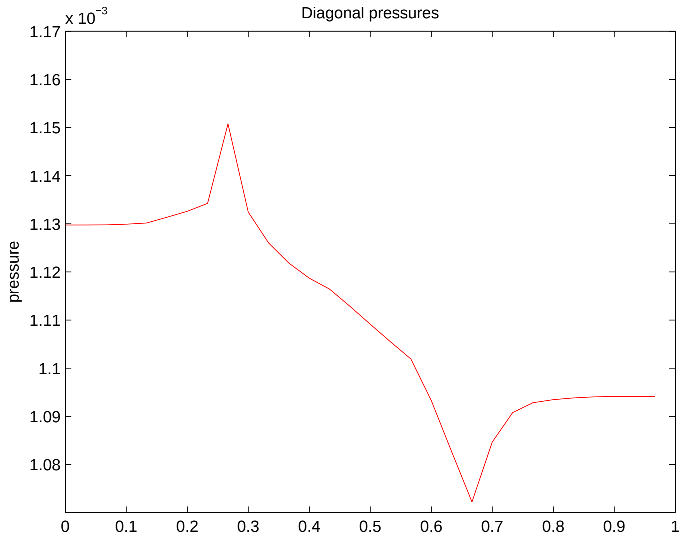

FIGURE 6.5: Diagonal values of pressure using the original permeability tensor on fine grid.

that region to have an "homogenized" permeability;

- Repeat for each subregion of the original domain to get a block-homogenized domain;

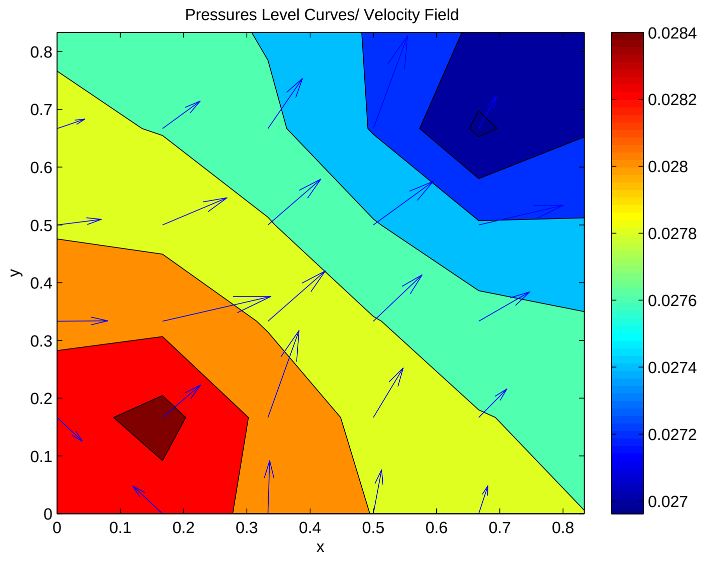

FIGURE 6.6: Velocity field using the effective permeability $\mathbf{K}^*$ on the coarse grid 6×6

- Solve the pressure equation to get homogenized pressure $P_H^\star$;
- Compare P_h and P_H^*.

This process is summarized, for a 4×4 grid, in Figure 6.7.

Using the effective permeabilities from **case 1** and **case 2** and the algorithm above we compute the pressures P_h, $P_H^{*,1}$, $P_H^{*,2}$, where the superscripts 1 and 2 refer to the cases 1 and 2, respectively. The 2-norm and ∞-norm of the pressures calculated using the Matlab code provided in the Appendix are shown in the Table 6.1.

	Original permeabilities	Upscaling to 3×3	Upscaling to 6×6
$\|P\|_\infty$	0.0011507511	0.001149681	0.0011440731
$\|P\|_2$	0.033335256835	0.03333522	0.033335210

TABLE 6.1: Norm of the solutions for original problem, **case 1**, **case 2**.

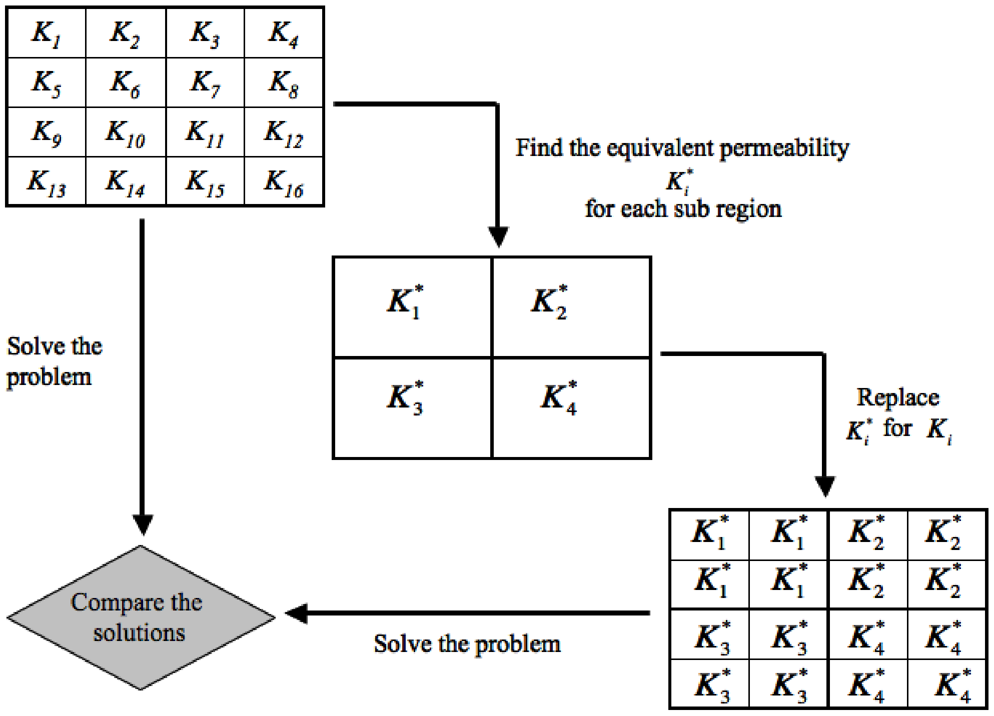
K_1 K_2 K_3 K_4
K_5 K_6 K_7 K_8
K_9 K_{10} K_{11} K_{12}
K_{13} K_{14} K_{15} K_{16}
Find the equivalent permeability K_i^* for each sub region
K_1^* K_2^* K_3^* K_4^*
Replace K_i^* for K_i
Solve the problem
Compare the solutions
Solve the problem

FIGURE 6.7: Process of comparison between fine and coarse scales.

We also compare the the absolute error between the original pressure and the homogenized pressure for both cases. These results are present in Table 6.2 and the comparison between the values of the pressure in the diagonal grid blocks are shown in Figure 6.9. The original and upscaled permeabilities are shown in Figure 6.8.

6.3. Problem 3 - Non-Darcy's Flow Upscaling

Consider the equation

$$-\nabla\left(A(\mathbf{K},\beta;\mathbf{u})\nabla p\right) = q, \qquad \mathbf{x}\in\Omega \tag{6.5}$$

where $A(\mathbf{K},\beta;\mathbf{u}) = \left(\mu\mathbf{K}^{-1} + \beta|\mathbf{u}|\right)^{-1}$, Ω is the unit square subject to the constant pressure and no-flow boundary conditions as in **problem 1**. We use a 30×30 grid discretization,

	Upscaling to 3×3	Upscaling to 6×6
$\|P_h - P_H^*\|_\infty$	7.25774×10^{-6}	6.79880×10^{-6}
$\|P_h - P_H^*\|_2$	2.49395×10^{-5}	2.75298×10^{-5}
$\frac{\|P_h - P_H^*\|_\infty}{\|P_h\|_\infty}$	0.00630	0.00590
$\frac{\|P_h - P_H^*\|_2}{\|P_h\|_2}$	8.33365×10^{-4}	8.2584×10^{-4}

TABLE 6.2: Absolute and relative errors between pressures of **cases 1** and **2** and the original problem.

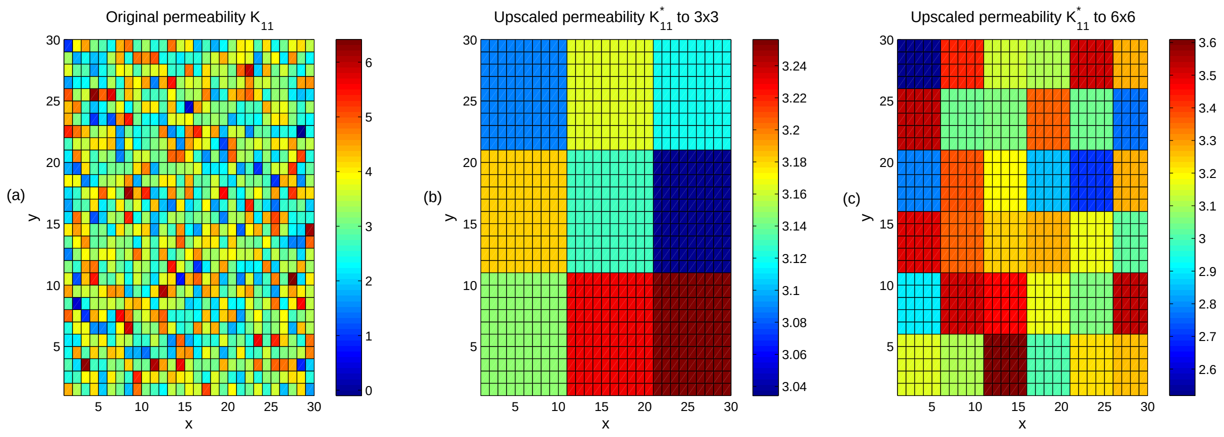

FIGURE 6.8: Log of original permeability K_{11} and permeabilities K_{11}^* for **case 1** and **case 2** for the Darcy flow.

K is given as in **problem 1**, and we use a variable Forchheimer parameter β given by the relation

$$\beta = C_1 (K_{**})_{k,j}^{-C_2},$$

where $C_1 = 4 \times 10^{-5}$ and $C_2 = 0.5$, $** = xx$ or yy. As we showed in section 5.2 the effective $\mathbf{K}^*$ depends nonlinearly on $\mathbf{G}$. In examples bellow we only compute $\mathbf{K}^*$ using unit value of G. The effective $\mathbf{K}^*$ is different from the one obtained in **problem 2**.

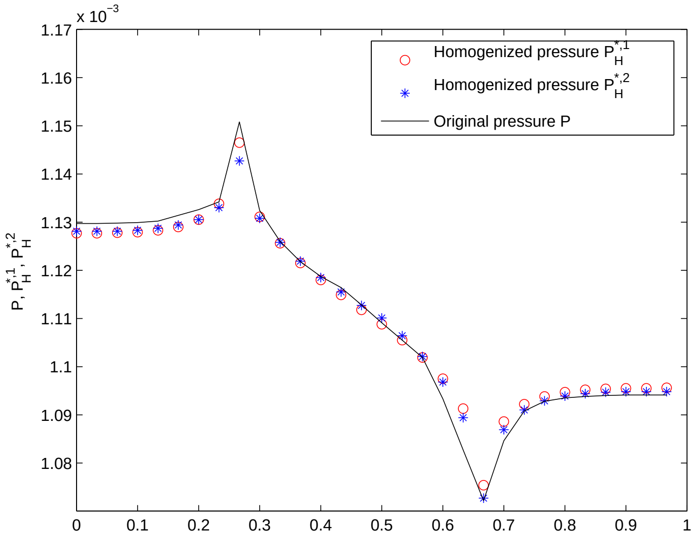

FIGURE 6.9: Pressures in the diagonal grid blocks for the Darcy Flow

Case 1

Here again we coarsen the original grid to a 3×3 grid. Using the algorithm described in Section 5.2. with $G_1 = 1$ we find K_{11}^*:

$K_{11}^* =$ (6.6)

16.988	17.995	18.226
17.478	16.735	15.547
16.223	17.091	16.865

and similarly with $G_2 = 1$ we find K_{22}^*:

$K_{22}^* =$ (6.7)

16.936	18.104	17.630
18.035	16.516	15.773
17.440	16.176	17.600

Case 2

The original grid is subdivided in eighteen sub-regions, i.e. we obtain a 6×6 grid. The effective permeabilities are calculated and are given by

$$K_{11}^{*} = \begin{array}{|c|c|c|c|c|c|}\hline 20.964 & 20.096 & 28.613 & 18.703 & 22.652 & 22.835 \\ \hline 16.714 & 28.147 & 26.221 & 21.346 & 19.080 & 28.095 \\ \hline 26.390 & 25.381 & 22.506 & 23.370 & 21.534 & 18.631 \\ \hline 15.136 & 25.576 & 21.652 & 16.167 & 13.853 & 22.901 \\ \hline 27.768 & 18.968 & 19.082 & 24.755 & 19.216 & 14.865 \\ \hline 11.939 & 25.567 & 21.136 & 19.835 & 28.485 & 23.582 \\ \hline \end{array} \qquad (6.8)$$

$$K_{22}^{*} = \begin{array}{|c|c|c|c|c|c|}\hline 18.620 & 20.341 & 28.794 & 20.840 & 22.183 & 20.870 \\ \hline 18.035 & 26.555 & 21.801 & 22.545 & 20.062 & 25.680 \\ \hline 22.696 & 24.941 & 19.767 & 24.869 & 22.500 & 18.624 \\ \hline 16.725 & 27.517 & 22.468 & 15.488 & 14.996 & 24.060 \\ \hline 33.578 & 19.794 & 18.129 & 19.122 & 23.385 & 15.892 \\ \hline 14.108 & 24.975 & 23.975 & 19.219 & 24.753 & 24.934 \\ \hline \end{array} \qquad (6.9)$$

Non-Darcy Well Problem Using Upscaling

Consider equation (6.5) subject to no-flow boundary conditions on the unit square with injection and production wells located at $(0.3, 0.3)$ and $(0.7, 0.7)$ respectively. We solve for pressure to get the approximated pressure which is shown in Figure 6.10.

Now we solve the problem using the effective permeabilities from **case 1** and **case 2** and use numerical "homogenization" algorithm described before to compare the results. Again we get satisfactory results that are shown in Table 6.3. In Figure 6.11 we show the three distributions for the original, upscaled to 3x3 grid and upscaled to a 6x6 grid permeabilities, respectively.

Figure 6.12 shows the plot of the diagonal values for pressure for the three different permeabilities used.

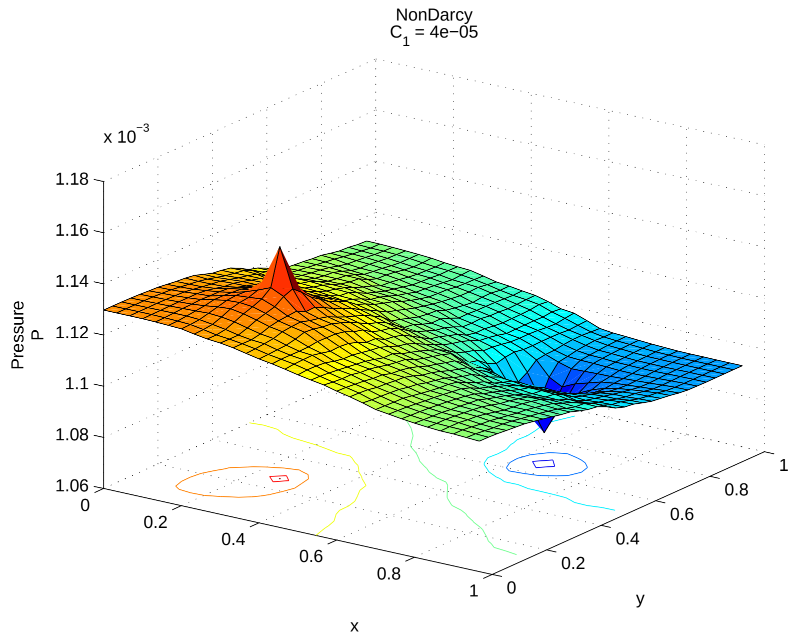

FIGURE 6.10: Pressure distribution on the unit square using the original permeability tensor

	Original permeabilities	Upscaling to 3×3	Upscaling to 6×6
$\|P\|_\infty$	0.001150755374	0.001164049926	0.001148355739
$\|P\|_2$	0.033335258601	0.033336802473	0.033335560748

TABLE 6.3: Norm of the solutions for original problem, **case 1**, **case 2**

	Upscaling to 3×3	Upscaling to 6×6
$\|P_h - P_H^*\|_\infty$	1.324992×10^{-5}	4.6291×10^{-6}
$\|P_h - P_H^*\|_2$	2.263433×10^{-5}	3.950248×10^{-5}
$\frac{\|P_h - P_H^*\|_\infty}{\|P_h\|_\infty}$	0.0115	0.0040
$\frac{\|P_h - P_H^*\|_2}{\|P_h\|_2}$	0.0038	0.0012

TABLE 6.4: Absolute and relative errors between pressures of **cases 1** and **2** and the original well problem for the non-Darcy flow.

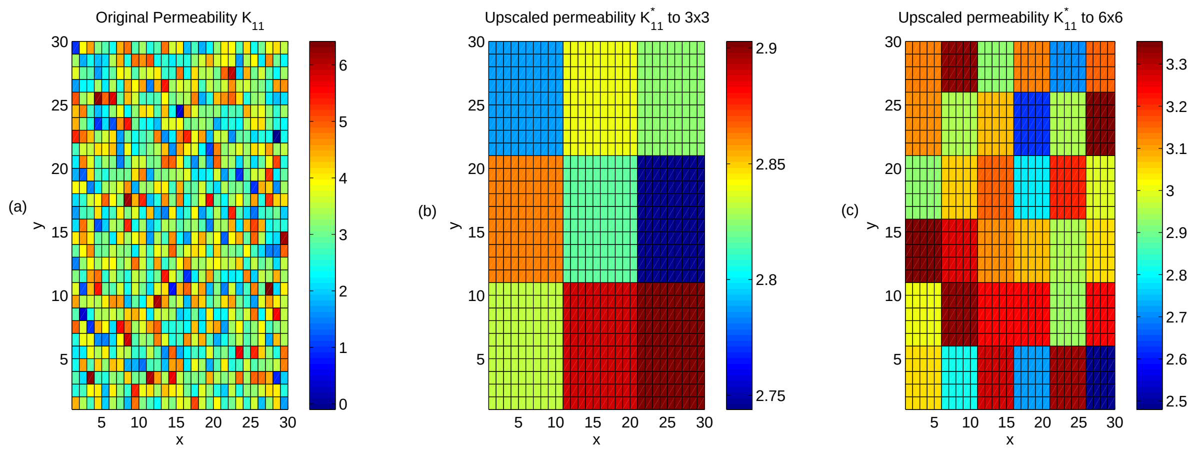

FIGURE 6.11: Original permeabilities and permeabilities for **case 1** and **case 2** for non-Darcy flow

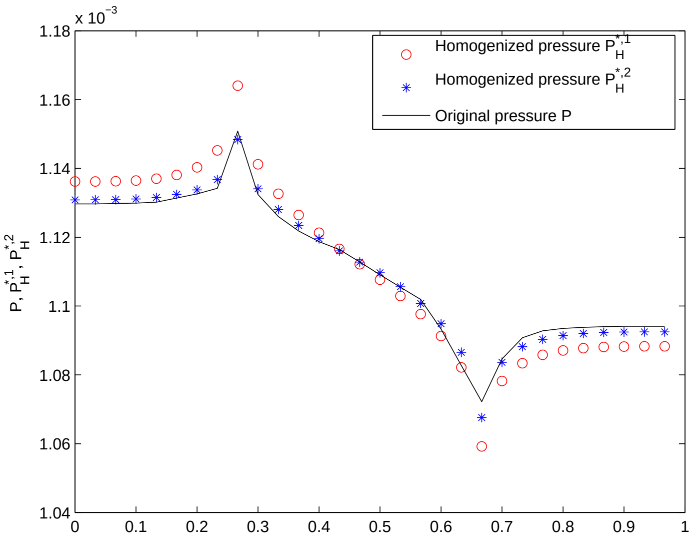

FIGURE 6.12: Pressures in the diagonal grid blocks

7. CONCLUSIONS

In this work we have understood and outlined the theoretical difficulties of the proposed problem. We have implemented a stable numerical solver for a well problem for non-Darcy flow using finite differences and the fixed point iteration, for which we have derived a sufficient condition on the Forchheimer parameter β that guarantee the convergence of the method.

The upscaling problem was theoretically presented and a numerical solver was implemented for the non-Darcy flow. We have shown how to calculate an effective permeability and how to calculate an effective Forchheimer parameter β^*.We also developed an algorithm to compare the solution for the different ways to coarsen the original fine grid.

For continued research, we would like to implement a nonlinear solver different from fixed point iteration, for example Newton's method. This is necessary because of restrictive conditions on β required for convergence of fixed point method.

Next we would like to perform more experiments for a large variability in the permeability $\mathbf{K}$ and parameter β, to understand better the nonlinear relations between $\mathbf{K}$, β, and $\mathbf{u}$. We also want to understand and derive the effective Forchheimer parameter β^*. Finally better ways to measure the quality of upscaling for Darcy and non-Darcy cases are necessary.

APPENDIX

A Matlab Code

```
% Code for solving the Forchhimer Equation on a rectangular domain.

% Using Block(cell-center) finite difference with "fixed point iteration".



clear

clc

format long


%Initialization of variables

n_x=30; n_y=30; BoundCond=1;

LeftRight=0; BottonTop=1; G1=1; G2=1;


nxnew=3;nynew=3;ng=1;

xx=1;yy=1;scalex=n_x/nxnew;scaley=n_y/nynew; factorx=scalex; factory=scaley;

n_x=factorx; n_y=factory;



if BoundCond == 1

    M=zeros((n_x*n_y),(n_x*n_y));

    q=zeros(n_x*n_y,1);

elseif BoundCond == 2

    M=zeros((n_x*n_y)+1,(n_x*n_y));

    M((n_x*n_y)+1,:)=1;
    q=zeros(n_x*n_y+1,1);
elseif (BoundCond == 3 && LeftRight == 1)
    M=zeros((n_x*n_y)+n_y+1,(n_x*n_y)+2*n_y);
    M((n_x*n_y)+n_y+1,:)=1;
    q=zeros(n_x*n_y+n_y+1,1);
elseif (BoundCond == 3 && BottonTop == 1)
    M=zeros((n_x*n_y)+n_x+1,(n_x*n_y)+2*n_x);
    M((n_x*n_y)+n_x+1,:)=1;
    q=zeros(n_x*n_y+n_x+1,1);
end
Th=zeros((n_x+1),(n_y));
Tv=zeros((n_x),(n_y+1));
beta1=zeros((n_x+1),(n_y));
beta2=zeros((n_x),(n_y+1));
Vx=zeros(n_x+1,n_y);
VxNew=Vx;
Vy=zeros(n_x,n_y+1);
VyNew=Vy;
beta=.00004;
a=0; b=1; c=0; d=1;
bc=zeros(n_x*n_y,1);

%Position of wells in the domain
x1=0.3; y1=0.3;
x2=0.9; y2=0.9;

%Grid definition.
```

```
hx=(b-a)/n_x; hy=(d-c)/n_y;

%%% Wells
k1=1; j1=1; k2=1; j2=1;
while x1 > k1*hx
    k1=k1+1;
end
while y1 > j1*hy
    j1=j1+1;
end

while x2 > k2*hx
    k2=k2+1;
end
while y2 > j2*hy
    j2=j2+1;
end


%%%Data for test%
load permeability.dat;
load bcl.dat
load bcr.dat


%Defining the index function
for k=1:n_x+2
```

```
    for j=1:n_y

        ind(k,j)=(k-1)*n_y +j;

    end

end


for k=1:n_x

    for j=1:n_y+2

        ind2(k,j)=(j-1)*n_x +k;

    end

end


for countx=1:nxnew

    for county=1:nynew

        K1=permeability;

        K2=permeability;

        K1=K1(xx:scalex,yy:scaley);

        K2=K2(xx:scalex,yy:scaley);

        beta1=beta*(K1.^(-.5));

        beta2=beta*(K2.^(-.5));

        yy=scaley+1;

        scaley=scaley+factory;


%Transmissibilities (Internal blocks)


for k = 1:n_x-1

    for j = 1:n_y
```

```
        Th(k+1,j)=(2*(K1(k,j)*K1(k+1,j))/((K1(k+1,j)+K1(k,j)))))/hx;

        betax(k+1,j)=(0.5*(beta1(k,j)+beta1(k+1,j)))/hx;

    end

end


for k=1:n_x

    for j=1:n_y-1

        Tv(k,j+1)=(2*(K2(k,j)*K2(k,j+1))/((K2(k,j+1)+K2(k,j)))))/hy;

        betay(k,j+1)=(0.5*(beta2(k,j)+beta2(k,j+1)))/hy;

    end

end



% Transmissibilities on the boundary


if (BoundCond == 1 || BoundCond == 3)

    if (LeftRight == 1 && BottonTop==1)

        for j=1:n_y

        % On the left side

            Th(1,j)=2*K1(1,j)/hx;

            betax(1,j)=beta1(1,j)/2;

        % On the right side

            Th(n_x+1,j)=2*K1(n_x,j)/hx;

            betax(n_x+1,j)=beta1(n_x,j)/2;

        end

        for k=1:n_x

        % On the botton
```

```
            Tv(k,1)=2*K2(k,1)/hy;

            betay(k,1)=beta2(k,1)/2;

        % On the top

            Tv(k,n_y+1)=2*K2(k,n_y)/hy;

            betay(k,n_y+1)=beta2(k,n_y);

        end

    elseif (LeftRight == 1 && BottonTop ~= 1)

        for j=1:n_y

        % On the left side

            Th(1,j)=2*K1(1,j)/hx;

            betax(1,j)=beta1(1,j)/2;

        % On the right side

            Th(n_x+1,j)=2*K1(n_x,j)/hx;

            betax(n_x+1,j)=beta1(n_x,j)/2;

        end

    elseif (LeftRight ~= 1 && BottonTop == 1)

        for k=1:n_x

        % On the botton

            Tv(k,1)=2*K2(k,1)/hy;

            betay(k,1)=beta2(k,1)/2;

        % On the top

            Tv(k,n_y+1)=2*K2(k,n_y)/hy;

            betay(k,n_y+1)=beta2(k,n_y);

        end

    end


end
```

```
%Defining the right hand side q

%%%%%%%%%%%%%%%%%%%%%    Dirichlet BC
if BoundCond == 1
    if (LeftRight == 1 && BottonTop==1)
        for j=1:n_y
            q(ind(1,j))=hy*Th(1,j)*bcl(j);
            q(ind(n_x,j))=hy*Th(n_x,j)*bcr(j);
        end
        for k=1:n_x
            q(ind(k,1))=hx*Tv(k,1)*bcb(k);
            q(ind(k,n_y))=hx*Tv(k,n_y)*bct(k);
        end
    elseif (LeftRight == 1 && BottonTop ~= 1)
        for j=1:n_y
            q(ind(1,j))=hy*Th(1,j)*bcl(j);
            q(ind(n_x,j))=hy*Th(n_x,j)*bcr(j);
        end
    elseif (LeftRight ~= 1 && BottonTop == 1)
        for k=1:n_x
            q(ind(k,1))=hx*Tv(k,1)*bcb(k);
            q(ind(k,n_y))=hx*Tv(k,n_y)*bct(k);
        end
    end
%%%%%%%%%%%%%%%%%%%%%    Neuman BC
elseif BoundCond == 2
```

```
       q(ind(k1,j1))=hx*hy*1;

       q(ind(k2,j2))=hx*hy*(-1);

       q(n_x*n_y + 1)=hx*hy*(n_x*n_y);


%%%%%%%%%%%%%%%%%%%%%   Periodic BC

elseif (BoundCond == 3 && LeftRight == 1)

    for j=1:n_y

        q((n_x*n_y+j))=1;

    end

    q((n_x*n_y+n_y+1))=n_x*n_y+n_y+1;

elseif (BoundCond == 3 && BottonTop == 1)

    for j=1:n_x

        q((n_x*n_y+j))=1;

    end

    q((n_x*n_y+n_x+1))=n_x*n_y+n_x+1;

end


%%%%%%%%%%%%%%%%%%%%%%%%% Stiffness Matrix %%%%%%%%%%%%%%%%

 for k=1:n_x

     for j=1:n_y

         M(ind(k,j),ind(k,j))=(hx*Tv(k,j)+hy*Th(k+1,j)+hy*Th(k,j)+hx*Tv(k,j+1));

     end

 end


 for k=1:n_x-1

     for j=1:n_y
```

```
            M(ind(k,j),ind(k+1,j)) = -hy*Th(k+1,j);
            M(ind(k+1,j),ind(k,j)) = M(ind(k,j),ind(k+1,j));
        end
    end
   for k=1:n_x-1
        for j=1:n_y
            M(ind(k+1,j),ind(k,j)) = M(ind(k,j),ind(k+1,j));
        end
    end
    for k=1:n_x
        for j=1:n_y-1
            M(ind(k,j),ind(k,j+1)) = -hx*Tv(k,j+1);
            M(ind(k,j+1),ind(k,j)) = M(ind(k,j),ind(k,j+1));
        end
    end

   if BoundCond == 3
       if (LeftRight==1 && BottonTop==1)
           for j=1:n_y
               M(ind(1,j),ind(n_x+1,j)) = -hy*Th(1,j);
               M(ind(n_x,j),ind(n_x+2,j))= -hy*Th(n_x+1,j);
               M(ind(n_x+1,j),ind(n_x+1,j))=1;
               M(ind(n_x+1,j),ind(n_x+2,j))=-1;
           end
           for j=1:n_y
               M(ind(1,j),ind(n_x+1,j)) = -hx*Tv(1,j);
               M(ind(n_x,j),ind(n_x+2,j))= -hx*Tv(n_x+1,j);

               M(ind(n_x+1,j),ind(n_x+1,j))=1;

               M(ind(n_x+1,j),ind(n_x+2,j))=-1;

           end

       elseif (LeftRight==1 && BottonTop ~= 1)

           for j=1:n_y

               M(ind(1,j),ind(n_x+1,j)) = -hy*Th(1,j);

               M(ind(n_x,j),ind(n_x+2,j))= -hy*Th(n_x+1,j);

               M(ind(n_x+1,j),ind(n_x+1,j))=1;

               M(ind(n_x+1,j),ind(n_x+2,j))=-1;

           end

       elseif (LeftRight ~= 1 && BottonTop == 1)

           for j=1:n_x

               M(ind2(j,1),ind2(j,n_y+1)) = -hx*Tv(j,1);

               M(ind2(j,n_y),ind2(j,n_y+2))= -hx*Tv(j,n_y+1);

               M(ind2(j,n_y+1),ind2(j,n_y+1))=1;

               M(ind2(j,n_y+1),ind2(j,n_y+2))=-1;

           end

       end

   end


   %%%%%%%%%%%%%%Solve the linear system for Darcy flow %%%%%%


   p=M\q;


   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%Rewrite p as a matrix
for k = 1:n_x
    for j=1:n_y
        P(k,j)=p(ind(k,j));
    end
end
%Figure
x=a:hx:b-hx;
y=c:hy:d-hy;
[X,Y]=meshgrid(y,x);
figure(1)
subplot(2,2,1)
surfc(X,Y,P)
title({'Darcy';'beta = 0'})
az = 35;
el = 26;
view(az, el);
subplot(2,2,2)
contourf(X,Y,P)
if (n_x == n_y)
    for i=1:n_y
        PdiagonalD(i)=P(i,i);
    end
end
Pdarcy=P;
%%%%%%%%%%%%%%%%%%%% Initial Guess %%%%%%%%%%%%%%%%%%%
if BoundCond == 1
```

```
    if (LeftRight == 1 && BottonTop==1)

        for j=1:n_y

            Vx(1,j)=-Th(1,j)*(P(1,j)-bcl(ind(1,j)));

            Vx(n_x+1,j)=-Th(n_x+1,j)*(-P(n_x,j)+bcr(ind(n_x,j)));

        end

        for k=1:n_x

            Vy(k,1)=-Tv(k,1)*(P(k,1)-bcb(ind(k,1)));

            Vy(k,n_y+1)=-Tv(k,n_y+1)*(-P(k,n_y)+bct(ind(k,n_y)));

        end

    elseif (LeftRight == 1 && BottonTop ~= 1)

        for j=1:n_y

            Vx(1,j)=-Th(1,j)*(P(1,j)-bcl(ind(1,j)));

            Vx(n_x+1,j)=-Th(n_x+1,j)*(-P(n_x,j)+bcr(ind(n_x,j)));

        end

    elseif (LeftRight ~= 1 && BottonTop == 1)

        for k=1:n_x

            Vy(k,1)=-Tv(k,1)*(P(k,1)-bcb(ind(k,1)));

            Vy(k,n_y+1)=-Tv(k,n_y+1)*(-P(k,n_y)+bct(ind(k,n_y)));

        end

    end

end

if BoundCond == 3

     if (LeftRight == 1 && BottonTop ~= 1)

       for j=1:n_y

           Vx(1,j)=-Th(1,j)*(-p(n_x*n_y+j)+p(j));

           Vx(n_x+1,j)=-Th(n_x+1,j)*(-p(n_x*n_y-n_y+j)+p(n_x*n_y+j+n_y));

       end
     elseif (LeftRight ~= 1 && BottonTop == 1)

       for j=1:n_x

           Vy(1,j)=-Tv(j,1)*(-p(n_x*n_y+j)+p(j));

           Vy(j,n_y+1)=-Tv(j,n_y+1)*(-p(n_x*n_y-n_x+j)+p(n_x*n_y+j+n_x));

       end

     end

end


%Internal nodes

for k=1:n_x-1

    for j=1:n_y

        Vx(k+1,j)=-Th(k+1,j)*(P(k+1,j)-P(k,j))/hx;

    end

end


for k=1:n_x

    for j=1:n_y-1

        Vy(k,j+1)=-Tv(k,j+1)*(P(k,j+1)-P(k,j))/hy;

    end

end


%Figure

for k=1:n_x

    for j=1:n_y

        VxAvrg(k,j)=(Vx(k+1,j)+Vx(k,j))/2;

        VyAvrg(k,j)=(Vy(k,j+1)+Vy(k,j))/2;

    end
```

```
end
 figure(2)
 quiver(X,Y,VxAvrg,VyAvrg)
 title({'Darcy Velocities';'beta = 0'})

%initialization of velocities for fixed point
Vx0=Vx;
Vy0=Vy;



%%%%%%% ******** Start fixed point iterations ********** %%%%%%%%%%%%%%%%

iter=1; Err_x=1; Err_y=1;
while (iter < 30 && (Err_y > 10^(-6) || Err_x > 10^(-6)))



%%%%%%%%%%%%%%%%%% Transmissibilities %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Transmissibilities (internal nodes)
for k = 1:n_x-1
    for j = 1:n_y
        Th(k+1,j)=((K1(k,j)+K1(k+1,j))/(2*(K1(k+1,j)*K1(k,j)))+betax(k+1,j)...
            *Vx(k+1,j))^(-1)/hx;
    end
end
%Th
for k=1:n_x
```

```
    for j=1:n_y-1
        Tv(k,j+1)=((K2(k,j)+K2(k,j+1))/(2*(K2(k,j+1)*K2(k,j)))+betay(k,j+1)...
            *Vy(k,j+1))^(-1)/hy;
    end
end


%Transmissibilities on the boundary
if (BoundCond == 1 || BoundCond == 3)
%Dirichlet Boundary Conditions
    if (LeftRight == 1 && BottonTop==1)
        for j=1:n_y
        % On the left side
            Th(1,j)=2*((K1(1,j))^(-1)+ betax(1,j)*Vx(1,j))^(-1)/hx;
        % On the right side
            Th(n_x+1,j)=2*((K1(n_x,j))^(-1)+ betax(n_x,j)*Vx(n_x,j))^(-1)/hx;
        end
        for k=1:n_x
        % On the botton
            Tv(k,1)=2*((K2(k,1))^(-1)+ betay(k,1)*Vy(k,1))^(-1)/hy;
        % On the top
            Tv(k,n_y+1)=2*((K2(k,n_y))^(-1)+ betay(k,n_y)*Vy(k,n_y))^(-1)/hy;
        end
    elseif (LeftRight == 1 && BottonTop ~= 1)
        for j=1:n_y
        % On the left side
            Th(1,j)=2*((K1(1,j))^(-1)+ betax(1,j)*Vx(1,j))^(-1)/hx;
        % On the right side
            Th(n_x+1,j)=2*((K1(n_x,j))^(-1)+ betax(n_x,j)*Vx(n_x,j))^(-1)/hx;
        end
    elseif (LeftRight ~= 1 && BottonTop == 1)
        for k=1:n_x
        % On the botton
            Tv(k,1)=2*((K2(k,1))^(-1)+ betay(k,1)*Vy(k,1))^(-1)/hy;
        % On the top
            Tv(k,n_y+1)=2*((K2(k,n_y))^(-1)+ betay(k,n_y)*Vy(k,n_y))^(-1)/hy;
        end
    end
end

% Defining the right hand side q

%%%%%%%%%%%%%%%%%%%%%    Dirichlet BC
if BoundCond == 1
    if (LeftRight == 1 && BottonTop==1)
        for j=1:n_y
            q(ind(1,j))=hy*Th(1,j)*bcl(j);
            q(ind(n_x,j))=hy*Th(n_x,j)*bcr(j);
        end
        for k=1:n_x
            q(ind(k,1))=hx*Tv(k,1)*bcb(k);
            q(ind(k,n_y))=hx*Tv(k,n_y)*bct(k);
        end
    elseif (LeftRight == 1 && BottonTop ~= 1)
        for j=1:n_y
            q(ind(1,j))=hy*Th(1,j)*bcl(j);
            q(ind(n_x,j))=hy*Th(n_x,j)*bcr(j);
        end
    elseif (LeftRight ~= 1 && BottonTop == 1)
        for k=1:n_x
            q(ind(k,1))=hx*Tv(k,1)*bcb(k);
            q(ind(k,n_y))=hx*Tv(k,n_y)*bct(k);
        end
    end
%%%%%%%%%%%%%%%%%%%%%%%%%%     Neuman BC
elseif BoundCond == 2
       q(ind(k1,j1))=hx*hy*1;
       q(ind(k2,j2))=hx*hy*(-1);
       q(n_x*n_y + 1)=hx*hy*(n_x*n_y);

%%%%%%%%%%%%%%%%%%%%%%%%%%     Periodic BC
elseif (BoundCond == 3 && LeftRight == 1)
    for j=1:n_y
        q((n_x*n_y+j))=1;
        %q((n_x*n_y+2))=bcl(j);
    end
    q((n_x*n_y+n_y+1))=n_x*n_y+n_y+1;
elseif (BoundCond == 3 && BottonTop == 1)
    for j=1:n_x
```

```
        q((n_x*n_y+j))=1;

        %q((n_x*n_y+2))=bcl(j);

    end

    q((n_x*n_y+n_x+1))=n_x*n_y+n_x+1;

end


%%%%%%%%%%%%%%%%%%%%%%%% Stiffness Matrix %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

 for k=1:n_x

     for j=1:n_y

         M(ind(k,j),ind(k,j))=(hx*Tv(k,j)+hy*Th(k+1,j)+hy*Th(k,j)+hx*Tv(k,j+1));

     end

 end


 for k=1:n_x-1

     for j=1:n_y

         M(ind(k,j),ind(k+1,j)) = -hy*Th(k+1,j);

         M(ind(k+1,j),ind(k,j)) = M(ind(k,j),ind(k+1,j));

     end

 end


 for k=1:n_x

     for j=1:n_y-1

         M(ind(k,j),ind(k,j+1)) = -hx*Tv(k,j+1);

         M(ind(k,j+1),ind(k,j)) = M(ind(k,j),ind(k,j+1));

     end

 end
```

```
if BoundCond == 3
    if (LeftRight==1 && BottonTop==1)
        for j=1:n_y
            M(ind(1,j),ind(n_x+1,j)) = -hy*Th(1,j);
            M(ind(n_x,j),ind(n_x+2,j))= -hy*Th(n_x+1,j);
            M(ind(n_x+1,j),ind(n_x+1,j))=1;
            M(ind(n_x+1,j),ind(n_x+2,j))=-1;
        end
        for j=1:n_y
            M(ind(1,j),ind(n_x+1,j)) = -hy*Th(1,j);
            M(ind(n_x,j),ind(n_x+2,j))= -hy*Th(n_x+1,j);
            M(ind(n_x+1,j),ind(n_x+1,j))=1;
            M(ind(n_x+1,j),ind(n_x+2,j))=-1;
        end
    elseif (LeftRight==1 && BottonTop ~= 1)
        for j=1:n_y
            M(ind(1,j),ind(n_x+1,j)) = -hy*Th(1,j);
            M(ind(n_x,j),ind(n_x+2,j))= -hy*Th(n_x+1,j);
            M(ind(n_x+1,j),ind(n_x+1,j))=1;
            M(ind(n_x+1,j),ind(n_x+2,j))=-1;
        end
    elseif (LeftRight~=1 && BottonTop == 1)
        for j=1:n_x
            M(ind2(j,1),ind2(j,n_y+1)) = -hx*Tv(j,1);
            M(ind2(j,n_y),ind2(j,n_y+2))= -hx*Tv(j,n_y+1);
            M(ind2(j,n_y+1),ind2(j,n_y+1))=1;
        M(ind2(j,n_y+1),ind2(j,n_y+2))=-1;
    end
  end
end
%%%%%%%%%%%%%%%%%%%%Solve the linear system%%%%%%%%%%%%%%%%%


p=M\(q);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%




%%%%%%%%%%%%%%%%%%%%%% Write the pressure in matrix form %%%%%%%%%%%%%%%
for k = 1:n_x
  for j=1:n_y
    P(k,j)=p(ind(k,j));
  end
end
%%%%%%%%%%%%%%%%%%%%%%%%-New Velocity-%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if BoundCond == 1
  if (LeftRight == 1 && BottonTop==1)
    for j=1:n_y
      VxNew(1,j)=-Th(1,j)*(P(1,j)-bcl(ind(1,j)));
      VxNew(n_x+1,j)=-Th(n_x+1,j)*(-P(n_x,j)+bcr(ind(n_x,j)));
    end
    for k=1:n_x
```

```
            VyNew(k,1)=-Tv(k,1)*(P(k,1)-bcb(ind(k,1)));

            VyNew(k,n_y+1)=-Tv(k,n_y+1)*(-P(k,n_y)+bct(ind(k,n_y)));

        end


    elseif (LeftRight == 1 && BottonTop ~= 1)

        for j=1:n_y

            VxNew(1,j)=-Th(1,j)*(P(1,j)-bcl(ind(1,j)));

            VxNew(n_x+1,j)=-Th(n_x+1,j)*(-P(n_x,j)+bcr(ind(n_x,j)));

        end


    elseif (LeftRight ~= 1 && BottonTop == 1)

        for k=1:n_x

            VyNew(k,1)=-Tv(k,1)*(P(k,1)-bcb(ind(k,1)));

            VyNew(k,n_y+1)=-Tv(k,n_y+1)*(-P(k,n_y)+bct(ind(k,n_y)));

        end


    end


end


if BoundCond == 3

     if (LeftRight == 1 && BottonTop ~= 1)

       for j=1:n_y

           VxNew(1,j)=-Th(1,j)*(-p(n_x*n_y+j)+p(j));

           VxNew(n_x+1,j)=-Th(n_x+1,j)*(-p(n_x*n_y-n_y+j)+p(n_x*n_y+j+n_y));

       end
     elseif (LeftRight ~= 1 && BottonTop == 1)

       for j=1:n_x

           VyNew(1,j)=-Tv(j,1)*(-p(n_x*n_y+j)+p(j));

           VyNew(j,n_y+1)=-Tv(j,n_y+1)*(-p(n_x*n_y-n_x+j)+p(n_x*n_y+j+n_x));

       end

     end

end


%Internal nodes


for k=1:n_x-1

    for j=1:n_y

        VxNew(k+1,j)=-Th(k+1,j)*(P(k+1,j)-P(k,j))/hx;

    end

end




for k=1:n_x

    for j=1:n_y-1

        VyNew(k,j+1)=-Tv(k,j+1)*(P(k,j+1)-P(k,j))/hy;

    end

end


%Error calculation

Err_x = norm((Vx-VxNew), inf)

Err_y = norm((Vy-VyNew), inf)
```

```
Err2_x(iter) = norm((Vx0-VxNew), inf)/norm(Vx0,inf);

Err2_y (iter)= norm((Vy0-VyNew), inf)/norm(Vy0,inf);


% *** Update the velocities ***


Vx=VxNew;

Vy=VyNew;


% Counting iterations

iter=iter+1;


end


%%%%%%%%%%%%%%%%%%% Post-Process %%%%%%%%%%%%


Vxuptotal=mean(Vx',1);

Vyuptotal=mean(Vy,1);

Vxup=1/2*(Vxuptotal(1)+Vxuptotal(n_x+1));

Vyup=1/2*(Vyuptotal(1)+Vyuptotal(n_y+1));


if BoundCond==1

    for j=1:n_y

        bcr2(j)=bcr(j);

        bcl2(j)=bcl(j);

    end
    Kupsc1=Vxup/(G1);


    for j=1:n_x
        bct2(j)=bct(j);
        bcb2(j)=bcb(j);
    end
    Kupsc2=Vyup/(G2);


elseif(BoundCond == 3 && LeftRight==1)
    pl=p(n_x*n_y+1:n_x*n_y+n_y);
    pr=p(n_x*n_y+n_y+1:n_x*n_y+2*n_y);
    Kupsc1=Vxup/(-mean(pl)+mean(pr));
    Kupsc2=Vyup/(-mean(pl)+mean(pr));
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
KUp1(ng)=Kupsc1;
KUp2(ng)=Kupsc2;
ng=ng+1;


for k=1:n_x
    for j=1:n_y
        VxAvrg(k,j)=(Vx(k+1,j)+Vx(k,j))/2;
        VyAvrg(k,j)=(Vy(k,j+1)+Vy(k,j))/2;
    end
end
```

```
%Figure

x=a:hx:b-hx;

y=c:hy:d-hy;

figure(1)

subplot(2,2,3)

[X,Y]=meshgrid(y,x);

surfc(X,Y,P)

title({'NonDarcy';['beta = ', num2str(beta)]})

az = 35;

el = 26;

view(az, el);

subplot(2,2,4)

contourf(X,Y,P)

figure(4)

%subplot(2,1,2)

quiver(X,Y,VxAvrg,VyAvrg)

title({'NonDarcy Velocities';['beta = ', num2str(beta)]})


if (n_x == n_y)

    for i=1:n_y

       PdiagonalF(i)=P(i,i);

    end

    figure(3)

    plot(y,PdiagonalF,'b*',y,PdiagonalD,'r-')

end


    end
    yy=1;

    scaley=factory;

    xx=scalex+1;

    scalex=scalex+ factorx;

end


% New index function

ind4=inline('(x-1)*z +y','x','y','z');

for k=1:nxnew

    for j=1:nynew

        Kupscalling11(k,j)=KUp1(ind4(k,j,nxnew));

        Kupscalling22(k,j)=KUp2(ind4(k,j,nynew));

    end

end

% Write to a data file

if BottonTop == 1

    dlmwrite('Kupscaling22.dat',Kupscalling22,'precision','%.15f')

else

    dlmwrite('Kupscaling11.dat',Kupscalling11,'precision','%.15f')

end
```

BIBLIOGRAPHY

1. K. Thomas A. Firoozabadi and B. Todd. High velocity gas flow in porous media. *SPE Reservoir Engineering Journal*, pages 149–152, May 1995.

2. Myron B. Allen, III and Eli L. Isaacson. *Numerical analysis for applied science*. Pure and Applied Mathematics (New York). John Wiley & Sons Inc., New York, 1998. , A Wiley-Interscience Publication.

3. A. Bourgeat. Homogenized behavior of two-phase flows in naturally fractured reservois with uniform fractures distributions. *Comput. Methods Appl. Mech. Eng.*, 47:205–216, 1984.

4. Dietrich Braess. *Finite Elements. Theory, fast solvers, and applications in solid mechanics*. Cambridge University Press, New York, second edition, 2001.

5. Zhangxin Chen. *Finite element methods and their applications*. Scientific Computation. Springer-Verlag, Berlin, 2005.

6. H. Darcy. *Les Fontaines Publiques de la Ville de Dijon*. Victor Dalmond, Paris, 1856.

7. Jim Douglas, Jr., Paulo Jorge Paes-Leme, and Tiziana Giorgi. Generalized Forchheimer flow in porous media. In *Boundary value problems for partial differential equations and applications*, volume 29 of *RMA Res. Notes Appl. Math.*, pages 99–111. Masson, Paris, 1993.

8. L. J. Durlofsky. Numerical calculation of equivalent grid block permeability tensors for heterogeneous porous media. *Water Resources Research*, 27(5):699–708, 1991.

9. Richard E. Ewing, Raytcho D. Lazarov, Steve L. Lyons, Dimitrios V. Papavassiliou, Joseph Pasciak, and Guan Qin. Numerical well model for non-darcy flow through isotropic porous media. *Comput. Geosci.*, 3(3-4):185–204, 1999.

10. A. Firoozabadi and D.L. Katz. An analysis of high velocity gas flow through porous media. *Journal Petroleum Technology*, pages 211–216, Feb. 1979.

11. P. Forchheimer. Wasserbewegung durch Boden. *Zeit. Ver. Deut. Ing.*, (45):1781–1788, 1901.

12. J. Geertsma. Discussion of the effects of non-darcy flow on the behavior of hydraulically fractured gas wells. *Journal of Petroleum Technology*, pages 211–216, October 1976.

13. C. T. Kelley. *Iterative methods for linear and nonlinear equations*. SIAM, Philadelphia, 1995.

14. P.R. King. The use of renormalization for calculating effective permeability. *Transport in Porous Media*, 4:37–58, 1989.

15. Peter Knabner and Lutz Angermann. *Numerical methods for elliptic and parabolic partial differential equations*, volume 44 of *Texts in Applied Mathematics*. Springer-Verlag, New York, 2003.

16. Kim M.-Y. and Park E.-J. Fully discrete mixed finite element approximations for non-darcy flows in porous media. *Computers and Mathematics with Applications*, 38:113–129, December 1999.

17. D. W. Peaceman. Interpretation of well-block pressure in numerical reservoir simulation. *Soc Petroleum Engrg. J.*, 253:183–194, 1978. Trans. AIME.

18. M. Peszyńska, E. Jenkins, and M. F. Wheeler. Boundary conditions for fully implicit two-phase flow model. In Xiaobing Feng and Tim P. Schulze, editors, *Recent Advances in Numerical Methods for Partial Differential Equations and Applications*, volume 306 of *Contemporary Mathematics Series*, pages 85–106. American Mathematical Society, 2002.

19. T. F. Russell and M. F. Wheeler. Finite element and finite difference methods for continuous flows in porous media. In R. E. Ewing, editor, *The Mathematics of Reservoir Simulation*, pages 35–106. SIAM, Philadelphia, 1983.

20. R. E. Showalter. *Hilbert space methods for partial differential equations*. Electronic Monographs in Differential Equations, San Marcos, TX, 1994. Electronic reprint of the 1977 original.

21. R. E. Showalter. *Monotone operators in Banach space and nonlinear partial differential equations*, volume 49 of *Mathematical Surveys and Monographs*. American Mathematical Society, Providence, RI, 1997.